SAFETY AND SENSITIVITY ANALYSIS OF THE ADVANCED AIRSPACE CONCEPT FOR NEXTGEN

John Shortle, Lance Sherry, George Mason University, Fairfax, VA
Arash Yousefi, Richard Xie, Metron Aviation, Fairfax, VA

Abstract

This paper presents a model and methodology for a safety and sensitivity analysis of the Advanced Airspace Concept. This analysis is part of a larger effort to analyze safety-capacity tradeoffs in NextGen concepts. A key part of the model is the definition of a dynamic event tree, which is like a standard event tree, but also includes the dimension of time in the state-space description. The model is constructed and evaluated in an automated fashion based on a set of input tables. Thus, changes to the model are easily implemented and results are automatically re-computed. The analytical implementation can be evaluated fairly quickly (a couple seconds per evaluation). A systematic sensitivity analysis shows that the transponder failure probability is a critical model parameter.

Introduction

The objective of this paper is to conduct a high-level safety analysis of the collision risk for the Advanced Airspace Concept (AAC), proposed by Erzberger et al. (e.g., [1] – [5]). This research is part of a larger effort to evaluate the trade-off between safety and capacity of proposed Next Generation Air Transportation (NextGen) concepts (e.g., [6], [7]). Establishing such relationships can be helpful in defining system requirements and refining the concepts.

In AAC, separation assurance is provided by a ground-based system that automatically detects and resolves conflicts and transmits resolutions to aircraft via a datalink. Because the separation-assurance function is provided by automation, this concept has the potential to reduce the human controller workload and thereby increase the overall airspace capacity.

Advanced Airspace Concept

In the concept described by Erzberger, the ground-based system consists of two main automated elements: (1) the AutoResolver (AR), which provides strategic conflict detection and resolution, approximately 20 to 3 minutes prior to a potential conflict, and (2) the Tactical Separation Assisted Flight Environment (TSAFE), which provides tactical conflict detection and resolution, approximately 3 to 1 minute prior to a potential conflict. The Tactical Collision Avoidance System (TCAS) and pilot visual avoidance provide the final line of defense against collisions, similar to today's practice.

As a centralized system, AAC collects state vectors (positions and speeds) and flight-plan information of all aircraft within a particular region of airspace. The ground automation detects conflicts, generates resolutions, and transmits the resolutions to the aircraft via a datalink. For equipped aircraft, the flight crew is responsible for accepting and executing the uplinked trajectories within defined tolerances (a similar role as today).

The automation is assisted by a human controller who is responsible for overall supervision of the system and serves as a backup to the automation. The controller is responsible for monitoring flows, issuing clearances to non-equipped aircraft, handling situations that cannot be handled by the automation, and serving as a backup in case of failure of the automation. For further details on this concept, see for example [1] – [5].

Relation to Previous Work

The AAC model in this paper is similar to and inspired by analyses given in [8], [9], and [10]. A higher level analysis (not at the sub-component level) was given in [11]. The main contributions of this paper are: (1) to provide a *generic* framework and an *automated* computing environment for defining the associated event trees and fault trees, (2) to compute NMAC probabilities *analytically* without Monte-Carlo simulation, and (3) to perform a comprehensive *sensitivity analysis* on the underlying model parameters. A sensitivity analysis is useful to identify critical components and to determine which redundancies are effective.

References [8] and [9] use Monte-Carlo simulation to evaluate a safety model of the AAC. One challenge with Monte-Carlo simulation is that many replications are necessary to estimate very small probabilities. For example, 10^9 replications are required to yield one observation, on average, of an event that occurs with probability 10^{-9}. More replications are required if a tight confidence interval around the probability estimate is needed. Thus, a large amount of computing time may be needed. This is particularly problematic for sensitivity analyses in which the rare-event probability must be estimated multiple times for a large number of parameter combinations. One approach to handle this issue is to use a rare-event acceleration technique. For example, [9] uses importance sampling to accelerate the simulation performance of the same model.

An alternate approach is to create a model that can be evaluated analytically. For example, [10] developed a probabilistic state-based model of the same system, very similar to the model given in [8]. The model in [10] is defined via sets of equations that specify transition probabilities between states, rather than by a simulation algorithm that is implemented as computer code. Because the equations can be evaluated analytically, the time to evaluate the model is essentially instantaneous.

Our work is similar to, though developed independently from, the implementation given in [10]. The main difference is that we provide a generic and automated framework for building the model structure and calculating the results. This allows for greater flexibility in terms of making changes to the model. In contrast, [10] derives problem-specific equations, so changes to the model require a re-derivation of the analytical equations. In our implementation, the underlying equations are re-derived automatically, which perhaps reduces the potential for analyst error. However, this added flexibility comes at the cost of additional computer time. Our model takes several seconds to evaluate (it is not instantaneous).

In addition, we provide several modeling extensions not given in the previous references, including a pilot-response model and controller-override model (controller action in the event that a resolution is not given by the automation).

Analysis Approach

The approach in this paper is to evaluate collision probabilities through a combination of fault trees (or reliability diagrams) and event trees. Figure 1 shows a high-level description of the methodology.

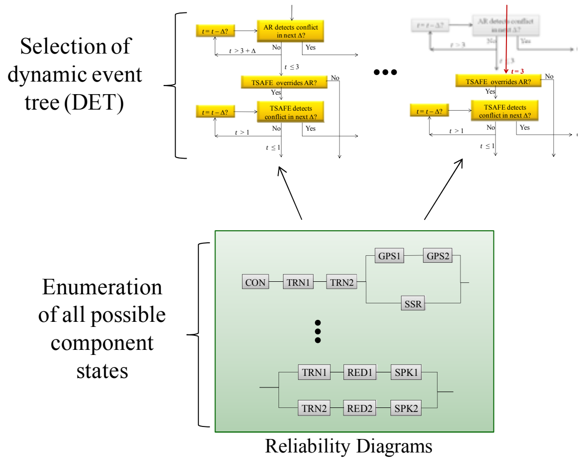


Figure 1. Overall Methodology

At the highest level, the system is modeled via a set of event trees. The event trees define a probabilistic sequence of events that could potentially occur when two aircraft are on course for a near-mid-air collision (NMAC). We extend the standard event-tree methodology by including the dimension of time in the state space of the event tree. This allows for the possibility of modeling not just the occurrence or non-occurrence of an event (for example, whether or not a conflict is successfully detected), but also the time at which the event occurs (for example, an earlier conflict detection has a different downstream consequence compared with a later conflict detection).

At a lower level, the system is modeled by a set of fault trees or reliability diagrams that specify relationships between physical components of the system (for example, the onboard transponder) and various system functions (for example, the ability of the ground host to transmit resolutions to the aircraft). The final output is the estimated probability of an NMAC or a collision.

The methodology is summarized by the following three steps, which will be discussed in more detail:

- Evaluate fault trees to determine the joint probability that various system functions are working or failed.
- Evaluate event trees to determine the conditional probability that a collision occurs between an aircraft pair, given that two aircraft are on course for a NMAC. Several different event trees are specified in the model. A "baseline" event tree corresponds to the nominal state in which all functions are working. Other event trees correspond to variants of the baseline tree in which different system functions have failed (for example, when the Auto-Resolver is unable to transmit resolutions to either aircraft).
- Synthesize results to determine the overall condition collision probability. The final result is a weighted sum of the event-tree probabilities weighted by associated probabilities computed via the fault trees.

Evaluation of Fault Trees

The objective of this step is to compute the joint probability of various functional failures, based on the failure probabilities of underlying physical components. We consider four different system functions (based on the model setup in [8]):

1. Locatability of both aircraft in a pair,
2. Ability of the Auto-Resolver to communicate a resolution to at least one aircraft,
3. Ability of TSAFE to communicate a resolution to at least one aircraft,
4. Functionality of TCAS components other than the transponder.

The relationships between component failures and functional failures are specified by reliability diagrams (Figure 2). Table 1 defines the notation used for elements in these diagrams. These reliability diagrams are based on fault trees given in [8]. Each function on the right is assumed to be working if there is a path from left to right through working components. For example, in the first diagram, a failure of the ground-to-host connection (CON) results in a broken path from left to right, indicating a failure of the locatability function.

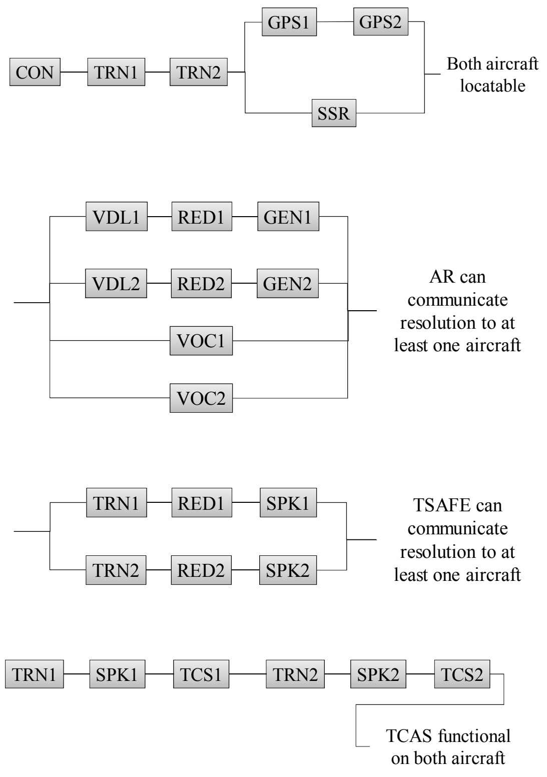

CON
TRN1
TRN2
GPS1
GPS2
SSR
Both aircraft locatable
VDL1
RED1
GEN1
VDL2
RED2
GEN2
VOC1
VOC2
AR can communicate resolution to at least one aircraft
TRN1
RED1
SPK1
TRN2
RED2
SPK2
TSAFE can communicate resolution to at least one aircraft
TRN1
SPK1
TCS1
TRN2
SPK2
TCS2
TCAS functional on both aircraft

Figure 2. Reliability Diagrams

Table 1. Components in Reliability Diagrams

Code	**Component / Function**
CON	Connection from ground station to host
SSR	Secondary radar signals aircraft to send data
TRN*i*	Transponder on aircraft *i*
GPS*i*	ADS-B on aircraft *i* gets GPS location
VDL*i*	VDL uplink to aircraft *i*
RED*i*	Resolution reader on aircraft *i*
GEN*i*	Resolution trajectory generator on aircraft *i*
VOC*i*	Voice communication to aircraft *i*
SPK*i*	Speaker on aircraft *i*
TCS*i*	TCAS elements on aircraft *i* (other than speaker and transponder)

Table 2 shows the failure probabilities and other parameters used in the model. Unless other references are given, the values come from [8].

Table 2. Model Parameters

Description	Baseline Probability
GPS / ADS-B fails	0.0005
Transponder fails	0.000097
Ground fails to signal AC to send data	0.00682
Ground station to host connection fails	0.00002
VDL2 fails	0.00004
Voice communication fails	0.00055
On-board resolution reader fails	.000001
On-board resolution trajectory generator fails	0.000097
On-board speaker fails	.000001
TCAS components other than speaker / transponder fail	.000292 [assumed, see also 12]
AR fails to generate 4-D flight-plan trajectory	.000001
TSAFE fails to generate dead-reckoning trajectory	.000001
TSAFE fails to override AR	.000001
TCAS fails to override TSAFE	.000001
Both pilots fail to respond to TCAS	0.3
NMAC results in collision (by chance)	0.0000672 [7]
AR fails to generate resolution	.000001
Backup controller fails to provide resolution in .5 min	0.30 [assumed]
Pilot fails to accept resolution in .5 min	0.1 [assumed]
AR conflict detection probabilities as a function of time until NMAC	Varies with time; see [8]
TSAFE conflict detection probabilities as a function of time until NMAC	Varies with time; see [8]

We assume that failed components fail at the beginning of the 8-minute time horizon of the model. This is a conservative assumption. A component that otherwise would have failed in the middle of the 8-minute interval is assumed to have failed at the beginning of the interval.

A key issue in evaluating functional failures is that some physical components contribute to multiple system functions. For example, the functions associated with aircraft locatability, TSAFE, and TCAS all rely on the aircraft transponder (Figure 2). This means that failures of these functions are not independent. The failure of one function indicates the potential failure of a transponder which in turn indicates an increased failure probability of the other functions. Thus, the joint failure probability cannot be calculated as the product of the marginal failure probabilities.

To calculate the joint failure probability of the system functions, we apply a straightforward brute-force enumeration of all component states, rolling up the results into the overall joint failure probability of the higher-level functions.

To define notation, let N be the number of components in the system. In our model, $N = 18$ (see Table 1; some components count twice – once for each aircraft in a pair). This gives a total of $2^{18} = 262{,}144$ different state combinations to check. Let X_i denote the state of component i, where $X_i = 1$ if component i is working and $X_i = 0$ if component i is failed. Let A_i denote the state of function i, where $A_i = 1$ if function i is working and $A_i = 0$ if function i is failed. The relationships between the component states and the function states are given by the reliability diagrams in Figure 2.

Let $p_i = \Pr\{X_i = 1\}$ denote the probability that component i is working. The components are assumed to function independently, so the probability of observing a particular state $\{X_1, X_2, ..., X_N\}$ is the N-fold product of the probabilities of the individual states:

$$p(X_1,\dots X_N) = p_1^{X_1}(1-p_1)^{1-X_1}\cdots p_N^{X_N}(1-p_N)^{1-X_N} \; .$$

To compute the joint probability of observing the system in a particular functional state $\Pr\{A_1, A_2, A_3, A_4\}$, loop through all 2^N possible component states $\{X_1, X_2, ..., X_N\}$. For each state:

1. Compute $p(X_1, \ldots, X_N)$ via the previous equation.

2. Determine the functional state $\{A_1, A_2, A_3, A_4\}$ associated with the component state $\{X_1, X_2, \ldots, X_N\}$ via the reliability diagrams.

3. Increment the joint probability $\Pr\{A_1, A_2, A_3, A_4\}$ by $p(X_1, \ldots, X_N)$.

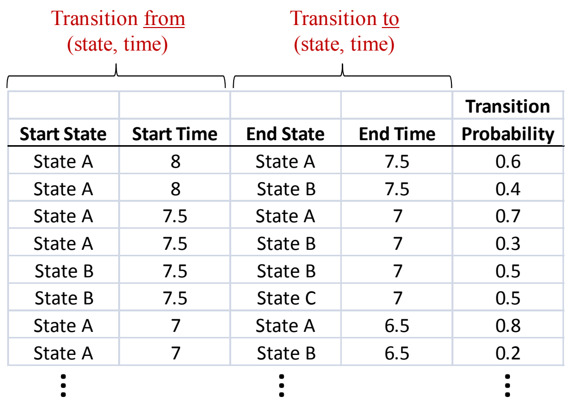
A key advantage of this method is that it is simple and straightforward to implement. The analyst only needs to specify the reliability diagrams that relate the underlying components to the higher level functions. All other calculations are automated. The reliability diagrams are entered via a simple format in an Excel spreadsheet. Changes to the model – for example, adding a redundant component – can be completed by modifying a line or two in the spreadsheet.

One potential disadvantage is that the method does not scale for large numbers of components. In our implementation, the evaluation of 2^{18} states takes about 2 seconds in Visual Basic on a PC, so computer time is not a serious problem. However, a larger number of components might pose a problem. One solution, which we have implemented, is to only enumerate a subset of the 2^N states – namely, states in which m or fewer components have failed (e.g., $m = 5$). The rationale is that states involving more than m component failures are exceedingly rare and contribute a negligible amount to the final results. This type of approximation can drastically reduce the computation time with little loss in accuracy. However, all results reported in this paper are derived from the complete enumeration of all states.

Specification of Dynamic Event Trees

A standard event tree specifies potential sequences of events in a tree-like format with transition probabilities at each branch of the tree. A standard event tree does not account for the particular *time* that an event occurs. For example, a standard event tree might have a branch that specifies whether or not a conflict is detected, but it does not account for the actual time that the conflict is detected. The time of detection may have an impact on downstream events, such as the ability of the pilot to execute the resolution. We extend standard event trees by adding the dimension of time to the state space. We call these trees *dynamic event trees* (DETs).

A DET is defined by a set of state-time pairs and the transition probabilities between such pairs. In our implementation, the transition probabilities are specified via a table, where each row in the table corresponds to one possible transition in the tree (Figure 3). The first two elements in a row specify the starting state-time pair, where a state-time pair (i, j) denotes that the system is in some state i at some time j. The next two entries specify the ending state-time pair. The final entry specifies the transition probability between the two state-time pairs. This is the probability that the system makes its next transition to the ending state-time pair, given that the system is currently in the starting state-time pair. For a given starting state-time pair, the outgoing transition probabilities must sum to 1.


Transition from (state, time)
Transition to (state, time)

Start State	Start Time	End State	End Time	Transition Probability
State A	8	State A	7.5	0.6
State A	8	State B	7.5	0.4
State A	7.5	State A	7	0.7
State A	7.5	State B	7	0.3
State B	7.5	State B	7	0.5
State B	7.5	State C	7	0.5
State A	7	State A	6.5	0.8
State A	7	State B	6.5	0.2
⋮		⋮		⋮

Figure 3. Transition Probabilities (Notional)

To populate the transition-probability table in Figure 3, we start with a flow diagram of potential event sequences in AAC. Figure 4 shows an example. These diagrams are constructed based on textual descriptions of the algorithm in [8]. For example, [8] describes a clock that counts backwards in time in increments of $\Delta = 0.5$ minutes, starting from 8 minutes prior to a potential NMAC. At each time step in the interval [8, 3), the Auto-Resolver attempts to detect and resolve the conflict. If unsuccessful, TSAFE overrides the Auto-Resolver and attempts to detect and resolve the conflict at each time step in the time interval [3, 1). This logic is captured in the upper loops of the flow diagram.

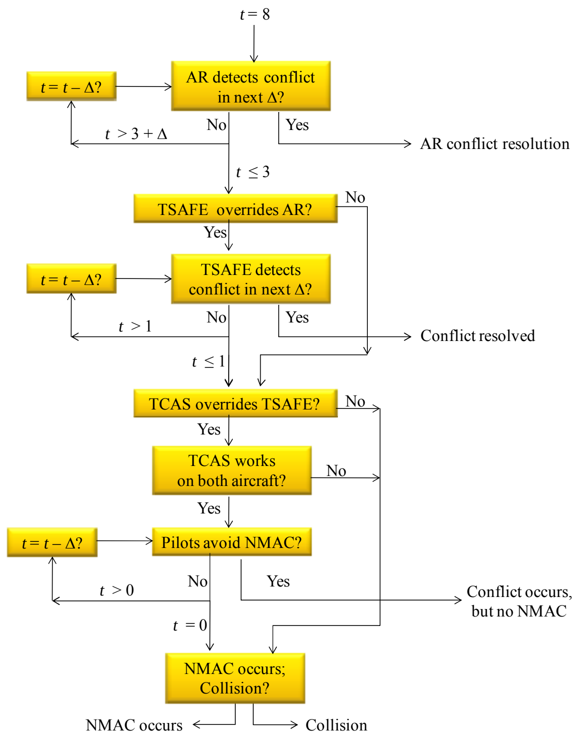


Figure 4. Flow Diagram for AAC Events

Figure 4 is based primarily on the model described in [8]. Figures 5 and 6 give additional branches to the flow diagram that are new to this paper. The complete flow diagram is defined by the combination of Figures 4, 5, and 6.

Figure 5 gives the logic for resolution generation by the Auto-Resolver. If the Auto-Resolver cannot generate a resolution, the controller may provide a resolution as a backup. The controller has a fixed amount of time to provide a resolution. If the controller cannot find a resolution prior to 3 minutes before the NMAC, TSAFE overrides and attempts to find a resolution. Figure 6 gives the event-tree logic for the pilot response to an Auto-Resolver resolution. At each time step, the pilot has some fixed probability of accepting the resolution. If the pilot delays too long in accepting the resolution, TSAFE overrides and attempts to find a resolution.

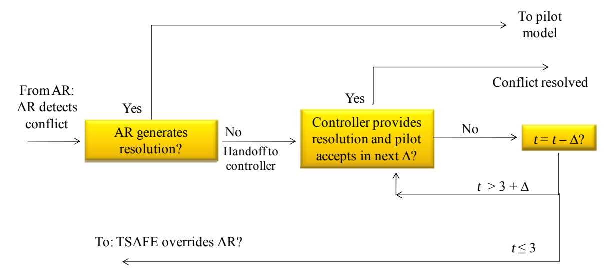
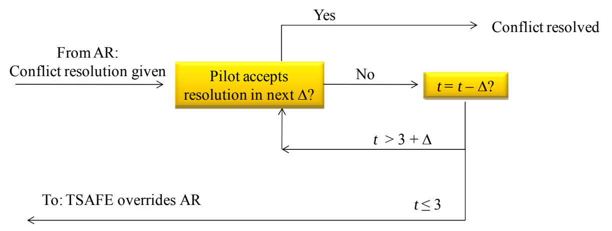


Figure 5. Controller Override of Resolution



Yes
Conflict resolved
From AR: Conflict resolution given
Pilot accepts resolution in next Δ?
No
t = t – Δ?
t > 3 + Δ
To: TSAFE overrides AR
t ≤ 3

Figure 6. Pilot Response

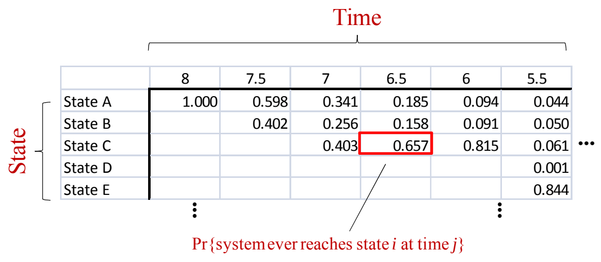
The transition probabilities for the branches in these diagrams are given by parameters in Table 2. Most probabilities are static, but the conflict-detection probabilities are time-dependent. This is because the probability of detecting a conflict is higher when the conflict is closer in time, since there is more certainty in the predicted aircraft trajectories over shorter time horizons.

In summary, the structure of the flow diagrams in Figure 4, 5, and 6 define the set rows in Figure 3. The transition probabilities are populated with values given in Table 2.

Evaluation of Dynamic Event Trees

To solve the DET, our implementation automatically constructs a table giving the computed probability of reaching every possible state-time pair in the tree, given the table of transition probabilities in Figure 3. Figure 7 shows an example portion of the constructed table (the example is notional). The rows specify the system states; the columns specify the set of possible times (the times represent minutes prior to a potential NMAC, so the values are in descending order). A cell at row *i* and column *j* denotes

$$q_{ij} \equiv \Pr\{\text{System ever reaches state } i \text{ at time } j\}.$$


Time
State
…
Pr{system ever reaches state i at time j}

	8	7.5	7	6.5	6	5.5
State A	1.000	0.598	0.341	0.185	0.094	0.044
State B		0.402	0.256	0.158	0.091	0.050
State C			0.403	0.657	0.815	0.061
State D						0.001
State E						0.844

Figure 7. Evaluation of DET (Notional)

The values q_{ij} are calculated recursively via the following equation:

$$q_{ij} = \sum q_{kl}\, p_{kl,ij}\ , \qquad (1)$$

where $p_{kl,ij}$ is the transition probability from state-time (k, l) to state-time (i, j); the summation is taken over pairs (k, l) such that $p_{kl,ij} > 0$. The recursive formula is well-defined provided the DET does not contain any loops, meaning that there is no way to visit the same state-time pair twice. (Note: this does not exclude the possibility of returning to the same state at a different time, or changing states without advancing time.) The recursive formula requires an initial condition, which is specified by $q_{ab} = 1$, where (a, b) is the "root" state-time pair of the tree.

The automation constructs a "live" version of (1) in each cell of the table. As a live spreadsheet, the cells are automatically populated with the calculated numerical values for q_{ij}.

In this model, the system terminates in one of four possible states: conflict resolved, conflict, NMAC, or collision. Terminating states have no outgoing transition arcs. The probabilities in the table associated with these states correspond to the key "outputs" of the DET.

Computing Final Results

The final step in the analysis (Figure 1) is to link the outputs of the reliability diagrams and the DETs. In our model, we define 4 different DETs. The main DET is the "baseline" tree corresponding to a nominal scenario in which all functions are working. This baseline tree is defined in Figures 4, 5, and 6. Three other DETs are defined corresponding to failures of different system functions. These are shown in Figures 8, 9, and 10. These trees correspond to the following scenarios:

- Auto-Resolver not functional. A scenario in which the Auto-Resolver cannot communicate a resolution to at least one aircraft.
- TSAFE not functional. A scenario in which TSAFE cannot communicate a resolution to a least one aircraft.
- Aircraft not locatable: A scenario in which one or both aircraft in a pair are not locatable.

Each variant is a modification of the baseline tree obtained by adding a "short-circuit" transition to the original tree. For example, Figure 8 shows the modified tree in the case that the Auto-Resolver cannot communicate a resolution to at least one aircraft in the pair. In this case, the event tree jumps directly to the part of the tree where TSAFE overrides the Auto-Resolver. Figure 9 shows a similar scenario in which TSAFE cannot communicate a resolution to either aircraft. Figure 10 shows a scenario in which one or more aircraft is not locatable.

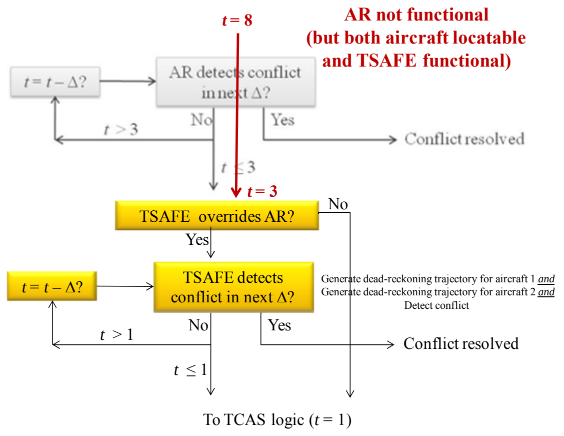


Figure 8. AR Cannot Communicate Resolution

The modified DETs are specified by a list of transition probabilities, just as in Figure 3. The list of probabilities for these DETs differs from the original list by only a few rows, since the modified trees differ from the original tree by a single additional transition branch.

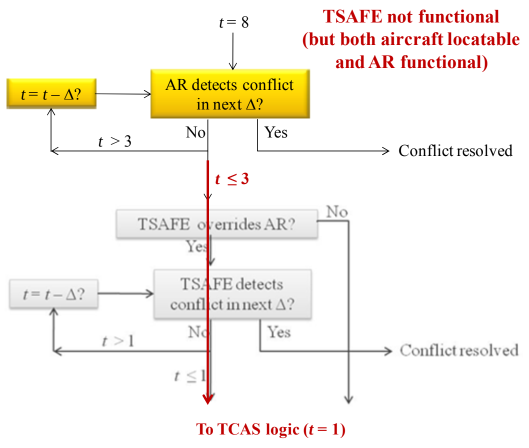


Figure 9. TSAFE Cannot Communicate Resolution

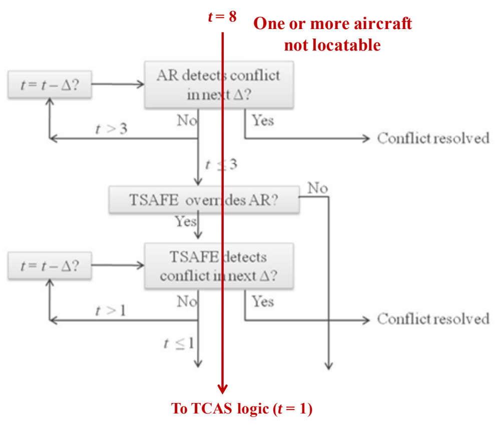


Figure 10. Aircraft not Locatable

The final collision probability is computed as a weighted sum of the collision probabilities obtained from evaluating each event tree, where the weights are the probabilities of the functional failures associated with each tree. For example, the probability of using the event tree from Figure 8 is the probability that (a) the Auto-Resolver cannot communicate a resolution to either aircraft, *and* (b) both aircraft are locatable, *and* (c) TSAFE can communicate a resolution to a least one aircraft. The joint probability of this event comes from the analysis of the reliability diagrams described previously. Alternate outcomes for (a), (b), or (c) result in using other trees (e.g., Figure 9 or 10).

The state of the TCAS function is implemented in a slightly different way – by changing appropriate probabilities within the event trees themselves, rather than by constructing new event trees. Specifically, within each DET, there is a branch that corresponds to whether or not TCAS is functional. That probability is populated with the conditional probability that TCAS is functional given the states of the other three functions corresponding to that dynamic event tree. This value comes from the joint evaluation of the four reliability diagrams, as described previously.

Results

This section shows the results of a sensitivity analysis applied to the model. A sensitivity analysis is important because many of the underlying model parameters are not known precisely, due to the rare-event nature of the problem (see model parameters in Table 2).

Furthermore, the model is designed at a high-level and makes many simplifying assumptions. For example, differences in conflict geometries are ignored. Thus, the absolute collision probabilities computed from the model may not be accurate; hence, they are not reported in this paper. On the other hand, a sensitivity analysis showing the *relative* impact of different input parameters may provide more convincing conclusions.

In the sensitivity analysis conducted here, each parameter is modified from its baseline value, one parameter at a time (see Table 2 for the baseline values.) We vary each parameter in the table *except for the conflict-detection probabilities*. These probabilities vary with time, so it is not as obvious how to parameterize the variation of these probabilities. For all other parameters, two cases are considered: Multiplying each parameter by 10, and dividing each parameter by 10. Figure 11 shows results in the case of multiplication by 10. Results in the reverse case are similar. (Some probabilities, when multiplied by 10, are greater than 1; such probabilities are capped at a maximum value of 1.)

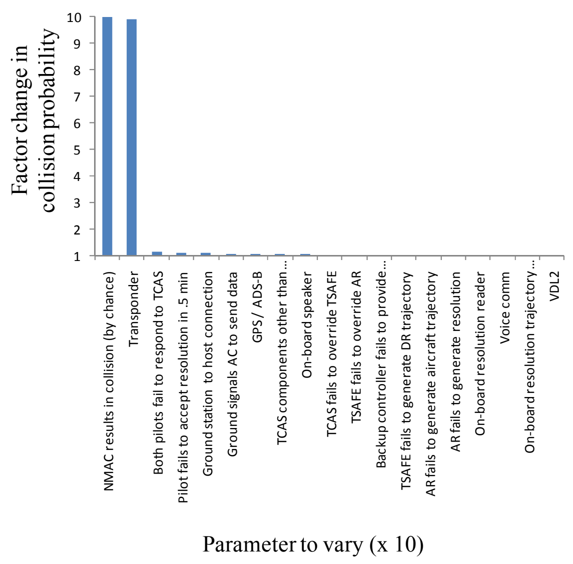


Figure 11. Sensitivity Analysis

The *x*-axis lists the parameters considered, sorted in decreasing order of impact on the collision probability. The *y*-axis shows the factor change in the final estimated collision probability as a result of multiplying the underlying parameter value by 10, holding the other parameters at their baseline values. A value of 1.0 on the *y*-axis indicates no change from the baseline. The two most significant factors are:

- Transponder failure probability
- Probability that NMAC results in a collision (by chance)

The latter probability is a multiplicative factor that converts the probability of an NMAC to the probability of a collision. Since this factor occurs in every path of the event tree terminating in a collision, the overall collision probability scales linearly with this parameter. Thus a 10-fold increase in this probability results in a 10-fold increase in the collision probability.

The collision probability is also quite sensitive to the transponder failure probability. In particular, if the transponder failure probability increases by a factor of 10, then the collision probability increases by a factor slightly less than 10. The transponder is significant, because it is used in three functional areas of AAC: to locate aircraft, to communicate TSAFE resolutions, and for use in TCAS. Furthermore, there are two transponders in the event tree – one for each aircraft in a pair – so an increase in the transponder failure probability results in modifying the failure probability for both transponders. Thus, the failure of the transponder results in a significantly increased risk of collision.

This sensitivity analysis has excluded variation of the conflict-detection probabilities. These probabilities are expected to also have a significant impact on the overall collision probabilities.

Comparatively speaking, the other parameters in the event trees and fault trees have significantly less impact. For most parameters, increasing or decreasing their values by a factor of 10 has virtually no effect on the overall collision probability. This is because the failure of other components dominates the contribution to the collision probability.

Summary and Conclusions

This paper presented a model and methodology for a safety and sensitivity analysis of the Advanced Airspace Concept. This analysis is part of a larger effort to analyze safety-capacity tradeoffs in NextGen concepts (e.g., [6], [7]).

The model is an analytical implementation of a similar Monte-Carlo simulation model given in [8] and [9]. The analytical implementation uses a combination of fault trees and dynamic event trees. We define a dynamic event tree as an event tree that also includes the dimension of time in the state-space description.

A key contribution is that the model is constructed and evaluated in an automated fashion based on a small number of input tables. Thus, changes to the model are easily implemented through simple changes to the input tables. For example, adding a redundant component requires changing the definition files for the reliability diagrams in Figure 2 (typically, changing one or two entries in a spreadsheet). All results can then be automatically re-calculated.

The analytical implementation also allows for relatively quick evaluation of the model (a couple seconds per evaluation). So it is possible to conduct a systematic sensitivity analysis of all parameters in the model. The sensitivity analysis showed that the transponder failure probability is one of the most critical parameters in the model. The transponder is significant, because it is used in three functional areas of AAC: to locate aircraft, to communicate TSAFE resolutions, and for use in TCAS. The conflict-detection probabilities are also expected to be significant, though this was not specifically tested in this analysis and is a subject for future work.

References

[1] Erzberger, H. 2001. The automated airspace concept. 4th USA/Europe Air Traffic Management R&D Seminar, Santa Fe, NM.

[2] Paielli, R., H. Erzberger. 2004. Tactical conflict detection methods for reducing operational errors. *Air Traffic Control Quarterly*, 13(1), 83-106.

[3] Erzberger, H. 2006. Automated conflict resolution for air traffic control. 25th International Congress of the Aeronautical Sciences.

[4] Erzberger, H., T.A. Lauderdale, Y. Chu. 2010. Automated conflict resolution, arrival management and weather avoidance for ATM. 27th International Congress of Aeronautical Sciences.

[5] Erzberger, H., K. Heere. 2010. Algorithm and operational concept for resolving short-range conflicts. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, 224(2): 225-243.

[6] Yousefi, A., R. Xie. 2011. Safety-capacity trade-off and phase transition analysis of automated separation assurance concepts. Digital Avionics Systems Conference, Seattle, WA, 1B5-1 – 1B5-9.

[7] Belle, A., J. Shortle, A. Yousefi, R. Xie. 2012. Estimation of potential conflict rates as a function of sector loading. To appear in 5th International Conference on Research in Air Transportation, Berkeley, CA.

[8] Blum, D.M., D. Thipphavong, T. Rentas, Y. He, X. Wang, M. Pate-Cornell. 2010. Safety analysis of the advanced airspace concept using Monte Carlo simulation. AIAA Guidance, Navigation, and Control (GNC) Conference, Toronto, Canada, August 2–5.

[9] Thipphavong, D. 2010. Accelerated Monte Carlo simulation for safety analysis of the advanced airspace concept. AIAA ATIO/ISSMO Conference.

[10] Volovoi, V., A. Balueva, R. Vega. 2011. Analytical risk model for automated collision avoidance systems. Submitted to *Journal of Guidance, Control, and Dynamics*.

[11] Andrews, J., J. Welch, H. Erzberger. 2005. Safety analysis for advanced separation concepts. 6th USA/Europe Air Traffic Management R&D Seminar, Baltimore, MD.

[12] Hemm, R., A. Busick. 2009. Safety analysis of the separation assurance function in today's national airspace system. LMI Research Institute, report NS801T2.

Acknowledgements

This work was sponsored by NASA Ames Research Center (NRA number NNH08ZEA001N). The authors thank Mr. David Thipphavong and Dr. Heinz Erzberger for their support and contribution.

Disclaimer

The opinions and results in this paper are solely those of the authors.