Minimum Cycle Time Analysis of IPTV Systems

Jaegeol Yim and Gyeyoung Lee
Dept. of Computer Engineering, Dongguk University at GyeoungJu
{yim, lky}@dongguk.ac.kr

Abstract

IPTV stands for Internet Protocol Television, and any user with an IP device such as a smart phone can get IPTV service anywhere and anytime as long as the user can access the Internet. Because of this advantage, IPTV is widely believed to be the next killer internet application, and many organizations provide IPTV services. As IPTV becomes popular, formal proofs of IPTV service quality have been attempted. One of them was the minimum cycle time analysis. The minimum cycle time analysis takes a long time when the given system is large. This paper proposes a divide-and-conquer method of minimum cycle time analysis that is much faster than the existing ones.

Keywords: *Petri net, IPTV, divide-and-conquer, minimum cycle time, live*

1. Introduction

IPTV stands for Internet Protocol television, and any user with an IP device such as a smart phone can get IPTV service anywhere and anytime as long as the user can access the Internet [1]. IPTV services may be classified into three main groups: live television, time-shifted programming, and video on demand [2]. Since the Internet is available virtually everywhere and IPTV services are so valuable to people, IPTV is widely believed to be the next killer internet application.

Various useful types of interactive services such as VOD (Video on Demand), education, entertainment, etc. are available on IPTV, and so many IPTV systems have recently been implemented. One of the key success factors of IPTV services is the quality of service, and one of the main aspects of quality is timeliness. Therefore, it is extremely important to guarantee that IPTV systems deliver contents in time.

It is also well known that the maintenance cost is much bigger than the installation cost, and verifying that the design of a system meets the user requirements early on of the design stage is very important. However, in practice, installing an IPTV system without verification and starting service in a hurry is common. For formal proofs of system performance, the Petri net minimum cycle time has long been used [3].

Cycle time is the measure of a business cycle from beginning to end. The minimum cycle time is equal to the longest task time in the series of tasks from beginning to end. For example, consider a service system that requires three sequential tasks to deliver a service. Task one takes 5 minutes; task two, 10 minutes; and task three, 7. The minimum cycle time for this service delivery would be 10 minutes [4]. This means that the service cannot be delivered within 10 minutes.

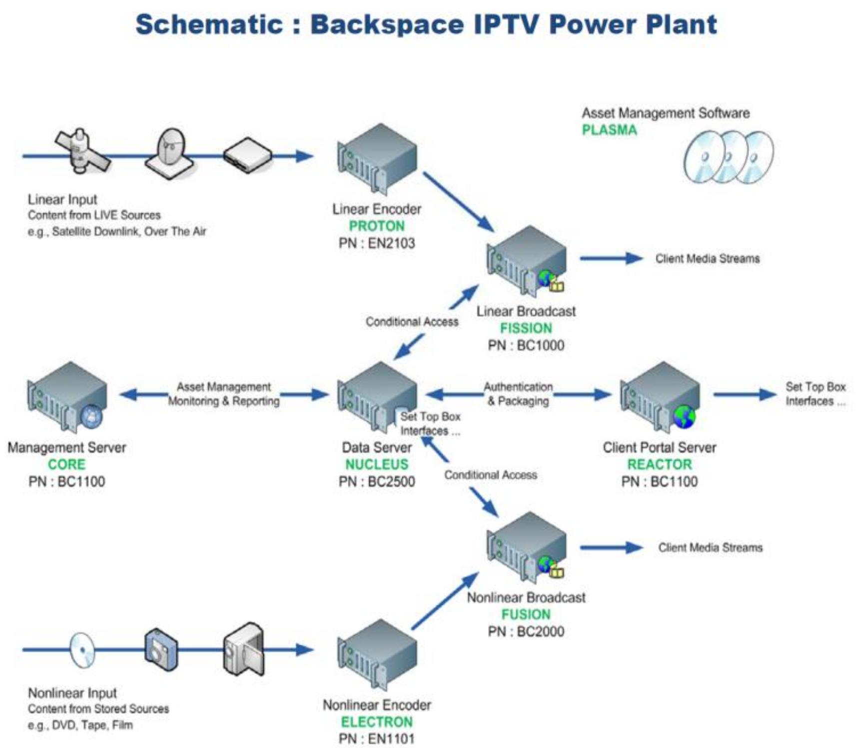

Figure 1. A Schematic Diagram to Describe the Structure of IPTV System

This paper presents a Petri net model of IPTV service. Backspace is a company that provides IPTV services and IPTV systems installation. Figure 1 describes its IPTV system architecture. There are two types of services a subscriber can choose through a set top box, one is live broadcasting and the other is VOD (Video on Demand). The live broadcasting is served through linear broadcast. Linear broadcast sequentially pumps out two kinds of streams, one is from a satellite receiver and the other is from a data server. The satellite receiver catches and passes the signals from TV broadcasting stations, while the data server sends the selected multimedia contents to the linear broadcast. VOD services provide subscribers with the menu so that they can select a video, then VOD streams out the selected video to the subscriber. PROTON, FISSION, CORE, NUCLEUS, REACTOR, FUSION, and ELECTRON found in Figure 1 are Backspace's products. For example, the trade mark CORE is the management server sold by Backspace.

In short, this paper presents Petri net models of the typical IPTV system such as Backspace IPTV system architecture and IPTV service delivery scenarios. Then, we discuss our simulation results performed on the models. After simulation, we employ the Petri net method of the minimum cycle time [3] to estimate the time duration required to deliver IPTV services to users. This time duration is a function of several variables of the IPTV system, such as the number of subscribers, server speed, DBMS speed, etc.

During the minimum cycle time analysis, we have to calculate the minimum positive T-invariant and the minimum S-invariants of the Petri net. It takes about n3 time units to obtain them if n represents the size of the Petri net. The main contribution of this paper is proposing a divide-and-conquer minimum cycle time analysis method that is faster than the existing ones.

2. Related Works

Most IPTV systems tend to multicast multimedia contents to efficiently use the communication bandwidth. An IPTV system stores all of its multimedia contents at its SHOs (super hub offices). A VOD system usually has several SHOs. A VOD system usually has many VHOs (video hub offices), one per region, too. A portion of the multimedia contents of SHO is stored at a VHO so that a local subscriber can get the service from the local VHO. If a subscriber requests to access a movie that is not stored at the local VHO, then the VHO needs to download the movie from a SHO. In [1], an idea is proposed for grouping several VHOs to avoid too frequently downloading video contents from a SHO.

The technology of providing IPTV services to mobile terminals is called mobile IPTV. Mobile IPTV enables users to enjoy IPTV services anywhere and anytime as long as they are connected with wireless networks. Mobile IPTV users can not only receive but also upload multimedia contents. In [2], a traditional internet based mobile IPTV service structure is introduced. Figure 2 is a schematic diagram of a mobile IPTV structure. During the early stage of mobile IPTV, information flew from the network entity to the receiver through wireless communication. The definition of IPTV given by ITU-T FG does not designate a specific network technology for its communication path, and various wireless networks such as WLAN, WiMAX, Cellular, etc were used. Nowadays, a mobile device can both send and receive information, and UCC (User Created Contents) has become very popular. That is, mobile terminal users can create and upload multimedia contents while on the move.

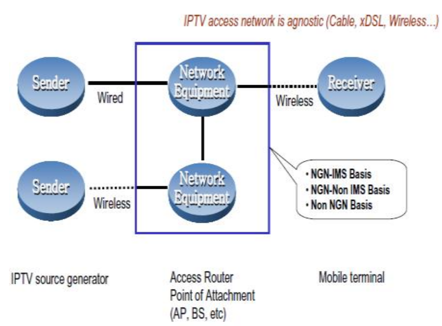

Figure 2. A Schematic Diagram of Mobile IPTV Structure

One of the main features of IPTV service is that it should run without jitter. One of the ways of guaranteeing continuous service is that we build an IPTV management platform with many OSSs (Operations Support Systems), so that a failure of an OSS can be immediately replaced with other sound OSSs. Another reason for using many OSSs is that an IPTV system usually consists of many components purchased from many vendors. Once we have one OSS per component, we can relatively easily integrate all OSSs by applying a service-oriented architecture (SOA) technique. In [5], an integrated network management system is introduced for an end-to-end IPTV network architecture including core network, access network, and home network. The management system consists of many OSSs integrated by SOA technique.

If an IPTV business provides users with the contents and services right when they need them and accessing the services is easy, then the business will probably succeed. In [6], the authors propose an IPTV business model that they believe creates a successful business model. In the model, IPTV contents and services are efficiently packaged, advertised, and delivered to the user.

Many more TVs than computers are available, and e-learning through TV not through a computer is the only possible choice for some people. Therefore, providing IPTV based e-learning is very valuable. However, the input device for IPTV is so simple that converting computer based e-learning software into IPTV based one is very difficult. In [7], the learning process of IPTV platforms is discussed and a method of designing IPTV based interactive e-learning systems is proposed.

We are assuming that readers are familiar with Petri net terminologies, such as transition, place, arc, weight, initial marking, firing, and so on. For introduction to Petri net theory and its applications, please refer to references [3] and [8-11]. A definition of Petri net is as follows:

[Def. 1] A Petri net is a bipartite graph defined as shown in Table 1.

Table 1. A Definition of Petri Net

A Petri net N is 5-tuple, N = (P, T, F, W, M0), where:
$P=\{p_1, p_2, ..., p_m\}$, a finite set of places,
$T=\{t_1, t_2, ..., t_n\}$, a finite set of transitions,
$F \subseteq (P\times T)\cup(T\times P)$ is a set of arcs,
$W: F \rightarrow \{1, 2, 3, ...\}$ is a weight function with $W(a)=0$ for $a \notin F$,
$M_0 : P \rightarrow \{0, 1, 2, 3, ...\}$ is the initial marking,
$P\cap T=\Phi$ and $P\cup T \neq \Phi$.

A system behavior can be described in terms of system states and their changes. In order to simulate the dynamic behavior of a system, a state or marking M in a Petri net is changed to M' according to the following transition firing rule:

1. A transition t is said to be enabled if each input place p of t is marked with at least $w(p,t)$ tokens.

2. An enabled transition may fire.

A firing enabled transition t removes $w(p,t)$ tokens from each input place p of t, and adds $w(t,p)$ tokens at each output place p of t.

We have submitted a paper [15] that proposes a divide-and-conquer method for minimum cycle time analysis. This paper is an extension of [15], i.e., analysis with CPNTools has been added.

3. Building a Petri net Model

We are building a Petri net model for an IPTV system. A satellite receiver catches PAL, NTSC, and MPEG2 signals [12], and the satellite STB (set-top-box) transmits the selected channel signals to the encoder. The real time encoder transforms the input signal into H.264 format, which is the most widely used form in recording, compressing, and distributing HD (High Definition) video and relay to the streaming server.

NLE (Non-Linear Edit) is an editor with which we can edit movies, videos, audio, and computer graphics into a desired video content. Outputs of NLE should also be transcoded into H.264. We can also directly transcode movies in avi (Audio Video Interleave), mov

(QuickTime File Format: QTFF), etc. formats into H.264 without editing. The H.264 format files are stored in NAS (Network-attached storage) devices. In the system, NAS for "Real-time VOD" is different from that for "VOD". In this paper, "Linear VOD" denotes the service of streaming out the videos in the sequence set ahead by the operator.

We can view the system in terms of subscribers as follows. When a subscriber accesses the system through the Internet, the authentication module of the system is invoked and checks if he is an authorized subscriber. An authorized subscriber has choices of "Live broadcast," "Linear VOD," and "VOD". After selection, the remaining procedures of accessing their services are very similar, and we are only describing the case of "Live broadcast". Once "Live broadcast" has been selected, "Relay for Streaming" provides the subscriber with the IP address of the "Streaming Server" so that the subscriber can access the "Streaming Server" and receive "Live Broadcast" through the Internet.

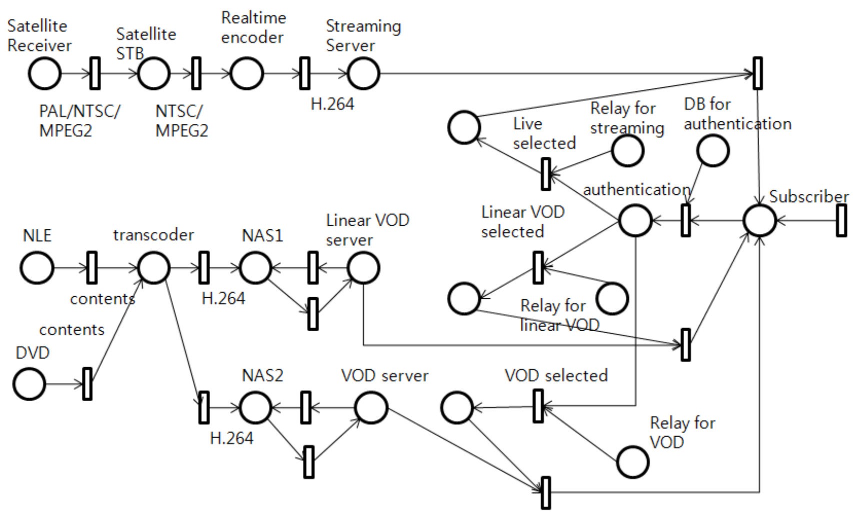

Figure 3. A Petri Net Model of an IPTV System

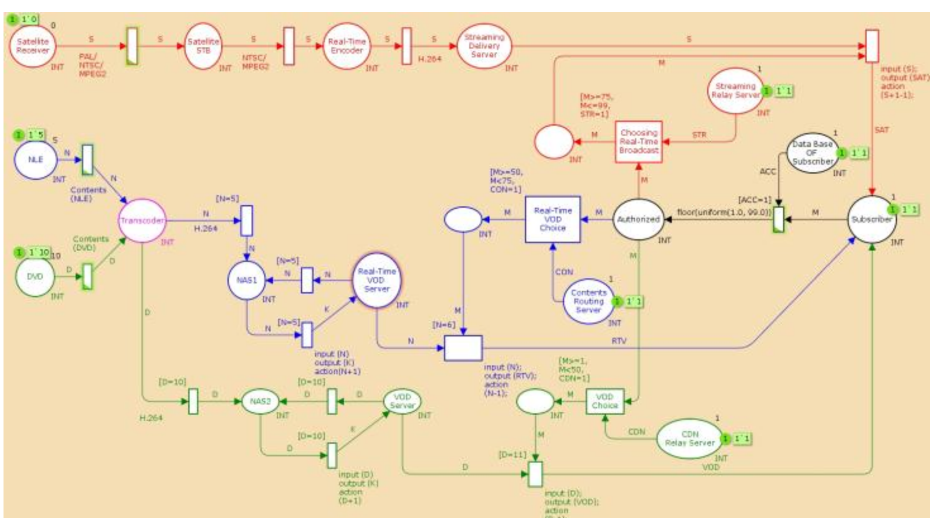

Figure 4. A Colored Petri Net in CPNTools Representing Figure 3

The IPTV system described above can be represented in a Petri net as shown in Figure 3. For the analysis of the Petri net, we have used CPNTools [13], a colored Petri net analysis tool that provides both an editor and a simulator. Using the editor, we can build up a colored Petri net model, and we can explore all the reachable markings of the Petri net model using the simulator. the colored Petri net model of Figure 3 built in CPNTools is shown in Figure 4. In the figure, a subscriber is denoted by integer "1". An authorized subscriber can select one out of the three services: "Live Broadcasting", "Linear VOD", and "VOD". Among the service types, VOD service is the most popular [1]. In order to reflect the frequencies of choices, we have associated "floor(uniform(1.0, 99.0))" with the input arc of the place "Authorized" in the figure. This function randomly returns an integer between 1 and 99. The random number is represented by M in the model and interpreted as follows:

75 <= M <= 99 -> Live Broadcast

50 <= M < 75 -> Linear VOD

1 <= M < 50 -> VOD

The sequence of video files, to be streamed out, from the "Linear VOD server" is established by a system operator. A video file is copied into the "Linear VOD server" from NAS1 when it is selected by the operator. In order to prevent files from being copied into the "Linear VOD server" more than once, we increase the value of the input token of the place "Linear VOD server" and put the guard, [N=5], on the input and output transitions of the place "NAS1". Similarly, we increase the value of the input token of "VOD Server", and we put guards on the transitions around NAS2. Assuming the number of subscribers is one, the number of content resources received from the satellite, the number of movies produced by NLE content, and the number of DVD content are all 1's; we marked the places Subscriber, Satellite Receiver, NLE, and DVD with one token of "1", "1", "5", and "10", respectively.

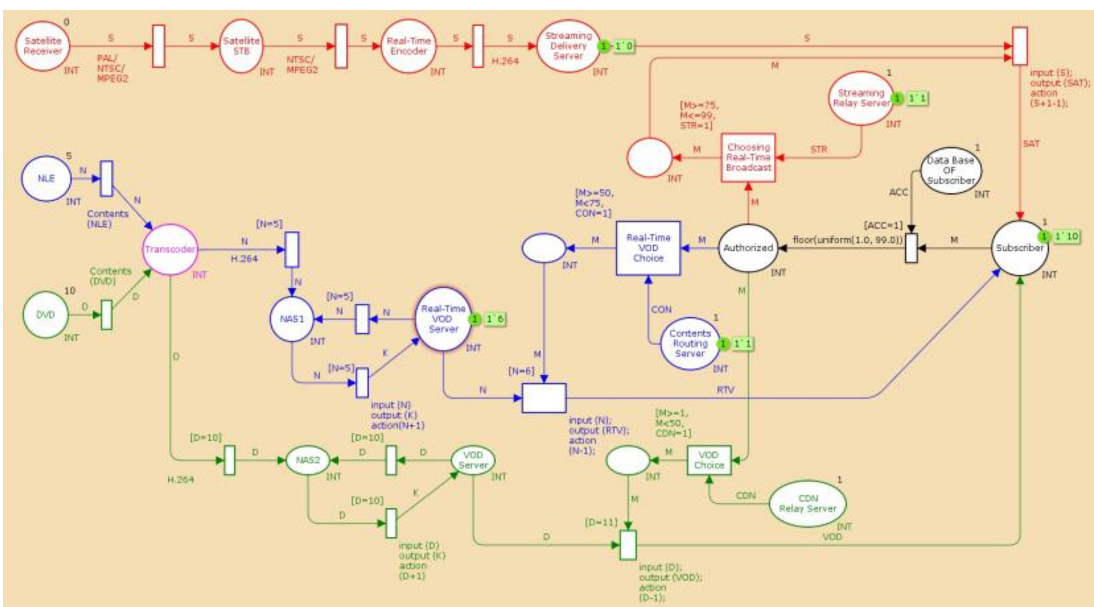

Figure 5. The Result of a Simulation Performed on the Model Shown in Figure 4

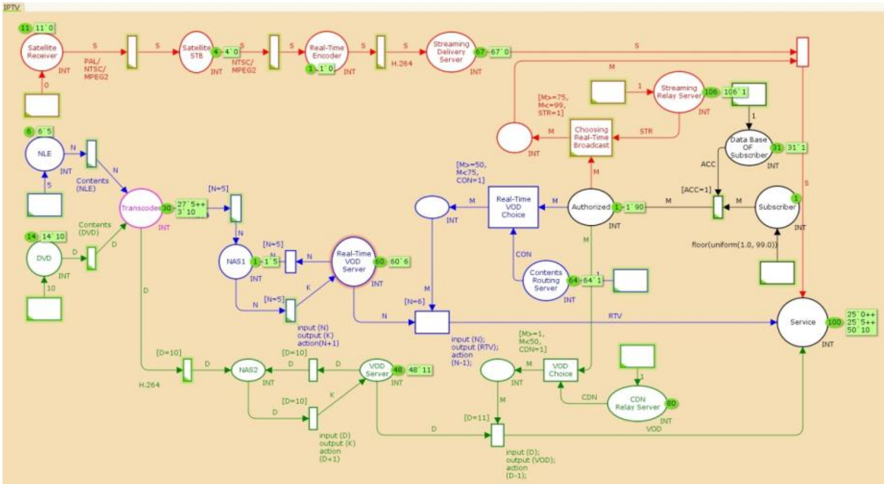

Figure 6. Modification of Figure 4 to Handle an Arbitrary Number of Subscribers

4. Experiments

A simulation result performed on Figure 4 is shown in Figure 5. Token '10' in Subscriber implies that the subscriber's choice was "VOD". Token "0"("5") would be there if the subscriber had chosen "Live Broadcasting"("Linear VOD"). Figure 5 shows that the model successfully handles the case of one subscriber. We have extended the model so that it can handle an arbitrary number of subscribers as shown in Figure 6. Each of the places "Subscriber", "Satellite Receiver", etc. has a source transition so that they can have an infinite number of tokens. We have also added the place "Service" to the model. The tokens of "0", "5" and "10" at the place "Service" represent "Real-Time Broadcast", "Linear VOD", and "VOD" services, respectively. Figure 6 shows the result of running the simulation of a few thousand steps. There are currently 25 "0", 25 "5", and 50 "10" tokens. The ratio of each of the numbers of tokens for the service types to the total number of tokens at the place "Service" coincides with the frequency ratios of selecting services at the place "Authorized".

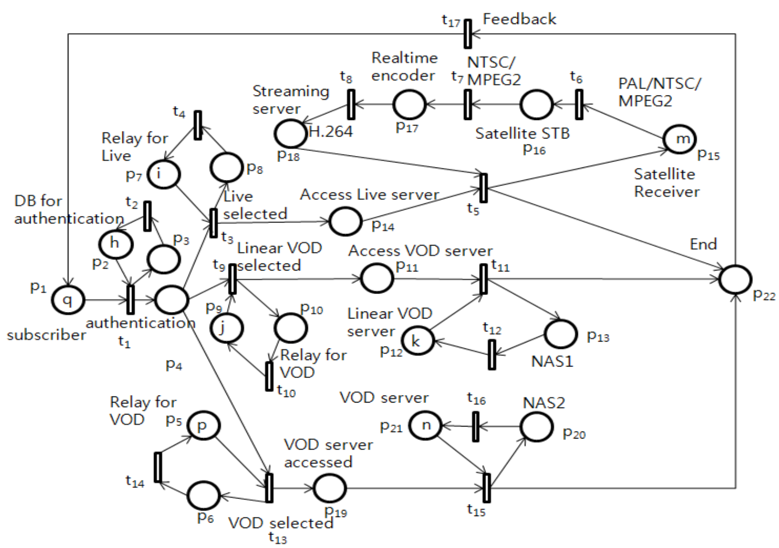

Figure 7. A Consistent Petri Net Representation of the IPTV System

5. Minimum Cycle Time

The service capacity of an IPTV system can be defined by the number of subscribers the system can handle without causing inconvenience. Instances of inconvenience include late response in authentication and in accessing a server and delay and disconnection in the middle of playing. A cycle time is defined as the time to complete a firing sequence from the initial marking back to the same marking after firing each transition at least once. This is closely related to the system response time and the service capacity of the system. Therefore, we are performing the minimum cycle time analysis process introduced in [3, 14] on the IPTV system. We denote a cycle time as τ. It is well known that a lower bound of the cycle time τ or the minimum cycle time is given by:

$$\tau_{\min} = \underset{k}{Max}\{y_k^T (A^-)^T Dx / y_k^T M_0\} \quad (1)$$

where, y_k is an S-invariant, $A^- = \left[a_{ij}^-\right]_{n\times m}$, x is a positive T-invariant, $a_{ij}^- = w(p_j, t_i)$, D is the diagonal matrix of d_i, $i =$1, 2, ..., n, and M_0 is the initial marking. In order to make our Petri net model of the IPTV system have a positive T-invariant, we change the Petri net shown in Figure 3 into the net shown in Figure 7.

Table 2. The Incidence Matrix of the Peri Net Shown in Figure 7

t \ p	1	2	3	4	5	6-18	19	20	21	22
1	-1	-1	1	1		…				
2		1	-1							
3				-1						
4										
5										1
6										
7										
8										
9				-1						
10										
11										1
12										
13				-1	-1		1			
14					1					
15							-1	1	-1	1
16								-1	1	
17	1									-1

The incidence matrix, A of the net is shown in Table 2. The minimal positive T-invariant, x of the net is equation (2).

$$x^T = (3311111111111113) \quad (2)$$

It has 8 S-invariants as follows:

$$y_1^T = (0110000000000000000000)$$
$$y_2^T = (0000110000000000000000)$$
$$y_3^T = (0000001100000000000000)$$
$$y_4^T = (0000000011000000000000)$$
$$y_5^T = (0000000000000011110000)$$
$$y_6^T = (0000000000011000000000)$$
$$y_7^T = (0000000000000000000110)$$
$$y_8^T = (1001000000100100001001)$$

Several places such as p_1, p_2, etc. of the Petri net shown in Figure 7 are inscribed with letters representing constants. For example, 'q' is written in p_1 to represent there are 'q' subscribers. In the same manner, 'h' in p_2 represents the number of processors in the authentication DBMS, while 'i' in p_7 represents the number of processors in the 'relay server' for 'live broadcasting server'. There can be 'm' number of 'live broadcasting servers' as 'm' is written in p_{15}, and a 'relay server' takes requests of 'live broadcast' from users and connects the user to the most appropriate server in the sense of load balancing and proximity. The initial marking M_0 of the net is equation (3).

$$M_0 = (\mathrm{q\,h\,0\,0\,p\,0\,i\,0\,j\,0\,0\,k\,0\,0\,m\,0\,0\,0\,0\,0\,n\,0}) \quad (3)$$

For each of the 8 S-invariants, we compute $y_k^T (A^-)^T Dx / y_k^T M_0$ as follows:

$$y_1^T (A^-)^T Dx / y_1^T M_0 = (3d_1 + 3d_2)/h$$
$$y_2^T (A^-)^T Dx / y_2^T M_0 = (d_{13} + d_{14})/p$$
$$y_3^T (A^-)^T Dx / y_3^T M_0 = (d_3 + d_4)/i$$
$$y_4^T (A^-)^T Dx / y_4^T M_0 = (d_9 + d_{10})/j$$
$$y_5^T (A^-)^T Dx / y_5^T M_0 = (d_5 + d_6 + d_7 + d_8)/m$$
$$y_6^T (A^-)^T Dx / y_6^T M_0 = (d_{11} + d_{12})/k$$
$$y_7^T (A^-)^T Dx / y_7^T M_0 = (d_{15} + d_{16})/n$$
$$y_8^T (A^-)^T Dx / y_8^T M_0 = (3d_1 + d_3 + d_5 + d_9 + d_{11} + d_{13} + d_{15} + 3d_{17})/q$$

In the expressions, d_i represents the delay time of transition t_i, and each of the results (right hand side of the expressions) represents the cycle time associated with S-invariant, y_i. The minimum cycle time of the net is the maximum of the eight cycle times. If the 'service capacity' of the IPTV system is 'q', then the last one should be the maximum among them, i.e., the minimum cycle time should be $(3d_1 + d_3 + d_5 + d_9 + d_{11} + d_{13} + d_{15} + 3d_{17})/q$. If this is the case, then we would either try to increase 'q' to maximize the number of subscribers or try to decrease h, i, …, p to save resources.

On the other hand, if something else, for example $y_2^T(A^-)^T Dx / y_2^T M_0 = (d_{13} + d_{14})/p$, is the maximum, then we can conclude that the system cannot serve 'q' subscribers at the same time, and the part associated with the maximum cycle time, 'relay for VOD' in this example, is the bottleneck. In this case, we have to enhance the performance of 'relay for VOD' by adding more processors, replacing secondary storages with faster ones, or replacing the software with a more efficient one. The enhancement will result in either shortening the delay times of d_{13} and/or d_{14} or increasing 'p'. Both cases will decrease $y_2^T(A^-)^T Dx / y_2^T M_0 = (d_{13} + d_{14})/p$. We have to repeat the identification of the maximum cycle time and enhancement of the part associated with it until $y_8^T(A^-)^T Dx / y_8^T M_0 = (3d_1 + d_3 + d_5 + d_9 + d_{11} + d_{13} + d_{15} + 3d_{17})/q$ becomes the maximum one.

6. Divide and Conquer Method

During the process of minimum cycle time analysis, we have to calculate the minimum positive T-invariant and the minimum S-invariants. It takes about n^3 time units to obtain them. If it is possible to divide the whole Petri net into a few small sub-Petri nets and apply the minimum cycle time analysis on the sub-Petri nets to come up with the same results as we obtained from the whole Petri net, then we can save a lot of calculation time. The main purpose of this paper is to introduce a divide-and-conquer method of minimum cycle time analysis. Before applying the method, we want to reduce a series of transitions and places that have only one input and only one output. For example, p_{15}-> t_6 -> p_{16}-> t_7 -> p_{17} -> t_8 -> p_{18} is such series and is reduced into p_{15}-> $t_{6,7,8}$ -> p_{18} where $t_{6,7,8}$ is the label of the transition.

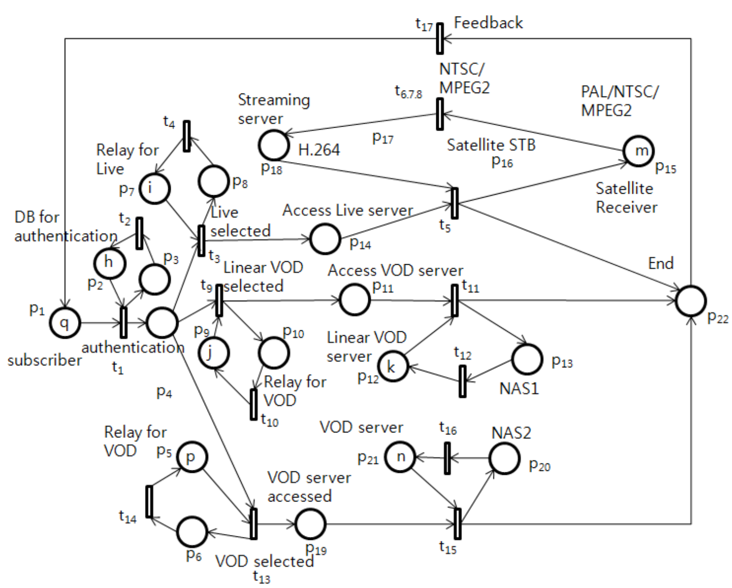

Figure 8. Result of a Serial Reduction

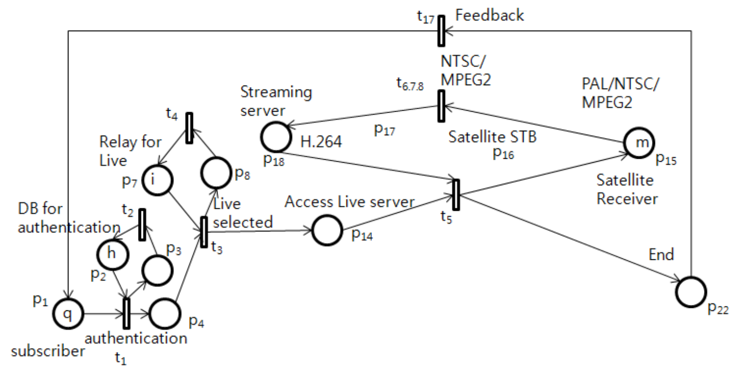

Figure 9. The Sub-graph based on the Cycle (p1, p4, p14, p22)

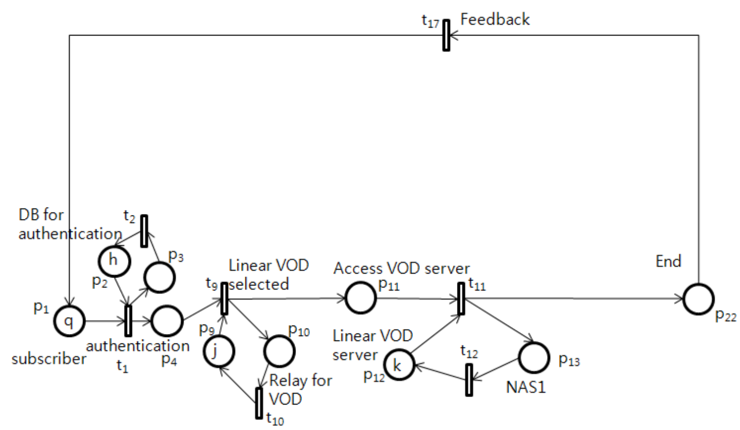

Figure 10. The Sub-graph based on the Cycle (p1, p4, p11, p22)

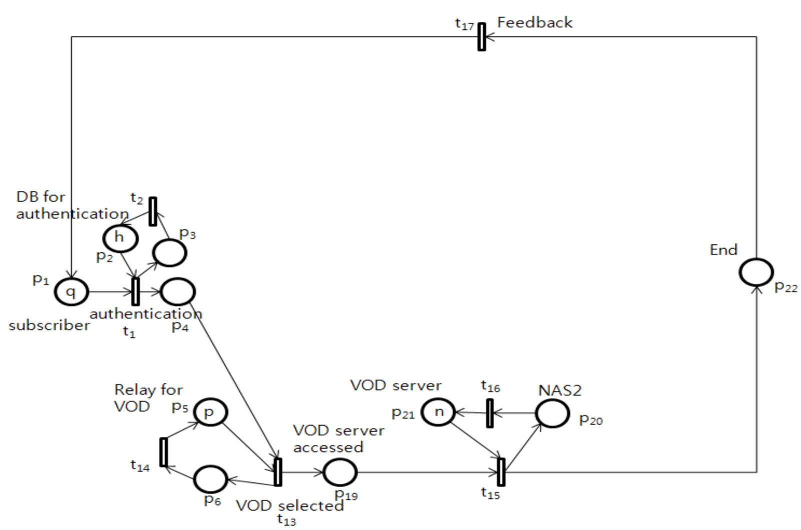

Figure 11. The sub-graph based on the cycle (p1, p4, p19, p22)

An IPTV service is initiated at subscriber place (p_1) and terminates at End place (p_{22}). That is, the cycle time of the system begins at p_1 and ends at p_{22}. Therefore, we can divide the Petri net into three sub Petri nets as shown in Figs. 9, 10, and 11.

Table 3. The Incidence Matrix of the Petri net shown in Figure 9

t \ P	1	2	3	4	7	8	14	15	18	22
1	-1	-1	1	1						
2		1	-1							
3				-1	-1	1	1			
4					1	-1				
5							-1	1	-1	1
6,7,8								-1	1	
17	1									-1

The incidence matrix of the Petri net shown in Figure 9 is shown in Table 3. The minimum positive T-invariant of the Petri net is the following:

$x^T = (1111111)$.

The minimum S-invariants of the Petri net in Figure 9 are:

$$y_1^T = (0\,1\,1\,0\,0\,0\,0\,0\,0\,0)$$
$$y_2^T = (0\,0\,0\,0\,1\,1\,0\,0\,0\,0)$$
$$y_3^T = (0\,0\,0\,0\,0\,0\,0\,1\,1\,0)$$
$$y_4^T = (1\,0\,0\,1\,0\,0\,1\,0\,0\,1).$$

The initial marking of the Petri net in Figure 9 is:

$M_0 = (q\,h\,0\,0\,i\,0\,0\,m\,0\,0)$.

Therefore, by substituting the x, y and M_0 into equation 1, we obtain the following:

$$y_1^T (A^-)^T Dx / y_1^T M_0 = (d_1 + d_2)/h \quad (4)$$
$$y_2^T (A^-)^T Dx / y_2^T M_0 = (d_3 + d_4)/i \quad (5)$$
$$y_3^T (A^-)^T Dx / y_3^T M_0 = (d_5 + d_6 + d_7 + d_8)/m \quad (6)$$
$$y_4^T (A^-)^T Dx / y_4^T M_0 = (d_1 + d_3 + d_5 + d_{17})/q \quad (7)$$

Similarly, we obtain the following for the Petri net in Figure 10:

$$y_1^T (A^-)^T Dx / y_1^T M_0 = (d_1 + d_2)/h \quad (8)$$
$$y_2^T (A^-)^T Dx / y_2^T M_0 = (d_9 + d_{10})/j \quad (9)$$
$$y_3^T (A^-)^T Dx / y_3^T M_0 = (d_{11} + d_{12})/k \quad (10)$$
$$y_4^T (A^-)^T Dx / y_4^T M_0 = (d_1 + d_9 + d_{11} + d_{17})/q \quad (11)$$

In the same manner, we obtain the following for the Petri net in Figure 11:

$$y_1^T\left(A^-\right)^T Dx / y_1^T M_0 = (d_1 + d_2)/h \qquad (12)$$

$$y_2^T\left(A^-\right)^T Dx / y_2^T M_0 = (d_{13} + d_{14})/p \qquad (13)$$

$$y_3^T\left(A^-\right)^T Dx / y_3^T M_0 = (d_{15} + d_{16})/n \qquad (14)$$

$$y_4^T\left(A^-\right)^T Dx / y_4^T M_0 = (d_1 + d_{13} + d_{15} + d_{17})/q \qquad (15)$$

With equations (4) through (15), we can obtain the same minimum cycle time analysis results for the original Petri net by summing up the common delays in the equations. For example, by summing up the common delays in equations (4), (8), and (12), we obtain:

$$\left(3d_1 + 3d_2\right)/h. \qquad (16)$$

Similarly, we obtain the following from equations 7, 11, and 15:

$$\left(3d_1 + d_3 + d_5 + d_9 + d_{11} + d_{13} + d_{15} + 3d_{17}\right)/q \qquad (17)$$

Finally, equations 16, 5, 6, 9, 10, 13, 14, and 17 are identical to the equations we obtained from the Petri net shown in Figure 7.

7. Conclusion

More and more IPTV systems are implemented and providing services. It is well known that the maintenance cost is usually much bigger than the initial implementation cost. Therefore, many efforts to reduce maintenance cost have been carried out by many researchers. One of the widely used methods of saving maintenance cost is making sure the system design satisfies the user requirements. In order to formally verify that the IPTV system under design satisfies the user requirements, we have introduced Petri net representation of IPTV systems. Once we have a Petri net model of an IPTV system, we can identify the characteristics of the system by running a simulation on the Petri net or by mathematically analyzing the Petri net.

In order to run simulations, we have built up a colored Petri net model of the IPTV system using CPNTools. Our simulation results showed that the system is live and deadlock free. They also showed that the distribution of being-served service types coincides with the distribution of requested service types. The execution times of the components of the IPTV system determine the service capacity of the system. In order to verify that the service capacity meets the user requirements, we used the minimum cycle time analysis. We have also showed that we can optimize the system using the minimum cycle time analysis.

During the minimum cycle time analysis, we have to calculate the minimum positive T-invariant and the minimum S-invariants of the Petri net. It takes about n3 time units to calculate them where n represents the size of the Petri net. For example, the minimum cycle time analysis of the Petri net in Figure 7 takes about 173+223 = 15,561 time units. Whereas, the minimum cycle time analysis of the Petri nets in Figures 9, 10, and 11 takes about 103+73 + 103+73 + 103+73 = 4,029 time units.

In summary, we have built a Petri net model of our IPTV system, we have run simulations to verify that the system is live and behaves as we expected, and we have performed the minimum cycle time analysis to show that the service capacity of the system meets the user requirements. We have also showed how the minimum cycle time analysis can be used to optimize the IPTV system. The main contribution of this paper is to propose a divide-and-conquer minimum cycle time analysis

Acknowledgements

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2011-0006942) and by 'Industry, Academy and Research Institute Co-development of Techniques' funded by the Small and Medium Business Administration (R00046281)

References

[1] S. Meng, L. Liu and J. Yin, "Scalable and Reliable IPTV Service Through Collaborative Request Dispatching", *IEEE International Conference on Web Services*, pp. 179-186, **(2010)**.

[2] S. Park, S. Jeong and C. Hwang, "Mobile IPTV Expanding the Value of IPTV", *Seventh International Conference on Networking*, pp. 296-301, **(2008)**.

[3] T. Murata, "Petri nets: Properties, analysis and applications", *Proceedings of the IEEE*, *77*(4), pp. 541-580, **(1989)**.

[4] http://www.enotes.com/cycle-time-reference/cycle-time-176693.

[5] C. Hu, Y. Hsu, C. Hong, S. Hsu, Y. Lin, C. Hsu and T. Fang, "Home network management for IPTV service operations - A service provider perspective", *IEEE/IFIP Network Operations and Management Symposium Workshops*, pp. 1-7, **(2010)**.

[6] B. Obele, S. Han, J. Choi and M. Kang, "On building a successful IPTV business model based on personalized IPTV content & services", *9th International Symposium on Communications and Information Technology*, pp. 809-813, **(2009)**.

[7] Z. Gang, W. Lin, Z. Yang, Q. Liu, M. Wang and R. Li, " Research and Design of Interactive IPTV based E-Learning System", *7th International Conference on Information Technology Based Higher Education and Training*, pp. 536-540, **(2006)**.

[8] J. L. Peterson, *Petri Net Theory and the Modeling of Systems*. Prentice-Hall, N. J. **(1981)**.

[9] K. Golriz and A. Mohammad, "DNS Spoofing Attack Simulation for Model-Based Security Evaluation", *International Journal of Advanced Science and Technology* Vol. 1, **(2008)** December, pp.1-8.

[10] J. Liu, L. Zhang, Y. Zhong and Y. Chen, "Middleware-based Distributed Systems Software Process", *International Journal of Advanced Science and Technology* Vol. 13, December, **(2009)**, pp. 27-48.

[11] A. Abdelli, "A Much Compact Abstraction of the State Space of Real time Preemptive Systems", *International Journal of Advanced Science and Technology* Vol. 27, **(2011)** February, pp. 45-58.

[12] http://en.wikipedia.org/wiki/MPEG-2.

[13] http://cpntools.org/

[14] J. Yim, J. Joo and Y. Lee, "Timed net with choice probability and its minimum cycle time - the case of location based service," Expert Systems With Applications 31(2), pp. 221-230, **(2006)**

[15] J. Yim and J. Woo, "A Divide-and-Conquer Method for Minimum Cycle Time Analysis", submitted to a journal for possible publication.

Authors

Jaegeol Yim

Jaegeol Yim received the M.S. and Ph.D. degrees in Computer Science from the University of Illinois at Chicago, in 1987 and 1990, respectively. He is a Professor in the Department of Computer Engineering at Dongguk University at Gyeongju Korea. His current research interests include Petri net theory and its applications, Location Based Service, AI systems, and multimedia systems. He has published more than 50 journal papers, 100 conference papers (mostly written in Korean Language), and several undergraduate textbooks.

Gyeyoung Lee

Gyegoung Lee received the M.S. degree in Computer Science from the Dongguk University in 1982 and the Ph.D. degree in Computer Engineering from the Dankook University in Korea in 1992, respectively. He is a Professor in the Department of Computer Engineering at Dongguk University at Gyeongju, Korea. His current research interests include Petri net theory, AI systems and Speech processing systems.