

Improving short-term conflict alert via tabu search

J.E. Beasley¹
H. Howells²
J. Sonander²

February 2001

¹The Management School
Imperial College
London SW7 2AZ
England

²National Air Traffic Services
1 Kemble Street
London WC2B 4AP
England

ABSTRACT

In this paper we describe some work carried out by National Air Traffic Services in the UK into developing optimisation tools to help improve the effectiveness of one of their air traffic control safety systems – short-term conflict alert. In short-term conflict alert a computer system continually monitors radar data and alerts air traffic controllers if it detects a situation where two aircraft are in danger of approaching too close to one other.

Within the computer program that makes up the short-term conflict alert system are a large number of parameters. Choosing appropriate values for these parameters is a task that is currently done via extensive human intervention. In this paper we describe how a modern heuristic technique, tabu search, can be used to make parameter choices.

Keywords: air traffic control, short-term conflict alert, tabu search, air transport

1. INTRODUCTION

National Air Traffic Services (NATS) are the principal providers of air traffic control in the UK. They provide air traffic control at a number of the UK's airports, including the busiest, London Heathrow. An important part of the air traffic control system is short-term conflict alert (STCA). This is a computer system that, given radar data as an input, alerts human air traffic controllers if it detects that two aircraft are approaching too close to one another (have a potential conflict). As its name suggests STCA is designed to detect conflicts that may occur in the immediate future (typically two minutes).

As the reader will appreciate two aircraft approaching too close to one another is potentially dangerous. Not only is there the risk of direct collision, but also the aerodynamic stability of one (or both) aircraft could be disrupted by the other. In addition passengers/crew may be injured if extreme manoeuvres are needed in order to avoid a collision. Safety considerations therefore demand that pre-specified minimum separation distances (both horizontal and vertical separation) be maintained between aircraft in flight at all times. As such it is clearly important for air traffic controllers to be given as early a warning as possible of situations that involve a loss of separation so that they can redirect the aircraft in question. Hence the objective for any STCA system is: **to alert controllers to a developing conflict (loss of separation) with sufficient warning time to enable them to resolve the situation.**

As an illustration of the value of STCA the report covering the period July 1999 to December 1999 issued by the UK Airprox Board [1] (which investigates incidents where in the opinion of a pilot or controller safety has been compromised) details a number of incidents in which STCA was instrumental in alerting controllers to a conflict situation.

Inherent in any air traffic control STCA system however is the need for tradeoff, between *genuine alerts* and *nuisance alerts*. A genuine alert is one where either the aircraft are already involved in a conflict, or they are going to be in conflict in the near future - for example they are heading towards one another and are only one nautical mile apart. A nuisance alert is one where there is no danger of conflict in the near future – for example the aircraft are heading towards one another but are 100 nautical miles apart. Any STCA system must be “tuned” so as not to miss situations where genuine alerts are needed, but also not to issue too many nuisance alerts. Nuisance alerts are distracting to controllers and, if too many are issued, degrade their confidence in the system.

The simplified example above makes it seem easy to distinguish between genuine and nuisance alert cases. In fact, because:

- many different situations can occur with two aircraft flying independently at high speed through three-dimensional space; and
- what aircraft will do in the future is unknown, all that is known is current (and historic) information;

then it is not an easy task for an automated system to quickly identify genuine conflict situations.

NATS first introduced STCA for controllers in charge of en-route airspace at its London Area and Terminal Control Centre (LATCC) at West Drayton in 1988. Since that time the system has been progressively modified and evolved and now STCA systems monitor air traffic at all of the UK's principal air traffic control centres. As an indication of the amount of traffic dealt with by these centres there were nearly two million aircraft movements in the financial year 1999-2000. Although there are minor differences between the STCA systems installed in different

control centres their underlying basic design and operation is the same. In this paper therefore we will concentrate on STCA as implemented for the busiest centre, London Terminal Control at LATCC – hereinafter referred to as LATCC-TC. In the next section we describe this STCA system in more detail.

Note here however that although we refer in this paper to a single STCA system for LATCC-TC there are in fact two *separate* implementations of this system operating. One is an on-line system that continually processes radar data in an operational environment to check for conflicts. The other is an off-line system that is used for experimentation (such as described in this paper) into system improvement. In terms of the underlying specification the two systems are identical (i.e. they have identical functionality), but they are separated for safety reasons.

2. THE STCA SYSTEM

The NATS STCA system at LATCC-TC is a complex computer program that processes current radar tracking data every four seconds (a **cycle**) to decide, for all pairs of aircraft within radar range, whether any pair should be brought to the attention of a human air traffic controller. As mentioned above aircraft pairs should be brought to the controller's attention if they are already in conflict, or are likely to be in conflict in the near future. Conflict alerts are displayed on a controller's screen and are of two types - low severity alerts and high severity alerts. The severity of an alert is determined automatically based upon factors such as the current positions of the conflict pair and the predicted time at which the aircraft will be closest together.

The algorithms used to decide, for a given pair of aircraft (known as a **track pair**), whether an alert is necessary or not are very complex and proprietary to NATS. They make use of information such as aircraft position, speed, current heading and current manoeuvre (e.g. turning). This information for each track pair is first passed to a coarse filter. This is designed to reject from further processing any pairs that cannot possibly conflict in the short-term (e.g. two aircraft that are far apart). Track pairs not rejected by the coarse filter are further processed by three fine filters that are designed to test for different conflict conditions:

- a current proximity filter – which checks for conflict based upon current horizontal and vertical separation
- a linear prediction filter – which checks for conflict based upon predicted horizontal and vertical separation assuming the aircraft continue in a straight line along their current tracks
- a manoeuvre hazard filter – which checks for conflict when one or both of the aircraft are turning.

At the end of this process the system will have determined which track pairs (if any) are judged to be in conflict.

Typically however, unless the conflicting track pair have already violated minimum separation (or are predicted to violate minimum separation in the very near future), an alert is not immediately communicated to the controller. Instead an alert confirmation process is invoked. In this process the conflicting track pair are monitored with respect to a moving time window and an alert communicated to the controller only when the pair have been in conflict for a sufficient number of cycles (generally two or three cycles) within this moving time window. This alert confirmation process helps prevent nuisance alerts, since a pair in conflict in one cycle may have manoeuvred out of conflict by the next cycle.

In the NATS STCA system the airspace is divided into different regions, each of which is assigned a 'region type' (such as stack and en-route). The reason for this is that aircraft tend to behave differently in each type of region. For example, aircraft in a stack are typically following procedural holding patterns involving 180° turns at constant height, whilst en-route aircraft are typically heading in a straight line (either climbing, descending, or at constant height). Obviously therefore knowing which region an aircraft is in, and therefore its likely behaviour, assists conflict detection. LATCC-TC, for example, currently have 13 region types defined within their STCA system.

In the current NATS STCA system the coarse filter involves approximately ten different numeric parameters. The fine filters and the alert confirmation process however involve approximately eighty numeric parameters and these eighty parameters can (and often do) have different values in each region. This implies that

for LATCC-TC the STCA system requires approximately $10+13 \times 80 = 1050$ numeric parameter values.

In order to evaluate the effect of different parameter values NATS has at its disposal a number of databases incorporating historic radar tracks, including cases where there was a serious encounter between aircraft. Within these databases each track pair has been individually categorised into one of five categories, as in Table 1.

Category	Definition	Meaning
1	Alert necessary	The situation involved a serious loss of separation or avoided such a loss of separation only by means of a late manoeuvre.
2	Alert desirable	Although there was no serious loss of separation the situation was such that an alert would have been useful in drawing the attention of the controller to a potential conflict.
3	Level off with risk (alert desirable)	Separation would have been lost if one or both aircraft had not performed a level off manoeuvre.
4	Alert undesirable or unnecessary	The situation presented little threat of loss of separation and an alert was more likely to have proved distracting than helpful.
5	Bad track data (alert undesirable or unnecessary)	No conflict existed, however invalid track data such as split tracks (the same aircraft appearing as two or more tracks in a multi-radar environment) or signal corruption of the altitude reported by the aircraft transponder led to the generation of an alert. This category is also used to cover military aircraft flying in formation – which would otherwise generate many alerts.

Table 1: Track categorisation

When deciding STCA parameter values therefore the objective is essentially to generate alerts for all category 1 track pairs and a high percentage of category 2 and 3 track pairs, whilst keeping to an acceptable level alerts for category 4 track pairs. Category 5 track pairs are usually ignored unless that part of STCA which deals with the conflict processing of split tracks is being considered. As well as generating an alert however, the issue of warning time for genuine conflicts must also be considered. Two parameter sets may generate exactly the same alerts, but one set may give more warning time in terms of the time between the alert and the predicted time at which the aircraft will be closest together.

To give some idea of the size of the problem a typical track database for LATCC-TC might contain 250,000 track pairs of which 250 would be category 1, 200

category 2 and 2000 category 3 – the remainder being category 4 or 5. Processing a track database of this size to determine the effectiveness of any set of parameters requires approximately 6 minutes on a fast powerful computer, a DEC Alpha 4100 (533 MHz, 2 processors).

Over the years NATS R&D staff have, via extensive human intervention, experimented and arrived at a set of numeric parameter values for their STCA system (1050 values for LATCC-TC) that seem appropriate – effectively tuning the system so as to achieve a balance between genuine and nuisance alerts, whilst also considering warning time. Essentially this has been done by a person sitting down and making a change to one or more parameter values, running these new values through the STCA computer program using a historic track database, and then inspecting the computer output provided to see whether these new values have improved upon the previous values or not. Prior to the work described in this paper NATS had no algorithmic approach to the systematic generation and evaluation of many different parameter sets.

In this paper we describe how a modern heuristic technique, tabu search, can be used to make parameter choices and hence increase the effectiveness of tuning the STCA system. Our tabu search algorithm is described in the next section.

3. ALGORITHM

In this section we shall first discuss our general algorithmic approach, then the objective (scoring) function we have adopted and finally present the tabu search algorithm we have developed.

3.1 General approach

In our approach, for any particular run of our algorithm, the parameters associated with the NATS STCA model are divided into two types:

- those which are fixed
- those which can be varied.

Fixed parameters are those which take a pre-set fixed value during the current run of the algorithm, whilst variable parameters are those for which values should be decided by the algorithm. To deal with the fact that many of the variable parameters may be region dependent we distinguish two possibilities:

- a parameter takes exactly the same value in all regions
- a parameter takes different values in each region.

Each (varying) parameter is allowed to vary between a *lower limit* and an *upper limit* from its *initial value* in terms of a *step value*. For example if the lower limit for a parameter is 180, the upper limit 210, the initial value 200 and the step value 10, then possible values for the parameter are 200, $200+10=210$, $200-10=190$ and $200-2\times 10=180$. Each parameter also has associated with it an *importance rating* (the higher the importance rating the more important the parameter is believed to be). This is used within the algorithm to derive an ordering for parameter variation, parameters which are more important are varied before parameters which are less important.

The choice of parameters that are fixed/variable must be made by NATS personnel. Only a person with knowledge and experience of STCA can judge:

- which parameters should be regarded as fixed
- which parameters should be regarded as variable
- the range of possible parameter values that are legitimate
- the importance of each parameter.

In addition, since our primary objective is to improve upon the current situation, we typically use the best parameter values known to NATS as the initial values for any run of the algorithm.

The reader may be wondering why, if choices still have to be made by NATS personnel (as detailed above), the work presented here improves upon what NATS currently does. The answer is two-fold:

- The current NATS system requires regular human intervention to produce new parameter sets to be evaluated in order to move towards an 'optimum' set of parameters. As such it is very labour intensive and, as a consequence, only a limited number of parameter sets are ever evaluated.
- In our tabu search algorithm many sets of parameters can be **systematically** generated and evaluated with the algorithm automatically guiding the search towards an 'optimum' set of parameters. As such many more parameters sets can be evaluated without the need for human intervention.

In summary then our system both lessens the workload on NATS personnel and enables many more parameter sets to be systematically investigated. We would also mention here that one of the motivations for NATS in undertaking the work reported in this paper was the belief that a computerised approach to investigating many different parameter sets in a systematic fashion would enable them to explore regions of parameter space that had hitherto been unexplored.

Our algorithmic design is governed by the computation time (6 minutes, as mentioned above) needed to evaluate a single set of parameters. Reflect that there are 1050 numeric parameter values in the STCA system for LATCC-TC. Were all of these to be varied (singly) just once only we would need $6 \times 1050 / 60 = 105$ computational hours (approximately 4½ days). As the computer system on which this work is done also has other tasks to perform the elapsed time required could become

very high. For this reason the tabu search algorithm presented below (e.g. in its use of priority ordering) has been designed to make improvements in parameter choice as quickly as possible.

3.2 Objective (scoring) function

In order to evaluate a given set of parameters we compute a single numeric score associated with “how good” that set is. In algorithmic terms we regard this as an objective function which is to be maximised. The current objective function (scoring function):

- uses the given initial set of input parameters as a **base case** against which new parameter sets are evaluated
- evaluates each new parameter set against this base case in terms of:
 - total number of alerts gained or lost
 - amount of warning time gained or lost (measured in terms of cycles).

In more detail the contribution to the overall objective function made by alerts and by warning time (cycles) is given in Table 2.

Track category	Alerts		Warning time (cycles)	
	Gained	Lost	Gained	Lost
1	$500 \times \text{number gained}$	$-500 \times (\text{number lost})^2$	$10 \times \text{number gained}$	$-8 \times (\text{number lost})^{1.2}$
2	$50 \times \text{number gained}$	$-50 \times \text{number lost}$	$1 \times \text{number gained}$	$-1 \times \text{number lost}$
3	$20 \times \text{number gained}$	$-20 \times \text{number lost}$	$0.5 \times \text{number gained}$	$-0.5 \times \text{number lost}$
4	$-10 \times (\text{number gained})^{1.2}$	$10 \times \text{number lost}$	0	0
5	$-2.5 \times \text{number gained}$	$2.5 \times \text{number lost}$	0	0

Table 2: Example scoring function

To illustrate the scoring (objective) function suppose that a particular parameter set, compared to the base case, has:

- 1 more category 2 alert, 5 less category 3 alerts, 7 less category 4 alerts; and
- 2 less cycles warning for category 1 tracks, 2 more cycles warning and 4 less cycles warning for category 3 tracks;

then its score is $(50 \times 1) + (-20 \times 5) + (10 \times 7) + (-8 \times (2)^{1.2}) + 0.5 \times 2 + (-0.5 \times 4) = 0.621$,

indicating that this parameter set is better than the base case (which by definition has

a score of zero).

Choosing the values used in Table 2 inevitably implies the existence of a tradeoff between factors. For example in terms of alerts, the value of -50 for category 2 alerts lost and the value of 10 for category 4 alerts lost mean that we would be prepared to lose a category 2 alert (which is desired, see Table 1) if it meant five ($=50/10$) less category 4 (nuisance) alerts.

Table 2 reflects the reality of STCA, typically improvements in one positive characteristic (e.g. a reduction in category 4 nuisance alerts) must be balanced by negative characteristics (e.g. the loss of a desired category 2 alert). Note here however that the values given in Table 2 are only example values. Attitudes to alerts and warning time (i.e. desired tradeoffs between positive and negative characteristics) vary both between different air traffic control centres and over time.

It is possible however to use our algorithm in an “all improve” mode whereby only solutions that improve all characteristics are sought. The easiest way to achieve this is by adjusting the scoring (objective) function so that each negative characteristic has a large negative score (e.g. $-M$ say) and each positive characteristic retains its original score, as shown in Table 3.

Track category	Alerts		Warning time (cycles)	
	Gained	Lost	Gained	Lost
1	$500 \times \text{number gained}$	$-M$	$10 \times \text{number gained}$	$-M$
2	$50 \times \text{number gained}$	$-M$	$1 \times \text{number gained}$	$-M$
3	$20 \times \text{number gained}$	$-M$	$0.5 \times \text{number gained}$	$-M$
4	$-M$	$10 \times \text{number lost}$	0	0
5	$-M$	$2.5 \times \text{number lost}$	0	0

Table 3: Example scoring function for “all improve” mode

With this scoring function any parameter set with a score greater than zero will dominate the base case, having no increase in negative characteristics but having gained positive characteristics.

3.3 Tabu search algorithm

We now consider how we **systematically** generate different parameter sets in an attempt to produce parameter sets that give better STCA performance. The algorithm given below is based on a technique called **tabu search** which is a local search heuristic due to Glover [2] and Hansen [3]. In tabu search the fundamental concept is that of a "move", a systematic operator that, given a single starting solution, generates a number of other possible solutions. In local search terms these other solutions are the "neighbourhood" of the single starting solution.

From the neighbourhood the "best" solution is chosen to become the new starting solution for the next iteration and the process repeats. This best solution may either be the first improving solution encountered as the move operator enumerates the neighbourhood, or it may be based upon complete enumeration of the neighbourhood.

In order to prevent cycling a list of "tabu moves" is employed. Typically this list prohibits certain moves which would lead to the revisiting of a previously encountered solution. This list of tabu moves is updated as the algorithm proceeds so that a move just added to the tabu list is removed from the tabu list after a certain number of iterations (the "tabu tenure") have passed. A more comprehensive overview of tabu search can be found in [4,5,6].

In our tabu search algorithm for STCA a move is defined as a change in a parameter value and corresponds to:

- increasing a parameter, by adding its step value to its current value; or
- decreasing a parameter, by subtracting its step value from its current value.

For simplicity we shall refer to these moves as the **up move** and the **down move** respectively.

In our tabu search algorithm we disallow (make tabu) **reverse moves**. For

example suppose we choose (because it gives a better overall value for the objective function) to increase a parameter by its step value of 10 from its current value of 200 to 210 (i.e. to make the up move). Then the reverse move, reducing the parameter by 10 (e.g. from 210 to 200), the down move, is declared tabu (i.e. it cannot be made) until after a given number of other moves have been examined. Colloquially we can say that after making a move we are not allowed to reverse the situation straightaway.

Before giving our tabu search algorithm in detail we need define some notation. Let:

- K be the number of parameters that can be varied
- $L(k)$ $k=1,2,\dots,K$ be a list of these parameters in *descending* importance order
- $VALUE(k)$ be the current value associated with parameter k
- $T(k,m)$ be the number of examined moves for which move m for parameter k is tabu, where $T(k,m)=0$ if the move is not tabu and we use $m=1$ for the up move and $m=2$ for the down move
- $BEST$ be the objective function value for the best set of parameter values found over all iterations
- $IBEST$ be the objective function value for the best set of parameter values found in the current iteration
- $IBEST_VALUE(k)$ be the parameter value for variable parameter k in the $IBEST$ solution
- $IMPROVED=1$ if an improved solution has been found, else $IMPROVED=0$
- $TABU_TENURE$ be how long a move is tabu for (e.g. $TABU_TENURE=20$ would tabu moves until 20 other moves have been examined)

Then our tabu search algorithm is as follows:

1. Initialise all parameter values $VALUE(k)$ $k=1,\dots,K$ to the base case initial values as set by NATS personnel and evaluate the solution. Set $BEST$ = the base case objective function value.
2. Set $T(k,m)=0$ $k=1,\dots,K$; $m=1,2$ (i.e. set all up and down moves as not tabu).
3. Set $p=1$ (p represents which of the K variable parameters is currently being examined). Set $IMPROVED=0$ and $IBEST=-\infty$.
4. For $m = 1$ to 2 do:
 5. Set $M=2$ and $S=+1$ if $m=1$ or $M=1$ and $S=-1$ if $m=2$ (M is the reverse move to m and S is $+1$ for an up move, -1 for a down move).
 6. Examine, if possible, move m for the parameter associated with $L(p)$. It will

not be possible to examine this move if the move is tabu (i.e. $T(p,m) > 0$) or if the current value $VALUE(p)$ of the parameter is already at its pre-set limit (upper limit if $m=1$, lower limit if $m=2$).

If it is possible to examine the move then:

7. Set $VALUE(p) = VALUE(p) + S \times [\text{step value for } L(p)]$ and let V be the resulting objective function value associated with the parameter set after the move has been made.
8. Set $T(k,j) = \max[0, T(k,j) - 1]$ $k=1, \dots, K$; $j=1, 2$ (reduce all tabu values by 1).
9. If $V > IBEST$ then set $q=p$, $r=M$, $IBEST=V$ and $IBEST_VALUE(k) = VALUE(k)$ $k=1, \dots, K$.
10. If $V > BEST$ (i.e. we have found a better solution) then:
 - set $IMPROVED=1$, $BEST=V$ and $T(p,M)=TABU_TENURE$ to tabu the reverse move; go to step 6 to examine the same move again. This is an aggressive strategy in that if a move is successful at finding an improved solution we immediately examine the same move for the same parameter again to see if we can get further improvement.
 - else
 - set $VALUE(p) = VALUE(p) - S \times [\text{step value for } L(p)]$ to reset the current value for parameter p
 - endif
- endif
- enddo
11. Set $p=p+1$ and if $p \leq K$ go to step 4 to examine moves for this new parameter.
12. If $IMPROVED=1$ then we have improved the BEST solution at some point in the last examination (steps 3 to 11) of up/down moves for the K variable parameters. Go to step 3 to see if further improvements can be made.
13. If $IMPROVED=0$ then we have failed to improve the BEST solution in the last examination (steps 3 to 11) of up/down moves. To proceed from this point we start another iteration but with the solution from which we examine moves being the IBEST solution. Set $VALUE(k) = IBEST_VALUE(k)$ $k=1, \dots, K$; $T(q,r) = TABU_TENURE$ (tabu the reverse of the move that led to IBEST); and go to step 3.

With regard to implementing the above algorithm we typically stop after a pre-specified computation time or if all moves are tabu ($T(k,m) > 0 \forall k,m$). In terms of

computation time one iteration (steps 3 to 11) requires the evaluation of $O(2K)$ different parameter sets.

Probably the most difficult aspect of the above algorithm to grasp is the role of IBEST. At step 12 above if BEST has been improved then the next iteration at step 3 starts with $VALUE(k)$ $k=1,\dots,K$ being the parameter values associated with this BEST solution. Hence in this case the value of IBEST is irrelevant. IBEST only becomes relevant if we do not improve BEST in an iteration. In such a case we have to start the next iteration with BEST (which has been unchanged since the last iteration) we would be repeating ourselves. Therefore we start the next iteration with IBEST (step 13).

To illustrate this suppose that we have an example with two parameters A and B currently set in the BEST solution to values $[a,b]$ with this solution having an objective function evaluation of value 10. Assuming (for simplicity) that the step value for both parameters is 1, and no moves are tabu, then an iteration through these parameters will involve examining four options: $[a+1,b]$, $[a-1,b]$, $[a,b+1]$ and $[a,b-1]$. Suppose the objective function evaluation of these solutions is 8, -2, 5 and 9 respectively. None of these solutions improves upon BEST and so the next iteration will start from the IBEST solution, which in this case is $[a,b-1]$ of value 9. This solution was produced by a down move for B so the up move for B will now be tabu. Hence the next iteration will involve only three options: $[a+1,b-1]$, $[a-1,b-1]$ and $[a,b-2]$. Note that none of these options were examined in the previous iteration.

4. RESULTS

The tabu search algorithm described in this paper has been used by NATS to test the combined effectiveness of new STCA logic proposed to improve alerting performance. By logic here we mean the rules that are embedded in the STCA computer system for conflict alert. In this section we describe this application of the algorithm.

New STCA logic (either changes to existing rules or the addition of new rules) is generally proposed following analysis of current alerting performance for a serious encounter incident (such as an incident that would be classified as category 1 in Table 1). As such new logic tends to be proposed in isolation, i.e. with reference to a particular encounter. Before any proposed logic can be implemented it is necessary to test the effect it has on overall system performance. This involves optimising any new parameters associated with the new logic, and also re-optimising some existing parameters to take into account the addition of the new logic. This task is made more difficult when more than one set of new logic has been proposed, as optimisation is then required for a range of new parameter values and a number of existing parameter values may need modification.

Previously, this process involved optimising each new set of logic individually and assessing the benefit provided to alerting performance. The combined effect was then tested by incrementally activating each new set of logic (starting with the set that provided the most benefit when tested individually) and measuring the cumulative benefit. Due to constraints upon the time/effort that could be devoted to this process it was generally only possible to undertake a limited re-optimisation of parameters at this stage.

This approach aimed to identify which subset of proposed logic enhancements provided the most benefit, however it was recognised that adding new logic with parameter sets optimised in isolation does not necessarily give a true indication of the potential benefit of a subset of new logic.

The tabu search algorithm described in this paper has been used to test, and optimise, the combined effectiveness of a set of proposed logic. This showed that the tabu search algorithm was effective as it allowed thorough testing of the new logic. Moreover analysis of the BEST solutions generated from a number of runs highlighted combinations of new logic that offered some potential benefit.

In certain situations the introduction of new logic into the STCA system can change the functionality required from existing logic, hence some quite radical changes in existing parameters may be required to improve performance. When this was the case it was found that the tabu search algorithm could have some difficulties in finding new BEST solutions as the addition of the new logic would initially lead to a worsening of performance – thus the search procedure could become trapped in a region of the parameter search space which did not utilise the new logic. This was overcome by manual analysis to find an appropriate starting point for the optimisation.

One problem, related to the objective function adopted as opposed to the tabu search algorithm itself, is with regard to the assessment of additional warning time for encounters. The value of additional warning time depends both on the encounter characteristics and on the characteristics of the alert before and after a parameter change. The objective function (Table 2) currently considers the former in terms of severity category only, and does not consider the latter at all. As a result the objective function would erroneously value a gain of seven cycles warning time for a category 1

encounter that already had sufficient warning time, above a gain of a single cycle of warning time for six category 1 encounters which had short warning times.

Overcoming this problem is only possible by adaptation of the objective function to consider each encounter individually and in detail. For example this could be done by defining a warning time value $W(j)$ for each category 1/2/3 track pair j in the track database such that only warning times $\leq W(j)$ for track pair j contribute to the objective function. Since a typical track database for LATCC-TC might contain 250,000 track pairs of which 250 would be category 1, 200 category 2 and 2000 category 3 this would involve individually setting values for $W(j)$ for $j=250+200+2000=2450$ track pairs. This would be a time-consuming task.

One other problem encountered was overfitting. Validation of the final BEST solution parameter set on an independent track database showed less benefit than demonstrated on the track database used for parameter determination. Whilst it may be possible to reduce this problem by increasing the size of the track database used for parameter determination, in practice this is difficult due to the time required to produce a track database containing sufficient detail for this type of optimisation. In any event it is clear that, however we choose to generate parameter sets for examination, whether by tabu search or some other algorithm or (as previously) by human insight, overfitting in the sense of the parameters that best fit the track database used for parameter determination performing less well when validated against an independent track database will often exist.

The above problems mean that some manual analysis by NATS personnel is still required, both before and after an optimisation. However despite these problems our tabu search algorithm has clearly demonstrated the potential to reduce the requirement for human intervention, thus enhancing NATS's ability to discover

parameter sets which improve STCA performance, whilst also improving the efficiency by which this may be achieved.

In organisational terms the algorithm described in this paper has highlighted two issues:

1. NATS STCA optimisation tools require some modifications to enable more effective comparisons to be made between different parameter sets; and
2. the time required to process a track database for a single set of parameter values means that some runs of the algorithm are resulting in elapsed times of a week or more.

This first issue is one of inherited software. NATS has a number of tools for generating text files that give a detailed analysis of precisely where a new parameter set differs from a base case, as well as summary statistics. However wading through text files to try and find desired figures, whilst feasible, is time-consuming for the personnel concerned.

This second issue relates to the size of the track database, typically around 250,000 track pairs, and the inherent nature of the process. All of these track pairs have to be individually processed to see if a conflict is occurring. As mentioned previously this can take around 6 minutes. As $O(2K)$ parameter sets need to be evaluated for one iteration though K variable parameters then this corresponds to a computation time of $O(12K)$ minutes. Moreover a significant number of such iterations may be desired, since we cannot guarantee to improve BEST at each and every iteration. The combined effect of this is elapsed times running to one week or more, hardly ideal in the 21st century.

It is clear that computationally the bottleneck is the 6 minutes required by the STCA system to process the track database. This equates to processing $250000/(6 \times 60)$

= approximately 700 track pairs per second. There are two possible options for reducing the time required by the STCA system:

1. reduce the size of the track database; and/or
2. process more track pairs per second.

NATS has experimented with the first option. However experience has been that unless the track database used in the tabu search algorithm for parameter determination is relatively large the parameters found, when validated against an independent track database, often do not improve upon the base case parameters applied to the same independent track database.

With respect to the second option the rate of processing of track pairs is governed by the underlying speed of the computer involved, and the complexity of the equations that must be considered to determine if a conflict is occurring. Currently NATS already uses a powerful computer, and the nature of the equations is fixed, so the only realistic option for significantly increasing the number of track pairs processed per second is to move to parallel (or distributed) processing. As each track pair is considered independently this is conceptually easily done. Whilst implementing the current NATS STCA computer program in a parallel/distributed computing environment was far beyond the scope of the study reported in this paper it is both a feasible and logical course of action for the future.

5. CONCLUSIONS

In this paper we have described a tabu search algorithm for improving the parameters associated with the short-term conflict alert system used by NATS. Prior to the work described here parameter values were determined via extensive human intervention. Experience with our tabu search approach indicates the potential for significantly less human intervention and the examination of a greater number of parameter sets.

REFERENCES

1. UK Airprox Board, Analysis of airprox in UK airspace – report number 3, (2000). Available from the UK Airprox Board, Hillingdon House, Uxbridge, Middlesex, UB10 0RU.
2. F. Glover, Future paths for integer programming and links to artificial intelligence. *Computers & Operations Research* 13 (1986) 533-549.
3. P. Hansen, The steepest ascent mildest descent heuristic for combinatorial programming. Presented at the Congress on Numerical Methods in Combinatorial Optimization, Capri, Italy (1986).
4. C.R. Reeves (editor), Modern heuristic techniques for combinatorial problems. Blackwell Scientific Publications, Oxford, (1993).
5. E.H.L. Aarts and J.K. Lenstra (editors), Local search in combinatorial optimization. Wiley, (1997).
6. F.W. Glover and M. Laguna, Tabu search. Kluwer Academic Publishers, (1997).