ATM Admission Models of Stochastic High Level Petri Nets Based on Hierarchical Modeling[1]

Chuang Lin & Samuel T. Chanson

Department of Computer Science
The Hong Kong University of Science and Technology, Hong Kong
Email: chanson@cs.ust.hk

Abstract

This paper presents a framework for modeling and analyzing ATM admission control using Stochastic High Level Petri Net (SHLPN). SHLPN is chosen because it is a powerful graphical and mathematical modeling tool that is able to handle concurrent, asynchronous, nondeterministic and stochastic events. In addition, there exists a set of well-developed performance analysis techniques for SHLPN. This paper uses a hierarchical modeling technique to specify complex ATM network mechanisms using a top-down approach. However, in analyzing the performance of the network, a bottom-up approach is adopted. To tackle the state space explosion problem, a SHLPN model is decomposed into submodels. These subnets are independently evaluated with all possible population which are then substituted by transitions with approximate equivalent performance in the original model. The technique is illustrated by modeling and evaluating a specific connection admission control policy.

1 Introduction

ATM (Asynchronous Transfer Mode) has received increased attention as a communication architecture capable of supporting multimedia applications in high speed. The key to a successful ATM network is the ability to support a wide variety of traffic, services and performance requirements [1]. An effective means to evaluate ATM network performance is to study the various network control models. Such studies are usually based on queuing analysis [2].

We present a modeling technique for ATM admission control based on Petri nets. To the best of our knowledge, this is the first systematic study on Petri-net-based modeling of ATM admission control mechanisms.

The Petri net is a graphical and mathematical model suitable for applications in many areas. It is capable of handling parallelism, concurrence, non-determinism and asynchronism. Since the development of Stochastic Petri Net (SPN)[3] studies in 1981, it has been applied in performance of computer networks and protocols. The advantages of SPNs as a tool for modeling system behaviours include Petri net's descriptive power and the visual-communication interface which is easy to understand. This is especially useful in modeling systems which include features that violate the product-form solution assumptions. However, in describing complex systems such as ATM networks, the SPN models require a large number of places, transitions and arcs so that the advantages of a graphical representation are compromised. In addition, the state space of complex models are so large that solving state equations of the systems becomes prohibitively expensive.

In this paper, we propose a solution based on Stochastic High Level Petri Nets (SHLPNs)[4]. By focusing on high level functions, the size of the graph is reduced. Furthermore, classes of equivalent states can be grouped together to further reduce the state space. Despite this, the potential explosion of the state space is still a significant barrier in modeling ATM networks.

One possible solution to this problem is to use a hierarchical approach based on decomposition and aggregation. It is well known that the basic idea of hierarchical decomposition is to analyse subsystems of a complex system in isolation and then combine the solutions to obtain the solution for the original model. Various related hierarchical approaches for SPN models have been proposed in the literature, each with emphasis on a different aspect like the traffic processes in Generalized Stochastic Petri Nets(GSPN)[5], the marking space in Coloured GSPNs[6] and the iterative delay equivalence in Stochastic Marked Graph Petri Nets [7].

Our proposed hierarchical approach for SHLPN models emphasizes on the syntax and semantic integration of submodels. Four simple interfaces for submodels have been identified and analyzed so that the original model can be easily decomposed into submodels for evaluation. We show how the approach can be used to model various aspects of an admission control mechanism for ATM networks, including bandwidth allocation and connection setup.

The rest of this paper is organized as follows. An informal introduction to SHLPNs, related concepts, and algorithms for performance analysis is given in Section 2. A new, efficient and general top-down modeling and bottom-up decomposing and aggregating approach for the evaluation of system performance is presented in Section 3. In Section 4, we describe the ATM admission control model using the approach presented in Section 3. Finally, Section 5 concludes the paper.

2 Informal Introduction of SHLPNs

In this section, we briefly introduce the main features

1. This research was supported by Hong Kong Telecom Institute of Information Technology and the Chinese Natural Science Foundation. C. Lin is with the Information Science Institute, State Information Centre, Beijing, China.

0-8186-7216-1/95 $04.00 © 1995 IEEE

of SHLPNs[4,9] and the related concepts such as token types, token variables, markings, the reachability graph and the steady state probabilities. The reader is assumed to have some basic knowledge of Petri nets such as that given in [8].

Coloured Petri Nets [10] and Predicate/Transition Nets [11] are sometimes called High Level Petri nets (HLPNs). SHLPNs are HLPNs augmented with the set of average (possible marking dependent) transition rates for the exponentially distributed transition firing times.

The tokens of a HLPN model may be atomic or compound tokens which may have multiple attributes. The element of a compound token can be another token (either atomic or compound). In the SHLPN model, we have the concepts of token type and token variable. The tokens of the same type run in the same subnet and have the same behaviours as well as the same firing rate. Atomic tokens may be represented by a two-ary vector variable. The first attribute is the token type and the second attribute is the token identity. For the compound token vector variable, an element can be an atomic token vector variable. In SHLPN, tokens without attributes are often used for synchronization purposes.

A transition may be associated with a predicate which can be expressed in terms of the attributes of the tokens on the input arc labels of the transition. New attributes may be added or deleted from the list of attributes of a token as it flows through a transition.

A marking of the SHLPN model is a distribution of tokens in each place of the model. A transition represents a class of possible changes of markings. Such a change, also called transition firing, consists of removing tokens from the input places of the transition and adding tokens to the output places of the transition according to the expressions labelled on the arcs. A transition is enabled whenever, given an assignment of individual tokens to the variables which satisfies the predicate associated with the transition, all input places carry enough copies of the proper tokens, and the capacity K of all output places will not be exceeded by adding the respective copies of tokens. The state space of the model consists of the set of all markings connected to the initial marking through the occurrences of transition firing.

In our SHLPN, there are two types of transitions: non-prime transitions and prime transitions. A non-prime transition can further be extended into a subnet. When a non-prime transition is fired, the corresponding subnet is invoked. In the subnet associated with a non-prime transition, there can be other non-prime transitions. A prime transition is the simplest transition which can not include any other transition. A prime transition is graphically represented by a bar. Non-prime transitions representing subnets are represented graphically as rectangles.

The following notations are used in this paper: M(x) is the marking of the place x, i.e., the number of tokens contained in the place x under the marking M; $M_0(x)$ is the initial marking of place x; C(x) is the capacity of the place x; R (x) is the firing rate of the transition x.

In SHLPNs, if H is a set of all enabled transitions under the marking M and $R(t_i)$ is λ_i, any transition in H is possible to fire, but the firing probability for these transitions is different and can be expressed as follows:

$$P(t_i|M) = \lambda_i / \left(\sum_{t_k \in H} \lambda_k \right) \qquad (2.1)$$

The throughput of a transition is useful in performance analyses. If E is a set of markings under which the transition t is fired, P(M) is the steady state probability for the marking (or state) and W(t) is the weight of the input arc for the transition t (i.e., the number of input tokens), then the throughput of the transition t can be expressed by:

$$F(t) = \sum_{M \in E} P(M) \times W(t) \times R(t) \qquad (2.2)$$

Let us assume a SHLPN has a state space $[M_0>$ which has n states, i.e., the corresponding MC has n states. We can construct the transition rate matrix $R = \|r_{ij}\|$, $1 \le i, j \ge n$ for the SHLPN. Let a line vector $X = (x_1, x_2, ..., x_n)$ represents the steady state probabilities for the n states in the Markov Chain. The following linear equations represent the solution of the steady state probabilities:

$$\begin{aligned} XR &= 0 \\ \sum_i x_i &= 1 \qquad (1 \le i \le n) \end{aligned} \qquad (2.3)$$

We define two classes of tokens and two classes of transitions in our ATM network model:

Tokens:

Class 1 tokens are used to represent the environment variables (e.g., counters) and the synchronization operations among concurrent activities.

Class 2 tokens are used to represent units of information flow such as data and control cells. The behaviours of this class of tokens are of prime interest in performance analyses.

Transitions:

Class 1 transitions are used to represent logical relation or determine if some conditions are satisfied. In the GSPNs, this class of transitions are called immediate transitions with zero firing time. In this paper, these transitions are prime transitions and are associated with higher firing rates than the class 2 transitions.

Class 2 transitions are used to represent the jobs's operations or the information process. They can be non-prime transitions which can further be extended into subnets.

There are two types of arcs in SHLPNs: the normal arc and the inhibitor arc. Only arcs from a place to a transition can be inhibitor arcs. The weight of an arc is labelled on the arc. When the label is absent, it is assumed to be '1'. When the input place associated with an inhibitor arc contains the proper tokens, the transition is disabled.

3 Model Decomposition and Aggregation

A new approach is presented in this section for state space reduction of SHLPNs that are decomposed into independent submodels. These subnets are aggregated and substituted by non-prime transitions to achieve net structure reduction. The substitution is based on the estimated token's residence time in the original submodel

as equivalence criterion for the firing time of the substituted transition. We shall discuss the interfaces of subnets that satisfy the decomposition requirements and estimate the token delay time through the subnet.

In the approximate equivalent substitution, a requirement for the subnets is that their states and transition rates are not affected by the other parts of the net.

We present four kinds of interfaces for subnets that can be easily decomposed from the original model and aggregated into transitions. These interfaces have the following forms: (1) the input interface and the output interface each consists of single transition; (2) the input interface and the output interface each consists of a single place; (3) the input interface is a transition, and the output interface is a place;(4) the input interface is a place, and the output interface is a transition. These four cases are described in detail below;

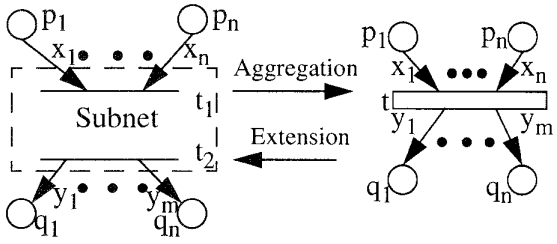

Figure 3.1 Substitution between a subnet and a non-prime transition in case (1)

(1) Consider the interface of a subnet in Figure 3.1. There is an input transition t_1 and an output transition t_2, and each transition can connect to multiple places outside of the subnet. In this case, a non-prime transition t can be used instead of the subnet which includes transitions t_1 and t_2. In this substitution procedure, the colours of places at the interfaces and the token variables in the arc labels are not changed, but the coefficients of the tokens in the arc labels have to be kept consistent according to the number of tokens flowing from transition t_1 to transition t_2 (an example is given in Figure 3.1 (a)). The predicate (or colour) of the non-prime transition is obtained from the structure of the transitions in the subnet in a straight forward manner. For example, in Figure 3.1 (b) the predicate of the non-prime transition is $P_1 \wedge (P_2 \vee P_3) \wedge P_4$.

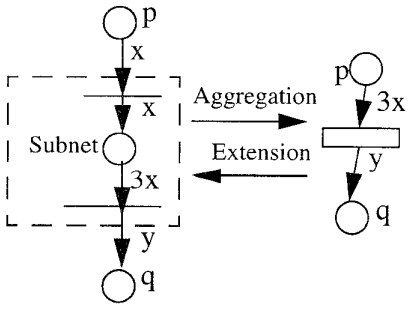

Figure 3.1 (a) An example of the consistent coefficients

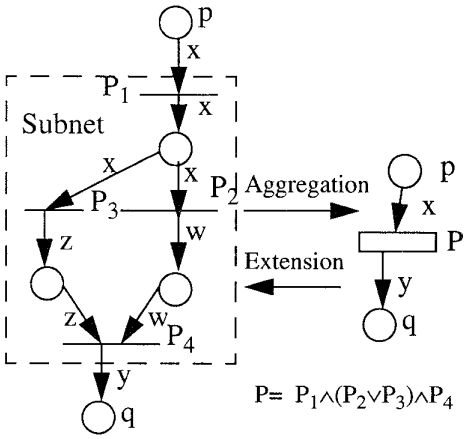

Figure 3.1 (b) An example of the logical expression for the predicate

(2) In Figure 3.2, the interface of a subnet has an input place p and an output place q, both of which can have a number of connections with the subnet. The arc label from p to t represents the join of all arc labels from p to the subnet, and the same representation is used for the arc label from t to q.

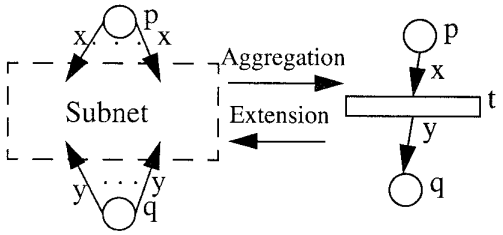

Figure 3.2 A substitution for case (2)

If the places p and q include multiple types of tokens which have different behaviours and performance in the subnet, we can substitute multiple non-prime transitions for the subnet so that different types of tokens go through different non-prime transitions (Figure 3.3). The requirement for the arc labels and the non-prime transition's colour is the same as that for case (1).

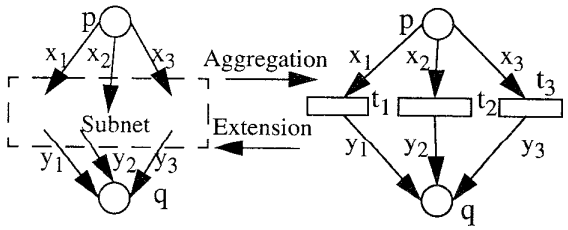

Figure 3.3 Another substitution for case (2)

(3) When the subnet has an input transition and an output place as its interface, we can substitute a non-prime transition for this subnet as shown in Figure 3.4.

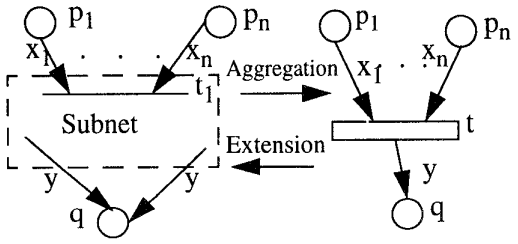

Figure 3.4 A substitution for case (3)

(4) In Figure 3.5, when the subnet has an input place and an output transition as its interface, we can substitute a non-prime transition for this subnet as follow.

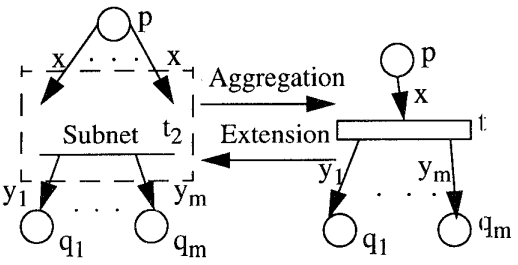

Figure 3.5 A substitution for case (4)

When the inputs and outputs of a subnet consist of a combination of the logical relations mentioned above, we can aggregate the subnet into a non-prime transition according to the rules of the four cases.

In order to estimate the token traffic process in a subnet, sometimes we need to add a transition in the subnet to construct a subsystem which contains the subnet so that the subsystem is a bounded, live and conservative net, and its state space is irreducible. The subsystems for the four cases mentioned above are shown in Figure 3.6. The subsystems are used to estimate the average delay time of a token through the subnet. A proper number of tokens are put in the places $q_1,\ldots, q_m$ and q so that the steady state probabilities of the states (with all possible token number in the initial marking) can be computed for the subsystems.

We can deduce the steady state probability of there being i tokens in the input places of the transition t_1, denoted as r_i, and the average token flow rate through the transition t_1, denoted as f_1. They have the relationship:

$$f_1 = \lambda_1 \sum_{i=1}^{n} i \times r_i \qquad (3.1)$$

where n is the maximum number of tokens in the input place of the transition t_1, and the transition rate $\lambda_1$associated with t_1 is supposed to be dependent on the input place marking.

The average number of tokens in the subsystem, denoted by N, can be deduced from the steady state probabilities. Since the subsystem conserves tokens, Little's Law can be applied to obtain the token's delay time through the subsystem:

$$T' = \frac{N}{f_1} \qquad (3.2)$$

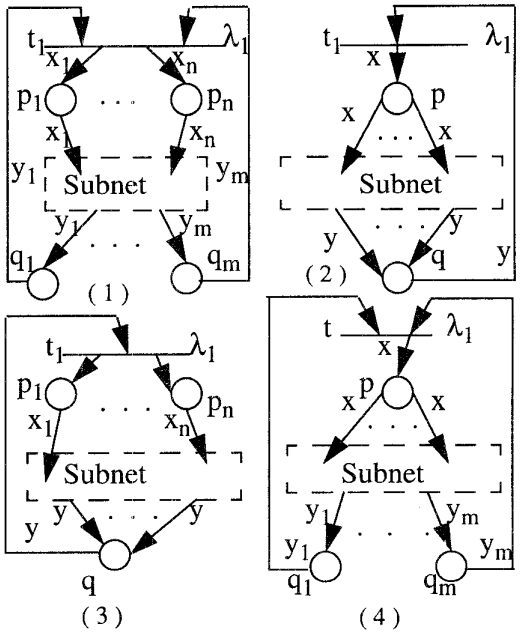

Figure 3.6 The estimation models

The subnet's delay time for a token traffic can be obtained from T':

$$T = T' - \frac{1}{\lambda_1} \qquad (3.3)$$

The approximate equivalent rate for the non-prime transition t can be represented as follow:

$$\lambda = \frac{1}{T} \qquad (3.4)$$

4 A SHLPN Performance Model for ATM Admission Control

In this section, we will apply the modeling and performance evaluation approaches mentioned in Sections 2 and 3 to study the ATM admission control model given in [12].

The bandwidth management procedures have been discussed by Gun and Guerin[12]. Figure 4.1 is a SHLPN model for the admission control mechanism. Familiarity of the mechanism described in [12] will help but is not essential in understanding the model. The SHLPN model is easy to understand given the meaning of the transitions and the places. We use a non-prime transition to represent a procedure. Four procedures are modelled, these are: bandwidth allocation, path selection, maintaining the local topology database and connection setup. In Figure 4.1, each transition is associated with a firing rate to represent the requirement of the process. However, the transitions t_1, t_2, t_3 and t_4 are logical transitions which do not require processing and so they can be considered to have high transition rates.

The transitions shown in Figure 4.1 are described below:

H *is the "user request" transition. When it fires, a user request vector is created. The transition rate is associated with the mean request arrival rate.*

B *is the "bandwidth allocation" transition. When it fires, the bandwidth allocation procedure is executed. The transition can be a non-prime transition and can be extended into a subnet which is used to describe the bandwidth allocation procedure in more detail.*

F *is the transition which models the maintenance of the local topology database. When it fires, the local topology database is updated. The transition can be a non-prime transition.*

P *is the "path selection" transition. When it fires, the path selection procedure is executed. The transition can be a non-prime transition.*

t_1 *is the "unacceptable path length" transition. When the path length D exceeds the maximum length threshold P_T, i.e., $D>P_T$, the transition is enabled. When it fires, the user request is rejected.*

t_2 *is the "acceptable path length" transition. When $D \leq P_T$, the transition is enabled. When it fires, the user request enters the setup procedure.*

C *is the "setup" transition. When it fires, the setup procedure is executed. The transition is a non-prime transition.*

t_3 *is the "negation" transition. When any of the intermediate nodes cannot grant the requested bandwidth, i.e., a denied confirmation message is received by the source, the transition is enabled. When it fires, the user request is rejected.*

t_4 *is the "acknowledge" transition. when each of the intermediate nodes confirms the user's request, the transition is enabled, i.e., y=a.*

The places in the model in Figure 4.1 and the tokens they contain are:

R *is the place which models the user request queue. It contains token vectors r(i, o, d, R, m, b) which represent user requests. The six attributes of the token vector are:*
i: request identity; o: original node;
d: destination node; R: peak bit rate;
m: mean bit rate; b: duration of a burst.

G *is the place which models Quality of Service and contains a token g that represents the quality of service with three attributes:*
e: loss probability; w: the end-to-end delay;
q: the network throughput.

S *is the "bandwidth allocated" place containing tokens r^* (g, r, σ^2,c^*) which represent the requested bandwidth. The token vector has four attributes:*
r: user request vector; $\sigma^2 = m(R-m)$;
c^*: equivalent capacity; g: the desired GOS.

T *is the "local topology database" place containing a token l (l_1, l_2,..., l_n) (or a revised replica l^*) with n link state vectors l_i (o_i, d_i, Σm_j, $\Sigma\sigma^2$, Σc_j^*). Each link state vector has two attributes:*
o_i: the source node of the link; d_i:the link's destination node.

The significances of Σm, $\Sigma\sigma^2$, and Σc^* will be described later in conjunction with the token parameter relations.

L *is the place which models path admission. It contains the token $p(r^*, L, D)$ which represents the selection path. The token vector has three attributes:*
r^*: the requested bandwidth; D: the path length;
L: l_1, l_2,...,l_k the path consisting of the link sequence.

U *is the place that models "entry setup". It contains the token $b(L, r^*)$ representing the setup packet. The token vector has two attributes:*
L: the link sequence; r^*: the requested bandwidth.

E *is the place that models "setup admission". It contains the token x(b, j, y). The attribute y is the confirmation message. When y=n, the setup is denied; when y=a, the setup is successful; j is the identity of the node returning the confirmation message (see transitions t_6 and t_7 in Figure 4.2).*

N *is the place that models the "rejected request queue" which contains the token r.*

A *is the place that models the "successful setup queue". It contains the token S(L, r).*

The non-prime transitions shown in Figure 4.1 can further be extended into subnets to describe their structures and behaviours in more detail. We will only extend the connection setup procedure (transition C) into a subnet as an example; and give token parameter relations or operations (transition predicates) for the bandwidth allocation procedure, the path selection procedure, and the connection setup procedure.

Token parameter relations or operations (transition predicates):

a. Bandwidth allocation procedure (transition B)

(1) Compute the equivalent capacity:

$$c^* = R\frac{y - x + \sqrt{(y-x)^2 - 4x\rho y}}{2y}$$

where x is the amount of buffer space,
$y=\ln(1/e)b(1-\rho)R$, and $\rho = m/R$.

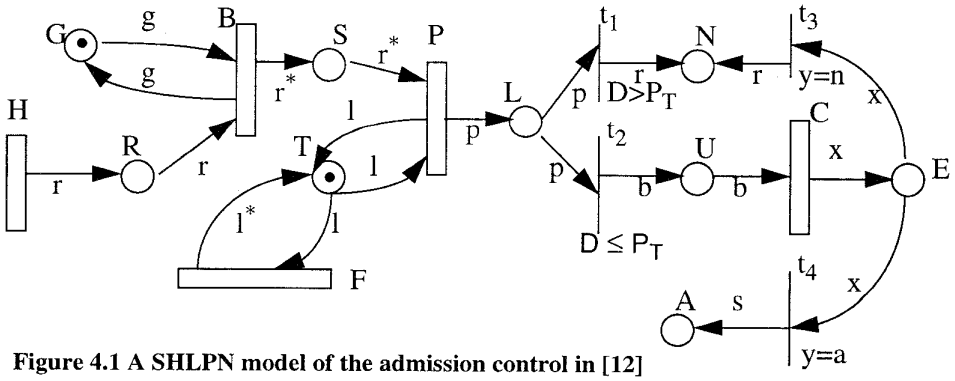

Figure 4.1 A SHLPN model of the admission control in [12]

(2) Compute the variance of the stationary bit rate:
$\sigma^2 = m (R - m)$.

b. Path selection procedure (transition P)
(1) Compute the load for each link in a path:

$$CP = \frac{C_i}{(C_i - C_1)(C_i - C_2)}$$

where C_i is the transmission capacity of the link i, the current bandwidth $C_1 = l_i (\Sigma c^*)$,and the requested bandwidth $C_2 = C_1 + r^*(c^*)$.

(2) Find the shortest path:
min $(D = \Sigma CP \mid r(d) = d_J$ and $r(o) = o_J)$
where o_j is the source node address of the path and d_j is the destination node address.

c. Connection setup procedure (transition C)
The transition C is extended into a subnet shown in Figure 4.2. Some transition predicates and token parameter operations are given below:
(1) Transition t_6
The transition predicate for the enabling condition is associated with $C_i < l_i (\Sigma c^*) + r^*(c^*)$ for link i along the path. The main operation is: x(y)=n.
(2) Transition t_7
The predicate is: $C_i \geq l_i (\Sigma c^*) + r^*(c^*)$.

The main operation is: $l_i (\Sigma c^*) \leftarrow l_i (\Sigma c^*) + r^*(c^*)$ and x(y)=a.
(3) Transition t_4
The predicate is: $k_1 = k$ (k nodes grant the connection request).
(4) Transition t_8
The predicate is: $(k_1 < k) \wedge (k_1 + k_2 = k)$ (k_2 nodes deny the connection request).
(5) Transition t_9
The operation is: $l_i (\Sigma c^*) \leftarrow l_i (\Sigma c^*) - r^*(c^*)$ (k_2 nodes restore their previous load values).

The transitions shown in Figure 4.2 are described in the following: (t_3 and t_4 have been presented in Figure 4.1)

t_5 is the "starting setup" transition. When it fires, k setup tokens are sent to k nodes over the path.
t_6 is the "node negation" transition. When it fires, a node along the path denies the requested bandwidth and y is set to n.
t_7 is the "node acknowledge" transition. When it fires, a node along the path confirms the resource requirement and y is set to a.
t_8 is the " path negation" transition. When it fires, all acknowledgement messages are sent to the restore procedure and a negation message is placed in p_4.
t_4 is the "acknowledge" transition. When it fires, an acknowledgement is sent to the place A and a virtual circuit is established.
t_9 is the "restore" transition. When it fires, the nodes along the path restore the link state vectors to their previous values.

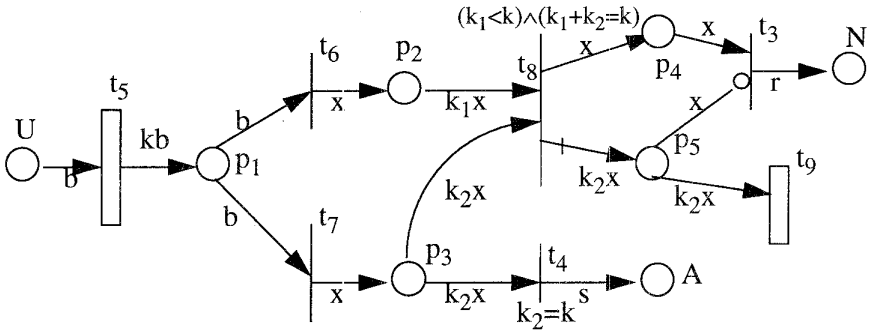

Figure 4.2 A SHLPN model of the connection setup procedure

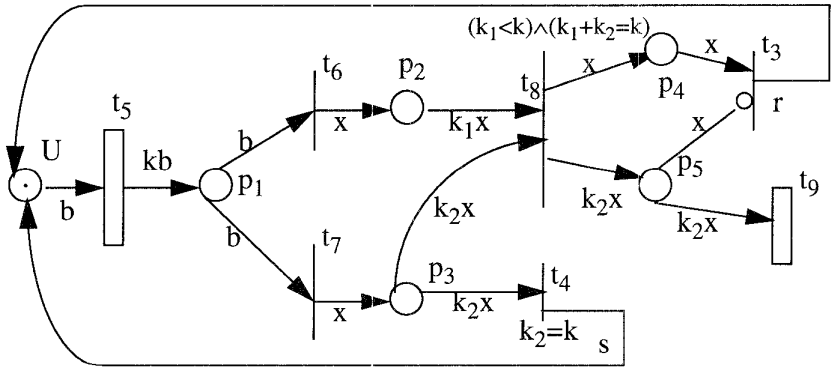

Figure 4.3 The bounded and conservative net model of the connection setup procedure

t_3 *is the "negation" transition. When the restore procedure is finished, the transition can be fired.*

The places shown in Figure 4.2 have the following significances:

p_1 *is the "intermediate node" place.*
p_2 *is the "negative" place.*
p_3 *is the "acknowledge" place.*
p_4 *is the "negative answer" place.*
p_5 *is the "restore" place.*

If we would like to observe the transfer procedure of the setup tokens and the restore procedure, transitions t_5 and t_9 can be further extended into subnets. The top-down modeling approach can be used to describe the system to any level of details.

Next, we shall discuss the performance evaluation of the models described above. According to the model decomposition approach described in Section 3, the subnet shown in Figure 4.2 can be constructed as a subnet for the model in Figure 4.1 and can be evaluated independently. After obtaining the delay time of a token through the subnet, we can use the transition C in Figure 4.1 as a substitute for the subnet.

In order to obtain the subnet's delay time, the model shown in Figure 4.2 has to be modified so that it becomes a bounded, live and conservative model (called a subsystem) shown in Figure 4.3.

For k=3,we can obtain the state transition table (Table 1) from the model in Figure 4.3 according to the algorithm for reachability trees in [9].

For illustration, we use λ_i to represent the rate of transition for t_i and set the following values:
$\lambda_5 = \lambda$, $\lambda_4 = \lambda_3 = \lambda_8 = \lambda_6 = 100\lambda$, $\lambda_9 = k\lambda$ / $M(p_5) = 3\lambda$ / $M(p_5)$ and $\lambda_7/\lambda_6 = n$ or $\lambda_7 = n\lambda_6 = 100n\lambda$.

From the Markov Chain (the state transition table), according to the equation (2.3) we get the steady state probability of state 1 as follow:

$$q_1 = \frac{100(n+1)^3}{101n^3 + 506n^2 + 409n + 105} \cong \frac{(n+1)^3}{(n+1)^3 + 2n^2 + n}$$

According to the equation (2.2), the throughput of transition t_5 is $f = \lambda \times q_1$, and applying Little's law, the delay time of the subnet is:

$$T = \frac{1}{f} = \frac{(n+1)^3 + 2n^2 + n}{(n+1)^3}\lambda$$

When we set n=1, 2 and 1/2, the throughput f is $8/11\lambda$, $27/37\lambda$ and $3.375/4.375\lambda$ respectively, which are the rates for the transition C.

Next, we compute the response time of a user request, i.e., a token's delay time through the net shown in Figure 4.1. The model shown in Figure 4.1 may be modified into a bounded, live and conservative model shown in Figure 4.4. The places G, T, E, A and N as well as the transitions F, t_3 and t_4 are removed since we are only interested in the user request token traffic in the net. We add the place I for the user request generator in the model.

Table 1 The state transition table for the model in Figure 4.3

State	Previous State and Transition	Place Index						Post State and Transition
		U	p_2	p_2	p_3	p_4	p_5	
1	<5,t_4> <9,t_3>	b	0	0	0	0	0	<2,t_5>
2	<1,t_5>	0	3b	0	0	0	0	<3,t_7> <6,t_6>
3	<2,t_7>	0	2b	0	x	0	0	<4,t_7> <10,t_6>
4	<3,t_7>	0	b	0	2x	0	0	<5,t_7> <11,t_6>
5	<4,t_7>	0	0	0	3x	0	0	<1,t_4>
6	<2,t_6>	0	2b	x	0	0	0	<7,t_6> <10,t_7>
7	<6,t_6>	0	b	2x	0	0	0	<8,t_6> <13,t_7>
8	<7,t_6>	0	0	3x	0	0	0	<9,t_8>
9	<8,t_8>	0	0	0	0	x	0	<1,t_3>
10	<3,t_6> <6,t_7>	0	b	x	x	0	0	<11,t_7><13,t_6>
11	<4,t_6> <10,t_7>	0	0	x	2x	0	0	<12,t_8>
12	<11,t_8>	0	0	0	0	x	2x	<9,t_9>
13	<7,t_7> <10,t_6>	0	0	2x	x	0	0	<14,t_8>
14	<13,t_8>	0	0	0	0	x	x	<9,t_9>

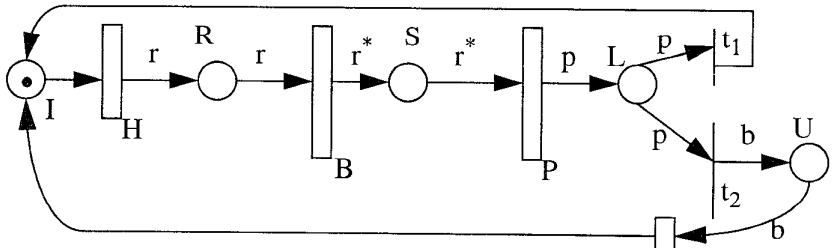

Figure 4.4 The simple net model for the admission control

Furthermore, we can aggregate the subnet which includes the transitions H, B, P and the places R and S by an equivalent non-prime transition HBP as a substitute for the subnet. The aggregated model is shown in Figure 4.5. The firing time for the transition HBP is equal to the sum of the firing times for the transitions H, B and P.

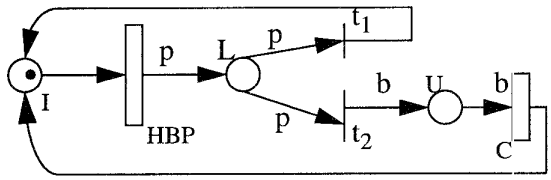

Figure 4.5 The aggregation model for the admission control

From Figure 4.5 we can obtain the state table and the Markov chain for the admission control model shown in Figure 4.6. In Figure 4.6, the transition rate f_0 is for the transition HBP, f_1 for t_1, f_2 for t_2 and f for C. The value of f can be obtained from the model in Figure 4.3.

State	place I	L	index H
1	1	0	0
2	0	p	0
3	0	0	b

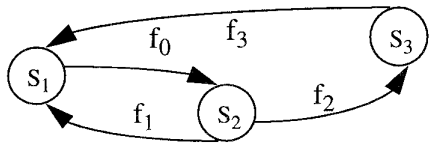

Figure 4.6 The state table and Markov Chain for the model in Figure 4.5

As an example, let $f_1 = 100\, f_0$ and $f_2 / f_1 = m$ or $f_2 = 100\, m\, f_0$. The steady state probability of state 1 is obtained according to the equation (2.3) as follow:

$$p_1 = \frac{100(m+1)f}{(100m+101)f+100mf_0} \cong \frac{(m+1)f}{(m+1)f+mf_0}$$

According to the equation (2.2) the throughput of the transition HBP is $f_3 = f_0 \times p_1$; and from Little's law, the response time of a user request is given by:

$$T = \frac{1}{f_3} \cong \frac{(m+1)f+mf_0}{(m+1)f\times f_0}$$

5 Conclusion

We have presented a top-down modeling and bottom-up decomposition and aggregation approach to study four kinds of subnet interfaces in Petri nets. The proposed framework is generally applicable to modeling and performance studies of large systems. Specifically, we have shown how the ATM admission control mechanism described in [12] can be modelled by SHLPNs, and how the performance models can be evaluated. This new approach in modeling ATM networks provides new insights and treats problems in ATM performance in a different and efficient way. An important contribution is the handling of the state explosion problem when modeling large systems in SHLPN for performance study. Our approach for SPN models emphasizes on the syntax and semantic integration of submodels. It is suitable for most classes of SPNs and the net interfaces can be easily handled. Our approach is considered to be more efficient than the existing approaches [5-7].

The SHLPN model can be automatically translated into computer programs. The translation method has been studied in [13] and has practical value in system implementation.

References

[1] J.J. Bae and T. Suda, "Survey of Traffic Control Schemes and Protocols in ATM Networks", *Proceedings of the IEEE*, Vol.7, No.2, Feb.1991, PP. 170-189.

[2] The proceedings of the Thirteenth Annual Joint Conference of the IEEE Computer and Communications Societies, *IEEE INFOCOM'94*, Toronto, Ontario, Canada, June 14-16, 1994.

[3] M.K. Molloy, "Performance Analysis Using Stochastic Petri Nets", *IEEE Trans. on Computers*, Vol.C-31, No.9, Sep. 1982, PP.913-917.

[4] C. Lin and D.C. Marinescu," Stochastic High-level Petri Nets and Applications", *IEEE Trans. on Computers*, Vol. 37, No.7, July 1988, PP.815-825, Also in K. Jensen and G. Rozenberg (eds.), High-level Petri Nets, Springer-Verlag, 1991.

[5] G. Klas, "Hierarchical Solution of Generalized Stochastic Petri Nets by Means of Traffic Processes", *Lecture Notes in Computer Science* 616, Application and Theory of Petri Nets 1992, PP. 279-298.

[6] P. Buchholz, "Hierarchies in Coloured GSPNs," *Lecture Notes in Computer Science* 691, Application and Theory of Petri Nets 1993, PP. 106-125.

[7] Y. Li and C. M. Woodside, "Iterative Decomposition and Aggregation of Stochastic Marked Graph Petri Nets", *Lecture Notes in Computer Science* 674, Advances in Petri Nets 1993, PP. 325-349.

[8] T. Murata, "Petri Nets: Properties, Analysis and Applications", *Proceedings of the IEEE*, Vol.77, No.4, April 1989, PP. 541-580.

[9] C. Lin and D.C. Marinescu, "On the Analysis of Stochastic High-level Petri Net Models", *Microelectron. Reliab.*, Vol.31, No.4, 1991, PP.747-767.

[10] K. Jensen, "Coloured Petri Nets", *Lecture Notes in Computer Science*.254, 1986, PP.248-299.

[11] H.J. Genrich, "Predicate/Transition Nets", *Lecture Notes in Computer Science* 254, 1986, PP.207-247.

[12] L.Gun and R. Guerin, "Bandwidth Management and Congestion Control Framework of the Broadband Network Architecture", *Computer Networks and ISDN systems* 26 (1993) PP.61-78.

[13] C. Lin and D. C. Marinescu, "Translation of Modified Predicate Transition Nets Models of Communication Protocols into Simulation Programs", Proceeding of the 1986 Winter Simulation Conference, Dec. 1986, USA, PP. 760-768.