This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2018.2852324, IEEE Access

Graphical Modelling and Analysis Software for State Space-based Optimization of Discrete Event Systems

Jun Tang, Feng Zhu

Abstract—**In view of the particular research objectives rather than the system's characteristics, almost all systems can be discretized regardless of original continuous or discrete pattern. Modelling oriented to discrete events system (DES) represents the dynamics of a system as a series of discrete events that perform changes in the state of the system, constituting the state space which supports the further analysis for scheduling and optimization. In this paper, the graphical modelling and analysis software (GMAS) as a platform for modelling DES is introduced, with the basic notions and a general perspective on the systems approach. It clearly provides the graphic modelling and analysis interface. Besides, the system evolution process is recorded and represented by state space, transforming the optimization problem into a search-based issue in the reachability tree of finding the optimal or near-optimal sequence of function component activations from some initial state to the goal state. To validate its efficacy and practicability, a causal encounter model of traffic collision avoidance system (TCAS) operations is proposed in the GMAS formalism. The model has been proved to not only provide a better comprehension of the potential collision occurrences for risk assessment by representing the cause-effect relationship of each action, but also aid the crews in the involved aircraft to make a cooperative and optimal option.**

Index Terms—**GMAS, State space, Discrete events, Optimization problem, Potential induced collisions**

I. Introduction

MOST systems can be roughly classified considering the time evolution of the properties of interest as continuous or discrete [1]. In a continuous system the state variables evolve continuously over time. These are called "continuous variables" in the sense that they can take on any real value as time itself "continuously" evolves. In a discrete system, the state variables change only at a certain instant or a sequence of instants (discrete set of points in time) known as the events, and remain constant between events [2].

It is well accepted that a continuous system can be described using a discrete representation, and vice versa a discrete system can also be described by a continuous model. The choice of employing a continuous or a discrete representation depends on the purpose of investigation (particular objectives) of each study rather than the characteristics of the system. Discrete event system (DES) is a unified modelling framework which recently emerged integrating traditionally separate disciplines such as queuing theory, supervisory control, and automata theory [3]. DES is defined as "a discrete-state, event-driven system, that is, its state evolution depends on the occurrence of asynchronous discrete events over time" [4]. In many situations, the system under consideration can be modelled as a DES and the problems can be translated into state estimation problems in a DES framework [5].

The distinction between DES and the more familiar time-driven dynamical systems studied under control theory, for example, is subtle but important: the state-transition mechanism in the latter is driven by time alone or is synchronized by "clock ticks", whereas state transitions in DES are driven by "discrete events" (e.g., press of a button, arrival of a shipment) which can occur asynchronously (at various time instants not necessarily known in advance or coinciding with clock ticks) [1].

In the discrete event-based models, events (i.e., the state changes) can be depicted by a graph-based notation with several nodes and the relations between those events are represented using the links [4]. Thus, a series of discrete events that form the model record the dynamics of a system to perform the state changes, and the links define the relations between events. The sequence of generated states constitutes a database called the state space which supports the further analysis for scheduling and optimization. These DES representations aim to describe the occurrence of finite number events in a discrete time base, (i.e., events happen in a continuous time base, but during a bounded time-span, only a finite number of relevant events occur) [5]. Typical DES includes queuing systems, communication systems and telephony, databases, manufacturing and traffic systems to mention a few [6]. Discrete-event formalisms help to develop a high level of abstraction appropriate for realistic representation of a system's behaviours.

In this research, the graphical modelling and analysis software (GMAS) oriented to DES, clearly provides the graphic modelling and analysis interface. It is extensively used to model, simulate, and analyze DESs characterized by concurrency, parallelism, causal dependency, resource sharing, and synchronization [7]. In addition, the state space analysis is introduced to represent the system evolution process, i.e., a

J. Tang and F. Zhu are with the College of Systems Engineering, National University of Defense Technology, Changsha 410073, China. (e-mail: zhufeng@nudt.edu.cn; jun.tang@e-campus.uab.cat).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2018.2852324, IEEE Access

global perspective on the scenario dynamics and a better understanding of the system principles.

Then, as a case, a causal encounter model absolutely in view of the traffic collision avoidance system (TCAS) [8] logic is proposed using GMAS, and its logic process is explicitly based on the detailed model specification and characterization. To precisely sense the effect of each action, the dynamics of TCAS-equipped aircraft encounters are modelled as a series of discrete events from which the different states of the system can be evaluated. Through the generation of state space, the implemented GMAS-based encounter model not only provides a better comprehension of the potential collision occurrences for risk assessment by representing the cause-effect relationship of each action, but also aids the pilots in the involved aircraft to make a cooperative and optimal option.

The structure of the paper is as follows: Section II gives a review of literature that investigates related work and further describes the state space. Section III introduces the GMAS as the DES modelling tool with detailed specifications. Section IV depicts the proposed GMAS causal model and explains its construction process. Section V represents the computing results and illustrates the in-depth analysis. Finally, the conclusions and future work are described in Section VI.

II. BACKGROUND

Since the early 1970s, various techniques and methods for DES modelling appeared and became very popular in different fields. This resulted in a progressive update and the definition of an advanced field of research on numerous occasions. They try to describe the occurrence of finite number events during a bounded time-span which consists of numerable discrete time points, and these events can cause a change in the system state available for the quantitative analysis.

A. Related Work

Discrete-event formalisms contribute to develop a high level of abstraction appropriate for realistic representation of a system's behaviours. Owed to the large number of methodologies for modelling and analysing DES [9], we do not intend to provide a comprehensive evaluation. Considering the research correlation, state space as the search criteria is used for filtering the long tool list and among them several typical ones are worthy to mention as follows.

MARIA [10] is one of the earliest platforms that contain a certain collection of tools, providing a reachability graph analyser exclusively for algebraic system. GPenSIM [11] is designed using the well-proven paradigms (i.e., the layered architecture, modular components, and natural language interface) in software engineering, and embedded into third-party commercial software packages, such as MATLAB, which demands the developers to have the usage ability of another language. Graphct [12] is a scalable framework for graph analysis using parallel multithreaded algorithms on shared memory platforms. SimQPN [13] used for the control and scheduling of queuing systems is currently a strictly sequential program and cannot exploit the parallelism provided by modern multi-core processors. A visual modelling toolkit [14] is presented to support model implementation, model execution, and experimentation for the extended activity cycle diagram models. SimEvents [15] provides a discrete-event simulation engine and component library; users can model event-driven communication between components to analyse and optimize end-to-end latencies, throughput, packet loss, and other performance characteristics. Specifically, Viskit [16] is a graphical front end for creating, editing, and composing DES simulation models using event graphs and the LEGO framework; each LEGO is an instance of an event graph, which is responsible for the events and state transitions that modify its state variables and produce its state paths. ASAP [17] is employed to support advanced state space methods, and it relies heavily on the Standard Markup Language (SML), a proprietary functional programming language, making it hard to integrate private search algorithm. As one of the most commonly used tools for modelling and simulating DES, CPN Tools [18] stands out as an industrial strength software that provides both a graphical editing interface and an interactive simulator for constructing and analyzing models. However, its early version supports extraordinarily simple calculation only and even with this extension, the up-to-date version is still difficult to integrate complex operations [19]. It has a state space analysis plug-in, but the absence of efficient search algorithms has limited its applicability and it cannot scale up to industrial-sized problems [20].

For the experiments, we previously used the state space analysis tool called TIMSPAT [21], developed at the Logistics and Aeronautics Unit of the Autonomous University of Barcelona. The tool has been shown to be effective for the performance analysis of very demanding and flexible industrial systems [22,23]. Yet it is failing to provide the graphic modelling interface faced to the model developers. In addition, the DES model developed by TIMSPAT is constituted by a set of text files so that it is not easy to understand the model architecture, and the errors are difficult to detect in the developing process.

Note that the widely used Discrete Event System Specification (DEVS) is a modular and hierarchical formalism for modelling and analyzing DESs which might be described by state transition tables, continuous state systems which might be described by differential equations, and hybrid continuous state and DESs [24]. DEVS formalizes what a model is, what it must contain, and what it does not contain (e.g., experimentation and simulation control parameters). It is utilized as a timed event system, which means that any system that accepts events as inputs over time and generates events as outputs over time could be represented as a DEVS, to describe its behaviour and structure [25]. Thus far the DEVS formalism and its variations have been applied in many areas of engineering, for instance, the manufacturing systems [26], the embedded systems [27], the hardware-software co-design [28], and the aircraft related problems [29,30] especially.

To absorb the advantages and overcome the shortcomings of the above-mentioned tools, the powerful GMAS has been developed to be extensively used to model, simulate, and analyse DESs. Certainly the above-mentioned tools, DEVS in

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2018.2852324, IEEE Access

particular, could be used to solve the TCAS problem in this research, GMAS proposed in this paper has the following distinguished features from the existing ones to build the encounter model:

- Succinct graphical expressiveness makes it easy to construct a DES model, and the developed model is highly readable and simply comprehensible.
- Easy-to-write syntactical structure facilitates the needless to learn an extra programming language.
- Capability to provide a concise and precise system representation with the use and manipulation of data attributes that are set in corresponding components.
- Modularization design by using MVC (model/view/control) pattern makes it easy to extend and maintain.
- Graphical display of the state space is constructed to explore all the possible alternatives in order to determine the best schedule that optimises a given performance objective.
- Support of complex algorithms is conducive to design ingenious methodologies to find optimal or near-optimal solutions of large-sized problems within acceptable computation time, that simplifies the exhaustive state enumeration to avoid the well-known state explosion problem.
- Various combinations via selecting and assembling model components into valid simulation systems promote the system's combinability, that fully satisfy the specific requirements of users.

B. State Space Analysis

The state space analysis enhances a quantitative approach, relying on computational tools to explore the different states that DES could reach, starting from a particular initial state [31]. The system state is characterized by the entities with its attributes distributed in the different data storage units. The state space is generated quantitatively by firing all the enabled data computing units at any system state, calculating the new states.

The state space also can be graphically displayed called reachability tree or occurrence diagrams [32]. The basic idea of state space analysis is to calculate all reachable states and state changes of the DES model and to represent these in a directed graph where the nodes correspond to the set of reachable states and the arcs correspond to events. Hence, the state space contains all the possible occurrence sequences and reachable states that can be achieved from a given state.

The reachability tree (first level) as a simple case shown in Fig. 1, and the state vector of the DES model with three data storage units is represented. In each position of the vector, the data stored in the corresponding data storage unit are illustrated. Given this initial marking, the only enabled events are those that are indicated by data computing units (i.e., function components in this research) f_1 and f_2. It should be noted that f_2 could be fired by using two different combinations of entities. Once a function component has been fired, a new state vector is generated (e.g., a new traffic scenario in the case study). Thus, a proper implementation of a DES model in a simulation environment should allow automatic analysis of the whole search space of the system by firing the different sequences of events without requiring any changes in the simulation model [33].

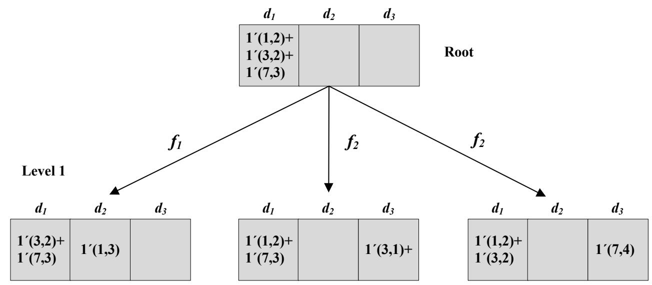

Fig. 1: A simple example of reachability tree.

There are several existing methods, such as time Dependent Markov Decision Process [34], Timed Automaton [35], etc., which could be used to generate and analyse the system states. For instance, in [36], aircraft flight is modelled using a Markov process and this means that the future state of the trajectory is only dependent on the current state. In this research, the reachability tree of system operations applied to a certain scenario provides a deeper understanding of the cause-effect relationship of each action and how the effects of an action are propagated upstream and downstream through the different actions. The state space is normally characterized by a plenty of nodes and arcs. Therefore, state space methods are closely tied to supporting computer tools. It is possible to analyse and verify an abundance of properties of the system such as reachability, boundness, activeness, among others [37].

In this research, the operations of TCAS also can be modelled as a discrete sequence of events in time; each event occurs at a particular instant in time and can cause a change of the system state. The GMAS encounter models based on TCAS logic can act as useful tools for better understanding the aircraft interdependence between the own aircraft and its surrounding traffic conditions (both at macro and micro levels) that could assist the air traffic controllers (ATCo)s and pilots, and also to check for future TCAS logic updates. In this research, the proposed discrete event-based models developed by GMAS have the following representative features:

- Dynamic, each event can determine the results of corresponding action. Its dynamics could form complex patterns of behaviour to represent the unknown effects especially unreasonable decision which may initiate undesirable consequences.
- Complex, the decisions and actions may be various in each step. The complex models have many interrelated causal relationships that interact between sub-modules, and these relationships could cause different results of the system.
- Conditional, the manoeuvres operate at the corresponding moment or with relevant conditions to achieve its goal. When several certain conditions are

satisfied the specific action can be activated, while it would be invalid if the conditions are not met or changed.

III. The Specifications of GMAS

The section introduces the GMAS which provides a platform for describing DES models as well as simulating the behaviour of the system, and records the system's evolutionary process (state space) to obtain optimal results. The following four aspects model formalism, graphical elements, activation rules of function components and model characteristics are modelled for the GMAS specifications.

A. Model Formalism

The graphical components of GMAS model include start component, data component, function component, nested function component, link component and end component.

Definition 1 A GMAS model can be defined as the following nine-tuple:

$$GM = (S, D, H, H', F, E, F_W, M, M_0) \quad (1)$$

where

$S = \{s_1\}$ represents the set of start components, and the element is unique.

$D = \{d_1, d_2, \cdots, d_a\}$ represents the set of data components, and a is the amount.

$H = \{h_1, h_2, \cdots, h_m\}$ represents the set of function components, and m is the amount.

$H' = \{h'_1, h'_2, \cdots, h'_n\}$ represents the set of nested function components, and n is the amount.

$F = \{f_1, f_2, ..., f_u\}$ represents the set of link components, and u is the amount.

$E = \{e_1\}$ represents the set of end components, and the element is unique.

$F_W : F \rightarrow \{f_{1,w}, f_{2,w}, ..., f_{u,w}\}$ is the set of functions on each link component.

$M : S \cup \quad \cup \quad {}_1, e_1, d_1, d_2, ..., d_a\}$ is the set of state identifications, which are the state data of start component, end component and data components during the model operation.

$M_0 : S \cup \quad {}_{,0}, d_{1,0}, d_{2,0}, ..., d_{a,0}\}$ is the set of initial identifications, which are the initial state data of start component and data components before the model operation.

$D \cap \quad \cup$ (set D does not intersect with the union of set H and H'), and $D \cup \quad \cup$ (set D and the union of set H and H' are not empty at the same time).

$F \subseteq [(S \cup \quad \cup \quad \cup \quad \cup \quad \cup \quad \cup \quad \cup$ indicates that link component connects start component, data component or end component with function component or nested function component, and it is the set of directed arcs.

Among them, $G = (S, D, H, H', F, E, F_W)$ forms the physical structure of the GMAS model, (G, M) is called identified GMAS model, and its feature is the introduction of state identification M which is a vector set of the data in start component, end component and data components at corresponding time. (G, M, M_0) expresses the complete GMAS model in which the initial states M_0 has been provided with the input data in start component and the initial data in data components at the beginning.

B. Graphical elements

The GMAS model components (start component, data component, function component, nested function component, link component, and end component) are shown in Fig. 2.

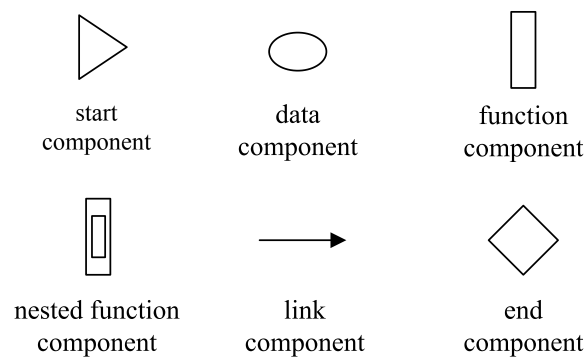

Fig. 2: Graphical model components.

Start component: It aims to configure the initial input data of the simulation model, and provides the data input interface for the model execution. Double-click start component to bring up the corresponding configuration dialog box, and then configure the properties of the initial input data, including data name, data type, data value and data description, etc.

Data component: It is used to store the state data of the simulation model. Double-click data component to bring up the corresponding configuration dialog box, and then configure the properties of the storable state data, including data name, data type, data values, and data description, etc.

Function component: It is employed to indicate the conditions and computing functions under which the state data change can be made. When the computation represented by a function component is allowed to execute, the state data on the connected data components will change. Double-click function component to bring up the corresponding configuration dialog box, and then configure the operating conditions, computing functions and function specification, etc.

Nested function component: It is applied to describe the simulation sub-model and assist in the construction of more complex GMAS models. A GMAS model can have multiple nested function components, which means a simulation model can contain multiple sub-models. Double-click nested function component to bring up the graphical build interface of the corresponding simulation sub-model. The components of the simulation sub-model GM_1 still is a nine-tuple $GM_1 = (S_1, D_1, H_1, H'_1, F_1, E_1, F_{W_1}, M_1, M_{0_1})$.

Link component: It is adopted to transfer data and represent the data flow. Double-click link component to bring up a corresponding configuration dialog box, and then configure the properties of the transfer data, including data name, data type and data amount, etc.

A directed arc from data component node d to function component node h can be expressed as (d,h), and this data component d is called an input of the function component h while the function component h is called a latter association of the data component d, and the state data on this directed arc is labelled a weight $W(d,h)$; A directed arc from function component node h to data component node d can be expressed as (h,d), and this data component d is called an output of the function component h while the function component h is called a forward association of the data component d, and the state data of this directed arc is labelled a weight $W(d,h)$.

The forward association set of data component d is defined as:

$$I(d)=\{h|(h,d)\in F\} \tag{2}$$

The latter association set of data component d is defined as:

$$O(d)=\{h|(d,h)\in F\} \tag{3}$$

The input set of function component h is defined as:

$$I(h)=\{d|(d,h)\in F\} \tag{4}$$

The output set of function component h is defined as:

$$O(h)=\{d|(h,d)\in F\} \tag{5}$$

Analogously, the latter association set of start component s_1 and the forward association set of end component e_1 are respectively defined as $O(s_1)=\{h|(s_1,h)\in F\}$ and $I(e_1)=\{h|(h,e_1)\in F\}$. The state data of their directed arcs are respectively labelled a weight $W(s_1,h)$ and $W(h,e_1)$.

End component: It is designed to store the computing result data of the simulation model, and provides GMAS model the data output interface. Double-click end component to bring up the corresponding configuration dialog box, and then configure the properties of the output data, including data name, data type, data value and data description, etc.

C. *Activation Rules*

The simulation model can only describe the static structure of the system, while the dynamic behaviour of the system is represented by transformation of the state identification. Whether the state identification can change is determined by the excitation rule of function components. Nested function components operate exactly the same as function components in the GMAS model, and therefore this description of activation rules takes function components as an example.

Rule 1 [Activation rules of function components] The function component node h is considered to be excitable, when the following conditions are satisfied:

(1) For each input data component $d_i\in I(h)$ of the function component h, the state identification $M(d_i)$ contained in the data component d_i is not less than the weight $W(d_i,h)$ of the corresponding directed arc (d_i,h), namely $M(d_i)\geq W(d_i,h)$.

(2) For each output data component $d_j\in O(h)$ of the function component h, the capacity $V(d_j)$ of the data component d_j is enough for new state identification, which means $V(d_j)\geq M(d_j)+W(h,d_j)$, among which $M(d_j)$ is the state identification contained in the data component d_j and $W(h,d_j)$ is the weight of the directed arc (h,d_j).

(3) For each simultaneous input and output data component $d_x\in I(h)\cap O(h)$ of the function component h, the data component d_x satisfies the above two relations at the same time, that is $M(d_x)\geq(d_x,h),V(d_x)\geq M(d_x)+W(h,d_x)$.

(4) For the function component h connecting with the start component s_1 or the end component e_1, the roles of them are the same as input and output data component respectively.

Rule 2 [Operation after the activation of function components] After the activation of a function component node h, the following operations will take place:

(1) Subtract the state identification from each input data component/start component of the function component node h, and the subtracted state identification is equal to the weight of the input directed arcs from each input data component/start component to the function component node h.

(2) Add state identification to each output data component/end component of the function component node h, and the added state identification is equal to the weight of the output directed arcs from the function component node h to each output data component/end component.

Therefore, after the activation of the function component node h, the state identification $M(S|E|D)$ will change to the new state identification $M'(S|E|D)$:

$$M'(S|E|D)=\begin{cases} M(s_1)-W(s_1,h), & if \quad s_1\in I(h) \\ M(e_1)+W(h,e_1), & if \quad e_1\in O(h) \\ M(d)-W(d,h), & if \quad d\in I(h), d\notin O(h) \\ M(d)+W(h,d), & if \quad d\in O(h), d\notin I(h) \\ M(d)-W(d,h)+W(h,d), & if \quad d\in I(h)\cap O(h) \\ M(s_1),M(e_1),M(d), & else \end{cases} \tag{6}$$

D. *Model Characteristics*

Different GMAS models have different structures, parameters and initial state identifications, thus the state change will exhibit different characteristics during the running of the models. The structural characteristics of the simulation model are independent of the initial state identification; the dynamic characteristics of the simulation model are related to the initial state identification.

(1) Reachability

Definition 2 For a GMAS model (G,M_0) with given initial state identification M_0, the reachable set $K(G,M_0)$ is defined as the set of all of the state identifications that can be reached following activation rules under the initial state identification M_0 in this simulation model.

(2) Boundness

Definition 3 For a GMAS model (G,M_0) with given initial state identification M_0, if the simulation model is called $T-$bounded, then: for any reachable state identification $M \in K(G,M_0)$ and data component node d_i, paying attention to the simulation model under the state identification M, the number of the state identifications of the data component node d_i satisfies $M(d_i) \leq T$, among which T is a finite positive integer.

(3) Structural boundness

Definition 4 Let $G=(S,D,H,H',F,E,F_W)$ be the physical structure of a GMAS model, and if the model is bounded under any initial state identification M_0, then G is called structural bounded model.

(4) Activeness

Definition 5 For a GMAS model (G,M_0) with given initial state identification M_0, if the function component node h is called active, then: for any reachable set $K(G,M_0)$ of the initial state identification M_0, an activation sequence of one state identification that contains the function component node h must be existent.

Definition 6 For a GMAS model (G,M_0) with given initial state identification M_0, the simulation model is called active if and only if every function component node of the model is active.

IV. GMAS-BASED ENCOUNTER MODEL

TCAS is designed to be the last-resort airborne system to prevent mid-air collisions (MACs) and significantly reduce near mid-air collisions (NMACs) between aircraft [8]. In essence it is an on-board conflict detection & resolution (CDR) system giving traffic advisories (TAs) to warn the pilots in the visual acquisition of intruder aircraft, and resolution advisories (RAs), to recommend the pilots of escape manoeuvres [38]. The influence of TCAS on safety flight has been effective, beneficial, and significant in reducing the collision probability [39,40]. However, the increased airspace usage can induce a secondary threat as a result of an RA issued by a TCAS, which may issue an inappropriate suggested resolution that resolves a one-on-one encounter with the first threat. This secondary threat may deteriorate to be an induced collision [41].

Though the widespread TCAS has been in application with new developments for more than 30 years, essential parts of its causal analysis, especially those for potential induced collision scenarios that could be considered to be TCAS weakness, seem to have not yet been clearly performed. Thus a GMAS model can be developed as a key approach to analyse the state space of a congested traffic scenario in which the events that could drive an encounter into a collision are explored, and provide enriching traffic alert information in which the optimal advisory could be selected to improve the TCAS collision avoidance performance.

A. Causal Encounter Model based on TCAS Logic

TCAS, independent of any ground inputs, performs surveillance of nearby aircraft to provide their state information so that the collision avoidance algorithms can perform their function. However, because of the wind influence, pilot behaviours or aircraft performance errors, the speed vectors of aircraft may be variable in a certain range during the execution phase. The main functions of TCAS are to communicate the detected threat to the pilot and to assist in resolving the threat by recommending an avoidance manoeuvre. Normally, TCAS, as an alert system operates quietly in the background most of the time. When the TCAS logic determines that an action is required, TCAS interrupts the flight crew to bring the threat to their attention. The developed causal model illustrated in Fig. 3 is based on the aircraft tracking waypoints consisting of three kinds of agents (Agent State variable, Agent Pilot response, and Agent Predictive states) and one nested function component (h_1': TCAS processor).

- Agent State variable contains one function component (h_1) which aims to improve robustness through considering the uncertainty of motion state. Several typical disturbances should be introduced in simulations to test the robustness of the amended trajectories suggested by TCAS advisories under conditions of uncertainties in the operational level. Specially, the speed variation owing to wind instability discussed in this model is identified as the most common factor affecting the en-route trajectory predictions.
- Agent Pilot response owns one function component (h_2) that is used to provide probabilistic pilot reaction for the TCAS advisories. TCAS as a last-resort means that it is a portion of the safety net in the fully complex socio-technical ATM system. Therefore, as a complement to TCAS, some other factors (typically, the pilot response time) can play a considerable role in the process of collision avoidance, and this agent covering the interactions in the sophisticated socio-technical system TCAS is essential.
- The nested function component TCAS processor (h_1') performs sensitivity level (SL) evaluation, threat detection, RA manoeuvre determination and selection, and generation of advisories. The intruder aircraft is also assumed to be equipped with TCAS II in this research, and the avoidance manoeuvre will be coordinated with the intruder aircraft. This sub-model obtains the state data of involved aircraft from the Agent State variable and the possible reactions of pilots from the Agent Pilot response, and provides the TCAS processing procedure for the Agent Predictive computation.
- Agent Predictive computation possesses five function components (h_1,h_2,h_3,h_4,h_5) to explore the complete

state space of the possible future situations. It integrates with Agent TCAS processor h_1' to obtain the related aircraft state which are in the respective CA process. Through generating the state space of a specific multi-aircraft scenario, not only the potential induced collision (TCAS weaknesses) can be identified, but also the optimal combination of advisories can be achieved.

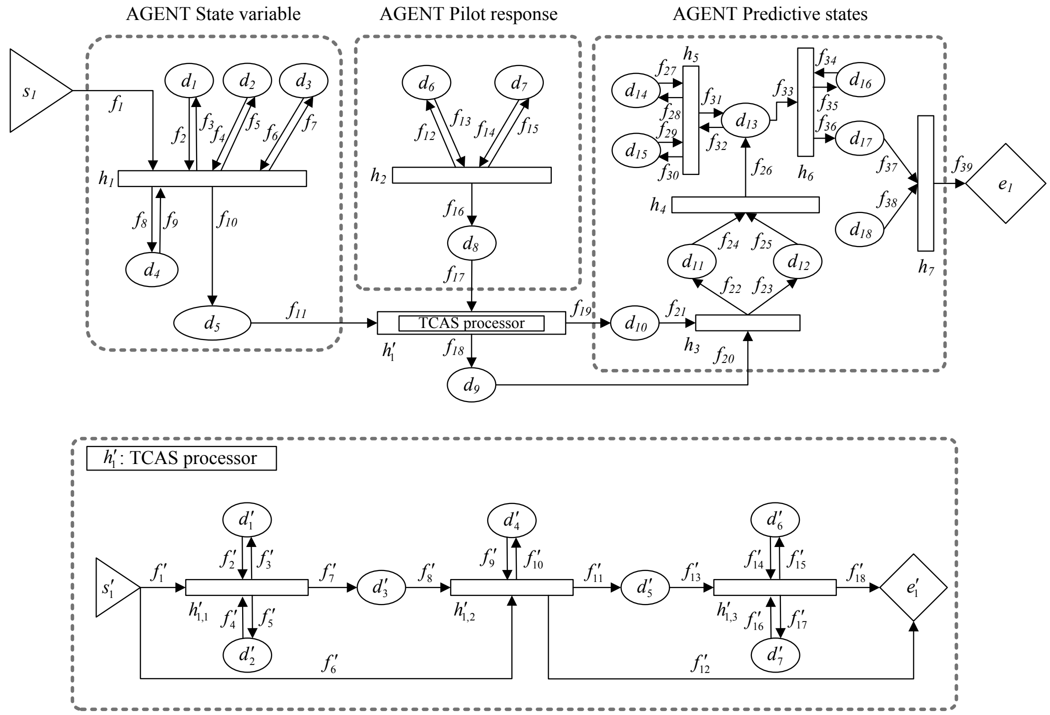

Fig. 3: GMAS-based causal encounter model.

B. Model Specification and Characterization

This causal model includes one start component (s_1), eighteen data components ($d_1,...,d_{18}$), one end component (e_1), depicted in TABLE I, as well as seven function components ($h_1,h_2,h_3,h_4,h_5,h_6,h_7$) and one nested function components (h_1').

h_1 : Generate the variable speed for each involved aircraft in a scenario. In this block, the event that generates the motion state of each aircraft through randomly selecting the variables in spatial coordinates (x,y,z) . The data components d_1,d_2,d_3 provide several options for random selection in corresponding x,y,z axis. Considering the initial speed in the *x* axis as an example, the changes in the initial speed could be various, from several negative to several positive options, such as $(-v_x^n,...,-v_x^r,...,-v_x^2,-v_x^1,0,v_x^1,v_x^2,...,v_x^r,...,v_x^n)$, $v_x^1 < v_x^2 < ... < v_x^r < ... < v_x^n$. The extremum v_x^n is primarily based on the simulated aircraft performance.

h_2 : Represent a core pilot entity in the collision avoidance to provide probabilistic pilot response, and it connects with d_6 reserving the reaction time and d_7 containing the sense choices. In modelling aircraft response to RAs, the expectation is the pilot will begin the initial acceleration manoeuvre within five seconds ($0 < \Delta t \le 5$). In terms of practicality, each aircraft has three choices of sense selection $s_{t_{RA}}^i = (1,0,-1)$ corresponding to climb, maintain and descend. Coordination interrogations contain information about an aircraft's intended RA sense to resolve the encounter with the other TCAS-equipped intruder. If an intent message has been received, TCAS chooses the opposite sense from that selected by the other aircraft and communicated via the coordination interrogation. Therefore, for *Aircraft i* and *Aircraft j* which are involved in the same threat, their response combination $C_{t_{RA}}^{ij} = [C_{t_{RA}}^i, C_{t_{RA}}^j]$ can be $\{[1,-1],[1,0],[0,1],[0,-1],[-1,0],[-1,1]\}$.

h_3 : Select the neighbouring threats between which there would be an interrelationship that may lead to a new secondary conflict or even potential collision. The proximity relationship can be calculated based on the distance and time at the closest point of approach (CPA) of both threats.

h_4 : Screen the approaching aircraft. To improve the computational efficiency, only the approaching aircraft which are resolving their separate primary threat should be screened out for the secondary threat detection. The function components h_3 and h_4 are ingeniously designed like filters to narrow the expanded state space, which can play an important role in dealing with the complex multi-aircraft scenarios (e.g., flocks).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2018.2852324, IEEE Access

h_5 : Update the fight state. Operating this function component one time indicates that the aircraft fly to the next waypoints, until all threats are resolved or a new secondary threat is detected.

h_6 : Detect and estimate the domino threat. It aims to check whether a secondary encounter exists between the involved aircraft which have been in the process of resolving their own primary threat, and meanwhile determine whether the new secondary threat could be resolved or not. A domino conflict would deteriorate into an induced collision if the diameter and height (D_{cl} and H_{cl}) of the collision area of both aircraft overlap.

h_7 : Generate the optimal advisory in the possible future states for the multi-aircraft scenario. The strategies to determine the optimal advisories are: I. the measures that would not or less induce a secondary threat (negative domino effect); II. the non-altitude crossing sense due to the rules of aviation safe even if the altitude crossing sense provides greater separation; III. the amendments of both aircraft's trajectories to resolve a threat for the sake of fairness in a TCAS/TCAS encounter. And the priority of these policies is $I > II > III$.

TABLE I: Start, data and end components specification of the main model

Num.	Components	Description
s_1	Initial waypoints	input the original state information of involved aircraft
d_1	Vx	hold the uncertainty of initial velocity in *x* bearing
d_2	Vy	hold the uncertainty of initial velocity in *y* bearing
d_3	Vz	hold the uncertainty of initial velocity in *z* bearing
d_4	Control 1	take the subsidiary condition cooperating with h_1 to realize its control function
d_5	Next waypoints	keep the data of serious waypoints in the normal flight without conflict
d_6	Response delay	preserve the pilot's reaction time to the TCAS advisory (within 5s)
d_7	Possible reaction	contain the choices of sense selection (climb, maintain and descend)
d_8	Potential manoeuvre	summarize the pilot's potential manoeuvre
d_9	RA waypoints	record the RA states of aircraft involved in this scenario
d_{10}	CPA position-time	keep the CPA position and time of each threat
d_{11}	Neighbouring threats	record the RA neighbouring threats
d_{12}	Corresponding response	process the corresponding pilot's reaction to resolve the neighbouring threats
d_{13}	Approaching aircraft	indicate the approaching aircraft which are in the previous CA process
d_{14}	Time control	provide the sequence control of time for the discrete aircraft flight
d_{15}	Control 2	take the subsidiary condition cooperating with h_5 to realize its control function
d_{16}	Collision volume	provide the horizontal and vertical criteria of potential collision
d_{17}	Domino threat	keep the aircraft involved in a secondary conflict
d_{18}	Criteria	exhibit the rules to select the optimal advisories
e_1	Optimal advisory	output the optimal advisory for the multi-aircraft scenario

This nested function component h'_1 represents the process of carrying out the basic TCAS operations and integrates with h_1 to obtain the initial states of the involved aircraft, and h_2 to receive the impact of pilot behaviour. It includes one start component (s'_1), seven data components ($d'_1, d'_2, d'_3, d'_4, d'_5, d'_6, d'_7$), one end component (e'_1), depicted in TABLE II, as well as three function components ($h'_{1,1}, h'_{1,2}, h'_{1,3}$).

$h'_{1,1}$: Evaluate the SL of involved aircraft. Different SLs correspond to different TA and RA thresholds, and it is determined based on their flight altitudes [8].

$h'_{1,2}$: Detect the threat. The TA is issued when another aircraft approaches and a collision would emerge within 20-48s (variables are provided in d'_4) on account of the SL. It tries to draw the pilot's attention and calculates the CPA to inform Agent Predictive computation.

$h'_{1,3}$: Resolve the threat. With the communication of Agent Pilot response, TCAS issues the RA when a collision would emerge within 15-35s (variables are provided in d'_6) that is depending on the SL. In order to explore the possible future situations, this transition also serves to transmit RA waypoint information to inform Agent Predictive computation.

TABLE II: Start, data and end components specification of the sub-model "TCAS processor"

Num.	Components	Description
s'_1	Aircraft state information and potential manoeuvre	input the states of involved aircraft and the pilot's potential manoeuvre
d'_1	Sensitivity level	provide TA and RA criteria of different levels
d'_2	Control1	take the subsidiary condition cooperating with h'_1 to realize its control function
d'_3	Aircraft in SL	preserve the aircraft states with corresponding flight level
d'_4	$Time_{TA}$-$Distance_{TA}$	keep the time and distance criteria of TA
d'_5	Threat involved aircraft	indicate the states of aircraft involved in a threat
d'_6	$Time_{RA}$-$Distance_{RA}$	keep the time and distance criteria of RA
d'_7	Control2	take the subsidiary condition cooperating with h'_3 to realize its control function
e'_1	CoC and data output	output the RA waypoints and record the CPA information

V. RESULTS

Discrete event simulation can be conducted to evaluate the performance of TACS encounter models using the GMAS. Each simulation run corresponds to a path in the reachability graph of state space. As such, the performance optimisation of

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2018.2852324, IEEE Access

encounter models with a large number of decision variables requires a large number of simulation runs.

TABLE III provides the relevant parameter used in the different experiments to validate the feasibility of the GMAS-based causal encounter model. The diameter and height of the collision cylinders are twice as long as the horizontal and vertical sizes of the aircraft, respectively (i.e., $D_{cl} = 0.044NM$ and $H_{cl} = 78.44ft$) [42]. The computer used for this simulation is a T450 laptop with a 2.6 GHz Intel i7 processor and 8GB of RAM, which is enough for the memory requirements of the algorithmic operations and simulation.

TABLE III: Parameter values for the scenarios

TCAS Equipment	**Kind of reaction**	**Detection range (NM)**	**RA acceleration (g)**	**Subsequent RA acceleration (g)**
TCAS II 7.1	Pilot react	40	0.25/-0.25	0.35/-0.35
Initial primary pilot delay (s)	**Subsequent pilot delay (s)**	**Horizontal size (m)**	**Vertical size (m)**	**SL**
5	3	40.74	11.95	6

For a closing target to be declared an intruder, the range test is based on the time to CPA. Because the SL is equal to 6, a conflict would be detected if the time to CPA is less than as 48s based on the time thresholds shown in [43].

To verify the feasibility of the proposed model, several simulation experiments have been performed on a complex scenario of five aircraft, *Aircraft 1, Aircraft 2, Aircraft 3, Aircraft 4,* and *Aircraft 5*. At 9:15:45, the state $1'(Aircraft, x, y, z, vx, vy, vz)$ of the five aircraft separately are: *1'(1,19.36,20.58,19000.00,0.20,0,0),1'(2,14.49,17.00,17850.00,0.18,0,0),1'(3,*34.27*,14.59,18660.00,-0.1,0.1,-10),1'(4,13.44,22.87,16730.00,0.08,-0.08,15),1'(5,33.88,17.03,17730.00,-0.2,0,0).* The fully TCAS-equipped aircraft are given with two initial predicted encounters, *Conflict 1* between *Aircraft 1* and *Aircraft 2* while *Conflict 2* between *Aircraft 3* and *Aircraft 4*.

A. State Space Analysis

The simulation experiments are carried out to evaluate the efficacy of the proposed causal encounter model for the improvement of the TCAS avoidance performance. The computational results represent feasible collision-free manoeuvres for multiple aircraft that are modelled with detailed dynamics, and the optimal advisories would be selected. The reachable states of this five-aircraft scenario generated by our causal encounter model are displayed in Fig. 4. Note that the illustrated state space in this section do not take the wind influence and probable pilot reaction time into consideration, and it would be propitious to the understanding of the process to obtain the optimal solutions.

For *Aircraft 2* and *Aircraft 3* which are involved in *Conflict 1*, their response combination $C_{t_{RA1}}^{23} = [C_{t_{RA1}}^{2}, C_{t_{RA1}}^{3}]$ can be $\{[1,-1],[1,0],[0,1],[0,-1],[-1,0],[-1,1]\}$, as shown in the first level of this reachability tree. Immediately afterwards, *Aircraft 4* and *Aircraft 5* try to resolve the nearby *Conflict 2* and similarly their possible reaction $C_{t_{RA2}}^{45} = [C_{t_{RA2}}^{4}, C_{t_{RA2}}^{5}]$ also has six options. In view of the overall situation, there would be $6 \times 6 = 36$ states in Level 2 for the four-aircraft response combination $C_{t_{RA}}^{2345} = [C_{t_{RA1}}^{2}, C_{t_{RA1}}^{3}, C_{t_{RA2}}^{4}, C_{t_{RA2}}^{5}]$. The two nearby threats may produce domino effects that would initiate a new secondary encounter or even a potential collision. In addition, the amending aircraft in the RA process also may encounter neighbouring vehicle, e.g., *Aircraft 1* in this scenario. The tests of domino effect are implemented on the involved aircraft, and these response combinations ($[1,-1,1,-1]$ $[1,-1,1,0]$ $[1,-1,-1,1]$ $[1,0,1,0]$ $[0,-1,1,0]$ $[0,1,0,1]$ $[-1,0,0,1]$ $[-1,0,-1,1]$ $[-1,1,1,-1]$ $[-1,1,1,0]$ $[-1,1,0,-1]$ $[-1,1,0,1]$ $[-1,1,-1,0]$ $[-1,1,-1,1]$) containing domino conflicts have been detected.

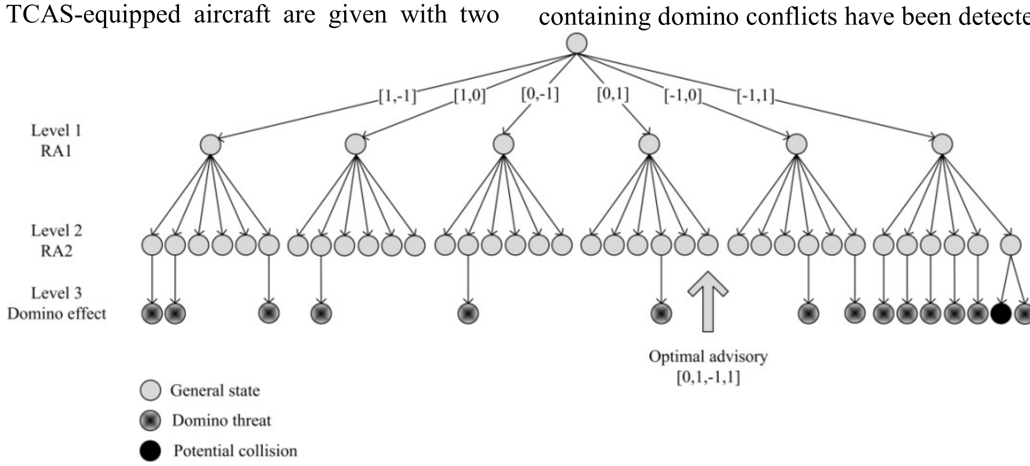

Fig. 4: State space of this five-aircraft scenario.

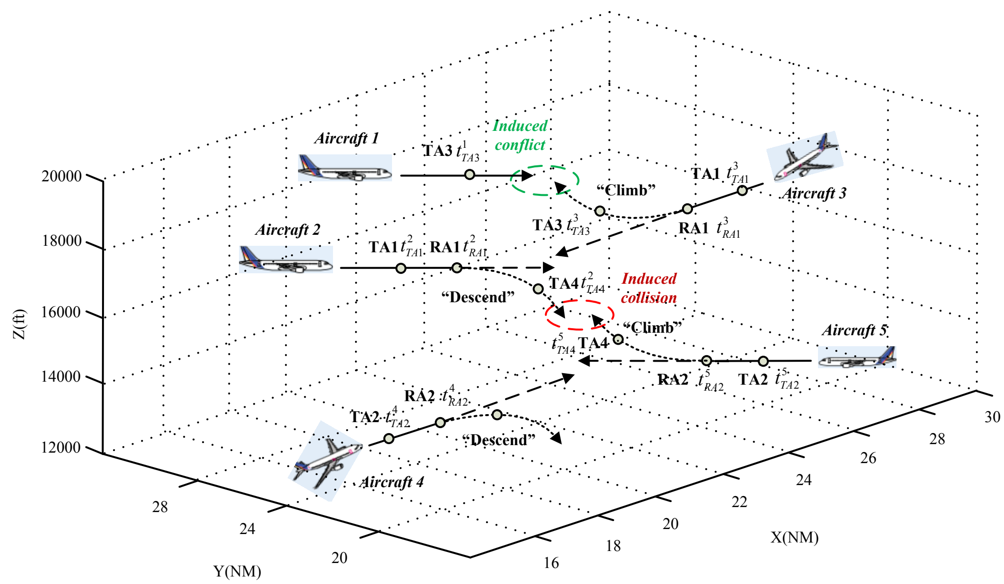

Fig. 5: Five-aircraft collision scenario.

Thus the remaining states without any domino effect are preferred. Based on the three policies ($I > II > III$), the optimal advisory for the multi-aircraft scenario can be generated by the function component h_7 , shown in the sub-section 'Model Specification and Characterization'. In this five-aircraft collision scenario, $[0,1,-1,1]$ corresponding to the response combination $[\text{Maintain}, \text{Climb}, \text{Descend}, \text{Climb}]$ is the optimal strategy based on the state prediction.

The following represents the simulation results of the specific scenario in which the TCAS logic is deterministic and the pilots rigorously follow the advisories. Note that $[\text{Descend}, \text{Climb}, \text{Descend}, \text{Climb}]$ is recommended for the initial TCAS logic because of the non-altitude crossing sense. When the response combination $[-1,1,-1,1]$ corresponding to $[\text{Descend}, \text{Climb}, \text{Descend}, \text{Climb}]$ is utilized, a new secondary conflict is induced between *Aircraft 3* and *Aircraft 1* which can be resolved, and *Aircraft 2* would encounter *Aircraft 5* while this emerging conflict may deteriorate into a collision, illustrated in Fig. 5.

In Fig. 5, variable $t_{TA}^i (i = 1,2,3,4,5)$ is the TA emergence time while $t_{RA}^i (i = 1,2,3,4,5)$ is used for the RA. For *Conflict 1*, the TA is issued at 9:16:11 and the RA is issued to ask the pilots in both aircraft to resolve this encounter at 9:16:26. For *Conflict 2*, the TA emerges at 9:16:13, and the RA is issued at 9:16:28. TABLE IV represents the waypoints from the time of 9:16:31 when *Aircraft 2* and *Aircraft 3* begin to amend their trajectories. At 9:16:33 an emergent encounter between *Aircraft 2* and *Aircraft 5* as the domino effect appears, and unfortunately the left time is not enough for the pilot reaction and trajectory amendment. Therefore, at 9:16:36 there would be a collision between them: the horizontal distance is $\sqrt{(23.67-23.68)^2 + (17.00-17.03)^2} = 0.032NM < D_{cl}(0.044NM)$ while the altitude interval is $|17824.00 - 17749.49| = 74.51 ft < H_{cl}(78.44 ft)$. The secondary conflict between *Aircraft 1* and *Aircraft 3* is detected at 9:16:35 and their distance is sufficient for the modification of trajectories to avoid collision.

TABLE IV: Partial waypoints of the four aircraft

Time	Aircraft	X(Nm)	Y(Nm)	Z(ft)
9:16:31	*Aircraft 1*	28.56	20.58	19000.00
9:16:31	*Aircraft 2*	22.77	17.00	17850.00
9:16:31	*Aircraft 3*	29.77	19.19	18200.00
9:16:31	*Aircraft 4*	17.12	19.19	17430.00
9:16:31	*Aircraft 5*	24.68	17.03	17730.00
9:16:32	*Aircraft 1*	28.76	20.58	19000.00
9:16:32	*Aircraft 2*	22.95	17.00	17845.00
9:16:32	*Aircraft 3*	29.67	19.29	18195.00
9:16:32	*Aircraft 4*	17.20	19.11	17440.00
9:16:32	*Aircraft 5*	24.48	17.03	17730.00
9:16:33	*Aircraft 1*	28.96	20.58	19000.00
9:16:33	*Aircraft 2*	23.13	17.00	17840.00
9:16:33	*Aircraft 3*	29.57	19.39	18190.00
9:16:33	*Aircraft 4*	17.28	19.03	17450.00
9:16:33	*Aircraft 5*	24.28	17.03	17730.00
9:16:34	*Aircraft 1*	29.16	20.58	19000.00
9:16:34	*Aircraft 2*	23.31	17.00	17835.00
9:16:34	*Aircraft 3*	29.47	19.49	18185.00
9:16:34	*Aircraft 4*	17.36	18.95	17453.83
9:16:34	*Aircraft 5*	24.08	17.03	17736.17
9:16:35	*Aircraft 1*	29.36	20.58	19000.00
9:16:35	*Aircraft 2*	23.49	17.00	17830.00

9:16:35	*Aircraft 3*	29.37	19.59	18180.00
9:16:35	*Aircraft 4*	17.44	18.87	17457.66
9:16:35	*Aircraft 5*	23.88	17.03	17742.34
9:16:36	*Aircraft 1*	29.38	20.58	19000.00
9:16:36	*Aircraft 2*	23.67	17.00	17824.00
9:16:36	*Aircraft 3*	29.27	19.69	18175.00
9:16:36	*Aircraft 4*	17.52	18.79	17461.49
9:16:36	*Aircraft 5*	23.68	17.03	17749.49

B. Further Investigations

The state space can be used to not just obtain the best or optimal solutions to a problem, but analyse the system behaviour to ensure real optimal configurations when the complete states are explored. It facilitates the design and validation of systems, e.g., TCAS in this research, assessment of strategies (TCAS advisories), and examination of the decision making process.

In the simulation model, the set of data components is $D=\{d_1,d_2,...,d_j,...,d_a\}, j=1,2,...,a$, the set of function components is $H=\{h_1,h_2,...,h_v,...,h_m\}, v=1,2,...,m$. The set of reachable states $K(G,M_0)=\{m_0,m_1,...,m_{k-1}\}$ is defined as the collection of all state identifiers that can be reached from the initial state identification M_0 according to the activation rules. They are distributed over the different layers ($0,1,...,l-1$) of reachability tree, and $k=|K(G,M_0)|$ indicates the number of timing states. The corresponding function components, activating probability and activating consuming time of the generated timing states $\{m_1,...,m_{k-1}\}$ based on m_0 are respectively set as $h_1,...,h_{k-1}$, $r_1,...,r_{k-1}$, and $t_1,...,t_{k-1}$, and therein there may be the same function components of $h_s (s=1,2,...,k-1)$.

1) The activation frequency of function component

The activation frequency of a specific function component is defined as the rate between its activating times and the all activations to generate the whole state space. That the value is higher illustrates the greater possibility of the function component to be activated, which is the more important process of the complex system. The calculation of this factor is convenient for system developers to improve and focus on the process which has higher impact on the system operations. In GMAS it can be graphically displayed that the abscissa shows the function components and the ordinate shows the values of their activation frequencies.

Assume $z(h_v)$ as the activation frequency of function component h_v and it can be computed:

$$z(h_v)=\frac{\sum_{h_s=h_v} r_s}{l-1}, \quad v=1,2,...,m; s=1,2,...,k-1 \tag{7}$$

First in the all function components corresponding to each generated timing state, it automatically check and search out the same ones $h_s (s=1,...,k-1)$ which are equal to h_v ; then sum the corresponding activating probability r_s of h_s ; finally, calculate the proportion of activation in all reachable timing states.

Fig. 6 depicts the activation frequency of function components in the GMAS-based causal encounter model and sub-model h_1' with the test case of different initial states (averaged over 125 Runs). Fig. 5(a) shows that the activation frequency of h_1' is the highest, validating the TCAS processor as the core of this encounter model. Then the activation frequency of function components is decreasing due to not all neighbouring threats would initiate a new secondary conflict or even potential collision. The original TCAS logic could resolve the most encounters, thus the function components in Agent Predictive computation may not be activated in most situations. It has the similar trend to the three function components ($h_{1,1}', h_{1,2}', h_{1,3}'$) of h_1' illustrated in Fig. 5(b), because not all detected conflicts require issuing RAs.

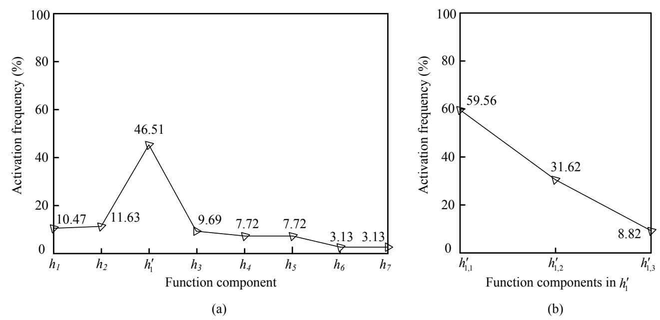

Fig. 6: Activation frequency of function components in the causal encounter model (a) and sub-model h_1' (b).

2) The average consuming time of function component

The average consuming time of a specific function component is defined as the quotient between the sum of its activating probability multiply by consuming time and the possible activation number. That the value is higher indicates the more average time it takes for each activation during the simulation, i.e., the more time-consuming process of the complex system. The calculation of this factor is convenient for system developers to analyze and optimize the process which has great influence on the system's working efficiency. In GMAS it can be graphically displayed that the abscissa shows the function components and the ordinate shows the values of their average consuming time.

Assume $t(h_v)$ as the average consuming time of function component h_v and it can be computed:

$$t(h_v)=\frac{\sum_{h_s=h_v} r_s t_s}{\sum_{h_s=h_v} 1}, \quad v=1,...,m; s=1,...,k-1 \tag{8}$$

First in the all function components corresponding to each generated timing state, it automatically checks and search out the same ones $h_s (s=1,...,k-1)$ which are equal to h_v ; then sum the product of activating probability r_s and its corresponding consuming time t_s; finally, divide the sum result by the possible activating times.

Fig. 7 shows the average consuming time of function components in the GMAS-based causal encounter model and sub-model h_1' with the test case of different initial states (averaged over 125 Runs). The total average consuming time is 4.53s for the five-aircraft scenarios. Fig. 6(a) illustrates that the highest average consuming time is the nested function component h_1' which contains the complete TCAS operational process. The second one is h_5 which updates the fight state with the interval of 1s, until all threats are resolved or a new secondary threat is detected; its average consuming time can be adjusted through changing the updating intervals (e.g., turn 1s to 2s). The strategy to select the optimal advisory in the generated state space is evident, thus the average consuming time of h_7 is relatively low. In Fig. 6(b), it represents the three function components ($h_{1,1}', h_{1,2}', h_{1,3}'$) of h_1' possessing a growing trend of the average consuming time, that corresponds to the operational complexity of their own underlying logic.

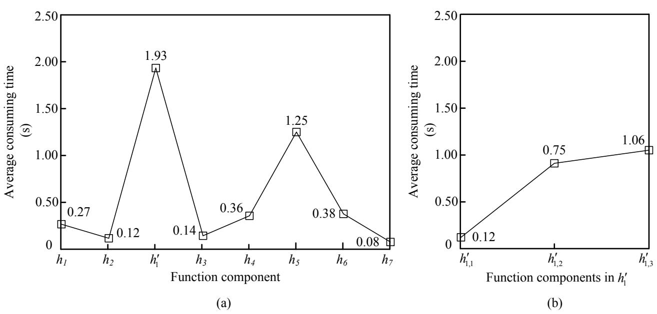

Fig. 7: Average consuming time of function components in the causal encounter model (a) and sub-model h_1' (b)

Obviously, it is challenging for the proposed encounter model to handle massive aircraft in the entire airspace, while TCAS as the last-resort is a tactical system to focus on the regional airspace in which several aircraft are involved. Based on the radar data from FAA and Department of Defense sites throughout the United States [44], over 95% of the multi-aircraft scenarios involve three aircraft, but only one involves seven aircraft in the total identified 3803 such multi-threat encounters. The total average consuming time with different number of aircraft are recorded in TABLE V. They are all in a reasonable range, smaller than 15s that is normally the minimum interval between TAs and RAs. Thus, the proposed causal model is competent for safety assessment and advisory optimization.

TABLE V: Total average consuming time of the causal model

Aircraft amount	Time Taken
3	1.37s
4	2.06s
5	4.53s
6	7.99s
7	13.81s

3) The utilization rate of data component

The utilization rate of a specific data component is defined as its frequency of timing state changes in the all activations to generate the whole state space. In the proposed model, the set of function components $H_{v,j} = \{h_{v_1,j},...,h_{v_x,j},...,h_{v_m,j}\},(v_x = 1,...,m)$ that are associated with the data component $d_j(j=1,2,...,a)$ through the input/output link components is known. That the value is higher represents the more important element that affects the performance of complex systems. In GMAS it can be graphically displayed that the abscissa shows the data components and the ordinate shows the values of their utilization rates.

Assume $\Delta(d_j)$ as the utilization rate of data component d_j and it can be computed:

$$\Delta(d_j) = \sum z(h_{v_x,j}) \Big/ \sum_{j=1}^{a}\sum z(h_{v_x,j}) = \sum \frac{\sum_{h_s=h_{v_x,j}} r_s}{l-1} \Big/ \sum_{j=1}^{a}\left(\sum \frac{\sum_{h_s=h_{v_x,j}} r_s}{l-1}\right) \quad (9)$$

$v_x = 1,2,...,m; s = 1,2,...,k-1; j = 1,2,...,a$

First search out the function components $h_{v_x,j}(v_x = 1,...,m)$ associated with the data component d_j; then sum of the activation frequency of connected function components; finally, normalize the generated $\sum z(h_{v_x,j})$ of all data components to obtain the utilization rates.

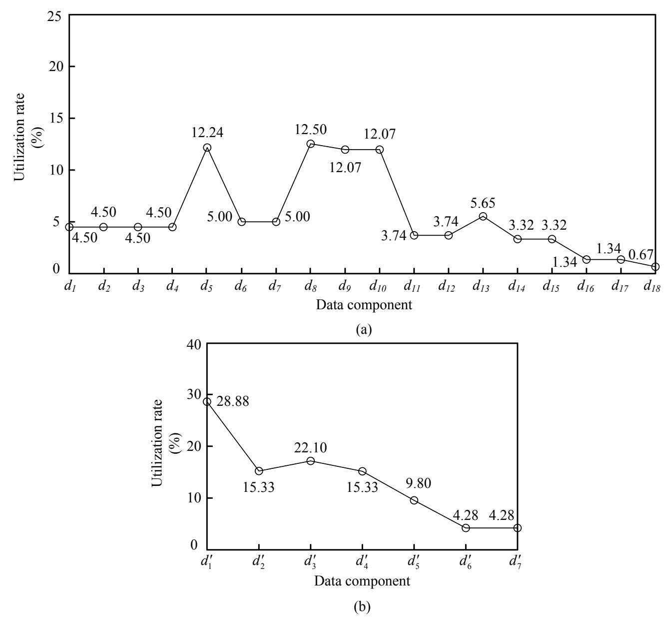

Fig. 8: Utilization rate of data components in the causal encounter model (a) and sub-model h_1' (b)

Fig. 8 depicts the utilization rate of data components in the GMAS-based causal encounter model and sub-model h_1' with the test case of different initial states (averaged over 125 Runs). Fig. 7(a) indicates that d_5, d_8, d_9, d_{10} own relatively high utilization rate, as the more important elements to affect the improved TCAS performance. They have the common characteristic connecting different function components like a bridge to transfer data between correlative agents/sub-models e.g., d_5 preserves the pilot's potential manoeuvre which would be sent to h_1' "TCAS processor" as the input. In Fig. 7(b), the utilization rate of d_1' is significantly greater than the other data components, because it controls the time or dimension thresholds for TA and RA issuance in different flight levels. The trend of utilization rate is decreasing due to the incidence of unnecessary TA/ RA alerts, which means that the approaching aircraft may not encounter each other or the detected conflict may not require a resolution measure.

VI. CONCLUSION

It is recommended to exploit the advantages of a DES approach for searching efficient solutions in the performance of operations, attributing to their capability of generating feasible solutions under the support of a quantitative analysis. The behaviour of DES can be represented by discrete state variables and is governed by asynchronous and instantaneous incidences, i.e., the activation of function component in GMAS, which are solely responsible for the state changes. It allows to generate a series of events or activities, to reach certain final states while minimizing the cost or risk. The main contributions of this paper are as follows:

- Introduce the GMAS as a platform for modelling DES with simulating the system's behaviours, and records its evolution process (i.e., state space) to resolve optimization problem. It transforms a search-based issue in the reachability tree of finding the optimal or near-optimal sequence of function component activations from some initial state to the goal state.
- Propose a causal encounter model of TCAS operations in the GMAS formalism for safety assessment and advisory optimization. Based on the detailed model specification and explanation, the model logic process is explicit. The implemented model not only provides a better comprehension of the potential collision occurrences by representing the cause-effect relationship of each action, but also aids the involved aircraft to make a cooperative and optimal option.
- Summarize the simulation results of a complex multi-aircraft scenario, and conduct further analysis to be a paradigm offering some significant analytical bite. Consequently, the quantitative measurement experiments are carried out to validate the feasibility and effectiveness of the GMAS-based encounter model.

In addition, the clearly calculated and recorded discrete waypoints can be directly used in the analysis of system performance, making for the advantage of expansibility. To accomplish the fundamental purpose of our research, the next step is to construct various models in different areas using GMAS for examining the performance of different system configurations and the operating procedures for complex logistic.

ACKNOWLEDGMENT

The author would also like to acknowledge the financial support from the National Natural Science Foundation of China (grant number 71601181), the Hunan Provincial Natural Science Foundation of China (grant number 2017JJ3357), and the National Defense Science and Technology Project Fund (grant number 3101175).

REFERENCES

[1] Wang P and Cai KY. Supervisory control of discrete event systems with state-dependent controllability. Int J Syst Sci 2009; 40(4): 357-366.

[2] Schmidt KW and Breindl C. A framework for state attraction of discrete event systems under partial observation. Inform Sci 2014, 281: 265-280.

[3] Shu S and Lin F. Generalized detectability for discrete event systems. Syst control lett 2011, 60(5): 310-317.

[4] Wainer GA. Discrete-event modeling and simulation: a practitioner's approach. CRC Press, 2009.

[5] Cassandras CG, Lafortune S. Introduction to discrete event systems. Springer Science & Business Media, 2008.

[6] Bakolas E and Saleh JH. Augmenting defense-in-depth with the concepts of observability and diagnosability from control theory and discrete event systems. Reliab Eng Syst Safe 2010, 96(1):184-93.

[7] Tang J and Zhu F. Simulation modelling of traffic collision avoidance system with wind disturbance. IEEE Aero El Sys Mag, 2018, 4: 36-45.

[8] US Department of Transportation. Introduction to TCAS II version 7.1. Department of Transportation, Federal Aviation Administration, 2011.

[9] Zhang J, Khalgui M, Li Z, et al. Reconfigurable coordination of distributed discrete event control systems. IEEE T Contr Syst T 2015, 23(1): 323-330.

[10] Makela M. Maria: Modular reachability analyser for algebraic system nets. In: International Conference on Application and Theory of Petri Nets, Springer Berlin Heidelberg, 2002, pp. 434-444.

[11] Davidrajuh R and Lin B. Exploring airport traffic capability using Petri net based model. Expert Syst Appl 2011, 38(9): 10923-10931.

[12] Ediger D, Jiang K, Riedy EJ, et al. Graphct: Multithreaded algorithms for massive graph analysis. IEEE T Paral Distr 2013, 24(11): 2220-2229.

[13] Brosig F, Meier P, Becker S, et al. Quantitative evaluation of model-driven performance analysis and simulation of component-based architectures. IEEE T Softw Eng 2015, 41(2): 157-175.

[14] Kang D and Choi BK. The extended activity cycle diagram and its generality. Simul Model Pract Th 2011, 19(2): 785-800.

[15] Gray MA. Discrete event simulation: A review of SimEvents. Comput Sci Eng 2007, 9(6): 62-66.

[16] Buss A and Blais C. Composability and component-based discrete event simulation. In: Proceedings of the 39th conference on Winter simulation, Washington, DC, USA, 2007, pp. 694-702.

[17] Westergaard M, Evangelista S and Kristensen L M. ASAP: an extensible platform for state space analysis. In: International Conference on Applications and Theory of Petri Nets, Springer Berlin Heidelberg, 2009, pp. 303-312.

[18] Jensen K and Kristensen LM. Coloured Petri nets: modelling and validation of concurrent systems. Berlin, Germany: Springer, 2009.

[19] Zhu X, vanden Broucke S, Zhu G, et al. Enabling flexible location-aware business process modeling and execution. Decis Support Syst 2016, 83: 1-9.

[20] McEneaney JE. Simulation-Based Evaluation of Learning Sequences for Instructional Technologies. Instructional Sci 2016, 44(1): 87-106.

[21] Baruwa OT, Piera MA and Guasch A. TIMSPAT-Reachability graph search-based optimization tool for colored Petri net-based scheduling. Comput Ind Eng 2016, 101: 372-390.

[22] Baruwa OT, Piera MA and Guasch A. Deadlock-free scheduling method for flexible manufacturing systems based on timed colored Petri nets and anytime heuristic search. IEEE T Syst Man Cybern Syst 2015, 45(5): 831-846.

[23] Baruwa OT and Piera MA. A coloured Petri net-based hybrid heuristic search approach to simultaneous scheduling of machines and automated guided vehicles. Int J Prod Res 2015, 54 (16): 4773-4792.

[24] Zeigler BP, Praehofer H and Kim TG. Theory of modeling and simulation: integrating discrete event and continuous complex dynamic systems. Academic press, 2000.

[25] Zeigler BP and Chi S. Symbolic discrete event system specification. IEEE T Syst Man Cy-S, 1992, 22(6): 1428-1443.

[26] Huang D, Sarjoughian H S, Wang W, et al. Simulation of semiconductor manufacturing supply-chain systems with DEVS, MPC, and KIB. IEEE T Semiconduct M, 2009, 22(1): 164-174.

[27] Schulz S, Ewing TC and Rozenblit JW. Discrete event system specification (DEVS) and statemate statecharts equivalence for embedded systems modeling., Proceedings. Seventh IEEE International Conference and Workshop on the Engineering of Computer Based Systems, 2000: 308-316.

[28] Gupta P. Hardware-software codesign. IEEE Potentials, 2001, 20(5): 31-32.

[29] Castonguay P and Wainer G. Aircraft evacuation DEVS implementation & visualization. Proceedings of the 2009 Spring Simulation Multi-conference, Article No. 144, 2009: 1-8.

[30] Jafer S and Mi W. Comparative Study of Aircraft Boarding Strategies Using Cellular Discrete Event Simulation. Aerospace, 2017, 4(57): 1-22.

[31] Piera MA, Buil R and Ginters E. State space analysis for model plausibility validation in multi-agent system simulation of urban policies. J Simul 2016, 10(3): 216-226.

[32] Kim HJ, Lee JH and Lee TE. Time-feasible reachability tree for noncyclic scheduling of timed Petri nets. IEEE T Autom Sci Eng 2015, 12(3): 1007-1016.

[33] Şucan IA and Kavraki LE. A sampling-based tree planner for systems with complex dynamics. IEEE T Robot 2012, 28(1): 116-131.

[34] Kochenderfer MJ and Chryssanthacopoulos JP. Collision avoidance using partially controlled Markov decision processes. International Conference on Agents and Artificial Intelligence. Springer, Berlin, Heidelberg, 2011: 86-100.

[35] Lime D and Roux OH. Model checking of time Petri nets using the state class timed automaton. Discrete Event Dynamic Systems, 2006, 16(2): 179-205.

[36] Temizer S, Kochenderfer M, Kaelbling L, et al. Collision avoidance for unmanned aircraft using Markov decision processes. AIAA guidance, navigation, and control conference, 2010: 1-21.

[37] Ng K M, Reaz M B I, Ali M A M. A review on the applications of Petri nets in modeling, analysis, and control of urban traffic. IEEE Transactions on Intelligent Transportation Systems, 2013, 14(2): 858-870.

[38] Tang J, Piera MA and Baruwa OT. A discrete-event modeling approach for the analysis of TCAS-induced collisions with different pilot response times. P I Mech Eng G-J Aer 2015, 229(13): 2416-2428.

[39] Tanyer I, Tatlicioglu E and Zergeroglu E. Model reference tracking control of an aircraft: a robust adaptive approach. Int J Syst Sci 2017, 48(7): 1428-1437.

[40] Jun T, Piera MA and Ruiz S. A causal model to explore the ACAS induced collisions. P I Mech Eng G-J Aer 2014, 228(10): 1735-1748.

[41] Tang J, Piera MA and Guasch T. Coloured Petri net-based traffic collision avoidance system encounter model for the analysis of potential induced collisions. Transport Res C-Emer 2016, 67: 357-377.

[42] Eurocontrol. The EUR RVSM pre-implementation safety case. 14 August 2001, version 2.0. Document RVSM 691.

[43] DoT, 2011. Introduction to TCAS II Version 7.1. Department of Transportation, Federal Aviation Administration, USA.

[44] Billingsley TB, Espindle LP and Griffith JD. TCAS multiple threat encounter analysis. Project Report ATC-359, 2009.