Peer-to-Peer Netw. Appl. (2015) 8:610–619
DOI 10.1007/s12083-014-0257-3

Estimating unreliable objects and system reliability in P2P networks

Kapsu Kim · Myunghui Hong · Kyungyong Chung · SangYeob Oh

Received: 15 January 2014 / Accepted: 26 February 2014 / Published online: 28 March 2014

Abstract It is a very difficult task to estimate abnormal objects and analyze reliability in peer-to-peer (P2P) networks. In the P2P network environment, successful execution of a program is conditional on the successful access of related files and software distributed throughout the P2P network. Abnormal phenomena in peer-to-peer systems occur very often. They occur in software and in data operated on by the computer itself or by a connected computer. This paper focuses on estimating abnormal objects and reliability in a P2P network that is very complex and different. We propose an estimation method for abnormal objects and reliability in a P2P network. First, we define a P2P static graph model, where the node is a computational unit and the edge is a communication unit in the P2P network. The computational unit is software or data installed on a computer. Second, we propose a converting algorithm where the P2P static graph model is converted to a colored Petri net model. Last, we estimate abnormal objects and the reliability of the P2P network. The technique successfully classifies abnormal objects and estimates reliability of the P2P network. This procedure provides a very useful method for detection of abnormal objects in a P2P network, which occur through abnormal faults of software and data. While we manage the P2P network, the reliability of the system can be predicted.

Keywords P2P Network · Abnormal Object · Colored Petri Net · Reliability

K. Kim · M. Hong
Department of Computer Education, Seoul National University of Education, 1650, Seocho-dong, Secho-gu, Seoul 137-742, South Korea

K. Kim
e-mail: kskim@snue.ac.kr

M. Hong
e-mail: mhhong@snue.ac.kr

K. Chung
School of Computer Information Engineering, Sangji University, 83, Sangjidae-gil, Wonju-si, Gangwon-do 220-702, South Korea
e-mail: Kyungyong.chung@gmail.com

S. Oh (✉)
Department of Interactive Media, Gachon University, Bokjeong-dong, Sujeong-gu, Seongnam-si, Gyeonggi-do 461-701, South Korea
e-mail: syoh1234@gmail.com

1 Introduction

The “peer” in a peer-to-peer (P2P) network is the computer that is connected to another, or connected to the Internet. So, P2P services in a P2P system are very complex and dynamic. Although Web services need a central server, P2P services are localized. So, local servers in a P2P system are needed. Local servers are the users’ local personal computers. Files in a P2P system can be shared directly between local computers on the network without the need for a central server. In other words, each local computer on a P2P network becomes a file server as well as a client. In a P2P network, tasks (such as searching for files or streaming audio/video) are shared amongst multiple interconnected peers that each make a portion of their resources directly available to other network participants, without the need for centralized coordination by servers [1].

The structure types for P2P network models are the structured P2P network, the unstructured P2P network, and the hybrid P2P network. The unstructured P2P network model does not have a particular structure for the layout of the network by design, but rather is formed by nodes that randomly form connections to each other [2, 3]. Because there is no structure globally imposed upon them, unstructured networks are easy to build and allow for localized optimizations to different regions of the overlay [4]. Also, because the role of all peers in the network is the same, unstructured networks are highly robust in the face of high rates of "churn" (when large numbers of peers frequently join and leave the network) [5–16]. A structured P2P network is organized into a specific topology, and the protocol ensures that any node can efficiently [17] search the network for a file/resource, even if the resource is extremely rare. Hybrid models are a combination of peer-to-peer and client–server models [15, 16, 18]. P2P services in a P2P network are very complex and difficult to manage. Many unusual symptoms occur in a P2P service, so such a phenomenon can be very hard to find. The only resources needed for a user's computer to use a P2P network are an Internet connection, P2P software, and files. General P2P software allows the user to access thousands of other users' systems on the network. The quality of the P2P service has a major impact on the overall P2P system. In general, it is also known that small abnormal actions of an object in a P2P service are responsible for the majority of abnormal objects. Handling faults is generally an expensive part of P2P service software development projects. Managing and completing the P2P service is a very difficult task. The greater the delay in detecting abnormal objects of a Web service after it occurs, the more expensive it is to correct. To predict the faults of a Web service for implementation, information such as the source code may be necessary [7–10, 13–15, 19–21]. This difficulty arises from abnormal objects in program and data files that are distributed among several computers. Many copies of the data and programs can be distributed in a P2P network, and an abnormal object in the data or programs has an effect on other data or programs. This makes it a very difficult task to estimate abnormal objects and analyze reliability of the P2P network. In this environment, successful execution of a program is conditional on successful access of related files distributed throughout the P2P system. We propose a reliability analysis method and an estimating method for abnormal objects based on the colored Petri net. The colored Petri net [19, 20, 22–25] is a very useful tool for analyzing systems. We adapt this model to estimate abnormal objects and analyze the reliability of the P2P network.

This paper consists of five sections as follows. In Section 2, we describe colored Petri nets. In Section 3, we explain the proposed system model. In Section 4, we validate the proposed model through simulation. In Section 5, we offer conclusions.

2 Related work

2.1 Colored petri net structure

Petri nets were originally developed by C. A. Petri in the 1960s [26], and have been one of the most widely used methods to adequately describe and analyze computer systems from then on. Petri nets themselves, however, have two problems. First, Petri nets themselves only control concepts, but not data concepts. Because all data manipulation has to be represented directly in the net structure, Petri net models can be excessively large. Second, there are no hierarchy concepts, and thus it is not possible to build a large model via a set of separate submodels with well-defined interfaces. The development of high-level Petri nets in the late 1970s, and hierarchical Petri nets in the late 1980s got rid of these two problems. The colored Petri net (CP-net) also shortened to CPN, is one of the two most well-known types of high-level Petri net.

CP-nets [27] incorporate both data structuring and hierarchical decomposition—without compromising the qualities of the original Petri nets. CP-nets are used for three different—but closely related—purposes. First of all, a CP-net model is a description modeling system and can be used as a specification for a system to be built or as a presentation of a system to be explained to other people, or for ourselves. By creating a model, we can investigate a new system before we construct it. This is an obvious advantage, particularly for systems where design errors may be expensive to correct. Second, the behavior of a CP-net model can be analyzed, either by means of simulation that is equivalent to program execution and program debugging, or by means of more formal analysis methods, which are equivalent to program verification. Finally, it should be understood that the process of creating the description and performing the analysis usually gives the modeler a dramatically improved understanding of the modeled system—and this is often more valid than the description and the analysis results themselves.

We redefine CP-nets as follows:

Definition 1 CP-nets structure is a tuple (P, T, I, O, C, μ) where

P $\{p_1, p_2,,,,, p_n\}$ is a finite set of places.
T $\{t_1, t_2,,,,, t_n)$ is a finite set of transitions.
I T -> P is a finite set of input functions where the elements map from places to multiple sets of transitions.
O T -> P is a finite set of output functions where elements map from transitions to multiple sets of places.

C is the color function, which defines P ∪ T into non-empty sets. It attaches each to place a set of possible token-colors and to each transition a set of possible occurrence colors. The initial marking M_0 is a function defined on P, such that $M_0(p) \in C(p)$ k for all $p \in P$, where k denotes a multi-set, which can contain multiple occurrences of the same element.

The places and their tokens represent states, while the transitions represent state changes. But each place may contain several tokens, and each of these carries a data value. The input function $I(t_j)=p_i$ means that there exists an arc from the p_i place to the t_j transition, and the input place of the t_j

transition is p_i. The output function $O(t_j)=p_i$ means that there exists an arc from the t_j transition to the p_i place, and the output place of the t_j transition is p_i. The dynamic aspect of a Petri net model is denoted by markings that are assignments of tokens to the places of a Petri net. The execution of a Petri net is controlled by tokens and their colors in places. The unreliable object estimation of a Petri net is controlled by the number of tokens.

In order to describe the behavior of a system easily, we can extend the input function I and output function O as follows : I : P -> T, O : T -> P, where the number of $(t_j,I(p_i))$ is the number of $(p_i,O(t_j))$, and the number of $(t_j,O(p_i))$ is the number of $(p_i,I(t_j))$.

2.2 Colored petri graph

A graphical representation of a colored Petri net structure is much more useful for illustrating the concepts of a colored Petri net. Therefore, we can define a colored Petri net graph with the following definition.

Definition 2 A colored Petri net graph is a bipartite-directed multi-graph, $G=(V, A)$, where $V=\{v_1, v_2,,,,, v_s\}$ is a set of vertices, and $A=\{a_1, a_2, a_3,,,, a_r\}$ is a bag of directed arcs, $a_i=(v_j, v_k)$, v_j, $v_k \in V$. The set V consists of P and T such that $V=P \cup T$, $P \cap T=\emptyset$ in the place P set and the transition T set of a colored Petri net. Set A consists of a bag of directed arcs such that for all $p_i \in P$ and $t_i \in T$, the number of $(p_i, I(t_j))$ is equal to the number of $((p_i,t_j),A)$, and the number of $(p_i, O(t_j))$ is equal to the number of $((t_j, p_i), A)$. If there exists $a_i=(v_j, v_k)$, for $v_j \in P$, $v_k \in T$, a_i is the element of A, and the token value of input function $v_j=I(v_k)$ is the weight of arc a_i. Also, if there exists $a_i=(v_j,v_k)$, for $v_j \in T$, $v_k \in P$, a_i is the element of A, the token value of output function $v_k=O(v_j)$ is weight of arc a_i. The value of tokens in a place is the value of dots of vertices generated from a place.

[Theorem 1] $G=(V,A)$ is a colored Petri net graph, which is equivalent to a colored Petri net structure where the CP-nets structure is a tuple (P, T, I, O, C, μ).

(Proof) Given that the CP-net structure is a tuple (P, T, I, O, C, μ), we can define a colored Petri net graph as Definition 6 (below). Now, we must convert in the opposite direction (from a colored Petri net graph to a colored Petri net structure). Given colored Petri net graph $G=(V, A)$, we can partition the set V into two disjoint sets, V_1 and V_2, such that $V=V_1 \cup V_2$, $V_1 \cap V_2=\emptyset$. As there can exist dots of vertices set V, the vertices set that includes its vertices is either one of sets V_1 and V_2. Assume it is the vertices set V_1; therefore, vertices set V_1 is the places set of a colored Petri net structure, and vertices set V_2 is the transitions set of a colored Petri net structure. If for $v_i \in V_1$, $v_j \in V_2$, there exists an arc $a_k=(v_i, v_j)$, the a_k arc is the input function $v_j=I(v_i)$ of the v_i transition, and the weight of its arc is a token value $f(v_j, I(v_i))$ of its input function. If for $v_i \in V_2$, $v_j \in V_1$, there exists an arc $a_k=(v_i, v_j)$, the a_k arc is the output function $v_j=I(v_i)$ of the v_i transition, and the weight of its arc is a token value $f(v_j, I(v_i))$ of its output function. The number of dots in the V_1 vertices is the number of tokens in the places. We, therefore, can derive a colored Petri net structure from a colored Petri net graph.

3 Estimating unreliable objects and system reliability

3.1 P2P static graph model

In general, because many programs and data are distributed among many computers [22–24] in the distributed environment, communication between peer computers frequently occurs in a P2P network. That is to say, a program on one peer computer needs a program or data on another peer computer. We call the peer computer the computation object, and the communication between peer computers the communication object. Our objective is to separate computation objects and communication objects. That is to say, the computation object comprises the peer computers and communication objects comprise the other computers' files or programs. We can describe a P2P abnormal objects estimation model as a graph G(N, E), where the set of nodes N corresponds to the computational units, and set E corresponds to the communication links [24]. Figure 1 shows a typical representation of a graph model showing the P2P abnormal objects estimation model.

So, we can define a P2P abnormal objects estimation model with Definition 3, calling it the P2P static graph model (PSGM).

Definition 3 The P2P static graph model is a tuple (N, E) where N is a set of nodes and E is a set of edges.

Figure 1 shows a typical representation for a P2P abnormal objects estimation model where computational units are n_1, n_2, n_3, and n_4. Edge e_{12} is communication link between nodes n_1 and n_2. Edge e_{23} is a communication link between nodes n_2 and n_3. Edge e_{13} is a communication link between nodes n_1

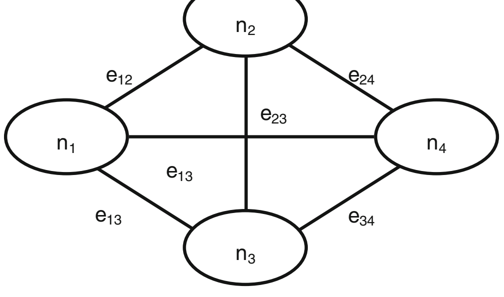


Fig. 1 Abnormal objects estimation model

and n_3. Edge e_{14} is a communication link between nodes n_1 and n_4. Edge e_{24} is a communication link between nodes n_2 and n_4. Edge e_{34} is a communication link between nodes n_3 and n_4.

3.2 Resource sharing table

In the P2P network environment, many programs and data are shared in the node of the P2P static graph model. All nodes in the PSGM share many programs (or software) and data. We describe all peer nodes that have programs and data, so we can define the resource sharing table (RST) with Definition 4.

Definition 4 The resource sharing table is a tuple (N, P, D) where N is a set of nodes, P is a set of programs or software, and D is a set of data files

We can share programs and data in all nodes of the PSGM model in Fig. 1. An example of a resource sharing table is shown in Table 1. In the table, n_1 shares program p_1, program p_2, data file f_1, and data file f_2; n_2 shares program p_3, program p_4, and data file f_3; n_3 shares program p_5, program p_6, data file f_4, data file f_5, and data file f_6; and n_4 shares program p_7, program p_8, program p_9, data file f_7, data file f_8, and data file f_9.

3.3 Object dependence set

An object or program in one node shares programs and data files in other nodes. The reliability of an object or program is affected by the reliability of programs and data files in the other nodes shared by it. So, in order to estimate the reliability of a program, we can make decisions about programs and data that affect it.

We can define an object dependence set (ODS) with Definition 5.

Definition 5 Object Dependence Set

Program P's PDS is defined as the data or other programs in the P2P network that are needed to execute program P.

We assume that every program's ODS is like those in Table 2. That is to say, if program p_1 needs data f_1, data f_2, data f_3, and program p_2, then program p_1's ODS is $\{f_1, f_2, f_3, p_2\}$.

We can see that the P2P network is very complex in dynamic behavioral aspects. It, then, is very difficult to analyze and estimate unreliable objects in the P2P network. In order to show that we share data and other programs in the P2P network to execute a program, we can convert a PSGM into an extended PSGM (EPSGM) comprising the PSGM, RST, and ODS.

Table 1 Resource Sharing Table (RST)

Node	Program	Data
n_1	p_1, p_2	f_1, f_2
n_2	p_3, p_4	f_3
n_3	p_5, p_6	f_4, f_5, f_6
n_4	p_7, p_8, p_9	f_7, f_8, f_9

Definition 6 An EPSGM is a tuple (N, E) where N is a set of nodes and E is a set of edges, where every element of node N has many ODSs.

Based on the EPSGM model, we can easily analyze and estimate an unreliable object from among all objects in the P2P network by using colored Petri nets.

3.4 Transform algorithm

The EPSGM model represents static information in the P2P network. In general, a P2P network has dynamic characteristics. Reliability that is measured or estimated under a dynamic operational environment is valuable. We cannot estimate reliability for the P2P network directly in the EPSGM model. So, we must estimate the reliability of the P2P network with a model that has a dynamic property. We propose an algorithm that converts an EPSGM model to a colored Petri net in order to easily analyze system reliability. The colored Petri net model is a model that describes the dynamic characteristics of the P2P network. Algorithm 1 converts an EPSGM model to a colored Petri net.

```
[Algorithm 1]
Inputs:  (1) PSGM (G) = { N, E }
         (2) RST (resource sharing table)
         (3) ODS
Output : colored Petri net (P, T, I, O, C, μ)

step 1: $P = N \cup E \cup \{p_i | \forall i, p_i \text{ in ODS}\} \cup \{p_s, p_f\}$
                /* $p_s$ is a start place and, $p_f$ is a terminate place*/
step 2: $T_1 = \{ t_i | \forall i, p_i \text{ in ODS}\}$
        $T = T_1 \cup \{t_s\}$   /* $t_s$ is a start transition */
step 3: $I_1(t_i) = \{p_i\} \cup \{ n_k | \forall k, p_i \in n_k \text{ in RST}\}$
        $\cup \{ n_k | \forall k, f_k \in p_i \text{ in ODS and } f_k \in \{ n_k | \forall k, p_i \in n_k \text{ in RST}\} \}$
        $\cup \{ e_{ij} | \forall i \forall j, p_j \in p_i \text{ in ODS and } p_j \in \{ n_k | \forall k, p_i \in n_k \text{ in RST}\} \}$
      $I_2(t_s) = p_s$
      $I = I_1 \cup I_2$
step 4: $O_1(t_i) = \{ p_f \}, O_2(t_s) = \{p_k | \forall i, p_k = I(t_i) \}$
        $O = O_1 \cup O_2$
step 5: $F(I) = \{ (f(p_i, I(t_j)) | \forall i, \forall j (f(p_i, I(t_j))) \}$
step 6: $F(O) = \{ f(p_k, O(t_j)) | \forall k, \forall j \; f(p_k, O(t_j)) = 1 - \prod_{i=1}^{i=n} (1 - f(p_i, I(t_j)) \}$
        (where $I(t_j) = \{p_1, p_2,,,, p_n\}$)
```

Table 2 ODS of Fig. 1

Program	ODS
p_1	$\{f_1, f_2, f_3, p_2\}$
p_2	$\{f_2, p_3\}$
p_3	$\{f_1, f_3\}$
p_4	$\{f_3, p_5\}$
p_5	$\{f_4, f_5, f_6\}$
p_6	$\{f_4, f_5, p_2\}$
p_7	$\{f_7, f_8, f_{9,} p_8\}$
p_8	$\{f_7, f_8, p_2\}$
p_9	$\{f_4, f_5, p_5\}$

There are five types of places in the colored Petri net. The first type comprises places that are made by nodes of the PSGM graph. We call them *node places*. A place is useful for representing computational objects such as a peer site of a P2P network. In this algorithm, all nodes of the PSGM model are a node place of the colored Petri net. The second type comprises places that are made by communication lines between two sites in a P2P network. We call them *edge places*. It is useful for representing communication lines over which a message in a peer computer is sent to another peer computer. The third type is the *program place*. It is useful for representing programs in a P2P network system. All programs of an RST are program places of the colored Petri net. The fourth type is a *start place*. It is a starting point of a colored Petri net, and is expressed by p_s. The last type is a *terminate place*. It is an ending point of a colored Petri net, and is expressed by p_f.

Algorithm 1 shows the six steps.

In the first step, all places of the colored Petri net will be decided. All places of the colored Petri net consist of the five types of places. First, all nodes of the PSGM become node places that are a type of place that is consistent with a colored Petri net. Second, all edges of the PSGM become edge places that are a type of place consistent with a colored Petri net. Third, all programs of the RST become program places that are a type of place consistent with a colored Petri net. Finally, a *start place* and *terminate place* are included as places of the colored Petri net.

In step 2, we define the *start transition* (t_s), and transitions correspond to program places. That is to say, all programs in the ODS are transformed to transitions of the colored Petri net. This step makes a decision about all transitions of the colored Petri net.

In step 3, we can determine that the input place for the *start transition* is p_s and its input is the start place. For example, the input places *start transition*(t_f) in Fig. 2 are places that are made by a start place. A transition's input corresponding to a program place are program places and node places with its transition.

In step 4, the places for output functions of a *start transition* are all program places. For example, the places for output functions of *start transition*(t_s) in Fig. 2 are places that represent place p_1, place p_2, place p_3, place p_4, place p_5, place p_6, place p_7, place p_8, and place p_9.

We can assert that the abnormal rate of the output function in a *start transition* and the abnormal rates of the input functions in a *terminate transition* are 0. For example, the abnormal rate of output function for transition t_s and the input function of transition t_f in Fig. 2 are 0. In step 5 and step 6, we assign a token to all *places* and their *start place* (which is p_s).

We assert that Algorithm 1 is very effective. We show that Algorithm 1 is very useful for analyzing and estimating unreliable objects of the P2P network by converting the abnormal object estimation model into the colored Petri model.

For example, we can convert Fig. 1 into Fig. 2 by Algorithm 1 based on the resource sharing table shown in Table 1, the object dependence set shown in Table 2, and the abnormal object estimation model shown in Fig. 1.

We assume that the abnormal rate of the input function is an exponential distribution in order to easily analyze and estimate system abnormality probability. We can then regard the abnormal rate $f(p_i,I(t_j))$ of the input function $I(t_j)=p_i$ as λ_{ij}, and the abnormal rate $f(p_i,O(t_j))$ of the input function $O(t_j)=p_i$ as λ_{ji}, given the interval time $(t, t+\Delta t)$. Therefore, we can regard a colored Petri net as a continuous time homogeneous Markov process, and we can analyze and estimate abnormal or unreliable objects by means of an analytic method.

A transition is enabled if and only if each of its input places contains at least as many tokens as arcs that exist from that place to the transition, and each input function is not abnormal. When a transition is enabled, it may fire. Whenever a transition fires, all enabling token values are removed from its input places, and a token value is deposited in each of its output places. When a transition fires, all input functions are not abnormal. We can therefore define the firing rate of transition t_j with Definition 7.

Definition 7 The firing rate of transition t_j is

$$R(tj)=\prod_{i=1}^{i=n}\left(1-f\left(pi,I(tj)\right)\right)$$

where $I(t_j)=\{p_1, p_2, p_3,,,,, p_n)$.

The abnormal rate of the output function is a rate at which its transition does not fire. The abnormal rates of all output functions in a transition are the same value as one another. We can define the abnormal rate of the output function that

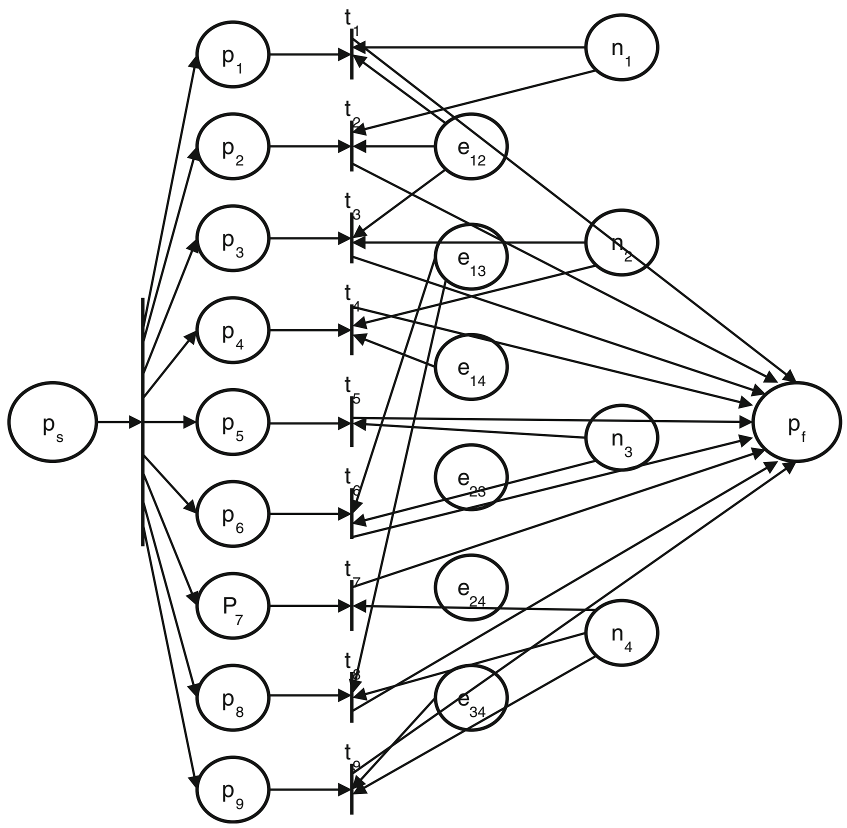


Fig. 2 Colored Petri net model

corresponds to transition t_j with Definition 8. We therefor consider a token value as an abnormal rate of input function or output function.

Definition 8 The abnormal rate is

$$f(p_1, O(t_j)) = f(p_2, O(t_j)) = f(p_3, O(t_j)) = ...f(p_n, O(t_j)) = 1-\prod_{i=1}^{i=n}\left(1-f\left(pi, I(tj)\right)\right)$$

where $O(t_j)$={p_1, p_2,,,,, p_n}.

After the firing rate of all transitions is determined, we can measure the abnormal rate of all output functions with Definition 8. In other words, this abnormal rate is a token value of output functions.

For example, we assume that all program abnormal rates in a P2P network are 0.02, all node abnormal rates are 0.01, and all communication abnormal rates are 0.03. Then, we can assign token values to the input functions. By Definition 7 and Definition 8, we can assign a token value to the output function. Therefore, under the above assumption and Fig. 2, we can calculate transition t_i's output functions' abnormal rates (= token value), which are shown in Table 3. For example, the output function's token value from transition t_1 to place p_f is 1-0.94(=1-0.98*0.99*0.97=0.941094).

Definition 9 Abnormal objects are objects that have a value less than a certain threshold value.

Table 3 Output functions' token values

Transition	Place	Output function's token value
t_1	p_f	1-0.94=0.06
t_2	p_f	1-0.94=0.06
t_3	p_f	1-0.94=0.06
t_4	p_f	1-0.94=0.06
t_5	p_f	1-0.97=0.03
t_6	p_f	1-0.94=0.06
t_7	p_f	1-0.97=0.03
t_8	p_f	1-0.94=0.06
t_9	p_f	1-0.94=0.06

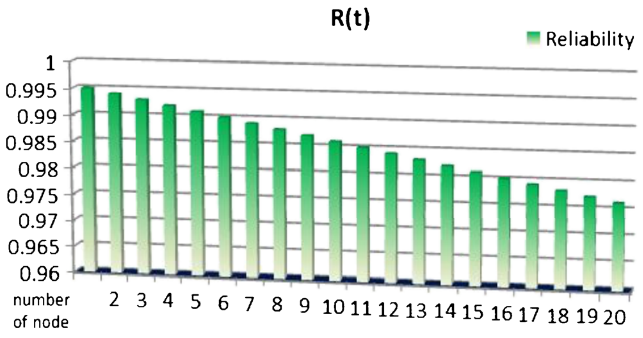


Fig. 3 Simulation result of Property 1

Next, the P2P network that consists of nodes and edges is dynamically operational. All programs in the P2P network must operate without abnormalities. We therefore define a P2P network system's reliability with Definition 10.

Definition 10 P2P system reliability is

$$R(system) = \prod_{i=1}^{i=n} R(ti) \times f(ti)$$

where f(t_j) is the probability that transition t_i is executed for the P2P network system, and n is the number of transitions.

$$\sum_{i=1}^{i=n} f(ti) = 1, 0 \leq f(ti) \leq 1$$

For example, in Table 3, by Definition 10, we can estimate that the probability that all programs in the P2P network are normal is 0.64.

4 Simulations

4.1 Property

In order to prove that the reliability analysis method of the colored Petri net model is effective, we assert six properties based on a P2P network model. In general, we assert that the extended P2P network model consists of a PSGM, an RST, and an ODS. As these model's attributes are added, the abnormality probability of the P2P network model varies. If a node or edge is added to the P2P network model, the reliability of the PSGM decreases. Then, we derive Property 1 and Property 2. A program or datum is added to the P2P network model. The reliability of the PSGM decreases. Then, we derive Property 3, Property 4, Property 5, and Property 6.

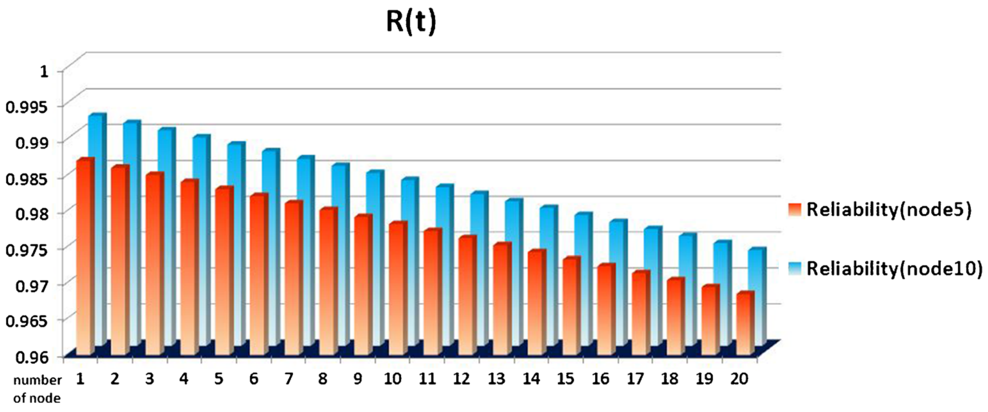
Now, we assert these properties.

Property 1. If we add a node to the PSGM, the reliability of the PSGM decreases.
Property 2. If we add an edge to the PSGM, the reliability of the PSGM decreases.
Property 3. If we add a program to the RST, the reliability of the PSGM decreases.
Property 4. If we add a datum to the RST, the reliability of the PSGM does not increase.
Property 5. If we add a program to the ODS, the reliability of the PSGM decreases.
Property 6. If we add a datum to the ODS, the reliability of the PSGM decreases.

4.2 Simulation result

In this section, we explain the result of the simulation. In order to model the reliability analysis of the P2P network model, we assume that the abnormal rate of all nodes is 0.001, the abnormal rate of all edges is 0.001, and the abnormal rate of programs is 0.001.

Fig. 4 Simulation result of Property 2



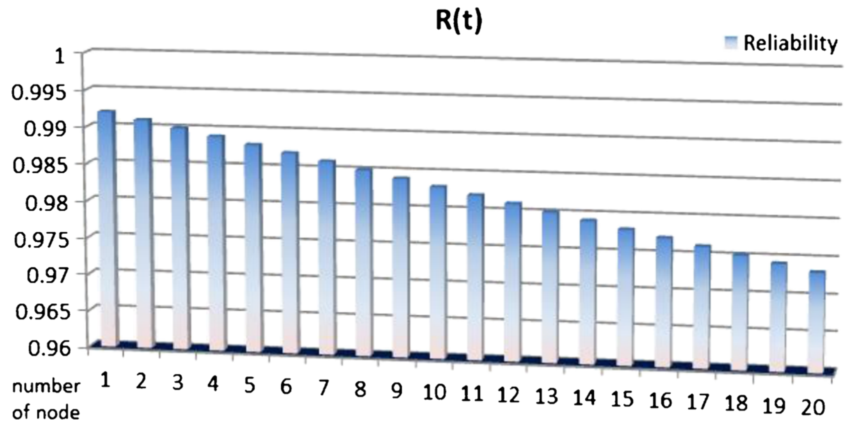


Fig. 5 Simulation result of Property 3

At first, we run a simulation in order to test whether Property 1 is satisfied or not. We define the average number of programs that exist in the system as the number of nodes times two. We can assume that the distribution of interactions between programs on the other computers is normal with a mean of 0.5 and a range of [0, 1]. We add each node to the P2P network from 1 to 20. We can assume that the distribution of programs is normal with a mean of 2, a standard deviation of 1, and a range of [0, 5]. We can assume that the distribution of data files is normal with a mean of 2, a standard deviation of 1, and a range of [0, 5]. We ran the simulation 100 times under the above conditions. The result is shown in Fig. 3. This chart shows that if we add a node to the PSGM, the reliability decreases. We conclude that Property 1 is satisfied.

Second, we run a simulation in order to test whether Property 2 is satisfied or not. If we add an edge to the PSGM under Property 2, the number of interactions between programs increases. We define the average number of programs that exist in the system as the number of nodes times two. We can assume that the average frequency of interactions between programs on the other computers is 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, and 0.9.

We ran simulations with the number of nodes at 5, 10, 15, and 20. We can assume that the distribution of programs is normal with a mean of 2, a standard deviation of 1, and a range of [0, 5]. We can assume that the distribution of data is normal with a mean of 2, a standard deviation of 1, and a range of [0, 5]. We ran the simulation 100 times under the above conditions. The result is shown in Fig. 4. This chart shows that if we add an edge to the PSGM, the reliability decreases. We conclude that Property 2 is satisfied.

Third, we ran a simulation in order to test whether Property 3 is satisfied or not. If we add a program to the RST under Property 3, the number of programs in the system increases. We can assume that the average frequency of interactions between programs on the other computers is 0.5. We ran simulations with the number of nodes at 5. We can assume that the distribution of data is normal with a mean of 2, a standard deviation of 1, and a range of [0, 5]. We ran the simulation 100 times under the above conditions. The result is shown in Fig. 5. This chart shows that if we add a program to the RST, the reliability of the PSGM decreases. We show that Property 3 is satisfied.

Fourth, we ran a simulation in order to test whether Property 4 is satisfied or not. If we add a datum to the RST under Property 4, the number of programs that need the datum in the system increases. We can assume that the average frequency of interactions between programs on the other computers is 0.5. We ran simulations

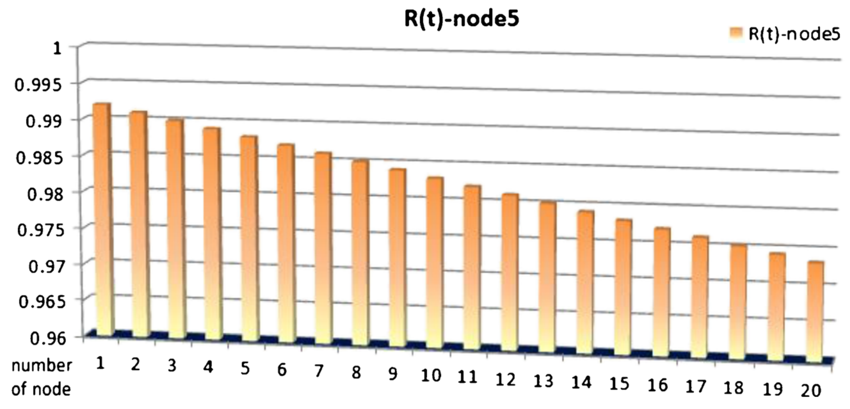


Fig. 6 Simulation result of Property 4

where the number of nodes is 5 and 10. We can assume that the distribution of programs is normal with a mean of 2, a standard deviation of 1, and a range of [0, 5]. We ran the simulation 100 times under the above conditions. The result is shown in Fig. 6. This chart shows that if we add data to the RST, the reliability of the PSGM decreases. We show that Property 4 is satisfied.

Fifth, we ran a simulation like the one for Property 3 in order to test whether Property 5 is satisfied or not. Finally, we ran a simulation like the one for Property 4 in order to test whether Property 6 is satisfied or not.

5 Conclusions

In this paper, we propose an estimation method for abnormal objects and for reliability in a P2P network. A colored Petri net is used as the analysis tool, and is useful for representing P2P system behaviors in order to estimate abnormal objects and the reliability of a P2P network. The procedure for analyzing system reliability and estimating abnormal objects is as follows. At first, a P2P network model is described with a general graph model, which is the P2P static graph model. Second, the resource sharing table and object dependence set must be constructed by the P2P network. Third, a colored Petri net is constructed with the proposed algorithm. Finally, we estimate abnormal objects and system reliability of the P2P network. We validate the proposed reliability model by simulation. So, the proposed reliability estimation method and abnormal objects prediction method for a P2P network is effective. The results of this study offer advantages where, while managing a P2P network, the reliability of the system and any abnormal objects can be predicted. The disadvantage in this study is finding all data and programs that run under the P2P network, and assuming the probability of abnormalities for each program and all the data.

Acknowledgments This work was supported by the Gachon University research fund of 2014. (GCU-2014-R017)

References

1. Schollmeier R (2002) “A Definition of Peer-to-Peer Networking for the Classification of Peer-to-Peer Architectures and Applications”, Proceedings of the First International Conference on Peer-to-Peer Computing, IEEE
2. Filali I et al (2011) A Survey of Structured P2P Systems for RDF Data Storage and Retrieval. In: Hameurlain A et al (eds) Transactions on Large-Scale Data- and Knowledge-Centered Systems III: Special Issue on Data and Knowledge Management in Grid and PSP Systems. Springer, p 21
3. Zulhasnine M et al (2013) P2P Streaming Over Cellular Networks: Issues, Challenges, and Opportunities. In: Pathan et al (eds) Building Next-Generation Converged Networks: Theory and Practice. CRC Press, p 99
4. Chervenak A, Bharathi S (2008) Peer-to-peer Approaches to Grid Resource Discovery. In: Danelutto M et al (eds) Making Grids Work: Proceedings of the CoreGRID Workshop on Programming Models Grid and P2P System Architecture Grid Systems, Tools and Environments 12–13 June 2007, Heraklion, Crete, Greece. Springer, p 67
5. Jin X, Chan S-HG (2010) Unstructured Peer-to-Peer Network Architectures. In: Shen et al (eds) Handbook of Peer-to-Peer Networking. Springer, p 119
6. Lv Q et al (2002) Can Heterogenity Make Gnutella Stable? In: Druschel P et al (ed) Peer-to-Peer Systems: First International Workshop, IPTPS 2002, Cambridge, MA, USA, March 7–8, 2002, Revised Papers. Springer, p 94
7. Chung KY (2013) Recent Trends on Convergence and Ubiquitous Computing. Pers Ubiquit Comput. doi:10.1007/s00779-013-0743-2
8. Kim JY, Chung KY, Jung JJ (2014) Single tag sharing scheme for multiple-object RFID applications. Multimedia Tools Appl 68(2): 465–477
9. Oh SY, Chung KY (2013) Target Speech Feature Extraction Using Non-Parametric Correlation Coefficient. Clust Comput. doi:10.1007/s10586-013-0284-5
10. Kim GH, Kim YG, Chung KY (2013) Towards Virtualized and Automated Software Performance Test Architecture. Multimedia Tools Appl. doi:10.1007/s11042-013-1536-3
11. Kim SH, Chung KY (2014) 3D simulator for stability analysis of finite slope causing plane activity. Multimedia Tools Appl 68(2):455–463
12. Ko JW, Chung KY, Han JS (2013) Model transformation verification using similarity and graph comparison algorithm. Multimedia Tools Appl. doi:10.1007/s11042-013-1581-y
13. Han JS, Chung KY, Kim GJ (2013) Policy on literature content based on software as service. Multimedia Tools Appl. doi:10.1007/s11042-013-1664-9
14. Kang SK, Chung KY, Lee JH (2014) Development of head detection and tracking systems for visual surveillance. Pers Ubiquit Comput 18(3):515–522
15. Baek SJ, Han JS, Chung KY (2013) Dynamic reconfiguration based on goal-scenario by adaptation strategy. Wirel Pers Commun 73(2): 309–318
16. Ha OK, Song YS, Chung KY, Lee KD, Park D (2014) Relation model describing the effects of introducing RFID in the supply chain: evidence from the food and beverage industry in South Korea. Pers Ubiquit Comput 18(3):553–561
17. Dilum Bandara HMN, Jayasumana Anura P (2013) 276 Collaborative applications over peer-to-peer systems–challenges and solutions. Peer-to-Peer Networking and Applications 6(3)
18. Darlagiannis V (2005) Hybrid Peer-to-Peer Systems. In: Steinmetz R, Wehrle K (ed) Peer-to-Peer Systems and Applications. Springer
19. Jensen K (1994) An Introduction to the Theoretical Aspects of Coloured Petri Nets. In: de Bakker JW, de Roever W-P, Rozenberg G (eds) A Decade of Concurrency, Lecture Notes in Computer Science, vol. 803. Springer-Verlag, pp 230–272
20. Jensen K (1991) Coloured Petri nets: a high-level language for system design and analysis. In Rozenberg G (ed) Advances in Petri nets 1990, lecture notes in computer science, vol. 483. Springer-Verlag, pp 342–416. Also in Jensen K, Rozenberg G (eds) High-level Petri Nets. Theory and Application, pp 44–122
21. Jung H, Chung KY (2013) Discovery of automotive design paradigm using relevance feedback. Pers Ubiquit Comput. doi:10.1007/s00779-013-0738-z

22. Huber P, Jensen K, Shapiro RM (1991) Hierarchies in coloured Petri nets. In: Rozenberg G (ed) Advances in Petri nets 1990, Lecture Notes in Computer Science, vol. 483. Springer-Verlag, pp 313–341. Also in Jensen K, Rozenberg G (eds) High-level Petri Nets. Theory and Application, pp 215–243
23. Berger J, Lamontagne L (1993) A colored Petri net model for a naval command and control system. In: Ajmone-Marsan M (ed) Application and theory of Petri nets 1993. Proceedings of the 14th International Petri Net Conference, Chicago 1993, Lecture Notes in Computer Science, vol. 691. Springer-Verlag, pp 532–541
24. Mortensen KH (1996) Modelling and analysis of distributed program execution in BETA using coloured Petri nets. In: Billington J, Reisig W (eds) Application and theory of Petri nets 1996. Proceedings of the 17th International Petri Net Conference, Osaka 1996, Lecture Notes in Computer Science, vol. 1091, Springer-Verlag, pp 249–268
25. McLendon WW, Vidale RF (1992) Analysis of an Ada system using colored Petri nets and occurrence graphs. In: Jensen K (ed) Application and theory of Petri Nets 1992. Proceedings of the 13th International Petri Net Conference, Sheffield 1992, Lecture Notes in Computer Science, vol. 616. Springer-Verlag, pp 384–388
26. Peterson JL (1981) Petri net theory and the modeling of systems. Prentice-Hall, Englewood Cliffs
27. Chiola G, Marsan MA, Balbo G, Conte G (1993) Generalized stochastic Petri nets: a definition at the net level and its implications. IEEE Trans Softw Eng 19(2):89–107

Kapsu Kim has received B.S., M.S., and Ph.D. degrees in 1985, 1987, and 1995, respectively, all from the Department of Computer Science, Seoul National University, Seoul, Korea. He has worked as a researcher in Samsung Electronics from 1987 to 1991. He is currently a professor of Department of Computer Education, SNUE(Seoul National University of Education), Seoul Korea. His research interests include Software Engineering, Computer Education, and Gifted Children Education. He is an editor chairman of the Journal of Korean Information Science.

Myunghui Hong is currently a professor of Department of Computer Education, SNUE(Seoul National University of Education). He received B.S. (1984) in Computer Science at Kwangwoon University, an M.S.(1986) in computer Science at KAIST(Korea Advanced Institute of Science and Technology), and a Ph. D.(1994) in Computer Science at Kwangwoon University, Korea. He was a Visiting Scholar of IST(Instructional Systems Technology) at Indiana University, and METU(Middle East Technical University) in Turkey, in 2004 and 2013, respectively. He has worked as a researcher in KT(Korea Telecom) from 1986 to 1991. His research interests include computer education contents providing and making teaching environments using new technology, tiny computer which are based mobile environment.

Kyungyong Chung has received B.S., M.S., and Ph.D. degrees in 2000, 2002, and 2005, respectively, all from the Department of Computer Information Engineering, Inha University, Korea. He has worked for Software Technology Leading Department, Korea IT Industry Promotion Agency (KIPA). He is currently a professor in the School of Computer Information Engineering, Sangji University, Korea. His research interests include Medical Data Mining, Healthcare, Knowledge System, HCI, and Recommendation.

SangYeob Oh received a BS in Computer Science from Kyungwon University in 1989 and an MS and a PhD from Kwangwoon University, Korea, in 1991 and 1999, respectively, from the Department of Computer Science. He is currently a professor in the Division of Computer Media, Gachon University, Korea. His research interests include Speech Recognition, Vehicle Safety Communications, and HCI. He has edited 27 books of computer science. He serves as Executive Editing Director of International Conference on Digital Policy Management, Executive Editing Director, Steering Committees of International Conference on Convergence Technology. Also, he is an editorial member of the International Journal of Wireless Personal Communications, and so on.