A Method for Logic Design of ATM Adaptation Layer Protocol*

MIN-CHANG HSU, SHI-CHUNG CHANG†
Department of Electrical Engineering
National Taiwan University
Taipei, Taiwan, ROC

Abstract

In an asynchronous transfer mode (ATM)-based broadband integrated services digital network (BISDN), an ATM adaptation layer (AAL) is needed to adapt each non-ATM application to the ATM layer. This paper proposes an abstract AAL logic design methodology which combines the concept of 'quotient problem' defined by Calvert and Lam, 1987, with the concept of supervisory control in the context of discrete event system theory. A 5-step AAL synthesis algorithm is developed. It is shown that if an AAL does exist, then the algorithm indeed finds a desired AAL FSM. Analysis also indicates that the synthesis algorithm is of polynonmial computation time complexity, which lays a foundation for future extension to realistic applications.

1 Introduction

A BISDN supports various types of applications such as voice, video and data services on a uniform transmission network based on asynchronous transfer mode(ATM) [1]. Each application has its own service requirement and traffic characteristics. An ATM adaptation layer(AAL) is needed between the ATM layer and higher application layers to adapt non-ATM services to the ATM layer and to enhance the service provided by the ATM layer for supporting the higher layer [1]. As there is a large variety of services and an AAL design is needed for each service, it is desirable to have the synthesis of AAL protocols automated.

This paper focuses on the logical aspect of AAL protocol design. In terms of finite state machine (FSM) formalism for protocol design [2], the AAL is a FSM that interconnects between the FSM of ATM layer protocol and the FSM of application layer protocol so that a specific service can be supported. Mathematically, the design of an AAL can be viewed as finding a FSM that is the 'quotient' of the service specification over the composition of the FSM's of ATM and application layer protocols [3]. From supervisory control theoretic point of view [4], AAL is a 'supervisor' that provides and supervises the interacting events with the application layer FSM and ATM FSM so that the service specification is satisfied.

In the literature, protocol synthesis is a less cultivated area than other areas in protocol engineering [5]. Merlin and Bochmann [6] presented a method to synthesize the FSM of a missing system submodule from the given system specification and FSM's of other known submodules. However, their method dealt only with the safety requirement of protocol design. Calvert and Lam conducted a series of research on protocol conversion [3][7][8]. Among their research results, they formulated the design of a protocol convertor as a 'quotient problem' in [3] and proposed a quotient algorithm that explicitly deals with both safety and progress requirements. They pointed out that the algorithm is computationally hard. Qin and Lewis [9] considered the same problem as a facterization problem, where the goal is to construct a submodule X so that the composition of X with all the other given submodules conforms to a given system specification. An algorithm was presented and proved correct to find the most general specification of X but it may create many undesirable states at the initial stages of the algorithm.

The AAL logic design problem can also be abstracted as a facterization problem in which the submodule X, i.e., AAL, lies between the application FSM and ATM FSM and does not directly support the system specification. Exploiting such a problem structure and combining the concept of quotient problem and the concept of supervisory control theory, we develop an AAL logic synthesis methodology which consists of 5-steps. Our methodology synthesizes a FSM of AAL to interconnect between the FSMs of the application and the ATM layers so that the composition of the three does not generate any behaviors beyond those specified by the service specification (i.e., satisfies safety property) but generates all the specified behaviors (i.e., satisfies the progress property).

The 5-step algorithm that realizes the methodolo-

*This work was supported in part by the National Science Council of the Republic of China under Grants NSC82-0416-E-002-248 and NSC83-0416-E-002-016 and by the Telecommunication Laboratory under Contract TL-81-1301.

†The authors would like to thank Prof. Sheng-Luen Chung of National Taiwan Institute of Technology and Prof. Hsu-Chun Yen of National Taiwan University for their very valuable discussions and suggestions.

0-7803-2492-7/95 $4.00 © 1995 IEEE

gy first composes FSM's of the application and ATM layer protocols into a communicating FSM B. A FSM G is then constructed through a modified composition between B and the FSM of service specification P_S, which simulates the behaviors of B confined by P_S. Along with the construction of G, states that must be unsafe and that may be nonprogressive are identified. After deleting all the unsafe states from G, we synthesize a safe FSM of AAL from the remaining FSM of G based on the admissible events of AAL. Progress is finally checked over all the may-be-nonprogressive states; actually non-progressive states are further deleted from the safe FSM of AAL. It is shown that if a nonempty FSM of AAL is obtained, it is a desired AAL logic design. Computational complexity analysis shows that this algorithm is of polynonmial time complexity.

The remainder of the paper is organized as follows. Section 2 presents the abstract quotient problem formulation of AAL design. Section 3 lays the foundation for synthesizing AAL. In Section 4, a safe AAL is first constructed and then its progress property is checked. Section 5 sketches both the correctness proof and computational complexity analysis of the algorithm. Finally, Section 6 concludes the paper.

2 Problem Formulation

A few basic definitions about communicating FSM (CFSM) are given as follows [3].

Definition 1 *A CFSM is a five-tuple $(S, \Sigma, T, \lambda, q_0)$, where S is a nonempty finite set of states, Σ is a finite set of observable events, $T \subseteq S \times \Sigma \times S$ is the observable transition set, $\lambda \subseteq S \times \{\tau\} \times S$ is the unobservable transition set with τ denoting the unobservable event, and $q_0 \in S$ is the initial state.*

Let $a \in \Sigma \cup \{\tau\}$. Denote $s \xrightarrow{a} s'$ if $(s, a, s') \in T \cup \lambda$, $s \xrightarrow{a} \star$ if $s \xrightarrow{a} s'$ for some $s' \in S$ and $s \overset{a}{\not\rightarrow} \star$ if there is no $s' \in S$ such that $s \xrightarrow{a} s'$. Define $\overset{\tau}{\Rightarrow}$ as an indefinite sequence of unobservable transitions, $(\xrightarrow{\tau})^*$, $\overset{\sigma}{\Rightarrow}$ as $\overset{\tau}{\Rightarrow}\xrightarrow{\sigma}$ for $\sigma \in \Sigma$, $\psi(s) \equiv \{\sigma \in \Sigma | s \xrightarrow{\sigma} \star\}$ for $s \in S$ and $\Psi(s) \equiv \{\sigma \in \Sigma | s \overset{\sigma}{\Rightarrow} s' \in S\}$ for $s \in S$.

Definition 2 *Let C and D be two CFSMs. The composition operation, $\|$, operates on two CFSMs into one CFSM $B = C\|D = (S_C \times S_D, \Sigma_B, T_B, \lambda_B, \langle q_{0_C}, q_{0_D}\rangle)$ where*

1. $\Sigma_B = (\Sigma_C \cup \Sigma_D) - (\Sigma_C \cap \Sigma_D)$,
2. $T_B = \{(\langle c, d\rangle, \sigma, \langle c', d'\rangle) : \sigma \in \Sigma_B \wedge ((c = c' \wedge d \xrightarrow{\sigma} d') \vee (d = d' \wedge c \xrightarrow{\sigma} c'))\}$,
3. $\lambda_B = \{(\langle c, d\rangle, \tau, \langle c', d'\rangle) : (c = c' \wedge d \xrightarrow{\tau} d') \vee (d = d' \wedge c \xrightarrow{\tau} c') \vee (\exists \sigma : \sigma \in \Sigma_C \cap \Sigma_D \wedge c \xrightarrow{\sigma} c' \wedge d \xrightarrow{\sigma} d')\}$.

Definition 3 *A CFSM $(S, \Sigma, T, \lambda, q_0)$ is deterministic iff $\lambda = \emptyset$ and when $\sigma \in \Sigma$, s, s', $s'' \in S$, $s \xrightarrow{\sigma} s'$ and $s \xrightarrow{\sigma} s''$, it must have $s' = s''$.*

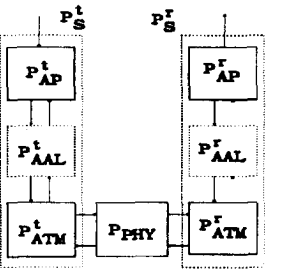

Figure 1: AAL Logic Design Problem

Definition 4 *A trace(or behavior) of a CFSM M consists of a finite sequence of observable events(with τ ignored) generated by M. The language of M, denoted by $L(M)$, is defined as the set of all traces(or behaviors) of M. Let w be a trace of M. Define $|w|$ as the number of events in w. If $|w| = 0$, $w \equiv \epsilon$. Denote $s \overset{w}{\mapsto} s'$ if s' is reachable from s via the occurrence of w.*

Let $A = (S_A, \Sigma_A, T_A, \lambda_A, q_{0_A})$ and $B = (S_B, \Sigma_B, T_B, \lambda_B, q_{0_B})$ be the CFSM representations of a service specification and a protocol system respectively.

Definition 5 *A and B are observation equivalent and denoted by $A \approx B$ iff for each $w \in L(A) \cap L(B)$, $\forall\, q_A \in S_A : q_{0_A} \overset{w}{\mapsto} q_A$, and $\forall\, q_B \in S_B : q_{0_B} \overset{w}{\mapsto} q_B$ imply that $\Psi(q_A) = \Psi(q_B)$.*

Definition 6 *B satisfies A with respect to safety iff $L(B) \subseteq L(A)$.*

Definition 7 *B satisfies A with respect to progress iff for each $w \in L(A) \cap L(B)$, $\forall\, q_A \in S_A : q_{0_A} \overset{w}{\mapsto} q_A$ and $\forall\, q_B \in S_B : q_{0_B} \overset{w}{\mapsto} q_B$ imply that $\Psi(q_A) \subseteq \Psi(q_B)$.*

If A is deterministic, it is straight forward to conclude from Definitions 5, 6 and 7 that B satisfies A with respect to both safety and progress iff $B \approx A$.

In the AAL logic design problem as shown in Figure 1, there are two parts of consideration: the transmitting end and the receiving end. Protocol entities involved include the application protocol(AP) entities P_{AP}^i and P_{AP}^r, AAL protocol entities P_{AAL}^t and P_{AAL}^r and ATM protocol entities P_{ATM}^t and P_{ATM}^r. Let the service specifications be P_S^t and P_S^r. In the remainder of this paper, all these protocol entities and service specifications are modeled as deterministic CFSMs. Given the transmitting(receiving) end CFSMs $P_{AP}^i(P_{AP}^r)$, $P_{ATM}^i(P_{ATM}^r)$ and $P_S^t(P_S^r)$, an AAL CFSM $P_{AAL}^t(P_{AAL}^r)$ needs to be constructed such that when $P_{AP}^i(P_{AP}^r)$, $P_{AAL}^i(P_{AAL}^r)$ and $P_{ATM}^i(P_{ATM}^r)$ are composed together, the composite system satisfies the service specification $P_S^t(P_S^r)$ with respect to both safety and progress. In other words, an AAL logic design problem can be formulated as follows.

AAL Logic Design Problem:

Given CFSMs P_{AP}^i, P_{ATM}^i and P_S^i, find a deterministic CFSM P_{AAL}^i such that $\Sigma_{P_{AAL}^i} \cap \Sigma_{P_S^i} = \emptyset$ and $(P_{AP}^i \| P_{AAL}^i \| P_{ATM}^i) \approx P_S^i$, for $i = t$ and r.

3 Basis of AAL Synthesis

As was defined in Section 2, our logic design problem is to find a P_{AAL} so that $P_{AP}||P_{AAL}||P_{ATM} \approx P_S$. Our approach for synthesizing P_{AAL} first composes P_{AP} and P_{ATM} into a CFSM, say B. The observable events of B can be classified into the set of *external* events(Ext), through which B directly supports P_S, and the set of *internal* events(Int), through which the desired P_{AAL} communicates with B. According to the role of an AAL, the observable events of P_{AAL} should only consist of Int but not Ext of B. If the desired P_{AAL} exists, $B||P_{AAL}$ should result in a CFSM whose observable events consist of Ext only and all events of Int become unobservable after the composition and that $B||P_{AAL} \approx P_S$.

Let the language of B after projection onto Ext be denoted by $L(B)|_{Ext}$. Since there may exist unsafe behaviors in $L(B)|_{Ext}$ with respect to $L(P_S)$, P_S is used to constrain the occurrence sequences of B's external transitions to generate a FSM G with properties $L(G) \subseteq L(B)$ and $L(G)|_{Ext} \subseteq L(B)|_{Ext} \cap L(P_S) \subseteq L(P_S)$. Along with the construction of G, we detect and mark states of G that may result in unsafe behaviors as *must-be-unsafe* and that are likely to result in nonprogressive behaviors as *may-be-nonprogressive*. The must-be-unsafe states of G are then removed to get a FSM R with properties $L(R) \subseteq L(G)$ and $L(R)|_{Ext} \subseteq L(G)|_{Ext} \subseteq L(P_S)$, i.e., safe with respect to P_S. This FSM R serves as the basis for constructing a safe and progressive P_{AAL} in the next Section. The steps for finding R are described as follows.

3.1 Step 1: Compose P_{AP} and P_{ATM} into B

Let us first take the composition $B \equiv P_{AP}||P_{ATM}$ (page 177 in [2]). The resultant CFSM B has no unobservable events since there are no direct interactions between P_{AP} and P_{ATM} and neither P_{AP} nor P_{ATM} has unobservable events. Thus, all events in B are observable events and are classified into internal and external events.

3.2 Step 2: Confine Behaviors of B under P_S

As there may be some unsafe behaviors in $L(B)|_{Ext}$ with respect to P_S, P_S is used to determine the inhibition of some B's external transitions such that B's external behaviors after the restriction are confined to the behaviors of P_S. This goal is achieved by conducting a modified composition of P_S and B to create a FSM G which simulates the behavior of B restricted by P_S.

Let a be a state of P_S and b be a state of B and define a pair $\langle a, b\rangle$ as a state of G. In constructing G, the procedure starts with P_S and B staying at their respective initial states, a_0 and b_0. Let $\langle a_0, b_0\rangle$ be the initial state of G.

Step 2 Algorithm

2.0: Let $S_G = \{\langle a_0, b_0\rangle\}$, $NEW = \{\langle a_0, b_0\rangle\}$. Set $\Xi_G = \emptyset$ and $\vartheta_G = \emptyset$.

2.1: Take a state $\langle a, b\rangle \in NEW$. If the set of admissible external events under b in B, denoted by $\psi_{Ext}(b)$, is not contained in the set $\psi_{Ext}(a)$, then $\langle a, b\rangle$ is marked as must-be-unsafe because an unsafe event may occur from $\langle a, b\rangle$. No new transitions and states are generated from $\langle a, b\rangle$.

2.2: If $\psi_{Ext}(b) \subseteq \psi_{Ext}(a)$, then

2.2.1: If $\psi_{Ext}(b) \subset \psi_{Ext}(a)$, then $\langle a, b\rangle$ is marked as may-be-nonprogressive because a nonprogress condition may occur from $\langle a, b\rangle$. Define $\Theta_G(\langle a, b\rangle) = \psi_{Ext}(a) - \psi_{Ext}(b)$.

2.2.2: For each $\sigma \in \psi_{Ext}(b)$, if $a \xrightarrow{\sigma} a'$ and $b \xrightarrow{\sigma} b'$, then an external transition $\langle a, b\rangle \xrightarrow{\sigma} \langle a', b'\rangle$ is defined and added to the set Ξ_G. If $\langle a', b'\rangle \notin S_G$, $\langle a', b'\rangle$ is added to S_G and NEW.

2.2.3: For each event $\sigma \in \psi_{Int}(b)$, if $b \xrightarrow{\sigma} b'$, then an internal transition $\langle a, b\rangle \xrightarrow{\sigma} \langle a, b'\rangle$ is defined and added to the set ϑ_G. If $\langle a, b'\rangle \notin S_G$, $\langle a, b'\rangle$ is added to S_G and NEW.

2.3: Remove state $\langle a, b\rangle$ from NEW. If $NEW = \emptyset$, then go to Step 3; otherwise, go to Step 2.1.

A formal definition of the modified composition operation is given as follows. For simplicity of notations, P_S is replaced by A.

Definition 8 *The modified composition operation, $\otimes$, operates on two CFSM entities into a FSM $G = A \otimes B = (S_G, \Sigma_G, \Xi_G, \vartheta_G, q_{0_G}, \Lambda_G, \Gamma_G, \Theta_G)$ where*

1. $S_G = S_A \times S_B$,
2. $\Sigma_G = \Sigma_A \cup \Sigma_B = \Sigma_B$ *(since $\Sigma_B = Int \cup Ext$ and $\Sigma_A = Ext$),*
3. $\Xi_G = \{(\langle a, b\rangle, \sigma, \langle a', b'\rangle) : \psi_{Ext}(b) \subseteq \psi_{Ext}(a) \wedge \sigma \in Ext \wedge a \xrightarrow{\sigma} a' \wedge b \xrightarrow{\sigma} b'\}$*; in other words, a transition in Ξ_G simulates that A and B can transit concurrently via the same event $\sigma \in Ext$,*
4. $\vartheta_G = \{(\langle a, b\rangle, \sigma, \langle a, b'\rangle) : \psi_{Ext}(b) \subseteq \psi_{Ext}(a) \wedge \sigma \in Int \wedge b \xrightarrow{\sigma} b'\}$*; in other words, a transition in ϑ_G simulates that A stays at the same state while B makes a state transition via an event $\sigma \in Int$,*
5. *q_{0_G} is the initial state of G and $q_{0_G} = \langle a_0, b_0\rangle$,*
6. $\Lambda_G = \{\langle a, b\rangle : \psi_{Ext}(b) \not\subseteq \psi_{Ext}(a)\}$*, which is the set of all must-be-unsafe states,*
7. $\Gamma_G = \{\langle a, b\rangle : \psi_{Ext}(b) \subset \psi_{Ext}(a)\}$*, which is the set of all may-be-nonprogressive states,*
8. $\Theta_G : \Gamma_G \longmapsto 2^{Ext}$ *with $\Theta_G(\langle a, b\rangle) = \psi_{Ext}(a) - \psi_{Ext}(b)$; hence, for a may-be-nonprogressive state $\langle a, b\rangle$, $\Theta_G(\langle a, b\rangle)$ is a set of external events which are admissible under $a \in S_A$ but not under $b \in S_B$.*

Remarks:

1. Since P_S and B are both finite state machines, the set of $\langle a, b\rangle$ pairs is also finite and the above procedure will terminate in at most $|S_{P_S}| \times |S_B|$ steps.
2. The FSM G thus constructed has properties $L(G) \subseteq L(B)$ and $L(G)|_{Ext} \subseteq L(P_S) \cap L(B)|_{Ext} \subseteq \overline{L}(P_S)$.

3.3 Step 3: Remove Unsafe States

Must-be-unsafe states in Λ_G are undesirable and should be removed from G. If s is a state of G, $s \notin \Lambda_G$ but there exists a sequence of Ext events that brings s to a state $s' \in \Lambda_G$, then the state s and the intermediate states along the transition sequence from s to s' are *essentially unsafe* and should also be deleted. This is because if G stays at either of these states, G may evolve through the occurrence of external transitions, which are not controllable by the design of P_{AAL}, to the must-be-unsafe state s' and then violate the safety condition.

If there are two states s and s' of G with s safe but s' unsafe and an event $\sigma \in Int$ such that $s \xrightarrow{\sigma} s'$, then the transition $s \xrightarrow{\sigma} s'$ must be removed from G because G may go to an unsafe state s' from a safe state s via an internal event σ. Such internal transitions are collected in a set ϑ_{unsafe} and will be used by P_{AAL} design in the next Section.

To identify the essentially unsafe states, a procedure is developed that traces in the backward direction of external transitions going into each state in Λ_G, identifies the intermediate upstream states and add these states into the to-be-removed-state set RM.

Step 3 Algorithm

3.0: Let $S_c = \Lambda_G$ and $RM = \Lambda_G$.

3.1: Take a state $s \in S_c$. If $s' \xrightarrow{\sigma} s$ and $\sigma \in Ext$ and $s' \notin RM$, then add the state s' to RM and S_c.

3.2: Remove state s from S_c. If $S_c = \emptyset$, then go to Step 3.3; otherwise, go to Step 3.1.

3.3: Remove the states in RM and all of their associated transitions from G. If a removed internal transition brings a safe state to an unsafe state, then add this internal transition to ϑ_{unsafe}.

3.4: If the initial state of G is unsafe and hence removed, then report 'P_{AAL} does not exist', set $R = \emptyset$ and STOP; otherwise, go to Step 4.

Remarks:

1. The reduced transition diagram, R, thus obtained has the properties $L(R) \subseteq L(G) \subseteq L(B)$ and $L(R)|_{Ext} \subseteq L(G)|_{Ext} \subseteq L(P_S)$, since all unsafe states and their associated transitions are removed from G.
2. If the initial state of G is not removed, the smallest safe P_{AAL} is a CFSM with a single state and without any transitions. Thus, the necessary and sufficient condition for the existence of a safe P_{AAL} is that $R \neq \emptyset$.

3.4 Example

The following simple example illustrates the key ideas and application of our protocol adaptation algorithm developed so far.

Example

CFSM of P_{AP}:
$S_{P_{AP}} = \{0, 1, 2\}$, $\Sigma_{P_{AP}} = \{O1, \Delta1, \Delta2\}$,
$T_{P_{AP}} = \{(0, O1, 1), (1, \Delta1, 2), (2, \Delta2, 0)\}$,
$\lambda_{P_{AP}} = \emptyset$ and $q_{0_{P_{AP}}} = 0$.

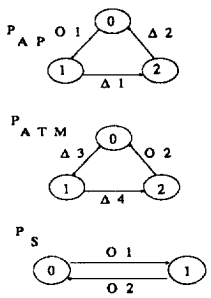

Figure 2: P_{AP}, P_{ATM}, P_S

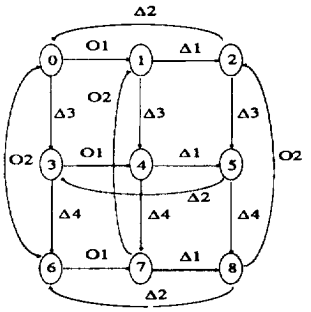

Figure 3: $B = P_{AP} || P_{ATM}$

CFSM of P_{ATM}:
$S_{P_{ATM}} = \{0, 1, 2\}$, $\Sigma_{P_{ATM}} = \{O2, \Delta3, \Delta4\}$,
$T_{P_{ATM}} = \{(0, \Delta3, 1), (1, \Delta4, 2), (2, O2, 0)\}$,
$\lambda_{P_{ATM}} = \emptyset$ and $q_{0_{P_{ATM}}} = 0$.

CFSM of P_S:
$S_{P_S} = \{0, 1\}$, $\Sigma_{P_S} = \{O1, O2\}$,
$T_{P_S} = \{(0, O1, 1), (1, O2, 0)\}$, $\lambda_{P_S} = \emptyset$, $q_{0_{P_S}} = 0$.

Diagrammatic illustrations of CFSMs P_{AP}, P_{ATM} and P_S are given in Figure 2, where a circle represents a state, an internal transition is labeled by $\Delta\#$ and an external transition by $O\#$. The resultant CFSM B by composing P_{AP} and P_{ATM} is shown in Figure 3. Note that $\lambda_B = \emptyset$. The FSM G shown in Figure 4 is $P_S \otimes B$, where heavily-dotted states are must-be-unsafe states and lightly-dotted states are may-be-nonprogressive states. Note that every must-be-unsafe state has no outgoing transitions. An emanating arrow from any lightly-dotted state $\langle a, b \rangle$ is labeled by an external event which is not admissible under b in FSM B but is admissible under a in FSM P_S. $\Theta_G(\langle a, b \rangle) = \{O_2\}$ when $\langle a, b \rangle \in \{\langle 1, 1 \rangle, \langle 1, 2 \rangle, \langle 1, 4 \rangle, \langle 1, 5 \rangle\}$ and $\Theta_G(\langle a, b \rangle) = \{O_1\}$ when $\langle a, b \rangle \in \{\langle 0, 1 \rangle, \langle 0, 2 \rangle, \langle 0, 4 \rangle, \langle 0, 5 \rangle\}$. The FSM R is circled by the dotted line in Figure 4. In this example, unsafe states of G are exactly those marked must-be-unsafe.

$\vartheta_{unsafe} = \{\langle 1, 2 \rangle \xrightarrow{\Delta 2} \langle 1, 0 \rangle,\ \langle 1, 5 \rangle \xrightarrow{\Delta 2} \langle 1, 3 \rangle,\ \langle 1, 8 \rangle \xrightarrow{\Delta 2} \langle 1, 6 \rangle,$
$\langle 0, 3 \rangle \xrightarrow{\Delta 4} \langle 0, 6 \rangle,\ \langle 0, 4 \rangle \xrightarrow{\Delta 4} \langle 0, 7 \rangle,\ \langle 0, 5 \rangle \xrightarrow{\Delta 4} \langle 0, 8 \rangle\}$.

4 Construction of AAL Protocol

Recall that a safe P_{AAL} exists iff $R \neq \emptyset$ and the smallest safe P_{AAL} is a CFSM of a single state without

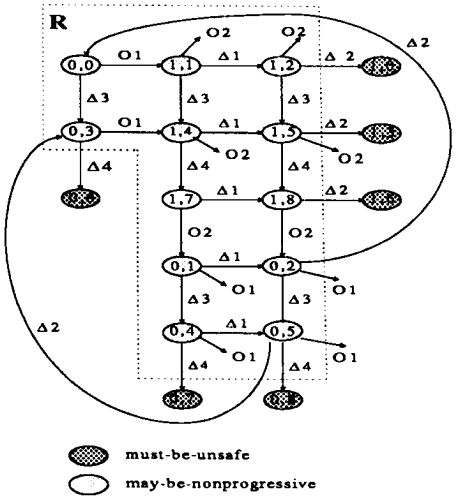

Figure 4: $G = P_S \otimes B$

transitions. In this Section, a method is first developed to construct the largest safe P_{AAL} for the case $R \neq \emptyset$ by starting from the smallest P_{AAL} and gradually adding transitions and states to it. This largest P_{AAL} is then checked to see if it is progressive.

4.1 Step 4: Construct a Safe P_{AAL}

This step constructs the largest safe P_{AAL}, say P^s_{AAL}, from R such that $(B||P^s_{AAL})$ satisfies P_S with respect to safety. Recall that the observable events of B are classified into two classes of Ext and Int events and that an AAL has only access (observation and control) to the Int events in B. The role of P_{AAL} is to properly enable or disable the occurrence of Int events in B based on the observed state information from R so that $L(B||P_{AAL}) \subseteq L(P_S)$. So, from supervisory control theoretic point of view, P_{AAL} is the supervisor of the discrete event system B, to which Int events are both observable and controllable while Ext events are neither observable nor controllable. Due to the limited observability of B to P_{AAL}, P_{AAL} cannot observe all the individual states of B.

As states of R reachable from a state r of R through purely Ext transitions are not distinguishable by P_{AAL}, such a subset of states in R may correspond to a state in P_{AAL}. Motivated by this observation, we first make the following definitions to assist in defining P_{AAL} states.

Definition 9 *Let r and r' be two states of R. Denote $r \; \Xi^*_R \; r'$ if r' is reachable from r via purely external transitions. Define $\pi_{\Xi^*_R}(r) = \{r' : r \; \Xi^*_R r'\} \cup \{r\}$.*

If $\pi_{\Xi^*_R}(r)$ is not a singleton, one state in $\pi_{\Xi^*_R}(r)$ is not distinguishable from another by the supervising P_{AAL}. By Definitions 9, we define $q_{0_{P_{AAL}}} \equiv \pi_{\Xi^*_R}(r_0)$, where $r_0 = \langle a_0, b_0 \rangle$ is the initial state of R. The smallest safe P_{AAL} is a CFSM with a single state $q_{0_{P_{AAL}}}$.

Definition 10 *Let q be a set of states of R and $\sigma \in Int$. Define σ to be admissible under q iff there is not a state r in q such that $(r, \sigma, r') \in \vartheta_{unsafe}$. Denote $\psi_a(q)$ the set of admissible events under q.*

Definition 11 *States of a nonempty P_{AAL} are defined iteratively as follows:*

1. *$q_{0_{P_{AAL}}}$ is a state of P_{AAL}.*
2. *Let q be a state of P_{AAL} and $\sigma \in \psi_a(q)$. Then, $\hat{q} \equiv \bigcup\{\pi_{\Xi^*_R}(\hat{r}) | r \in q, r \xrightarrow{\sigma} \hat{r}\}$ is also a state of P_{AAL} if $\hat{q} \neq \emptyset$.*

To construct P^s_{AAL} with five-tuple $(S_{P_{AAL}}, \Sigma_{P_{AAL}}, T_{P_{AAL}}, \lambda_{P_{AAL}}, q_{0_{P_{AAL}}})$, we begin with $\Sigma_{P_{AAL}} = Int$, $\lambda_{P_{AAL}} = \emptyset$, $q_{0_{P_{AAL}}} = \pi_{\Xi^*_R}(r_0)$, $T_{P_{AAL}} = \emptyset$ and $S_{P_{AAL}} = \{q_{0_{P_{AAL}}}\}$. Based on the above iterative definition, steps for constructing P^s_{AAL} from R are summarized as follows.

Step 4 Algorithm

4.0: Let $q_{0_{P_{AAL}}} = \pi_{\Xi^*_R}(\langle a_0, b_0 \rangle)$. Let $S_{P_{AAL}} = NEW = \{q_{0_{P_{AAL}}}\}$. Set $T_{P_{AAL}} = \lambda_{P_{AAL}} = \emptyset$ and $\Sigma_{P_{AAL}} = Int$.

4.1: Take a state $q \in NEW$. For every pair of states $\langle a, b \rangle$ and $\langle a', b' \rangle \in q$, create a link labeled u from state $\langle a, b \rangle$ to state $\langle a', b' \rangle$ in q if $\langle a, b \rangle \xrightarrow{u} \langle a', b' \rangle$ and $u \in Ext$.

4.2: For each event $\sigma \in Int$ that is admissible under q and $\hat{q} \equiv \bigcup\{\pi_{\Xi^*_R}(\hat{r}) | r \in q, r \xrightarrow{\sigma} \hat{r}\} \neq \emptyset$, do Step 4.2.1 to Step 4.2.3:

4.2.1: A transition $q \xrightarrow{\sigma} \hat{q}$ is defined and added to the set $T_{P_{AAL}}$.

4.2.2: If $\langle a, b \rangle \in q$ and $b \xrightarrow{\sigma} b'$, create a link labeled σ from $\langle a, b \rangle$ in q to $\langle a, b' \rangle$ in $\hat{q}$.

4.2.3: If $\hat{q} \notin S_{P_{AAL}}$, $\hat{q}$ is added to $S_{P_{AAL}}$ and NEW.

4.3: Remove state q from NEW. If $NEW = \emptyset$, then go to Step 5; otherwise, go to Step 4.1.

Remarks:

1. A state q of P^s_{AAL} corresponds to a set of states in R, which can be called 'detailed states' of q.
2. A state in R may be one of the detailed states for more than one P^s_{AAL} states.
3. The introduction of detailed states and the construction of links among them maintain the detailed CFSM structural information about the relation between P^s_{AAL} and R.

4.2 Step 5: Check the Progress Property

By construction in Step 4, P^s_{AAL} satisfies $L(B||P^s_{AAL}) \subseteq L(P_S)$. Moreover, if any other P_{AAL} satisfies $L(B||P_{AAL}) \subseteq L(P_S)$, then $L(B||P_{AAL}) \subseteq L(B||P^s_{AAL})$. This step checks the progressiveness of P^s_{AAL}. Since a state q of P^s_{AAL} corresponds to a subset of states in R, q is also used to represent the subset in the following discussions. A state q of P^s_{AAL} is defined to be nonprogressive if there exists a may-be-nonpropressive detailed state $\langle a, b \rangle \in q$ such that $a \xrightarrow{\sigma} a'$, $b \not\xrightarrow{\sigma} \star$ for some $\sigma \in Ext$, and if there does not exist a sequence of links such that $\langle a, b \rangle \xrightarrow{\sigma_1} \langle a, b_1 \rangle \xrightarrow{\sigma_2}$

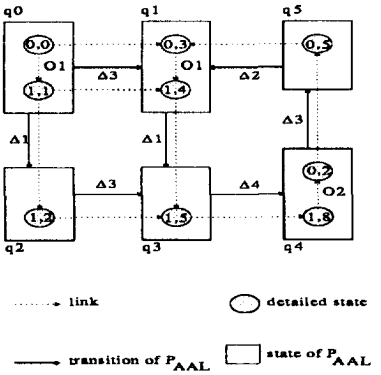

Figure 5: Largest Safe P_{AAL}

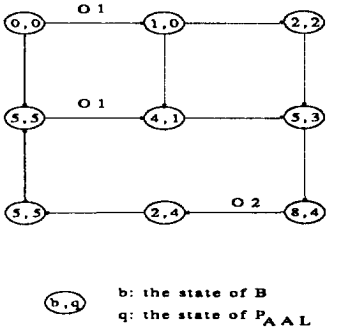

Figure 6: $B||P_{AAL}$

$\cdots \xrightarrow{\sigma_n} \langle a, b_n\rangle \xrightarrow{\sigma} \langle a', b'\rangle$ for some $n \geq 1$, $\sigma_k \in Int$ and $\langle a, b_k\rangle \in q_k$ for $1 \leq k \leq n$. Note that if the aforementioned sequence of links exists, then $(B||P^s_{AAL})$ has a corresponding sequence of unobservable transitions such that $\langle b, q\rangle \xrightarrow{\tau} \langle b_1, q_1\rangle \xrightarrow{\tau} \cdots \xrightarrow{\tau} \langle b_n, q_n\rangle \xrightarrow{\sigma} \langle b', q_n\rangle$ and thus $\sigma \in \Psi(\langle b, q\rangle)$.

The progressiveness of P^s_{AAL} is checked by the following steps.

Step 5 Algorithm

5.1: For every state $q \in S_{P^s_{AAL}}$, check if $q \cap \Gamma_G = \emptyset$. If $q \cap \Gamma_G \neq \emptyset$, then for every state $\langle a, b\rangle \in q \cap \Gamma_G$ and for each event $\sigma \in \psi_{Ext}(a) - \psi_{Ext}(b)$, check if there exists a sequence of links such that $\langle a, b\rangle \xrightarrow{\sigma_1} \langle a, b_1\rangle \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} \langle a, b_n\rangle \xrightarrow{\sigma} \langle a', b'\rangle$ for some $n \geq 1$ and $\sigma_k \in Int$, $1 \leq k \leq n$. As long as there is one pair of $(\langle a, b\rangle, \sigma)$ fails the check, q is marked as nonprogressive. Let NP be the set of nonprogressive states identified by this Step.

5.2: If for $q_j, q_i \in S_{P^s_{AAL}}$, q_i is nonprogressive and $\exists \sigma \in Int$ such that $q_j \xrightarrow{\sigma} q_i$, delete all transitions from q_j that involve event σ and delete all σ-labeled links from detailed states in q_j to detailed states in q_i.

5.3: If NP$\neq \emptyset$, then remove all the nonprogressive states in NP from $S_{P^s_{AAL}}$ and related transitions and links, reset NP$= \emptyset$ and go to Step 5.1; otherwise, go to Step 5.3.

5.4: If $q_{0_{P_{AAL}}}$ is marked nonprogressive, report 'P_{AAL} does not exist'. Otherwise, the remaining CFSM is the desired P_{AAL}.

4.3 Example

We apply Steps 4 and 5 to the previous example.

Example(Continued)

$q_{0_{P_{AAL}}} = q_0 = \pi_{\Xi^*_R}(\langle(0,0)\rangle) = \{\langle 0,0\rangle, \langle 1,1\rangle\}$. Since the admissible events under q_0 are $\Delta 3$ and $\Delta 1$, transitions $q_0 \xrightarrow{\Delta 3} q_1$ and $q_0 \xrightarrow{\Delta 1} q_2$ are added to $T_{P_{AAL}}$ where $q_1 = \{\langle 0,3\rangle, \langle 1,4\rangle\}$, $q_2 = \{\langle 1,2\rangle\}$ are new states and hence added to $S_{P_{AAL}}$. From q_1 and q_2, new states q_3, q_4, q_5 and new transitions $q_1 \xrightarrow{\Delta 1} q_3$, $q_2 \xrightarrow{\Delta 3} q_3$, $q_3 \xrightarrow{\Delta 4} q_4$, $q_4 \xrightarrow{\Delta 3} q_5$, $q_5 \xrightarrow{\Delta 2} q_1$ are then identified and added to $S_{P_{AAL}}$ and $T_{P_{AAL}}$ respectively, where $q_3 = \{\langle 1,5\rangle\}$, $q_4 = \{\langle 1,8\rangle, \langle 0,2\rangle\}$ and $q_5 = \{\langle 0,5\rangle\}$. P^s_{AAL} is depicted in Figure 5. Note the links among detailed states contained in P^s_{AAL} states.

In P^s_{AAL}, all states contain a may-be-nonprogressive detailed state. Step 5 is then applied to q_0–q_5 to check the progress property. For example, q_0 contains a may-be-nonprogressive detailed state $\langle 1,1\rangle$ and $\Theta_G(\langle 1,1\rangle) = \{O_2\}$, but there is a link sequence such that $\langle 1,1\rangle \xrightarrow{\Delta 1} \langle 1,2\rangle \xrightarrow{\Delta 3} \langle 1,5\rangle \xrightarrow{\Delta 4} \langle 1,8\rangle \xrightarrow{O_2} \langle 0,2\rangle$. Thus, q_0 is progressive. In fact, q_1–q_5 are all progressive. Hence, P^s_{AAL} is the desired P_{AAL}. $B||P^s_{AAL}$ is shown in Figure 6, which can be easily verified that $B||P^s_{AAL} \approx P_S$ for this example.

5 Algorithm Properties

Let the AAL design algorithm developed in Sections 3 and 4 be named as the adaptor algorithm. In this section, we first give an outline of its correctness proof and then analyze its computational complexity.

5.1 Correctness

The correctness of the adaptor algorithm is verified by proving the following two Theorems. Interested readers may refer to [10] for the details of proof.

Theorem 1 *If the adaptor algorithm finds a P_{AAL}, then $(B||P_{AAL}) \approx P_S$.*

Sketch of proof:

(i) *$B||P_{AAL}$ satisfies P_S with respect to safety.* This is proved by showing that if $w \in L(B||P_{AAL})$ then $w \in L(P_S)$ via induction on the length of w.

(ii) *$B||P_{AAL}$ satisfies P_S with respect to progress.* Let b_0, q_0 and a_0 be the initial states of B, P_{AAL} and P_S respectively. It can be shown that if $a_0 \overset{t}{\mapsto} a$ in P_S and $\langle b_0, q_0\rangle \overset{t}{\mapsto} \langle b, q\rangle$ in $B||P_{AAL}$, then $\Psi(a) \subseteq \Psi(\langle b, q\rangle)$.

Theorem 2 *If there exists a P'_{AAL} such that $B||P'_{AAL} \approx P_S$, then the adaptor algorithm finds a P_{AAL} such that $B||P_{AAL} \approx P_S$.*

Sketch of proof:

It is intuitively clear that the construction of P_{AAL} is rooted from the initial state $q_0 = \pi_{\Xi_R^*}(\langle a_0, b_0\rangle)$. If the adaptor algorithm finds a P_{AAL}, q_0 must be a state of it. The basic idea of proof is to show that if there exists a P'_{AAL} such that $B \| P'_{AAL} \approx P_S$, neither can $\langle a_0, b_0\rangle$ be identified as unsafe by Step 3 nor can q_0 be marked as nonprogressive by Step 5; in other words, $q_0 \in S_{P_{AAL}}$, the P_{AAL} is not empty and Theorem 1 applies.

5.2 Computational Complexity

The worst case computation time complexity of the adaptor algorithm is analyzed as follows.

Step 1: $P_{AP} \parallel P_{ATM}$

The composition has $|S_{P_{AP}}| \times |S_{P_{ATM}}|$ states. Suppose that the maximum number of the admissible events under a given state of $P_{AP}(P_{ATM})$ is $k_1(k_2)$. The computational time complexity of this step is $O(|S_{P_{SP}}| \cdot |S_{P_{ATM}}| \cdot (k_1 + k_2))$.

Step 2: $P_S \otimes B$

Let the maximal number of admissible internal and external events under a given state of B be k_3 and k_4 respectively, and the maximal number of admissible (external) events under a given state of P_S be k_5. The maximal number of comparison operations for defining admissible events in $P_S \otimes B$ is then $|S_{P_S}| \cdot |S_B| \cdot (k_4 \cdot k_5 + k_3)$

Step 3: Remove Unsafe States

Let k_6 denote the maximal number of states that can reach a given unsafe state via an external transition. Note that the number of unsafe states in G is less than $|S_G|$. So, $k_6 \leq S_G$ and the *while* loop between Step 3.1 and Step 3.2 loops at most $|S_G|$ times. Since each loop is to find the immediately upstream unsafe states of a given unsafe state, the computation time complexity of this step is $O(|S_G|^2)$.

Step 4: Construct a Safe P_{AAL}

The construction of P^s_{AAL} is to group the states that are reachable from given states of R via purely external transitions. The AAL thus constructed is a deterministic CFSM that does not generate any Ext event; each external transition of R can be replaced with an unobservable transition. This step is equivalent to transforming a nondeterministic FSM (R) into a deterministic FSM (P^s_{AAL}), which is known of polynonmial computation time complexity [11][9]. Therefore, this step is of polynonmial time computation complexity.

Step 5: Check the Progress Property

Let $\langle a, b\rangle$ be a may-be-nonprogressive state of R. The states that are reachable from $\langle a, b\rangle$ via purely internal transitions can be identified within $|S_R| - 1$ trace steps. The maximal number of Ext events to be searched under each state is k_4. Thus, the computation time complexity of this step is $O(|S_R| \cdot (|S_R| - 1) \cdot k_4)$.

It is obvious from the above analysis that the AAL logic synthesis algorithm is of polynomial time computation complexity. The most time consuming step is Step 4 because it involves both the grouping of detailed states into states of X and the creation of links among the detailed states.

6 Conclusions

In this paper, a logic synthesis methodology has been developed to assist the automatic generation of AAL protocols. The methodology combines the concepts of quotient problem and the supervisory control theory and addresses both safety and progress requirements. It has been shown that the adaptor algorithm developed based on this methedology indeed finds a solution to the AAL logic design problem if the solution does exists; otherwise, it reports no solution. Analysis has indicated that the algorithm is of polynonmial computation time complexity, which may facilitate its potential for realistic AAL design applications. As the methodology developed so far deals only with the logical aspect of AAL protocol design, incorporation of timing considerations is a topic of future research.

References

[1] R. Handel and M. N. Huber, "Integrated Broadband Networks, an Introduction to ATM-based Networks," *ADDISON-WELSLEY*, pp.92-102, Chap. 5.

[2] G. J. Holzmann, "Design and Validation of Computer Protocols," *Prentice-Hall*, 1991.

[3] K. L. Calvert and S. S. Lam, "Deriving a Protocol Converter: a Top-Down Method," *ACM SIGCOMM'89*, pp.247-258.

[4] P. J. Ramadge and W. M. Wonham, "The Control of Discrete Event Systems," *Proceeding of IEEE*, Vol. 77, pp.81-98, 1989.

[5] M. T. Liu, "Protocol Engineering," *Advances in Computers*, Vol.29, pp.79-168.

[6] P. Merlin and G. V. Bochmann, "On the Construction of Submodule Specification and Communication Protocols," *ACM Trans. on Programming Languages and Systems*, Vol.5, No.1, Jan. 1983, pp.1-25.

[7] K. L. Calvert and S. S. Lam, "An Exercise in Deriving a Protocol Conversion," *ACM SIGCOMM'88*, pp.151-160.

[8] K. L. Calvert and S. S. Lam, "Formal Methods for Protocol Conversion," *IEEE JSAC*, Vol. 8, No. 1, Jan. 1990.

[9] H. Qin and P. Lewis, "Facterization of Finite State Machines under Observational Equivalence," *CONCUR'90*, pp.427-441.

[10] S. C. Chang, "ATM Adaptation Layer Design," Technical report of project NSC 83-0416-E-002-016, National Taiwan University, 1994.

[11] J. E. Hopcroft and J. D. Ullman, "Introduction to Automata Theory, Languages, and Computation," *ADDISON-WESLEY*, Chap. 3.