# Performance Analysis of $\chi$ Models using Discrete-Time Probabilistic Reward Graphs

N. Trčka, S. Georgievska, J. Markovski[1], S. Andova[2], and E.P. de Vink

Department of Mathematics and Computer Science

Technische Universiteit Eindhoven

P.O. Box 513, NL-5600 MB Eindhoven, The Netherlands

**Abstract**

We propose the model of discrete-time probabilistic reward graphs (DTPRGs) for performance analysis of systems exhibiting discrete deterministic time delays and probabilistic behavior, via their interpretation as discrete-time Markov reward chains. We build on the $\chi$ environment, a full-fledged platform for qualitative and quantitative analysis of timed systems based on the modeling language $\chi$. The extension proposed in this paper is based on timed branching bisimulation reduction followed by inclusion of probabilities and rewards. The approach is applied in an industrial case study of a turntable drilling system. The resulting performance measures are shown to be comparable to those obtained by two other methods of the $\chi$ environment, viz. simulation and continuous-time Markovian analysis.

## I. Introduction

The $\chi$ language [1] is a modeling language for control and analysis of of industrial systems (machines, manufacturing lines, warehouses, factories, etc.). It has been successfully applied to a large number of industrial cases, such as a car assembly line, a multi-product, multi-process wafer fab [2], a beer brewery, a fruit juice blending and packaging plant [3], and process industry factories [4]. Initially, $\chi$ came equipped with features for the modeling of discrete-event systems only, and was not supported by a formal semantics. Recently, it has been redesigned and converted to a formal specification language [5]. Currently, $\chi$ can be characterized as a process algebra with data. In addition, it was extended to handle both discrete-event and continuous aspects, allowing for the modeling of hybrid systems [1].

Originally, simulation was the only means to analyze $\chi$ models. For the verification of functional requirements, however, simulation renders insufficient. Although it can, for instance, reveal that a system has a deadlock or that the system may exhibit a specific behavior, it cannot show that the system is deadlock-free nor that it will always have the specific behavior.

[1]This research has been funded by the Dutch BSIK/BRICKS project AFM 3.2

[2]Corresponding author, e-mail `s.andova@tue.nl`.

Therefore, a new approach has been taken, connecting $\chi$ to state-of-the-art verification tools and techniques. Currently, a $\chi$ model can be compiled to the input language of a number of model checkers, including SPIN [6], [7], $\mu$CRL [8], [9] and UPPAAL [10], [11] (see Fig. 1). The translated model can subsequently be checked against the functional properties formulated in the target setting.

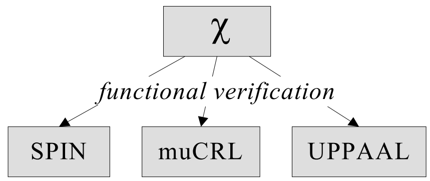


Fig. 1. Verification environment for $\chi$

Successful verification is usually succeeded by performance analysis and design optimization. At present, performance analysis of a $\chi$ model can be carried out either by simulation, or by analysis of the underlying continuous-time Markov (reward) chain [12], CTMC for short (see Fig. 2). Simulation is a powerful method for performance analysis, but its disadvantages in comparison to analytical methods are well-known [13]. The approach based on CTMC turns $\chi$ into a powerful stochastic process algebra [14], [15]. It is analytical, and builds on a vast and well-established theory. However, the generation of a CTMC from a $\chi$ model requires that all delays in the system are exponentially distributed. This is a serious drawback since in industrial systems, particularly in controllers, delays are often closer to being deterministic. Moreover, the state space of the generated CTMC is usually large, due to the interleaving of stochastic transitions. The later problem especially appears when deterministic delays are approximated by sequences of exponential delays into so-called phase-type distributions [16].

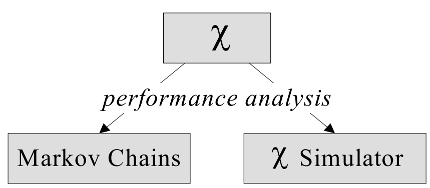


Fig. 2. Current performance analysis environment for $\chi$

In this paper, we propose a model in which time delays are discrete and deterministic, while random behavior can be expressed in terms of immediate probabilistic choices. This model is referred to as *discrete-time probabilistic reward graphs*, DTPRGs for short. We define a method for obtaining performance measures of a DTPRG by transforming it to a discrete-time Markov reward chain [17], DTMRC for short. We augment the $\chi$ environment so that for a given $\chi$ specification, the corresponding DTPRG can be obtained automatically. Usually, in contrast to the CTMC approach, the DTPRG generated from a $\chi$-model is considerably smaller (more than three times for our case study). The time itself does not determine the outcome of a choice and, as such, interleaving of timed transitions does not occur [18]. As an illustration, a case study is discussed on the performance of a turntable drilling system. Although relatively small, this system is realistic and incorporates many complex modeling issues. The case has been studied previously to illustrate the verification techniques of functional requirements [5], [19]. We put our performance results exploiting DTPRGs in perspective, by comparing them to results from simulation and the approach exploiting CTMCs.

## II. DTPRGs

In this section we introduce the notion of a DTPRG, and give, regarding performance, two equivalent DTMRC interpretations: one that is straightforward and general and another one less intuitive and less general, but computationally more efficient.

*Definition 1:* A DTPRG is a tuple $\mathsf{G} = (S, \rightsquigarrow, \mapsto, \rho)$, where (1) $S = S' \cup S''$, for two finite disjoint sets $S'$ and $S''$ of *probabilistic and timed states*, respectively, (2) $\rightsquigarrow \subseteq S' \times [0,1] \times S$ is the *probabilistic transition relation*, (3) $\mapsto \subseteq S'' \times \mathbb{N}^+ \times S$ is the *timed transition relation* such that $(s, n, s'), (s, m, s'') \in \mapsto$ implies $n = m$ and $s' = s''$, and (4) $\rho \colon S \to \mathbb{R}$ is the *reward rate assigning function*.

The interpretation of a DTPRG is as follows. In probabilistic states the process spends no time and it jumps to a next state chosen according to the probabilistic transition relation. In a timed state the process spends as many time units as specified by the timed transition relation and jumps to the unique subsequent state. The uniqueness requirement is to support the natural time determinism property [18]. In every state a reward is gained per time unit, as determined by the reward rate assigning function. Note that, although we allow reward rates to be assigned also to probabilistic states, the process actually gains no reward as it spends no time in them. We use infix notation and write, e.g., $s \overset{p}{\rightsquigarrow} s'$ rather than $(s, p, s') \in \rightsquigarrow$.

We visualize a DTPRG as in Fig. 3a. For this DTPRG states $1$, $2$, and $3$ are timed, whereas states $4$ and $5$ are probabilistic. The reward rates are put in italics in the top right corner of each state; the reward rate of the state $i$ is $r_i$, for $1 \leq i \leq 5$.

### A. From DTPRG to DTMRC

Most performance measures that we would like to obtain can be standardly defined as, for example, the percentage of time the system spends in some state, the throughput or a utilization of a transition, etc. To obtain these measures we exploit a translation from

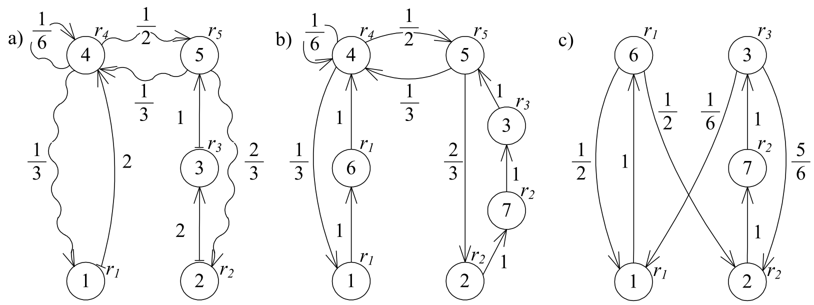

Fig. 3. a) A DTPRG, b) its unfolding, and c) aggregated unfolding

DTPRGs into DTMRCs, as the latter are well-established models for performance analysis.[1] For the definition and properties of DTMRCs we refer to the standard literature (e.g. [17]). A DTMRC is a triple $\mathsf{M} = (S, \rightarrow, \rho)$, where $S$ is a finite set of states, $\rightarrow$ is the probabilistic transition relation over $S$, and $\rho$ is the reward assigning function.[2] Operationally, a DTMRC is considered to wait one time unit in a state, gain the reward for this state determined by the function $\rho$, and then make an immediate step to another state with a probability specified by the relation $\rightarrow$. When required by the context, we also often standardly represent a DTMRC as a pair $(P, \rho)$, where $P$ is the probability transition matrix and $\rho$ is the state reward vector.

The main idea behind the translation from a DTPRG into a DTMRC is to represent a timed transition of duration $n$ in the DTPRG as a sequence of $n$ states in the DTMRC, connected by transitions labeled with probability 1, all having the same reward. Probabilistic transitions remain unchanged. We refer to this transformation as the *unfolding* of a DTPRG.

*Definition 2:* Let $\mathsf{G} = (S_\mathsf{G}, \rightsquigarrow, \mapsto, \rho_\mathsf{G})$ be a DTPRG with $S_\mathsf{G} = \{s_1, \ldots, s_n\}$. Associate with every state $s_i \in S_\mathsf{G}$ a number $m_i \in \mathbb{N}^+$ as follows: if $s_i$ is a probabilistic state, then $m_i = 1$; if $s_i$ is a timed state, then $m_i = m$ for the unique $m$ such that $s_i \overset{m}{\mapsto} s_k$, for some $s_k \in S_\mathsf{G}$. The *unfolding* of $\mathsf{G}$ is the DTMRC $\mathsf{M} = (S_\mathsf{M}, \rightarrow, \rho_\mathsf{M})$ with $S_\mathsf{M} = \{s_{ij} \,|\, 1 \leq i \leq n, 1 \leq j \leq m_i\}$, $s_{i1} \xrightarrow{1} \ldots \xrightarrow{1} s_{im_i}$, $\rho_\mathsf{M}(s_{ij}) = \rho_\mathsf{G}(s_i)$, and $s_{im_i} \xrightarrow{1} s_{k1}$ if $s_i \overset{m}{\mapsto} s_k$ or $s_{i1} \xrightarrow{p} s_{k1}$ if $s_i \overset{p}{\rightsquigarrow} s_k$.

In order to stress the correspondence, the states in the DTMRC that relate to timed (probabilistic) states in the original DTPRG will be referred to as timed (probabilistic) states, and similarly for transitions.

The unfolding of the DTPRG from Fig. 3a is given by the DTMRC depicted in Fig. 3b. The unfolded timed delays originating from states 1 and 2 introduce the new states 6 and 7,

[1]An alternative, computationally less efficient, approach would be to consider DTPRGs as special instances of Semi-Markov Reward chains [20]

[2]Note that we abstract from the initial probability vector.

respectively.

Note that the DTMRC obtained by unfolding does not always truthfully represent the semantics of the original DTPRG, in the sense that probabilistic states are immediate in the DTPRG, whereas they take one unit of time in the DTMRC. For example, in the DTPRG from Fig. 3a, state $5$ can be reached from state $1$ with probability $\frac{1}{2}$ after a delay of $2$ time units (via $1 \overset{2}{\mapsto} 4 \overset{1/2}{\rightsquigarrow} 5$), whereas in the unfolded version this cannot be done in less than $3$ time units (that are required for a sojourn in the states $1$, $6$ and $4$). The solution to this problem is to eliminate the immediate probabilistic states appropriately. This elimination is achieved by the aggregation method initially developed in the setting of continuous-time Markov processes [21]–[23]. Intuitively, this method computes the probabilities of reaching one timed state from another and adjusts the delays. Technically, the process of aggregation is as follows: In a DTMRC $\mathsf{M} = (P, \rho)$ the transition probability matrix $P$ is represented as $P = P_t + P_p$, where $P_t$ holds the unfolded timed transitions and $P_p$ holds the immediate probabilistic transitions. Next, the Cesaro sum of $P_p$ is computed and its canonical product decomposition $(L, R)$ is found [21], [23]. The *aggregated* chain is defined by $\hat{\mathsf{M}} = (LP_tR, L\rho)$.

The DTMRC from Fig. 3b is aggregated to the one from Fig. 3c. The aggregation 'splits' the transitions of states $6$ and $3$ according to the exiting probabilities of the loop between states $4$ and $5$. The splitting of states conforms to the Markovian semantics, i.e. after a delay of one time unit there is an immediate probabilistic choice. It is straightforwardly observed that this DTMRC models the same system as the DTPRG in Fig. 3a when observed in the states $1$, $2$ and $3$.

## B. *Performance metrics*

Having mapped a DTPRG to a DTMRC, we can use the standard theory and tools to compute all the desired performance measures. For the present paper however, we focus on the long-run behavior of systems, i.e. when the system stabilizes in the steady state, and on one particular measure called *the long-run expected reward rate* (long-run reward for short). This measure is powerful enough to obtain most of the interesting performance properties. If the resulting DTMRC is ergodic,[3] the long-run reward is standardly computed as $R = \pi\rho$ , where $\pi$ is the long-run probability vector (in Cesaro sense), and $\rho$ is the state reward vector. The full process of obtaining the performance measures of a DTPRG is visualized by the left branch in Fig. 5.

The performance measure of the DTPRG depicted in Fig. 3a is thus obtained by computing the long-run probability vector $\pi$ of the DTMRC from Fig. 3c. This vector is $\pi =$

[3]Note that in case the resulting process is not ergodic, we can always partition the original DTPRG into subgraphs that produce ergodic processes and analyze them separately, so we do not consider this case as restrictive to our analysis.

$\left(\frac{1}{11}\ \frac{3}{11}\ \frac{3}{11}\ \frac{1}{11}\ \frac{3}{11}\right)$, where states $6$ and $7$ are renamed in the matrix notation to states $4$ and $5$. The reward vector $\rho$ equals $\left(r_1\ r_2\ r_3\ r_1\ r_2\right)$, so $R = \frac{2}{11}r_1 + \frac{6}{11}r_2 + \frac{3}{11}r_3$.

### *C. Optimization by geometrization*

Note that the unfolded DTMRC has, in general, substantially more states than the original DTPRG, as every delay of duration $n$ introduces $n{-}1$ new states. This means that the unfold & aggregate method, although straightforward to serve as a definition, leads to computations on large state spaces. In the rest of this section, we optimize our method, using 'geometrization' of time delays to obtain a DTMRC of the same size as the original graph. The main idea is to replace discrete delays by geometrically distributed delays instead of unfolding them.

*Definition 3:* A DTPRG $\mathsf{G} = (S, \mapsto, \rightsquigarrow, \rho)$, is *geometrized* to the DTMRC $\mathsf{M} = (S, \rightarrow, \rho)$, if (1) for each timed transition $s \overset{n}{\mapsto} s'$ in $\mathsf{G}$ we have $s \xrightarrow{1/n} s'$ and $s \xrightarrow{(n-1)/n} s$ in $\mathsf{M}$; and (2) for each probabilistic transition $s \overset{p}{\rightsquigarrow} s'$ in $\mathsf{G}$ we have $s \overset{p}{\rightarrow} s'$ in $\mathsf{M}$.

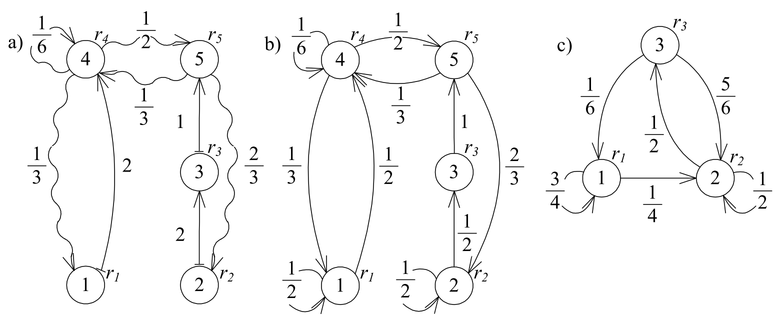


Fig. 4. a) A DTPRG, b) its geometrization, and c) aggregated geometrization

Consider again the DTPRG from Fig. 3a, now repeated in Fig. 4a. In Fig. 4b its geometrized DTMRC is shown. For the same reason as before this DTMRC still needs to be aggregated; the result is depicted in Fig. 4c.

The geometrize & aggregate method is depicted by the right branch in Fig. 5. The following theorem justifies the figure by showing that the two methods indeed commute, i.e. both give DTMRCs with the same long-run rewards.[4]

*Theorem 1:* Let $\mathsf{M}_1$ and $\mathsf{M}_2$ be the unfolded & aggregated, and the geometrized & aggregated DTMRC of the same DTPRG, respectively. Let $R_1$ and $R_2$ be the long-run rewards of $\mathsf{M}_1$ and $\mathsf{M}_2$, respectively. Then $R_1 = R_2$.

[4]Note that geometrization method is correct only for long-run analysis.

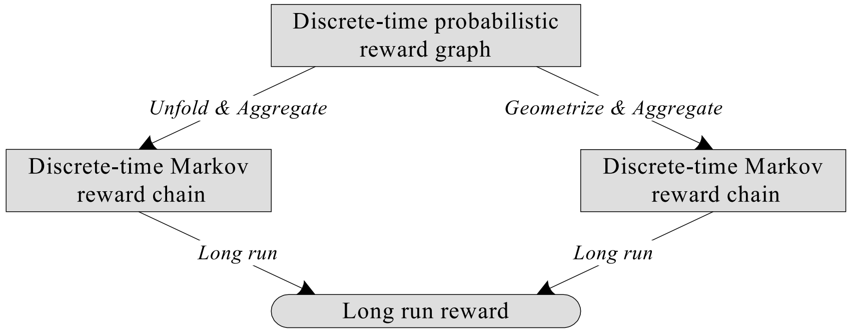


Fig. 5. Performance measuring for DTPRGs

The proof of the theorem can be found in the appendix. Here, we illustrate the result by an example. The long-run probability vector $\pi'$ of the DTMRC in Fig. 4c is $\pi' = \left( \frac{2}{11}\ \frac{6}{11}\ \frac{3}{11} \right)$. Its reward vector is $\rho' = \left( r_1\ r_2\ r_3 \right)$, and so its long-run reward $R' = \frac{2}{11}r_1 + \frac{6}{11}r_2 + \frac{3}{11}r_3$ coincides with the $R$ of the DTMRC from Fig. 3c.

## III. Case study using $\chi$: Performance analysis of a turntable drilling system

In this section we apply DTPRGs in an industrial case study by measuring performance of a small drilling system. We first explain the system and how it is modeled in $\chi$. We then show how to extend the $\chi$ environment to support, in particular generate, DTPRGs. Finally, following the methods discussed above, we calculate some relevant measures such as throughput and utilization of the system.

### *A. Description of the system*

The turntable drilling system is a concrete example of a small but realistic manufacturing system [5], [19]. Its purpose is to make holes in products. The system consists of a round turntable and devices for adding, drilling, testing, and removing a product. The turntable has four slots and transports the products counterclockwise (see Fig. 6). The drilling device consists of a drill and a clamp. The drill makes a hole in the product, whereas the clamp is used to lock the product while drilling. The testing device measures the depth of the hole in the drilled product. If its reaches its down position, meaning that the product is drilled sufficiently deep, the test result of the product is good. In that case the product is removed, when it gets to the removing position in the next rotation of the table. Otherwise, it stays in the system to be drilled again. The turntable can treat up to four products at the same time, doing the operations in parallel.

The various operations are modeled to require a fixed amount of time. The system takes $3$ time units to add a product, and $2$ time units to remove a product. The clamp needs $2$ time

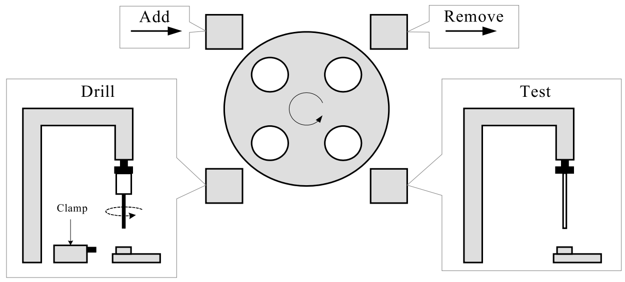


Fig. 6. The turntable drilling system

units to lock or unlock a product. The drilling operation takes $3$ time units, returning the drill to its up position takes $2$ time units. Testing and returning the tester to its initial (up) position require $2$ time units each. For performance analysis, we make the assumption that the adding and the drilling device behave probabilistically; these actions are succesful with a certain probability. We assume that when the systems is about to add a new product, it is available with a certain probability. Thus, adding a product may fail. Also, a product is drilled correctly with a certain probability, hence incorrectly drilled with the complementary probability.

### *B. $\chi$ model*

For an introduction to $\chi$ we refer to [1], [5]. Here, we only illustrate the features of the language by presenting the $\chi$ specification of the control system for the testing device:

```
Tester_Control( cStartTest, cTesterUpDone,
   cTesterDownDone, cTested, cTesterUpDown : chan ) =
|[ x, TstRes: bool |
   *( cStartTest?x;
      cTesterUpDown!true;
      ( cTesterDownDone?TstRes |
        delay 2.5; TstRes:=false );
      cTesterUpDown!true;
      cTesterUpDone?x;
      cTested!TstRes
    )
]|
```

The process `Tester_Control` receives a command to perform testing from the main control via the channel `cStartTest`. It then instructs the tester to go down via channel `cTesterUpDown`. If the tester has reached its down position within $2.5$ time units (recall that it needs $2$ time units if the hole is properly drilled), the test of the product is considered succesful (the input action `cTesterDownDone?TstRes` sets `TstRes` to true). If the sensor does not react within the $2.5$ time units available, the controller marks the test result as bad. This timeout is modeled by means of the non-deterministic choice '`|`' and uses the time deterministic semantics of $\chi$ according to which alternatives must always delay together. At the end, the controller instructs the tester to go up, along the channel `cTesterUpDown`, waits for the acknowledgement over the channel `cTesterUpDone`, and sends the test result to the main control via the channel `cTested`. The cycle then repeats itself which is modeled by the iterative construct '`*`'.

### *C. From $\chi$ to DTPRG*

The standard semantics of $\chi$ is in terms of timed transition systems [24]. The main idea underlying the construction of a DTPRG from a timed transition system, as proposed here, is to hide all actions, i.e., to rename them to the special internal action $\tau$, and then use the concept of timed branching bisimulation [18], [22] to reduce the system while abstracting from its internal transitions. If there is no real non-determinism in the model, a timed transition system without any action labeled transition is obtained, i.e., a DTPRG without probabilistic transitions. If there is one or more non-deterministic transitions left, then the system is underspecified, and its performance cannot be measured in the standard way.

Since $\chi$ has no features to model probabilistic choice,[5] the random behavior of the adding and drilling devices is modeled in $\chi$ by a non-deterministic choice. When the corresponding DTPRG is generated from the $\chi$ model these non-deterministic choices must be appropriately replaced by probabilistic ones. For this we slightly adjust the method described in the previous paragraph. Instead of hiding all actions, the special actions used to indicate probabilistic branching are not hidden. After the minimization, the probabilities that were intentionally left out are put as labels on the non-deterministic transitions. Again, if there is still non-determinism remaining in the model, we cannot proceed with performance analysis. Note that although the method is not always sound (in case of multiple probabilistic transitions from the same state) as it requests manipulation on the resulting graph, it serves its purpose for this example. Of course, another approach is to extend $\chi$ with an explicit probabilistic choice operator (e.g. the one in [25]). However, this requires drastic changes of the language and tools, and as such goes beyond the scope of this paper.

For the turntable system the reliability of the drill is captured in the tester process by a non-deterministic choice between sending the signal along the channel `cTesterDownDone`

[5]Strictly speaking, it has, but only for simulation purposes.

or doing the dummy action `skip`. Similarly, the availability of a product is captured in the process modeling the adding device.

The $\chi$ language does not directly support reward specification either. We take a similar approach as for the absence of a probabilistic choice, and add rewards by manipulating the $\chi$ specification (again side-stepping changes in $\chi$). We add, for each reward criterion, an ever repeating parallel component to the specification. The result is that in the timed transition system yielded, every state has a self-loop labeled by a special action denoting the reward rate of the state. These actions will not be hidden in the ranching bbisimulation reduction process and, therefore, persist in the resulting DTPRG. As in the case for the probabilistic choice, a systematic technique rendering the above can in principle be incorporated into the $\chi$ environment.

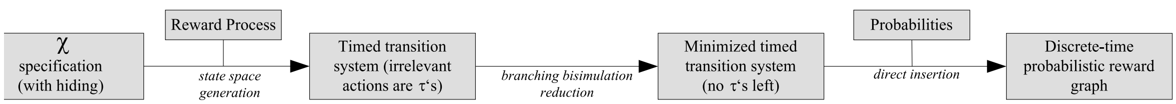


Fig. 7. $\chi$ to DTPRG

The complete pipeline of generating DTPRGs from $\chi$ specifications is illustrated in Fig. 7. Currently we employ scripts tweaked into the $\chi$ environment that insert probabilities and rewards, in order to automatically produce the desired DTPRG from a given $\chi$ specification.

### *D. Performance analysis of the drilling system*

We perform quantitative analysis of the turntable drilling system by applying the method proposed and we compare the results with those obtained from simulation and CTMC analyses. We consider the following performance measures: (1) throughput, i.e. the number of products that leave the system per time unit; (2) utilization of the drilling machine, i.e. the percentage of time that the drill is actually drilling; and (3) the average number of products in the system. All measures are considered in the long-run.

In order to obtain the above measures, we assign rewards as follows. For throughput, we put the reward rate of $\frac{1}{2}$ only to the states in which the removal operation is performed. This is because in $2$ time units, $1$ product is removed from the system. To obtain utilization, we give a reward rate of $1$ to the states where drilling is performed. Finally, for the average number of products, every state is given a reward rate equal to the number of products present in the system when residing in that state.

The result of performance analysis is presented in Fig. 8, where each measure is represented as a function of the reliability of the drill and the availability of products. In Fig. 9 we additionally give a comparison to the results obtained by simulation and CTMC analysis, when the probability for availability of a product is set to $0.5$. We note that the model required

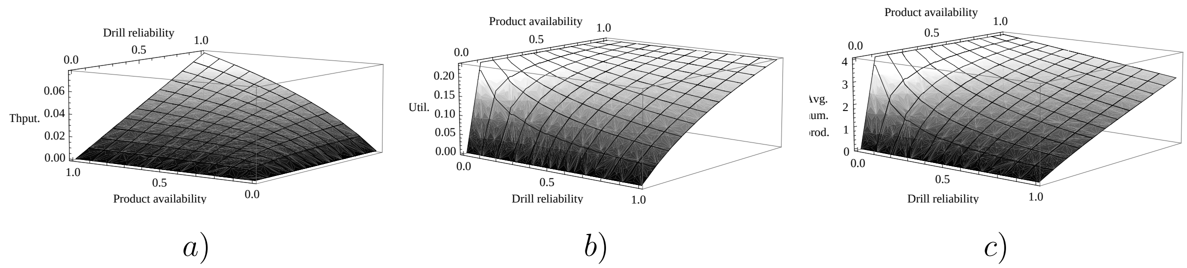


Fig. 8. a) Throughput, b) utilization, and c) average number of products of the turntable drilling system

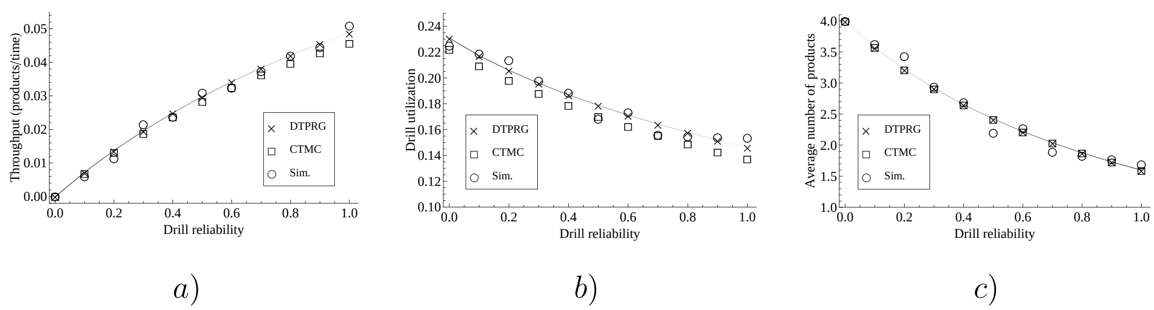


Fig. 9. Comparing the DTPRG method to CTMC analysis and simulation with product availability of $0.5$

for the degeneration of the DTPRG had $19023$ states before reduction, whereas the DTPRG itself has $164$; the model required for the CTMC analysis had $65529$ states that reduced to a CTMC with $360$ states by the weak Markovian bisimulation reduction of [14].

Note that the CTMC analysis gives the worst performance measures. This is anticipated, because the expected value of the maximum of two exponential delays is greater than the expected values of both delays, which increases the average cycle length of the system. In the context of this paper we shortened the simulation experiments. For that reason, the simulation results do not align perfectly with the ones of the DTPRG analysis.[6]

## IV. Conclusion

We have introduced a mathematical model, called discrete-time probabilistic reward graphs, abbreviated DTPRGs, for performance measuring of systems featuring deterministic delay and probabilistic choice. We have extended the $\chi$-environment to a prototype that supports the new model, enabling an effective qualitative and quantitative analysis of timed systems within the same framework. We have applied our method on a turntable drilling device, a small but

[6]For simulation, we used the batch means method (cf. [13]), where each experiment lasted for $5000$ time units.

realistic industrial system. The results are shown to be comparable to those obtained by other methods in $\chi$.

As future work we schedule the extension of the $\chi$ language to fully support the developed theory, relieving the script-based short-cuts taken presently intervening at proper place in the tool environment. We foresee that this can be achieved by introducing a probabilistic choice operator, and by facilitating the assignment of rewards in the toolset.

*Related Work.* The modeling of deterministic-time systems with probabilities as Markov chains has been studied previously, for different settings, in Petri net theory (see, e.g. [26]) and process algebras (see, e.g. [25]). Our method differs in its incorporation of rewards, and that it is based on timed branching bisimulation reduction combined with the aggregation method for elimination of immediate transitions in Markov chains.

*Acknowledgement.* We would like to thank Bas Luttik for extensive comments on the draft of this paper.

## REFERENCES

[1] D. A. v. Beek, K. L. Man, M. A. Reniers, J. E. Rooda, and R. R. H. Schiffelers, "Syntax and consistent equation semantics of hybrid Chi," *Journal of Logic and Algebraic Programming*, vol. 68, pp. 129–210, 2006.

[2] E. J. J. van Campen, "Design of a multi-process multi-product wafer fab," Ph.D. dissertation, Eindhoven University of Technology, 2000.

[3] J. J. H. Fey, "Design of a fruit juice blending and packaging plant," Ph.D. dissertation, Eindhoven University of Technology, 2000.

[4] D. A. van Beek, A. van der Ham, and J. E. Rooda, "Modelling and control of process industry batch production systems," in *IFAC'02*, Barcelona, 2002.

[5] V. Bos and J. J. T. Kleijn, "Formal specification and analysis of industrial systems," Ph.D. dissertation, Eindhoven University of Technology, 2002.

[6] G. J. Holzmann, "The model checker SPIN," *IEEE Transactions on Software Engineering*, vol. 23, pp. 279–295, 1997.

[7] N. Trčka, "Verifying $\chi$ models of industrial systems in Spin," in *ICFEM'06*. LNCS 4260, 2006, pp. 132–148.

[8] S. Blom, W. Fokkink, J. F. Groote, I. van Langevelde, B. Lisser, and J. C. van de Pol, "$\mu$CRL: A toolset for analysing algebraic specifications," in *CAV 2001*. LNCS 2102, 2001, pp. 250–254.

[9] A. J. Wijs, "From $\chi_t$ to $\mu$CRL: Combining performance and functional analysis," in *ICECCS'05*. IEEE Computer Society, 2005, pp. 184–193.

[10] K. G. Larsen, P. Pettersson, and W. Yi, "Uppaal in a nutshell," *STTT*, vol. 1, pp. 134–152, 1997.

[11] E. M. Bortnik, D. A. van Beek, J. M. van de Mortel-Fronczak, and J. E. Rooda, "Verification of timed Chi models using UPPAAL," in *ICINCO*, Barcelona, 2005, pp. 486–492.

[12] V. G. Kulkarni, *Modeling and Analysis of Stochastic Systems*. Chapman & Hall, 1996.

[13] J. Banks, J. S. Carson II, B. L. Nelson, and D. M. Nicol, *Discrete-event system simulation*. Prentice Hall, 2000.

[14] H. Hermanns, *Interactive Markov Chains: The Quest for Quantified Quality*. LNCS 2428, 2002.

[15] J. Hillston, *A Compositional Approach to Performance Modelling*. Cambridge University Press, 1996.

[16] M. F. Neuts, *Matrix-geometric solutions in stochastic models, an algorithmic approach*. John Hopkins University Press, 1981.

[17] J. G. Kemeny and J. L. Snell, *Finite Markov chains*. Springer, 1976.

[18] J. C. M. Baeten, J. A. Bergstra, and M. A. Reniers, "Discrete time process algebra with silent step," in *Proof, language, and interaction: essays in honour of Robin Milner*. MIT Press, 2000, pp. 535–569.

[19] E. M. Bortnik, N. Trčka, A. J. Wijs, S. P. Luttik, J. M. van de Mortel-Fronczak, J. C. M. Baeten, W. J. Fokkink, and J. E. Rooda, “Analyzing a $\chi$ model of a turntable system using Spin, CADP and UPPAAL,” *Journal of Logic and Algebraic Programming*, vol. 65, pp. 51–104, 2005.

[20] R. Howard, *Semi-Markov and Decision Processes*. Wiley, 1971.

[21] M. Coderch, A. Willsky, S. Sastry, and D. Castanon, “Hierarchical aggregation of singularly perturbed finite state Markov processes,” *Stochastics*, vol. 8, pp. 259–289, 1983.

[22] N. Trčka, “Silent steps in transition systems and Markov chains,” Ph.D. dissertation, Eindhoven University of Technology, 2007.

[23] J. Markovski, A. Sokolova, N. Trčka, and E. de Vink, “Compositionality for Markov reward chains with fast transitions,” in *EPEW’07*. LNCS 4748, 2007, pp. 18–32.

[24] J. Baeten and C. Middelburg, *Process Algebra with Timing*. Springer, 2002.

[25] H. A. Hansson, *Time and Probability in Formal Design of Distributed Systems*. Elsevier, 1994.

[26] W. M. Zuberek, “Timed Petri nets and preliminary performance evaluation,” in *ISCA’80*. ACM Press, 1980, pp. 88–96.

## APPENDIX

This appendix gives the notions and intermediate results required for the proof of the central theorem in the paper, viz. Theorem 1, restated below:

*Theorem 1:* Let $\mathsf{M}_1$ and $\mathsf{M}_2$ be the unfolded & aggregated, and the geometrized & aggregated DTMRC of the same DTPRG, respectively. Let $R_1$ and $R_2$ be the long-run rewards of $\mathsf{M}_1$ and $\mathsf{M}_2$, respectively. Then $R_1 = R_2$.

First, we define the notion of the unfolding set of a timed state. Recall that the unfolding of a timed transition $s_i \overset{n}{\mapsto} s_k$ of a DTPRG is given by the sequence $s_{i1} \xrightarrow{1} \ldots \xrightarrow{1} s_{in} \xrightarrow{1} s_{k1}$ in its unfolded DTMRC, as stated in Definition 2. The *unfolding set* of $s_i$ is given by *us*$(s_i) = \{\, s_{ij} \,|\, 1 \leq j \leq m_i \,\}$. The following lemma shows that all states in the unfolding set have the same long-run probability vector.

*Lemma 1:* Let $\pi$ be the long-run probability vector of the DTMRC obtained by unfolding and aggregation of a DTPRG $\mathsf{G}$. Then for every state $k \in \mathsf{G}$ and $i, j \in$ *us*$(k)$ it holds $\pi[i] = \pi[j]$.

*Proof:* Let $P$ be the matrix of transition probabilities of $\mathsf{M}$. As $\pi$ is the long-run probability vector of $\mathsf{M}$, it holds that $\pi P = \pi$. Now, assume that $i, j \in$ *us*$(k)$, for some state $k$, are two subsequent states of the unfolding sequence of some timed transition, i.e. $i \xrightarrow{1} j$. Then, $P[i, j] = 1$ and $P[i', j] = 0$ for every $i' \neq i$, as there are no other incoming transitions in $j$. Now, it can be observed that $\pi[j] = \pi P^{(-,j)} = \pi[i]$, where $P^{(-,j)}$ denotes the $j$-th column of $P$. Hence $\pi[i] = \pi[j]$, for any two subsequent states in *us*$(k)$. From this the lemma follows. ∎

By using the unfolding sets, we can define the *unfolding collector matrix* that partitions the state space according to the unfolding sets of states. Given a partitioning $\{C_1, \ldots, C_N\}$ of the state space of a DTMRC, we distinguish the following matrices. The collector matrix $V$ defined as $V[i, j] = 1$ if $i \in C_j$, $V[i, j] = 0$ otherwise. The $j$-th column of $V$ has an entry $1$ for elements corresponding to states in $C_j$. A matrix $U$ such that $U \geq 0$ and $UV = I$, with $I$ denoting the identity matrix, is a distributor matrix for $V$. It can be readily seen that $U$ is actually any matrix of which the elements of the $i$-th row that correspond to elements in $C_i$ sum up to $1$, while the other elements of the row are $0$. However, in our case we will use the *uniform distributor*, where all non-zero elements in a row are equal.

Now, the unfolding collector is given by $V[i, j] = 1$ iff $j \in$ *us*$(i)$ and $V[i, j] = 0$, otherwise. The following corollary of Lemma 1 states that the long-run probability vector is invariant when multiplied by the unfolding collector and folded again by the uniform distributor.

*Corollary 1:* Let $\pi$ be a steady state vector of the unfolded and aggregated DTMRC of a DTPRG $\mathsf{G}$ and let $V$ be the unfolding collector of $\pi$ and $U$ be its uniform distributor. Then it holds that $\pi = \pi V U$.

*Proof:* Let, for any $i \in \mathsf{G}$, $k$ be the state such that $i \in \textit{us}(k)$. Then it holds that $(\pi V U)[i] = \sum_{h \in \textit{us}(k)} \pi[j] U[h,i] = \pi[i] \sum_{h \in \textit{us}(k)} U[h,i] = \pi[i] \cdot 1 = \pi[i]$. ■

The unfolding collector is of particular interest, because it allows a representation in matrix terms of the notion of a geometrized DTPRG as captured by Definition 3. Given a DTPRG $\mathsf{G}$, its unfolding $\mathsf{M}_1 = (P, \rho)$, the unfolding collector $V$, and the uniform distributor $U$, we have that the geometrization of $\mathsf{G}$ is given by $\mathsf{M}_2 = (UPV, U\rho)$. The following lemma states that the long-run probability vector of an unfolded and aggregated DTPRG $\mathsf{G}$ when multiplied by the unfolding collector equals the long-run probability vector of the geometrized and aggregated version of $\mathsf{G}$.

*Lemma 2:* Let the DTMRC $\mathsf{M} = (P, \rho)$ be the unfolding of a DTPRG. Say $P = P_t + P_p$, where $P_t$ is the transition matrix of the unfolded timed transitions and $P_p$ is the transition matrix of the immediate probabilistic transitions. Let $L, R$ be the canonical product decomposition of the Cesaro sum of $P_p$. Let $\mathsf{M}_1 = (LP_tR, L\rho)$ be the unfolded and aggregated DTMRC, let $\pi$ be its long-run probability vector and let $V$ be its unfolding collector. Let furthermore $V'$ be the unfolding collector of $\mathsf{M}$, $U'$ the uniform distributor, and let $(L', R')$ be the canonical product decomposition of the Cesaro sum of $U'P_pV'$. Then it holds that $\mathsf{M}_2 = (L'U'P_tV'R', L'U'\rho)$ is the geometrized and aggregated DTMRC, and $\pi V$ its long-run probability vector.

*Proof:* It can be straightforwardly checked that if a timed transition $s_i \overset{m}{\mapsto} s_j$ is unfolded to $s_{i1} \xrightarrow{1} \ldots \xrightarrow{1} s_{im} \xrightarrow{1} s_{j1}$, then the multiplication of the transition probability matrix $P$ by $U'$ on the left and by $V'$ transforms the sequence into $\textit{us}(s_i) \xrightarrow{1/m} \textit{us}(s_j)$ and $\textit{us}(s_i) \xrightarrow{(m-1)/m} \textit{us}(s_i)$. This corresponds to geometrizing the delays of the original DTPRG after the renaming of $\textit{us}(s)$ to $s$.

Without loss of generality, we assume that $\mathsf{G}$ has $k$ timed transitions, $\ell$ closed loops of probabilistic transitions and $m$ open loops or sequences of probabilistic transitions. They correspond to $t_1 + \ldots + t_k$ trivial ergodic classes of one element for the duration of the delays $t_1, \ldots, t_k$, $\ell$ ergodic classes with more than one element and $m$ transient states as referred to in [21]–[23]. To alleviate the computations, again without loss of generality, we assume a numbering of the states such that unfolding sets contain states with consecutive indices, after which we place the closed loops and, finally, the transient states. In such numbering $L$, $R$, $U$ and $V$ have the following form:

$$L = \begin{pmatrix} I_{t_1} & \dots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathbf{0} & \dots & I_{t_k} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \pi_1 & \dots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \pi_\ell & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0}_m \end{pmatrix} \quad R = \begin{pmatrix} I_{t_1} & \dots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathbf{0} & \dots & I_{t_k} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1}_{E_1} & \dots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{1}_{E_\ell} & \mathbf{0} \\ \begin{pmatrix} \delta_1 \mathbf{0} \end{pmatrix} & \dots & \begin{pmatrix} \delta_k \mathbf{0} \end{pmatrix} & d_1 & \dots & d_\ell & \mathbf{0}_m \end{pmatrix}$$

$$V = \begin{pmatrix} \mathbf{1}_{t_1} & \dots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathbf{0} & \dots & \mathbf{1}_{t_k} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & 1 & \dots & \mathbf{0} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & 1 \end{pmatrix} \quad U = \begin{pmatrix} u_{t_1} & \dots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ \mathbf{0} & \dots & u_{t_k} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & 1 & \dots & \mathbf{0} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & 1 \end{pmatrix}$$

where $I_{t_1}, \dots, I_{t_k}$ and $\mathbf{1}_{t_1}, \dots, \mathbf{1}_{t_k}$ denote identity matrices and vectors of $1$'s of the size of the duration of the $t_i$-th timed delay, $\mathbf{0}_m$ denotes a zero matrix of the size of the transient states, $\pi_1, \dots, \pi_\ell$ are the ergodic probability row vectors of the ergodic classes, $\mathbf{1}_{E_1}, \dots, \mathbf{1}_{E_l}$ are vectors of $1$'s of the size of the ergodic classes, $\delta_1, \dots, \delta_k$ are the transient probability vectors of the first states in the unfolding sequences, $\begin{pmatrix} \delta_i & \mathbf{0} \end{pmatrix}$ is a square matrix where the first column is $\delta_i$, $d_1, \dots, d_l$ are the transient probability vectors of the ergodic classes, where $u_{t_1}, \dots, u_{t_k}$ are positive stochastic row vectors.

Similarly, the matrices $U'$, $V'$, $L'$ and $R'$ have the following forms:

$$U' = \begin{pmatrix} u_{t_1} & \dots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathbf{0} & \dots & u_{t_k} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & I_{E_1} & \dots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & I_{E_\ell} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & I_m \end{pmatrix} \quad V' = \begin{pmatrix} \mathbf{1}_{t_1} & \dots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathbf{0} & \dots & \mathbf{1}_{t_k} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & I_{E_1} & \dots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & I_{E_\ell} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & I_m \end{pmatrix}$$

$$
L' = \begin{pmatrix}
1 & \dots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
\vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\mathbf{0} & \dots & 1 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \pi_1 & \dots & \mathbf{0} & \mathbf{0} \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \pi_\ell & \mathbf{0} \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0}_m
\end{pmatrix}
\qquad
R' = \begin{pmatrix}
1 & \dots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
\vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\mathbf{0} & \dots & 1 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1}_{E_1} & \dots & \mathbf{0} & \mathbf{0} \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{1}_{E_l} & \mathbf{0} \\
\delta_1 & \dots & \delta_k & d_1 & \dots & d_l & \mathbf{0}_m
\end{pmatrix}
$$

with $I_m$ denoting the identity matrix of the size of the transient states.

We claim that $\pi V$ is the long-run probability vector of the transition matrix $ULP_tRV$. To show this, we use Corollary 1 and the fact that $\pi$ is the long-run probability vector of $LP_tR$, and obtain $\pi V = \pi LP_tRV = \pi VULP_tRV$. Now, to prove the lemma it remains to show that $UL = L'U'$ and $RV = V'R'$. This is obtained by direct multiplication of the matrices given above, which completes the proof. ∎

We note that Lemma 2 actually states a stronger result than Theorem 1 as it gives a relation between the long-run probability vectors of the unfolded & aggregated and the geometrized & aggregated versions of the graph. Finally, the proof of Theorem 1 follows as a direct consequence.

*Proof:* (of Theorem 1) By Lemma 2, it holds that

$$R' = \pi V L' U' \rho = \pi V U L \rho = \pi L \rho = R$$

which was to be shown. ∎