The United Nations
University

UNU/IIST

**International Institute for
Software Technology**

Verifying Linear Duration Properties of Probabilistic Real-Time Systems

Hou Jianmin and Dang Van Hung

February 1999

UNU/IIST Report No. 155

UNU/IIST and UNU/IIST Reports

UNU/IIST is a Research and Training Center of the United Nations University. It was founded in 1992, and is located in Macau. UNU/IIST is jointly funded by the Governor of Macau and the Governments of China and Portugal through contribution to the UNU Endowment Fund.

The mission of UNU/IIST is to assist developing countries in the application and development of software technology.

UNU/IIST contributes through its programmatic activities:

1. advanced development projects in which software techniques supported by tools are applied,
2. research projects in which new techniques for software development are investigated,
3. curriculum development projects in which courses of software technology for universities in developing countries are developed,
4. courses which typically teach advanced software development techniques,
5. events in which conferences and workshops are organised or supported by UNU/IIST, and
6. dissemination, in which UNU/IIST regularly distributes to developing countries information on international progress of software technology.

Fellows, who are young scientists and engineers from developing countries, are invited to actively participate in all these projects. By doing the projects they are trained.

At present, the technical focus of UNU/IIST is on formal methods for software development. UNU/IIST is an internationally recognised center in the area of formal methods. However, no software technique is universally applicable. We are prepared to choose complementary techniques for our projects, if necessary.

UNU/IIST produces a report series. Reports are either Research R, Technical T, Compendia C or Administrative A. They are records of UNU/IIST activities and research and development achievements. Many of the reports are also published in conference proceedings and journals.

Please write to UNU/IIST or visit UNU/IIST home page: http://www.iist.unu.edu, if you would like to know more about UNU/IIST and its report series.

Zhou Chaochen, Director — 01.8.1997 – 31.7.2001

The United Nations University

UNU/IIST

International Institute for Software Technology

P.O. Box 3058
Macau

Verifying Linear Duration Properties of Probabilistic Real-Time Systems

Hou Jianmin and Dang Van Hung

Abstract

In this paper we present a method for deciding whether a probabilistic real-time system, modelled as a Markov chain, satisfies a linear duration property at a given time interval with a given lower bound of the probability. With this method we can reduce the problem into a finite number of integer linear constraint systems, which have been well studied. Therefore, the problem can be solved with a polynomial-time algorithm.

Hou Jianmin is a Ph.D. student in the Department of Computer Science and Technology, Nanjing University, Nanjing, Jiangsu, China. He is a Fellow of UNU/IIST from September 1998 to February 1999. His research interests are Formal Methods, Model Checking Real-time Systems and Probabilistic Real-time Systems. E-mail: hjm@iist.unu.edu

Dang Van Hung is from the Institute of Information Technology of National Center for Natural Science and Technology of Vietnam, where he is a researcher. He was Fellow of UNU/IIST from April 1994 till July 1995. He becomes a Research Fellow of UNU/IIST since Octomber 1995. His research interests include Formal Techniques of Programming, Concurrent and Distributed Systems. E-mail: dvh@iist.unu.edu

Contents

1 Introduction

Model checking has been used successfully as a verification technique in the fields of hardware, communication protocol and software design. Model checking for a system involves two inputs: a requirement for the system and a model of the implementation (system abstraction) of the system, and gives answer to the question "does the system satisfy its specification?".

In practice, we cannot expect that every system satisfies its specification perfectly at any time because of possible failures of its components. Therefore, the study of the system dependability is very important. It gives us more information about a system and let us know with how much confidence we can trust the system. The dependability requirement of a system expresses that the probability for undesirable but unavoidable behaviour must be below a certain limit. Model checking the dependability of a system can be formulated as follows: Given a specification, given an operation time and a specified probability, given a system model, verify whether the model satisfies its specification within the operation time with the probability not lower than the specified one.

Most of the earlier papers on probabilistic model checking only dealt with either linear time propositional temporal logic [3, 14] or branching time temporal logic [1, 2]. In those papers, the system requirements are written as Temporal Logic formulas, and the probabilistic model checking is reduced to non-probabilistic one. For the case of linear time propositional temporal logic, in the papers [1, 2] the authors gave an algorithm to solve the following problem: given a Markov chain $\mathcal{M}$ and a linear time Temporal Logic formula ψ, decide whether the set of those transition sequences of $\mathcal{M}$ which satisfy ψ has the probability measure one. In the paper [9], the author introduced $PCTL^*$ by enriching CTL^* [7], a branching time logic, with a probabilistic quantifier $Æ_p$ ($Æ_p\psi$ denotes that ψ holds with probability at least p), and gave an algorithm for checking a probabilistic program against its specification in $PCTL^*$. A real-time property to be verified in that paper is of the form: if the failure rate of transitions is less than 5% then the probability that every message sent by the sender will reach the receiver in less than 15 steps (i.e. clock ticks) is greater than 95%. The time-complexity of the decision procedure presented in [9] for the verification is very high, and seems to be infeasible in practice.

In this paper, we shall deal with the same kind of problems but we give a different approach to checking the dependability and obtain a much better results. We also use Markov chain to model a probabilistic real-time system, but we use simple Duration Calculus formula as the system requirement, and try to give a simple algorithm to solve the problem. Namely, we will give a technique for checking whether a Markov chain satisfies a linear constraint on the durations of its states at a given time interval with the probability not lower then a given number. Using our technique, we can reduce the problem into several systems of integer linear constraints, which have been well studied, and therefore we can give a polynomial time algothims to solve the problem.

There have been several work on model-checking for Duration Calculus, such as [8, 15, 12]. But these technique could not be used in this context. A technique for deductive reasoning about

the dependability using Probability Duration Calculus has been developed in [5] and [13] which can be considered to be complementary to our work.

2 Model Checking for Probabilistic Real-Time Systems

In this section we give a formal description of model checking for Probabilistic Real-Time Systems. We use Markov chain as the abstract model for probabilistic real-time system, and Linear Duration Calculus properties as its requirements. For the comfort of the readers who are not familiar with Duration Calculus, we also give in this section a definition for the satisfaction probability of a linear duration property by a Markov chain.

A *stochastic process* [4] evolves in time and space in accordance with probabilistic laws. We consider in this paper a real-time system as a stochastic process with the discrete time which can have some possible positions at a time n (n is a natural number, $n \in \mathcal{N}$). Each possible position of the system at a time is called a *state* of the system, and the set of all states of the system is call the *state space*.

A stochastic process is a Markov process if for arbitrary time $(n-1)$, and n, $(n \in \mathcal{N})$), the possibility of the process changing its position from time $(n-1)$ to n depends only its positions at $(n-1)$ and n, and never refers to any other time points. Formally, we give a definition of Markov chain as follows.

Definition 1 *A Markov chain $\mathcal{M}$ is a pair (Q, P), where*

1. *Q is a finite set of states, enumerated as $q_1, q_2, q_3, \ldots$,*
2. *$P : (Q \cup \{q_0\}) \times Q \to [0,1]$ is a probabilistic transition function ($q_0 \notin Q$).*

The function P satisfies that for any $q \in (Q \cup \{q_0\})$

$$\sum_{q' \in Q} P(q, q') = 1$$

The state q_0 is a virtual state and is introduced just for the simplicity of the denotation. The value $P(q_0, q)$ expresses the probability that the process starts at state q, and the value $P(q, q')$ expresses the probability that the process moves to the state q' for the next moment, given that the process is currently at state q.

Given a Markov chain $\mathcal{M}$, we can define a sequence space $\Theta(\mathcal{M}) = (\Omega, \mathcal{F}, \mu)$, where

- $\Omega = Q^\omega$ is the set of all infinite sequences of states of $\mathcal{M}$,

- $\mathcal{F} : Q^* \to 2^{Q^\omega}$ is a Borel field generated by the cylindric sets

  $$\mathcal{F}(q_1 \dots q_n) = \{\pi \in \Omega \mid \pi = q_1 \dots q_n \dots\}$$

- μ is a probability measure function defined as

  $$\mu(\mathcal{F}(q_1 \dots q_n)) = p_{0,1} * p_{1,2} * \dots * p_{n-1,n}$$

  where $p_{i-1,i} = P(q_{i-1}, q_i),\ 1 \leq i \leq n$.

So, each set $\mathcal{F}(q_1 \dots q_n)$ expresses the fact that in the interval of time $[0, n]$ the system will travel from q_1 to q_n via states $q_2, \dots q_{n-1}$, and stays in each of the states for one time unit. Hence, $\mu(\mathcal{F}(q_1 \dots q_n))$ is the probability that the system will behave as the sequence $q_1 \dots q_n$ for the first n time units.

Now let a Markov chain $\mathcal{M}$ be given. Let $\mathcal{D}$ be a predicate on the Cartesian product of the set Ω of the (infinite) behaviours of $\mathcal{M}$ and the set $\mathcal{N}$ of natural numbers (representing time). $\mathcal{D}(\sigma, t) = \mathit{true}$ means that the property $\mathcal{D}$ is true for the behaviour σ at time t, and this fact will be denoted by $\sigma, t \models \mathcal{D}$ for our convenience. Let

$$S^t(\mathcal{D}) \mathrel{\widehat{=}} \{\sigma \mid \sigma, t \models \mathcal{D}\}$$

Assume that $S^t(\mathcal{D})$ is μ-measurable. Then $\mu(S^t(\mathcal{D}))$ will be the probability that $\mathcal{M}$ satisfies $\mathcal{D}$ at time t.

Note that a Markov chain $\mathcal{M}$ can be represented as a finite probabilistic automaton. Figure 1 depicts a finite probabilistic automaton which models a simple gas-burner. There are two states s_1 and s_2 in the automaton which express *non-leak* state and *leak* state of gas respectively, and four transitions to express the evolution of the system. Let $p_{0,1} \mathrel{\widehat{=}} P(s_0, s_1)$ and $p_{0,2} \mathrel{\widehat{=}} P(s_0, s_2)$ be the probability for the system to start at s_1 and s_2 respectively. Let $p_{1,1} \mathrel{\widehat{=}} P(s_1, s_1)$, $p_{1,2} \mathrel{\widehat{=}} P(s_1, s_2)$, $p_{2,2} \mathrel{\widehat{=}} P(s_2, s_2)$ and $p_{2,1} \mathrel{\widehat{=}} P(s_2, s_1)$ be the probability for the transitions.

For the simplicity, in this paper, we overload the symbol $p_{i,j}$ $(i, j \neq 0)$ to express not only the probability (a number between 0 and 1), but also the transition from state q_i to state q_j. Similarly, $p_{0,i}$ expresses not only the probability for the system to start at q_i, but also the initial (virtual) transition entering state q_i. A transition $p_{k,m}$ is said to be a successive one of a transition $p_{i,j}$ iff $j = k$. According to our convention, $\sigma \mathrel{\widehat{=}} p_{0,i_1} p_{i_1,i_2} \dots p_{i_{k-1},i_k}$ is a consecutive sequence of transitions corresponding to the state sequence $q_{i_1} q_{i_2} \dots q_{i_k}$, while $p_{0,i_1} * p_{i_1,i_2} * \dots * p_{i_{k-1},i_k}$ is a number and, in this case, the probability that the system takes σ as its behaviour for the first k time units. Note that any sequence of states corresponds to exactly one consecutive sequence of transitions with the same length in the obvious way and vice-versa. Hence, from now on, we can use two of them interchangeably for a behaviour.

One of the requirement for the gas-burner is that if it is observed for more than 60 time units then the accumulated time for gas-leaking is not more than one twentieth of the observation

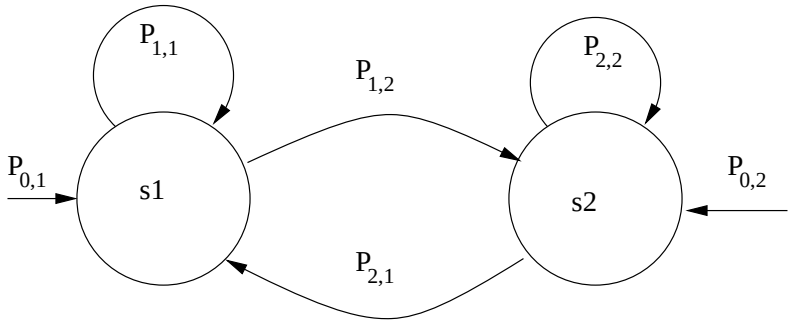

Figure 1: The state diagram of Gas-Burner with probability

time (see [15]). In Duration Calculus the accumulated time for the present of a state q in an interval of time is called duration of the state q over the interval, and is denoted as $\int q$ when the interval is understood from the context. For example, let $p_{0,1}p_{1,1}p_{1,2}p_{2,2}$ be a finite behaviour of the automaton in Figure 1 in the (discrete) time interval $[0, 4]$. For this behaviour and this time interval, $\int s_1$ is 2, and $\int s_2$ is also 2 (this means that the accumulated time for gas leaking is 2 time units for this behaviour). We define the duration of a state over a sequence of transitions as follows.

Definition 2 *Given a finite sequence of transitions $\sigma \mathrel{\widehat{=}} p_{i_1,j1}p_{i_2,j_2} \ldots p_{i_k,j_k}$ (not necessary to be consecutive) of a Markov chain $\mathcal{M}$, the duration of state q_j ($\int q_j$) over σ is defined as $\int q_j(\sigma) \mathrel{\widehat{=}} \sum_{t=1}^{k} \delta(j_t, j)$, where*

$$\delta(j_t, j) = \begin{cases} 1 & \text{if } j_t = j \\ 0 & \text{otherwise} \end{cases}$$

(So, $\int q_j(\sigma)$ is the accumulated time that the system stays in state q_j when travelling along σ which is the number of times it enters q_j.)

In this paper, we are interested in the following kind of the predicates $\mathcal{D}$, which are called Linear Duration Properties:

$$(K1) \quad t_1 \leq \ell \leq t_2 \Rightarrow \sum_{i=1}^{m} c_i * \int q_i \leq M$$
$$(K2) \quad \boxdot(\sum_{i=1}^{m} c_i * \int q_i \leq M)$$

where t_1, t_2 are nonnegative integers, and c_i's and M are integers. (ℓ is a term to be evaluated to the length of the reference interval, see below.)

To our experience, linear duration properties form an important class of Duration Calculus formulas in specifying requirements of real-time systems, and the problem of checking a real-

time system for a linear duration property has attracted a great deal of attention (see, e.g. [10, 15, 12, 6]).

A predicate $\mathcal{D}$ of the form $(K1)$ is evaluated to true for a finite sequence of transitions σ iff whenever the length $|\sigma|$ (the number of elements) of σ satisfies the premise $t_1 \leq |\sigma| \leq t_2$, the durations of the state of the process satisfy the inequality $\sum_{i=1}^{m} c_i * \int q_i \leq M$. A predicate $\mathcal{D}$ of the form $(K2)$ is evaluated to true for a finite sequence of transitions σ iff for any prefix σ' of σ the durations of the state of the process satisfy the inequality $(\sum_{i=1}^{m} c_i * \int q_i \leq M)$. When a predicate $\mathcal{D}$ is evaluated to true for a finite sequence of transitions σ, we say that σ satisfies $\mathcal{D}$ and write $\sigma \models \mathcal{D}$. For example, for the system in Fig, 1, the sequence $p_{0,1}p_{1,1}p_{1,2}p_{2,2}$ satisfies $\int s_1 - \int s_2 \leq 0$ (because both $\int s_1$ and $\int s_2$ are evaluated to 2 over the sequence).

For an infinite sequence σ, let us denote by $\sigma_{[t]}$ the prefix of σ with length t.

Definition 3 *Given a Markov chain $\mathcal{M}$, given a Linear Duration Property $\mathcal{D}$, given a time t, let $\sigma \in \Omega$ be an infinite behaviour of $\mathcal{M}$. The behaviour σ satisfies formula $\mathcal{D}$ at time t iff $\sigma_{[t]} \models \mathcal{D}$.*

Now, we can formulate our model checking problem for discrete probabilistic real-time systems as follows.

Given a Markov chain $\mathcal{M}$, given a Linear Duration Property $\mathcal{D}$, given a probability p, decide whether $\mathcal{M}$ satisfies $\mathcal{D}$ at time t with the probability not lower than p. In other words, verify $\mu(S^t(\mathcal{D})) \leq p$.

Note that

$$\mu(S^t(\mathcal{D})) = \sum_{p_{0,i_1} \cdots p_{i_{t-1},i_t} \models \mathcal{D}} p_{0,i_1} * p_{i_1,i_2} * \ldots * p_{i_{t-1},i_t}$$

and it is decidable that $p_{0,i_1}p_{1_1,i_2} \ldots p_{i_{t-1},i_t} \models \mathcal{D}$, so the problem is decidable. However, the complexity of the computation using the definition is so high. In this paper, we look for a better algorithm to solve the problem.

3 Techniques for Improvement

In this section, we give some much simpler techniques for checking a Markov chain $\mathcal{M}$ for a linear duration property $\mathcal{D}$.

Given a time t, in order to calculate the satisfaction probability of the linear duration property $\mathcal{D}$ by $\mathcal{M}$ at time t, we have to consider only those behaviours whose prefixes with the length

t have no occurrence of a transition $p_{i,j}$ with the probability $p_{i,j} = 0$. Let W be a regular expression over the set $\Sigma \mathrel{\widehat{=}} \{p_{i,j} \mid p_{i,j} \neq 0 \text{ and } i,j \leq n\}$ expressing all possible prefixes with no occurrence of a transition with the probability 0 of the behaviours of $\mathcal{M}$. In practice, W is much smaller than the set of all prefixes of the behaviours. The set of all prefixes with length t that we are interested in is then

$$\mathcal{R} \quad \widehat{=} \quad \Sigma^t \cap W$$

Now, assume that $\mathcal{R}$ can be written as

$$(1) \quad \mathcal{R} \quad = \quad \mathcal{R}_1 \bigoplus \ldots \bigoplus \mathcal{R}_k$$

such that

- k is relatively small (in comparison to $|\Sigma|^t$),
- $\mathcal{R}_i \cap \mathcal{R}_j = \emptyset$ for $i \neq j$, and
- it is easy to calculate the probability $P_i \mathrel{\widehat{=}} \mu({\mathcal{S}_i}^t(\mathcal{D}))$, where ${\mathcal{S}_i}^t(\mathcal{D}) = \{\sigma \mid \sigma_{[t]} \models \mathcal{D} \wedge \sigma_{[t]} \in \mathcal{R}_i\}$, $1 \leq i \leq k$.

Then, the problem can be solved easily by simply checking whether $\sum_{i=1}^{k} P_i \geq p$.

To illustrate our idea, let us consider the simple gas burner in Figure 1. Assume that the probability for the system to start at S_2 is zero, i.e. $p_{0,2} = 0$, $p_{0,1} = 1$, and the probability for other transitions are non-zero. Then, the regular expression W expressing all the prefixes of the behaviours of Gas Burner with no occurrence of $p_{0,2}$ is the following.

$$\begin{array}{rcl} W & = & p_{0,1}(p_{1,1}^* p_{1,2} p_{2,2}^* p_{2,1})^* p_{1,1}^* \bigoplus \\ & & p_{0,1}(p_{1,1}^* p_{1,2} p_{2,2}^* p_{2,1})^* p_{1,1}^* p_{1,2} p_{2,2}^* \end{array}$$

Let $t = 6$, $\mathcal{D} \mathrel{\widehat{=}} 2 * \int S_1 - \int S_2 \leq 0$. Then $\mathcal{R} \mathrel{\widehat{=}} \Sigma^6 \cap W$ can be written as $\mathcal{R}_1 \bigoplus \ldots \bigoplus \mathcal{R}_6$, where

$$\begin{array}{rcl} \mathcal{R}_1 & = & \{p_{0,1}(p_{1,1}^{k_1} p_{1,2} p_{2,2}^{k_2} p_{2,1}) p_{1,1}^{k_3} \mid \\ & & \quad k_1 + k_2 + k_3 = 3\} \\ \mathcal{R}_2 & = & \{p_{0,1}(p_{1,1}^{k_1} p_{1,2} p_{2,2}^{k_2} p_{2,1})(p_{1,1}^{k_3} p_{1,2} p_{2,2}^{k_4} p_{2,1}) p_{1,1}^{k_5} \mid \\ & & \quad \sum_{j=1}^{5} k_i = 1\} \\ \mathcal{R}_3 & = & \{p_{0,1}(p_{1,1}^{k_1} p_{1,2} p_{2,2}^{k_2} p_{2,1}) p_{1,1}^{k_3} p_{1,2} p_{2,2}^{k_4} \mid \\ & & \quad \sum_{j=1}^{4} k_i = 2\} \\ \mathcal{R}_4 & = & \{p_{0,1} p_{1,2} p_{2,1} p_{1,2} p_{2,1} p_{1,2}\} \\ \mathcal{R}_5 & = & \{p_{0,1} p_{1,1}^5\} \\ \mathcal{R}_6 & = & \{p_{0,1} p_{1,1}^{k_1} p_{1,2} p_{2,2}^{k_2} \mid k_1 + k_2 = 4\} \end{array}$$

where each k_i is a non-negative integer.

It is easy to see that $\mathcal{R}_i \cap \mathcal{R}_j = \emptyset$ for $i \neq j$. Let us take the set $\mathcal{R}_3$ and calculate the probability P_3. For a sequence of the form $p_{0,1}(p_{1,1}^{k_1} p_{1,2} p_{2,2}^{k_2} p_{2,1}) p_{1,1}^{k_3} p_{1,2} p_{2,2}^{k_4}$, by the definition of durations we have $\int S_1 = 1 + k_1 + 1 + k_3$, $\int S_2 = 1 + k_2 + 1 + k_4$. Therefore, the set of all sequences in $\mathcal{R}_3$ satisfying $\mathcal{D}$ is

$$\begin{array}{l}\{p_{0,1}(p_{1,1}^{k_1} p_{1,2} p_{2,2}^{k_2} p_{2,1}) p_{1,1}^{k_3} p_{1,2} p_{2,2}^{k_4} \mid \\ \quad \sum_{j=1}^{4} k_i = 2 \text{ and} \\ \quad 2 * (k_1 + k_3) - k_2 - k_4 + 2 \leq 0\}.\end{array}$$

Because each sequence in the set corresponds to exactly one solution of the integer constraint, the probability P_3 can be calculated as

$$\begin{array}{rcl} P_3 & = & (p_{1,2})^2 * p_{2,1} * \\ & & \quad \sum_{\substack{f=2, \\ g \leq 0, \\ k_i \geq 0}} (p_{1,1})^{(k_1+k_3)} * (p_{2,2})^{(k_2+k_4)} \\ & = & 3 * (p_{1,2})^2 * (p_{2,2})^2 * p_{2,1} \end{array}$$

where $f \mathrel{\widehat{=}} k_1 + k_2 + k_3 + k_4$ and $g \mathrel{\widehat{=}} 2 * (k_1 + k_3) - k_2 - k_4 + 2$.

In the same way, we can calculate $P_1 = p_{1,2} * (p_{2,2})^3 * p_{2,1}$, $P_2 = 0$, $P_4 = p_{0,1} * (p_{1,2})^3 * (p_{2,1})^2$, $P_5 = 0$ and $P_6 = p_{0,1} * p_{1,2} * (p_{2,2})^4 + p_{0,1} * p_{1,1} * (p_{1,2}) * (p_{2,2})^3$. Therefore, the probability for the gas burner satisfying $\mathcal{D}$ at time 6 is $P = \sum_{i=1}^{6} P_i$.

Roughly speaking, with this technique we can reduce the model-checking problem in discrete probabilistic real-time systems into a number of the problems of solving linear integer constraint systems. The number of the problems is the same as the number of operator $\bigoplus$ in the representation (1), and the size of the linear integer constraint systems is the number of the occurrences of the operator * in each components, provided that there is no nested occurrence of the operator *.

Note that the calculation of the probability in the above example can be correct only if each sequence in the set $\mathcal{R}_i$ corresponds to exactly one integer solution of the corresponding constraints. We formalise this condition by the following definition. Namely, we define the notion of so-called well-formed regular expressions with the following property: each string in the language generated by a well-formed regular expression has the unique representation according to the structure of the expression.

For a regular expression $\mathcal{R}$, let *prefix*($\mathcal{R}$) (*posfix*($\mathcal{R}$)) denote the set of non-empty proper prefixes (postfixes) of the language represented by $\mathcal{R}$. As usual, we use $\mathcal{R}^+$ as a regular expression to

represent the language $\{\sigma \mid \sigma = \sigma_1 \dots \sigma_k, k \geq 1, \sigma_k \in \mathcal{R}\}$. Because every expression $\mathcal{R}^*$ can be written as $\mathcal{R}^+ \bigoplus \{\epsilon\}$, in the sequel, we will use the operator $^+$ instead of the operator * in the regular expressions for the simplicity of our presentation.

Definition 4 *Well-formed regular expressions (w.f. REs) are defined recursively as*

1. *For any non-empty sequence σ the expression $\mathcal{R} = \{\sigma\}$ is a well-formed regular expression.*
2. *If $\mathcal{R}_1$ and $\mathcal{R}_2$ are well-formed regular expressions satisfying that $(posfix(\mathcal{R}_1)\mathcal{R}_2) \cap \mathcal{R}_2 = \emptyset$ or $posfix(\mathcal{R}_2) \cap \mathcal{R}_2 = \emptyset$ or $prefix(\mathcal{R}_1) \cap \mathcal{R}_1 = \emptyset$ or $(\mathcal{R}_1 prefix(\mathcal{R}_2)) \cap \mathcal{R}_1 = \emptyset$ then $\mathcal{R}_1 \frown \mathcal{R}_2$ is a well-formed regular expression.*
3. *If $\mathcal{R}$ is a well-formed regular expression satisfying that $(posfix(\mathcal{R})\mathcal{R}) \cap \mathcal{R} = \emptyset$ or $(\mathcal{R} prefix(\mathcal{R})) \cap \mathcal{R} = \emptyset$ or $prefix(\mathcal{R}) \cap \mathcal{R} = \emptyset$ or $posfix(\mathcal{R}) \cap \mathcal{R} = \emptyset$ then $\mathcal{R}^+$ is a well-formed regular expression.*

For example, $(ab^+)^+$ is a w.f. RE, and each sequence σ in its language corresponds to exactly one tuple $(h_1, \dots, h_k)$ for which $\sigma = (ab^{h_1})(ab^{h_2}) \dots (ab^{h_k})$. Expression $(a^+)^+$ is not a w.f. RE, and a^3 can be written as $(a^2)(a)$ and as $(a)(a^2)$. However, a^+ is a w.f. RE. Expression $(a^+ba^+)^+$ is not a w.f. RE, and there are several ways of presenting the string aba^3ba according to the structure of $(a^+ba^+)^+$, such as $(aba)(a^2ba)$ and also $(aba^2)(aba)$. Note that if $(\mathcal{P}\mathcal{Q})\mathcal{R}$ is a w.f. RE then $\mathcal{P}(\mathcal{Q}\mathcal{R})$ is a w.f. RE too (the concatenation operator is associative for w.f. REs).

Let us denote by $\mathcal{L}(\mathcal{R})$ the language represented by a regular expression $\mathcal{R}$.

Proposition 1

1. *If $\mathcal{R} = \mathcal{R}_1\mathcal{R}_2$ is a w.f. RE, then for any $\sigma \in \mathcal{L}(\mathcal{R})$ there exist uniquely $\sigma_1 \in \mathcal{L}(\mathcal{R}_1)$ and $\sigma_2 \in \mathcal{L}(\mathcal{R}_2)$ such that $\sigma = \sigma_1\sigma_2$*
2. *If $\mathcal{R} = (\mathcal{R}_1)^+$ is a w.f. RE, then for any $\sigma \in \mathcal{L}(\mathcal{R})$ there exist uniquely an integer k and sequences $\sigma_1, \dots, \sigma_k \in \mathcal{L}(\mathcal{R})$ such that $\sigma = \sigma_1 \dots \sigma_k$*

Proof: The proof is easy by induction on the structure of w.f. RE and is omitted.

Now, we are ready to present our technique for computing the satisfaction probability of duration properties formally. The following theorems formalise our technique for the case that there is no nested occurrence of the operator $^+$ in the expression representing the behaviour of the system.

Theorem 1 *Given a Markov chain $\mathcal{M}$, given a w.f. RE $\mathcal{R}$ over the set of transitions with non-zero probability of $\mathcal{M}$, given a duration property $\mathcal{D}$ of the form $(K1)$, given a time t, we can construct a function $P(k_1, \ldots, k_l)$, linear functions $f(k_1, \ldots, k_l)$ and $g(k_1, \ldots, k_l)$ of integer variables $k_1, \ldots, k_l$ for which the probability $p \mathrel{\widehat{=}} \mu(\mathcal{S}^t(\mathcal{D}))$, where $\mathcal{S}^t(\mathcal{D}) = \{\sigma \mid \sigma_{[t]} \models \mathcal{D} \wedge \sigma_{[t]} \in \mathcal{R}\}$ is calculated as*

$$p = \sum_{\substack{k_1, \ldots, k_l \geq 1,\ f(k_1, \ldots, k_l) = t, \\ g(k_1, \ldots, k_l) \leq M,}} P(k_1, \ldots, k_l)$$

Proof: By induction on the structure of $\mathcal{R}$ using the definition of the satisfaction probability, the well-formedness of $\mathcal{R}$ and the construction technique shown in the previous example.

Theorem 2 *Given a Markov chain $\mathcal{M}$, given a w.f. RE $\mathcal{R}$ over the set of transitions with non-zero probability of $\mathcal{M}$, given a duration property $\mathcal{D}$ of the form $(K2)$, given a time t, we can construct a function $P(k_1, \ldots, k_l)$, linear functions $f(k_1, \ldots, k_l)$ and $g_j(k_1, \ldots, k_l)$, $(j = 1, \ldots, h,\ h \leq t)$ of integer variables $k_1, \ldots, k_l$ for which the probability $p \mathrel{\widehat{=}} \mu(\mathcal{S}^t(\mathcal{D}))$, where $\mathcal{S}^t(\mathcal{D}) = \{\sigma \mid \sigma_{[t]} \models \mathcal{D} \wedge \sigma_{[t]} \in \mathcal{R}\}$ is calculated as*

$$p = \sum_{\substack{k_1, \ldots, k_l \geq 1,\ f(k_1, \ldots, k_l) = t, \\ g_j(k_1, \ldots, k_l) \leq M, j = 1, \ldots, m}} P(k_1, \ldots, k_l)$$

Proof: There are two cases to consider.

(a). If there is no occurrence of the operator $^+$ in $\mathcal{R}$ then $\mathcal{R} = \sigma$ where σ is a non empty sequence with the length $|\sigma| = t$, then we can check whether every prefix of σ (with length from 1 to t) satisfies $\mathcal{D}$. The constructions of f and g are trivial. So, the theorem stands.

(b). If there is an occurrence of the operator $^+$ in $\mathcal{R}$, then by the definition of the well-formedness, $\mathcal{R}$ should be of the form $\mathcal{R} = \sigma_1\gamma_1^+\sigma_2\gamma_2^+ \ldots \sigma_l\gamma_l^+\sigma_{l+1}$, where $l \geq 1$, $\gamma_i \neq \epsilon$, $\sigma_i \neq \epsilon$, $1 \leq i \leq l$, σ_{l+1} can be ϵ (ϵ is the empty word). Therefore, any string $\sigma_{[t]} \in \mathcal{L}(\mathcal{R})$ can be written as $\sigma = \sigma_1\gamma_1^{k_1}\sigma_2\gamma_2^{k_2} \ldots \sigma_l\gamma_m^{k_l}\sigma_{l+1}$, $k_i \geq 1$. Because $\sigma_{[t]}$ has the length t, $k_1, \ldots, k_l$ should satisfy the constraint (denoted by $f = t$):

$$\textstyle\sum_{j=1}^{l}(|\sigma_j| + k_j * |\gamma_j|) + |\sigma_{l+1}| = t$$

It is easy to compute the set of all non-empty prefixes of $\sigma_{[t]}$ as

$$\begin{aligned} \mathit{prefix'}(\sigma) \;=\; & \mathit{prefix'}(\sigma_1)\cup \\ & \sigma_1(\textstyle\bigoplus_{i=0}^{k_1-1}\gamma_1^i)\mathit{prefix'}(\gamma_1)\cup \\ & \sigma_1\gamma_1^{k_1}\mathit{prefix'}(\sigma_2)\cup \\ & \sigma_1\gamma_1^{k_1}\sigma_2(\textstyle\bigoplus_{i=0}^{k_2}\gamma_2^i)\mathit{prefix'}(\gamma_2)\cup \\ & \quad\vdots \\ & \sigma_1\gamma_1^{k_1}\sigma_2\gamma_2^{k_2}\ldots\gamma_2^{k_l}\mathit{prefix'}(\sigma_{l+1}) \end{aligned}$$

where for a string δ, $\mathit{prefix'}(\delta) \mathrel{\widehat{=}} \mathit{prefix}(\delta)\cup\{\delta\}$ (the set of all non-empty prefixes of δ). Now, for a sequence δ let $\Psi(\delta)$ denote the value of the expression $\sum_{j=1}^{m} c_j \int q_i$ evaluated over δ. It is obvious that $\Psi(\delta_1\delta_2) = \Psi(\delta_1) + \Psi(\delta_2)$ for any sequences δ_1, δ_2. In order for $\sigma_{[t]}$ to satisfy $\mathcal{D}$, all prefixes σ' of $\sigma_{[t]}$ should satisfy $\Psi(\sigma') \le M$. Therefore, taking into account that $\max\{\Psi(\gamma_j^i) \mid 0 \le i \le k_j - 1\} = (1/2) * (1 + \mathit{sign}(\Psi(\gamma_j))) * (k_j - 1) * \Psi(\gamma_j)$, where for a real number a

$$\mathit{sign}(a) \quad\mathrel{\widehat{=}}\quad \begin{cases} 1 \text{ if } a > 0 \\ 0 \text{ if } a = 0 \\ -1 \text{ if } a < 0\,, \end{cases}$$

the numbers $k_1, \ldots, k_l$ should satisfy the following constraints (denoted by $g_j's \le M$) as well,

$$\begin{aligned} & \Psi(\delta) \le M \text{ for } \delta \in \mathit{prefix'}(\sigma_1), \\ & \Psi(\sigma_1)+ \\ & \quad (1/2) * (1 + \mathit{sign}(\Psi(\gamma_1))) * (k_1 - 1) * \Psi(\gamma_1)+ \\ & \qquad \Psi(\delta) \le M \\ & \qquad\quad \text{for } \delta \in \mathit{prefix'}(\gamma_1), \\ & \Psi(\sigma_1) + k_1 * \Psi(\gamma_1) + \Psi(\delta) \le M \text{ for } \delta \in \mathit{prefix'}(\sigma_2), \\ & \Psi(\sigma_1) + k_1 * \Psi(\gamma_1) + \Psi(\sigma_2)+ \\ & \quad (1/2) * (1 + \mathit{sign}(\Psi(\gamma_2))) * (k_2 - 1) * \Psi(\gamma_2)+ \\ & \qquad \Psi(\delta) \le M \\ & \qquad\quad \text{for } \delta \in \mathit{prefix'}(\gamma_2), \\ & \quad\vdots \\ & \textstyle\sum_{j=1}^{l}(\Psi(\sigma_j) + k_j * \Psi(\gamma_j)) + \Psi(\delta) \le M \\ & \qquad\quad \text{for } \delta \in \mathit{prefix'}(\sigma_{l+1}). \end{aligned}$$

Since the number of non-empty prefixes of a string is its length, the number h of the inequalities in the above constraint system is at most t. Let
$P(k_1, \ldots, k_l) \mathrel{\widehat{=}} (\Pi_{j=1}^{l}(\sigma_j * (\gamma_j)^{k_j})) * \sigma_{l+1}$, where σ_j, γ_j are 'products of probability' (remember that we have overloaded these symbols!), and if the sequence σ_{l+1} is empty then the 'product' σ_{l+1} is 1. Because of the well-formedness, each tuple $(k_1, \ldots, k_l)$ satisfying the above constraints

($f = t$ and $g_j \leq M$, $j \leq h$) corresponds to exactly one sequence in $\mathcal{S}^t(\mathcal{D})$ and vice-versa. Therefore, the theorem is verified for all cases. □.

Note that we can simplify the inequality system in the proof of Theorem 2 to an equivalent one with no more than l inequalities in the obvious way. The complexity of the procedure for reducing is of linear-time on t. Thus, the complexity of the computation of $\mu(\mathcal{S}^t(\mathcal{D}))$ using Theorem 2 is of linear-time on t and polynomial-time on the number of the occurrences of the operator $^+$.

Next we will give a technique to decompose a well-formed regular expression with nested occurrence of the operator $^+$ into well-formed ones with the number of nested occurrences of the operator $^+$ decreasing through the following theorem.

Theorem 3 *Given a well-formed regular expression $\mathcal{B}$ with a nested occurrence of the operator $^+$, given a time t, we can find well-formed regular expressions $\mathcal{R}_0$, $\mathcal{R}_1$, ..., $\mathcal{R}_k$ such that*

1. *Each $\mathcal{R}_i$ is well-formed and has fewer occurrences of the operator $^+$ than $\mathcal{R}$,*
2. *$\mathcal{R}_i \cap \mathcal{R}_j = \emptyset$ for $i \neq j$.*
3. *$\mathcal{B} \cap \Sigma^t = \bigoplus_{j=1}^{k} (\mathcal{R}_j \bigcap \Sigma^t)$.*

Proof: By the associativity of the concatenation operator in the definition of the w.f. REs as has been noted earlier, we can assume that any w.f. regular expression $\mathcal{B}$ with nested occurrences of the operator $^+$ is of one of the following forms

$$\begin{aligned} \mathcal{B} &= \mathcal{Q}_1 {\mathcal{Q}_2}^+ \mathcal{Q}_3 \\ \mathcal{B} &= \mathcal{Q}_1 {\mathcal{Q}_2}^+ \\ \mathcal{B} &= {\mathcal{Q}_2}^+ \mathcal{Q}_3 \\ \mathcal{B} &= {\mathcal{Q}_2}^+ \end{aligned}$$

where there is an occurrence of the operator $^+$ in $\mathcal{Q}_2$.

We prove the theorem for the case that $\mathcal{B}$ is of the first form. The other cases of $\mathcal{B}$ can be proved in the same way.

Assume that $\mathcal{B} = \mathcal{Q}_1 {\mathcal{Q}_2}^+ \mathcal{Q}_3$ where there is an occurrence of the operator $^+$ in $\mathcal{Q}_2$. Because we are interested in only the strings in $\mathcal{B}$ which have the length t, we can replace the operator $^+$ with a number of repetitions that is big enough. Let $m_j \mathrel{\widehat{=}} \min(\{|\sigma| \mid \sigma \in \mathcal{L}(\mathcal{Q}_j) \wedge \sigma \neq \epsilon\})$, $j = 1, 2, 3$. Let $k \mathrel{\widehat{=}} (t - m_1 - m_3)/m_2$. Then, a string in $\mathcal{L}(\mathcal{R})$ with more than k repetitions will be longer than t. Replacing ${\mathcal{Q}_2}^+$ with $\bigoplus_{i=1}^{k} {\mathcal{Q}_2}^i$ in $\mathcal{Q}_1 {\mathcal{Q}_2}^+ \mathcal{Q}_3$ gives $\mathcal{Q}_1 (\bigoplus_{i=1}^{k} {\mathcal{Q}_2}^j) \mathcal{Q}_3$. Let $\mathcal{R}_i \mathrel{\widehat{=}} \mathcal{Q}_1 {\mathcal{Q}_2}^i \mathcal{Q}_3$. It is easy to verify that $\mathcal{R}_i$'s satisfy the three items in the theorem. □

Note that the procedure for removing the nested occurrences of the operator $^+$ has linear-time complexity on t and the number n of the nested occurrence of the operator $^+$, and the resulting expression has no more than $n * t$ occurrences of $\bigoplus$.

Based on Theorems 1, 2 and 3, our algorithms for checking probabilistic real-time systems are given as follows.

Algorithm 1

Input A Markov chain $\mathcal{M}$, a duration property $\mathcal{D}$ of the form $(K1)$, a time t, a probability r and a regular expression $\mathcal{B}$ expressing all the prefixes having no occurrence of the transitions with zero probability of the behaviour of $\mathcal{M}$.

Output "Yes" if $\mathcal{M}$ satisfies $\mathcal{D}$ at time t with the probability at least r, and "No" otherwise.

Method

1. If $t < t_1$ or $t > t_2$ then output "Yes" and stop.
2. Convert $\mathcal{B}$ into an equivalent expression of the form $\bigoplus_{j=1}^{\nu} \mathcal{B}_j$ which satisfies that $\mathcal{B}_j$'s are w.f. REs and that $\mathcal{L}(\mathcal{B}_i) \cap \mathcal{L}(\mathcal{B}_j) = \emptyset$ for $i \neq j$.
3. Apply the technique in Theorem 3 repeatedly until we find w.f. regular expressions $\mathcal{R}_0$, $\mathcal{R}_1$, ..., $\mathcal{R}_k$ having no nested occurrences of the operator $^+$ that satisfy all the items in Theorem 3.
4. Apply the technique in Theorem 1 for each $\mathcal{R}_i$, $i = 0, 1, \ldots, k$, we can compute the probability $P_i \mathrel{\widehat{=}} \mu({\mathcal{S}_i}^t(\mathcal{D}))$, where ${\mathcal{S}_i}^t(\mathcal{D}) = \{\sigma \mid \sigma_{[t]} \models \mathcal{D} \ \wedge \sigma_{[t]} \in \mathcal{R}_i\}$.
5. If $\sum_{i=1}^{k} P_i \geq r$ then output "Yes", otherwise output "No"

Algorithm 2

Input A Markov chain $\mathcal{M}$, a duration property $\mathcal{D}$ of the form $K2$, a time t, a probability r and a regular expression $\mathcal{B}$ expressing all the prefixes having no occurrence of the transitions with zero probability of the behaviour of $\mathcal{M}$.

Output "Yes" if $\mathcal{M}$ satisfies $\mathcal{D}$ at time t with the probability at least r, and "No" otherwise.

Method

1. Convert $\mathcal{B}$ into an equivalent expression of the form $\bigoplus_{j=1}^{\nu} \mathcal{B}_j$ which satisfies that $\mathcal{B}_j$'s are w.f. REs and that $\mathcal{L}(\mathcal{B}_i) \cap \mathcal{L}(\mathcal{B}_j) = \emptyset$ for $i \neq j$.
2. Apply the technique in Theorem 3 repeatedly until we find w.f. regular expressions $\mathcal{R}_0$, $\mathcal{R}_1$, ..., $\mathcal{R}_k$ having no nested occurrences of the operator $^+$ that satisfy all the items in Theorem 3.
3. Apply the technique in Theorem 2 for each $\mathcal{R}_i$, $i = 0, 1, \ldots, k$, we can compute the probability $P_i \mathrel{\widehat{=}} \mu({\mathcal{S}_i}^t(\mathcal{D}))$, where ${\mathcal{S}_i}^t(\mathcal{D}) = \{\sigma \mid \sigma_{[t]} \models \mathcal{D} \ \wedge \sigma_{[t]} \in \mathcal{R}_i\}$.

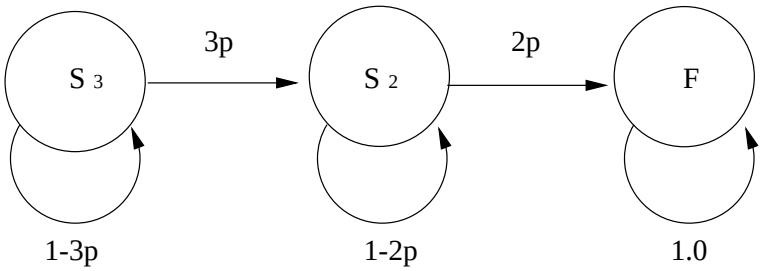

Figure 2: Reduced model of the TMR system with a minimal number of states

4. If $\sum_{i=1}^{k} P_i \geq r$ then output "Yes", otherwise output "No"

Note that we can generalise Theorem 2 to handle the case that the duration property $\mathcal{D}$ is of the form $\Box(t_1 \leq l \leq t_2 \Rightarrow \sum_{j=1}^{m} c_j * \int q_j \leq M)$ easily, and Algorithm 2 can be modified to work with this kind of input as well. For simplicity, we do not present this case in this paper.

4 Example

In this section, we apply our technique to check the reliability of a simple system taken from the literatures (see [11]).

In order to improve reliability, passive hardware redundancy is a common method to be used in real-time systems. Passive hardware redundancy can mask the occurrence of faults. The passive approaches achieve fault tolerance without the need for fault detection or system reconfiguration, the passive designs inherently tolerate the faults.

The most common form of passive hardware redundancy is called triple modular redundancy (TMR). The basic concept of TMR is to triplicate the hardware and perform a majority vote to determine the output of the system. If one of the modules becomes faulty, the two remaining fault-free modules mask the results of the faulty module when the majority vote is performed. In typical applications, the replicated modules are processors, memories, or any hardware entity. In addition, TMR can be applied to software where three different versions of the programs the perform the same function are used to protect against software faults in any one of the three.

Suppose that we let state S_3 correspond to the state in which all three modules in the TMR system are functioning correctly; state S_2 is the state in which two modules are working correctly, state F is the failed state in which two or more modules have failed. The figure 2 illustrates the reduced model of the TMR system.

The state transition probabilities shown in figure 2 have been derived to account for one of several failure occurring. For example, the probability of transitioning from state S_3 to state S_2 depends

on the probability of any one of three modules failing. Consequently, the transition probability assigned to the transition from state S_3 to state S_2 is $3p$. Likewise the transition probability assigned to the state transition from state S_2 to state F is $2p$. p is a failure probability of one module in TMR system.

The requirement for the reliability of the TMR system is that *the system works correctly with probability* p_{TMR} *at least* 95% *within operating interval of 100 time units.* The requirement is captured by the following linear duration property $\mathcal{D}$:

$$\mathcal{D} \quad \widehat{=} \quad \boxdot \int F \leq 0$$

which means that no failure occurs.

From the nature of the requirement, we now adopt Algorithm 2 in the previous section to check whether the TMR satisfies $\mathcal{D}$ at time 100 (for this special case, because $\int F \leq 0 \Rightarrow \boxdot \int F \leq 0$, we can use the Algorithm 1 as well). The regular expression expressing all the possible prefixes of the behaviours of the reduced TMR system is

$$\begin{aligned}\mathcal{B} \quad = \quad & p_{0,S_3} p^*_{S_3,S_3} p_{S_3,S_2} p^*_{S_2,S_2} p_{S_2,F} p^*_{F,F} \bigoplus \\ & p_{0,S_3} p^*_{S_3,S_3} p_{S_3,S_2} p^*_{S_2,S_2} \bigoplus p_{0,S_3} p^*_{S_3,S_3}\end{aligned}$$

where $p_{i,j}$, $i,j \in \{0, S_3, S_2, F\}$, denotes the transition from state i to state j (overloaded with the probability for the transition), and $p_{0,S_3} = 1.0$, which expresses that TMR starts at S_3. By converting the operator * into the operator $^+$ we get

$$\begin{aligned}\mathcal{B} \quad = \quad & p_{0,S_3} p^+_{S_3,S_3} p_{S_3,S_2} p^+_{S_2,S_2} p_{S_2,F} p^+_{F,F} \bigoplus \\ & p_{0,S_3} p^+_{S_3,S_3} p_{S_3,S_2} p^+_{S_2,S_2} p_{S_2,F} \bigoplus \\ & p_{0,S_3} p^+_{S_3,S_3} p_{S_3,S_2} p_{S_2,F} p^+_{F,F} \bigoplus \\ & p_{0,S_3} p_{S_3,S_2} p^+_{S_2,S_2} p_{S_2,F} p^+_{F,F} \bigoplus \\ & p_{0,S_3} p^+_{S_3,S_3} p_{S_3,S_2} p_{S_2,F} \bigoplus \\ & p_{0,S_3} p_{S_3,S_2} p^+_{S_2,S_2} p_{S_2,F} \bigoplus \\ & p_{0,S_3} p_{S_3,S_2} p_{S_2,F} p^+_{F,F} \bigoplus \\ & p_{0,S_3} p^+_{S_3,S_3} p_{S_3,S_2} p^+_{S_2,S_2} \bigoplus \\ & p_{0,S_3} p^+_{S_3,S_3} p_{S_3,S_2} \bigoplus p_{0,S_3} p_{S_3,S_2} p^+_{S_2,S_2} \bigoplus \\ & p_{0,S_3} p^+_{S_3,S_3} \cdot\end{aligned}$$

Each of components of the $\bigoplus$ is well-formed and has no nested occurrence of the operator $^+$. Obviously, we can omit those sequences which have the length less than $t = 100$.

Suppose $p = 0.001$. The satisfaction probability of $\mathcal{D}$ by the components of the $\bigoplus$ is calculated using Theorem 2 as follows.

For the components with an occurrence of $p_{S_2,F}$, there exists a constraint of the form $g_j \leq M$ which is $1 \leq 0$. Therefore, there is no solution for the corresponding constraint system, and the satisfaction probability of $\mathcal{D}$ by these components is 0. Therefore, we have to calculate the satisfaction probability of $\mathcal{D}$ by the last four components only.

The result is

$$
\begin{array}{ll}
 & p_{TMR} \\
= & \sum_{k_1+k_2=98,k_1,k_2\geq 1}(1-3p)^{k_1} * 3p * (1-2p)^{k_2} + \\
 & \sum_{k_1=98,k_1\geq 1}(1-3p)^{k_1} * 3p + \\
 & \sum_{k_2=98,k_1\geq 1} 3p * (1-2p)^{k_2} + \\
 & \sum_{k_1=99,k_1\geq 1}(1-3p)^{k_1} \\
= & \sum_{k_1+k_2=98,k_1,k_2\geq 0}(1-3p)^{k_1} * 3p * (1-2p)^{k_2} + \\
 & (1-3p)^{99} \\
= & 3 * ((1-2p)^{99} - (1-3p)^{99}) + (1-3p)^{99} \\
\geq & 0.232 + 0.742 \\
= & 0.974\,.
\end{array}
$$

Therefore, Algorithm 2 gives the answer "Yes" which means that the TMR satisfies it reliability requirement.

5 Conclusion

We have presented our technique for checking whether a Markov chain satisfies a linear duration property within a given time with the probability not lower than a given number. The main advantage of our technique is that it reduces the complexity of checking significantly in comparison to the methods in the literature (e.g. in [13]). Besides, it makes use of the classical results on regular expressions and integer linear constraints that are well studied.

We believe that the techniques for solving the problem using Markov chains as the model for probabilistic real-time systems are fundamental for checking the system dependability because the problem for the probabilistic real-time automata with continuous time formulated in [5] can be converted into the problem for Markov chains easily by using simple discretization technique.

Although the techniques presented in this paper can deal with only those simple duration properties (of the form $(K1)$ and $(K2)$), they can also be extended to deal with the linear duration invariants.

References

[1] R. Alur, C. Courcoubetis and D. L. dill. *Model Checking for Probabilistic Real-time Sys-*

tems. In Proceedings of the 18th International Conference on Automata, Languages and Programming (ICALP), LNCS 510, 1991.

[2] R. Alur, C. Courcoubetis and D. L. dill. *Verifying Automata Specifications of Probabilistic Real-time Systems.* In Proceedings of International Conference on Real-time: Theory in Practice, LNCS 600, 1991.

[3] C. Courcoubetis and M. Yannakakis. *Verifying Temporal Properties of Finite-state Probabilistic Programs.* in Proceedings of 1998 IEEE Symposium on the Foundations of Computer Science, 1998.

[4] D. R. Cox and H. D. Miller. *The Theory of Stochastic Process.* Methuen & Co LTD, 1964.

[5] H. Dang Van and C.C. Zhou. *Probabilistic Duration calculus for Continuous Time.* To apear in *Formal Aspects of Computing.*

[6] H. Dang Van and H.T. Pham. On Checking Parallel Real-Time Systems for Linear Duration Invariants. In Proceedings of PDSE'98, Bernd Kramer, Naoshi Uchihira, Peter Croll and Stefano Russo (Eds), IEEE Computer Society Press, 1998, pp. 61 – 71.

[7] E. A. Emerson and J. Y. Halpen. *"Sometimes" and "Not ever" Revisited: On Branching Time versus Linear Time Temporal Logic.* JACM, 33(1):151-178, 1986.

[8] M. R. Hansen. *Model-Checking Discrete Duration Caculus.* Formal Aspects of Computing, 6A:826-845, 1994.

[9] P. Iyer and M. Narasimha. *"Almost always" and "Definitely sometime" are not enough: Probabilistic quantifiers and Probabilistic model-checking.* Tech Report TR-96-16, North Carolina State University, 1996.

[10] Y. Kesten, A. Pnueli, J. Sifakis, and S. Yovine. Integration Graphs: A Class of Decidable Hybrid Systems. In *Hybrid Systems*, volume 736 of *Lecture Notes in Computer Science*, pages 179–208. Springer Verlag, 1994.

[11] B. W. Johnson, Design and Analysis of Fault Tolerant Digital Systems, Reading, MA: Addison-Wesley, 1989.

[12] X.D. Li, H. Dang Van and Zheng Tao. Checking Hybrid Automata for Linear Duration Invariants. In R. Shyamasundar and K. Ueda, editors, *Advances in Computing Science - ASIAN'97*, volume 1345 of *Lecture Notes in Computer Science*, pages 166–180. Springer Verlag, 1997.

[13] Z. Liu, P. Ravn, E. V. Sørensen and Zhou Chaochen. *Towards a Calculus of Systems Dependability.* In Proceedings of Workshop on Theory of Hybrid Systems, Lyngby, Denmark, 1992.

[14] M. Y. Vardi. *Automatic Verification of Probabilistic Concurrent Finite-state Programs.* In proceedings of IEEE Symposium on Foundations of Computer Science (FOCS), 1983.

[15] C.C. Zhou, J.Z. Zhang, Yang Lu, and X.S. Li. *Linear Duration Invariants.* in Proceedings of International Conference on Formal Techniques in Real-Time and Fault-Tolerant systems, LNCS 863, 1994.