Contents lists available at ScienceDirect

Chemical Engineering Research and Design

journal homepage: www.elsevier.com/locate/cherd

# An automata based hybrid modeling approach to synthesize sequential diagnostic tests

*Shih-Ting Feng, Yi-Chung Chen, Chuei-Tin Chang**

*Department of Chemical Engineering, National Cheng Kung University, Tainan, 70101, Taiwan*

ARTICLE INFO

*Article history:*
Received 28 October 2018
Received in revised form 11 February 2019
Accepted 20 February 2019
Available online 11 March 2019

*Keywords:*
Untimed automaton
Diagnostic test
Dynamic simulation
Hybrid model
Sequential operation

ABSTRACT

Although the automata based diagnostic tests have already been developed in the past for differentiating the originally inseparable fault origins in simple chemical processes, their applicability in realistic systems is still questionable. This is primarily due to the facts that automata are basically modeling tools for the discrete event systems (DESs) but most process variables are continuous. To address this practical issue, the dynamic behavior of every processing unit involved in a given operation is modeled in this work by incorporating both the generic engineering knowledge and also the ASPEN-generated dynamic simulation data into a single automaton. The improved test plans can then be synthesized according to the system model obtained by assembling all such automata. The feasibility of this automata based model building strategy is demonstrated with two examples concerning the startup operations of a flash process and also a distillation column.



## 1. Introduction

Unexpected faults and failures in a chemical plant often result in catastrophic consequences. The offline practices of hazard assessment can reduce the total expected loss of accidents only to a certain extent, while fault diagnosis is an alternative way for further improving operational safety. Numerous effective modeling methods have already been proposed to facilitate diagnosis according to the online measurements obtained after fault inception, e.g., see Nomikos and MacGregor (1994; 1995), Chen and Jiang (2011) and Dai and Zhao (2011). On the other hand, Yeh and Chang (2011) showed that another viable approach to improve the diagnostic performance of an existing system is to implement the *test procedures* generated according to automata.

It should be noted from the outset that the automata have been traditionally utilized for modeling the discrete-event systems (DESs) (Debouk et al., 2000; Benveniste et al., 2003; Zad et al., 2003; Qiu and Kumar, 2006; Malik et al., 2011). Although Gascard and Simeu-Abazi (2013) and Gomes Cabral et al. (2015) constructed fault diagnosers according to such models, these automata-based approaches preclude the potentially useful sequential tests for pinpointing fault origins. Yeh and Chang (2011) proposed to use a trial-and-error method to identify extra operation steps needed for enhancement of diagnostic resolution, while Kang and Chang (2014) developed a systematic procedure to search for the optimal test plans with the software DESUMA. Although two subsequent studies have also been carried out to address various practical issues related to diagnostic tests (Hsieh and Chang, 2016; Wang et al., 2017), all of them dealt with only simple material-handling processes and, thus, it may not always be feasible to apply these approaches directly to realistic systems that involve coupled heat and mass transfer processes. This drawback can be primarily attributed to the fact that automata are basically modeling tools for the DESs while most process variables are continuous.

To rectify the aforementioned deficiency, a hybrid modeling strategy has been developed in the present work to synthesize credible operating procedures needed for realistic diagnostic tests. Specifically, the extended finite automata (Åkesson et al., 2006) have been adopted to facilitate model building and procedure synthesis. An extensive set of new configuration techniques have been proposed to incorporate both the simulation data generated by commercial software, e.g., ASPEN Plus Dynamics, and also the generic engineering knowledge into the system automaton. The simulation data are used to construct a verifiable model that characterizes the normal operation succinctly. This is because of the fact that only generic engineering knowledge may not be specific enough for depicting a manageable number of event paths in a complex dynamic system. On the other hand, since not all possible failure-induced scenarios can be exhaustively simulated in advance,

* *Corresponding author.*
E-mail addresses: ctchang@mail.ncku.edu.tw, chueitinchang@msn.com (C.-T. Chang).
https://doi.org/10.1016/j.cherd.2019.02.033

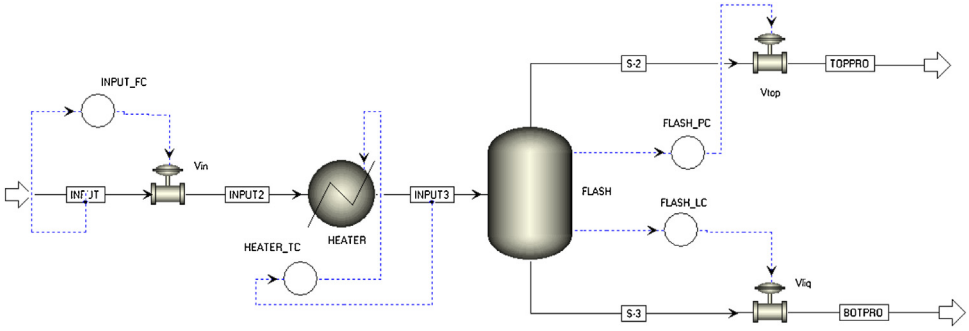



**Fig. 1 – Process flow diagram of flash process.**

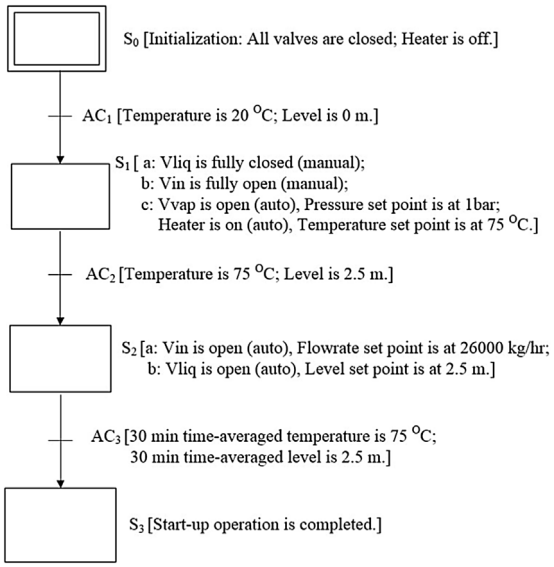



**Fig. 2 – Sequential function chart for startup operation of flash process.**

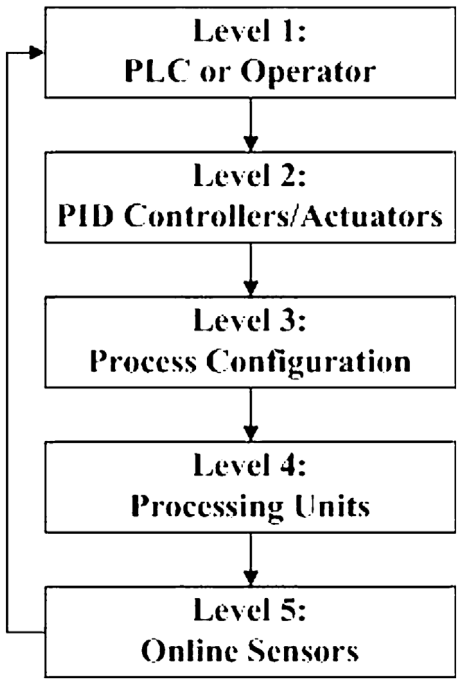



**Fig. 3 – Hierarchical structure needed to facilitate sequential operations.**

the knowledge based models are utilized to predict the fault propagation behaviors during diagnostic tests. The feasibility and effectiveness of this hybrid modeling approach are demonstrated in this paper with two examples, that is, the startup operations of a two-component flash drum and a three-component distillation column.

## 2. Model building strategy

### 2.1. *An illustrative example*

In order to clearly illustrate the proposed modeling strategy, let us consider the flash startup operation described in Fig. 1 (process flow diagram, PFD) and Fig. 2 (sequential function chart, SFC) as an example. In this system, there are a heater (HEATER), an inlet valve (Vin), two outlet valves (Vvap and Vliq) and four PID controllers (INPUT_FC, HEATER_TC, FLASH_PC and FLASH_LC) for controlling the feed rate and temperature, and the vapor pressure and liquid level in the flash drum, respectively. It is assumed that, at steady state, the feed is a mixture of 30 wt% $H_2O$ and 70 wt% methanol and its flowrate, temperature and pressure are kept at 1000 kmol/hr, 20°C and 1.1 bar, respectively. In addition, the initial conditions are set as follows:

- Vin, Vvap and Vliq are all closed, while the heater is off;
- All controllers are on manual;
- Flash drum is empty and at room temperature.

Finally, let us limit the scope of our consideration to only six failures in this example for simplicity, i.e.,

1 F1 (f_VinSC): Vin sticks at close position;
2 F2 (f_H_failed): Heater failure resulting in stoppage of energy output;
3 F3 (f_leaking): Flash drum leaks;
4 F4 (f_VinSO): Vin sticks at open position;
5 F5 (f_VliqSC): Vliq sticks at close position;
6 F6 (f_H_setpoint): An erroneous set point of temperature controller HEATER_TC causing an excessively large heat flow.

### 2.2. *Hierarchical system structure*

All sequential operations can be performed within a hierarchical system structure (see Fig. 3). A total of five levels can always be identified, i.e., (1) programmable logic controller (PLC) or operator, (2) PID controller and actuator, (3) process configuration, (4) processing units, and (5) online sensors. The components in the aforementioned flash process can be classified accordingly as follows:

- Level 1: PLC;

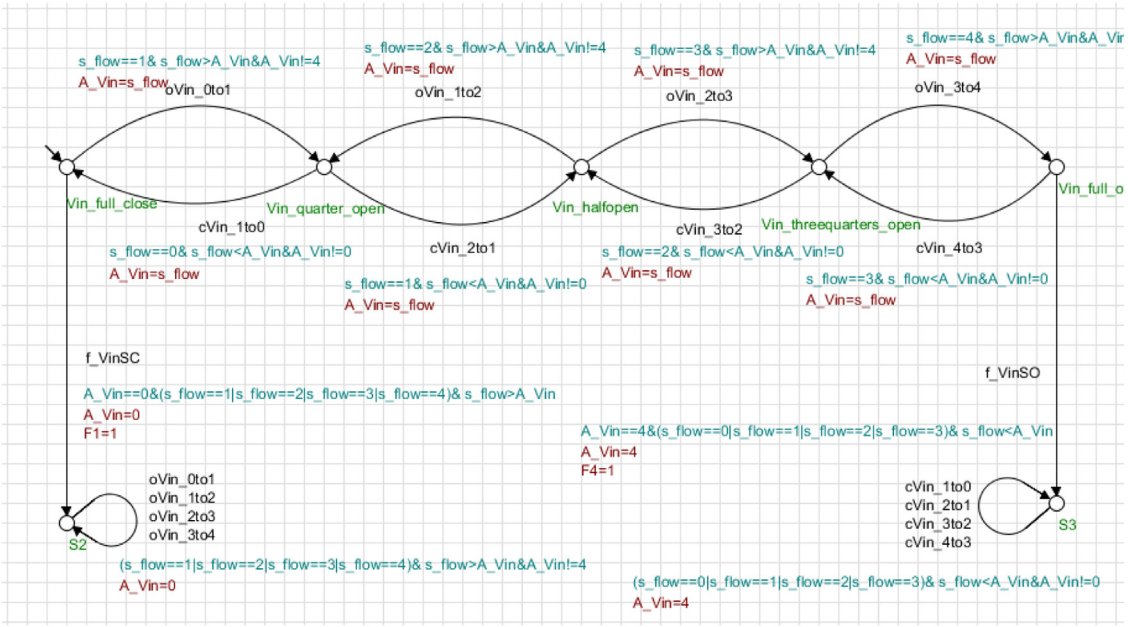



**Fig. 4 – Traditional automaton model of INPUT_FC/Vin in flash startup example.**

- Level 2: INPUT_FC/Vin, HEATER_TC/Heater, FLASH_PC/Vvap, FLASH_LC/Vliq;
- Level 3: material input and output flows and energy input flow;
- Level 4: flash drum;
- Level 5: level, temperature and pressure sensors.

### 2.3. *Automata built with engineering knowledge*

Every component of a process can be modeled with an untimed automaton according to the generic engineering knowledge. A simple ad hoc construction procedure is briefly summarized below (Wang et al., 2017):

All normal states of the component under consideration must be first enumerated and, if there is a need to analyze the effects of its failure(s), all corresponding failed states should also be taken into account. Each normal or failed state is treated as a distinct place in the automaton, and the place representing initial state should be marked with an input arrow. Every realizable state-transition event should be introduced into the automaton by connecting the corresponding states with a directed arc. If necessary, the guard(s) and variable(s) on this arc can also be included to specify its prerequisites and outcomes, respectively.

For the aforementioned flash startup operation, the component models built with the above approach and/or their compressed versions are presented below:

#### 2.3.1. *PID controller and actuator*

To simplify the model representations in level 2, let us treat the PID controller and the corresponding actuator as a single component. The lumped model of INPUT_FC/Vin (see Fig. 4) can be built with the knowledge based approach according to the assumptions that INPUT_FC is direct acting and Vin is air-to-open (A/O). The places Vin_full_close and Vin_full_open in this automaton respectively denote two boundary states of actuator, i.e., the fully closed and open positions of Vin, while the other three places between these two extremes are adopted to represent the intermediate states of 25%, 50% and 75%. Any such state can be driven to a different one by a collection of the state-transition events, i.e., the valve opening actions (oVin_0to1, oVin_1to2, oVin_2to3 and oVin_3to4) and the valve closing actions (cVin_4to3, cVin_3to2, cVin_2to1 and cVin_1to0). It is assumed that Vin is fully closed before the startup operation. Two additional attributes of events, i.e., variable and guard (Åkesson et al., 2006), are also utilized in this automaton. An integer variable can be used to update the component state after completing an event-driven transition, while the guard(s) is used to stipulate the sufficient condition(s) of state transition. Let us consider event oVin_0to1 as an example. Its guards (prerequisites) are expressed as s_flow == 1 &s_flow > A_Vin&A_Vin! = 4 and they can be interpreted as follows

i s_flow == 1: The controller input is adjusted to the qualitative value of 1 and, for simplicity, the controller output is assumed to always follow the input;
ii s_flow > A_Vin: The input/output signal of INPUT_FC is larger than the current air pressure at valve Vin;
iii A_Vin! = 4: The pressure at valve Vin does not reach maximum.

Note that the guards of other events in Fig. 4 can be interpreted in a similar fashion.

In addition, two additional places, Vin_SC and Vin_SO, are included in this model to characterize the failures when Vin is stuck at the closed and open positions, respectively. These failed states can be reached via failure events f VinSC and f_VinSO, and the resulting values of A_Vin should be maintained at 0 and 4 respectively.

Since each component state is described with at least two places in the above automaton, it is clear that the traditional approach is only effective for modeling simple systems with relatively few state variables. To facilitate easy construction and concise presentation of automata, the component models have been "compressed" in this paper. Specifically, the lumped component INPUT_FC/Vin is modeled alternatively with an automaton using significantly fewer places and transitions (see Fig. 5). The place Vin_normal in this compressed model can be obtained by merging all five places that represent normal valve positions in the traditional model. The partial

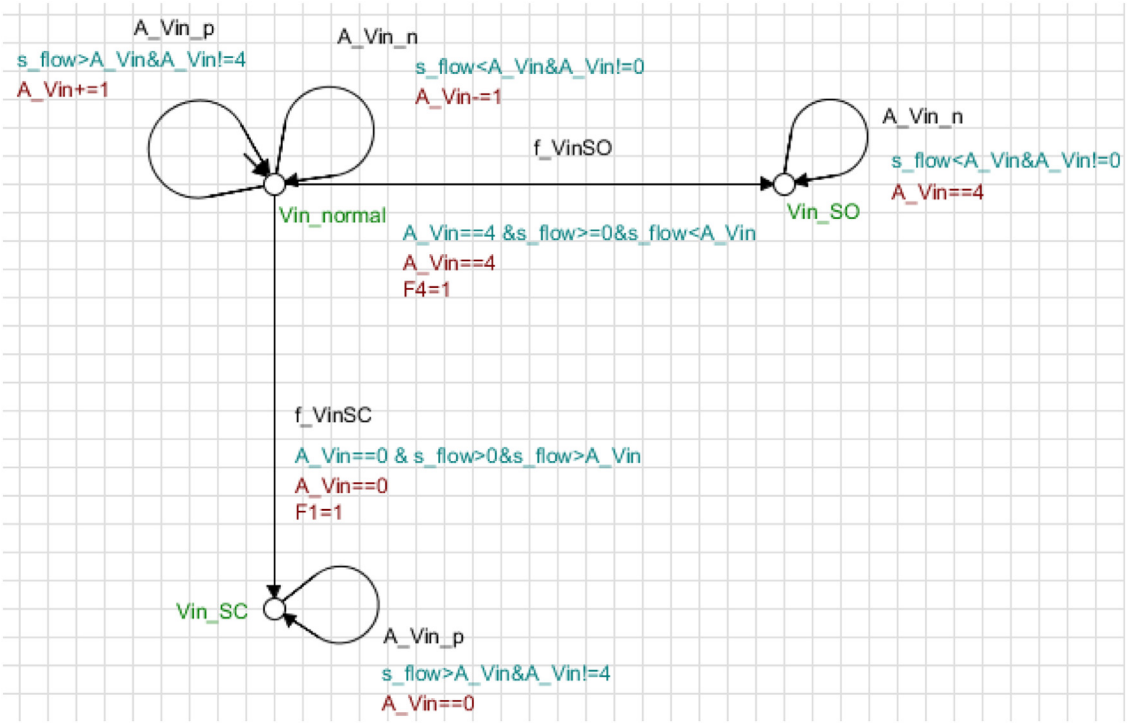



**Fig. 5 – Compressed automaton model of INPUT_FC/Vin in flash startup example.**

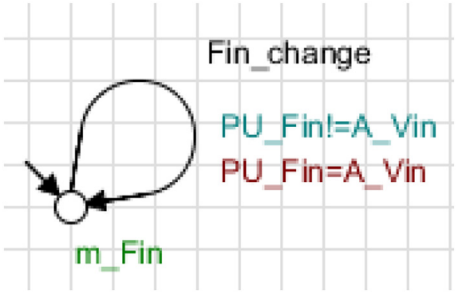



**Fig. 6 – Automaton model of inlet flow (Fin) in flash startup operation.**

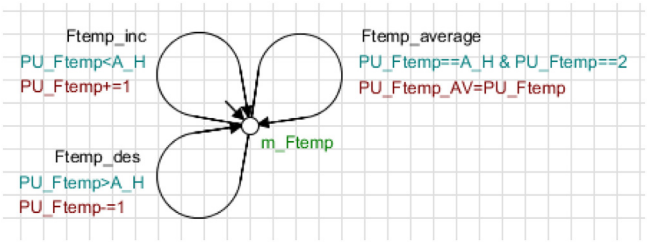



**Fig. 7 – Automaton model of temperature variation in flash drum.**

valve opening and closing actions in Fig. 4 are also combined and represented with two corresponding events (A_Vin_p and A_Vin_n) on the recycle loops of Vin_normal, respectively. The variable A_Vin on either loop is updated according to a C-like code, i.e., A_Vin+ = 1 (or A_Vin = A_Vin + 1) and A_Vin− = 1 (or A_Vin = A_Vin − 1), while the guards can be interpreted in the same way as those in the traditional model. Note that the other level-2 components, i.e., HEATER_TC/Heater, FLASH_PC/Vvap and FLASH_LC/Vliq, can be modelled with the same approach.

### 2.3.2. *Process configuration*

The so-called process configuration is represented in this work with automata to qualitatively describe the material and energy flows that connect major processing units. Since in the aforementioned example there is only a single unit, i.e., the flash drum itself, it is necessary to model its inlet flow (Fin), its outlet vapor and liquid flows (Fvap and Fliq) and its energy input flow from HEATER (Feng). Let us use the automaton in Fig. 6 as an example to illustrate the modeling approach for these connecting flows. Notice first that the guard of Fin_change in Fig. 6 is PU_Fin! = A_Vin, where PU_Fin is the qualitative value of inlet flow rate. In other words, this self-recycle transition is triggered if the inlet flow rate is not the same as the flow rate facilitated by the valve opening. The former should be updated by assuming the latter value right after transition. Finally, it should be noted that the other material and energy flows, i.e., Fvap, Fliq and Feng, can also be modelled with the same approach.

### 2.3.3. *Processing units*

For illustration simplicity and clarity, let us neglect the pressure effects and use only two state variables, i.e., liquid temperature and level, to model the flash drum in the aforementioned example. Let us further assume that, on the basis of prior operational experiences, the entire ranges of temperature and level variations may be discretized into 3 and 4 qualitative states, respectively. The former range is partitioned into $[25^{-}\,^{\circ}C, 25^{+}\,^{\circ}C)$, $[25^{+}\,^{\circ}C, 75^{+}\,^{\circ}C)$ and $[75^{+}\,^{\circ}C, \infty\,^{\circ}C)$ and each is named with a distinct qualitative integer value, e.g., 0, 1 and 2, while the latter may be divided into $[0\,m, 0^{+}\,m)$, $[0^{+}\,m, 2.5^{-}\,m)$, $[2.5^{-}\,m, 2.5^{+}\,m)$, $[2.5^{+}\,m, 4.2^{-}\,m)$ and $[4.2^{+}\,m, \infty\,m)$ and each is labelled with an integer value, e.g., 0, 1, 2, 3 and 4. Thus, the flash drum can be characterized with 15 $(= 3 \times 5)$ states and at most 210 $(= 15 \times 14)$ transitions. Since it is quite difficult to depict and visualize the corresponding automaton obtained with the traditional modeling approach, the compressed automata in Figs. 7 and 8 are used instead in this work to respectively represent the dynamic behaviors of temperature and level in flash operation.

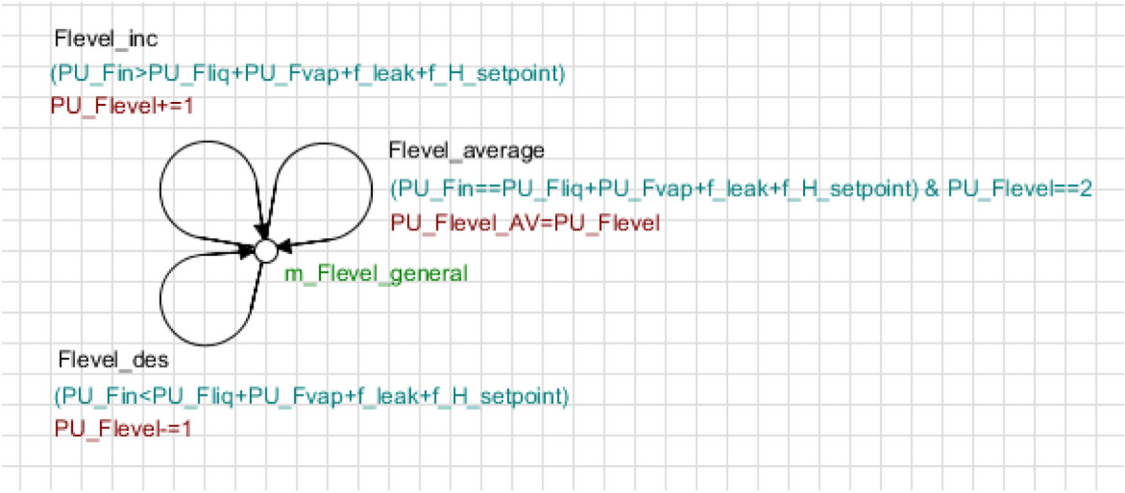



**Fig. 8 – Automaton model of level variation in flash drum.**

Ftemp_inc in Fig. 7 represents the event causing an increase in temperature, while Ftemp_des is the event resulting in a temperature decrease. On the other hand, Ftemp_average is used to model the steady-state scenario. The corresponding causal relations can be summarized as follows:

- Ftemp_inc: If the liquid temperature in flash drum is lower than that of the heating fluid in HEATER (i.e., PU_Ftemp < A_H), then the former should be increased by one unit (PU_Ftemp+ = 1);
- Ftemp_des: If the mixture temperature in flash drum is higher than that of the heating fluid in HEATER (i.e., PU_Ftemp > A_H), then the former should be decreased by one unit (PU_Ftemp− = 1);
- Ftemp_average: If in a specified period of time (1) the mixture temperature in flash drum always equals that of the heating fluid in HEATER (i.e., PU_Ftemp == A_H) and also (2) this temperature stays at the designated steady-state value of 2 (PU_Ftemp == 2), then the time-averaged temperature should be at the same value of 2 (PU_Ftemp_AV = PU_Ftemp).

Similarly, Flevel_inc in Fig. 8 clearly denotes the event resulting in a level increase, while Flevel_des an event producing the opposite effect. The third event Flevel_average is used to characterize the steady-state operation in which material balance is satisfied. The sufficient conditions of these three events and their outcomes are specified in the corresponding guards and variable updates respectively. More specifically, these causal relations can be described as follows:

- Flevel_inc: If the inlet flow rate is greater than the sum of liquid and vapor product flow rates and leak rate (i.e., PU_Fin > PU_Fliq + PU_Fvap + f_leak), then the height of liquid level should be increased by one unit (PU_Flevel+ = 1);
- Flevel_des: If the inlet flow rate is less than the sum of liquid and vapor product flow rates and leak rate (i.e., PU_Fin < PU_Fliq + PU_Fvap + f_leak), then the height of liquid level should be decreased by one unit (PU_Flevel− = 1);
- Flevel_average: If in a specified period of time (1) the inlet flow rate always equals the sum of liquid and vapor product flow rates and leak rate (i.e., PU_Fin == PU_Fliq + PU_Fvap + f_leak) and also (2) the liquid level stays at the designated steady-state value of 2 (PU_Flevel == 2), then the time-averaged liquid level should be at the same value of 2 (PU_Flevel_AV = PU_Flevel).

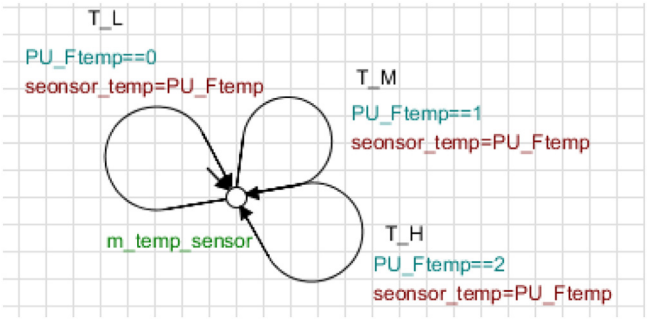



**Fig. 9 – Automaton model for temperature sensor in flash drum.**

### 2.3.4. *Sensors*

Each sensor can also be modelled with a compressed automaton. Every measurement-taking action is represented with a self-recycle loop as shown in Fig. 9 for temperature sensor in the flash drum example. As mentioned in the above subsection, the entire range of temperature variation is discretized into 3 intervals. A distinct self-recycle loop is introduced into the sensor model to represent each of these measurement taking actions. To simplify discussion, sensor failures are excluded in the present example and it is assumed that the sensor measurements always truthfully reflect the temperature and level in flash drum. Finally, note that other sensor models can be built with exactly the same approach.

### 2.3.5. *PLC or operator*

The PLC (or operator) model can be constructed according to the given sequential function chart (SFC). The operation steps in SFC are represented with the corresponding events in an automaton, while the activation conditions of these steps are specified in the event guards. The actuator states reached after executing a particular group of simultaneous steps in SFC should be explicitly stipulated in the variable updates of the corresponding transition. The precedence order of activation conditions and operation steps in the given SFC should be exactly the same as that of guards and events in the automaton.

By following the aforementioned principles, one should be able to transform the SFC in Fig. 2 into the automaton

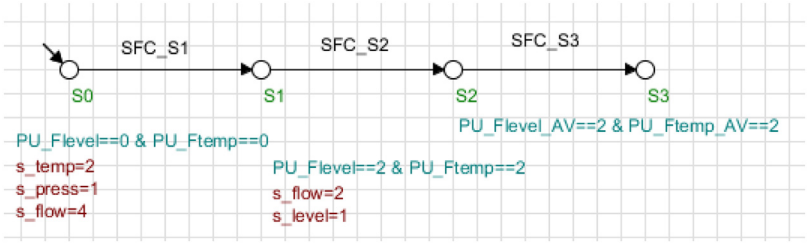



**Fig. 10 – Automaton model of operating procedure in flash startup example.**

in Fig. 10. It can be observed from Fig. 2 that a time period of at least 30 min is required to confirm activation condition $AC_3$ after completing the operation steps in $S_2$,. Since the elapsed time cannot be expressed explicitly in an untimed automaton, the wait action required in $S_3$ is treated as a fictitious step and it is reflected with the self-recycle loops in Figs. 7 and 8, i.e., Flevel_average and Ftemp_average. These two events should take place after implementing $S_2$when s_flow = 2 and s_level = 1, and the level and temperature updates, i.e., PU_Flevel_AV == 2 and PU_Ftemp_AV == 2, can be regarded as the results of online measurements and calculations needed for confirmation of steady state and, consequently, they are used as the guards of SFC_F3 for terminating the startup operation.

### 2.3.6. *System hierarchy*

To impose the causal relationships implied by the generic system hierarchy (see Fig. 3), it is necessary to build an automaton accordingly. Fig. 11 shows this model for the flash startup operation. Notice that there is a one-to-one correspondence between the five places in this automaton and the hierarchical levels in Fig. 3, and that the events allowed in each level are specified on the self-recycle loop at the corresponding place. All such events are listed and discussed below

- Place layer1: SPC_S1, SPC_S2 and SPC_S3 represent the operation steps $S_1$, $S_2$ and $S_3$ in SFC (see Figs. 2 and 10);
- Place layer2: Two types of events are included here. The normal actuator movements, i.e., A_Vin_n, A_Vin_p, A_Vliq_n, A_Vliq_p, A_H_n and A_H_p, can be classified as events of the first type, while the abrupt failures of actuator and/or PID controller, i.e., f_Vin_SC (F1), f_Vin_SO (F4), f_Vliq_SC (F5), f_H_failed (F2) and f_H_setpoint_fails (F6), should be regarded as the second-type events.
- Place layer3: The system configuration is manipulated by varying the material and energy flows among processing units. For the flash drum, the inlet and outlet material flows are adjusted with two correspnding events, i.e., Fin_change and Fliq_change, and the heat flow from heater to flash drum is controlled via Energy_output_change.
- Place layer4: The variations of operating conditions of the flash drum, i.e., level and temperature, are characterized with Flevel_des, Flevel_inc, Ftemp_des and Ftemp_inc. As mentioned before, the vessel failure f_leaking (F3) should also be included as an event in level 4.
- Place layer5: The events here can be regarded as dsicretized measurement-taking actions, i.e., L_LL, L_L, L_H, L_HH, T_L, T_M and T_H.

### 2.3.7. *Path explosion*

The system model can be synthesized by integrating all automata developed in the previous six subsections via the parallel composition operation (Cassandras and Lafortune, 2008). Although several equally effective tools are available, the free software package SUPREMICA (Åkesson et al., 2006) has been adopted in the present work to perform this synchronization task. Since only the generic engineering knowledge is utilized to build qualitative component models, an overwhelmingly large number of paths (strings) may be extracted from this integrated automaton even when the system dynamics is moderately complex.

To illustrate this drawback more clearly, let us consider the flash startup procedure under normal operating conditions only. After removing all failures and failed states from the aforementioned component models, the parallel composition operation can be carried out to produce the complicated path network given in Figure S.1 in Supplementary material. This undesirable phenomenon of path explosion can be attributed to the fact that, since the dynamic behavior of any MIMO system cannot be adequately described with the untimed automata developed on the basis of generic engineering knowledge only, all unspecified combinations are exhaustively enumerated. For example, as indicated in the SFC in Fig. 2, the level and temperature in flash drum are supposed to be raised to 2.5 m and 75 °C respectively (i.e., $AC_2$) after implementing the operation steps in $S_1$. However, since the precedence order of level and temperature increases is not given, all possible paths ending at these targets are identified according to the fundamental definition of parallel composition. Thus, it is obvious that some of the paths generated by SUPREMICA may not be real and, furthermore, the operationally infeasible *deadlocks* and/or *livelocks* (Cassandras and Lafortune, 2008) may even be present in this unrealistic path network.

## 2.4. *Automata built with simulation and/or historical data*

From the above discussions, it is clear that the generic models mentioned above must be further constrained to facilitate effective identification of the critical path to be followed in a diagnostic test plan. To this end, additional automata are constructed according to simulation and/or historical data for better representations of the dynamic behaviors of processing units under *normal* operating conditions. This modeling strategy can be justified on the basis of the following observations:

- The undesired path explosion can be mainly attributed to the complex dynamic behaviors of processing units during normal operations;
- Every failure-induced abnormal trace is always emanated from a particular normal state and there can be a large number of paths leading to this normal state;
- Simulation and historical data of fault propagation scenarios are rarely available, while such data are abundant for the normal processes.

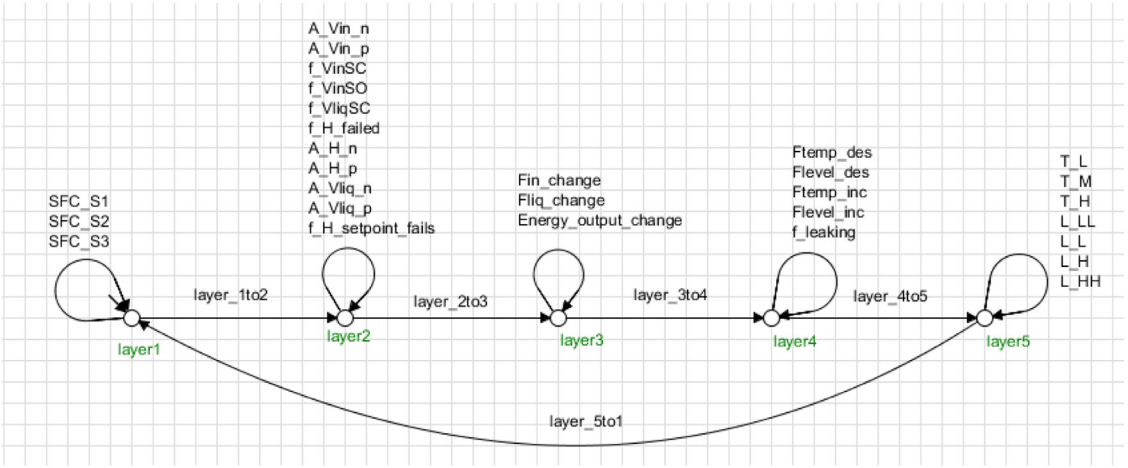



**Fig. 11 – Automaton representing system hierarchy.**

**Table 1 – Discretized mass and energy flowrates for configuring flash process.**

| Interval | PU_Fin (kg/hr) | PU_Fliq(kg/hr) | PU_Fvap(kg/hr) | PU_Feng(MMkcal/hr) |
|---|---|---|---|---|
| 0 | $[0, 0^+)$ | $[0, 0^+)$ | $[0, 0^+)$ | $[0, 0^+)$ |
| 1 | $[0^+, 26000^-)$ | $[0^+, 13000^-)$ | $[0^+, 13000^-)$ | $[0^+, 3.0)$ |
| 2 | $[26000^-, 26000^+)$ | $[13000^-, \infty)$ | $[13000^-, \infty)$ | [3.0, 5.0) |
| 3 | $[26000^+, 50000)$ | Undefined | Undefined | $[5.0, \infty)$ |
| 4 | $[50000, \infty)$ | Undefined | Undefined | Undefined |

**Table 2 – Discretized simulation data in normal startup operation of flash process.**

| Time | PU_Ftemp | PU_Flevel | PU_Feng | PU_Fin | PU_Fliq | PU_Fvap |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0.01 | 0 | 0 | 2 | 4 | 0 | 0 |
| 0.03 | 1 | 0 | 2 | 4 | 0 | 0 |
| 0.09 | 1 | 1 | 2 | 4 | 0 | 0 |
| 0.55 | 1 | 1 | 2 | 4 | 0 | 0 |
| 0.65 | 2 | 1 | 2 | 4 | 0 | 0 |
| 0.89 | 2 | 1 | 2 | 4 | 0 | 1 |
| 1.05 | 2 | 2 | 2 | 2 | 1 | 1 |
| 2.05 | 2 | 2 | 2 | 2 | 1 | 1 |
| 3 | 2 | 2 | 2 | 2 | 1 | 1 |

**Table 3 – State-transition events in normal startup operation of flash process.**

| | Guard | | | | | | Variable | |
|---|---|---|---|---|---|---|---|---|
| | PU_Ftemp | PU_Flevel | PU_Feng | PU_Fin | PU_Fliq | PU_Fvap | db_Ftemp | db_Flevel |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 0 | 2 | 4 | 0 | 0 | 1 | 0 |
| 3 | 1 | 0 | 2 | 4 | 0 | 0 | 0 | 1 |
| 4 | 1 | 1 | 2 | 4 | 0 | 0 | 1 | 0 |
| 5 | 2 | 1 | 2 | 4 | 0 | 1 | 0 | 1 |
| 6 | 2 | 2 | 2 | 2 | 1 | 1 | 0 | 0 |

To facilitate clear illustration of the above model building strategy, let us again consider the flash startup operation described in Figs. 1 and 2. Figs. 12–15 show the corresponding simulation data generated with ASPEN PLUS DYNAMICS, and these continuous data must be first discretized into several discrete intervals. The discretization schemes of the state variables of processing unit, i.e., the liquid level and temperature of flash drum, have already been given previously in Subsection 2.3.3, while those of the material and energy input and/or output flows, can be found in Table 1. Based on these discretization schemes, all simulation data in Figs. 12–15 can be converted to their qualitative values in Table 2. An abridged version of this data set can be produced next by removing

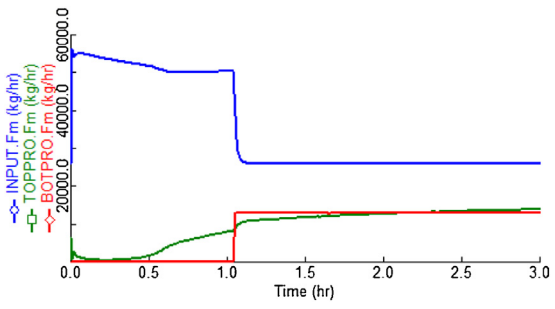



**Fig. 12 – Simulated material flow rates in flash startup operation.**

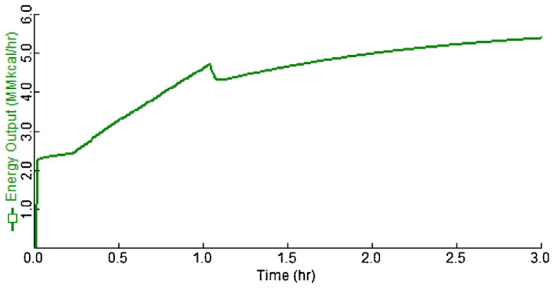



**Fig. 13 – Simulated energy flow rate in flash startup operation.**

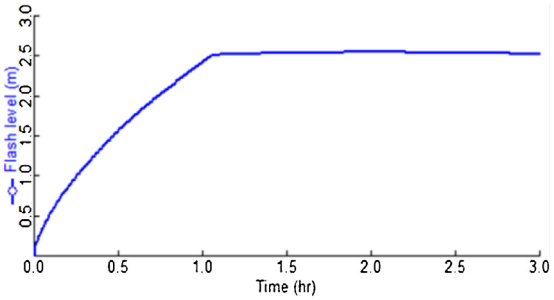



**Fig. 14 – Simulated liquid level in drum in flash startup operation.**

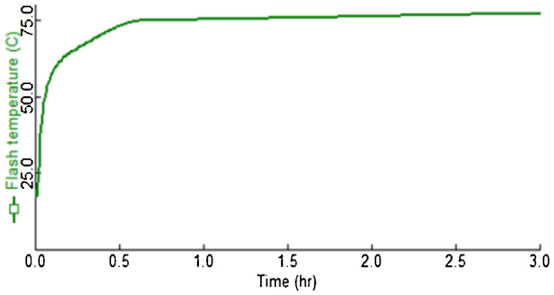



**Fig. 15 – Simulated liquid temperature in drum in flash startup operation.**

every row in which the state variables, i.e., temperature and level, are identical to those in the previous row. All state changes during startup operation can be easily extracted from this abridged set and they are incorporated in the automaton model as a sequence of state-transition events. A total of six (6) consecutive events were found in the present example and their guards and variable updates are listed in Table 3. An alternative graphical representation of the corresponding automaton is presented in Figure S.2 in Supplementary material.

### 2.5. *Hybrid models*

- Since simulation and historical data of fault propagation scenarios are often not available in practical applications, it becomes necessary to predict the fault propagation paths based solely on the engineering knowledge in these situations. Thus, for characterizing the normal and abnormal operation modes of every processing unit in different scenarios, the data-based model and its knowledge-based counterpart are both incorporated into a hybrid automaton in the present study. In the flash startup example, the switch actions from the data-based models of the normal operations to the knowledge-based models after inception of failure(s) are triggered with the automata shown in

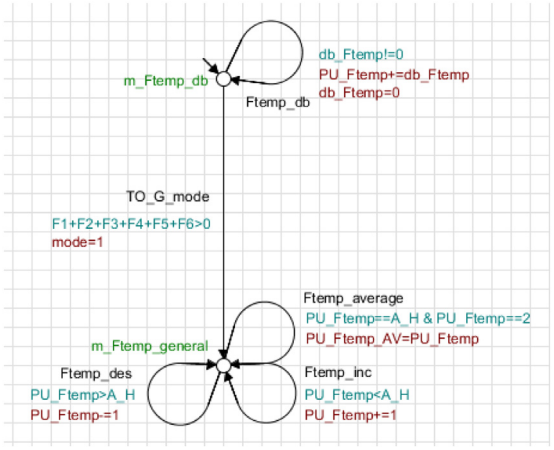



**Fig. 16 – Switching mechanism for predicting temperature variation.**

Figs. 16 and 17. The implied mechanisms are summarized below:

- The state m_Ftemp_db in Fig. 16 refers to the data-based temperature prediction model, while m_Ftemp_general denotes the knowledge-based counterpart. Since the system is assumed to be normal initially, the former is adopted to determine the temperature variation during the beginning stage of startup operation. Notice that the guard of event Ftemp_db can be satisfied after finishing the transitions specified in the 2nd and 4th rows of Table 3 (or the corresponding events db_2 and db_4 in Figure S.2 in Supplementary material). However, if one or more failure occurs at an instance between any two consecutive transitions, i.e., $F1 + F2 + F3 + F4 + F5 + F6 > 0$, the switch action To_G_mode can be triggered to start utilizing the knowledge-based model for predicting temperature. It should also be noted that this model is essentially the same as that given in Fig. 7.
- The state m_Flevel_db in Fig. 17 refers to the data-based level prediction approach, while m_Flevel_general denotes the knowledge-based counterpart. Since the initial system state is normal, the former is adopted to determine the level variation after initiation of startup operation. Notice that the guard of event Flevel_db is satisfied after completing the 3rd and 5th transitions in Table 3 (or the corresponding events db_3 and db_5 in Figure S.2 in Supplementary material). Notice that, since F3 (i.e., a leak develops in the flash drum) is a failure of the processing unit under consideration, it is modelled as a self-loop on m_Flevel_db. If at an instance between any two consecutive transitions one or more failure occurs, i.e., $F1 + F2 + F3 + F4 + F5 + F6 > 0$, the switching mechanism To_G_mode can be activated to predict level variation according to the knowledge-based model. It should also be noted that this model is essentially the same as that given in Fig. 8.

  Finally, in order to integrate the above hybrid models for generating the realistic event paths, it is necessary to incorporate the additional events in data-based models into the event list on layer4 of the system hierarchy. For the flash startup example in Fig. 11, these extra events should be db_1 - db_6, Ftemp_db and Flevel_db.

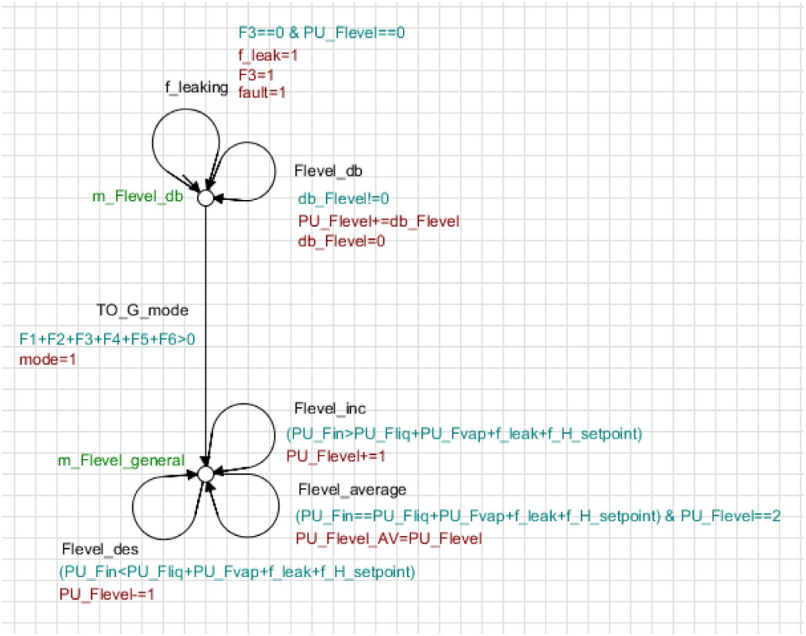



**Fig. 17 – Switching mechanism for predicting level variation.**

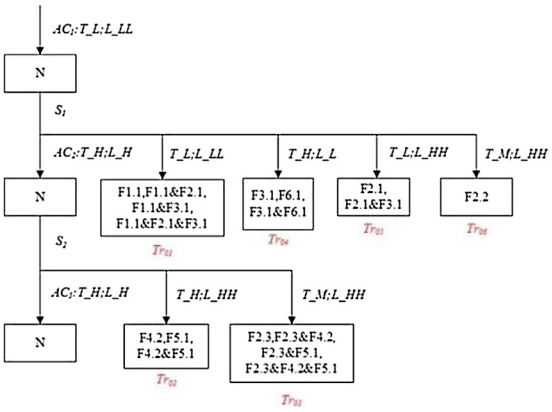



**Fig. 18 – Diagnoser in flash startup example.**

## 3. Observable event traces

Although any number of hardware items may fail at any instance during routine operations, these failures are usually unobservable. Fault diagnosis can be performed to identity the root causes of abnormal system state based on the available online information. It is assumed in this work that the observable events are limited to those associated with actuator actions and sensor measurements. On the basis of this assumption, parallel composition can be applied to integrate all automata mentioned above so as to synthesize a diagnoser (Cassandras and Lafortune, 2008) in which all observable event traces (OETs) are embedded.

Let us again consider the flash startup example for illustration convenience. The aforementioned models can be synchronized in SUPREMICA to produce the diagnoser in Fig. 18. Notice that the normal system states are represented by nodes with label "N" and they are connected by arcs marked with appropriate activation conditions ($AC_1$, $AC_2$ and $AC_3$) and operation steps ($S_1$ and $S_2$). If executing a particular step does not yield the anticipated activation condition, then the corresponding abnormal state should be represented with an additional branch emanating from such step. Notice that more than one state may be generated and they can usually be identified according to the online measurements. It can be observed from Fig. 18 that four OETs emerge after $S_1$ and two after $S_2$. For each OET in this diagnoser, the abnormal measurements are underlined and the corresponding fault origins are listed in the end node. Note that the fault origins of an OET are separated by commas and, if there are multiple failures in a fault origin, then they are connected with the symbol &. All multi-origin OETs in Fig. 18 are clearly not diagnosable, while $Tr_{06}$ is the only exception.

Note that all six failures considered in the flash startup example have been expressed previously in Subsection 2.1 by using a standard format, i.e., Fi ($i = 1, 2, \cdots, 6$). In order to further specify the occurrence time of every failure in each fault origin, it is necessary to attach an additional index to this notation. In particular, a failure in diagnoser is represented in the form of Fi.j and this additional index j ($= 1, 2, 3$) denotes the time interval between the two instances when the consecutive activation conditions $AC_{j-1}$ and $AC_j$ are satisfied. Notice also that, for any given failure, not all intervals are considered in the present example for the purpose of facilitating concise illustration. Following is a list of failure events included in the diagnoser:

- Since F1 and F5 are both concerned with valve sticking events and all valves are closed initially, their effects can be detected when PLC starts to open Vin and Vliq, respectively. Thus, it is only necessary to analyze the scenarios in which F1 and F5 take place before $AC_1$ and $AC_2$, respectively, and only F1.1 and F5.1 are included in diagnoser for illustration simplicity.
- A heater failure (F2) may be revealed at any time when the heater is on and, thus, the possible events should be F2.1, F2.2 and F2.3.
- A leak in the flash drum (F3) may develop at any time during the startup operation. However, only F3.1 is considered here to simplify the subsequent analysis.
- Since F4 (Vin sticks at the open position) can happen after it is opened manually in $S_1$ and this failure is detectable after the inlet flow controller is switched to the auto mode in $S_2$, it is therefore only necessary to consider F4.2.

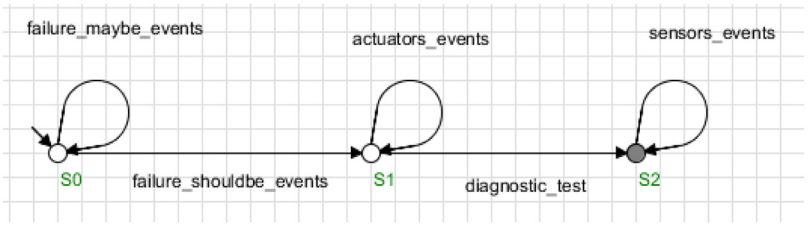



**Fig. 19 – Auxiliary automaton for conjecturing a diagnostic step.**

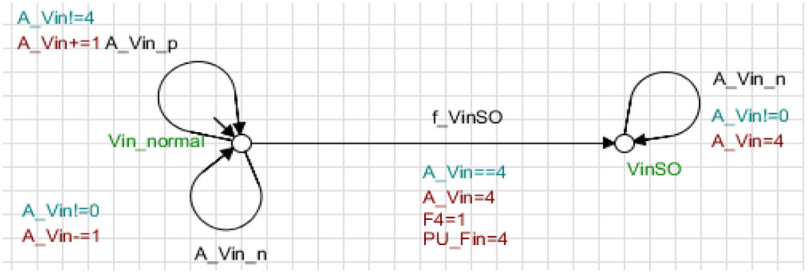



**Fig. 20 – The modified automaton model of INPUT_FC/Vin for $Tr_{02}$.**

- Since F6 is due to an incorrect controller setting, it can be revealed after HEATER_TC is switched to the auto mode at step $S_1$. Consequently, F6.1 should be the only possible failure event in this situation.

## 4. Synthesis of test plans

If more than one fault origin is implied by an OET, a test plan may be devised to enhance diagnostic resolution by implementing a dedicated operating procedure (Kang and Chang, 2014). To this end, a series of structurally identical automata can be built to drive the given system to produce a unique set of sensor measurements for each fault origin of the given OET. Fig. 19 shows the generalized auxiliary automaton for creating a diagnostic step. The general events in this automaton are defined below:

- failure_maybe_events: Dispensable failures implied by the OET under consideration;
- failure_shouldbe_events: Indispensable failures implied by the OET under consideration;
- actuators_events: All possible actuator adjusting actions;
- sensors_events: All possible measurement taking actions;
- disgnostic_test: The guards of this event should be an exhaustive list of all possible combinations of sensor measurements except that associated with the operating conditions when the test procedure begins. In other words, each entry in this list reflects a unique reachable system state from the initial state.

From Fig. 19 it is clear that multiple actuator actions may be required in a single diagnostic step, and it should also be noted that more than one step may be needed in a complete test plan. Kang and Chang (2014) suggested that the total number of steps ($L$) should be bounded from above and below, i.e., $m \leq L \leq M$, and these bounds for a given OET can be determined with the following formulas

$$M = F - 1 \tag{1}$$

$$m = \left\lceil \frac{\ln F}{\ln R} \right\rceil \tag{2}$$

where, F is the number of fault origins; $R$ denotes the total number of online sensors; $\lceil \cdot \rceil$ represents the ceiling operator. Since the above range significantly limits the search space, it is possible to synthesize the actual sequential function chart accordingly with a trial-and-error approach. Since a generalized search procedure can be found in Kang and Chang (2014), it is omitted for the sake of brevity. Instead, let us revisit the diagnoser of flash startup process in Fig. 18 and consider only trace $Tr_{02}$ as an example to provide a concise illustration of the test plan synthesis strategy. Specifically, following are the required steps:

i Discard the PLC/operator model in level 1 (see Fig. 10).
ii Identify the final states of all components on $Tr_{02}$, and use them as the initial conditions of the corresponding test procedure. If a component is always normal in this scenario, its state achieved after implementing step $S_2$ can be easily extracted according to the given SFC. On the other hand, since the fault origins of $Tr_{02}$ may be F4 (f_VinSO), F5 (f_VliqSC) or F4&F5 (f_VinSO and f_VliqSC), the failed states of Vin and Vliq cannot be confirmed without the diagnostic tests. Consequently, the initial states of Vin and Vliq in the test plan should be set respectively as follows: A_Vin = 4 (i.e., the inlet valve is normal and fully open) and A_Vliq = 0 (i.e., the outlet valve of the liquid output is normal and fully closed). Consequently, all corresponding component states can be listed below
   - level 2: A_Vin = 4, A_Vliq = 0, A_H = 2;
   - level 3: PU_Fin = 4, PU_Fliq = 0, PU_Fvap = 1, PU_Feng = 2;
   - level 4: PU_Flevel = 3, PU_Ftemp = 2;
   - level 5: sensor_level = 3, sensor_temp = 2.
iii Modify the component models for synthesis of diagnostic tests.
   - Since the SFC of a test plan is supposed to be synthesized with this proposed procedure and therefore the level-1 component is unavailable a priori, let us start with the level-2 components. The original automaton model of INPUT_FC/Vin in Fig. 5 can now be tailored to a dedicated version for $Tr_{02}$ (see Fig. 20). Because of the fact that the diagnostic steps are concerned with the actuator actions only, the above automaton may be simplified by elimi-

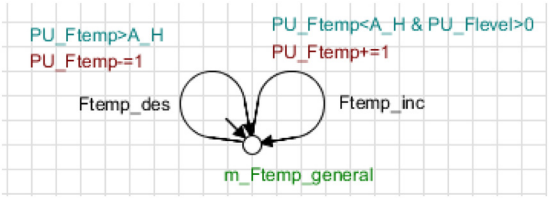



**Fig. 21 – The modified automaton model of drum temperature variation for $Tr_{02}$.**

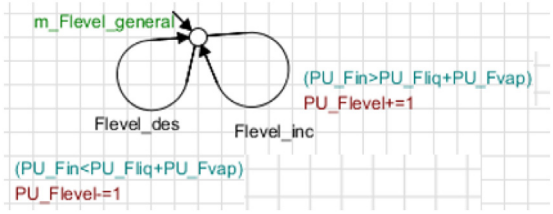
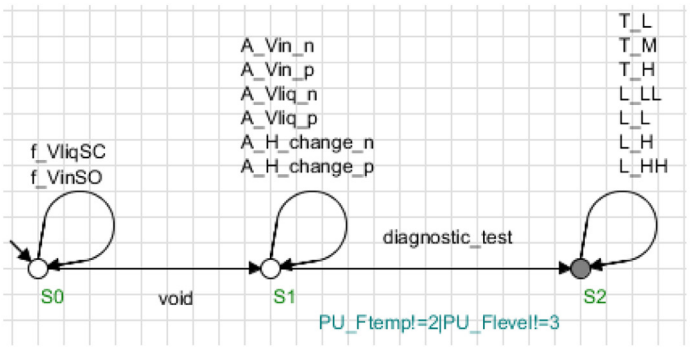



**Fig. 22 – The modified automaton model of drum level variation for $Tr_{02}$.**

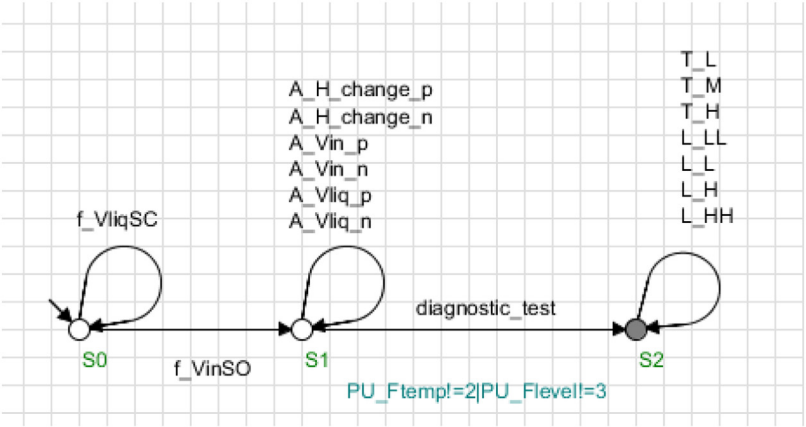
nating all specifications on controller input/output, i.e., all logic constraints concerning s_flow. In addition, based on the observation that the implied failures of $Tr_{02}$ do not include F1 (f_VinSC), the corresponding places and arcs can be removed from the original model in Fig. 5. On the other hand, since the presence (or absence) of failure F4 (f_VinSO) cannot be categorically confirmed after observing $Tr_{02}$, it is necessary to set the initial condition of valve Vin to be that just before this failure, i.e., A_Vin = 4, so as to incorporate all possibilities in diagnostic test. Finally, the other level-2 component models, i.e., those for HEATER_TC/Heater, FLASH_PC/Vvap and FLASH_LC/Vliq, can be modified with the same method and, for the sake of brevity, they are omitted in this paper.

- Since none of the assumed failures in the present example, i.e., F1 – F6, are characterized in the level-3 automata, no modifications (except the initial conditions specified in step 2) are needed to model the process configuration.
- The level-4 automata in Figs. 21 and 22 are the modified versions of Figs. 16 and 17 for modeling the temperature and level variations of flash drum respectively in the diagnostic tests of OET $Tr_{02}$. It is determined in step 2 that, although the drum temperature is still normal (PU_Ftemp = 2) before implementing the test plan, the liquid level is abnormally high (PU_Flevel = 3). Since failure F3 is not included in the fault origins of $Tr_{02}$, the self-looping event f_leaking in Fig. 17 should be removed. Finally, because of the fact that the data-based models are constructed according to simulation results of the normal operations only, they should also be removed from the hybrid models.
- Since the assumed failures in the present example, i.e., F1 – F6, are not concerned with the level-5 components, no modifications (except the initial conditions specified in step 2) are needed to model the online sensors.

iv Construct the auxiliary automata and perform parallel compositions iteratively to produce a unique trace for every




(a) Diagnostic step 1




(b) Diagnostic step 2

**Fig. 23 – Auxiliary automata for diagnostic tests of $Tr_{02}$.**

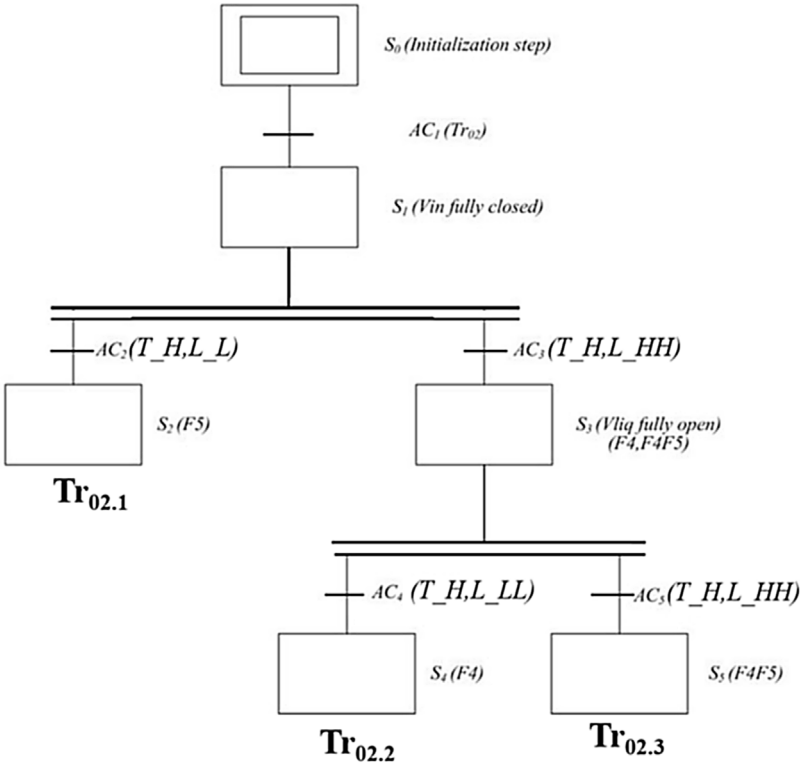



**Fig. 24 – Diagnostic test plan of $Tr_{02}$ in flash startup example.**

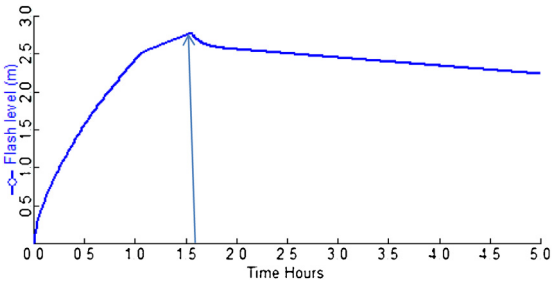



**Fig. 25 – Simulated time profile of level variation for $Tr_{02.1}$.**

fault origin and each ends at a distinct set of online measurements.

- Since the number of fault origins of $Tr_{02}$ is three, i.e., F = 3, the upper bound (M) and lower bound (m) of the step number of the diagnostic tests (L) can be determined according to Eqs. (1) and (2), i.e., M = 2 and m = 1. Therefore at most two auxiliary automata are needed.
- Fig. 23(a) shows the auxiliary automaton used for generating the first diagnostic step. The self-looping events attached on node S0 are failures that may or may not be present, i.e., F4 (f_VinSO) and F5 (f_VliqSC), while those on S1 should be all feasible actuator actions. Finally, the self-looping events on the marked state S2 should include all possible sensor readings. The guardless event between S0 and S1 (void) indicates that no incidences are required to take place, while either of the two guards of the next event between S1 and S2 (disgnostic_test), i.e., PU_Ftemp! = 2 or PUFlevel! = 3, is adopted primarily to prohibit reaching the starting conditions of the test process. By applying parallel composition on all aforementioned modified automata and the auxiliary automaton given in Fig. 23(a), it is possible to isolate only one fault origin, i.e., F5 (f_VliqSC), on a unique trace that represents this first diagnostic step.
- Fig. 23(b) shows the auxiliary automaton used for generating the next diagnostic step. Since the two indistinguishable fault origins before the second diagnostic step are F4 (f_VinSO) and F4&F5 (f_VinSO and f_VliqSC), this automaton can be constructed simply by moving f_VliqSO from the self-looping arc on S0 in Fig. 23(a) and replacing event void with this failure. This practice implies that failure F4 is treated as a certain event.

v Summarize the OETs generated in step iv with a SFC. This test plan for $Tr_{02}$ can be found in Fig. 24.

## 5. Simulation and validation

### 5.1. $Tr_{02.1}$

This OET and the subsequent diagnostic tests were simulated with ASPEN PLUS DYNAMICS and, for simplicity, the implied fault origin F5 (f_VliqSC) was introduced in the beginning of normal operation (at time 0 hr) before condition $AC_1$. Let us consider only the simulated level variation in this scenario (see Fig. 25). A vertically upward arrow is introduced in this figure at 1.55 hr to denote the time of fault detection and the entire horizon is divided into two distinct periods accordingly.

The SFC in Fig. 2 was followed in the first period. Notice that, after observing condition $AC_2$ (i.e., level reaching 2.5 m) at time 1.5 hr and then executing step $S_2$, i.e.,

- switching the level controller to AUTO and adjusting the set point to 2.5 m, and
- switching the flow controller to AUTO and adjusting the set point to 260000 Kg/hr,

the level-rising trend still continued until 1.55 hr due to the presence of failure F5 (f_VliqSC). Since condition $AC_3$ in the normal operating procedure could not be satisfied in this case, the first diagnostic action shown in Fig. 24, i.e., manually clos-

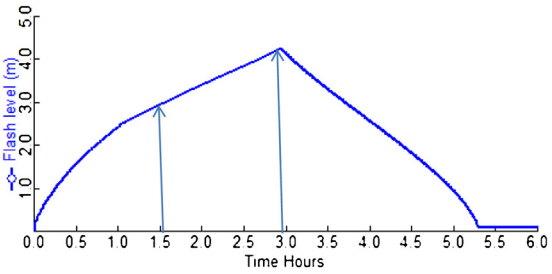



**Fig. 26 – Simulated time profile of level variation for $Tr_{02.2}$.**

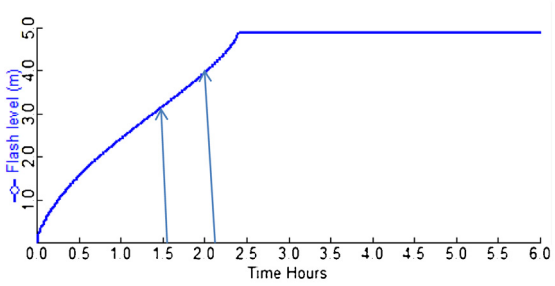



**Fig. 27 – Simulated time profile of level variation for $Tr_{02.2}$.**

ing the inlet valve Vin, was performed next. Since the heater was still on afterwards, the liquid level in flash drum was gradually lowered due to vaporization. This symptom, i.e., $AC_2$ in Fig. 24, confirmed the corresponding fault origin F5.

### 5.2. $Tr_{02.2}$

Fig. 26 shows the simulated level variation resulting from fault origin F4 (f_VinSO). Since F4 is supposed to take place after implementing $S_1$ and before reaching $AC_2$, it was introduced in the simulation run at the earliest possible time near 0 hr. The SFC in Fig. 2 was also followed in the period before the fault detection time at 1.55 hr. After observing $AC_2$ (i.e., level reaching 2.5 m) at time 1.5 hr and then executing $S_2$, i.e.,

- switching the level controller to AUTO and adjusting the set point to 2.5 m, and
- switching the flow controller to AUTO and adjusting the set point to 260000 Kg/hr,

the subsequent level-rising trend was still evident due to the presence of F4 (f_VinSO), i.e., Vin failed at the open position. Since it was confirmed that condition $AC_3$ in the normal operating procedure could not be met after half an hour at 1.55 hr, the first diagnostic action, i.e., manually closing the inlet valve Vin, was taken at once. However, when compared with the level-decreasing trend produced by the same action in the case of $Tr_{02.1}$, it can be concluded that the system responded differently after 1.55 hr in the present scenario. According to $AC_3$ in Fig. 24, the next online measurement to be confirmed is L_HH, i.e., the level is extremely high and, more specifically, L_HH is set to be 4.2 m or higher in this example. As shown by the second arrow in Fig. 26, this chosen criterion was realized at 2.95 hr. Notice that the next diagnostic action in test plan, i.e., $S_3$ in Fig. 24, is to manually open the outlet valve Vliq so as to produce the maximum possible liquid flow. Since the heater was still on at this time, the combined flow rate of vapor and liquid outputs was larger than the input flow rate despite the fact that the inlet valve Vin was stuck at the open position. As a result, the liquid level in flash drum dropped to the next designated level L_LL in $AC_4$ of the test plan (see Fig. 24) at 5.25 hr and, in the present example, L_LL was chosen to be 0 m. In this scenario, observing $AC_3$ and then $AC_4$ confirmed the corresponding fault origin F4.

### 5.3. $Tr_{02.3}$

Fig. 27 shows the combined effects of F4 (f_VinSO) and F5 (f_VliqSC) on the transient behavior of liquid level in flash startup operation. Failure F5 was again introduced in the simulation run at time 0 hr, while F4 at a slightly later time. As in the previous two scenarios, the SFC in Fig. 2 was followed in the period before 1.55 hr. After observing $AC_2$ (i.e., when the liquid level was raised to 2.5 m) at time 1.5 hr and then executing $S_2$ in the normal operating procedure, i.e.,

- switching the level controller to AUTO and adjusting the set point to 2.5 m, and
- switching the flow controller to AUTO and adjusting the set point to 260000 Kg/hr,

the fast level-rising rate was still maintained due to both F4, i.e., Vin stuck at the open position, and F5, i.e., Vliq stuck at the closed position. Therefore, after confirming at 1.55 hr that $AC_3$ in the normal operating procedure could not be satisfied, the diagnostic step $S_1$ in the test plan, i.e., manually closing the inlet valve Vin, was again carried out at this point. According to condition $AC_3$ in Fig. 24, the next online observation in this scenario should be L_HH, i.e., the level reaching 4.2 m or higher. As shown by the second arrow in Fig. 27, this condition is realized at around 2.15 hr and the subsequent action in test plan, i.e., $S_3$ in Fig. 24, was implemented at this time. However, due to coexistence of F4 (f_VinSO) and F5 (f_VliqSC), the fast level-rising rate was unaffected by $S_3$ and eventually the liquid filled the entire drum. Thus, in this scenario, observing $AC_3$ and then $AC_5$ indicated that both F4 and F5 are present.

## 6. Additional case studies

Let us next consider a modified version of the three-component distillation startup process in the demonstrative example (ColumnStratup) published by ASPEN PLUS DYNAMICS (Al-Malah, 2017). Specifically, this startup operation is characterized here with the PFD in Fig. 28 and the SFC in Fig. 29. Since these figures are self-explanatory, further descriptions are omitted for the sake of brevity.

It is assumed that the available raw material is a mixture of 6 wt% $CH_2Cl_2$, 54 wt% $CHCl_3$ and 40 wt% $CCl_4$ and, at steady state, the feed flowrate, temperature and pressure are kept at 10000 kg/hr, 20° and 6.0 bar, respectively. A total of 20 equilibrium stages are chosen and the feed plate of the column is located at 10. In addition, this column is equipped with a partial condenser (stage 1) and its temperature and pressure are set at 80.3 °C and 2 bar, respectively. The pressure chosen for stage 2 is 2.02 bar and the overall pressure drop from stage 20 to stage 2 is 0.235 bar. The concentration of light key $CHCl_3$ in the overhead stream is required to be higher than 81 mol%, while that of heavy key $CCl_4$ in the bottom product not lower than 81 mol%. Finally, the initial conditions are chosen as follows:

- All valves are all closed, while the condenser and reboiler are both off;
- All controllers are on manual;

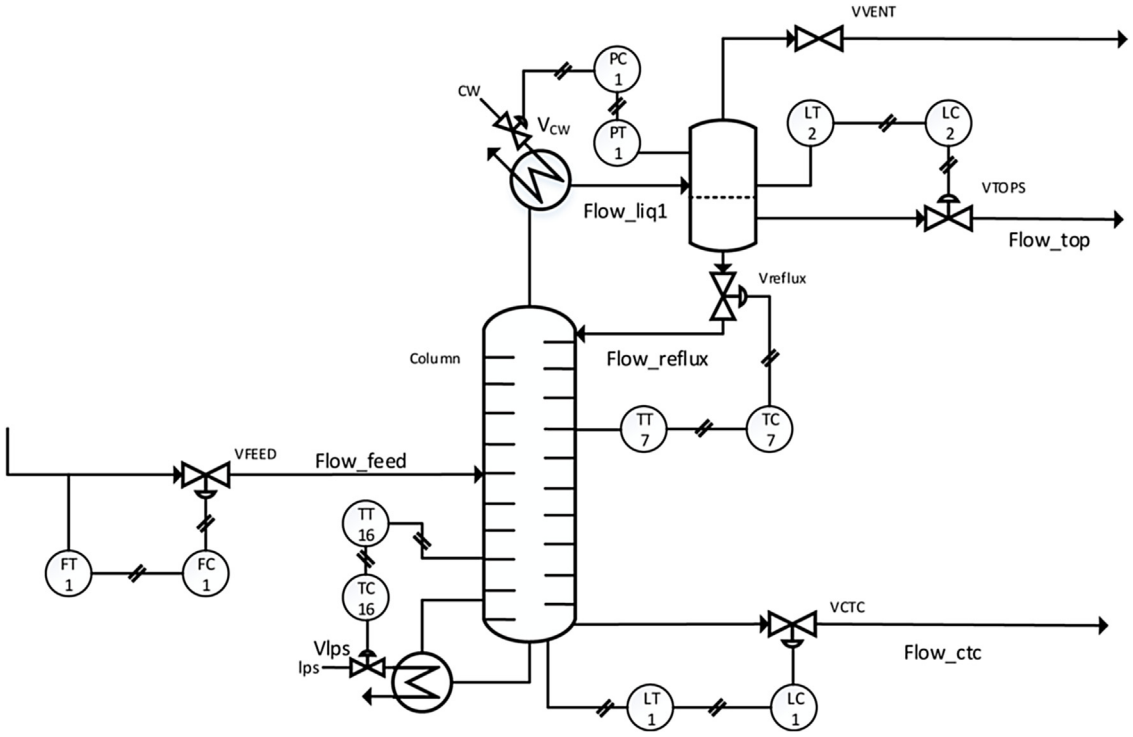



**Fig. 28 – Process flow diagram of a three-component distillation process.**

- Distillation column is empty and at room temperature.

For illustration simplicity, let us assume that there can be only five prominent failure events in this example, i.e.,

- F1 (f_reboiler_failed): A partial reboiler malfunction causing a drop in its heat transfer rate to one half of the normal level;
- F2 (f_T7controller_failed): A hardware failure in temperature controller TC7 that cuts off the reflux flow;
- F3 (f_VtopsSC): Valve Vtops sticks at close position;
- F4 (f_cond_failed): A condenser failure that disables heat removal function;
- F5 (f_VctcSC): Valve Vctc sticks at close position.

### 6.1. *System hierarchy*

All components in the present example can be classified into five hierarchical levels as follows:

- Level-1 component is a PLC or operator;
- Level-2 components include six controller/actuator pairs, i.e., FC1/Vfeed, LC1/Vctc, LC2/Vtops, PC1/Condenser, TC16/Reboiler, and TC7/T7_controller;
- Level-3 components include all material and energy flows surrounding every level-4 unit;
- Level-4 components are essentially four identifiable units in the distillation system, i.e., reflux drum, bottom sump, rectifying section and stripping section;
- Level-5 components are six online sensors for measuring the feed flowrate (FT1), the levels in bottom sump and reflux drum (LT1 and LT2), the overhead pressure (PT1) and the temperatures at plates 7 and 16 (TT7 and TT16).

Notice that, instead of the mass flowrates of cooling and heating media indicated in PFD (see Fig. 28), only the heat-transfer rates (or the temperatures of cooling and heating media) of condenser and reboiler can be adjusted by PC1 and TC16 in ASPEN environment. Therefore, to maintain consistency between automata predictions and simulation results in the validation studies, these energy flows were treated as manipulated variables and assumed to be directly adjustable with fictitious actuators in the corresponding component models. Notice also that, although the actuator of TC7 is a control valve on the reflux flow in the actual system (see Fig. 28), its failure cannot be simulated with ASPEN and, consequently, the identical effects were produced by introducing F2.

### 6.2. *Component models*

All component models in the present example can be constructed with the same approach detailed in Section 2. Since an extremely large volume of engineering-knowledge based automata were generated, they are omitted in this paper for the sake of conciseness without causing confusion. Notice that a detailed description of these structures is available elsewhere (Feng, 2017). On the other hand, the simulation-data based automata are delineated explicitly below to facilitate clear understanding of the hybrid modeling strategy.

#### 6.2.1. *Lumped automaton derived from simulation-data*

To avoid creating a set of unnecessarily complex component models, the ASPEN simulated data during normal operations were used to construct a lumped model for charactering all processing units in the fourth level, i.e., the reflux drum, the bottom sump, the rectifying and stripping sections. The discretized values of their state variables are defined in Tables 4. The mass flowrates (in kg/hr) of all inputs and outputs of each processing unit are discretized and represented with 4 qualitative values, i.e., value 0 for $[0, 0^+)$, value 1 for $[10000, 20000)$, value 2 for $[20000, 30000)$ and value 3 for $[30000, \infty)$, while the entire ranges of energy flowrates (in GJ/hr) facilitated by

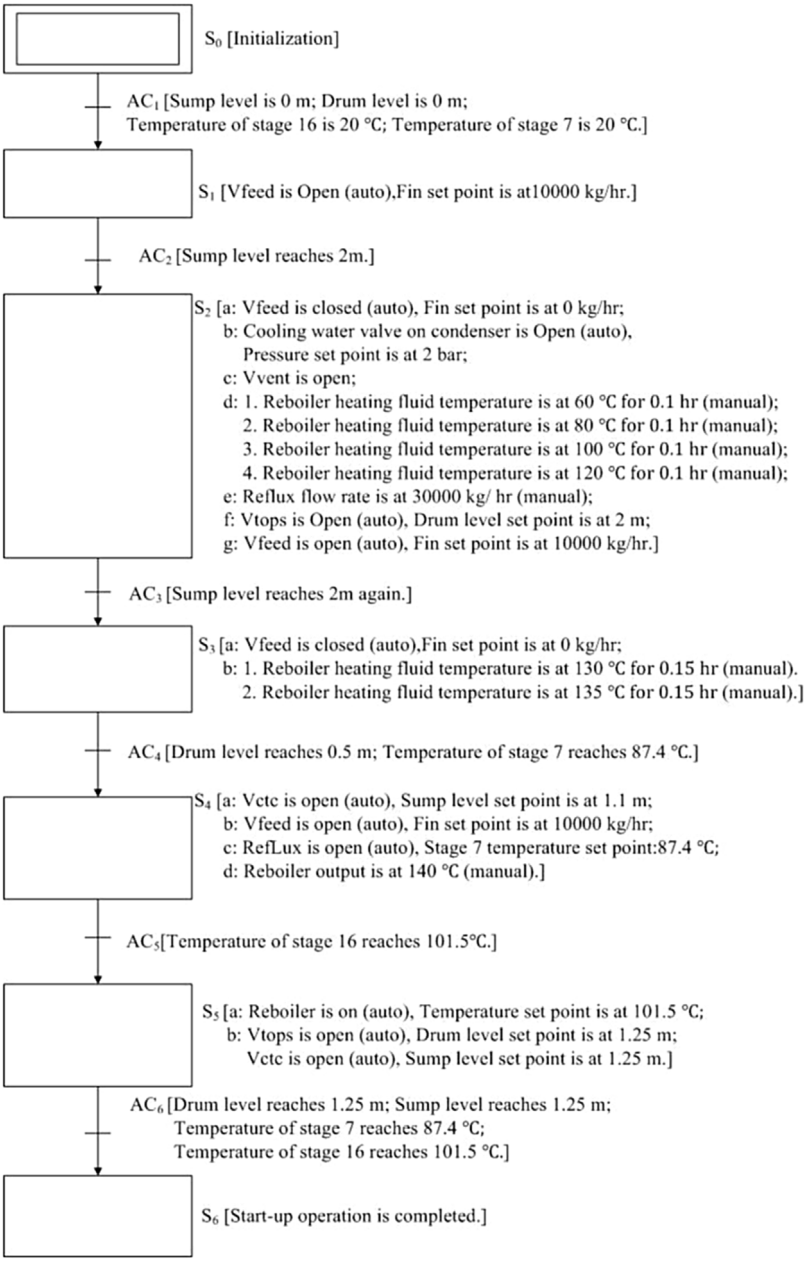



**Fig. 29 – Sequential function chart for startup operation of distillation process.**

**Table 4 – Discretized state variables in distillation process.**

| Interval | PU_Tdrum(°) | PU_Tsump(°) | PU_T7 (°) | PU_T16(°) | PU_Ldrum(m) | PU_Lsump(m) |
|---|---|---|---|---|---|---|
| 0 | [20, 20+) | [20, 20+) | [20, 20+) | [20, 20+) | [0, 0+) | [0, 0+) |
| 1 | [20+, 60) | [20+, 60) | [20+, 60) | [20+, 60) | [0+, 0.5) | [0+, 0.5) |
| 2 | [60, 80) | [60, 80) | [60, 80) | [60, 80) | [0.5, 1.25) | [0.5, 1.25) |
| 3 | [80, 100) | [80, 100) | [80, 100) | [80, 100) | [1.25, ∞) | [1.25, ∞) |
| 4 | [100, 120) | [100, 120) | [100, 120) | [100, 120) | Undefined | Undefined |
| 5 | [120, 140) | [120, 140) | [120, 140) | [120, 140) | Undefined | Undefined |

reboiler and condenser are both divided into 6 intervals, i.e., [0, 0+), [0+, 2.5), [2.5, 5.0), [5.0, 7.5), [7.5, 10.0) and [10.0, ∞), and these intervals are labelled sequentially from 0 to 5. Based on the above discretization schemes, the simulation data during the normal startup can be converted accordingly (see Table 5). An abridged version of this data set can then be produced by removing every row in which the state variables are identical to those in the previous row. All state changes during normal startup operation can be easily extracted from this abridged set and they are incorporated in the automaton model as a sequence of state-transition events. A total of twelve (12) consecutive events were found in the present example and their guards and variable updates are listed in Table 6.

### 6.2.2. *Aggregated hybrid model*

As described before in the flash startup example, the data-based model and its knowledge-based counterpart should both be included in a hybrid automaton to characterize the normal and abnormal modes of every processing unit, respec-

**Table 5 – Discretized simulation data in normal startup of distillation process.**

| Time | PU_T16 | PU_T7 | PU_Lsump | PU_Ldrum | PU_Fin | PU_Ftops | PU_Fctc | PU_Fref | PU_Fliq1 | PU_Fv20 | PU_cond | PU_reb |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0.156 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0.318 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0.512 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0.807 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 0.852 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 1 | 1 |
| 0.897 | 1 | 1 | 2 | 1 | 0 | 0 | 0 | 3 | 1 | 2 | 2 | 2 |
| 0.915 | 1 | 1 | 2 | 1 | 0 | 0 | 0 | 3 | 1 | 2 | 2 | 2 |
| 0.958 | 2 | 2 | 1 | 1 | 1 | 0 | 0 | 3 | 2 | 3 | 3 | 3 |
| 1.125 | 2 | 2 | 1 | 1 | 1 | 0 | 0 | 3 | 2 | 3 | 3 | 3 |
| 1.321 | 3 | 2 | 2 | 1 | 0 | 0 | 0 | 3 | 2 | 3 | 4 | 4 |
| 1.498 | 3 | 2 | 1 | 2 | 1 | 1 | 0 | 3 | 3 | 3 | 4 | 4 |
| 1.659 | 3 | 2 | 1 | 2 | 1 | 1 | 0 | 3 | 3 | 3 | 4 | 4 |
| 1.745 | 4 | 3 | 1 | 2 | 1 | 1 | 1 | 3 | 3 | 3 | 5 | 5 |
| 2.12 | 4 | 3 | 1 | 2 | 1 | 2 | 0 | 3 | 3 | 3 | 5 | 5 |
| 2.325 | 4 | 3 | 1 | 1 | 1 | 1 | 1 | 3 | 3 | 3 | 5 | 5 |
| 3.585 | 4 | 3 | 1 | 1 | 1 | 1 | 1 | 3 | 3 | 3 | 5 | 5 |

**Table 6 – State-transition events during normal startup of distillation process.**

| | Guard | | | | | | | | | | | | Variable | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | PU_T16 | PU_T7 | PU_Lsump | PU_Ldrum | PU_Fin | PU_Ftops | PU_Fctc | PU_Fref | PU_Fliq1 | PU_Fv20 | PU_cond | PU_reb | d_Lsump | d_Ldrum | d_temp16 | d_temp7 |
| db_1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| db_2 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| db_3 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| db_4 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 |
| db_5 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 1 | 1 | 0 | 1 | 0 | 1 |
| db_6 | 1 | 1 | 2 | 1 | 0 | 0 | 0 | 3 | 1 | 2 | 2 | 2 | −1 | 0 | 1 | 1 |
| db_7 | 2 | 2 | 1 | 1 | 1 | 0 | 0 | 3 | 2 | 3 | 3 | 3 | 1 | 0 | 1 | 0 |
| db_8 | 3 | 2 | 2 | 1 | 0 | 0 | 0 | 3 | 2 | 3 | 4 | 4 | −1 | 1 | 0 | 0 |
| db_9 | 3 | 2 | 1 | 2 | 1 | 1 | 0 | 3 | 3 | 3 | 4 | 4 | 0 | 0 | 0 | 0 |
| db_10 | 4 | 3 | 1 | 2 | 1 | 1 | 1 | 3 | 3 | 3 | 5 | 5 | 0 | 0 | 1 | 1 |
| db_11 | 4 | 3 | 1 | 2 | 1 | 2 | 0 | 3 | 3 | 3 | 5 | 5 | 0 | −1 | 0 | 0 |
| db_12 | 4 | 3 | 1 | 1 | 1 | 1 | 1 | 3 | 3 | 3 | 5 | 5 | 0 | 0 | 0 | 0 |

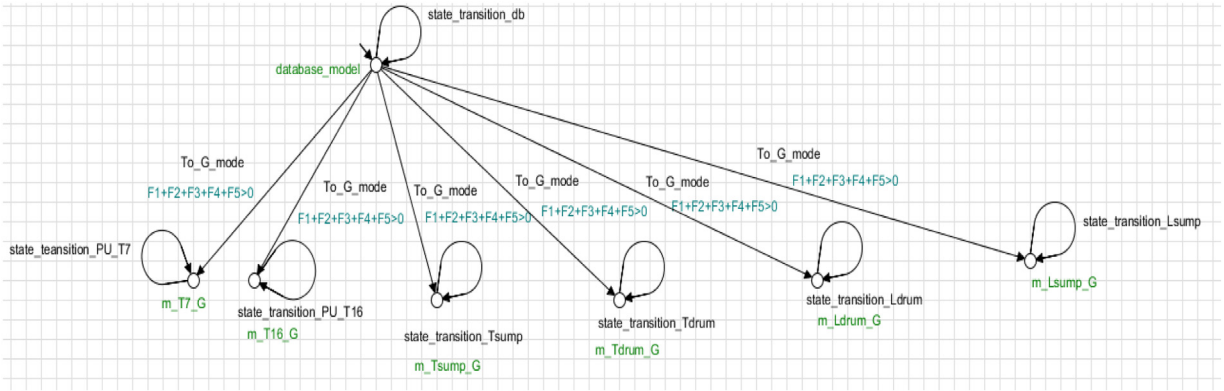



**Fig. 30 – Switching mechanism for predicting system states of distillation process.**

tively. In the present example, the switch actions from the lumped model of the normal operation (which is derived from the simulation data) to the individual component models (which are built with the engineering knowledge) are triggered according to the aggregated automaton shown in Fig. 30. The system state, database_model, in the aggregated automaton refers to the lumped model of normal system behavior, while all other states denote the knowledge-based component models for predicting the same set of state variables during the abnormal scenarios. The system starts normally at database_model. If one or more failure occurs, i.e., $F1 + F2 + F3 + F4 + F5 > 0$, the switch action To_G_mode can be activated to utilize the knowledge-based models for predicting all state variables listed in Table 4.

### 6.3. *Diagnoser*

All aforementioned models were synchronized in SUPREMICA to produce the diagnoser in Fig. 31. Since the same conventions are used to in this figure to label the OETs, redundant

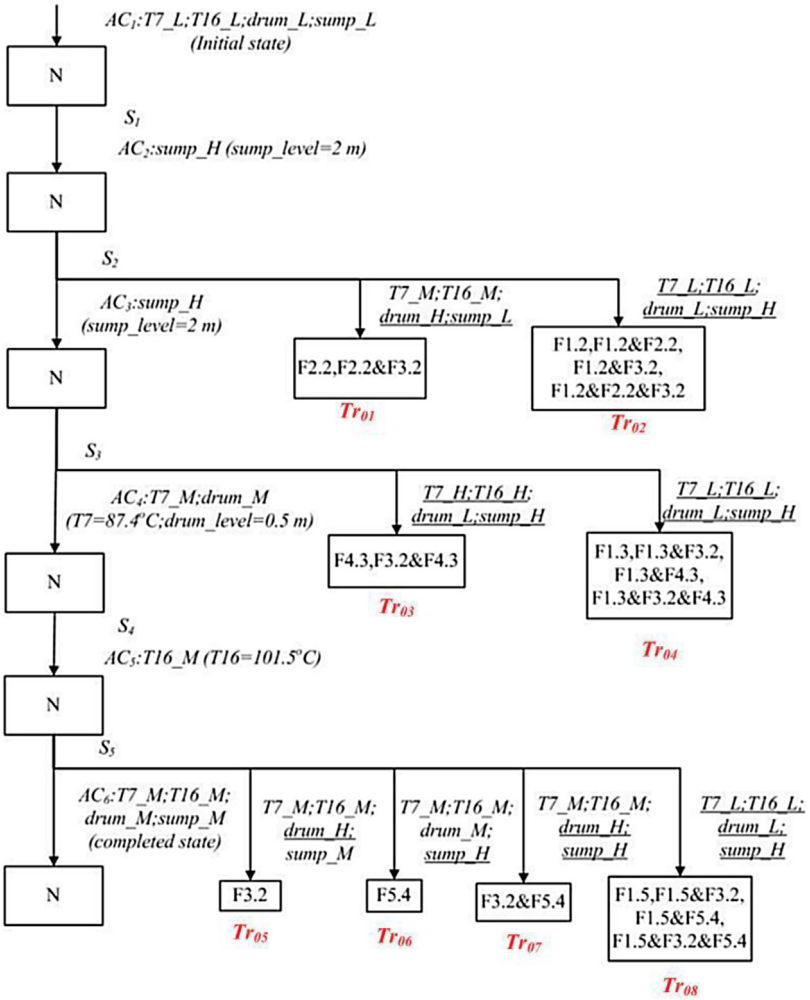



**Fig. 31 – Diagnoser in distillation startup example.**

explanations are not given here for the sake of brevity. Notice that the five failures considered in the present example, i.e., Fi and $i = 1, \cdots, 5$, have already been defined in the beginning of Section 6. Every failure event in diagnoser is again expressed according to the format Fi.j and the attached additional index j ($= 1, \cdots, 6$) denotes the time interval between the two instances when conditions $AC_{j-1}$ and $AC_j$ are satisfied. Finally, the qualitative sensor readings in the OETs can be more specifically characterized as follows:

- T7_L, T7_M, T7_H $\Rightarrow$ T7 $\lessgtr$ 87.4°C;
- T16_L, T16_M, T7_H $\Rightarrow$ T16 $\lessgtr$ 101.5°C;
- drum_L, sump_L $\Rightarrow$ drum_level, sump_level $< 0.5$ m;
- drum_M, sump_M $\Rightarrow 0.5\,\text{m} \leq$ drum_level, sump_level $\leq 1.25$ m;
- drum_H, sump_H $\Rightarrow$ drum_level, sump_level $> 1.25$ m.

### 6.4. *Diagnostic test plans*

The synthesis procedure given in Section 4 can be applied to find the diagnostic test plans of all undiagnosable OETs in Fig. 31. For illustration simplicity, only the diagnostic procedures of $Tr_{01}$ (Fig. 32) and $Tr_{08}$ (Fig. 33) are described in detail in the sequel:

The first test plan calls for an operator action that fully opens Vtops after observing $Tr_{01}$. There may be two possible outcomes. If the sensor readings show T7_M, T16_M, drum_L and sump_L, then it can be concluded that Vtops is normal and the fault origin is F2 (f_T7controller_failed). However, if the online measurements reveal that T7_M, T16_M, drum_H and sump_L, then it can be certain that, in addition to f_T7controller_failed, valve Vtops also sticks at the close position (f_VtopsSC). In other words, both F2 and F3 are present in this scenario.

The second test plan calls for two simultaneous actions that fully closes Vin and opens Vctc after observing $Tr_{08}$. This test divides the fault origins into two separate groups, i.e., (1) {F1, F1&F3} and (2) {F1&F5, F1&F3&F5}, and they can be identified respectively according to the online measurements specified in $AC_2$ (T7_L, T16_L, drum_L and sump_L) and $AC_3$ (T7_L, T16_L, drum_L and sump_H). Notice that sump_L in $AC_2$ implies that Vctc is functional and thus F5 must be excluded from the first group of fault origins. In response to condition $AC_2$, steps in $S_2$, i.e., opening Vin and Vtops fully and increasing flowrate of cooling water, should be applied and two possible outcomes may be produced:

- $AC_4$ (T7_L, T16_L, drum_L and sump_L) implies that the fault origin is a partial reboiler failure, i.e., F1 (f_reboiler_failed);
- $AC_5$ (T7_L, T16_L, drum_H and sump_L) indicates that, other than F1, Vtops is also stuck at the close position, i.e., F3 (f_VtopsSC).

On the other hand, it is necessary to implement the test steps in $S_3$, i.e., opening Vtops fully and increasing flowrate

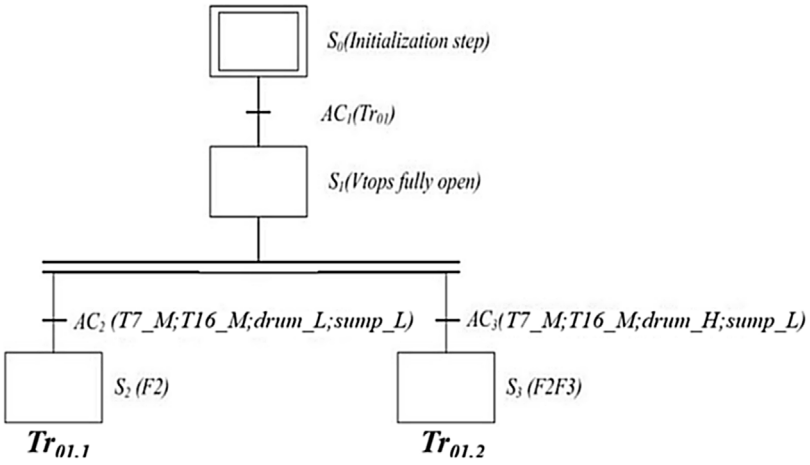



**Fig. 32 – Diagnostic test plan of $Tr_{01}$ in distillation startup example.**

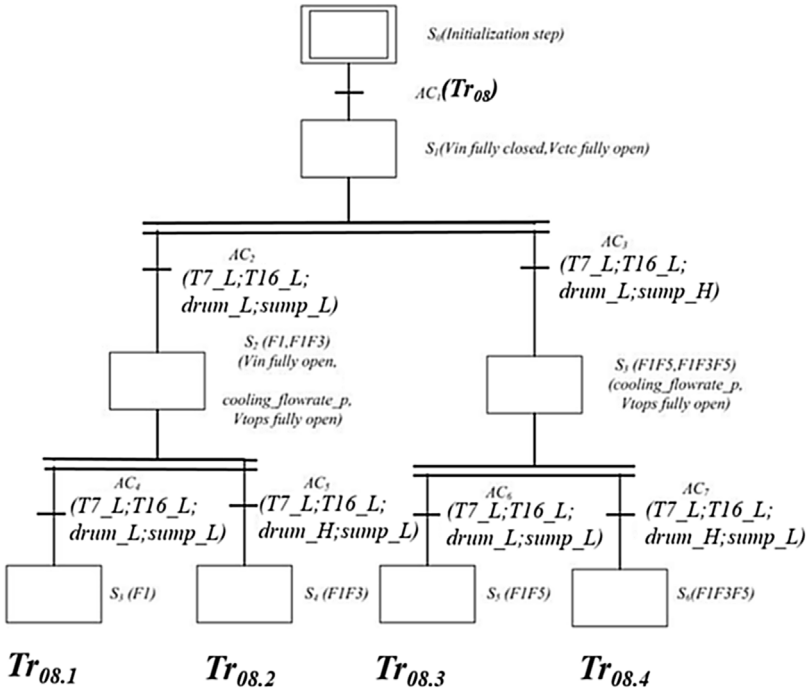



**Fig. 33 – Diagnostic test plan of $Tr_{08}$ in distillation startup example.**

of cooling water, when condition $AC_3$ is confirmed. The corresponding diagnostic results can be summarized below:

- $AC_6$ (T7_L, T16_L, drum_L and sump_L) implies that the fault origin consists of two failures, i.e., F1 and F5 (f_VctcSC);
- $AC_7$ (T7_L, T16_L, drum_H and sump_L) indicates that, other than F1 and F5, Vtops is also stuck at the close position, i.e., F3 (f_VtopsSC).

## 7. Conclusions

An efficient modeling approach has been developed in this work to automatically generate diagnostic test plans for differentiating the originally inseparable fault origins in realistic chemical systems. Specifically, the dynamic behavior of every component in a given process is modeled by integrating both the generic engineering knowledge and also rigorous simulation data into a hybrid automaton. This automata-building strategy effectively reduces the unnecessarily large search space caused by path explosion in the traditional DES model. The diagnostic test plans can then be synthesized according to the system model obtained by synchronizing the hybrid automata with the standard DES operation of parallel composition. The feasibility of the proposed modeling approach is demonstrated with two examples concerning the startup operations of a flash drum and also a distillation column. The validity of the resulting test plans have also been rigorously confirmed in extensive dynamic simulation studies with ASPEN PLUS DYNAMICS.

## Appendix A. Supplementary data

Supplementary material related to this article can be found, in the online version, at doi:https://doi.org/10.1016/j.cherd.2019.02.033.

## References

Åkesson, K., Fabian, M., Malik, R., 2006. SUPREMICA — an integrated environment for verification, synthesis and

simulation of discrete event systems. In: IEEE Proceedings of the 8th International Workshop on Discrete Event Systems., pp. 384–385.
Al-Malah, K., 2017. Aspen Plus® — Chemical Engineering Applications. John Wiley & Sons, Inc., Hoboken, New Jersey.
Benveniste, A., Fabre, E., Haar, S., Jard, C., 2003. Diagnosis of asynchronous discrete-event systems: a new unfolding approach. IEEE Trans. Autom. Control 48 (5), 714–727.
Cassandras, C.G., Lafortune, S., 2008. Introduction to Discrete Event Systems, 2nd edition. Springer Science + Business Media, LLC, New York, NY, USA.
Chen, J., Jiang, Y.C., 2011. Development of hidden semi-Markov models for diagnosis of multiphase batch operation. Chem. Eng. Sci. 66 (15), 1087–1099.
Dai, Y., Zhao, J., 2011. Fault diagnosis of batch chemical processes using a dynamic time warping (DTW)-based artificial immune system. Ind. Eng. Chem. Res. 50, 4534–4544.
Debouk, R., Lafortune, S., Teneketzis, D., 2000. Coordinated decentralized protocols for failure diagnosis of discrete event systems. Discrete Event Dyn. Syst. Theory Appl. 10 (1–2), 33–86.
Feng, S.T., 2017. A Hybrid Modeling Strategy to Build Automata for Synthesizing Diagnostic Tests in Sequential Operations. MS Thesis. National Cheng Kung University, Tainan, Taiwan.
Gascard, E., Simeu-Abazi, Z., 2013. Modular modeling for the diagnostic of complex discrete-event systems. IEEE Trans. Autom. Sci. Eng. 10 (4), 1101–1123.
Gomes Cabral, F., Moreira, M.V., Diene, O., Basilio, J.C., 2015. A Petri net diagnoser for discrete event systems modeled by finite state automata. IEEE Trans. Autom. Control 60 (1), 59–71.
Hsieh, W.C., Chang, C.T., 2016. Timed-automata based method for synthesizing diagnostic tests in batch processes. Comput. Chem. Eng. 84, 12–27.
Kang, A., Chang, C.T., 2014. Automata generated test plans for fault diagnosis in sequential material- and energy-transfer operations. Chem. Eng. Sci. 113, 101–115.
Malik, R., Fabian, M., Akesson, K., 2011. Modelling large-scale discrete-event systems using modules, aliases, and extended finite-state automata. Proceedings of 18th IFAC World Congress 18, 7000–7005.
Nomikos, P., MacGregor, J.F., 1994. Monitoring batch processes using multiway principal component analysis. AIChE J. 40 (8), 1361–1375.
Nomikos, P., MacGregor, J.F., 1995. Multivariate SPC charts for monitoring batch processes. Technometrics 37 (1), 41–59.
Qiu, W.B., Kumar, R., 2006. Decentralized failure diagnosis of discrete event system. IEEE Trans. Syst. Man Cybern. A Syst. Hum. 36 (3), 384–395.
Wang, C.J., Chen, Y.C., Feng, S.T., Chang, C.T., 2017. Automata-based operating procedure for abnormal situation management in batch processes. Comput. Chem. Eng. 97, 220–241.
Yeh, M.L., Chang, C.T., 2011. An automaton-based approach to evaluate and improve online diagnostic schemes for multi-failure scenarios in batch processes. Chem. Eng. Res. Des. 89, 2652–2666.
Zad, S.H., Kwong, R.H., Wonham, W.M., 2003. Fault diagnosis in discrete-event systems: framework and model reduction. IEEE Trans. Autom. Control 48 (7), 1199–1204.