Counterexample Generation for Conditional Probability in Probabilistic Model Checking

Mingyu Ji
College of Information and Computer Engineering, Northeast Forestry University, Harbin, China
College of Computer Science and Technology, Harbin Engineering University, Harbin, China
Email: jimingyunfu@yeah.net

Di Wu
College of Information and Computer Engineering, Northeast Forestry University, Harbin, China

Yanmei Li, Zhiyan Chen
College of Computer Science and Technology, Harbin Engineering University, Harbin, China

***Abstract*—with the wide application of probabilistic systems, the performance analysis for probabilistic system with model checking have attracted wide attention. For conditional probability formulae of complex parametric system, this paper proposes a counterexample generation method of conditional probability properties based on continuous time probabilistic model. We use continuous time Markov reward model with comprehensive feature representation ability as the system model need to be verified, give satisfiability probability solution algorithm of probabilistic computation tree logic multiple constraints until formulae path properties after model pretreatment, put forward the counterexample generation method of conditional probability on multiple constraints until formulae and give the example analysis. The theoretical analysis and example result show that the feasibility and validity of the method.**

***Index Terms*—probabilistic system, performance analysis, model checking, conditional probability, counterexample generation**

I. INTRODUCTION

Model checking [1] as a highly automated formal verification technology has been successfully applied to communication protocols [2], biological modeling, control systems [3] and other fields [4-5]. Because it can provides counterexample information of properties violation, so that it becomes an efficient mean for automatic detecting faults.

The probabilistic model checking [6], which can comprehensively consider random behavior, time and space characteristics in the system state transition process, is gradually become a new research hotspot in the field of model checking. The counterexample solution methods of quantitative validation for the probabilistic model become important research topics that international experts and scholars are very interested in.

The minimal counterexample algorithms of time boundary reachability probability for continuous time Markov decision process are given in literature [7].In subsequent work [8], different counterexample solution algorithms for continuous time Markov decision process are compared.

In recent years, with the deeply study of model checking techniques, approximate and effective model checking on multiple until formulae (k>2) become an open research problem and more and more importance in the field of system biology. One of the typical applications is observe oscillatory behavior of CTMC model [9]. Literature [10] gives the detailed description of property verification method on time constraints multiple until formulae, and gives the algorithm description on stratified continuous time Markov chain model. The k-shortest(hop-constrained) paths algorithms for counterexample generation of until formulae reachability probability in discrete time Markov chains and a simple algorithm used to generate minimal regular expressions [11] of counterexample paths were both proposed in literature [12]. Literature [13] proposes a counterexample generation algorithm for model checking probabilistic timed automata based on the weighted directed graph. In the literature [14], the semantic representation of the probabilistic timed automata was gave by Markov decision process, and the until formulae counterexample generation algorithm was proposed. In summary, the research on the issue of counterexample mainly focused on the basic probabilistic model and the time constraint until formulae properties.

In the practical application, conditional probability, as a fundamental concept in probability theory, is usually used to solve and validate practical problems, e.g. assessment of risk assets, anonymous protocol analysis and so on. The basic representation of conditional

Manuscript received April 3, 2013; revised May 10, 2013; accepted June 4, 2013.

This work is supported by the Fundamental Research Funds for the Central Universities (Grant No. DL11BB08), the National Undergraduate Innovative Training Project of China (Grant No. 201310225064) and the National Natural Science Foundation of China (Grant No. 71001023).

doi:10.4304/jcp.8.12.3272-3279

probability is: $p[\varphi|\psi] = p[\varphi \wedge \psi]/p[\psi]$, where φ and ψ are temporal logic formulae to be verified. The performance verification and counterexample analysis for condition probability properties have practical significance.

In literature [15], the conditional probability counterexample detection method of computation tree logic properties is proposed for the specific discrete time Markov decision process model. In subsequent work [16], for continuous time Markov chain conditional probability state formulae, an approximate property verification algorithm based on the parameterized product model is presented and analyzed, which adopts an extended continuous stochastic logic to describe system properties.

In this paper, we propose one kind conditional probability counterexample generation method for special multiple until formulae properties with transition step and transition resource consumption constraints on continuous probability model.

Overview of article. Section 1 presents the necessary background on probability model with complex parameters. In section 2 we introduce the syntax of temporal logic and semantic of multiple until formulae and conditional probability. Section 3 discusses the computation method of multiple until formulae probability and gives the basic algorithm description. Section 4 explains how to solve conditional probability counterexample and describes counterexample algorithm. Section 5 combines with concrete example, the calculated results of conditional probability counterexample are given. Finally, in Section 6 we give the conclusions and directions of future research.

II. THE PROBABILITY MODEL WITH COMPLEX PARAMETERS

Probability model has been widely applied in the analysis and design process of software and hardware system. According to the different characteristics of the system, there have different probability models including discrete time Markov chain (DTMC) and continuous time Markov chain (CTMC) and so on. Because they have no ability to describe the characteristics of resource consumption in the transition process, so it is not suitable to represent the specific model mentioned in this paper. Below we introduce the definition of continuous time Markov rewards model (CTMRM), which has the ability to reason about transition resource consumption on the basis of CTMC model.

Definition 1. A DTMC is a tuple M = (S,AP,L,P) where S is a finite set of states, AP is a set of atomic propositions， $L: S \to 2^{AP}$ is a labeling function that assigns to each $s \in S$ a set $L(s)$ of atomic propositions, and $P: S \times S \to [0,1]$ is a probability matrix satisfying $\sum_{s' \in S} P(s,s') \in \{0,1\}$ for all $s \in S$.

Definition 2. A CTMC is a tuple M =(S,AP,L,R) where $R: S \times S \to R_{\geq 0}$ is transition rate matrix, and other notations are defined as for DTMC.

Because DTMC and CTMC have no ability to describe the characteristics of resource consumption in the transition process, so we introduce the definition of Markov reward model (MRM) based on Markov chain model.

Definition 3. Corresponding to the DTMC and CTMC, the MRM are called DTMRM and CTMRM respectively, and expressed as M =(S,AP,L,P,N) and M =(S,AP,L,R,N) where $N: S \times S \to R_{\geq 0}$ is transition resource consumption matrix, and other notations are defined as for DTMC and CTMC.

In CTMRM, transition rewards can be used to represent a variety of different aspects of a system model, e.g. the number of messages successfully transmitted. The terminology rewards here can equally be considered as consumption, e.g. power consumption or number of message packages dropped [17].

The transition probabilities in a CTMRM are exponentially distributed over time. If s' is a current state and $R(s',s'') > 0$, then the probability that take a particular transition to s'' is $\frac{R(s',s'')}{E(s')}(1-e^{-E(s')\cdot t})$ (within t units), where $E(s') = R(s',S') = \sum_{x \in S'} R(s',x)$.

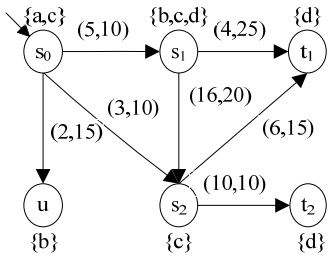

Figure 1. An example CTMRM

Figure 1 shows a CTMRM with six states and seven transitions. s_0 is initial state, $E(s_0) = 10$ and the proposition set that s_1 satisfied is $\{b,c,d\}$. $R(s_1,s_2) = 16$ represents the transition exit rate is 16 that from s_1 to s_2 and $N(s_1,s_2) = 20$ represents the number of resource consumption is 20 units in this transition process.

III. TEMPORAL LOGIC

For the model-based verification of functional properties, temporal logics provide powerful means to specify complex requirements that a system has to satisfy. Several temporal logics had been adapted to reason about probabilistic properties. At present, the fundamental temporal logics applied to the discrete model include mainly probabilistic computational tree logic (PCTL) [18] and its variants, for example PCTL* and so on. In this section, we present the syntax and semantic of the proper temporal logics that with multiple until property description ability.

Definition 4. The syntax of PCTL* is defined as the set of state Φ and set of path φ , respectively, over the set of atomic propositions AP:

$$\Phi ::= true \left| f \right| \Phi_1 \wedge \Phi_2 \left| \neg \Phi \right| P_{\Delta p}(\varphi)$$

$$\varphi ::= \Phi \left| \varphi_1 \wedge \varphi_2 \right| \neg \varphi \left| \bigcirc^n \varphi \right| \varphi_1 \cup^n \varphi_2$$

Where $f \in AP$ be an atomic proposition, $p \in [0,1]$ be probability interval, $\Delta \in \{<, \leq, \geq, >\}$ be a comparison operator, $\cup$ be until operator and $n=[n_l, n_h] \subseteq R_{\geq 0}$ be transition step interval of non-negative real number.

PCTL* are interpreted over the states of a DTMC. We say that a state $s \in S$ satisfies a PCTL* formula Φ, denoted $s \models \Phi$, iff Φ is true for s. The key operator in PCTL* is $P_{\Delta p}(\varphi)$ which means that the probability of set of φ paths satisfies the bound Δp. The path formula $\varphi_1 \cup^{\leq n_1} \varphi_2$ represents that φ_1 holds until φ_2 holds within n_1 steps.

Because the PCTL* does not have the ability to describe resources consumption of transition process in probability model, so here we join the transition resource consumption characteristics into until path formulae of PCTL* for support the transition resource consumption constraint and the transition step constraint at the same time. The improved logic named probabilistic reward computation tree logic star (PRCTL*). PRCTL* path formulae are defined as follows:

$$\varphi ::= \Phi \left| \varphi_1 \wedge \varphi_2 \right| \neg \varphi \left| \varphi_1 \cup_r^n \varphi_2 \right.$$

Where $n=[n_l, n_h] \subseteq R_{\geq 0}$ and $r=[r_l, r_h] \subseteq R_{\geq 0}$ respectively represent transition step and transition resource consumption intervals of non-negative real number.

In order to make the PRCTL* having the ability to describe the conditional probability, we extend the PRCTL* with a new probabilistic operator of the form $p_{\Delta p}[\varphi | \psi]$. $p_{\leq a}[\varphi | \psi]$ expresses that the probability that φ is true given that ψ is true is at most a, where φ and ψ are path formulae of PRCTL*. The new temporal logic named conditional probabilistic reward computation tree logic star (CPRCTL*).

In the following, we study the verification problems of conditional probability formula $p_{\leq a}[\varphi | \psi]$, where φ and ψ are multiple until formulae in the form of $\Phi_1 \cup_{r_1}^{n_1} \Phi_2 \cup_{r_2}^{n_2} \cdots \Phi_k$, where k≥2.

The multiple until formulae semantics are described as follows：

let σ be a path and $\varphi = \Phi_1 \cup_{r_1}^{n_1} \Phi_2 \cup_{r_2}^{n_2} \cdots \Phi_k$ be a path formula of PRCTL*, then $\sigma \models \varphi$ established if and only if σ was divided into a series of transition processes, so that: the last state in the first transition process is a reachable state from the first state of σ at most $Sup(n_1)$ steps and all states except the last state in this transition process satisfy the formula Φ_1 and the last state in this transition process satisfies Φ_2, meanwhile the cumulative sum of transition resource consumption in this transition process is in the range of r_1. Then the next transition process begin from the last state of prior transition process, and the last state in this transition process is a reachable state from the last state of prior transition process at most $Sup(n_2)$ steps and all states except the last state in this transition process satisfy the formula Φ_2 and the last state in this transition process satisfies Φ_3, meanwhile the cumulative sum of transition resource consumption in this transition process is in the range of r_2, and so on. Final, the transition process end in the last state of σ, and this state satisfies Φ_k, where $Sup(n_i)$ denotes the upper bound of interval n_i.

For the transition constraints n_i and r_i in multiple until formulae, we assume that the lower bound is zero and the upper bound is positive real number. In the remainder of this paper, the interval constraints are expressed as $\leq n_i$ and $\leq r_i$.

IV. THE PROBABILITY SOLVING OF MULTIPLE UNTIL FORMULAE

In order to avoid the complex integral computation in the solving probabilistic satisfiability of single until formulae $\phi_1 \cup^{\leq h} \varphi$ and $\phi_1 \cup^{\leq t} \varphi$ in continuous probability model, where h is transition step constraint and t is time constraint, the general approach is as follows: do discretization and uniformization pretreatment first, and transform the original model into discrete probability model, then the problem solving was transformed into the computation of satisfiability probability in the corresponding weighted directed graph.

Example 1. CTMRM pretreatment

For the CTMRM shown in Figure 1, the pretreatment model named uniformized continuous time Markov rewards model (UDTMRM) is depicted in Figure 2. The transition rate matrix of CTMRM, denoted by R, is transformed into the discrete transition probability distribution of UDTMRM, denoted by P_u.

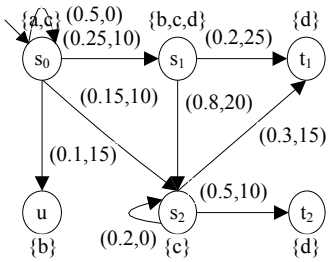

Figure 2. An example UDTMRM

The computing method of P_u is:

$$P_u(s', s'') = \frac{R(s', s'')}{E(s')} \bullet \frac{E(s')}{\max\{E(s_i)\}} \text{ and}$$

$$P_u(s', s') = 1 - \sum_{s'' \in S \wedge s'' \neq s'} P_u(s', s'').$$

In this paper, the pretreatment process of CTMRM model is similar to the above method. After, we can calculate the value of conditional probability state formula $prob(s, \varphi | \psi)$ by means of basic satisfiability probability algorithm on multiple until constraints formulae. In the following, we give the description of calculation method of path satisfiability probability on

multiple until constraints path formula $\Phi_1 \bigcup_{\le r_1}^{\le n_1} \Phi_2 \bigcup_{\le r_2}^{\le n_2} \cdots \Phi_k$.

Definition 5. Satisfiability probability semantic of multiple until constraints path formulae

Let $\varphi = \Phi_1 \bigcup_{\le r_1}^{\le n_1} \Phi_2 \bigcup_{\le r_2}^{\le n_2} \cdots \Phi_k$ be a multiple until constraints path formula. Then the satisfiability probability for the state *s* is expressed as:

$P_s(\Phi_1 \bigcup_{\le r_1}^{\le n_1} \Phi_2 \bigcup_{\le r_2}^{\le n_2} \cdots \Phi_k)$

$\Leftrightarrow prob(s, \Phi_1 \bigcup_{\le r_1}^{\le n_1} \Phi_2 \bigcup_{\le r_2}^{\le n_2} \cdots \Phi_k)$

$\Leftrightarrow prob(Path(s, \Phi_1 \bigcup_{\le r_1}^{\le n_1} \Phi_2 \bigcup_{\le r_2}^{\le n_2} \cdots \Phi_k))$

$\Leftrightarrow$ Looking for the finite path set C that starts from state s and compute $\sum_{\sigma \in C} Prob(\sigma)$, where σ satisfies $\sigma[0] = s \wedge pre(\sigma) \big| \neq \varphi \wedge \sigma \big| = \varphi$.

By the satisfiability probability semantic of multiple until constraint path formulae $\Phi_1 \bigcup_{\le r_1}^{\le n_1} \Phi_2 \bigcup_{\le r_2}^{\le n_2} \cdots \Phi_k$, the basic idea of multiple until formula probability algorithm can be derived as follows: First, we use HK shortest path algorithm [11] to looking for multiple until formulae satisfiability path sets under transition step constraint conditions and check the satisfiability of resource consumption constraint, then obtain the whole multiple until formulae satisfiability path sets and complete the calculation of satisfiability probability.

Algorithm 1. Satisfiability probability solution algorithm of path formulae $\Phi_1 \bigcup_{\le r_1}^{\le n_1} \Phi_2 \bigcup_{\le r_2}^{\le n_2} \cdots \Phi_k$

1: input: state sets S of UDTMRM, state *s* needed to verified, transition step constraint one dimensional array n and transition resource consumption constraint one dimensional array r that both length are *k*-1, state formulae set one dimensional array Φ that contained in multiple until formula.

Output: multiple until constraints path formulae satisfiability probability

```
2: {m=1; pr=0;
3: for (i=1 to k-1)
4:  for ( each s ∈ S )
5:  {pn[s][i]=0;}
6: s=s0;
7: s'=null;
9: dowhile:
10: do {
11:  for (i=1 to k-1)
12:    {pa[m][i] = null;}
13:  for (i=1 to k-1)
14:    {Next:
15:    do {pn[s][i]++;
16:      ρ =FNPbyhkSP(s,Φ[i],n[i],Φ[i+1], pn[s][i]);
17:      if ( ρ ==null)
18:       {if (i<>1)
19:        {i--;
20:        pa[m][i] = null;
21:        s=s';
22:        pn[s][i]++;
23:        goto next ;}
24:       else
25:        break dowhile ; }
26:     } While (sat ( ρ ,r[i])<>true)
27:     pa[m][i]= ρ ;
28:     s'=s;
29:     s=last( ρ );
30:     i++;}
31:   pr=pr+ Prob(pa[m]);
32:   m=m+1;
33:   } while(true)
34: return pr;}
```

Algorithm 2. Resource consumption constraint satisfaction decision algorithm

1:input: finite path ρ , resource consumption constraint upper bound *r*.

Output: true if ρ satisfies the resource consumption constraint or false otherwise.

```
2: {s=first( ρ );totalr=0;
3: while((s'=Next( ρ ,s))!=null)
4:  { totalr= totalr +R(s,s');
5:   s= s';
6:   if (totalr>r) return false ;}
7: return true ;}
```

The time complexity of algorithm is as follows:

The core algorithm of algorithm 1 is function FNPbyhkSP on line 16, which can search the next transition step constraint satisfaction path that starting from s of the i-th formula in multiple until formulae by means of HKSP algorithm. For DTMC and PCTL state formulae $\phi_1 \bigcup^h \varphi$, the HKSP algorithm can complete the calculation of k shortest counterexample path. The time complexity of HKSP algorithm is $O(hm + hk \log(m/n))$, where n and m are the number of states and transitions, h is the step bound, and k is the number of shortest paths.

V. THE SOLUTION OF CONDITIONAL PROBABILITY

For the solution problem of conditional probability $prob(s, \varphi | \psi)$ where φ and ψ are all multiple until constraint formulae, we calculated it by means of the following equivalent form:

$prob(s, \varphi | \psi)$

$\Leftrightarrow prob(s, \varphi \wedge \psi) \Big/ prob(s, \psi)$

$\Leftrightarrow prob(path(s, \varphi \wedge \psi)) \Big/ prob(path(s, \psi))$

Refer to the calculation method of $prob(s, \psi)$ is proposed in algorithm 1, the computation of $prob(s, \varphi \wedge \psi)$ takes the following method:

According to the logical semantic of multiple until constraint formulae φ and ψ , we give one kind of merger and simplification method of multiple until constraint formula conjunctive operation from the angle of formula satisfaction path sets, and then conjunctive operation on φ and ψ is transformed into a multiple

until constraint formulae, so the solution of $prob(s,\varphi\wedge\psi)$ can be completed by algorithm 1.

Because the essence of the merger process of multiple until constraint formula conjunctive operation is the merger of single until constraint formula, so in the following paragraphs we mainly illustrate the merger and simplification process of single until constraint formula, then the merger of multiple until constraint formulae can be completed by this method iteratively.

The merger and simplification method of single until constraint formula conjunctive operation based on formula satisfaction path sets is described as follows:

Let $\varphi=\Phi_1\bigcup_{\le r_1}^{\le n_1}\Phi_2$ and $\psi=\Phi_1'\bigcup_{\le r_1'}^{\le n_1'}\Phi_2'$, then we define $C_1=path(s,(\Phi_1\wedge\Phi_1')\bigcup_{\le\min(r_1,r_1')}^{\le\min(n_1,n_1')}(\Phi_2\wedge\Phi_2'))$ and $C_2=path(s,(\Phi_1\wedge\Phi_1')\bigcup_{\le\min(r_1,r_1')}^{\le\min(n_1,n_1')}(\Phi_2\wedge\Phi_2'))\bullet path(s',\Phi_1'\bigcup_{\le r_1'-z}^{\le|n_1'-n_1|}\Phi_2')$, and $\sigma_1\in C_1,\sigma_2\in C_2$ represent finite path start from s, where symbol $\bullet$ represents the join operation of two finite paths, z represents the resource consumption sum of the left path of C_1, and s' is the last state of the left path. Then $path(s,\varphi\wedge\psi)$ be composed of the following finite path sets. Here we divide into two cases to illustrate how to construct $path(s,\varphi\wedge\psi)$. (W.l.o.g., we assume $n_1\le n_1'$.)

(1) If σ_1 exists and σ_2 which prefix is σ_1 does not exist, then $\sigma_1\in path(s,\varphi\wedge\psi)$.

(2) If σ_1 exists and σ_2 which prefix is σ_1 exists, then $\sigma_2\in path(s,\varphi\wedge\psi)$.

After construction of the above $path(s,\varphi\wedge\psi)$, we adopt the longest path subsets operation for paths with the same prefix to obtain the final $path(s,\varphi\wedge\psi)$.

Definition 6. The longest path subsets with the same prefix

Let C be finite path set, for each finite path σ in C we do the following operation: if σ is the prefix of path in C except σ, then σ is removed from C. After the above processing, we can get the longest path subsets with the same prefix of C, expressed as $lp(C)$.

Example 2. Consider the UDTMRM in Fig. 2 and the path formulae $\varphi=a\bigcup_{\le 20}^{\le 1}b$ and $\psi=c\bigcup_{\le 40}^{\le 3}d$, the path sets from s_0 that satisfy $\varphi\wedge\psi$ after merger and simplification as follows:

$$
\begin{aligned}
&path(s_0,\varphi\wedge\psi)\\
&=path(s_0,(a\bigcup_{\le 20}^{\le 1}b)\wedge(c\bigcup_{\le 40}^{\le 3}d))\\
&=lp(path(s_0,(a\wedge c)\bigcup_{\le 20}^{\le 1}(b\wedge d))\bullet path(s',c\bigcup_{\le 40-z}^{\le 2}d))\\
&=lp(s_0s_1\bullet path(s_1,c\bigcup_{\le 40-10}^{\le 2}d))\\
&=\{s_0s_1t_1,s_0s_1s_2t_2\}
\end{aligned}
$$

By the merger and simplification method of single until constraint formula conjunctive operation, we can extend to the solution of multiple until constraint formula conjunctive operation formula satisfaction path sets. Then we can further obtain the satisfiability probability and the counterexample path sets of multiple until constraint conditional state formula.

Definition 7. The semantic of conditional probability counterexample path sets

If $path(s,\varphi\wedge\psi)$ or $path(s,\psi)$ does not exist, then the counterexample path sets of $P_{\le p}(\varphi|\psi)$ is empty set, otherwise are the minimum path subsets that path probability value beyond p in $path(s,\varphi\wedge\psi)$ and $path(s,\psi)$. Among them, $path(s,\psi)$ are non critical path sets.

Algorithm 3. Counterexample path sets solution algorithm of $P_{\le p}(\varphi|\psi)$

1: input: state sets *S* of UDTMRM, state *s* needed to verified, n and r are transition step constraint one dimensional array and transition resource consumption constraint one dimensional array in $\varphi\wedge\psi$ respectively, n' and r' are respectively transition step constraint one dimensional array and transition resource consumption constraint one dimensional array in ψ, Φ and Φ' are respectively state formulae set one dimensional array that containedin $\varphi\wedge\psi$ and ψ.

Output: counterexample path sets array if exists or empty set otherwise.

```
2: {m=1; p1=0; p2=0;
3: p1=getPro(S,s0,n,r,Φ,pa1);
4: p2=getPro(S,s0, n',r',Φ'[],pa2);
5: if (p1==0 or p2==0)
6:   return null;
7: if (p1/p2>p)
8:   {p1=0;
9:   porder (pa1);
10:  For (i=1 to pa1.length)
11:   { p1=p1+prob(pa1[i]);
11:    ans[i]=pa1[i];
13:    if (p1/p2>p)
14:      break;
15:    i++ ;}
16:  return (ans[1]…ans[i]);}
17: else
18: return null ;}
```

By the algorithm 3, we can obtain the set of finite paths as the counterexamples of $P_{\le p}(\varphi|\psi)$. But the number of finite paths is usually very big since system model is usually very complex and the size of model is usually very big. So this results in two problems. First, it leads to the storage problem as counterexamples may simply get too large to be kept in memory. Secondly, and more importantly, counterexamples will be incomprehensible to the user. We therefore need to find ways to reduce the number of evidences in a counterexample, and to obtain a compact and user friendly representation. To that purpose we introduce how to use regular expressions to describe counterexamples in the following. We adopt the approach proposed by Daws [19] and Damman [20]. They use regular expressions to represent sets of paths and calculate the exact rational value of the probability

measure and provide a function to evaluate the probability measure of those expressions.

Definition 8. Automaton of UDTMRM

For UDTMRM M =(S,AP,L,P_u,N), the corresponding deterministic finite automaton(DFA) denoted as $A_M = (S', \sum, \tilde{s}, \delta, f)$, where $S' = S \bigcup \{\tilde{s}\}$ is the state space, $\sum \subset [0,1] \times N \times S$ is finite alphabet, f is the accepting state, $\delta \subset S' \times \sum \times S'$ is the transition relation such that $\delta(s_1, (p, n, s_2)) = s_2$ if $P(s_1, s_2) = p$ and $N(s_1, s_2) = n$ and $\delta(\tilde{s}, (1, 0, s_0)) = s_0$ if s_0 is initial state of UDTMRM and $\tilde{s} \notin S$ is the starting state of DFA.

Note that $\tilde{s}$ is a new starting state in A_M connection to the initial state of M with a transition labeled with $(1, 0, s_0)$. Different from [20], for transition $s_1 \xrightarrow{p, s_2} s_2$, we add the target state s_2 and transition probability p together with the resource consumption n as the symbol (p, n, s_2) to $\sum$.

Example 3. Consider the UDTMRM in Fig. 2, its DFA depicts in Fig. 3. The new starting state is $\tilde{s}$ and the accepting state set is $\{u, t_1, t_2\}$.

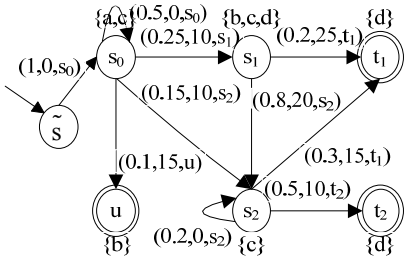

Figure 3. UDTMRM DFA

Definition 9. Acceptable language of DFA

Let $A_M = (S', \sum, \tilde{s}, \delta, f)$ is a DFA of UDTMRM, $A_i \in \Sigma$ is input alphabet of DFA, $s \in S'$ represents any state in DFA, $\sum$ contains elements of the form (p_u, n, s) where $p_u \in [0,1]$ be transition probability, $n \in N$ be resource consumption number in transition process of UDTMRM. Finite state series $s_0 s_1 \ldots s_n$ is called an acceptable word of A_M if and only if s_0 is a starting state, for all $0 \le i < n$ $s_i \xrightarrow{A_{i+1}} s_{i+1}$ and s_n is a accepting state. The set of acceptable words of A_M is called acceptable language of A_M.

The set $R(\sum)$ of regular expressions over the finite alphabet $\sum$ is the set of expressions containing the elements of $\sum$, the empty word ε, and which is closed under union (|), concatenation (.) and Kleene star (*).

Let $L(r)$ denote the regular language described by the regular expression $r \in R(\sum)$ and $L(\sum)$ denote the regular language that can be generated by any regular expression over $\sum$. For a word ω, $|\omega|$ denotes the number of symbols in ω. We sometimes omit · and write $r.r'$ as rr' for short.

Definition 10. The regular expressions of UDTMRM DFA can be evaluated by two functions, i.e. the probability function $val_p : R(\sum) \to R$ and resource function $val_r : R(\sum) \to R$ as follows:

$$val_p(\varepsilon) = 1$$
$$val_p(r|r') = val_p(r) + val_p(r')$$
$$val_p(p, n, s) = p$$
$$val_p(r.r') = val_p(r) \times val_p(r')$$
$$val_p(r^*) = 1 \text{ if } val_p(r) = 1 \text{ and } val_p(r^*) = \frac{1}{1 - val_p(r)}$$

otherwise .

$$val_r(\varepsilon) = 0$$
$$val_r(r|r') = val_r(r) + val_r(r')$$
$$val_r(p, n, s) = n$$
$$val_r(r.r') = val_r(r) + val_r(r')$$
$$val_r(r^*) = \sum_{i=1}^{\infty} val_r(r)$$

VI. COUNTEREXAMPLE ANALYSIS

Now, for the state s_0 of UDTMRM in Fig. 4, we verify the satisfiability of conditional probability formula $P_{\le 0.3}(a \bigcup_{\le 20}^{\le 1} b \bigcup_{\le 35}^{\le 2} d \,|\, c \bigcup_{\le 40}^{\le 3} d \bigcup_{\le 20}^{\le 1} e)$ by our algorithms and give the detail description of counterexample set generation process.

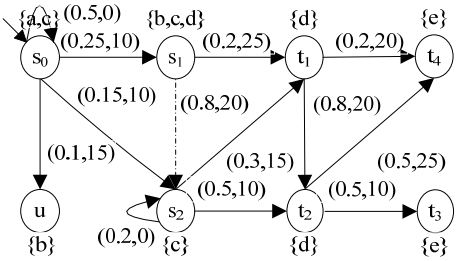

Figure 4. Instance model

By the equivalent formulae of conditional probability, we can obtain:

$$prob(s_0, a \bigcup_{\le 20}^{\le 1} b \bigcup_{\le 35}^{\le 2} d \,|\, c \bigcup_{\le 40}^{\le 3} d \bigcup_{\le 20}^{\le 1} e)$$
$$= prob(s_0, a \bigcup_{\le 20}^{\le 1} b \bigcup_{\le 35}^{\le 2} d \wedge c \bigcup_{\le 40}^{\le 3} d \bigcup_{\le 20}^{\le 1} e) \,|\, prob(s_0, c \bigcup_{\le 40}^{\le 3} d \bigcup_{\le 20}^{\le 1} e)$$

Among them,

$$prob(s_0, a \bigcup_{\le 20}^{\le 1} b \bigcup_{\le 35}^{\le 2} d \wedge c \bigcup_{\le 40}^{\le 3} d \bigcup_{\le 20}^{\le 1} e)$$
$$= prob(\{s_0 s_1 t_1 t_4, s_0 s_1 s_2 t_2 t_3\})$$
$$= 0.01 + 0.05 = 0.06,$$
$$prob(s_0, c \bigcup_{\le 40}^{\le 3} d \bigcup_{\le 20}^{\le 1} e)$$
$$= prob(path(s_0, c \bigcup_{\le 40}^{\le 3} d \bigcup_{\le 20}^{\le 1} e))$$
$$= prob(\{s_0 s_1 t_1 t_4,\ s_0 s_1 s_2 t_2 t_3,\ s_0 s_2 t_1 t_4,\ s_0 s_2 s_2 t_1 t_4,\ s_0 s_2 t_2 t_3,\ s_0 s_2 s_2 t_2 t_3,\ s_0 s_0 s_2 t_1 t_4,\ s_0 s_0 s_2 t_2 t_3\})$$

=0.13905.

Therefore, the final result is:

$prob(s_0, a \cup_{\le 20}^{\le 1} b \cup_{\le 35}^{\le 2} d \,|\, c \cup_{\le 40}^{\le 3} d \cup_{\le 20}^{\le 1} e)$

=0.06/0.13905

=0.4315

Because of 0.4315 is greater than 0.3, so state formula $P_{\le 0.3}(a \cup_{\le 20}^{\le 1} b \cup_{\le 35}^{\le 2} d \,|\, c \cup_{\le 40}^{\le 3} d \cup_{\le 20}^{\le 1} e)$ is not established for the state s_0 .One of the counterexample path sets include $\{s_0 s_1 s_2 t_2 t_3\}$ and $path(s_0, c \cup_{\le 40}^{\le 3} d \cup_{\le 20}^{\le 1} e)$, namely,

$\{s_0 s_1 s_2 t_2 t_3, s_0 s_1 t_1 t_4, s_0 s_1 s_2 t_2 t_3, s_0 s_2 t_1 t_4, s_0 s_2 s_2 t_1 t_4, s_0 s_2 t_2 t_3, s_0 s_2 s_2 t_2 t_3, s_0 s_0 s_2 t_1 t_4, s_0 s_0 s_2 t_2 t_3\}$.

VII. Conclusion and Future Work

In this paper we extended the probabilistic temporal logic PCTL* to CPRCTL*, in which it is possible to express conditional probability of multiple until constraints path formulae. We presented one algorithm to check if a CTMRM satisfies a multiple until constraints path formulae. Our algorithm first reduces the CTMRM to UDTMRM and then computes satisfiability probability based on this reduction. Then we presented another algorithm to solve the counterexamples of multiple until constraints path formulae conditional probability. Counterexamples for conditional formulae consist of two sets of paths in the CTMRM. At last, we illustrated the validity of the algorithms proposed in this paper by an instance.

A natural direction for future research is to investigate ways to find better counterexamples in probabilistic model checking and research how to apply conditional probability model checking to verify the correctness of specific practical problems.

Acknowledgment

This work was supported by the Fundamental Research Funds for the Central Universities (Grant No.DL11BB08), the National Undergraduate Innovative Training Project of China (Grant No.201310225064) and the National Natural Science Foundation of China (Grant No. 71001023).

References

[1] Huimin Lin and Wenhui Zhan, "Model checking: theories, techniques and applications," *Acta Electronica Sinica*, vol. 30, no. l2, pp.1907-1912, 2002.

[2] Weibo Liu, Wenping Ma and Yuanyuan Yang, "Analysis and improvement of the BAN modified andrew secure RPC protocol," *Journal of Networks*, vol. 6, no. 4, pp.662-669, 2011.

[3] Huiling Shi, Wenke Ma, Meihong Yang and Xinchang Zhang, "A case study of model checking retail banking system with SPIN," *Journal of Computers*, vol. 7, no. 10,pp.2503-2510, 2012.

[4] Wei Zhang, Wenke Ma, Huiling Shi, and Fuqiang Zhu, "Model checking and verification of the internet payment system with SPIN," *Journal of Software*, vol. 7, no. 9, pp.1941-1949, 2012.

[5] Gueffaz M, Rampacek S and Nicolle C, "Temporal logic to query semantic graphs using the model checking method," *Journal of Software*, vol. 7, no. 7, pp.1462-1472, 2012.

[6] Jun Niu,Guosun Zeng and Wei Wang, "An approach of model checking time or space performance," Chinese *Journal of Computers*, vol. 33, no. 9, pp.621-1633, 2010.

[7] Junhua Zhang, Zhiqiu Huang and Zining Cao, "Counterexample for timed probabilistic reachability in uniform CTMDP," *International Seminar on IEEE*, pp.612-615, 2008.

[8] H. Aljazzar and S. Leue, "Generation of counterexamples for model checking of Markov decision processes," *International Conference on the Quantitative Evaluation of Systems*, pp.197-206, 2009.

[9] Ballarini P, Mardare R and Mura I, "Analysing biochemical oscillation through probabilistic model checking". *Electr. Notes Theor. Comp. Sc*, vol. 229, no. 1, pp.3-19, 2009.

[10] L. Zhang, D. N. Jansen, F. Nielson and H. Hermanns, "Automata-based CSL model checking," *Proceedings of 38th International Colloquium on Automaton, Languages and Programming*, pp.271-282, 2011.

[11] Gruber H, Holzer M and Tautschnig M, "Short regular expressions from finite automata: empirical results," *Implementation and Application of Automata*, pp.188-197, 2009.

[12] T. Han, J.-P. Katoen and B. Damman, "Counterexample generation in probabilistic model checking," *IEEE TSE*, vol. 35, no. 3, pp.241-257, 2009.

[13] Junhua Zhang, Zhiqiu Huang and Zining Cao, "Counterexample generation for probabilistic timed automata model checking," *Journal of Computer Research and Development*, vol. 45, no. 10, pp.1638-1645, 2008.

[14] Jing Wang and Guangquan Zhang, "Counterexample representation for probabilistic timed automata model checking," *Journal of Soochow University(Natural Science Edition)*, vol. 27, no. 2, pp.36-43, 2011.

[15] Andrés M E and Van Rossum P, "Conditional probabilities over probabilistic and nondeterministic systems," *Tools and Algorithms for the Construction and Analysis of Systems*, pp.157-172, 2008.

[16] Yang Gao, Ming Xu, Naijun Zhan and Lijun Zhang, "Model checking conditional CSL for continuous-time Markov chains," *Information Processing Letters*, pp.44-55, 2013.

[17] Kwiatkowska M, Norman G and Parker D, "Advances and challenges of probabilistic model checking," *2010 48th Annual Allerton Conference on. IEEE*, pp.1691-1698, 2010.

[18] Conghua Zhou, Zhifeng Liu and Changda Wang, "Bounded model checking for probabilistic computation tree logic". *Journal of Software*, vol.23, no.7, pp.1656-1668, 2012.

[19] Daws C., "Symbolic and parametric model checking of discrete-time Markov chains," *Theoretical Aspects of Computing-ICTAC*, pp. 280-294, 2004.

[20] Damman, Berteun, Tingting Han and J-P. Katoen, "Regular expressions for PCTL counterexamples," *Quantitative Evaluation of Systems, QEST'08*, pp.179-188, 2008.

Mingyu Ji was born in 1980, Harbin, Heilongjiang province, China. He received the computer application technique master degree in 2004 from Harbin Engineering University, china. Now he is currently a Ph.D. candidate in computer science and technology in

Harbin Engineering University and works as a lecturer in Northeast Forestry University, china. His research interests include specification and verification of probabilistic and stochastic systems and model checking.

Yanmei Li was born in Harbin, Heilongjiang, P.R. China, 1980. She received her master's degree in computer science from Harbin Engineering University in 2008, and she is now pursuing her doctor's degree in College of computer science and technology, Harbin Engineering University. Her research interests include model checking, temporal logic.

Zhiyuan Chen is a Ph.D. candidate in computer science and technology in Harbin Engineering University. He received his B.S. and M.S. degrees from Jilin University in 2002 and 2005 respectively, both in computational mathematics. He is currently a lecturer in the College of Computer Science and Technology, Harbin Engineering University. His research interests include model checking and model logic.