Failsafe system design for autonomous refueling of multiple aerial vehicles using state tree structures

Rui Wang and W. M. Wonham

Systems Control Group, Department of Electrical and Computer Engineering, University of Toronto

1. Problem description and research objectives

The lack of aerial refueling capabilities limits the current use of unmanned aerial vehicles. Autonomous aerial refueling is a key technology to extend the flight endurance and range of unmanned aerial vehicles by refueling them in flight. More and more institutions, for example, NASA, DARPA, and US Air Force, start to evaluate technologies that could be used for autonomous aerial refueling. The autonomous refueling of multiple aerial vehicles is vulnerable to various failures of each vehicle and resource conflicts between vehicles. The vehicle's autonomous receivers and external tankers may cause possible failures. Because of frequent interactions between the on-board receiver controller and external tanker controller, command conflicts may occur and endanger the autonomous aerial refueling mission. Other system failures of each vehicle such as probe damage and navigation error alse effect the mission. The refueling process is divided into different sub-phases. Each sub-phase is open to limited vehicles. Dangerous flight maneuvers may be executed when unexpected failures or resource conflicts happen. The successful autonomous aerial refueling depends on communications between different receiver aircrafts' on-board controller and the tanker aircraft's controller. Therefore, it needs a high-level logic controller that coordinates all these components and produces the desired safe commands in the presence of faults, failures, and conflicts. The synchronous product of these components is a complex and large system. Before obtaining the high-level logic controller, it requires various descriptions to model the strategies and decisions under different failures and conflicts. This project will investigate the failsafe logic design of every unmanned aerial vehicle and the non-conflict logic design of neighboring unmanned aerial vehicles in the mission of autonomous aerial refueling. It will study high-level non-conflicting problem-solving strategies from the perspective of autonomous aerial refueling for multiple aerial vehicles. The high-level logic controllers for this project can be applied to similar problems such as autonomous charging of drones and unmanned vehicles.

Fig. 1 Aerial refueling for multiple aerial vehicles

2. Mode discretization and decomposition of the autonomous aerial refueling process for multiple aerial vehicles

In the autonomous aerial refueling mission, the receiver is required to take up formation and maneuver around the tanker after first rendezvousing with the tanker. The primary low-level task required for every receiver across the autonomous aerial refueling mission is relative position control to the tanker. The areas around the tanker are divided into two parts, task space, and withdrawal space [1]. The task space contains three subareas that are observation area, astern area, reform area. The withdrawal space is the air space around the task space. When the autonomous aerial refueling task is achieved or failed, the receiver will enter into the withdrawal space. When the receiver's health allows it to perform the autonomous aerial refueling task, the receiver will enter into the task space from the withdrawal space. Corresponding to three subareas in the task space, the task phase contains three sub-phases, namely Joining sub-phase, Refueling sub-phase, and Reform sub-phase. In the withdrawal space, there are three modes, namely standby mode, base landing, and emergency landing mode.

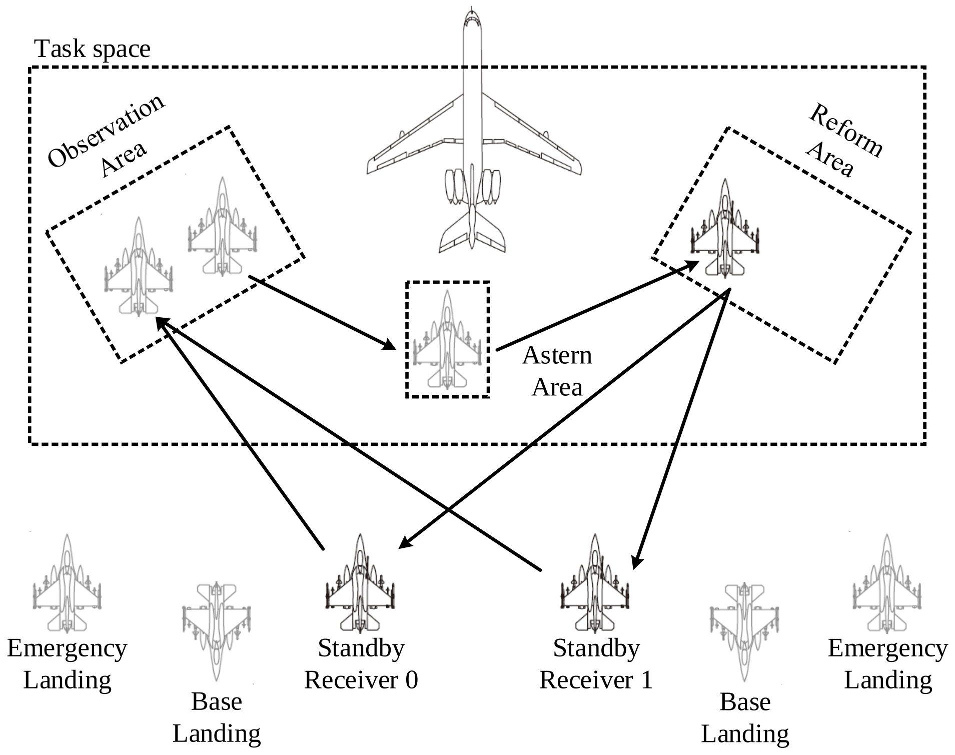


Fig. 2 Autonomous aerial refueling procedure for multi-vehicles

Table 1 Modes of the withdrawal phase

Standby (STANDBY) mode	The receiver is waiting in its formation for the next instruction. In this mode, the receiver can carry on autonomous aerial refueling tasks.
Base landing (RTL) mode	The receiver returns to the nearest airbase to land, which may result from its subsystem failures. In this mode, the receiver may lose the ability to carry on AAR tasks but still keep the ability to return to base.
Emergency landing (EL) mode	The receiver makes an emergency landing, which may result from its subsystem failures. In this mode, the receiver may lose the ability to return to base and can only make an emergency landing.

Table 2 Sub-phases of the task phase

Joining sub-phase	The receiver maneuvers from the STANDBY mode to the observation area. Then, the receiver stays in the observation area and waits for the command given by the tanker to allow the receiver to fly to the astern area and connect its probe with the drogue.
Refueling sub-phase	The receiver flies to the astern area, initiates the connection between the probe and the tanker’s drogue. Then, the receiver keeps the position for connecting the probe with the drogue. After a successful capture, the receiver and the tanker keep relatively stationary to transfer the fuel. When the fuel transfer is finished, the

	receiver should still keep connected with the tanker and wait for the command given by the tanker to allow the receiver to disconnect from the drogue.
Reforming sub-phase	The receiver disconnects with the tanker, flies to the reform area and keeps the position in the reform area. Then, the receiver leaves the reform area and returns to the STANDBY mode in the withdrawal space.

3 Formal description of generalized aerial refueling processes and logic design of associated refueling specifications

Autonomous aerial refueling is a dynamic process. In general, the successful process of autonomous aerial refueling is the same for every receiver. The first step in generating a high-level logic controller is to describe the logic transitions of the aerial refueling process. Thus, how to describe the logic transitions of the aerial refueling process using formal methods for each receiver is one of the project goals. Safety issues of every receiver connect to its subsystems such as sensor subsystems. These safety issues affect the failure and health behavior of every receiver directly, and they should also be described by formal methods. Besides the safety issues of each receiver, a possible conflict of using limited sub-phases between neighboring receivers is another influencing factor of the safety in the autonomous aerial refueling operations for multiple receivers. The above influencing factors of safety need appropriate strategies to remove their negative influences. The strategies ensure the safe and stable operation of autonomous aerial refueling for multiple receivers. Hence, how to design the strategies is one of the goals of the project.

3.1 Formal description of the autopilot for every receiver

Autopilot is the central logic control unit of a receiver. It is responsible for recording the task procedures of autonomous aerial refueling. In this project, *state tree* and *holons* are used to achieve the formal description of each autopilot. The state tree of autopilot describes static state dependency. Each sub-phase of the task phase and each mode of the withdrawal phase is abstracted into several states. In the design, autopilot is an OR superstate with three simple states (STANDBY, RTL, and EL) and three OR superstates (Joining, Refueling, and Reforming).

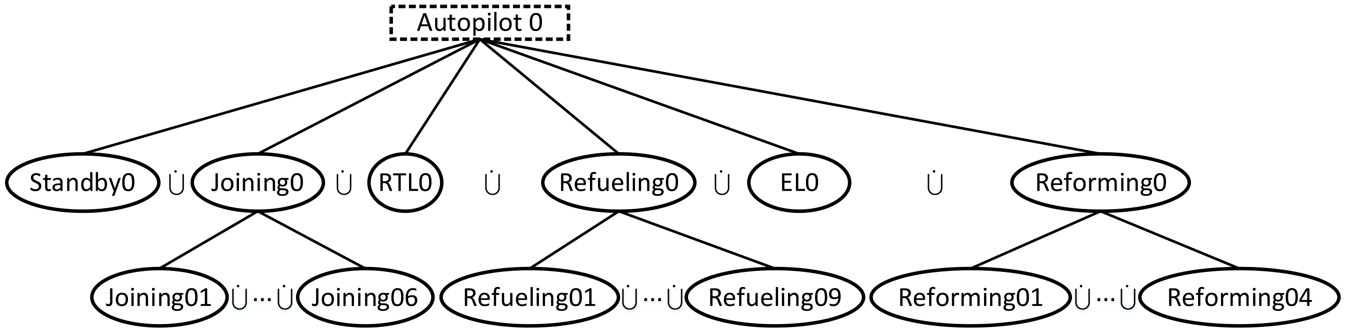


Fig 3. The state tree of Receiver 0's autopilot

The dynamical behavior, or transitions, of autopilot is modeled by holons. An event drives a transition. Hence, there is an event set of the autopilot model. The event set can be divided into two subsets that are controllable event set and uncontrollable event set. The controllable (resp. uncontrollable) event set is composed of all events labeled with odd (resp. even) numbers. Let's take Receiver 0's autopilot as an example. The abstracted holon of Receiver 0's autopilot is designed, as shown in Fig. 4.

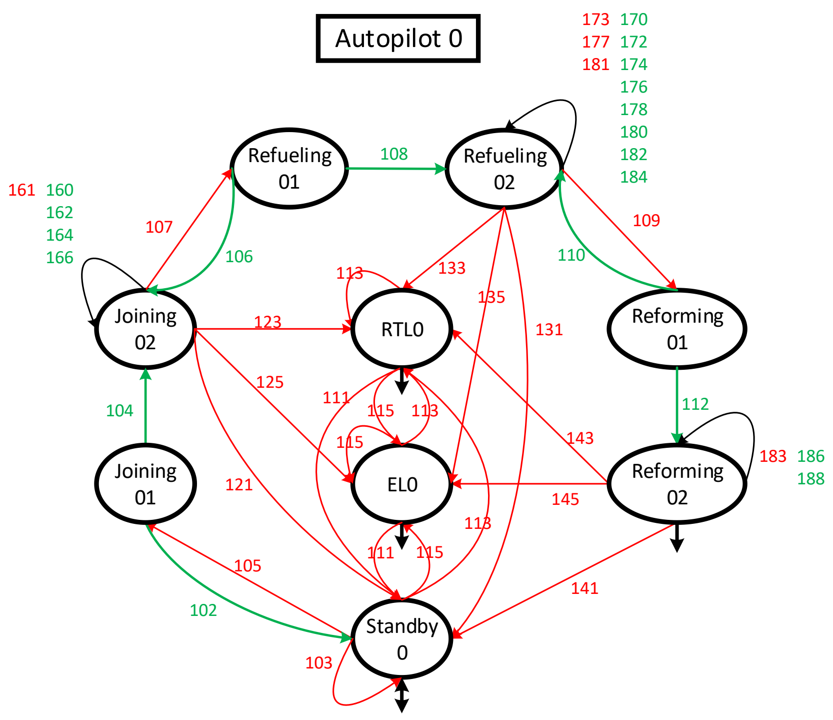


Fig 4. The holon of Receiver 0's autopilot

Table 3 Event definitions for Receiver 0's autopilot

Event label	Description
102	The event 105 fails, i.e., Receiver 0 does not enter into the observation area
103	Control Receiver 0 to stay in its STANDBY position
104	The event 105 succeeds, i.e., Receiver 0 has arrived at the observation area
105	Control Receiver 0 to fly from its STANDBY position to the observation area
106	The event 107 fails, i.e., Receiver 0 does not enter into the astern area
107	Control Receiver 0 to fly from the observation area to the astern area.
108	The event 107 succeeds, i.e., Receiver 0 has arrived at the astern area
109	Control Receiver 0 to fly from the astern area to the reform area.
110	The event 109 fails, i.e., Receiver 0 does not enter into the reform area
112	The event 109 succeeds, i.e., Receiver 0 has arrived at the reform area
111 (121, 131, 141)	Control Receiver 0 to fly to its STANDBY position from EL and RTL mode (or from the observation area, the astern area, the reform area)
113 (123, 133, 143)	Control Receiver 0 to stay in RTL mode (or from the observation area, the astern area, the reform area)
115 (125, 135, 145)	Control Receiver 0 to stay in EL mode (or from the observation area, the astern area, the reform area)

In the normal work cycle, Receiver 0 controlled by its autopilot starts from STANDBY state, goes through the Joining sub-phase (105), the Refueling sub-phase (107), the Reforming sub-phase (109) and finally returns to the STANDBY state (141). In detail, taking event 105 as an example, it leads Receiver 0 from STANDBY state to Joining sub-phase. When this maneuver fails, namely event 102 happens, Receiver 0 retreats to Standby state. Otherwise, event 104 happens, which leads to Receiver 0 to Joining sub-phase. When failures occur, Receiver 0 controlled by its autopilot can retreat to STANDBY, RTL, and EL states from Joining, Refueling, and Reforming sub-phase through events 121, 123 and 125, respectively. Meanwhile, according to the real-time health conditions, Receiver 0 can make transitions among states in the withdrawal phase like from RTL to STANDBY.

In Fig. 4, States Joining02, Refueling02, and Reforming02 are OR superstates. The remaining states are simple states. States Joining01 and Joining02 represent Joining sub-phase in Fig. 4. Detailed information about the Joining sub-phase is shown in Fig. 5.

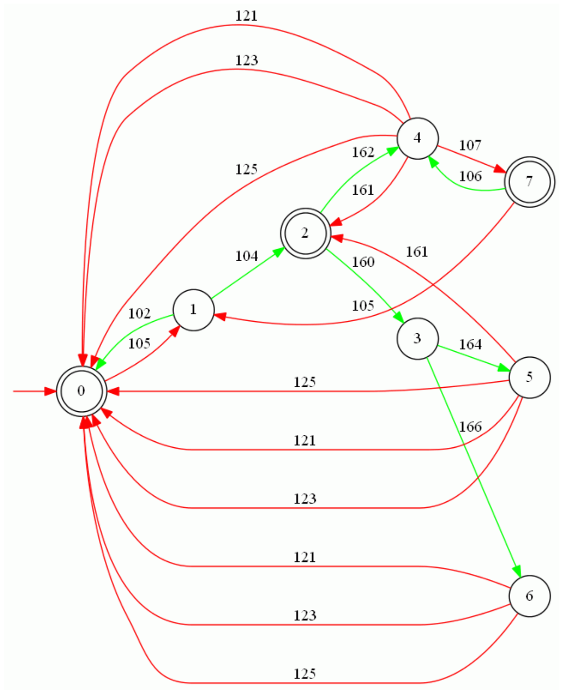


Fig. 5 The holon of Joining sub-phase

For simplicity, State 0 in the holon of Joining sub-phase represents three separate states, including STANDBY, RTL, and EL. The following two holons for Refueling sub-phase and Reforming sub-phase have been simplified as well.

Table 4 Event definitions for Joining sub-phase

Event label	Description
161	Control Receiver 0 to wait in the observation area while avoiding collision with other aerial vehicles
160	Receiver 0 is waiting for the command by the tanker to allow it to fly to the astern area and connect its probe with the drogue
162	Receiver 0 received for the command by the tanker to allow it to fly to the astern area and connect its probe with the drogue
164	The waiting time at the observation does not exceed the specified threshold
166	The waiting time at the observation area exceeds the specified threshold

The dynamical behavior related to the Joining sub-phase is shown in Fig. 5. When event 105 happens, Receiver 0 controlled by its autopilot enters the Joining sub-phase from STANDBY and stays in state 1 of Fig. 5. When this maneuver succeeds (event 104), Receiver 0 enters state 2 of Fig. 5, where the receiver has to wait for the command given by the tanker to allow the receiver to fly to the astern area and connect its probe with the drogue. A timer is set for the preparation of flying to the astern area. When Receiver 0 does not receive the command, it has to check whether the timer has ended. If so, Receiver 0 has to abandon the current refueling task and return to the withdrawal phase. Otherwise, Receiver 0 can keep waiting (event 161) or retreat to the withdrawal phase according to the health conditions. Once Receiver 0 has received the command of allowing connection (event 162) before the timer ends, Receiver 0 has to check whether the safety requirements or health conditions are satisfied for event 107. If so, Receiver 0 can execute the event 107. Otherwise, Receiver 0 has to wait or withdraw.

Table 5 Event definitions for Refueling sub-phase

Event label	Description
170	The event 173 fails, i.e., Receiver 0 fails to connect its probe with the tanker's drogue
172	The event 173 succeeds, i.e., Receiver 0 successfully connects its probe with the tanker's drogue
173	Receiver 0 starts to connect its probe with the tanker's drogue
174	The event 177 fails, i.e., the tanker fails to transfer the fuel to Receiver 0
176	The event 177 succeeds, i.e., the tanker successfully transfers the fuel to Receiver 0
177	Receiver 0 starts to keep relatively stationary to the tanker, and open the valve to receive fuel
178	Receiver 0 fails to receive the command by the tanker to allow it to disconnect from the drogue
180	Receiver 0 receives the command by the tanker to allow it to disconnect from the drogue
181	Receiver 0 waits in the astern area while keeping connected with the tanker
182	The waiting time at the astern area does not exceed the specified threshold
184	The waiting time at the astern area exceeds the specified threshold

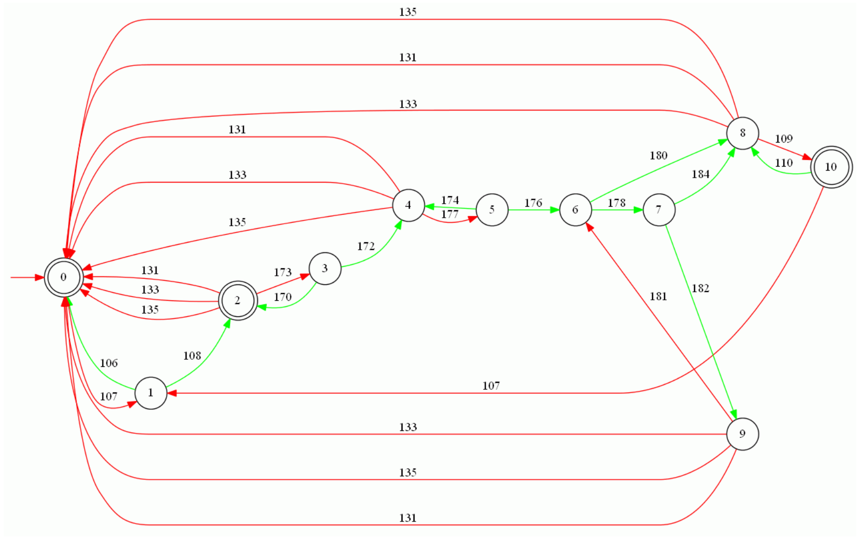


Fig. 6 The holon of Refueling sub-phase

The dynamical behavior related to the Refueling sub-phase is shown in Fig. 6. Event 107 brings Receiver 0 from the Joining sub-phase to the Refueling sub-phase. In state 2 of Fig. 6, according to the safety requirement, Receiver 0 can choose to initiate a connection (event 173), or return to the withdrawal phase. The connection action has two results: failure brings Receiver 0 return to state 2 of Fig. 6 (event 170), while success leads Receiver 0 to state 4 of Fig. 6 (event 172). The success of fuel transfer leads Receiver 0 to state 6 of Fig. 6, where Receiver 0 has to wait for the command of disconnection clearance within a specified time interval. If Receiver 0 doesn't receive the command, it has to check the waiting time. If time runs out (event 184), Receiver 0 will directly disconnect with the tanker instead of returning to the withdrawal phase. If Receiver 0 receives the command of disconnection (event 180) before the timer ends, it can initiate event 109 to the Reform sub-phase or return to the withdrawal phase because of some possible health issues of subsystems.

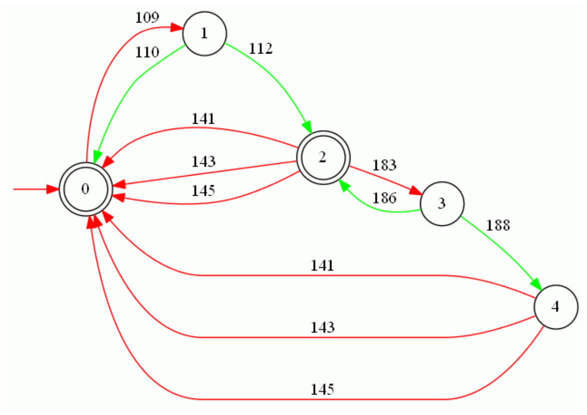


Fig. 7 The holon of Reforming sub-phase

Table 6 Event definitions for Reforming sub-phase

Event label	Description
183	Control Receiver 0 to rejoin the formation in the reform area
186	The event 183 fails, i.e., Receiver 0 has not rejoined the receiver formation
188	The event 183 succeeds, i.e., Receiver 0 has successfully rejoined the receiver formation

The dynamical behavior related to the Reforming sub-phase is shown in Fig. 7. Event 109 will bring Receiver 0 from the Refueling sub-phase to the Reforming sub-phase. Its success will lead to state 2 of Fig. 7, where Receiver 0 can return to the withdrawal phase due to health conditions or form the formation (event 183). Event 188 implies the completion of a work cycle of autonomous aerial refueling, and then Receiver 0 should control aerial vehicle 0 to return to a new STANDBY state in the withdrawal phase.

The formal description of the autopilot of Receiver 0 is completed. In total, the autopilot of Receiver 0 has 23 states and 65 transitions. We considered multiple aerial vehicles in this project. The formal description of Receiver 1's autopilot is similar to that of Receiver 0's autopilot except for the relabeling of all events in Receiver 0's autopilot. Both autopilots have the same state dependency and dynamical transitions, except event labels.

3.2 Formal description of the safety issues for every receiver

Common failures in autonomous aerial refueling affect the decision-making by every receiver. They are the health information that should be taken into consideration. These failures are usually related to control and sensor subsystems of the receiver. A holon is used to depict the abstraction of a subsystem in this project. For each subsystem, its holon contains three different health states, namely Normal, Minor-Damage, and Critical-Damage. State Normal represents that the subsystem is normal and satisfies basic requirements with all components being healthy. State Minor-Damage represents that although there are some failures in the subsystem, it can still satisfy basic requirements. State Critical-Damage represents that the subsystem has sustained critical damage and can no longer satisfy basic requirements. Three different events, namely Suspension, Recovery, and Breakdown, are defined in the holon of a subsystem. These events represent the diagnosis results by low-level modules. Subsystem information and description of Receiver 0 are shown in Table 7 and Fig. 9, respectively.

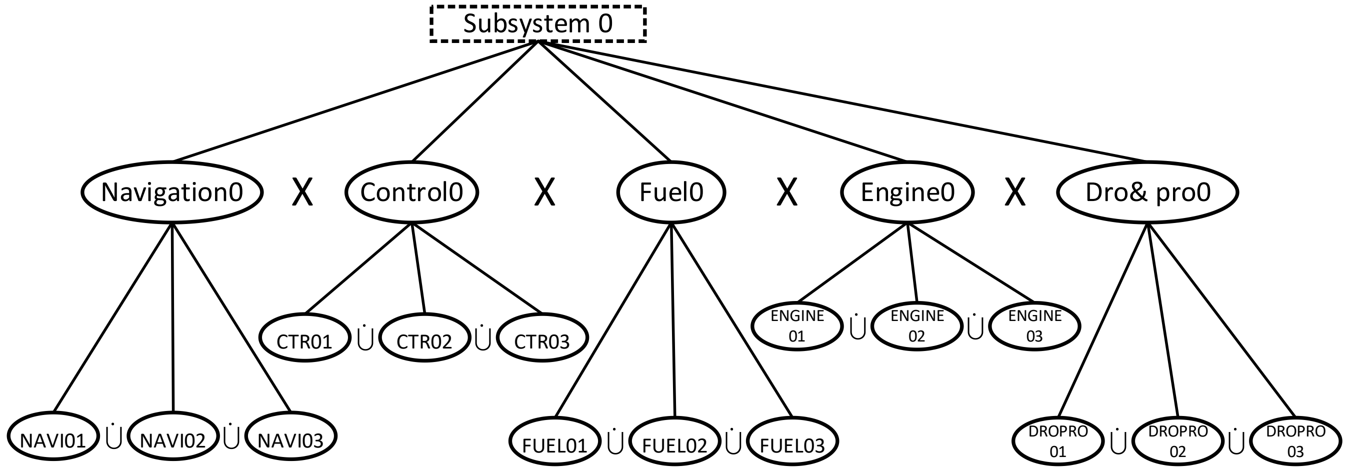
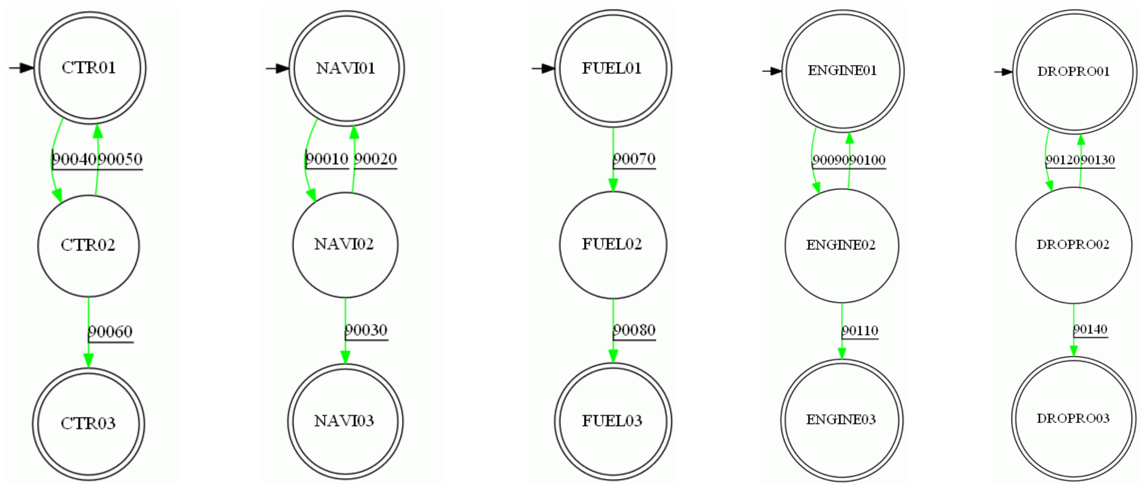


Fig. 8 The state tree of Subsystem 0

Table 7 Subsystem information

Subsystem name	Basic requirement
Control0	Keep the receiver's position and velocity within an allowable range according to its current mode, to avoid collision with other aerial vehicles and achieve successful docking
Navigation0	Provide data about the relative position and velocity among receivers and the tanker to facilitate docking
Fuel0	Provide the necessary fuel for the flight
Engine0	Provide the necessary power and thrust for flight
Drogue&probe0	Establish robust contact between the drogue and probe, and then transfer fuel from the tanker to the receiver



(a) (b) (c) (d) (e)

Fig. 9 The holons of (a) Control subsystem, (b) Navigation subsystem, (c) Fuel subsystem, (d) Engine subsystem, and (e) Drogue & probe connection subsystem

Table 8 State abstraction for every subsystem

State meaning	State label in the subsystem
Normal	CTR01 / NAVI01 / FUEL01 / ENGINE01 / DROPRO01
Minor-Damage	CTR02 / NAVI02 / FUEL02 / ENGINE02 / DROPRO02
Critical-Damage	CTR03 / NAVI03 / FUEL03 / ENGINE03 / DROPRO03

Table 9 Event definition for every subsystem

Event meaning	Event label in the subsystem
Suspension	90010 / 90040 / 90070 / 90090 / 90120
Recovery	90020 / 90050 / 90100 / 90130
Breakdown	90030 / 90060 / 90080 / 90110 / 90140

For the detailed meaning of each event in subsystems, please refer to [2]. Because of the special characteristics, event Recovery is not defined in the Fuel subsystem.

The formal description of the subsystems of Receiver 0 is completed. The formal description of Receiver 1's subsystems is similar to that of Receiver 0's subsystems except for the relabeling of all events.

3.3 Subsystem health management strategy

Possible failures will affect the normal refueling process of the receiver. We need to restrict the undesired and unsafe behavior of the receiver. Subsystem-related specifications are designed and used to restrict system behavior when certain subsystems are damaged or break down. Detailed safety requirements can be referred to in [2]. Five specifications are used in this project; they are Spec-Navigation, Spec-Control, Spec-Fuel, Spec-Engine, and Spec-DroPro.

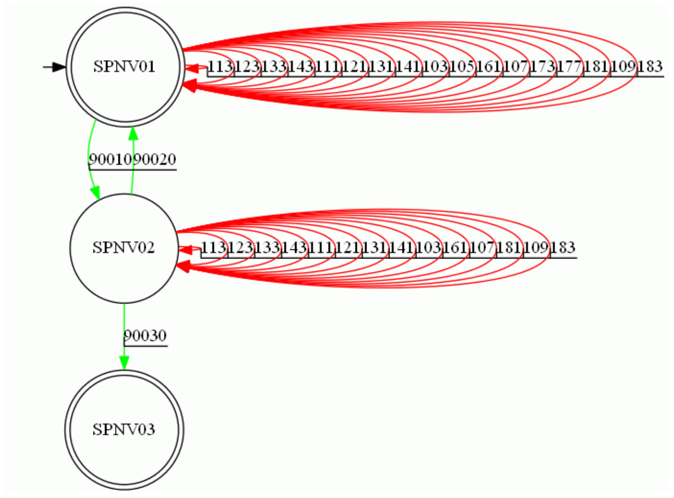


Fig. 10 The holon of Spec-Navigation for Receiver 0

When event Suspension (event 90010) happens in Spec-Navigation, Events 105, 173, and 177 should be disabled. Therefore, as shown in Fig. 10, events 105, 173, and 177 are not defined at state SPNV02. When event Breakdown (event 90030) happens, all controllable events should be forbidden except for those events (115, 125, 135, and 145) that control Receiver 0 to stay in Emergency Landing (EL) mode. This means that when the Navigation subsystem breaks down, Receiver 0 should make an emergency landing. Other subsystem-related specifications are depicted as follows.

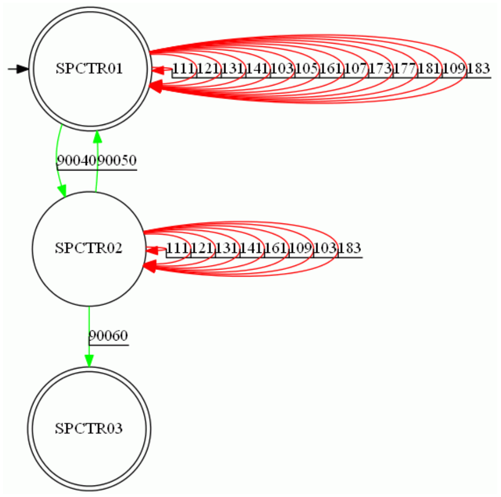


Fig. 11 The holon of Spec-Control for Receiver 0

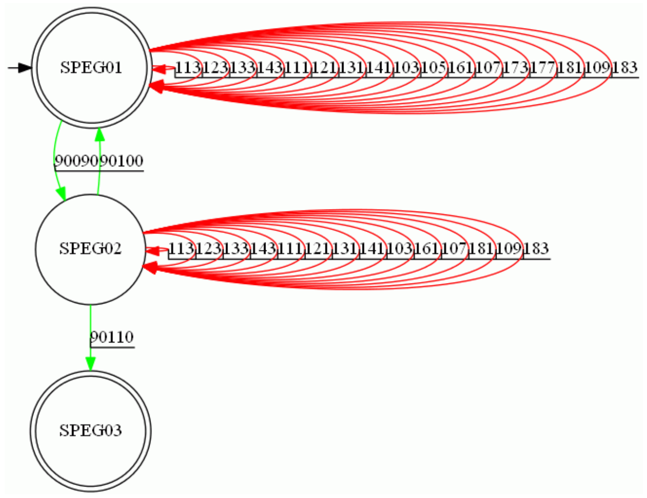


Fig. 12 The holon of Spec-Engine for Receiver 0

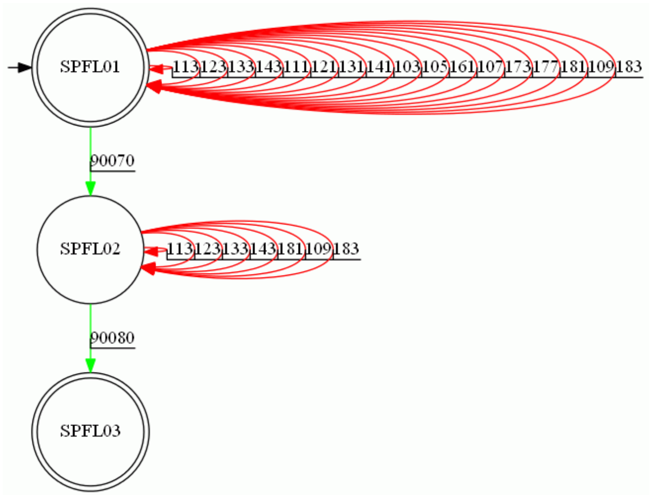


Fig. 13 The holon of Spec-Fuel for Receiver 0

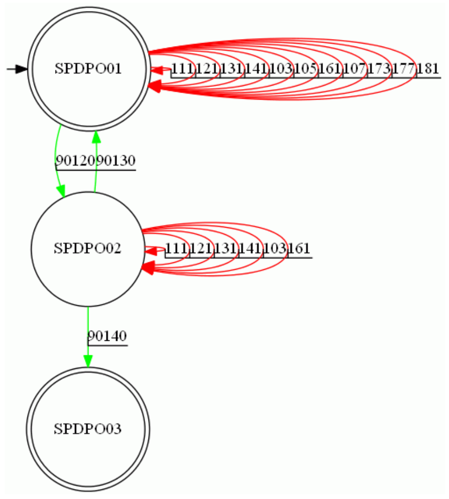


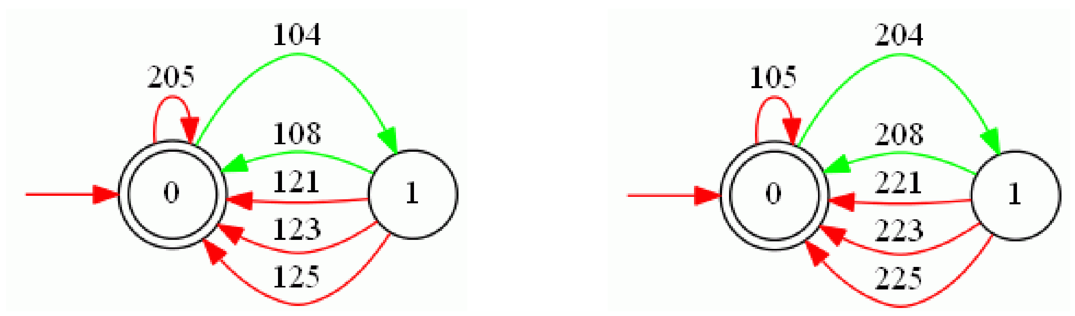
Fig. 14 The holon of Spec-Dropro for Receiver 0

The subsystem-related specifications for Receiver 1 have the same structures as the above holons except for event labels. In summary, the health conditions of subsystems have an important role in the high-level logic controller. The health conditions determine whether the receiver can continue to perform the refueling task. When a subsystem is critically damaged, the receiver must abort the refueling task for an emergency landing. In a fault situation, the decisions made by the receiver should be considered in the high-level logic controller.

3.4 Mutual-exclusion control strategy

As shown in Fig. 2, the refueling procedure needs to be performed in different areas. Each area is a public resource for multiple aerial vehicles. In the task space, every public resource is limited. Although it is required that the control subsystem keeps the receiver's position and velocity to avoid collision with other aerial vehicles, unsafe conflicts may still occur when using public resources. A mutual-exclusion control strategy is developed to prevent collision between multiple aerial vehicles and improve refueling efficiency. The mutual-exclusion control strategy is composed of four groups of specifications.

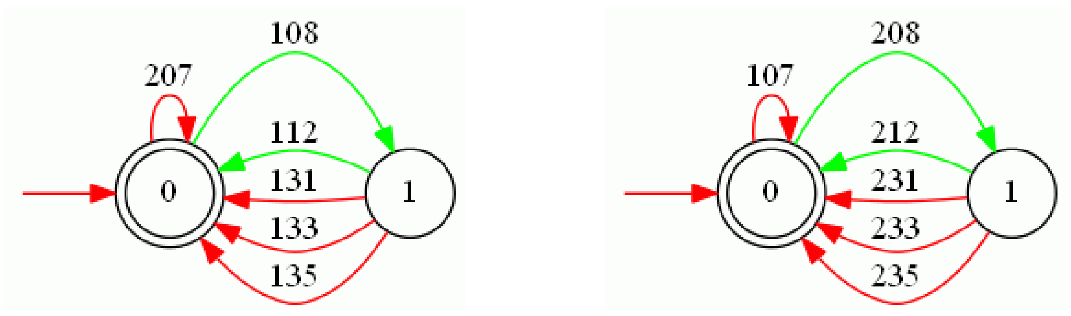
The first group has six specifications. These parallel specifications eliminate all conflict behavior when using every public resource. These public resources include the observation area related to the Joining sub-phase, the astern area related to the Refueling sub-phase, and the reform area related to the Reforming sub-phase.



(a) (b)

Fig. 15 (a) The holon of Spec-J0; (b) The holon of Spec-J1

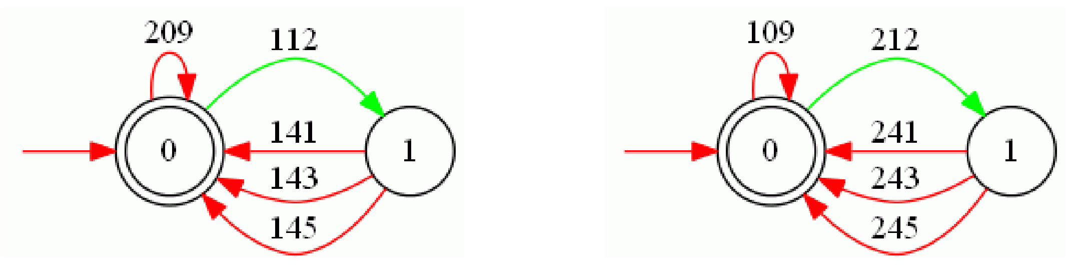
Spec-J0 and Spec-J1 are the mutual-exclusion controllers related to the Joining sub-phase. In Spec-J0, event 205 is disabled at state 1 because state 1 means that Receiver 0 is already in the observation area. Unless Receiver 0 leaves the observation area through events 108, 121, 123, or 125, Receiver 1 can start to fly to the observation area (event 205 in Fig. 15(a)). The desired behavior in Spec-J1 is similar to those in Spec-J0. Receiver 0 can start to fly to the observation area (event 105 in Fig.15(b)) when this area is empty.



(a) (b)

Fig. 16 (a) The holon of Spec-RFU0; (b) The holon of Spec-RFU1

Spec-RFU0 and Spec-RFU1 are the mutual-exclusion controllers related to the Refueling sub-phase. In Spec-RFU0, event 207 is disabled at state 1 because state 1 means that Receiver 0 is already in the astern area. Unless Receiver 0 leaves the astern area through events 112, 131, 133 or 135, Receiver 1 can start to fly to the astern area (event 207 in Fig. 16(a)). The desired behavior in Spec-RFU1 is similar to those in Spec-RFU0.

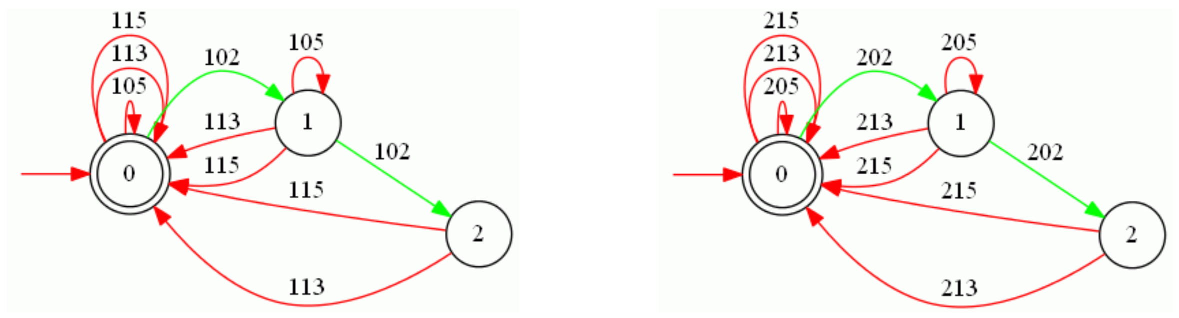


(a) (b)

Fig. 17 (a) The holon of Spec-RFM0; (b) The holon of Spec-RFM1

Spec-RFM0 and Spec-RFM1 are the mutual-exclusion controllers related to the Reforming sub-phase. In Spec-RFM0, event 209 is disabled at state 1 because state 1 means that Receiver 0 is already in the reform area. Unless Receiver 0 leaves the reform area through events 141, 143, or 145, Receiver 1 can start to fly to the reform area (event 209 in Fig. 17(a)). The desired behavior in Spec-RFM1 is similar to those in Spec-RFM0.

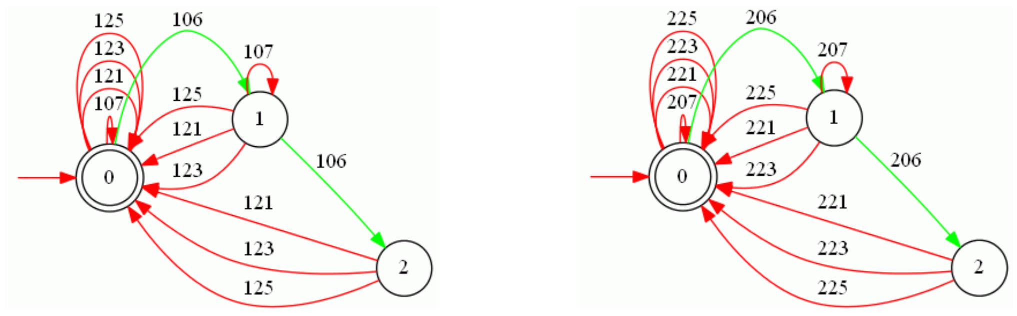
The second group also has six specifications. These parallel specifications eliminate all inefficient behavior that is unlimited continuous attempts by the receiver to use public resources. Because Receiver 0 may fail to fly to the observation area (events 102), it can try to drive event 105 to fly to the observation area a second time. If it fails again, it will lose the third chance and fly to the withdrawal space. Then, Receiver 1 obtains two continuous attempts to fly to the observation area. Other inefficient behavior for entering the astern area and the reform area is similar to those for entering the observation area.



(a) (b)

Fig. 18 (a) The holon of Spec-JF0; (b) The holon of Spec-JF1

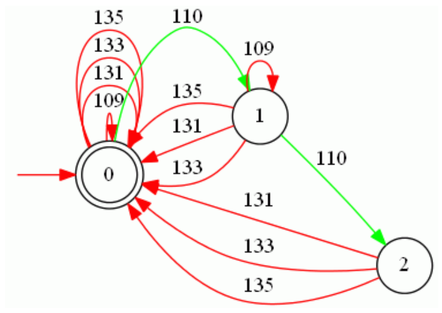
Spec-JF0 and Spec-JF1 are the mutual-exclusion controllers to eliminate inefficient attempts for the observation area. For example, in Spec-JF0 that is designed for Receiver 0, event 105 is disabled at state 2 because state 2 means that Receiver 0 has failed twice to enter the observation area. This strategy requires that Receiver 0 cannot continuously try to fly to the observation area and let the other aerial vehicle keep waiting. Spec-JF0 supports two chances for Receiver 0 to fly to the observation area. At state 2 of Spec-JF0, Receiver 0 has exhausted two chances. It should enter the withdrawal space and leave the chance to another aerial vehicle. Receiver 0 may have some health issues of control and sensor subsystems in achieving the strategy in Spec-JF0. In an unhealthy situation, Receiver 0 should stop the current refueling task and enter the withdrawal space. Events 113 and 115 are to control Receiver 0 enter the withdrawal space, and thus they are enabled at every state of Spec-JF0. When aerial vehicle 0 is normal, or it returns to the STANDBY state, event 105 can be driven by Receiver 0 to fly to the observation area. A similar strategy has been designed in Spec-JF1 for Receiver 1.

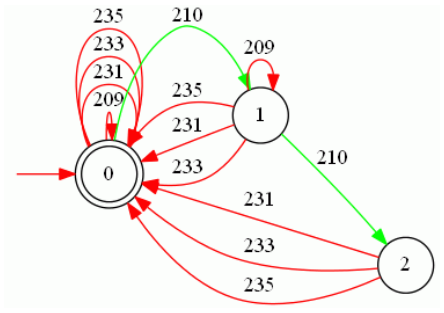


(a) (b)

Fig. 19 (a) The holon of Spec-RFUF0; (b) The holon of Spec-RFUF1

Spec-RFUF0 and Spec-RFUF1 are the mutual-exclusion controllers to eliminate inefficient attempts for the astern area. In Spec-RFUF0, event 107 is disabled at state 2 because state 2 means that Receiver 0 has failed twice to enter the astern area. Receiver 0 should enter the withdrawal space and let Receiver 1 try to enter the astern area. Due to possible health issues of control and sensor subsystems in achieving the strategy in Spec-RFUF0, Receiver 0 in the observation area should stop the current refueling task and enter the withdrawal space by events 121, 123, and 125. Thus, events 121, 123, and 125 are enabled at every state of Spec-RFUF0 to respond to possible health issues. When aerial vehicle 0 is normal, or it returns to the STANDBY state, event 107 can be driven by Receiver 0 to fly to the astern area. The desired behavior for Receiver 1 in Spec-RFUF1 is similar to those in Spec-RFUF0.





(a) (b)

Fig. 20 (a) The holon of Spec-RFMF0; (b) The holon of Spec-RFMF1

Spec-RFMF0 and Spec-RFMF1 are the mutual-exclusion controllers to eliminate inefficient attempts for the reform area. In Spec-RFMF0, event 109 is disabled at state 2 because state 2 means that Receiver 0 has failed twice to enter the reform area. Receiver 0 should enter the withdrawal space and let Receiver 1 try to enter the reform area. Due to possible health issues of control and sensor subsystems in achieving the strategy in Spec-RFMF0, Receiver 0 in the astern area should stop the current refueling task and enter the withdrawal space by events 131, 133, and 135. Thus, events 131, 133, and 135 are enabled at every state of Spec-RFMF0 to respond to possible health issues. When aerial vehicle 0 is normal, or it returns to the STANDBY state, event 109 can be driven by Receiver 0 to fly to the reform area. The desired behavior for Receiver 1 in Spec-RFMF1 is similar to those in Spec-RFMF0.

The third group has one specification, namely Spec-ORD. The specification is used to manage the order of starting the refueling procedure. According to Spec-ORD, usually, Receiver 0 starts first, and Receiver 1 starts later. Event 105 is enabled while event 205 is disabled at state 0. Event 105 is disabled and event 205 is enabled at state 1 of Spec-ORD, which means that Receiver 1 should start its refueling task without waiting for Receiver 0 to complete its refueling task. The specification Spec-ORD also gives the conditions for Receiver 1 to start its refueling procedure. When Receiver 0 leaves the observation area by events 108, 121, 123, or 125, Receiver 1 should start its refueling procedure. When Receiver 0 cannot fly to the observation area because of possible health issues of control and sensor subsystems by events 113 or 115, Receiver 1 should start its refueling procedure. Events 113 and 115 are allowed to self-loop at state 1 of Spec-ORD. Because Receiver 0 is allowed to enter and stay in EL mode due to the critical damage of some control and sensor subsystems if Receiver 0 is already staying in RTL mode, regardless of whether Receiver 1 starts the refueling procedure.

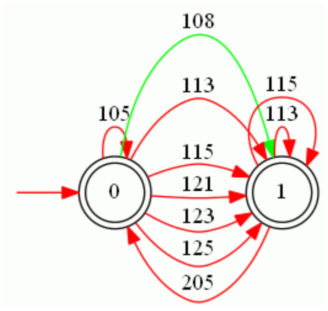


Fig. 21 The holon of Spec-ORD

The fourth group has one specification, namely Spec-STP. The specification restricts the behavior according to the task completion of multiple aerial vehicles. In the scenario of this project, there are four cases from the perspective of task completion. State 0 in Spec-STP represents that no aerial vehicle has completed the refueling task. If an aerial vehicle has completed the refueling task, it should not start the task again by entering into the observation area. Event 105 (resp. event 205) is disabled at state 1 (resp. state 2). State 3 in Spec-STP represents that both aerial vehicles have completed the refueling task.

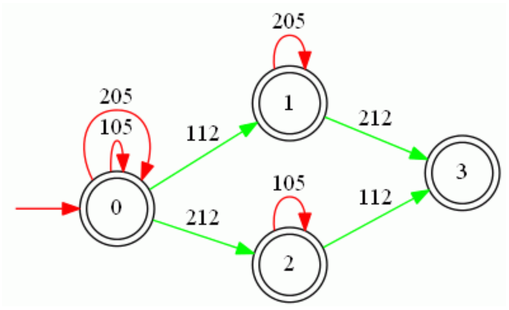


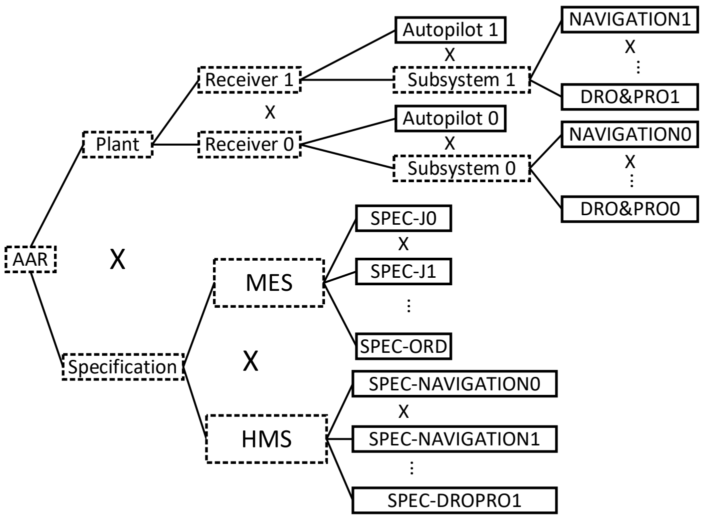
Fig. 22 The holon of Spec-STP

This completes the design of the mutual-exclusion control strategy.

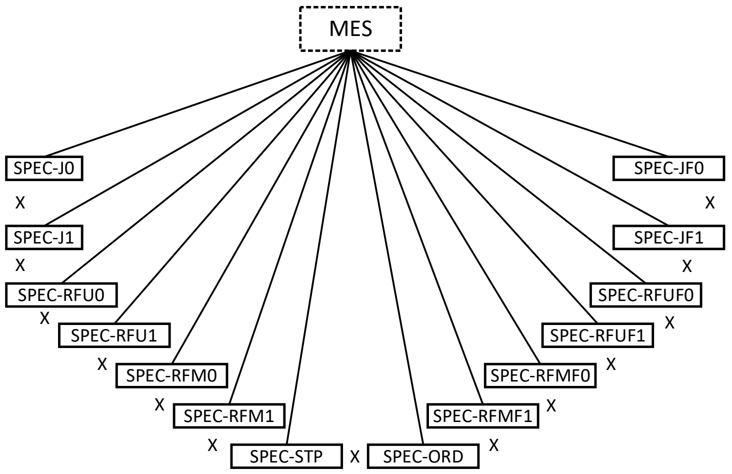
4 Realization of successful logic control for autonomous aerial refueling operation with multiple receivers

According to the above formal description, supervisory control theory based on state tree structure is used to synthesize the supremal nonblocking behavior of the autonomous refueling task for multiple aerial vehicles. The plant includes all defined behavior of two receivers in the autonomous refueling task. The behavior is modeled in the holons of autopilots and subsystems. The specifications are models of requirements, which identify the desired and undesired states and transitions. Here, all requirements are included in the mutual-exclusion control strategy and the subsystem health management strategy.

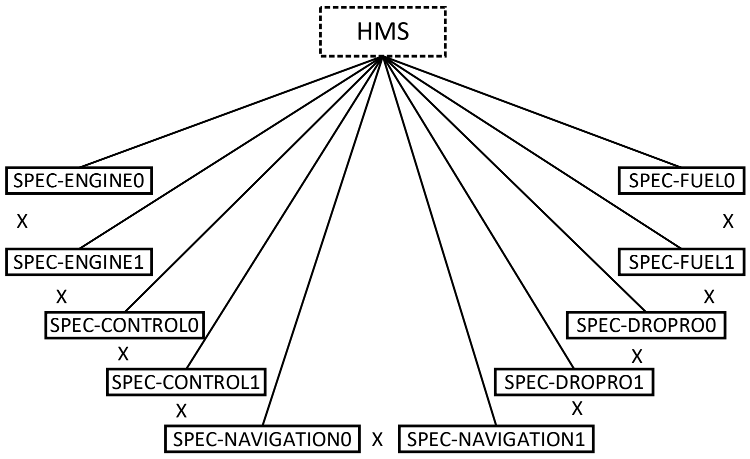
In Fig. 23, the components within the solid boxes are OR superstates, and those within the dotted boxes are AND superstates. The AND superstate **MES** is the synchronous product of 14 OR superstates. The AND superstate **HMS** is the synchronous product of 10 OR superstates.



(a)



(b)



(c)

Fig. 23 (a) The state tree of the autonomous aerial refueling task with two receivers; (b) The state tree of the mutual exclusion control strategy (MES); (c) The state tree of the health management strategy (HMS)

The state size of the AND superstate **Plant** of the whole system is $(23\times3\times3\times3\times3\times3)^2 = \mathbf{31{,}236{,}921}$.

The state size of the AND superstate **MES** considered here is $(2^3\times3^3)^2\times2\times4 = \mathbf{373{,}248}$.

The state size of the AND superstate **HMS** is $(3^5)^2 = \mathbf{59{,}049}$.

For the state tree structure of the whole system in Fig. 23, the state size is nearly 2.0×10^{18}. It takes 3827.99 seconds (more than 1 hour) to compute the supervisor of the problem with 2.6GHz i7 CPU and 16GB RAM. The supervisor is implemented in binary decision diagrams (BDD) for every controllable event. In Table 10 below, "BDD size" means the number of nodes.

Table 10 Comparison of supervision results after considering different subsystems

Subsystems considered	**None**	**Control**	**Control+ Navigation**	**Control+ Navigation+ Fuel**	**Control+ Navigation+ Fuel+ Engine**	**Control+ Navigation+ Fuel+ Engine+ Dro&pro**
State size	$<2.0\times10^{8}$	$<2.0\times10^{10}$	$<2.0\times10^{12}$	$<2.0\times10^{14}$	$<2.0\times10^{16}$	$<2.0\times10^{18}$
The Time spent in synthesis (seconds)	0.15	0.32	0.66	1.48	26.84	**3413.55**
The BDD size of the specification	126	246	2166	32886	524406	8388726
The BDD size of the illegal state set	126	246	2166	32886	524406	8388726
The BDD size of the optimal controlled behavior	970	945	2673	18225	158193	1417905
The BDD size of all control functions	90	105	179	267	341	395
The maximum BDD size of control functions	19	8	11	14	17	20
The minimum BDD size of control functions	0	0	0	0	0	0

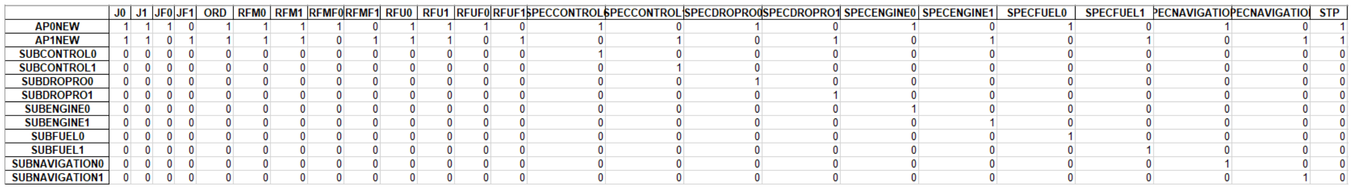
According to Table 10, the more complicated the system is, the longer the computation time is. The cut-off point of computing is when the system size is 10^{15}. The current state tree structure library cannot quickly solve a supervision problem with more than 10^{15} states. For the autonomous aerial refueling task, the limit to consider all specifications related to the mutual-exclusion control strategy and the specifications related to five subsystems. Based on the previous description and the test, if we add the specifications related to the sixth subsystem, it cannot obtain the optimal controlled behavior in one day. The above results are obtained in solving the autonomous aerial refueling task for multiple aerial vehicles. Other specific issues need to be analyzed specifically.

In the supervision results, some controllable events may be enabled at all defined states because no specification restricts the occurrence of them. Some controllable events may be disabled at all defined states. In these two extreme cases, their BDDs are null which means that the corresponding BDD size is zero. Table 10 shows the prompt information output by the current STS software, and the information includes a minimum BDD size.

5 Multi-level STS synthesis for autonomous aerial refueling operation with multiple receivers

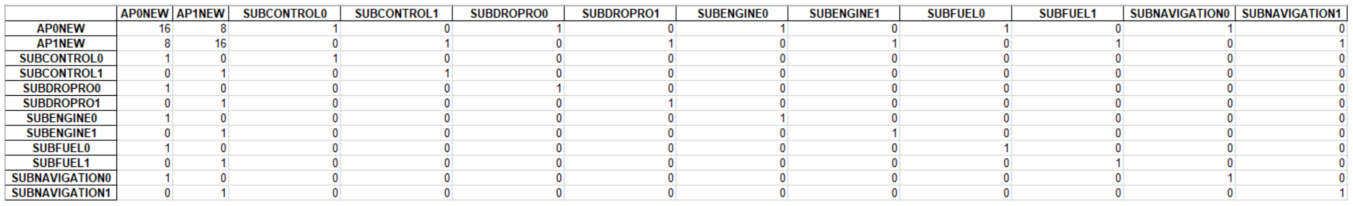
In this section, we will use multi-level Markov clustering algorithm introduced in [3] to classify the plant modules and generate tree-structured multi-level STS for this problem.

First, based on the plant modules and specifications constructed in the previous sections, the domain mapping matrix (DMM) PR and the dependence structure matrix (DSM) P can be obtained as follows.



	J0	J1	JF0	JF1	ORD	RFM0	RFM1	RFMF0	RFMF1	RFU0	RFU1	RFUF0	RFUF1	SPECCONTROL	SPECCONTROL	SPECDROPRO0	SPECDROPRO1	SPECENGINE0	SPECENGINE1	SPECFUEL0	SPECFUEL1	PECNAVIGATIO	PECNAVIGATIO	STP
AP0NEW	1	1	1	0	1	1	1	1	0	1	1	1	0	1	0	1	0	1	0	1	0	1	0	1
AP1NEW	1	1	0	1	1	1	1	0	1	1	1	0	1	0	1	0	1	0	1	0	1	0	1	1
SUBCONTROL0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
SUBCONTROL1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
SUBDROPRO0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
SUBDROPRO1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
SUBENGINE0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
SUBENGINE1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
SUBFUEL0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
SUBFUEL1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
SUBNAVIGATION0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
SUBNAVIGATION1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

$$PR = \begin{bmatrix}
1&1&1&&1&1&1&1&&1&1&1&&1&&1&&1&&1&&1&&1\\
1&1&&1&1&1&1&&1&1&1&&1&&1&&1&&1&&1&&1&1\\
&&&&&&&&&&&&&1&&&&&&&&&&\\
&&&&&&&&&&&&&&1&&&&&&&&&\\
&&&&&&&&&&&&&&&1&&&&&&&&\\
&&&&&&&&&&&&&&&&1&&&&&&&\\
&&&&&&&&&&&&&&&&&1&&&&&&\\
&&&&&&&&&&&&&&&&&&1&&&&&\\
&&&&&&&&&&&&&&&&&&&1&&&&\\
&&&&&&&&&&&&&&&&&&&&1&&&\\
&&&&&&&&&&&&&&&&&&&&&1&&\\
&&&&&&&&&&&&&&&&&&&&&&1&
\end{bmatrix}$$



	AP0NEW	AP1NEW	SUBCONTROL0	SUBCONTROL1	SUBDROPRO0	SUBDROPRO1	SUBENGINE0	SUBENGINE1	SUBFUEL0	SUBFUEL1	SUBNAVIGATION0	SUBNAVIGATION1
AP0NEW	16	8	1	0	1	0	1	0	1	0	1	0
AP1NEW	8	16	0	1	0	1	0	1	0	1	0	1
SUBCONTROL0	1	0	1	0	0	0	0	0	0	0	0	0
SUBCONTROL1	0	1	0	1	0	0	0	0	0	0	0	0
SUBDROPRO0	1	0	0	0	1	0	0	0	0	0	0	0
SUBDROPRO1	0	1	0	0	0	1	0	0	0	0	0	0
SUBENGINE0	1	0	0	0	0	0	1	0	0	0	0	0
SUBENGINE1	0	1	0	0	0	0	0	1	0	0	0	0
SUBFUEL0	1	0	0	0	0	0	0	0	1	0	0	0
SUBFUEL1	0	1	0	0	0	0	0	0	0	1	0	0
SUBNAVIGATION0	1	0	0	0	0	0	0	0	0	0	1	0
SUBNAVIGATION1	0	1	0	0	0	0	0	0	0	0	0	1

$$P=\begin{bmatrix}16 & 8 & 1 & & 1 & & 1 & & 1 & & 1 & \\ 8 & 16 & & 1 & & 1 & & 1 & & 1 & & 1 \\ 1 & & 1 & & & & & & & & & \\ & 1 & & 1 & & & & & & & & \\ 1 & & & & 1 & & & & & & & \\ & 1 & & & & 1 & & & & & & \\ 1 & & & & & & 1 & & & & & \\ & 1 & & & & & & 1 & & & & \\ 1 & & & & & & & & 1 & & & \\ & 1 & & & & & & & & 1 & & \\ 1 & & & & & & & & & & 1 & \\ & 1 & & & & & & & & & & 1\end{bmatrix}$$

The bus nodes and non-bus nodes in P are obvious. Let $\alpha=2$, $\beta=2.0$, $\mu=2.0$ and $\gamma=2$ in the clustering algorithm. The clustered result is given as follows.

$$\begin{aligned}[A,M]=[&\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11\},\\ &[[\{0, 1\}, [[\{0\}, [0]], [\{1\}, [1]]]],\\ &[\{2, 3, 4, 5, 6, 7, 8, 9, 10, 11\}, [[\{2\}, [2]], [\{3\}, [3]], [\{4\}, [4]], [\{5\}, [5]], [\{6\}, [6]],\\ &[\{7\}, [7]], [\{8\}, [8]], [\{9\}, [9]], [\{10\}, [10]], [\{11\}, [11]]]]]].\end{aligned}$$

The bus nodes are A0NEW (index 0) and A1NEW (index 1). All other nodes are non-bus nodes. Then, the tree-structured multi-level STS for the autonomous aerial refueling problem has 4 non-empty nodes.

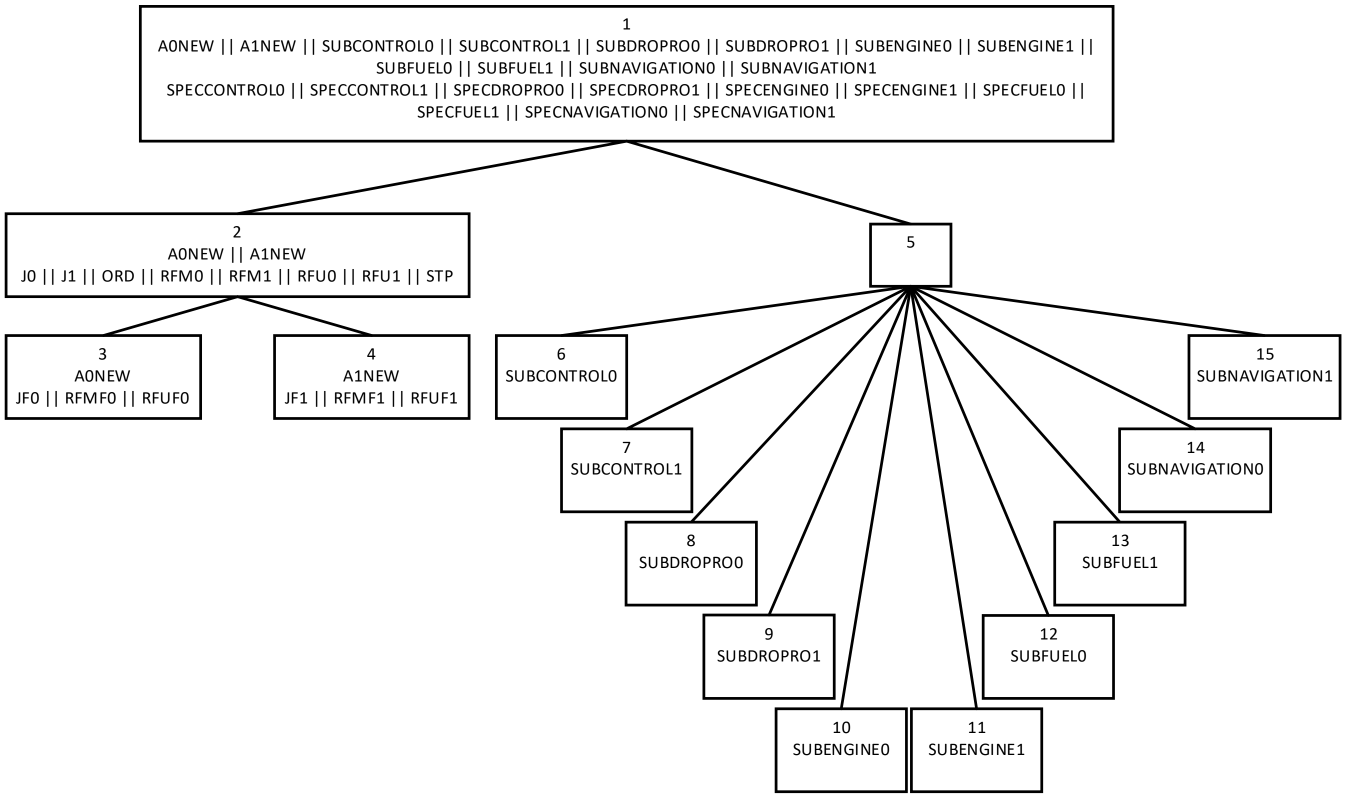

1
A0NEW || A1NEW || SUBCONTROL0 || SUBCONTROL1 || SUBDROPRO0 || SUBDROPRO1 || SUBENGINE0 || SUBENGINE1 ||
SUBFUEL0 || SUBFUEL1 || SUBNAVIGATION0 || SUBNAVIGATION1
SPECCONTROL0 || SPECCONTROL1 || SPECDROPRO0 || SPECDROPRO1 || SPECENGINE0 || SPECENGINE1 || SPECFUEL0 ||
SPECFUEL1 || SPECNAVIGATION0 || SPECNAVIGATION1
2
A0NEW || A1NEW
J0 || J1 || ORD || RFM0 || RFM1 || RFU0 || RFU1 || STP
5
3
A0NEW
JF0 || RFMF0 || RFUF0
4
A1NEW
JF1 || RFMF1 || RFUF1
6
SUBCONTROL0
7
SUBCONTROL1
8
SUBDROPRO0
9
SUBDROPRO1
10
SUBENGINE0
11
SUBENGINE1
12
SUBFUEL0
13
SUBFUEL1
14
SUBNAVIGATION0
15
SUBNAVIGATION1

Fig. 24 The tree-structured multi-level STS of the autonomous aerial refueling task

The first non-empty node is Node 1 that describes the dependency between the bus nodes and the non-bus nodes. It can be seen that all specifications in Node 1 are used to describe subsystem health management strategy in Section 3. The role of Node 1 is to generate the decisions of each receiver in a fault situation. The second non-empty node is Node 2 that describes the dependency between the bus nodes. It can be seen that all specifications in Node 2 are built for the mutual-exclusion control strategy in Section 3. Three out of four groups of specifications are included. The third non-empty node is Node 3 that considers the remaining specifications for the plant module A0NEW (index 0). The three specifications in Node 3 (JF0, RFMF0 and RFUF0) are the second group of specifications for the mutual-exclusion control strategy. The fourth non-empty node is Node 4 that considers the remaining specifications for the plant module A1NEW (index 1). The three specifications in Node 4 (JF1, RFMF1 and RFUF1) are the second group of specifications for the mutual-exclusion control strategy. The non-bus nodes are independent of one another. Hence, Node 5 in the multi-level STS is an empty node. No specification is built for an individual plant module of non-bus nodes. Nodes 6-15 are all empty nodes. All 24 specifications are assigned to different nodes in different level in the multi-level STS. The tree-structured multi-level STS for the task has been completed.

Then, we will solve 4 STS synthesis problems for 4 non-empty nodes. The first STS synthesis problem, which is for Node 1, involves 12 plant modules and 10 specifications. The dependency between the bus nodes (A0NEW and A1NEW) and all non-bus nodes is described in Node 1. Based on the detailed modules of Node 1 in Fig. 24, it has that the first STS synthesis problem is to achieve the proposed strategy of subsystem health management. The state space size and the computation time cost of the STS synthesis for Node 1 are shown as follows. The first STS synthesis problem is the biggest problem in the 4 STS synthesis problems. It is fast to achieve the STS synthesis for Node 1 and obtain the non-empty optimal controlled behavior.

The state space size is 23*23*3*3*3*3*3*3*3*3*3*3=**3.1*10^7**
The time spent in synthesis is **92.5038** seconds.
The bdd size of the specification: 2097150
The bdd size of the illegal state set: 2097150
The bdd size of the optimal controlled behavior: 236206
The bdd size of all control functions: 352
The maximum bdd size of control functions: 15
The minimum bdd size of control functions: 0

The second STS synthesis problem involves 2 plant modules and 8 specifications. In Node 2, the plant modules are the bus nodes A0NEW and A1NEW. The 8 specifications are from the proposed mutual-exclusion control strategy. The dependency between A0NEW and A1NEW is described in the second STS synthesis problem. It is to prevent the possible collision of two UAVs in the autonomous air refueling task. The state space size and the computation time cost of the STS synthesis for Node 2 are shown as follows.

The state space size is 23*23*2*2*2*2*2*2*2*4=**2.7*10^5**
The time spent in synthesis is **0.0149601** seconds.
The bdd size of the specification: 86
The bdd size of the illegal state set: 86
The bdd size of the optimal controlled behavior: 256
The bdd size of all control functions: 31

The maximum bdd size of control functions: 3
The minimum bdd size of control functions: 0

The third STS synthesis problem is for Node 3. It involves only one plant module A0NEW and 3 specifications. The 3 specifications are included in the proposed mutual-exclusion control strategy. They are designed to eliminate the inefficient behavior of Receiver 0 that is unlimited continuous attempts to use public resources. The state space size and the computation time cost of the STS synthesis for Node 3 are shown as follows.

The state space size is 23*3*3*3=**621**
The time spent in synthesis is **0.00398493** seconds.
The bdd size of the specification: 16
The bdd size of the illegal state set: 16
The bdd size of the optimal controlled behavior: 27
The bdd size of all control functions: 6
The maximum bdd size of control functions: 2
The minimum bdd size of control functions: 0

The last STS synthesis problem for Node 4 is similar to the third STS synthesis problem. The plant module A1NEW for Receiver 1 is considered and The 3 specifications are designed to eliminate the inefficient behavior of Receiver 1 that is unlimited continuous attempts to use public resources. The state space size and the computation time cost of the STS synthesis for Node 4 are shown as follows.

The state space size is 23*3*3*3=**621**
The time spent in synthesis is **0.00398993** seconds.
The bdd size of the specification: 16
The bdd size of the illegal state set: 16
The bdd size of the optimal controlled behavior: 27
The bdd size of all control functions: 6
The maximum bdd size of control functions: 2
The minimum bdd size of control functions: 0

Nodes 2 to 4 are the nodes in the left tree of the multi-level STS. Nodes 5 to 15 of the right tree are prepared to describe the dependency of all non-bus nodes. Since Nodes 5 to 15 are all empty nodes, no STS synthesis problems should be considered for all non-bus nodes. We need to focus on the generated control functions of the 4 STS synthesis problems.

Table 11 gives the comparison of the control functions in the monolithic STS synthesis and the multi-level STS synthesis. In Table 11, "en.all" means that the corresponding controllable event is enabled at all its defined states. Different from "en.all", "en" means that the related controllable event should be disabled at some of its defined states.

Table 11 Comparison of the control functions in the Monolithic STS synthesis and the multi-level STS synthesis

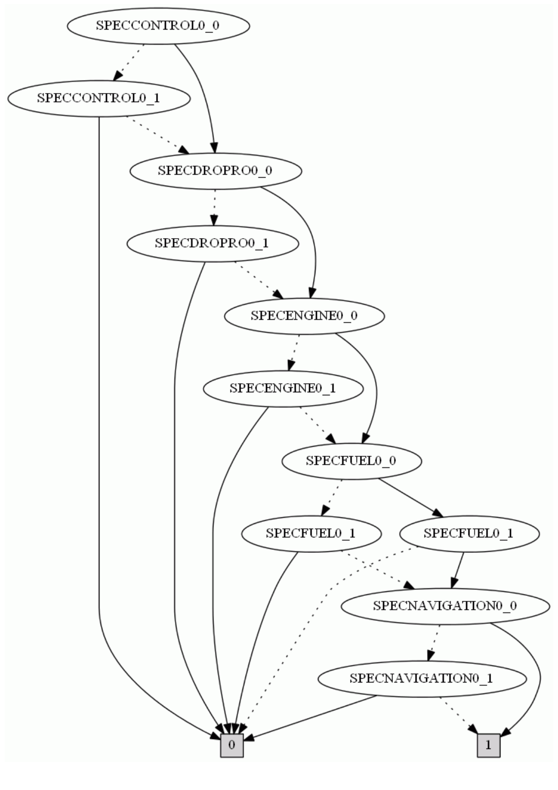
Monolithic results			Multi-level STS Node 1			Multi-level STS Node 2			Multi-level STS Node 3			Multi-level STS Node 4		
en.all	en	en	en.all	en	en	en.all	en	en	en.all	en	en	en.all	en	en
115	*103*	*203*	115	*103*	*203*	*103*	**105**	**205**	*103*	**105**		*203*		**205**
215	**105**	**205**	*125*	**105**	**205**	*111*	**107**	**207**	*111*	**107**		*211*		**207**
	107	**207**	*135*	**107**	**207**	*113*	**109**	**209**	*113*	**109**		*213*		**209**
	109	**209**	*145*	**109**	**209**	115	**121**	**221**	115			215		
	111	*211*	215	*111*	*211*	*161*	**123**	**223**	**121**			**221**		
	113	*213*	*225*	*113*	*213*	*173*	*125*	*225*	**123**			**223**		
	121	**221**	*235*	**121**	**221**	*177*	**131**	**231**	*125*			*225*		
	123	**223**	*245*	**123**	**223**	*181*	**133**	**233**	**131**			**231**		
	125	*225*		**131**	**231**	*183*	*135*	*235*	**133**			**233**		
	131	**231**		**133**	**233**	*203*	**141**	**241**	*135*			*235*		
	133	**233**		**141**	**241**	*211*	**143**	**243**	**141**			**241**		
	135	*235*		**143**	**243**	*213*	*145*	*245*	**143**			**243**		
	141	**241**		*161*	*261*	215			*145*			*245*		
	143	**243**		*173*	*273*	*261*			*161*			*261*		
	145	*245*		*177*	*277*	*273*			*173*			*273*		
	161	*261*		*181*	*281*	*277*			*177*			*277*		
	173	*273*		*183*	*283*	*281*			*181*			*281*		
	177	*277*				*283*			*183*			*283*		
	181	*281*												
	183	*283*												

Table 11 gives the comparison of the control functions in the monolithic STS synthesis and the multi-level STS synthesis. In the three columns on the left, all control functions are listed for the monolithic STS synthesis. Events 115 and 215 are enabled at all states where they are defined. If we track the control functions of events 115 and 215 in the multi-level STS synthesis in the remaining 12 columns, it can be found that events 115 and 215 are enabled at all their defined states. The control results for events 115 and 215 are consistent in the monolithic STS synthesis and the multi-level STS synthesis.

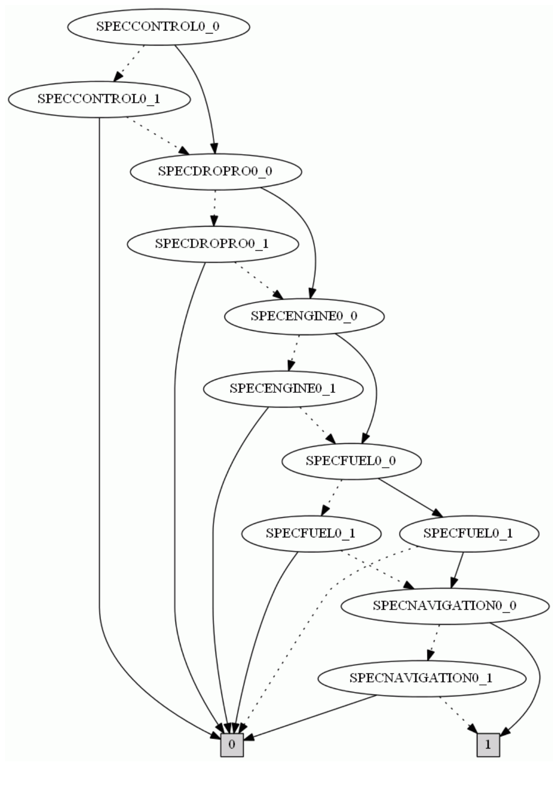
There are two cases for all other controllable events.

- Case 1 (the italic parts in Table 11): The control function of the event is constrained in the multi-level STS synthesis for only **one node**.
- Case 2 (the bold parts in Table 11): The control function of the event is constrained in the multi-level STS synthesis for **multiple nodes**.

An example for Case 1 is the control function of event 103. In Table 11, event 103 is constrained in the multi-level STS synthesis for Node 1. Event 103 is enabled at all states in the multi-level STS synthesis for Nodes 2 and 3. Event 103 is independent of the modules involved in Node 4. We can compare the BDD results of event 103 in the multi-level STS synthesis for Node 1 with that in the monolithic STS synthesis. It is found that the two BDD results are identical. Let $P_0^{[103]}$ be the predicate for disabling event 103 in the monolithic STS synthesis. $\neg P_0^{[103]}$ means the predicate for enabling event 103. Let $P_1^{[103]}$, $P_2^{[103]}$ and $P_3^{[103]}$ be the predicate for disabling event 103 in the multi-level STS synthesis for Nodes 1, 2 and 3, respectively. Based on the results shown in Fig. 25, we can obtain that $P_0^{[103]} = P_1^{[103]} \vee P_2^{[103]} \vee P_3^{[103]} = P_1^{[103]} \vee 0 \vee 0 = P_1^{[103]}$ and $\neg P_0^{[103]} = \neg P_1^{[103]} \wedge \neg P_2^{[103]} \wedge \neg P_3^{[103]} = \neg P_1^{[103]} \wedge 1 \wedge 1 = \neg P_1^{[103]}$. The relationship is clear and comprehensible. Another example is event 125 in Fig. 26. $P_0^{[125]} = P_1^{[125]} \vee P_2^{[125]} \vee P_3^{[125]} = 0 \vee P_2^{[125]} \vee 0 = P_2^{[125]}$ and $\neg P_0^{[125]} = \neg P_1^{[125]} \wedge \neg P_2^{[125]} \wedge \neg P_3^{[125]} = 1 \wedge \neg P_2^{[125]} \wedge 1 = \neg P_2^{[125]}$. Similar relationships can be obtained for events 111, 113, 135, 145, 161, 173, 177, 181, 183 of Receiver 0. Events 203, 211, 213, 225, 235, 245, 261, 273, 277, 281, 283 of Receiver 1 also have similar relationships between the results of the monolithic STS synthesis and the multi-level STS synthesis. It shows that the results of the monolithic STS synthesis are consistent with those of the multi-level STS synthesis.

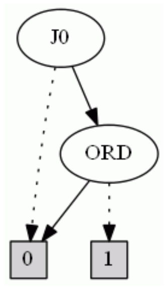


(a)

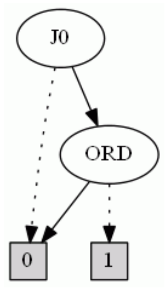


(b)

Fig. 25 The control function of event 103 in (a) the monolithic STS synthesis, (b) the multi-level STS synthesis for Node 1

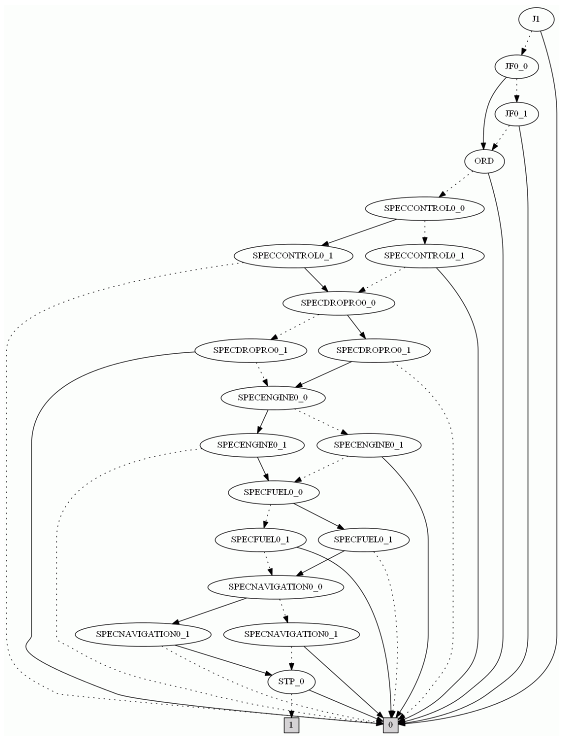


(a)

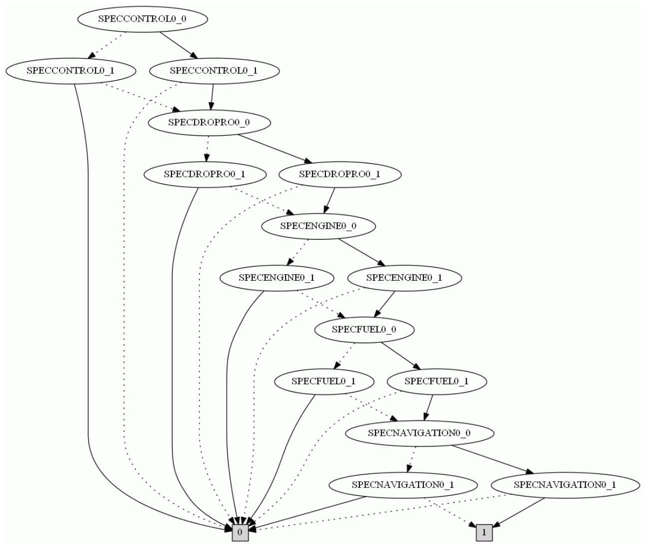


(b)

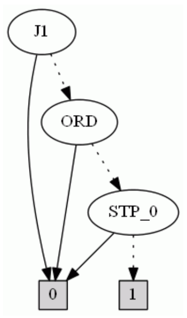
Fig. 26 The control function of event 125 in (a) the monolithic STS synthesis, (b) the multi-level STS synthesis for Node 2



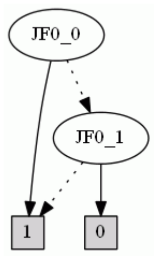
(a)



(b)



(c)

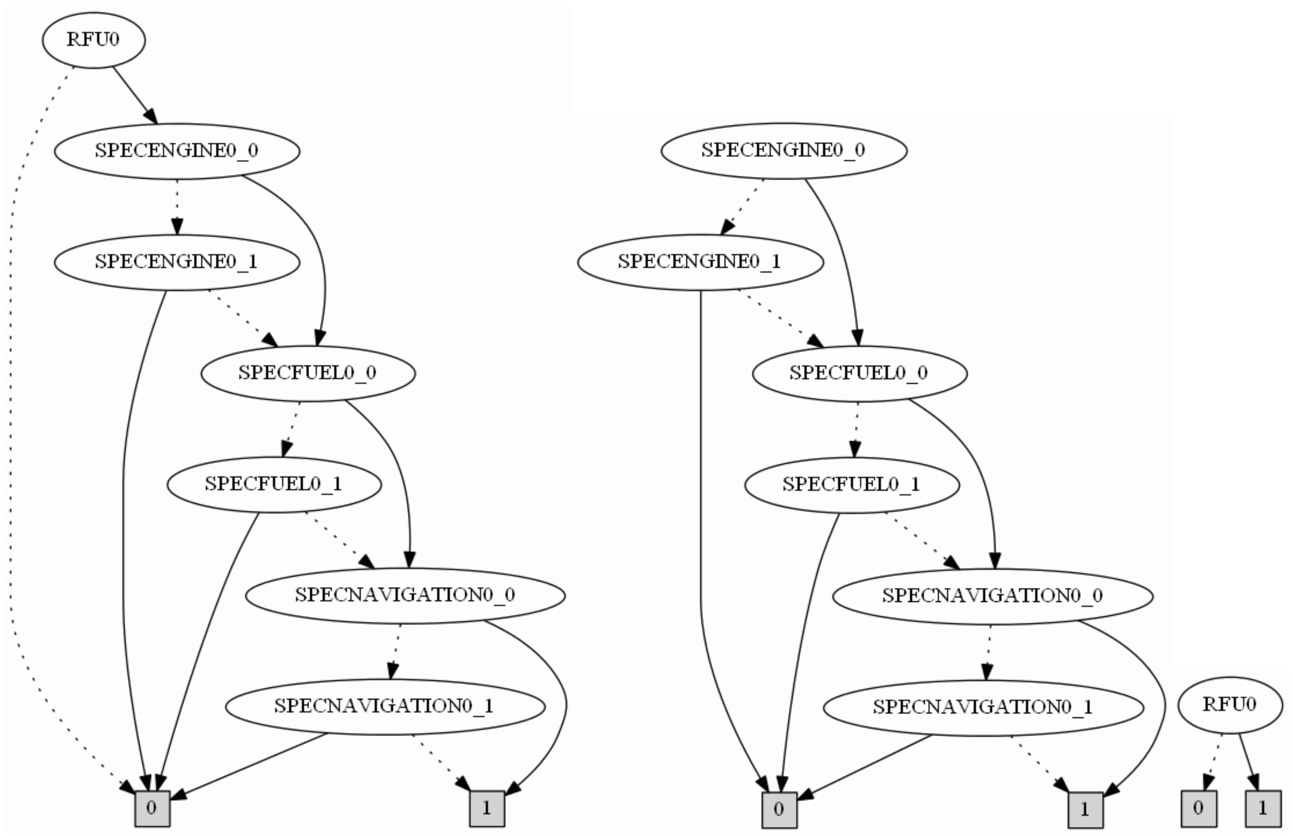


(d)

Fig. 27 The control function of event 105 in (a) the monolithic STS synthesis, (b) the multi-level STS synthesis for Node 1, (c) the multi-level STS synthesis for Node 2, (d) the multi-level STS synthesis for Node 3

An example for Case 2 on Page 23 is the control function of event 105. In Table 11, event 105 is constrained in the multi-level STS synthesis for multiple nodes that are Nodes 1, 2 and 3. Event 105 is independent of the modules involved in Node 4. Fig. 27 gives the BDD results of event 105 in the monolithic STS synthesis and the multi-level STS synthesis.

In the BDD results in Fig. 27, event 105 is constrained according to different specifications. In Fig. 27(b), event 105 should be enabled based on the specifications of the subsystem safety management strategy. In Fig. 27(c) and (d), event 105 should be enabled based on the specification of the mutual exclusion control strategy. We can obtain that $P_0^{[105]} = P_1^{[105]} \vee P_2^{[105]} \vee P_3^{[105]}$ and $\neg P_0^{[105]} = \neg P_1^{[105]} \wedge \neg P_2^{[105]} \wedge \neg P_3^{[105]}$. Another example is event 133 in Fig. 28. $P_0^{[133]} = P_1^{[133]} \vee P_2^{[133]} \vee P_3^{[133]} = P_1^{[133]} \vee P_2^{[133]} \vee 0 = P_1^{[133]} \vee P_2^{[133]}$ and $\neg P_0^{[133]} = \neg P_1^{[133]} \wedge \neg P_2^{[133]} \wedge \neg P_3^{[133]} = \neg P_1^{[133]} \wedge \neg P_2^{[133]} \wedge 1 = \neg P_1^{[133]} \wedge \neg P_2^{[133]}$. Similar relationships can be obtained for events 107, 109, 121, 123, 131, 141, 143 of Receiver 0. Events 205, 207, 209, 221, 223, 231, 233, 241, 243 of Receiver 1 also have similar relationships between the results of the monolithic STS synthesis and the multi-level STS synthesis. The results of the monolithic STS synthesis are consistent with the combination of controlled results for multiple nodes in the multi-level STS synthesis.



(a) (b) (c)

Fig. 28 The control function of event 133 in (a) the monolithic STS synthesis, (b) the multi-level STS synthesis for Node 1, (c) the multi-level STS synthesis for Node 2

6 Conclusion

It is fast and consistent to finish the STS synthesis of the autonomous air refueling problem using the multi-level STS method. Based on the proposed plant modules and specifications, we can obtain non-empty optimal controlled behavior of each node in the multi-level STS. The computation time cost of the multi-level STS synthesis is less than the monolithic STS synthesis. The multi-level STS method is a good method to solve the large-size autonomous air refueling problem for multiple UAVs that cannot be solved directly using the monolithic control method in TCT and STS.

It is worth using the multi-level STS method to deal with the autonomous air refueling problem for multiple UAVs with more subsystems. The analysis in Section 5 shows that the multi-level STS method can handle the problem with 2.0×10^{18} states and solve the problem in less than 100s (**92.5038+0.0149601+0.00398493+0.00398993 < 92.53s**). With the same state space size, the monolithic STS synthesis costs more than 3000s. The multi-level STS method or the multi-level STS synthesis is more efficient than the monolithic STS synthesis.

The contributions of the proposed method in the above sections are twofold. First, we considered the autonomous aerial refueling problem with two aerial vehicles. A mutual exclusion control strategy is proposed to prevent the collision of two aerial vehicles. Second, a multi-level STS method is applied to solve the large-scale problem with two aerial vehicles.

Reference

[1] NATO Standardization Agency, Air to air refueling, ATP-3.3.4.2, April 2019.

[2] Ke Dong, Quan Quan, W. Murray Wonham, Failsafe mechanism design for autonomous aerial refueling using state tree structures, Unmanned Systems, 7(4), 2019, 261-279.

[3] Goorden, M., van de Mortel-Fronczak, J., Reniers, M., Fokkink, W., & Rooda, J. (2019). Structuring Multilevel Discrete-Event Systems With Dependence Structure Matrices. IEEE Transactions on Automatic Control, 65(4), 1625-1639.