Computers and Chemical Engineering 38 (2012) 151–170

Contents lists available at SciVerse ScienceDirect

Computers and Chemical Engineering

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

An automata based method for online synthesis of emergency response procedures in batch processes

Ming-Li Yeh, Chuei-Tin Chang*

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

ARTICLE INFO

Article history:
Received 23 March 2011
Received in revised form
14 November 2011
Accepted 17 November 2011
Available online 26 November 2011

Keywords:
Automaton
Batch operation
Emergency response
Process safety
Supervisory control

ABSTRACT

Rapid response to remove (or reduce) the detrimental effects of accidents has always been an important safety issue for the chemical industries. A systematic strategy is presented in this paper to synthesize emergency response procedures in any given batch system. Specifically, two distinct sets of automata are first constructed offline to model the plant behaviors and the control specifications, respectively. On the basis of these automata, an admissible supervisor can be synthesized online for a diagnosed failure-induced system state by applying the parallel composition operation. For the purpose of identifying an efficient operating procedure to steer the system away from hazardous conditions while still maintaining an acceptable production rate, an additional set of auxiliary automata can be augmented with this supervisor to set the operation targets and to limit the total number of actuator actions. Two examples are presented in this paper to demonstrate the feasibility of the proposed approach.

1. Introduction

Hardware failures in the chemical plants, e.g., controller malfunction, valve sticking and vessel leakage, etc., should be considered as unavoidable but random events. Any such event may cause severe deterioration in product quality, drastic decrease in productivity and, in the worst case, catastrophic outcome such as fire, explosion, or toxic release. To avoid (or abate) losses, the synthesis of appropriate fail-safe procedures during emergency situations has always been regarded as an important issue in plant operation (Blanke, Kinnaert, Lunze, & Staroswiecki, 2003; Hashimme, Yajima, Kuwashita, & Onogi, 2008; Patton, 1997; Tan & Yamashita, 2010; Yamashita, 2007; Zhang & Jiang, 2003). However, because of the extreme complexity of modern chemical processes, it is in general very difficult to follow an intuitive ad hoc approach for such a time-constrained task.

The pioneering works on the automatic synthesis of operating procedures were first performed by Rivas and Rudd (1974). Subsequent studies towards the design and verification of procedural controllers under normal plant conditions have also been carried out extensively (Chen & Chen, 1994; Hamid, Sin, & Gani, 2010; Kaspar & Ray, 1992; Kim & Moon, 2009; Naka, Lu, & Takiyama, 1997; Panjapornpon, Soroush, & Seider, 2006; Sanchez & Macchietto, 1995). It can be observed that this issue has been tackled with numerous different modeling/reasoning mechanisms, e.g., the AI-based linear and nonlinear planning strategies (Fusillo & Powers, 1987; Lakshmanan & Stephanopoulos, 1988; Viswanathan, Johnsson, Srinivasan, Venkatasubramanian, & Arzen, 1998), the mathematical programming models (Crooks & Macchietto, 1992; Galán & Barton, 1997; Li, Lu, & Naka, 1997), the symbolic model verifiers (Kim, Kim, & Moon, 2009), and various different qualitative models such as the state graphs (Hoshi, Nagasawa, Yamashita, & Suzuki, 2002; Ivanov, Kafarov, Perov, & Reznichenko, 1980; Kinoshita, Umeda, & O'Shima, 1982) and Petri nets (Chou & Chang, 2005; Hashizume, Yajima, Ito, & Onogi, 2004; Lai, Chang, & Hwang, 2007; Wang, Chou, & Chang, 2005; Yamalidou & Kantor, 1991). Generally speaking, although these different approaches were effective for synthesizing the *normal* operating procedures, very few of them can be applied to generate proper emergency response strategies. This is mainly due to the difficulties in (1) characterizing the failure-induced scenarios, and (2) synthesizing and validating the corresponding response procedures. As a result, there is a definite need to develop a systematic method to automatically conjecture a collection of reliable operation actions for any given abnormal system condition.

The aforementioned procedure synthesis problem has been analyzed and solved in the present study on the basis of supervisory control theory (Ramadge & Wonham, 1987, 1989). In its original framework, every discrete-event system is characterized with a set of event sequences (or the so-called "language") which can be predicted according to an automaton model. An admissible "supervisor" can usually be synthesized with two distinct automata, i.e.,

* Corresponding author. Tel.: +886 6 2757575x62663; fax: +886 6 2344496.
E-mail address: ctchang@mail.ncku.edu.tw (C.-T. Chang).

0098-1354/$ – see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.compchemeng.2011.11.008

the plant model and the specification model. The former is used to represent how a system behaves with *or* without hardware failures, while the latter for defining the "legal" events or actions allowed in plant operations. Although this modeling approach has already been successfully applied in many previous studies (Brandin & Wonham, 1994; Dietrich, Malik, Wonham, & Brandin, 2002; Falkman, Lennartson, & Tittus, 2009; Koutsoukos, Antsaklis, Stiver, & Lemmon, 2000; Malik & Malik, 2006; Ouedraogo, Khoumsi, & Nourelfath, 2010; Wonham, 2000; Yeh & Chang, in press-a), none of them offered a specific step-by-step automata-building procedure for generating the emergency response procedures in batch chemical processes.

A systematic implementation procedure is proposed in the present work to address the modeling issue mentioned above. In the first step, all components specified in the piping and instrumentation diagram (P&ID), i.e., the programmable logic controller, the actuators, the major processing units, and the online sensors, are modeled respectively with standard automata. A similar approach is adopted next to stipulate the control specifications which are common for all possible failure-induced scenarios and to construct the corresponding automaton models. An admissible supervisor can then be assembled automatically according to these automaton models and also a given abnormal system state. Finally, for the purpose of identifying the most efficient emergency response procedure(s), a set of auxiliary automata can be augmented to the admissible supervisor so as to set the operation target(s) and to impose upper bound of the total number of actuator actions by producing the *supremal controllable sublanguage* (Cassandras & Lafortune, 1999).

In summary, the novel contributions of this work can be outlined as follows:

- Automata have been adopted in the published studies to build the supervisors for normal operations only, while a *new application* is considered in the present study, i.e., such models are used for generating emergency response procedures in failure-induced scenarios.
- It can be observed from the literature that the automaton models were conjectured in an ad hoc fashion in the past. A systematic *new model-building method* is presented in this paper to construct the plant model by assembling the component automata in a standard hierarchical framework.
- A *new synthesis procedure* is developed in this study to produce the most efficient operating procedure(s) *online* for any given failure-induced system state in a batch chemical process.
- The feasibility and effectiveness of the proposed approach are confirmed with rigorous case studies concerning a realistic beer filtration plant.

The remainder of this paper is organized as follows. To facilitate explanation of the proposed automata-building methodology, the general model structure is first illustrated in the next section. A systematic implementation strategy is then outlined in Section 3 for the purpose of synthesizing the most efficient operating procedure(s) during emergency situations. A simple liquid-transfer system is adopted as an example in this section for illustration convenience. In order to further demonstrate the feasibility and correctness of the proposed approach, additional case studies of a realistic beer filtration plant are reported in detail in Section 4. Finally, conclusions are provided at the end of this paper.

2. General model framework

To facilitating clear description of the proposed method, a brief review of the automaton structure is first given here.

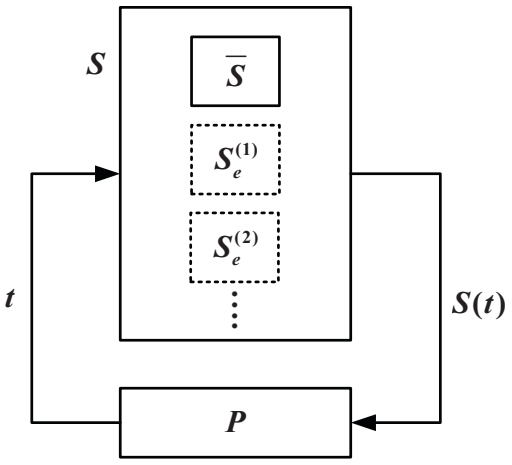


Fig. 1. The feedback loop of supervisory control.

Specifically, a deterministic automaton A can be regarded as a six-tuple (Cassandras & Lafortune, 1999):

$$A = (X, E, f, \Sigma, x_0, X_m) \tag{1}$$

where, X is the set of system states; E is the event set; $f : X \times E \to X$ represents the state transition function; $\Sigma : X \to 2^E$ denotes the active event function and 2^E is the power set of E (i.e., the set of all possible subsets of E); $x_0 \in X$ is the initial system state; $X_m \subseteq X$ is the set of marked states. The transition function $f(x, e) = x'$ means that a transition from state $x \in X$ to another state $x' \in X$ is caused by the feasible event $e \in E$, while the active event function $\Sigma(x)$ can be regarded as the set of active events at state x. Notice that every automaton can be viewed as a *language-generating machine*. The events in set E should be regarded as the *alphabets* of this language and an event sequence allowed in automaton is regarded as a trace, string or word (*trace* is used in this work). The event set E can be further partitioned into subsets of controllable and uncontrollable events, i.e., $E = E_c \dot{\cup} E_{uc}$. The events in E_c are those that can be *forbidden* with a supervisor or controller, whereas the events in E_{uc} are bound to occur in due course.

In the supervisory control paradigm (see Fig. 1), the plant to be operated is represented with an automaton P and the supervisor S is viewed as a *mapping* or *function* from the language generated by P to the power set of E, i.e.,

$$S : \mathscr{L}(P) \to 2^E \tag{2}$$

where, $\mathscr{L}(P)$ represents the set of all traces obtained from automaton P. If $t \in \mathscr{L}(P)$, then $S(t)$ is interpreted as the set of actuator actions allowed after executing trace t. In traditional applications, automaton P is a model of the normal plant behaviors and its supervisor S is used to represent the corresponding operating procedure. In the present study, since additional mechanisms are incorporated into automaton P to model the fault propagation behaviors, separate supervisors must be applied accordingly. Firstly, the SFC under normal process conditions and the corresponding normal supervisor $\bar{S}$ are assumed to be *given* a priori in this study. On the other hand, each emergency supervisor $S_e^{(i)}$ $(i = 1, 2, \ldots)$ should be regarded as an *unavailable* function, which must be synthesized according to a specific failure-induced system state.

For the purpose of synthesizing $S_e^{(i)}$ on demand, a set of common control specifications must be stipulated in advance. Notice that the plant automaton may generate "illegal" traces because they are physically inadmissible, e.g., an attempt to fill a tank when it is full, or they violate a desired sequence of events, e.g., an attempt to heat a vessel when it is empty. To eliminate these unacceptable traces in $\mathscr{L}(P)$, a set of specifications (which can be modeled respectively

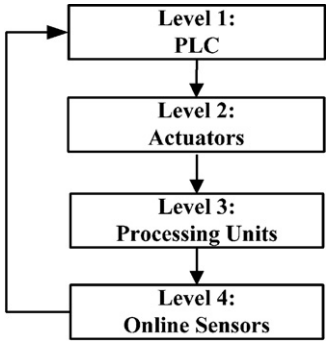


Fig. 2. Hierarchical structure of a batch process (Yeh & Chang, in press-a, in press-b).

with automata $H_{spec,j}$ and $j = 1, 2, \ldots$) should be introduced to restrict the system behavior to be within an admissible subset $\mathscr{L}_a \subset \mathscr{L}(P)$. The emergency supervisor $S_e^{(i)}$ can then be extracted from this subset so as to ensure $\mathscr{L}(S_e^{(i)}/P) \subseteq \mathscr{L}_a$, where $\mathscr{L}(S_e^{(i)}/P)$ represents the set of all traces obtained during scenario i from the closed-loop system in Fig. 1.

In this work, all identifiable hardware items (components) in a batch process are classified into a 4-level hierarchy according to Fig. 2 (Yeh & Chang, in press-a, in press-b), i.e., (1) the programmable logic controller (PLC), (2) the actuators, (3) the processing units and (4) the online sensors. Notice that this system framework is actually very similar to that of a standard feedback control loop for the continuous processes. The connections between adjacent levels can be viewed as "information flows" and they can be characterized more specifically as follows:

- For level 1/level 2 interface, the information flows are controller signals that trigger the actuator actions;
- For level 2/level 3 interface, the information flows are actuator states that govern the operation modes of processing units;
- For level 3/level 4 interface, the information flows are operating conditions that dictate the sensor measurements;
- For level 4/level 1 interface, the information flows are online measurements that drive the controller signals.

All actuator actions required in the supervisor S are further assumed to be *executable* with the available PLC in the first level, while automaton P should be a model of the controller-free system which consists of all components in the last three levels.

3. Procedure synthesis strategy

A suitable emergency response procedure can be identified systematically according to the flowchart presented in Fig. 3. This procedure synthesis strategy can also be described alternatively as:

(i) Build automata to model all components in the given batch plant;
(ii) Construct automata to represent the common control specifications;
(iii) Combine all automata prepared in the above two steps to produce the *admissible* emergency supervisor for the assigned faulty system state;
(iv) Produce an *implementable* emergency supervisor by augmenting the admissible supervisor with auxiliary automata and then identify the most efficient operating procedure accordingly.

To facilitate clear illustration of this synthesis strategy, let us consider the liquid transfer system shown in Fig. 4 (which will be referred to as Example 1 throughout this paper). The system is made of a storage tank, a supply system, two 3-way valves (V-1 and V-3) and two gate valves (V-2 and V-4). Notice that each 3-way valve can be switched to one of two alternative positions, i.e., OPEN or CLOSE, to manipulate the flow directions. The fluid in vertical pipeline P-1

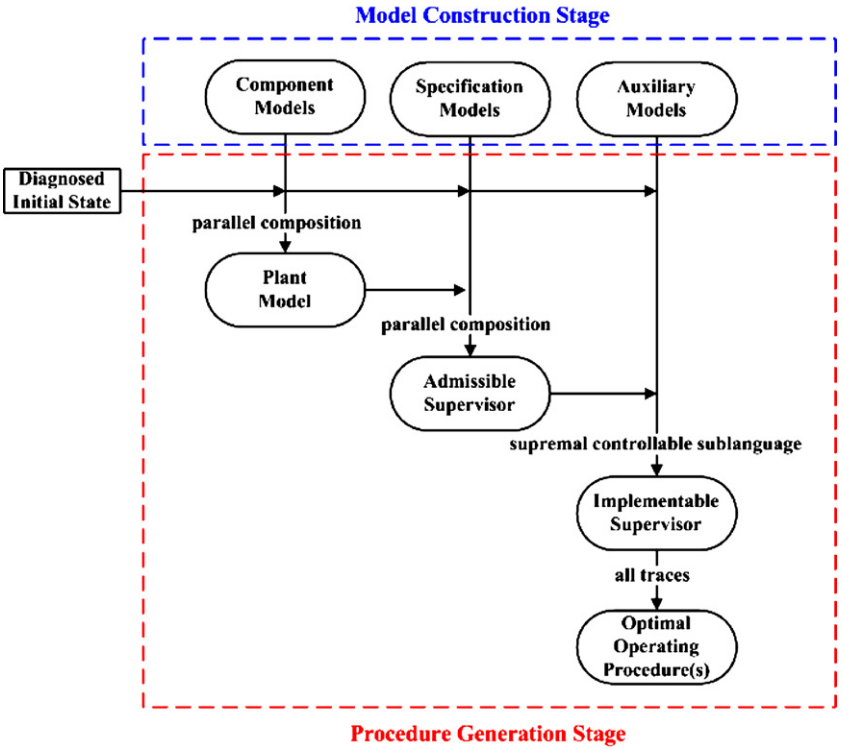


Fig. 3. Systematic strategy for synthesizing emergency response procedure(s).

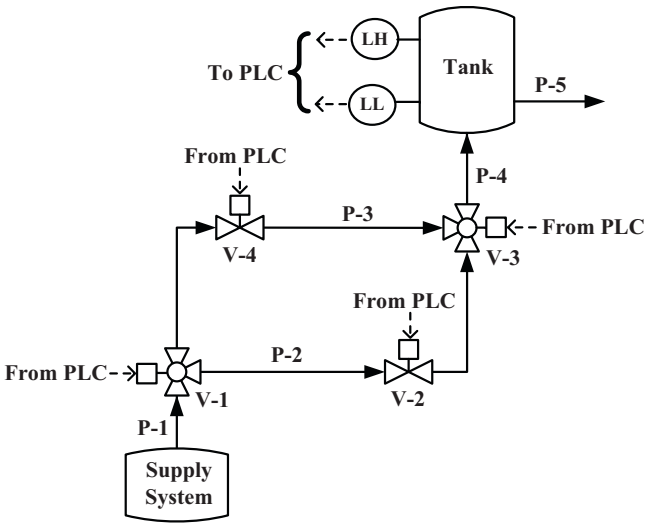


Fig. 4. A liquid-transfer system (Example 1).

is allowed to flow into the horizontal line P-2 via the opened V-1, while the horizontal flow in P-3 can join the vertical flow in P-4 via the opened V-3. On the other hand, if V-1 or V-3 is switched to the CLOSE position, the valve connection to/from horizontal pipeline must be blocked completely. While the inlet flow to the storage tank can be controlled with these valves, this tank is drained continuously via pipeline P-5 as long as it is not empty. The height of liquid level is monitored online with a sensor, and two distinct signals, i.e., (1) SH (level signal high) and (2) SL (level signal low), are transmitted to a PLC to actuate the aforementioned four valves in this system. Under the assumptions that the initial liquid level in storage tank is low and valve V-4 is at the OPEN position initially while the others are all closed, the sequential function chart (SFC) in Fig. 5 can be stipulated to represent the *normal* periodic operating procedure. Notice that OS_i $(i = 0, 1, 2)$ and AC_j $(j = 1, 2, 3)$ denote the operation steps and the activation conditions of these steps, respectively. The control actions taken in each step and the sensor signals used in each condition are also specified in this chart. Finally, let us assume that valve V-3 may stick at either CLOSE ($V3SC$) or OPEN ($V3SO$) position. When failure $V3SC$ occurs, the liquid transfer operation can be performed via an alternative route by opening V-1 and V-2. Finally, it is assumed in this example that additional flow sensors are also available so that each failure can be unambiguously diagnosed online after it occurs.

3.1. Component models

As mentioned before, every hardware item in the given batch process can be viewed as a component. For any component under consideration, a finite set of identifiable states should be obtained first and the active events of each of these states should then be

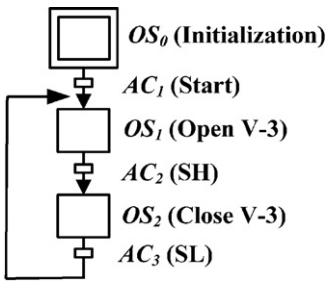


Fig. 5. Sequential function chart of the normal operating procedure in Example 1.

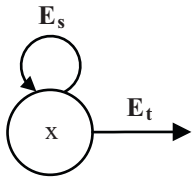


Fig. 6. General model structure for each component state.

conjectured on the basis of process knowledge (see Fig. 6). The active events considered in this work can be further classified into two types, i.e., state-transferring events $e_i^t \in E_t$ and state-sustaining events $e_j^s \in E_s$. Notice that the corresponding transition functions can be written respectively as

$$f(x, e_i^t) = x_i \tag{3}$$

$$f(x, e_j^s) = x \tag{4}$$

where, $x_i \neq x$. It should also be noted that a state-sustaining event can almost always be used as the state-transferring event in a next-level component model.

The automaton representations of all hardware items in Fig. 2 are outlined in the sequel:

- Level 1: The controller model under normal plant conditions is presented in Fig. 7(a), which can be constructed in a straightforward fashion according to Fig. 5. For simplicity, it is assumed that the control actions in OS_1 can always be performed initially and thus the event specified in OS_0 is omitted. The level-4 events *SLcon* and *SHcon* are adopted to represent the situations when the reading of level sensor continues at the high and low values for a long enough period, respectively. The events *openV*3 and *closeV*3 are the control actions to open and close valve V-3, respectively. Notice that failure *V*3*SO* can only occur after V-3 is opened and can only be diagnosed after executing *closeV*3, i.e., at state 3, while failure *V*3*SC* may occur only when V-3 is closed and can only be diagnosed after carrying out *openV*3, i.e., at state 1. The loop formed by states 1–4 represents the cyclic state-transition process during normal operation, while at the failed state 5 all events are allowed. Notice that, since the goal here is to obtain an emergency response procedure, state 5 should be designated as the initial condition for synthesizing the required admissible supervisor.
- Level 2: It should be first noted that the process configuration of a batch system is governed by the collective states of actuators. Since there are four valves in Fig. 4, all possible combinations of the valve states can be enumerated (see Table 1) and each is obviously associated with a particular configuration GV_k $(k = 1, 2, \ldots, 16)$. Notice that, since the assumed valve failures do not result in extra liquid transfer paths, it is not necessary to differentiate the normal and failed valve states in this table. Let us use V-3 as an example to illustrate the automaton representation of a valve (see Fig. 7(b)). The normal valve states, i.e., *V*3*C* and *V*3*O*, denote the CLOSE and OPEN positions, respectively, while events *openV*3 and *closeV*3 trigger the corresponding CLOSE-to-OPEN and OPEN-to-CLOSE processes in normal operations. The symbols $GV_i con$ and $GV_j con$ are used to characterize the events that the corresponding process configurations are maintained for a sufficiently long period of time, and they obviously allow V-3 staying at CLOSE and OPEN positions, respectively. The abnormal valve states of V-3, i.e., "V-3 sticks at the CLOSE position" and "V-3 sticks at the OPEN position", are represented respectively with *V*3*CS* and *V*3*OS*. Notice that, if either state is reached after a failure, this valve state should remain permanently unchanged despite occurrence of any realizable event in the given system.

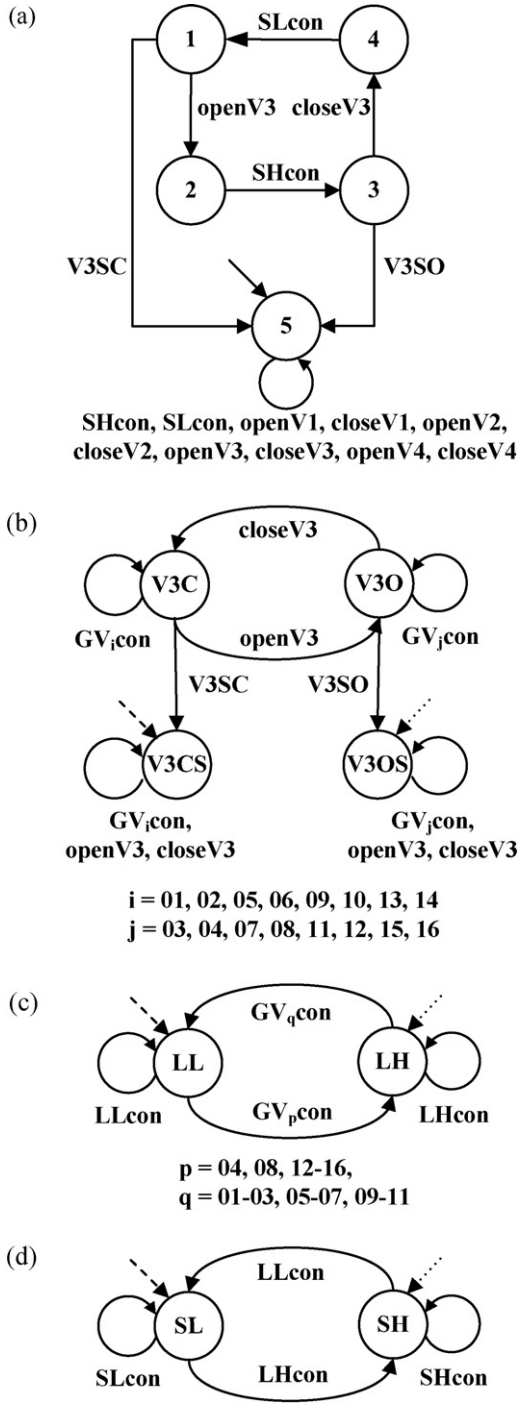


Fig. 7. Component models in Example 1: (a) controller model that contains additional failure modules; (b) V-3 model that contains additional failure modules; (c) tank level model; (d) level sensor model.

Notice also that one of these failed valve states should be designated as the initial condition for synthesizing the corresponding emergency supervisor. Finally, it should be emphasized that all other valves can be modeled in the same fashion.

- Level 3: The liquid level in storage tank is described with the automaton in Fig. 7(c). It can be observed that two tank states are considered here, i.e., *LL* and *LH*. The event GV_pcon is the process configuration that causes the *LL*-to-*LH* transition, while GV_qcon is the configuration that facilitates the opposite *LH*-to-*LL* process. Notice that, the events *LLcon* and *LHcon* represent the liquid level continues at low and high positions, respectively. The tank state *LL* should be assigned as the initial state for supervisor synthesis after *V3SC*, while *LH* should be chosen as the starting state after *V3SO*.
- Level 4: The level sensors can be modeled with the automata presented in Fig. 7(d). Notice that states *SL* and *SH* denote the sensor measurements of liquid levels *LL* and *LH*, respectively. On the other hand, the prior-level events *LHcon* and *LLcon* should cause the *SL*-to-*SH* and *SH*-to-*SL* transitions, respectively. To simplify illustration of the proposed methodology without loss of generality, the sensor failures are not considered in the present example and, thus, the online level measurements should be identical to the tank states.

Table 1
Valve combinations in Example 1.

V-1	V-2	V-3	V-4	Symbol
C	C	C	C	GV_1
C	C	C	O	GV_2
C	C	O	C	GV_3
C	C	O	O	GV_4
C	O	C	C	GV_5
C	O	C	O	GV_6
C	O	O	C	GV_7
C	O	O	O	GV_8
O	C	C	C	GV_9
O	C	C	O	GV_{10}
O	C	O	C	GV_{11}
O	C	O	O	GV_{12}
O	O	C	C	GV_{13}
O	O	C	O	GV_{14}
O	O	O	C	GV_{15}
O	O	O	O	GV_{16}

Notice that letter O means open, while C means close.

3.2. Control specifications

In this present study, the control specifications are used to ensure system safety and/or operability in emergency situations. Specifically, it can be used to achieve or forbid a prescribed event/state sequence to avoid physically inadmissible behaviors, e.g., filling a tank when it is full, heating a vessel when it is empty, and transferring material(s) to an improper destination or to form a hazardous mixture, etc. In particular, three typical specifications are considered in this work and the general structures of their automaton models are presented below:

- **Spec 1**: *stipulate the precedence order of an observable event sequence and, after confirming each of these events, allow execution of a designated set of actuator actions*. The generalized automaton representation of this specification can be found in Fig. 8(a). In this automaton, $e_1e_2\cdots e_n$ ($e_i \in E_i$, $i = 1, 2, \ldots, n$) denotes the desired observable event sequence, and $e'_i \in E'_i$ represents the actuator action allowed to be executed after event e_i. For the liquid transfer system described previously in Example 1, the corresponding control specification can be summarized as
  (a) Every valve in the system should be allowed to be switched to the OPEN position after event *SLcon*;
  (b) Every valve may be closed after *SHcon*.
  The automaton representation is given in Fig. 9(a), in which states 1 and 2 should be considered as the starting conditions after *V3SC* and *V3SO*, respectively.
- **Spec 2:** *stipulate the allowed actuator actions at the normal system state and also at each diagnosed state*. The generalized automaton representation of this specification can be found in Fig. 8(b), where f_j ($j = 1, 2, \ldots, m$) denotes the jth failure and $e''_j \in E''_j$ ($j = 0, 1, 2, \ldots, m$) represent an allowed actuator action at state j. For Example 1, this control specification can be outlined as follows:

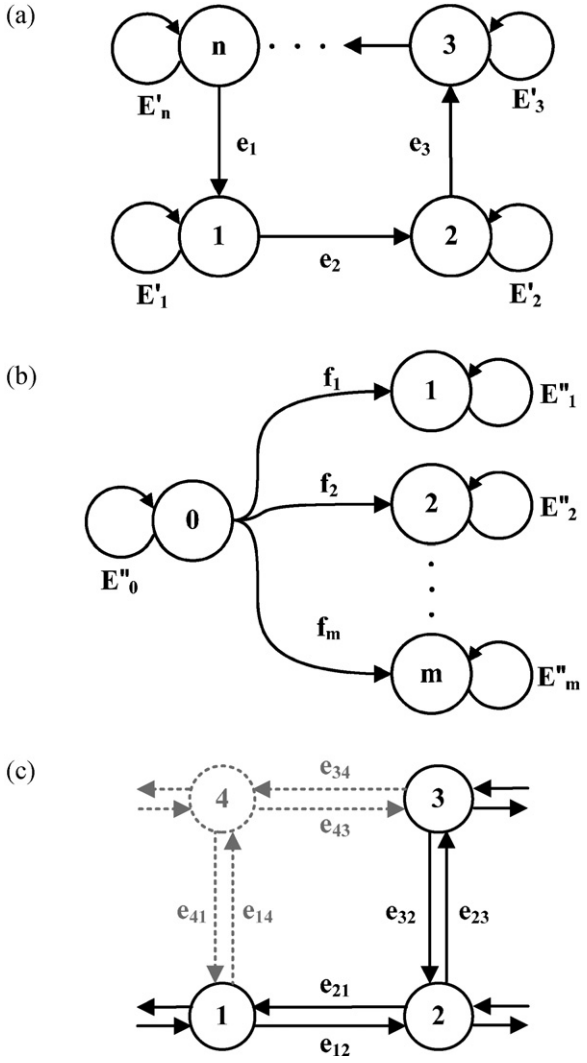


Fig. 8. Generalized specification models. (a) Spec 1; (b) Spec 2; (c) Spec 3.

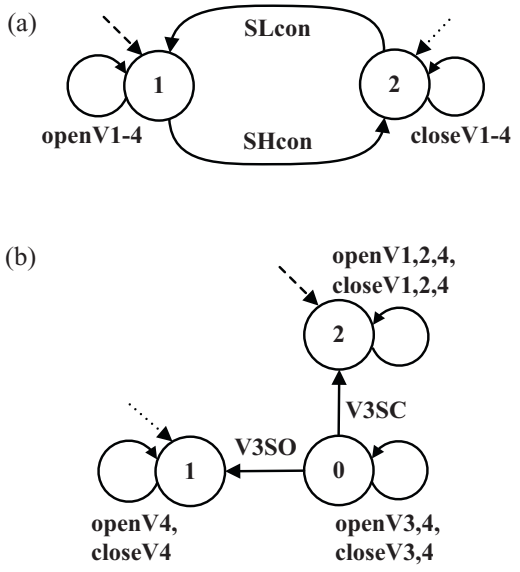


Fig. 9. Specification models used in Example 1: (a) Spec 1; (b) Spec 2.

(a) Only valves V-3 and V-4 can be manipulated when the system is normal, i.e., at state 0;
(b) Only valve V-4 can be operated after *V3SO* occurs;
(c) Only the valve states of V-1, V-2 and V-4 can be altered after *V3SC* occurs.

The corresponding automaton representation is given in Fig. 9(b), in which states 1 and 2 respectively represent the diagnosed initial conditions after *V3SO* and *V3SC*.

- **Spec 3:** *prohibit illegal process configurations.* For illustration convenience, let us consider a fictitious system with two 2-position actuators. Since there are four different combinations of actuator positions, this specification can be imposed by making use of Fig. 8(c). Each state in this automaton can be associated with a unique process configuration and every event represents a *single* actuator action. If any configuration is regarded as illegal, the corresponding state and all the attached events should be

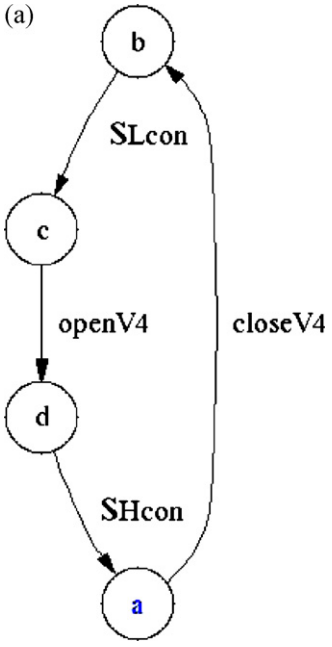


Fig. 10. Admissible supervisor for Example 1: (a) after *V3SO*; (b) after *V3SC*.

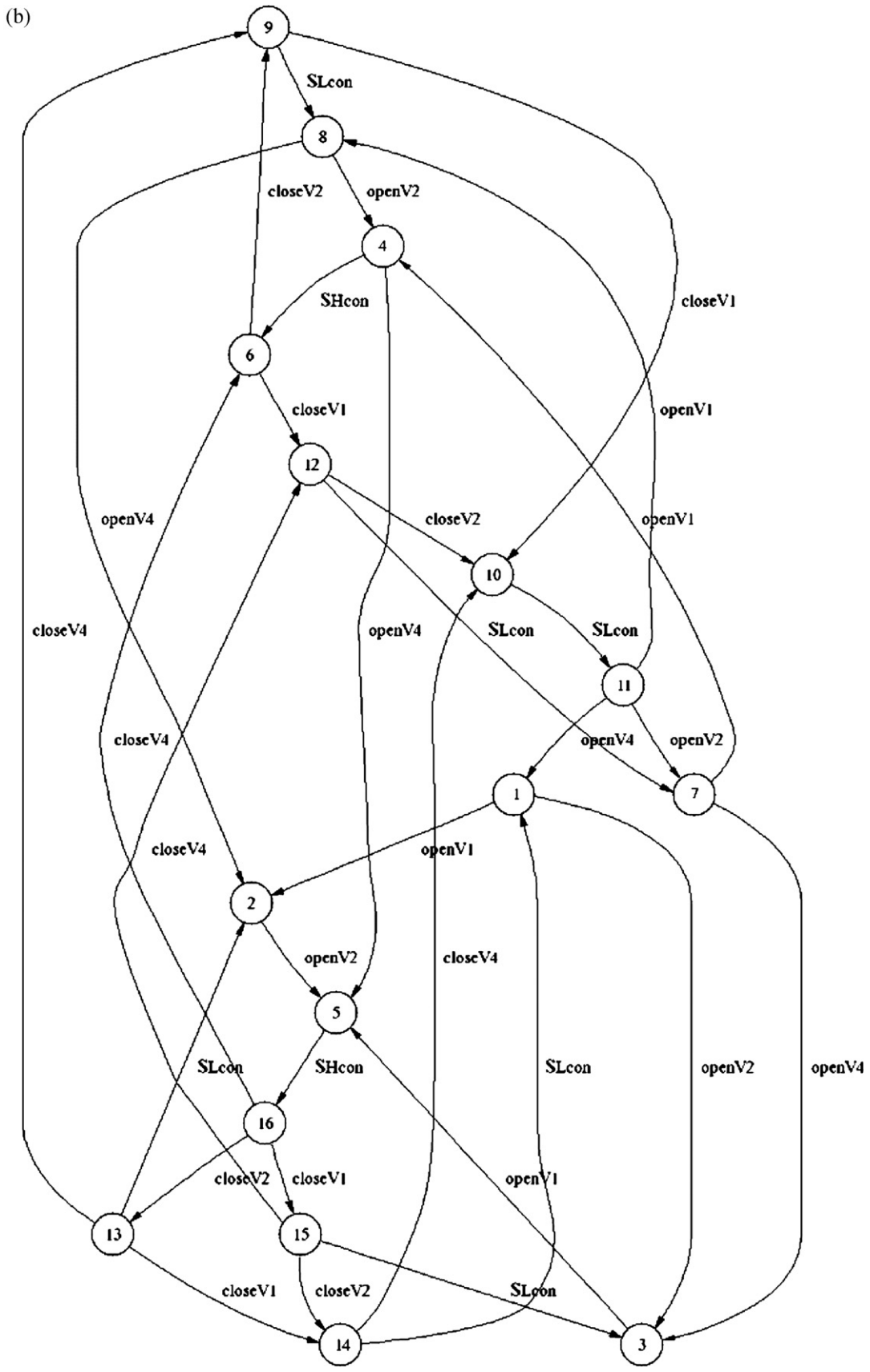


Fig. 10. (*Continued*)

removed from this model. For example, if state 4 is corresponding to an undesired process configuration, then the shaded nodes and arcs in Fig. 8(c) should all be eliminated. Finally, notice that this approach can be easily extended to systems with any number of actuators. Additional examples can be found later in Example 2 (Section 4.4).

3.3. Admissible supervisors

Based on a given failure-induced system state, an admissible supervisor can be built by applying the parallel composition operation (Cassandras & Lafortune, 1999) to combine all aforementioned component and specification automata. The admissible emergency supervisors for failures *V3SO* and *V3SC* in Example 1 are presented in Fig. 10(a) and (b), respectively. States a in the former automaton represent the diagnosable *V3SO*-induced system state reached right after event *SHcon*, whereas State 1 in the latter case is the diagnosable *V3SC*-induced system state reached immediately after event *SLcon* It can be observed that, although many different alternative strings are allowed after *V3SC*, only *one* emergency operating procedure can be adopted to handle failure *V3SO*, i.e., (1) close V-4 when the liquid level is high and (2) open V-4 when the liquid level is low (see Fig. 10(a)).

3.4. Implementable supervisors

The most efficient emergency procedure(s) can be identified by extracting the *supremal controllable sublanguage* (Cassandras & Lafortune, 1999) from the admissible supervisor. For this purpose, two standard types of auxiliary automata can be constructed to define the target state(s) or event(s) of emergency operation and also to set the upper limit on the total number of actuator actions. Let us again consider Example 1 here for illustration convenience:

- **Type I:** *define target state(s)*. A target state can be marked in the auxiliary automaton with double circles. The auxiliary automaton H_{A1} in Fig. 11(a) is produced to specify a termination mechanism for a *periodic* emergency response operation in Example 1. If a failure occurs when liquid level in tank is low, e.g., *V3SC*, the event *SHcon* should occur at least twice so as to ensure realization of repeated operation cycles. By extracting the supremal controllable sublanguage (Lafortune & Teneketzis, 2000) from H_{A1} and the aforementioned admissible supervisor in Fig. 10(b), the terminated admissible supervisor can be built (see Fig. 12).
- **Type II:** *impose upper bound on the total number of actuator actions.* The standard automaton H_{A2} in Fig. 11(b) can be adopted to limit the total number of actuator actions in the emergency procedure for any given system. The symbol β in this automaton represents all possible actuator actions, while α denotes the remaining events. Since the initial state 0 is driven to state n ($n = 0, 1, \ldots, N$) after n actuator actions, the maximum number of actuator actions, i.e., N, in the emergency procedure can be imposed by augmenting the admissible supervisor with this automaton. Notice also that, in order to allow fewer actuator actions to be taken in the emergency response operation, all states are marked in this model. Consequently, this auxiliary automaton facilitates easy identification of all feasible procedures with $n \leq N$ actuator actions and also the most efficient one(s) among them. By extracting the supremal controllable sublanguage (Lafortune & Teneketzis, 2000) from H_{A2} (with $N = 11$) and the terminated admissible supervisor in Fig. 12, a set of implementable supervisors and also the smallest among them (see Fig. 13) can finally be obtained.

For Example 1, the most efficient emergency response procedures, i.e., the ones with minimum actuator actions, are listed in Table 2. It can be observed that, if *V3SC* occurs after event *SLcon*, two equally effective emergency response procedures can be identified from the implementable supervisor. Notice also that each procedure requires only 3 actuator actions.

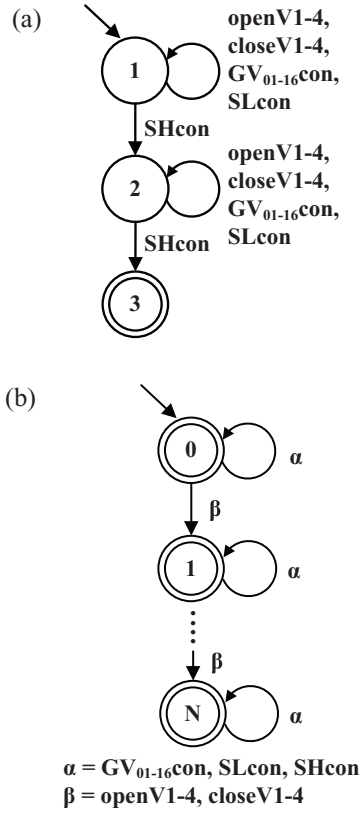


Fig. 11. Auxiliary automata in Example 1: (a) terminating the admissible supervisor when *V3SC* occurs after *LLcon*; (b) limiting the number of actuator actions.

Table 2
Emergency SFCs for *V3SC* in Example 1: (a) operation steps; (b) activation conditions.

(a)

Operation step	Control actions (SFC 1)	Control actions (SFC 2)
OS_0	Failure *V3SC* is diagnosed after SLcon immediately	
OS_1	(1) Open V-1 (2) Open V-2	(1) Open V-1(2) Open V-2
OS_2	Close V-1	Close V-2

(b)

Symbol	Conditions (SFCs 1 & 2)
AC_1	SH
AC_2	SL

4. Application

To test the effectiveness of the proposed approach in realistic systems, the proposed synthesis strategy has been applied to a beer filtration plant (Chung & Lai, 2008; Lai et al., 2007). It should be noted that this example was originally adopted for analyzing *normal* operating procedures, while it is used here for a *new application*, i.e., for generating the emergency response procedures in failure-induced scenarios. In the following case studies, the software tools DESUMA and UMDES (Lafortune & Teneketzis, 2000) were adopted to perform various standard automata-based operations,

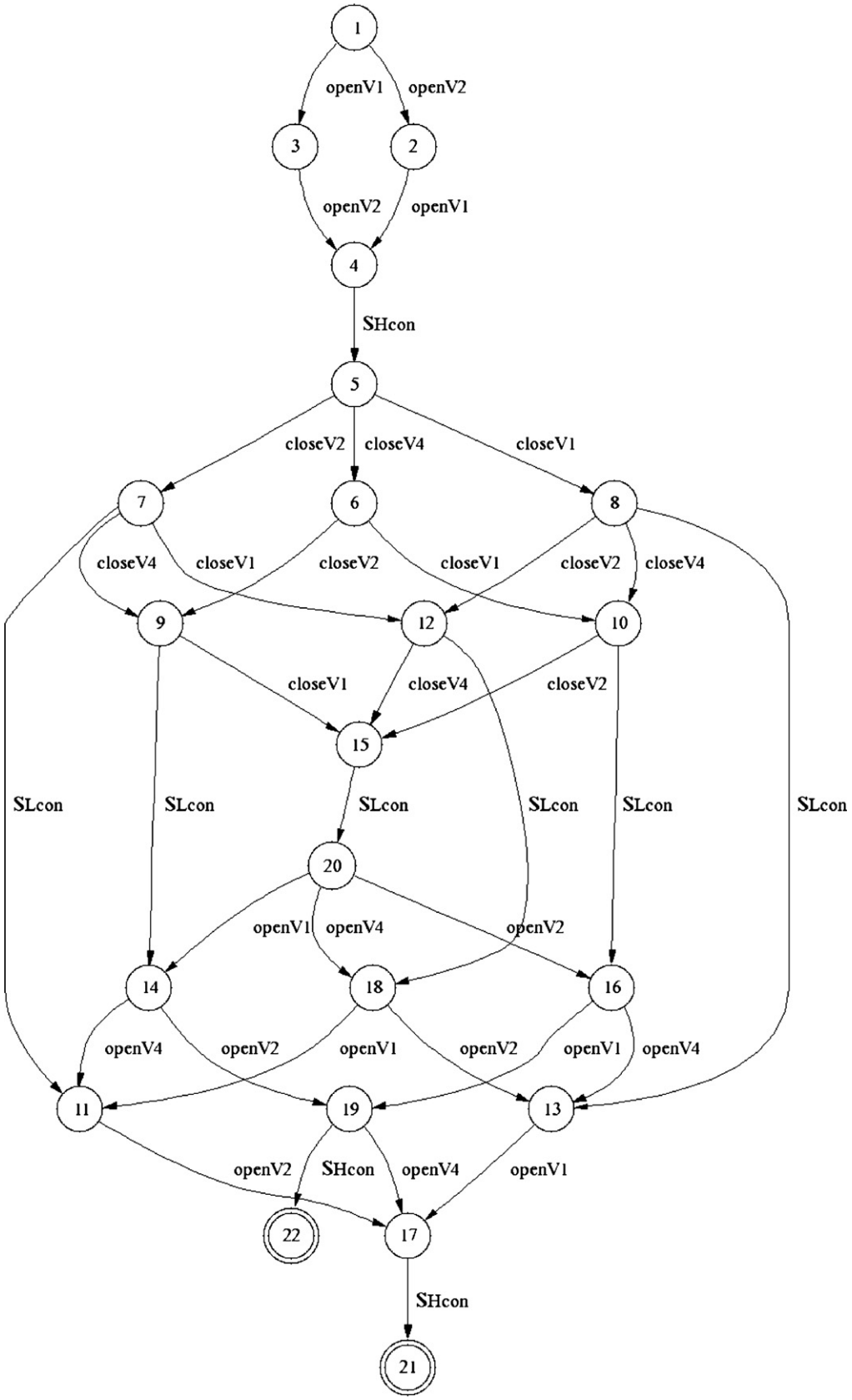


Fig. 12. Terminated admissible supervisor when *V*3*SC* occurs after *SLcon* (Example 1).

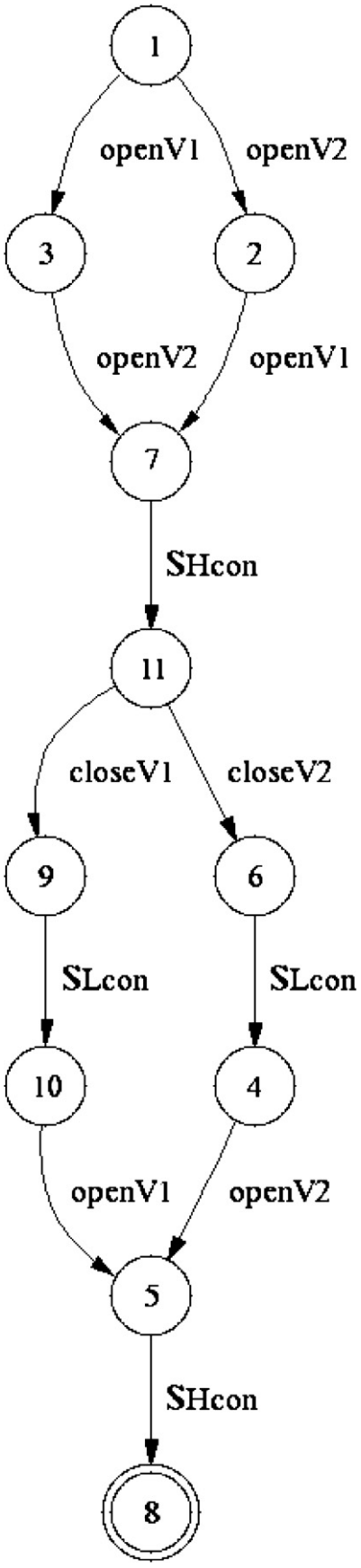


Fig. 13. The smallest implementable supervisor when *V*3*SC* occurs after *SLcon* (Example 1).

Table 3
SFCs under normal operation in Example 2: (a) operation steps; (b) activation conditions.

(a)

Operation step	Control actions
OS_0	Initialization
OS_1	(1) Close V-8; (2) Close V-9; (3) Close V-14; (4) Close V-15; (5) Open V-12; (6) Open V-13
OS_2	(1) Close V-12; (2) Close V-13; (3) Open V-3; (4) Open V-4; (5) Open V-11; (6) Open V-16
OS_3	(1) Close V-3; (2) Close V-11; (3) Close V-16; (4) Open V-5
OS_4	(1) Close V-4; (2) Close V-5; (3) Open V-2; (4) Open V-3; (5) Open V-7; (6) Open V-10
OS_5	(1) Close V-2; (2) Close V-3; (3) Close V-7; (4) Close V-10; (5) Open V-1; (6) Open V-6; (7) Open V-13; (8) Open V-14
OS_6	(1) Close V-1; (2) Close V-6; (3) Close V-13; (4) Open V-8; (5) Open V-9; (6) Open V-15

(b)

Symbol	Conditions
AC_1	Start
AC_2	T1H
AC_3	M1F & M2C & T1L & T2H
AC_4	T2L
AC_5	T2C & T1H
AC_6	M1C & M2F & T1L & T2H
AC_7	T1C & T2L

e.g., parallel composition and supremal controllable sublanguage generation, etc. The detailed system description is first described below:

4.1. System description

The process flow diagram of beer filtration plant is shown in Fig. 14 (Chung & Lai, 2008; Lai et al., 2007). This system consists of two multi-micro system filters (MMS-1 and MMS-2), two beer buffer tanks (T-1 and T-2), a supply and collection system for the cleanser (CIP), and 16 double-disk piston valves (V-1 to V-16). Notice that each valve can be switched to either OPEN or CLOSE position. When a valve is opened, the fluids entering the valve from vertical and horizontal pipelines will be mixed and then flow out via all outlet pipelines, whereas the fluids in vertical and horizontal pipelines flow separately when this valve is at the CLOSE position. There are four basic tasks to be performed in this plant, i.e., filling, filtration, bottling and cleaning. The purpose of filling is to transport fresh beer from a source tank to the buffer tank T-1 by opening either (1) V-2 and V-3 or (2) V-12 and V-13. In the filtration operation, beer is transferred from tank T-1 to T-2 via filter MMS-1 or MMS-2. Valves V-3 and V-4 should be both switched to the OPEN positions in the former case, while V-13 and V-14 must be opened in the latter. Clearly, the filtered beer in T-2 should be moved to the bottling station either by opening V-4 and V-5 or by opening V-14 and V-15. Finally, the tasks of cleaning processing units can also be considered as four different material-transport operations and they are listed below:

- Opening V-8 and V-9 to clean T-1;
- Opening V-7 and V-10 to clean T-2;
- Opening V-1 and V-6 to clean MMS-1;
- Opening V-11 and V-16 to clean MMS-2.

The *normal* operation steps and their activation conditions can be found in Table 3. Notice that, to enhance production efficiency, it is clearly a good practice to clean equipment concurrently with at least one beer processing step. It is required in this procedure that the filters and tanks are cleaned after being used once and twice respectively in every production cycle.

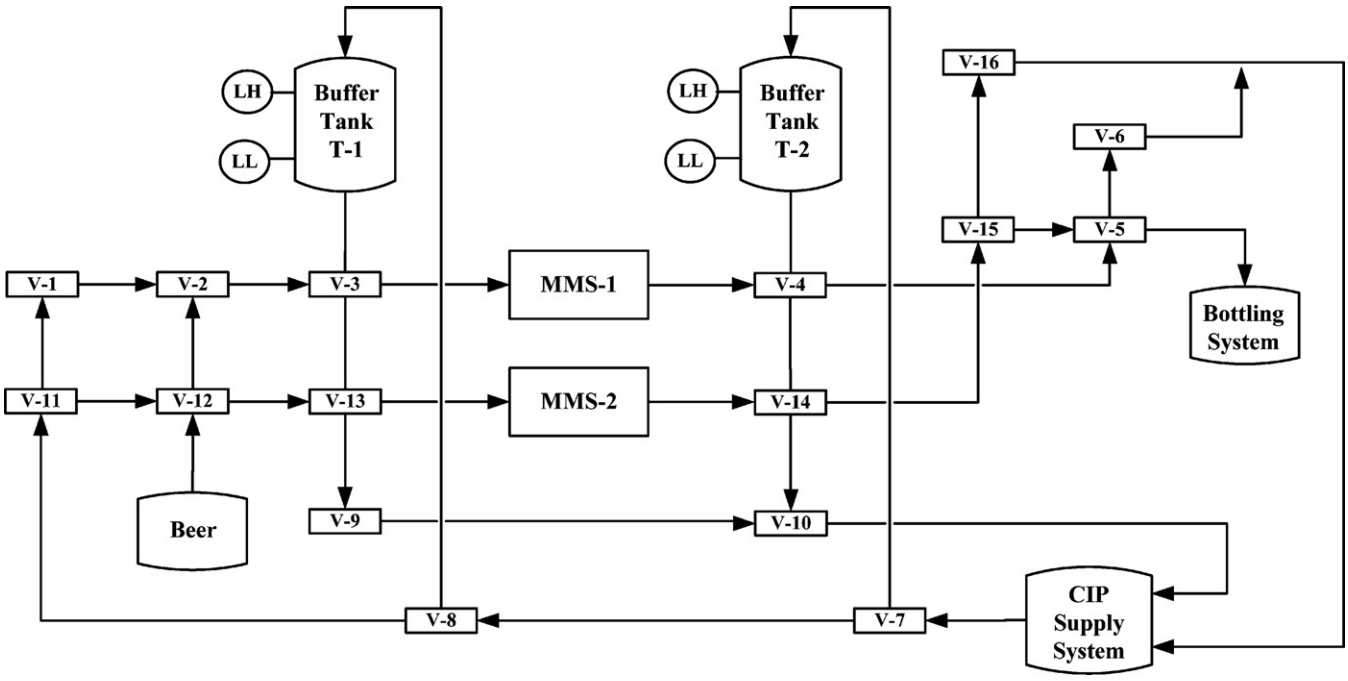


Fig. 14. A beer filtration plant (Example 2) (Chung & Lai, 2008; Lai et al., 2007).

Table 4(a)
Illegal process configurations for unallowable material transfers in Example 2.

Material transfer paths	Path No.	Process configurations	Required conditions	Material transfer paths	Path No.	Process configurations	Required conditions
Source to T-2	1-1	V2O, V4O		T-2 to CIP	6-1	V10O	T2H
	1-2	V12O, V14O			6-2	V14O, V16O	
	1-3	V2O, V3O, V13O, V14O			6-3	V4O, V6O	
	1-4	V3O, V4O, V12O, V13O			6-4	V5O, V6O, V14O, V15O	
Source to BottlingStation	2-1	V2O, V5O		CIP to T-1	7-1	V8O	V9 C
	2-2	V12O, V15O			7-2	V1O, V3O	
	2-3	V2O, V4O, V14O, V15O			7-3	V11O, V13O	
	2-4	V2O, V3O, V13O, V15O			7-4	V2O, V3O, V11O, V12O	
	2-5	V3O, V5O, V12O, V13O		CIP to T-2	8-1	V7O	V10 C
	2-6	V4O, V5O, V12O, V14O			8-2	V1O, V4O	
Source to CIP	3-1	V2O, V6O			8-3	V11O, V14O	
	3-2	V12O, V16O			8-4	V1O, V3O, V13O, V14O	
	3-3	V2O, V4O, V14O, V16O			8-5	V3O, V4O, V11O, V13O	
	3-4	V2O, V3O, V13O, V16O			8-6	V2O, V4O, V11O, V12O	
	3-5	V3O, V6O, V12O, V13O		CIP to bottling station	9-1	V1O, V5O	
	3-6	V4O, V6O, V12O, V14O			9-2	V11O, V15O	
	3-7	V2O, V3O, V9O			9-3	V1O, V3O, V13O, V15O	
	3-8	V2O, V4O, V10O			9-4	V1O, V4O, V14O, V15O	
	3-9	V9O, V12O, V13O			9-5	V2O, V5O, V11O, V12O	
	3-10	V10O, V12O V14O			9-6	V3O, V5O, V11O, V13O	
	3-11	V5O, V6O, V12O, V15O			9-7	V4O, V5O, V11O, V14O	
	3-12	V2O, V3O, V10O, V13O, V14O		CIP to CIP	10-1	V1O, V3O, V9O	
	3-13	V3O, V4O, V10O, V12O, V13O			10-2	V1O, V4O, V10O	
T-1 to BottlingStation	4-1	V3O, V5O			10-3	V9O, V11O, V13O	
	4-2	V13O, V15O			10-4	V11O, V10O, V14O	
	4-3	V4O, V5O, V13O, V14O			10-5	V1O, V3O, V10O, V13O, V14O	
	4-4	V3O, V4O, V14O, V15O			10-6	V3O, V4O, V10O, V11O, V13O	
T-1 to CIP	5-1	V9O	T1H		10-7	V2O, V4O, V10O, V11O, V12O	
	5-2	V13O, V16O			10-8	V2O, V3O, V9O, V11O, V12O	
	5-3	V3O, V6O			10-9	V1O, V3O, V13O, V16O	
	5-4	V3O, V4O, V10O			10-10	V1O, V4O, V14O, V16O	
	5-5	V10O, V13O, V14O			10-11	V2O, V6O, V11O, V12O	
	5-6	V3O, V4O, V14O, V16O			10-12	V3O, V6O, V11O, V13O	
	5-7	V4O, V6O, V13O, V14O			10-13	V4O, V6O, V11O, V14O	
	5-8	V5O, V6O, V13O, V15O			10-14	V1O, V4O, V6O, V14O	
					10-15	V5O, V6O, V11O, V15O	

4.2. Process configurations

As mentioned previously, only four distinct types of tasks are to be accomplished in the beer filtration plant and each can be facilitated by performing a material-transfer operation via one of several alternative paths. Since all other material-transfer paths do not serve these purposes, the corresponding process configurations should be considered as "illegal" and they are exhaustively enumerated in Table 4(a). For example, the fresh beer from source tank may be transported *illegally* to the buffer tank T-2 by keeping all items in any of the following valve sets at the OPEN positions:

(1) {V-2, V-4},
(2) {V-12, V-14},
(3) {V-2, V-3, V-13, V-14},
(4) {V-3, V-4, V-12, V-13}.

Notice that, as long as the chosen valves are open, the remaining valve states are irrelevant and, therefore, each material-transfer path listed in Table 4(a) could be facilitated by more than one illegal process configuration.

In addition, since beer and cleanser are not allowed to be mixed in this system, the barriers between to two materials must always be kept intact. Consequently, although material-transfer paths may not be formed when such barriers are removed, the corresponding process configurations given in Table 4(b) should also be regarded as illegal. Notice that only the required OPEN valve states are specified in this table.

Finally, it should be emphasized that all process configurations are considered as legal in this study except those listed in Tables 4(a) and 4(b).

Table 4(b)
Illegal process configurations for non-transfer but mix of beer and cleaner in the pipeline (Example 2).

Path No.	Process configurations
11-1	V1O, V2O
11-2	V11O, V12O
11-3	V1O, V3O, V12O, V13O
11-4	V1O, V4O, V12O, V14O
11-5	V1O, V5O, V12O, V15O
11-6	V2O, V3O, V11O, V13O
11-7	V2O, V4O, V11O, V14O
11-8	V2O, V5O, V11O, V15O
11-9	V2O, V3O, V8O
11-10	V2O, V4O, V8O, V13O, V14O
11-11	V2O, V5O, V8O, V13O, V14O, V15O
11-12	V2O, V4O, V7O
11-13	V2O, V5O, V7O, V14O, V15O
11-14	V8O, V12O, V13O
11-15	V3O, V4O, V8O, V12O, V14O
11-16	V3O, V5O, V8O, V12O, V15O
11-17	V7O, V12O, V14O
11-18	V4O, V5O, V7O, V12O, V15O

4.3. Component models

The proposed methodology has been adopted to build the component models in this example. These models are briefly described as follow:

- Level 1: The normal PLC model can be constructed in a straightforward fashion according to Table 3 (see Fig. 15). Notice that this model is used to represent only the normal operation cycle. The failure mechanisms can be introduced by following the modeling approach described in Fig. 7(a).

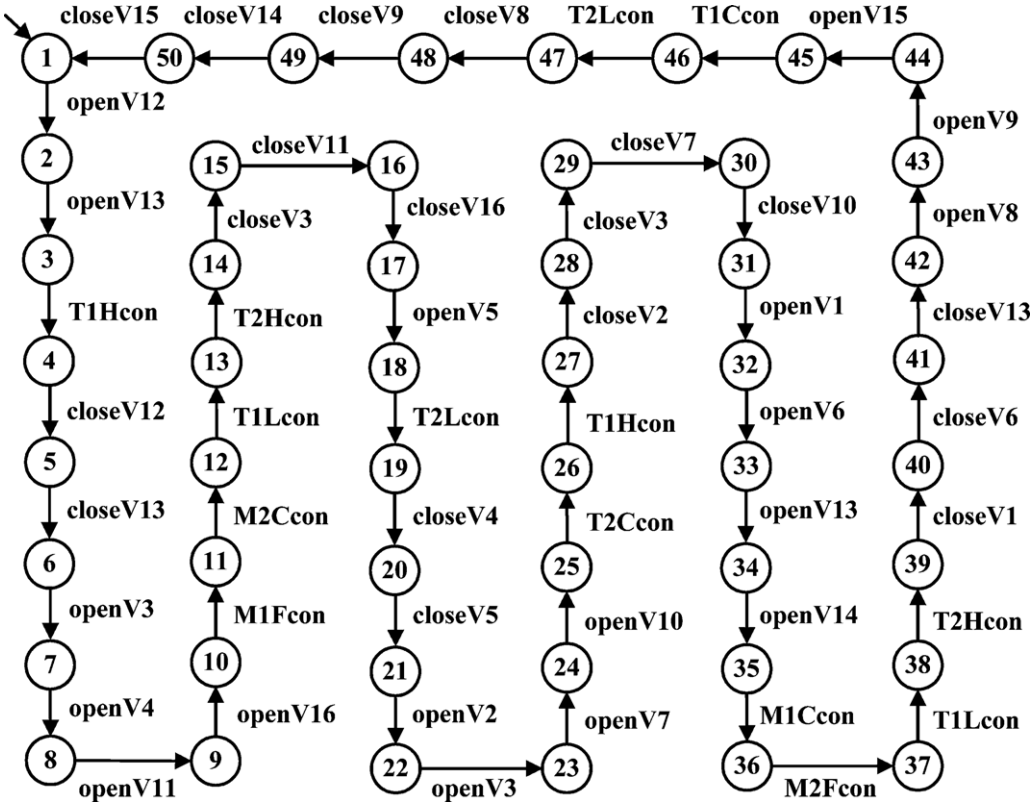


Fig. 15. Automaton model of level 1 for the normal operations in Example 2.

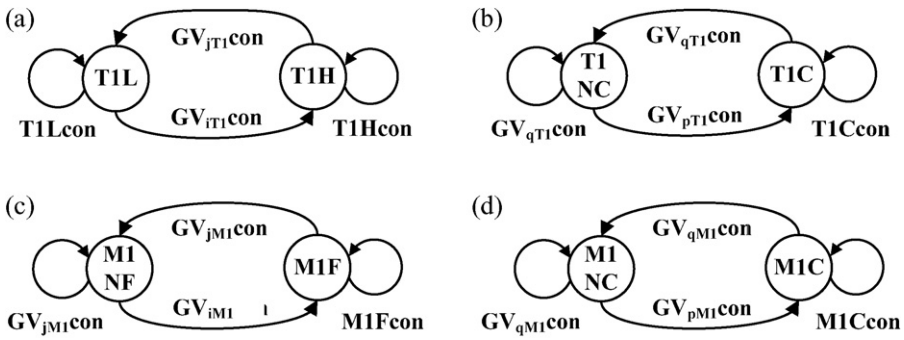


Fig. 16. Automaton models of level 3 in Example 2: (a) T-1 level model; (b) T-1 cleaning model; (c) MMS-1 filtration model; (d) MMS-1 cleaning model.

- Level 2: The automaton representations of level-2 components, i.e., valves V-1 to V-16, should be the same as that described in Fig. 7(b). For the sake of brevity, these models are not repeated here.
- Level 3: The level-3 components, i.e., the buffer tanks and the filters, can be modeled with the automata presented in Fig. 16. Their main features are outlined below:
  - The beer level in tank T-1 can be described with the automaton in Fig. 16(a), while the presence/absence of cleanser in this tank is modeled in Fig. 16(b). States $T1L$ and $T1H$ represent the beer level in T-1 is low and high, respectively; States $T1C$ and $T1NC$ reflect whether or not the cleaning operation of T-1 is in progress; Events $T1Lcon$ and $T1Hcon$ are used to respectively represent the scenarios that beer in tank T-1 stays at high and low levels for a sufficiently long period; $T1Ccon$ means that T-1 remains clean for a long enough time. Notice that $GV_{iT1}con$ and $GV_{jT2}con$ denote the events of maintaining the specified process configuration(s) to facilitate $T1L$-to-$T1H$ and $T1H$-to-$T1L$ transition processes respectively, while $GV_{pT1}con$ and $GV_{qT1}con$ represent the events causing $T1NC$-to-$T1C$ and $T1C$-to-$T1NC$ processes, respectively.
  - The automaton models of MMS-1 in Fig. 16(c) and (d) respectively reflect whether or not the filtration and cleaning operations take place. The state $M1F$ reflects filter MMS-1 is in service, while and $M1NF$ denotes the opposite condition, i.e., not in service. The symbols $M1C$ and $M1NC$ are used to represent whether or not the cleaning operation of MMS-1 is in progress. Events $M1Fcon$ and $M1Ccon$ can be associated with the scenarios that MMS-1 stays at the in-service state of the filtration operation and in-progress state of the cleaning operation, respectively. Notice also that $GV_{iM1}con$ and $GV_{jM1}con$ represent the events of maintaining the corresponding process configurations which result in the $M1NF$-to-$M1F$ and $M1F$-to-$M1NF$ transitions, respectively, while $GV_{pM1}con$ and $GV_{qM1}con$ facilitate $M1NC$-to-$M1C$ and $M1C$-to-$M1NC$ processes, respectively.

  Notice that the configuration maintaining events in the above automata will be more explicitly identified later in the descriptions of specific emergency supervisors. Finally, it should be noted that tank T-2 and filter MMS-2 can also be modeled with the same approach.
- Level 4: Due to the assumption that the probability of any sensor malfunction is negligibly low, the sensor model is again omitted in this example and the online measurement readings should be identical to the tank states.

Table 5
The control specifications for illegal conditional actuator action(s) derived according to configurations 5-1, 6-1, 7-1, and 8-1 in Table 4(a) respectively (Example 2).

Specification No	Description
S_1	Avoid opening valve V-9 when tank T-1 is at high level.
S_2	Avoid opening valve V-10 when tank T-2 is at high level.
S_3	Avoid opening valve V-8 when valve V-9 is at closed position.
S_4	Avoid opening valve V-7 when valve V-10 is at closed position.

Table 6
The control specifications for illegal unconditional actuator actions generated from Table 4 (Example 2).

Specification No	Description
S_5	Avoid opening both valves V-1 and V-2 simultaneously.
S_6	Avoid opening both valves V-1 and V-3 simultaneously.
S_7	Avoid opening both valves V-1 and V-4 simultaneously.
S_8	Avoid opening both valves V-1 and V-5 simultaneously.
S_9	Avoid opening both valves V-2 and V-4 simultaneously.
S_{10}	Avoid opening both valves V-2 and V-5 simultaneously.
S_{11}	Avoid opening both valves V-2 and V-6 simultaneously.
S_{12}	Avoid opening both valves V-3 and V-5 simultaneously.
S_{13}	Avoid opening both valves V-3 and V-6 simultaneously.
S_{14}	Avoid opening both valves V-4 and V-6 simultaneously.
S_{15}	Avoid opening both valves V-8 and V-3 simultaneously.
S_{16}	Avoid opening both valves V-8 and V-13 simultaneously.
S_{17}	Avoid opening both valves V-7 and V-4 simultaneously.
S_{18}	Avoid opening both valves V-7 and V-14 simultaneously.
S_{19}	Avoid opening both valves V-11 and V-13 simultaneously.
S_{20}	Avoid opening both valves V-11 and V-14 simultaneously.
S_{21}	Avoid opening both valves V-11 and V-15 simultaneously.
S_{22}	Avoid opening both valves V-12 and V-11 simultaneously.
S_{23}	Avoid opening both valves V-12 and V-14 simultaneously.
S_{24}	Avoid opening both valves V-12 and V-15 simultaneously.
S_{25}	Avoid opening both valves V-12 and V-16 simultaneously.
S_{26}	Avoid opening both valves V-13 and V-15 simultaneously.
S_{27}	Avoid opening both valves V-13 and V-16 simultaneously.
S_{28}	Avoid opening both valves V-14 and V-16 simultaneously.
S_{29}	Avoid opening three valves V-2, V-3 and V-9 simultaneously.
S_{30}	Avoid opening three valves V-3, V-4 and V-10 simultaneously.
S_{31}	Avoid opening three valves V-12, V-13 and V-9 simultaneously.
S_{32}	Avoid opening three valves V-10, V-13 and V-14 simultaneously.
S_{33}	Avoid opening four valves V-2, V-3, V-13 and V-14 simultaneously.
S_{34}	Avoid opening four valves V-3, V-4, V-12 and V-13 simultaneously.
S_{35}	Avoid opening four valves V-3, V-4, V-14 and V-15 simultaneously.
S_{36}	Avoid opening four valves V-3, V-4, V-14 and V-16 simultaneously.
S_{37}	Avoid opening four valves V-4, V-5, V-13 and V-14 simultaneously.
S_{38}	Avoid opening four valves V-4, V-6, V-13 and V-14 simultaneously.
S_{39}	Avoid opening four valves V-5, V-6, V-14 and V-15 simultaneously.

Table 7
Legal valve combinations for the events of level-3 automaton models in Fig. 16: (a) Scenario 1; (b) Scenario 2 (Example 2).

(a)

V-1	V-2	V-3	V-4	V-5	V-6	V-7	V-8	V-9	V-10	V-11	V-12	V-13	V-14	V-15	V-16	Symbol
O	C	C	C	C	C	C	C	C	C	C	O	O	C	C	C	GV_{01}
O	C	C	C	C	C	C	C	C	O	C	O	O	C	C	C	GV_{02}
O	C	C	C	C	C	O	C	C	O	C	O	O	C	C	C	GV_{03}
O	C	C	C	C	O	C	C	C	C	C	O	O	C	C	C	GV_{04}
O	C	C	C	C	O	C	C	C	O	C	O	O	C	C	C	GV_{05}
O	C	C	C	C	O	O	C	C	O	C	O	O	C	C	C	GV_{06}
O	C	C	C	C	C	C	C	C	C	C	C	O	O	C	C	GV_{07}
O	C	C	C	C	O	C	C	C	C	C	C	O	O	C	C	GV_{08}
O	C	C	C	C	C	C	C	C	C	C	C	C	O	O	C	GV_{09}
O	C	C	C	C	C	C	C	O	C	C	C	C	O	O	C	GV_{10}
O	C	C	C	C	C	C	O	O	C	C	C	C	O	O	C	GV_{11}
O	C	C	C	C	O	C	C	C	C	C	C	C	O	O	C	GV_{12}
O	C	C	C	C	O	C	C	O	C	C	C	C	O	O	C	GV_{13}
O	C	C	C	C	O	C	O	O	C	C	C	C	O	O	C	GV_{14}
O	C	C	C	C	C	C	C	C	C	O	C	C	C	C	O	GV_{15}
O	C	C	C	C	C	C	C	C	O	O	C	C	C	C	O	GV_{16}
O	C	C	C	C	C	C	C	O	C	O	C	C	C	C	O	GV_{17}
O	C	C	C	C	C	C	C	O	O	O	C	C	C	C	O	GV_{18}

(b)

V-1	V-2	V-3	V-4	V-5	V-6	V-7	V-8	V-9	V-10	V-11	V-12	V-13	V-14	V-15	V-16	Symbol
C	O	O	C	C	C	C	C	C	C	O	C	C	C	C	C	GV_{19}
C	O	O	C	C	C	C	C	C	C	O	C	C	C	C	O	GV_{20}
C	O	O	C	C	C	C	C	C	O	O	C	C	C	C	C	GV_{21}
C	O	O	C	C	C	C	C	C	O	O	C	C	C	C	O	GV_{22}
C	O	O	C	C	C	O	C	C	O	O	C	C	C	C	C	GV_{23}
C	O	O	C	C	C	O	C	C	O	O	C	C	C	C	O	GV_{24}
C	C	O	O	C	C	C	C	C	C	O	C	C	C	C	C	GV_{25}
C	C	O	O	C	C	C	C	C	C	O	C	C	C	C	O	GV_{26}
C	C	C	O	O	C	C	C	C	C	O	C	C	C	C	C	GV_{27}
C	C	C	O	O	C	C	C	C	C	O	C	C	C	C	O	GV_{28}
C	C	C	O	O	C	C	C	O	C	O	C	C	C	C	C	GV_{29}
C	C	C	O	O	C	C	C	O	C	O	C	C	C	C	O	GV_{30}
C	C	C	O	O	C	C	O	O	C	O	C	C	C	C	C	GV_{31}
C	C	C	O	O	C	C	O	O	C	O	C	C	C	C	O	GV_{32}
O	C	C	C	C	O	C	C	C	C	O	C	C	C	C	C	GV_{33}
O	C	C	C	C	O	C	C	C	O	O	C	C	C	C	C	GV_{34}
O	C	C	C	C	O	C	C	O	C	O	C	C	C	C	C	GV_{35}
O	C	C	C	C	O	C	C	O	O	O	C	C	C	C	C	GV_{36}

Notice that the letter O means open, while C means close.

4.4. Control specifications

First of all, it is obviously necessary to build specification models corresponding to those adopted in Example 1, i.e.,

- By following the modeling rationale adopted in Fig. 9(a), the automaton in Fig. 17(a) can be constructed to permit actuator actions before or after both $GV_{iT1}con$ and $T1Hcon$ take place. Notice that $GV_{iT1}con$ denote the process configuration(s) which could result in $T1Hcon$. It should be stressed that the same approach can be used to impose the required precedence order of process configurations, other level-3 events (e.g., $T1Lcon$) and actuator actions.
- By following the modeling rationales utilized in Fig. 9(b), the automaton in Fig. 17(b) facilitates representation of the conditional events, i.e., a particular group of actuator actions can be taken only at a given system state. All valves are allowed to be manipulated during normal operation in this system, but V-1 is inoperable after failure $V1SO$ occurs. Notice that the same modeling approach can also be adopted for other failure-induced scenarios.

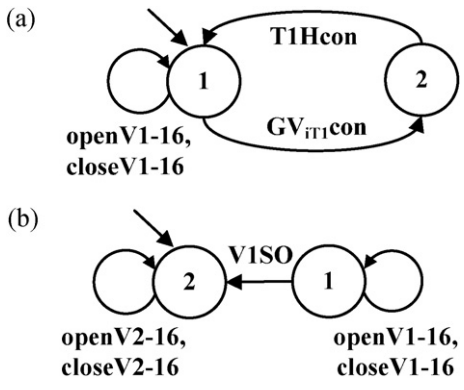


Fig. 17. Control specification models associated with (a) Spec model for $T1Hcon$; (b) Spec model after $V1SO$ (Example 2).

In this example, additional control specifications are stipulated to eliminate the possibilities of forming the illegal process configurations given in Tables 4(a) and 4(b) while still maintaining uninterrupted production. More specifically, these specifications can be classified into two types, i.e., the illegal conditional and unconditional actuator action(s) (see Tables 5 and 6). Notice that the first type of constraints, i.e., S_1–S_4 in Table 5, can be derived according to configurations 5-1, 6-1, 7-1, and 8-1 in Table 4(a), respectively. On the other hand, the specifications in Table 6 (i.e.,

S_5–S_{39}) can be generated easily by carrying out the following simple enumeration procedure (Procedure I):

1. Let $n = 2$.
2. Search for and list an illegal process configuration in Tables 4(a) and 4(b) that requires exactly n valves to be at the OPEN positions.
3. Enumerate all configurations that contain the n OPEN states identified in Step 2.
4. Remove the process configurations which have already been enumerated previously.
5. Repeat Steps 2–4 until all n-valve combinations are exhausted.
6. Let $n = n + 1$. If $n \leq 5$, repeat Steps 2–6. Otherwise, stop.

For instance, it can be observed that configuration 1-1 can be prevented with specification S_9, i.e., avoid opening both V-2 and V-4 simultaneously. As a result, the configurations implied by this specification, i.e., 2-3, 3-3, 3-8, 8-6, 10-7, 11-7, 11-10 and 11-12, should all be removed. Notice that the automaton representations of control specifications in Table 5 can be constructed with the modeling approach described in Fig. 9(a), while those in Table 6 can be represented according to Fig. 8(c).

4.5. Scenario 1

Let us first consider the failure-induced scenarios associated with *V1SO*, i.e., valve V-1 sticks at the OPEN position. It is assumed in this case that *V1SO* can be diagnosed when the controller command to close V-1 is being executed. This is due to the fact that the actuator actions (i.e., close V-1, V-6 and V-13, and open V-8, V-9 and V-15) are taken immediately after the activation condition *T2Hcon* (see Fig. 15). Thus, it can be further deduced that failure *V1SO* may occur at state 45 when

(a) T-1 is at low level and the cleaning operation is not in progress
(b) T-2 is at high level and the cleaning operation is not in progress
(c) MMS-1 is being cleaned and not in service
(d) MMS-2 is in service and not being cleaned
(e) V-1, V-8, V-9, V-14 and V-15 are at the OPEN positions, and
(f) all other valves are closed.

By excluding illegal configurations listed in Tables 5 and 6, the legal valve combinations in this scenario can be exhaustively enumerated (see Table 7(a)) and, in addition, the resulting equipment conditions of the tanks and filters can be determined as well. Notice that the events considered in the automaton representations of level-3 components in Fig. 16 are clearly defined in Table 8(a). Notice also that Tables 7(a) and 8(a) can be obtained with another enumeration procedure (Procedure II), which is outlined below:

1. Enumerate all failure-induced configurations.
2. From the results of Step 1, identify all configurations which could facilitate the four basic tasks, i.e., filling, filtration, bottling and cleaning.
3. Let $n = 1$.
4. From the configurations found in Step 2, search for those that violate specification S_n.
5. Let $n = n + 1$. If $n \leq 39$, then repeat Steps 4–5. Otherwise, stop.

It can be observed that the aforementioned legal valve combinations, i.e., GV_{01}–GV_{18}, could lead to changes in level-3 component states. Event GV_{iT1} in Fig. 16(a) can be interpreted as GV_{01}–GV_{06}, while GV_{jT1} in the same figure should be GV_{07}–GV_{08}. Notice that the definitions of other state-transition events in other level-3 component models, i.e., Fig. 16(b), (c) and (d), can also be found in Table 8(a).

Table 8
Symbols of level-3 automaton models in Example 2: (a) Scenario 1; (b) Scenario 2.

(a)

Automaton	Symbols
T-1 Level	iT1 = 01–06
	jT1 = 07, 08
T-1 Cleaning	pT1 = 11, 13
	qT1 = 01–10, 12, 14–18
T-2 Level	iT2 = 07, 08
	jT2 = 09–14
T-2 Cleaning	pT2 = 06, 03
	qT2 = 01, 02, 04, 05, 07–18
MMS-1 Filtration	iM1 = nothing
	jM1 = 01–18
MMS-1 Cleaning	pM1 = 04–06, 08, 12–14
	qM1 = 01–03, 07, 09–11, 15–18
MMS-2 Filtration	iM2 = 07, 08
	jM2 = 01–06, 09–18
MMS-2 Cleaning	pM2 = 11, 15-18
	qM2 = 01–10, 12–14

(b)

Automaton	Symbols
T-1 Level	iT1 = 19–24
	jT1 = 25, 26
T-1 Cleaning	pT1 = 31, 32
	qT1 = 19–30, 33–36
T-2 Level	iT2 = 25, 26
	jT2 = 27–32
T-2 Cleaning	pT2 = 23, 24
	qT2 = 19–22, 25–36
MMS-1 Filtration	iM1 = 25, 26
	jM1 = 19–24, 27–36
MMS-1 Cleaning	pM1 = 33–36
	qM1 = 19–32
MMS-2 Filtration	iM2 = nothing
	jM2 = 19–36
MMS-2 Cleaning	pM2 = 20, 22, 24, 26, 28, 30, 32
	qM2 = 19, 21, 23, 25, 27, 29, 31, 33–36

By assembling the component models and specification models, an admissible supervisor that ensures safe operation during emergency situations can then be produced for the present scenario. The two auxiliary automata described below can be incorporated next to identify the most appropriate operation procedures:

- For the purpose of maintaining a periodic operation cycle, the automaton in Fig. 18 can be used to terminate the corresponding emergency response procedure after performing a complete cycle of production operation, i.e., finishing the bottling operations three times. Notice that, according to the given normal operating procedure, tanks T-1 and T-2 should be cleaned after being used twice, while filters MMS-1 and MMS-2 should be cleaned after being used only once. Since failure *V1SO* is diagnosable at state 45 in Fig. 15, the termination mechanisms can be incorporated on the basis of the following requirements:
  1. T-1 must be cleaned before use,
  2. T-2 must be cleaned after being used twice,
  3. MMS-1 should be cleaned after being used once, and
  4. MMS-2 should be cleaned before use.

  More specifically, the following automata can be constructed to satisfy these requirements by following the modeling rationale adopted in Fig. 11(a):
  - The auxiliary automaton in Fig. 18(a) is built to regulate T-2 behavior. It is produced to specify the termination mechanism after performing the bottling operations three times, i.e., after the trace *T2Lcon-T2Lcon-T2Ccon-T2Lcon*. Thus, state 5 in this automaton is marked as the target state to prevent any further event.

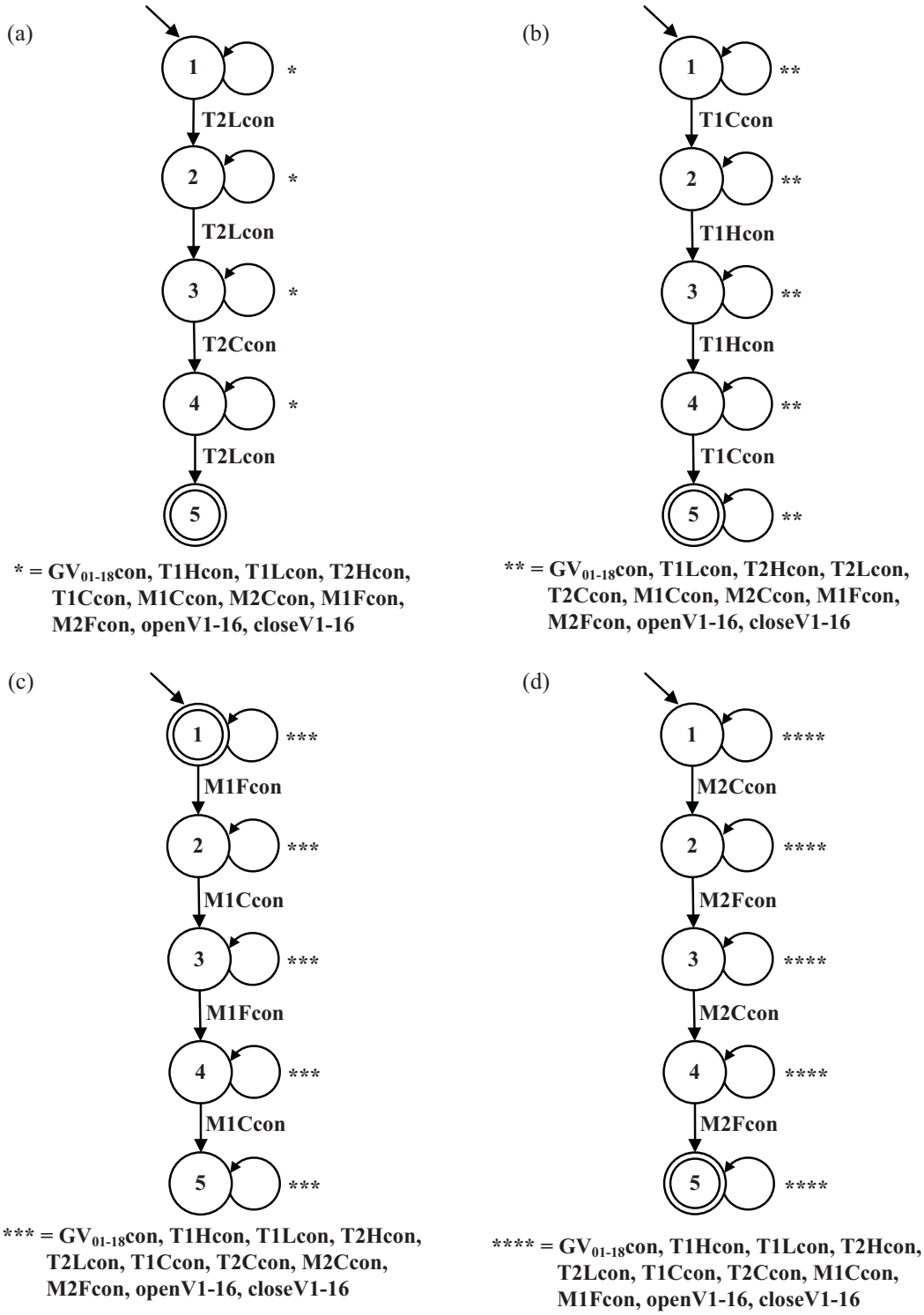


Fig. 18. Auxiliary automaton models (Scenario 1 in Example 2) for: (a) termination mechanism associated with T-2; (b) target state specification associated with T-1; (c) target state specification associated with MMS-1; (d) target state specification associated with MMS-2.

- The auxiliary automaton in Fig. 18(b) is built to limit T-1 behavior. Note that the target state is assigned after the trace *T1Ccon-T1Hcon-T1Hcon-T1Ccon*.
- The auxiliary automaton in Fig. 18(c) is built to impose constraint on MMS-1. Note that initial condition is the target state. This is due to the fact that MMS-1 should not be used after *V1SO*.
- The auxiliary automaton in Fig. 18(d) is built to constrain the behavior of MMS-2, which is used solely for the purpose of ending operation after trace *M2Ccon-M2Fcon-M2Ccon-M2Fcon*.

• By following the modeling rationale adopted in Fig. 11(b), the automaton in Fig. 19 can be used to limit the number of actuator actions.

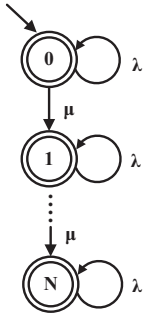
Table 9
Emergency SFCs for scenario 1 in Example 2: (a) Operation steps; (b) Activation conditions.

(a)

Operation Step	Control Actions (SFC 1)	Control Actions (SFC 2)	Control Actions (SFC 3)	Control Actions (SFC 4)
OS_0	Failure V1SO is diagnosed	Failure V1SO is diagnosed	Failure V1SO is diagnosed	Failure V1SO is diagnosed
OS_1	(1) Close V-8	(1) Close V-8	(1) Close V-8	(1) Close V-8
	(2) Close V-9	(2) Close V-14	(2) Close V-9	(2) Close V-14
	(3) Close V-14	(3) Close V-15	(3) Close V-14	(3) Close V-15
	(4) Close V-15	(4) Open V-11	(4) Close V-15	(4) Open V-11
	(5) Open V-11	(5) Open V-16	(5) Open V-11	(5) Open V-16
	(6) Open V-16		(6) Open V-16	
OS_2	(1) Close V-11	(1) Close V-9	(1) Close V-11	(1) Close V-9
	(2) Close V-16	(2) Close V-11	(2) Close V-16	(2) Close V-11
	(3) Open V-12	(3) Close V-16	(3) Open V-12	(3) Close V-16
	(4) Open V-13	(4) Open V-12	(4) Open V-13	(4) Open V-12
		(5) Open V-13		(5) Open V-13
OS_3	(1) Close V-12	(1) Close V-12	(1) Close V-12	(1) Close V-12
	(2) Open V-14	(2) Open V-14	(2) Open V-14	(2) Open V-14
OS_4	(1) Close V-13	(1) Close V-13	(1) Close V-13	(1) Close V-13
	(2) Open V-15	(2) Open V-15	(2) Open V-15	(2) Open V-15
OS_5	(1) Close V-14	(1) Close V-14	(1) Close V-14	(1) Close V-14
	(2) Close V-15	(2) Close V-15	(2) Close V-15	(2) Close V-15
	(3) Open V-10	(3) Open V-10	(3) Open V-11	(3) Open V-11
	(4) Open V-11	(4) Open V-11	(4) Open V-16	(4) Open V-16
	(5) Open V-16	(5) Open V-16		
OS_6	(1) Close V-11	(1) Close V-11	(1) Close V-11	(1) Close V-11
	(2) Close V-16	(2) Close V-16	(2) Close V-16	(2) Close V-16
	(3) Open V-7	(3) Open V-7	(3) Open V-7	(3) Open V-7
	(4) Open V-12	(4) Open V-12	(4) Open V-10	(4) Open V-10
	(5) Open V-13	(5) Open V-13	(5) Open V-12	(5) Open V-12
			(6) Open V-13	(6) Open V-13
OS_7	(1) Close V-7	(1) Close V-7	(1) Close V-7	(1) Close V-7
	(2) Close V-10	(2) Close V-10	(2) Close V-10	(2) Close V-10
	(3) Close V-12	(3) Close V-12	(3) Close V-12	(3) Close V-12
	(4) Open V-14	(4) Open V-14	(4) Open V-14	(4) Open V-14
OS_8	(1) Close V-13	(1) Close V-13	(1) Close V-13	(1) Close V-13
	(2) Open V-8	(2) Open V-8	(2) Open V-8	(2) Open V-8
	(3) Open V-9	(3) Open V-9	(3) Open V-9	(3) Open V-9
	(4) Open V-15	(4) Open V-15	(4) Open V-15	(4) Open V-15

(b)

Symbol	Conditions (SFCs 1–4)
AC_1	T1C & T2L
AC_2	M2C
AC_3	T1H
AC_4	M2F & T1L & T2H
AC_5	T2L
AC_6	M2C
AC_7	T2C & T1H
AC_8	M2F & T1L & T2H



λ = $GV_{01\text{-}36}$con, T1Hcon, T1Lcon, T2Hcon, T2Lcon, T1Ccon, T2Ccon, M1Ccon, M2Ccon, M1Fcon, M2Fcon
μ = openV1-16, closeV1-16

Fig. 19. Auxiliary automaton model for limiting the number of actuator actions (Example 2).

From the corresponding implementable supervisor, it can be found that the minimum number of actuator actions is thirty-two (32) and there are four equally effective SFCs (see Table 9). Notice that there are eight operation steps in these SFCs, while there are only six in the normal operation.

4.6. Scenario 2

Let us next consider the scenario when valve V-11 sticks at the OPEN position, i.e., *V*11*SO*. It can be deduced that this failure is diagnosable at state 18 in Fig. 15 when

1. the beer level in T-1 is low and the cleaning operation is not in progress,
2. the beer level in T-2 is high and the cleaning operation is not in progress,
3. MMS-1 is in service and the cleaning operation is not in progress,
4. MMS-2 is not in service and the cleaning operation is in progress,

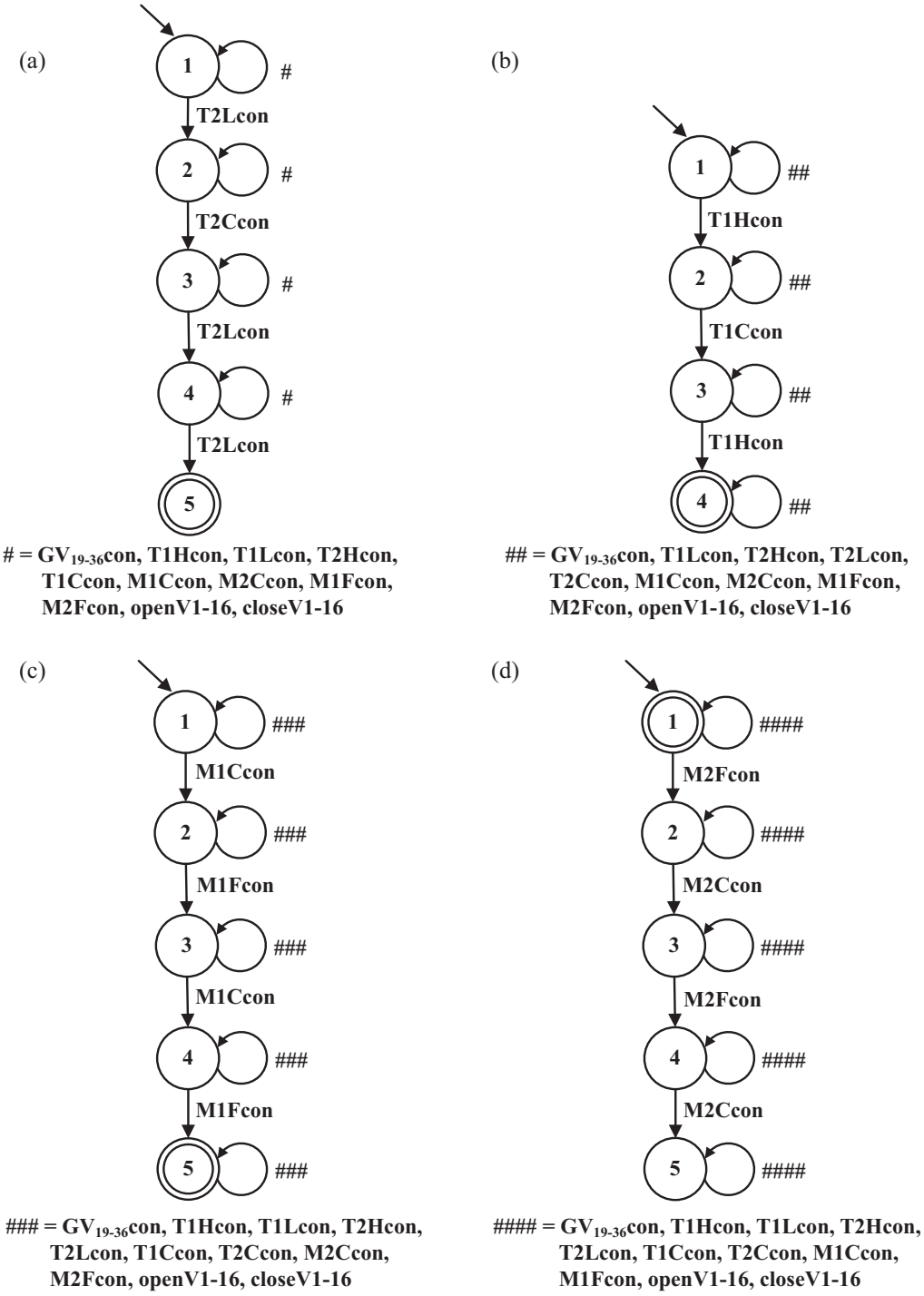


Fig. 20. Auxiliary automaton models (Scenario 2 in Example 2) for: (a) termination mechanism associated with T-2; (b) target state specification associated with T-1; (c) target state specification associated with MMS-1; (d) target state specification associated with MMS-2.

5. V-11, V-4 and V-5 are at the OPEN positions, and
6. all other valves are closed.

By following Procedure II, the legal valve combinations in Table 7(b) and the corresponding state-transition events in Fig. 16(a)–(d) (see Table 8(b)) can be determined for this scenario. An admissible supervisor can then be produced by assembling the aforementioned component models and specifications model. The auxiliary automata shown in Figs. 19 and 20 can be incorporated to limit the total number of actuator actions and to terminate the corresponding emergency response procedure after performing a complete production cycle (via finishing the bottling operations three times). It should be noted that the automaton models in Fig. 20 can be constructed with the same approach adopted for Fig. 18. Finally, after generating the implementable supervisor, it can be found that the minimum number of actuator actions in the optimal emergency response procedures is thirty-two (32) and there are four equally effective SFCs (see Table 10). Notice finally that there are eight operation steps in these SFCs, while six is needed in a normal operation cycle.

Table 10
Emergency SFCs for scenario 2 in Example 2: (a) operation steps; (b) activation conditions.

(a)

Operation step	Control actions (SFC 1)	Control actions (SFC 2)	Control actions (SFC 3)	Control actions (SFC 4)
OS_0	Failure V11SO is diagnosed	Failure V11SO is diagnosed	Failure V11SO is diagnosed	Failure V11SO is diagnosed
OS_1	(1) Close V-4	(1) Close V-4	(1) Close V-4	(1) Close V-4
	(2) Close V-5	(2) Close V-5	(2) Close V-5	(2) Close V-5
	(3) Open V-1	(3) Open V-1	(3) Open V-1	(3) Open V-1
	(4) Open V-6	(4) Open V-6	(4) Open V-6	(4) Open V-6
		(5) Open V-10	(5) Open V-10	
OS_2	(1) Close V-1	(1) Close V-1	(1) Close V-1	(1) Close V-1
	(2) Close V-6	(2) Close V-6	(2) Close V-6	(2) Close V-6
	(3) Open V-2	(3) Open V-2	(3) Open V-2	(3) Open V-2
	(4) Open V-3	(4) Open V-3	(4) Open V-3	(4) Open V-3
	(5) Open V-7	(5) Open V-7	(5) Open V-7	(5) Open V-7
	(6) Open V-10			(6) Open V-10
OS_3	(1) Close V-2	(1) Close V-2	(1) Close V-2	(1) Close V-2
	(2) Close V-7	(2) Close V-7	(2) Close V-7	(2) Close V-7
	(3) Close V-10	(3) Close V-10	(3) Close V-10	(3) Close V-10
	(4) Open V-4	(4) Open V-4	(4) Open V-4	(4) Open V-4
OS_4	(1) Close V-3	(1) Close V-3	(1) Close V-3	(1) Close V-3
	(2) Open V-5	(2) Open V-5	(2) Open V-5	(2) Open V-5
	(3) Open V-8	(3) Open V-8	(3) Open V-8	(3) Open V-8
	(4) Open V-9	(4) Open V-9	(4) Open V-9	(4) Open V-9
OS_5	(1) Close V-4	(1) Close V-4	(1) Close V-4	(1) Close V-4
	(2) Close V-5	(2) Close V-5	(2) Close V-5	(2) Close V-5
	(3) Close V-8	(3) Close V-8	(3) Close V-8	(3) Close V-8
	(4) Close V-9	(4) Close V-9	(4) Open V-1	(4) Open V-1
	(5) Open V-1	(5) Open V-1	(5) Open V-6	(5) Open V-6
	(6) Open V-6	(6) Open V-6		
OS_6	(1) Close V-1	(1) Close V-1	(1) Close V-1	(1) Close V-1
	(2) Close V-6	(2) Close V-6	(2) Close V-6	(2) Close V-6
	(3) Open V-2	(3) Open V-2	(3) Close V-9	(3) Close V-9
	(4) Open V-3	(4) Open V-3	(4) Open V-2	(4) Open V-2
			(5) Open V-3	(5) Open V-3
OS_7	(1) Close V-2	(1) Close V-2	(1) Close V-2	(1) Close V-2
	(2) Open V-4	(2) Open V-4	(2) Open V-4	(2) Open V-4
OS_8	(1) Close V-3	(1) Close V-3	(1) Close V-3	(1) Close V-3
	(2) Open V-5	(2) Open V-5	(2) Open V-5	(2) Open V-5

(b)

Symbol	Conditions (SFCs 1–4)
AC_1	T2L
AC_2	M1C
AC_3	T2C & T1H
AC_4	M1F & T1L & T2H
AC_5	T1C & T2L
AC_6	M1C
AC_7	T1H
AC_8	M1F & T1L & T2H

5. Conclusions and future works

A systematic automata-based procedure is presented in this paper to automatically generate emergency operation steps in a given batch chemical process. Specifically, an admissible emergency supervisor can be synthesized for any given failure-induced system state by combining two distinct sets of automata that model the plant behaviors and the control specifications, respectively. For the purpose of identifying an efficient procedure, a set of auxiliary automata can also be augmented with this supervisor to set the operation target(s) and to limit the total number of actuator actions allowed in the emergency response operation. The feasibility and effectiveness of this proposed method have been verified with two examples in this paper.

Finally, it should be noted that there are still a few unsettled issues for future studies. For example, in order to relieve the heavy work load in building and interpreting the automaton models, additional works are needed to develop generic computer codes for automating the proposed procedure synthesis method. Also, since the current manual verification practice is cumbersome and error-prone, it is more desirable to systematically validate the synthesized procedure in rigorous dynamic simulation studies. Future effort should therefore be devoted to the development of efficient and reliable validation strategies.

Acknowledgement

This work is supported by the National Science Council of Taiwan under Grant NSC 100-2221-E-006-139-MY2.

References

Blanke, M., Kinnaert, M., Lunze, J., & Staroswiecki, M. (2003). *Diagnosis and fault-tolerant control*. Berlin: Springer-Verlag.

Brandin, B. A., & Wonham, W. M. (1994). Supervisory control of timed discrete-event systems. *IEEE Transactions on Automatic Control, 39*, 329–342.

Cassandras, C. G., & Lafortune, S. (1999). *Introduction to discrete event systems*. Boston: Kluwer Academic.

Chen, C. L., & Chen, W. C. (1994). Fuzzy controller-design by using neural-network techniques. *IEEE Transactions on Fuzzy Systems*, *2*, 235–244.
Chou, H. H., & Chang, C. T. (2005). Petri-net-based strategy to synthesize the operating procedures for cleaning pipeline networks. *Industrial & Engineering Chemistry Research*, *44*, 114–123.
Chung, S. L., & Lai, Y. H. (2008). Process control of brewery plants. *Journal of the Chinese Institute of Engineers*, *31*, 127.
Crooks, C. A., & Macchietto, S. A. (1992). Combined milp and logic-based approach to the synthesis of operating procedures for batch plants. *Chemical Engineering Communications*, *114*, 117–144.
Dietrich, P., Malik, R., Wonham, W. M., & Brandin, B. A. (2002). Implementation considerations in supervisory control. In B. Caillaud, P. Darondeau, L. Lavagno, & X. Xie (Eds.), *Synthesis and control of discrete event systems* (pp. 185–201). Kluwer.
Falkman, P., Lennartson, B., & Tittus, M. (2009). Specification of a batch plant using process algebra and petri nets. *Control Engineering Practice*, *17*, 1004–1015.
Fusillo, R. H., & Powers, G. J. (1987). A synthesis method for chemical plant operating procedures. *Computers & Chemical Engineering*, *11*, 369–382.
Galán, S., & Barton, P. I. (1997). Dynamic optimization formulations for operating procedure synthesis. In *Paper Presented at the Annual Meeting of the American Institute of Chemical Engineers*.
Hamid, M. K. A., Sin, G., & Gani, R. (2010). Integration of process design and controller design for chemical processes using model-based methodology. *Computers & Chemical Engineering*, *34*, 683–699.
Hashimme, S., Yajima, T., Kuwashita, Y., & Onogi, K. (2008). Integration of fault analysis and interlock controller synthesis for batch processes. *Chinese Journal of Chemical Engineering*, *16*, 57–61.
Hashizume, S., Yajima, T., Ito, T., & Onogi, K. (2004). Synthesis of operating procedures and procedural controllers for batch processes based on Petri nets. *Journal of the Chinese Institute of Chemical Engineers*, *35*, 363–369.
Hoshi, K., Nagasawa, K., Yamashita, Y., & Suzuki, M. (2002). Automatic generation of operating procedures for batch production plants by using graph representations. *Journal of Chemical Engineering of Japan*, *35*, 377–383.
Ivanov, V. A., Kafarov, V. V., Perov, V. L., & Reznichenko, A. A. (1980). On algorithmization of the start-up of chemical productions. *Engineering Cybernetics*, *18*, 104–110.
Kaspar, M. H., & Ray, W. H. (1992). Chemometric methods for process monitoring and high-performance controller-design. *AIChe Journal*, *38*, 1593–1608.
Kim, J., Kim, J., & Moon, I. (2009). Error-free scheduling for batch processes using symbolic model verifier. *Journal of Loss Prevention in the Process Industries*, *22*, 367–372.
Kim, J., & Moon, I. (2009). Automatic verification of control logics in safety instrumented system design for chemical process industry. *Journal of Loss Prevention in the Process Industries*, *22*, 975–980.
Kinoshita, A., Umeda, T., & O'Shima, E. (1982). An approach for determination of operational procedure of chemical plants. *Proceedings of the International Symposium on Process Systems Engineering*, 114–120.
Koutsoukos, X. D., Antsaklis, P. J., Stiver, J. A., & Lemmon, M. D. (2000). Supervisory control of hybrid systems. *Proceedings of the IEEE*, *88*, 1026–1049.
Lafortune, S., & Teneketzis, D. (2000). UMDES-LIB, Library of commands for discrete event systems modeled by finite state machines. http://www.eecs.umich.edu/umdes/toolboxes.html.
Lai, J. W., Chang, C. T., & Hwang, S. H. (2007). Petri-net based binary integer programs for automatic synthesis of batch operating procedures. *Industrial & Engineering Chemistry Research*, *46*, 2797–2813.
Lakshmanan, R., & Stephanopoulos, G. (1988). Synthesis of operating procedures for complete chemical plants – I. Hierarchical, structured modeling for nonlinear planning. *Computers & Chemical Engineering*, *12*, 985–1002.
Li, H. S., Lu, M. L., & Naka, Y. A. (1997). Two-tier methodology for synthesis of operating procedures. *Computers & Chemical Engineering*, *21S*, S899.
Malik, P., & Malik, R. (2006). Modular control-loop detection. In *Proceedings of the 8th international workshop on discrete event systems* Ann Arbor, Michigan, USA, July 10–12,
Naka, Y., Lu, M. L., & Takiyama, H. (1997). Operational design for start-up of chemical processes. *Computers & Chemical Engineering*, *21*, 997–1007.
Ouedraogo, L., Khoumsi, A., & Nourelfath, M. (2010). A new method for centralised and modular supervisory control of real-time discrete event systems. *International Journal of Control*, *83*, 1–39.
Panjapornpon, C., Soroush, M., & Seider, W. D. (2006). Model-based controller design for unstable, non-minimum-phase, nonlinear processes. *Industrial & Engineering Chemistry Research*, *45*, 2758–2768.
Patton, R. J. (1997). Fault-tolerant control, the 1997 situation. In *Proceedings of safe process* Hull, UK, (pp. 1033–1055).
Ramadge, P. J., & Wonham, W. M. (1987). Supervisory control of a class of discrete event processes. *SIAM Journal on Control and Optimization*, *25*, 206–230.
Ramadge, P. J., & Wonham, W. M. (1989). The control of discrete event systems. *Proceedings of the IEEE*, *77*, 81–98.
Rivas, J. R., & Rudd, D. F. (1974). Synthesis of failure-safe operation. *AIChe Journal*, *20*, 320–325.
Sanchez, A., & Macchietto, S. (1995). Design of procedural controllers for chemical processes. *Computers & Chemical Engineering*, *19*, S381–S386.
Tan, K. S., & Yamashita, Y. (2010). Design of a dependable process control system. In *Proceedings of the 5th international symposium on design, operation and control of chemical processes* (pp. 446–453).
Viswanathan, S., Johnsson, C., Srinivasan, R., Venkatasubramanian, V., & Arzen, K. E. (1998). Automating operating procedure synthesis for batch processes: Part I. Knowledge representation and planning framework. *Computers & Chemical Engineering*, *22*, 1673–1685.
Wang, Y. F., Chou, H. H., & Chang, C. T. (2005). Generation of batch operating procedures for multiple material-transfer tasks with petri nets. *Computers & Chemical Engineering*, *29*, 1822–1836.
Wonham, W. M. (2000). Supervisory control of discrete-event systems: An introduction. In *Proceedings of the IEEE International Conference on Industrial Technology* Goa, India, January 19–22, (pp. 474–479).
Yamalidou, E. C., & Kantor, J. C. (1991). Modeling and optimal control of discrete-event chemical processes using petri nets. *Computers & Chemical Engineering*, *15*, 503–519.
Yamashita, Y. (2007). Toward dependable process control systems: Integration of fault diagnosis and controller redesign. PSE Asia 2007. Xi'an.
Yeh, M. L., & Chang, C. T. An automata-based approach to synthesize untimed operating procedures in batch chemical processes. *The Korean Journal of Chemical Engineering*, in press-a.
Yeh, M. L., & Chang, C. T. An automaton-based approach to evaluate and improve online diagnosis schemes for multi-failure scenarios in batch chemical processes. *Chemical Engineering Research and Design*, in press-b.
Zhang, Y., & Jiang, J. (2003). Bibliographical review on reconfigurable fault-tolerant control systems. In *Proceedings of safe process* Washington, USA, (pp. 265–276).