Analysis of partial loss reward models and its application*

M. Telek[1], S. Rácz[2]

[1]Department of Telecommunications
Technical University of Budapest, 1521 Budapest, Hungary
[2]Ericsson Research, Hungary
telek@hit.bme.hu, Sandor.Racz@eth.ericsson.se

Abstract

This paper studies a class of Markov reward models where a portion of the accumulated reward is lost at the state transitions and provides the analytical description of this model class in Laplace transform domain. Based on the transform domain description a numerical method is introduced to evaluate the moments of transient cumulative reward. For the special case when the underlying CTMC is stationary a more effective analysis approach is proposed. The applicability of partial loss reward models and the proposed numerical analysis methods are demonstrated via the performance analysis of a computer system executing long running batch programs with checkpointing.

Keywords: Markov reward models, partial reward loss, reverse Markov chain, checkpointing.

1 Introduction

Reward models have been effectively used for performance analysis of real life computer and communication systems for a long time [11]. Reward models are composed of a discrete state continuous time stochastic process (referred to as systems state process or background process) describing the behavior of the studied system, and an associated reward function describing the performance measure of interest. It is most frequently assumed that the background process is a Continuous time Markov chain (CTMC), but there are analysis results available for Semi-Markov [9] and Markov regenerative [15] background processes as well. It is a common feature of the applied reward functions that during the sojourn of the background process in state i reward is accumulated at rate r_i ($r_i \geq 0$). The difference of the reward functions of different reward models lies in the effect of the state transition on the accumulated reward. Let T_n be the time of the nth state transition of the background process. The reward models studied in the literature so far, can be classified according to the effect of a state transition on the accumulated reward ($B(t)$):

- *preemptive resume (prs)*: the amount of accumulated reward is not affected by the state transition: $B(T_n^-) = B(T_n^+)$ (Fig. 1a) [9];

*This work was supported by OTKA Grant No. T-34972

- *preemptive repeat (prt)*: the accumulated reward is completely lost: $B(T_n^+) = 0$ (Fig. 1b) [2];
- *impulse reward model*: the accumulated reward is increased by a random quantity: $B(T_n^+) = B(T_n^-) + D_{ij}$ (Fig. 1c) [14];
- *partial loss model*: a portion of the accumulated reward is lost at the state transition: $0 \le B(T_n^+) \le B(T_n^-)$ (Fig. 1d).

The two subclasses of partial loss models are:

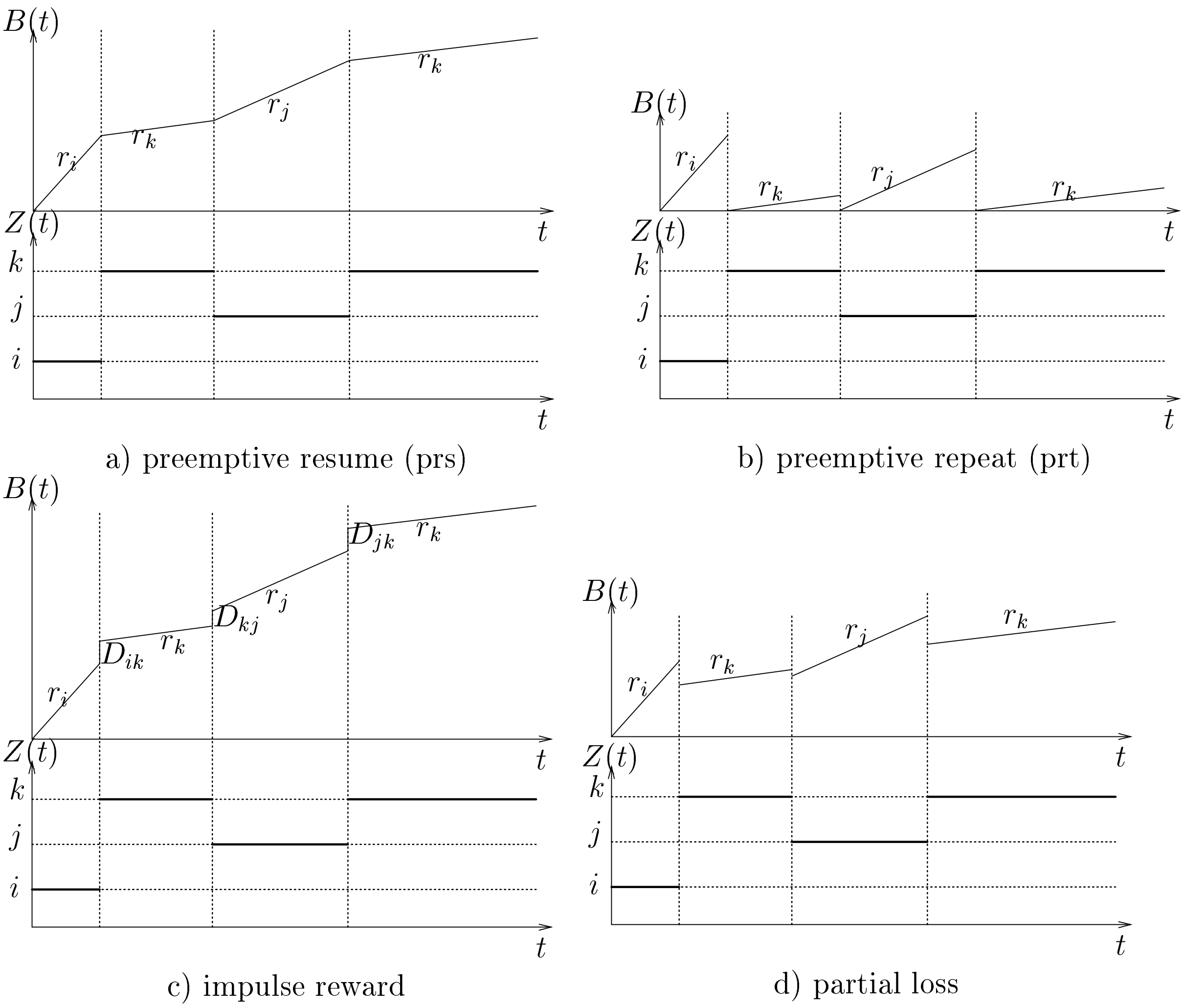
- *partial total loss* models: a fraction of the total accumulated reward is lost at state transitions, $B(T_n^+) = \alpha_i B(T_n^-)$, where $0 \le \alpha_i \le 1$ and i is the state of the system between T_{n-1} and T_n. [1]
- *partial incremental loss* models: the amount of lost reward is a fraction of the reward accumulated during the sojourn in the last state, $B(T_n^+) = B(T_{n-1}^+) + \alpha_i[B(T_n^-) - B(T_{n-1}^+)]$, with $0 \le \alpha_i \le 1$ [1, 13].

From the reward loss point of view the prs and the prt reward models are the two extreme cases of rate-reward models because the first one represents no reward loss at all and the second one represents complete loss of all previously accumulated reward. There are analytical results available for these two extreme cases. There was an evident need to handle intermediate cases as well, because real life systems often behave between these two extremes, but analytical results were not available. The classification and the analytical description of partial loss reward models were presented in [1, 13] and a computation method was proposed for the analysis of partial incremental loss models in [13]. In this paper we present two new computationally effective analysis methods for the evaluation of the accumulated reward of partial incremental loss models.

Several effective numerical procedures were proposed for the analysis of reward models mainly with CTMC background process and with prs reward accumulation [5, 6, 12, 16]. It turned out that the analysis of the distribution of the reward measures [5, 6, 12] is computationally much harder than the analysis of the moments of the same measures [16]. The moments of reward measures of prs Markov reward models can be calculated approximately at the same computational cost as the transient analysis of the background CTMC. In general, the analysis of partial reward loss models is more complex than the analysis of prs reward models. Both numerical analysis methods presented in this paper use the analysis of prs reward models as an elementary step of the procedure. To keep the overall computational cost as low as possible we calculate only the moments of the accumulated reward and apply the effective method presented in [16] for the embedded calculation of prs models.

There are two main classes of reward measures [8]. From the *system oriented* point of view the most significant measure is the total amount of reward accumulated by the system in a finite interval. This measure is often referred to as *performability*. From the *user oriented* (or *task oriented*) point of view the system is regarded as a server, and the emphasis of the analysis is on the time the system needs to accomplish an assigned task. Consequently, the most characteristic measure becomes the completion time. The numerical analysis of the first measure is considered in this paper.

The aim of this paper is to present numerical methods to evaluate partial loss models and to demonstrate the applicability of partial loss models in the analysis of computer systems executing long running batch programs with checkpointing.



a) preemptive resume (prs)

b) preemptive repeat (prt)

c) impulse reward

d) partial loss

Figure 1: Change of accumulated reward at state transitions

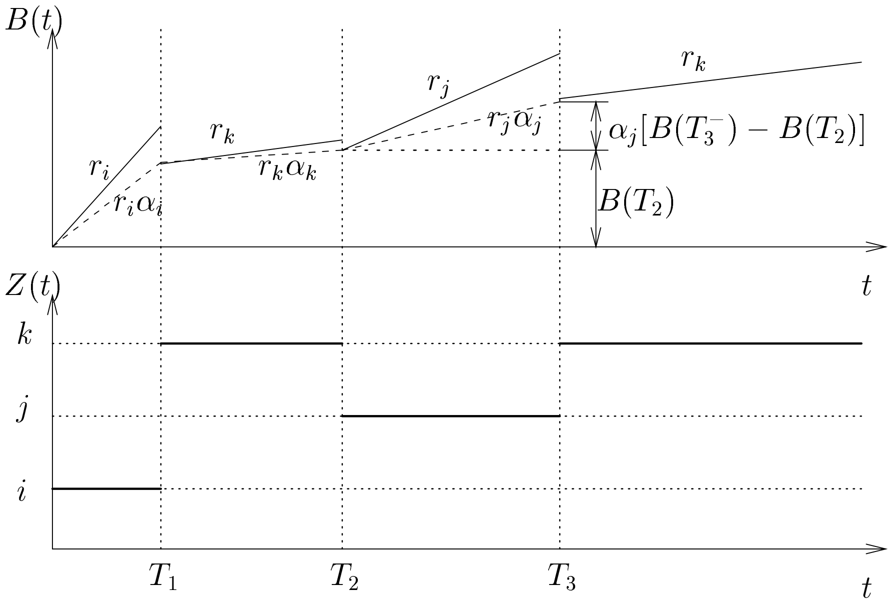


Figure 2: Reward accumulation in partial incremental loss model

The rest of the paper is organized as follows. Section 2 provides the analytical description of the accumulated reward of partial incremental loss models in double transform domain. Based on this double transform domain expression a numerical method is proposed in Section 3 for the analysis of the accumulated reward. There is a computationally hard step in the proposed numerical method, a numerical integration. A special numerical analysis method is proposed in Section 4 which is free of the computationally hard numerical integration, but this special method is applicable only when the background stochastic process is stationary. An application example of partial loss reward models is presented in Section 5. The numerical properties of the proposed numerical methods are also demonstrated here. Finally, the paper is concluded in Section 6.

2 Partial incremental loss in Markov reward models

Let $\{Z(t), t \geq 0\}$ be a continuous time Markov chain (CTMC) on state space $S = \{1, 2, ..., N\}$ with generator $Q = \{q_{ij}\}$ and initial probability vector $\underline{\pi}$. Whenever the CTMC stays in state i, reward is accumulated at rate r_i. r_i is a non-negative real number. When the CTMC undergoes a transition from state i to another state, the $1 - \alpha_i$ fraction of the reward obtained during the last sojourn in state i is lost and only the α_i fraction of the reward obtained during the last sojourn in i remains. α_i is a real number, such that $0 \leq \alpha_i \leq 1$. $B(t)$ denotes the amount of accumulated reward at time t. $B(t)$ is right continuous, i.e., $B(t) = B(t^+)$. Let T_n be the time of the nth transition in the CTMC. Then the dynamics of the right continuous process $\{B(t), t \geq 0\}$ can be described as follows (see Figure 2):

$$\frac{dB(t)}{dt} = r_{Z(t)} \qquad \text{for} \quad T_n < t < T_{n+1} \tag{1}$$

$$B(T_n) = B(T_{n-1}) + \alpha_{Z(T_n^-)}[B(T_n^-) - B(T_{n-1})] \tag{2}$$

Analysis of the accumulated reward

The state dependent distribution of the accumulated reward is defined as

$$P_{ij}(t,w) = Pr(B(t) \le w, Z(t) = j \mid Z(0) = i)$$

and $P(t,w) = \{P_{ij}(t,w)\}$.

Theorem 1 *The following double transform domain equation holds for $P(t,w)$:*

$$P^{*\sim}(s,v) = (sI + vR_\alpha - Q)^{-1} D(s,v) \tag{3}$$

where I is the identity matrix, $*$ denotes the Laplace transform with respect to t ($\rightarrow s$), $\sim$ denoted the Laplace-Stieltjes transform with respect to w ($\rightarrow v$) and the diagonal matrices R_α and $D(s,v)$ are defined as $R_\alpha = \mathrm{diag}\langle r_i\alpha_i \rangle$ and $D(s,v) = \mathrm{diag}\left\langle \dfrac{s + vr_i\alpha_i + q_i}{s + vr_i + q_i} \right\rangle$.

The proof of the Theorem is provided in Appendix A.

Theorem 1 is a general result that provides, as a special case, the previously known results for the prs and for the prt cases when $\forall \alpha_i, i \in S$ is set to 1 and 0, respectively. E.g., when $\alpha_i = 1, i \in S$ R_α becomes $R = \mathrm{diag}\langle r_i \rangle$ and $D(s,v)$ vanishes in (3).

The partial loss models are the transition between the prs (no reward loss) and the prt (complete reward loss) reward models. The numerical methods that are commonly used for the analysis of the prs and the prt reward models utilize the special features of those models and cannot be applied for the analysis of partial loss models.

The behavior of the partial incremental loss model can be interpreted as follows. The reward accumulation between 0 and t^* is according to a traditional prs model with reduced reward rates ($\alpha_i r_i$), and from time t^* the prs reward accumulation goes on with the original reward rates (r_i), where t^* ($0 \le t^* < t$) is the instant of the last state transition before t. If there is no state transition till time t, then $t^* = 0$. Unfortunately, t^* is a complex quantity (since it depends on the evolution of the CTMC over the whole $(0,t)$ interval) and it is hard to evaluate the partial loss models with effective numerical methods. The transform domain expression in eq. (3) reflects this model interpretation. Matrix $(sI + vR_\alpha - Q)^{-1}$ describes the distribution of the reward accumulated by a prs Markov reward model with generator Q and reward rates $\alpha_i r_i$ and the $D(s,v)$ diagonal matrix captures the effect of the "different" reward accumulations during the (t^*, t) interval.

As a consequence of this complex behavior the mean accumulated reward at time t cannot be evaluated based on the cumulative transient probabilities of the CTMC, as it was possible for the prs reward models.

3 Numerical evaluation of the accumulated reward

In this section we propose a numerical method to evaluate the accumulated reward of partial incremental loss models. The proposed method is based on Theorem 1 and the effective numerical method published in [16] that provides the moments of the accumulated reward of prs Markov reward models with a low computational cost and memory requirement.

To obtain a numerical procedure to evaluate the accumulated reward at time t, we inverse Laplace transform (3) with respect to the time variable ($s \rightarrow t$). First we introduce

$$F(s,v) = \mathrm{diag}\left\langle \frac{vr_i(1-\alpha_i)}{s + vr_i + q_i} \right\rangle,$$

and substitute $D(s,v)$ with $I - F(s,v)$ in (3). The inverse Laplace transform of $F(s,v)$ with respect to the time variable is

$$F(t,v) = \mathrm{diag}\left\langle vr_i(1-\alpha_i)e^{-(vr_i+q_i)t}\right\rangle \ .$$

Using these matrices we can perform a symbolic inverse Laplace transformation of (3) which results in:

$$P^{\sim}(t,v) = e^{(-vR_\alpha+Q)t} - \int_{\tau=0}^{t} e^{(-vR_\alpha+Q)\tau} F(t-\tau,v)d\tau \tag{4}$$

The moments of the accumulated reward is obtained from (4) as

$$E(B^n(t)) = \underline{\pi}\ (-1)^n\ \frac{d^n}{dv^n}P^{\sim}(t,v)\bigg|_{v=0} \underline{h}^T \ ,$$

where $\underline{\pi}$ is the initial probability vector and $\underline{h}$ is the vector of ones. The nth derivative of $P^{\sim}(t,v)$ at $v=0$ can be calculated as

$$\begin{aligned} \frac{d^n}{dv^n}P^{\sim}(t,v)\bigg|_{v=0} &= \frac{d^n}{dv^n}\ e^{(-vR_\alpha+Q)t}\bigg|_{v=0} \\ &- \int_{\tau=0}^{t}\sum_{\ell=0}^{n}\binom{n}{\ell}\frac{d^\ell}{dv^\ell}\ e^{(-vR_\alpha+Q)\tau}\bigg|_{v=0}\ \frac{d^{n-\ell}}{dv^{n-\ell}}F(t-\tau,v)\bigg|_{v=0} d\tau \end{aligned} \tag{5}$$

where the 0th derivative is the function itself. Since $F(\tau,v)$ is a diagonal matrix the ℓth derivative of $F(\tau,v)$ at $v=0$ can be calculated in a computationally cheap way as

$$\frac{d^\ell}{dv^\ell}F(\tau,v)\bigg|_{v=0} = \mathrm{diag}\left\langle r_i(1-\alpha_i)\ \ell(-r_i\tau)^{\ell-1}\ e^{-q_i\tau}\right\rangle \ .$$

Two computationally expensive steps have to be performed to evaluate the nth derivative of $P^{\sim}(t,v)$ at $v=0$ based on (5). The first one is the calculation of the first n derivatives of $e^{(-vR_\alpha+Q)\tau}$ at $v=0$ and at some time points $\tau\in(0,t]$, and the second one is the numerical integration with respect to τ. The numerical integration is not expensive itself, but it requires the calculation of the first step several times. The numerical method presented in [16] is an effective way of calculating the first n derivatives of $e^{(-vR_\alpha+Q)\tau}$ at $v=0$, hence we use it for the calculation of the first step.

The complexity of the proposed numerical procedure is much higher than the analysis of the same Markov reward model without reward loss for two reasons. The first one is the mentioned numerical integration, and the second one is related to the complexity of the elementary steps of the computation of $d^n/dv^n\ e^{(-vR_\alpha+Q)t}$. Basically, the first term in (5) provides the moments of the Markov reward model of the same CTMC with reduced reward rates $(\alpha_i r_i)$ and without reward loss. For the calculation of the moments it is enough to calculate only the row sum of the first term, $e^{(-vR_\alpha+Q)t}$, since it is multiplied by $\underline{h}^T$ from the right. It is much faster to calculate the row sum of $e^{(-vR_\alpha+Q)t}$ instead of the calculation of the whole matrix, because the row sum can be obtained by vector-matrix multiplications, while the calculation of the whole matrix requires matrix-matrix multiplications in each elementary step of the computation [16]. Unfortunately, the second term in (5) requires the calculation of the whole matrix (using matrix-matrix multiplications), because of the multiplication by the diagonal matrix $F(t-\tau,v)$ from the right. This is why we defined and calculated $P(t,w)$ as a matrix all along the above derivations.

Finally, we note that the product of two double transform functions in (3) results in double convolutions in the original (t, w) domain. In our approach one convolution is avoided due to the calculation of the moments of the accumulated reward. Since the calculation of the distribution of a prs Markov reward model is very expensive itself (it is much more expensive than to calculate its moments), a direct method to calculate the distribution of the accumulated reward by double numerical convolution becomes infeasible even for small models ($\sim$10 states). Instead, the numerical method for the analysis of the moments of the accumulated reward is applicable for models of $\sim$100 states.

4 Stationary analysis of accumulated reward

The previous sections provide a numerical method to calculate the moments of the accumulated reward of partial incremental loss models. Using that method the evaluation of partial loss reward models is computationally much more expensive than the calculation of the prs reward models of the same size.

In this section we provide an effective computational approach that makes possible to evaluate much larger partial incremental loss models ($\sim 10^6$ states). This numerical approach allows the analysis of a special class of partial loss models where the background process is in stationary state. Note that the reward accumulation of partial incremental loss models with stationary background process has non-stationary increment on the $(0, t)$ interval (e.g., $E(B(t)) \neq 2E(B(t/2))$), because the reward accumulated in the last state may have different effects on the overall accumulated reward.

The main idea of the proposed method is to define an equivalent prs reward model, whose accumulated reward equals the reward accumulated by the original partial loss reward model, and to evaluate the accumulated reward of the equivalent model.

The reward accumulation process of a partial loss reward model can be divided into two main parts as it is mentioned above. During the $(0, t^*)$ interval the system accumulates reward at reduced reward rates $(\alpha_i r_i)$ (without reward loss), and during the (t^*, t) interval it accumulates at the original reward rate (r_i). If t^* (and $Z(t^*)$) was known it would be straightforward to calculate the accumulated reward, but t^* depends in a complex way on the CTMC behavior over the whole $(0, t)$ interval. t^* is not a stopping time.

To overcome this difficulty one can interpret the reward accumulation from time t towards time 0. In this case t^* is simply the time instant of the first state transition in the reverse CTMC, and the reverse reward model is such that it accumulates reward at the original rate (r_i) in its first state and it accumulates reward at the reduced rate $(\alpha_i r_i)$ after leaving the first state. To apply this approach we need the generator of the reverse CTMC.

The probability that the process is in state i at time t and in state j $(j \neq i)$ at $t + \Delta$, i.e., $Pr(Z(t) = i, Z(t + \Delta) = j)$, can be calculated as:

$$Pr(Z(t) = i)\ Pr(Z(t+\Delta) = j \mid Z(t) = i) = Pr(Z(t+\Delta) = j)\ Pr(Z(t) = i \mid Z(t+\Delta) = j).$$

Dividing both sides by Δ and letting $\Delta \to 0$ we have

$$Pr(Z(t) = i)\ q_{ij} = Pr(Z(t) = j)\ \overleftarrow{q_{ji}}(t)\ ,$$

where $\overleftarrow{q_{ji}}(t)$ is the generator of the reverse CTMC. One can see that the generator of the reverse CTMC depends on the transient probabilities of the original CTMC, hence it is time inhomogeneous, in general. In the stationary case the state probabilities are constant and the generator of the reverse CTMC becomes time homogeneous:

$$\overleftarrow{q_{ji}} = \frac{\gamma_i}{\gamma_j} q_{ij}, \tag{6}$$

where γ_i is the stationary probability of state i in the original (as well as the reverse) CTMC. The stationary probabilities can be obtained solving $\forall j \in S\ \sum_{i \in S} \gamma_i q_{ij} = 0$ with the normalizing condition $\sum_{i \in S} \gamma_i = 1$. The diagonal elements of the generator of the stationary reverse CTMC are the same as the original diagonal elements (since the reverse process spends the same time in each state as the original one). It is easy to check that matrix $\overleftarrow{Q} = \{\overleftarrow{q_{ij}}\}$ defined by (6) is a proper generator matrix.

In case the original partial loss model starts from the stationary state, we can define an equivalent prs Markov reward model that accumulates the same amount of reward during the $(0, t)$ interval as our original partial loss model using the reverse interpretation of the reward accumulation. The original partial loss model is defined by $(\underline{\gamma}, Q, R, R_\alpha)$ (the initial probability vector, which is the stationary distribution of the CTMC, the generator matrix, the diagonal matrix of the reward rates, the diagonal matrix of the reduced reward rates). Based on this description we define an equivalent prs Markov reward model with state space of $2|S|$ states by initial probability vector π', generator matrix Q', and reward rate matrix R' as follows:

$$\underline{\pi'} = \{\underline{\gamma}, 0\}, \quad Q' = \begin{array}{|c|c|} \hline Q_D & \overleftarrow{Q} - Q_D \\ \hline 0 & \overleftarrow{Q} \\ \hline \end{array}\,, \quad R' = \begin{array}{|c|c|} \hline R & 0 \\ \hline 0 & R_\alpha \\ \hline \end{array}\,, \tag{7}$$

$Q_D = \mathrm{diag}\langle q_{ii} \rangle$ is the diagonal matrix composed of the diagonal elements of Q. Each state of the original CTMC is represented by two states in the equivalent prs Markov reward model. States 1 to $|S|$ represent the reward accumulation with the original reward rate (r_i). The equivalent model starts from this set of states according to the stationary distribution $\underline{\gamma}$. States $|S|+1$ to $2|S|$ represent the reward accumulation after the first state transition with the reduced reward rates. The structure of the Q' matrix is such that the equivalent process moves from the first set of states (states 1 to $|S|$) to the second one (states $|S| + 1$ to $2|S|$) at the first state transition and remains there. The distribution of the reward accumulated during the $(0, t)$ interval by a prs Markov reward model with initial probability vector π', generator matrix Q', and reward rate matrix R' is (see e.g., [16])

$$\underline{\pi'}(sI' + vR' - Q')^{-1}\underline{h'}^T \tag{8}$$

where the cardinality of the identity matrix I' and summing vector $\underline{h'}$ is $2|S|$.

The formal relation of the original partial loss model and the reverse prs Markov reward model is presented in the following theorem.

Theorem 2 *The distribution of reward accumulated by the prs Markov reward model (π', Q', R') is identical with the distribution of reward accumulated by the partial incremental loss Markov reward model $(\underline{\gamma}, Q, R, R_\alpha)$, that is (from eq. (3) and (8)):*

$$\underline{\gamma}(sI + vR_\alpha - Q)^{-1}D(s, v)\underline{h}^T = \underline{\pi'}(sI' + vR' - Q')^{-1}\underline{h'}^T \tag{9}$$

The proof of the Theorem is provided in Appendix B.

The equivalent reward model is a prs Markov reward model. Its analysis can be performed with effective numerical methods available in the literature. E.g., the distribution of the accumulated reward can be calculated using [12, 5, 6] and its moments using [16].

It is easy to evaluate the limiting behavior of a partial loss model with stationary background CTMC. We use the following notation. $B(t)$ is the reward accumulated by a stationary partial incremental loss model defined by (Q, R, R_α). $B'(t)$ and $B"(t)$ are the rewards accumulated by stationary prs reward models defined by (Q, R) and (Q, R_α), respectively. The stationary distribution of the CTMC with generator Q is γ. For short intervals the loss at the first transition does not play role, hence

$$\lim_{t\to 0} B(t)/t \equiv \lim_{t\to 0} B'(t)/t,$$

and for very long intervals the reward accumulated from the last state transition to the end of the interval is negligible with respect to the total accumulated reward

$$\lim_{t\to \infty} B(t)/t \equiv \lim_{t\to \infty} B"(t)/t$$

E.g., the limiting behavior of the mean accumulated reward can be calculated as

$$\begin{aligned} \lim_{t\to 0} \frac{E(B(t))}{t} &= \lim_{t\to 0} \frac{E(B'(t))}{t} = \sum_{i\in S} \gamma_i r_i \ , \\ \lim_{t\to \infty} \frac{E(B(t))}{t} &= \lim_{t\to \infty} \frac{E(B"(t))}{t} = \sum_{i\in S} \gamma_i \alpha_i r_i \ . \end{aligned} \tag{10}$$

5 Performance analysis of computer systems with checkpointing

Checkpointing is a widely applied technique to improve the performance of computing servers executing long running batch programs in the presence of failures [4, 3, 7, 10]. Long running batch programs need to be re-executed in case of a system failure before the completion of the program. To reduce the extra re-execution work of the system the actual state of the program is saved occasionally during the operational time of the system. This saved program state is used when a failure occurs. After a failure and the subsequent repair the saved program state is reloaded and the program is re-executed from its saved state. The operation of saving the current state of the program is referred to as checkpointing and the reload of the saved program state is called rollback.

It is a common feature of all checkpointing models that a portion of work executed since the last system failure is lost at the next system failure, hence the amount of executed work can be analyzed using partial loss models. To find the relation between the applied checkpointing policy and the parameters of the partial loss reward model is out of the scope of this paper. Here, we follow a system level approach, which means that the parameters of the partial loss model of the analyzed computing server are assumed to be known. However, some considerations on the behavior of the analyzed system are provided below.

It is important to note that our analysis approach contains a simplifying assumption. The portion of work lost at a system failure is a random quantity. The analysis of partial loss reward models with random loss ratio is studied in [1], but unfortunately, there is no effective numerical method available for their analysis. This is the reason for using (state dependent) deterministic loss ratio in our model.

The overall performance analysis of computing systems with checkpointing is composed of two major steps:

I. Generation of partial loss Markov reward model based on the system behavior:

- characterize the state space of the model based on the system load and the failure process.
- evaluate the failure rate and computing power assigned to the jobs under execution in each system state r_i.
- calculate the (optimal) checkpointing rate in each system state.
- calculate the state dependent loss ratio (the portion of work that needs to be re-executed), based on the failure rate and the checkpointing rate.

II. Solution of the obtained partial loss Markov reward model.

In the following numerical example we utilize the result of step I. and perform step II.

Consider a computing server executing long running batch programs. Jobs of two classes arrive to the server. Class 1 (class 2) jobs arrive according to a Poisson process with rate λ_1 (λ_2). Each of these jobs requires an exponentially distributed execution time with parameter μ_1 (μ_2) with the full computing capacity of the server. The server has finite capacity (N_{MAX}) and the number of class 1 (class 2) jobs cannot exceed N_1 (N_2), i.e., $n_1 \leq N_1$, $n_2 \leq N_2$, $n_1+n_2 \leq N_{MAX}$, where n_1 (n_2) is the number of class 1 (class 2) jobs in the system. The failure rate is load dependent: $\nu(n_1, n_2) = \omega_a + \omega_b(n_1 + n_2)$, where ω_a and ω_b are the parameters of the load independent and load dependent parts of the failure rate, respectively. The repair time, including the rollback time, is exponentially distributed as well. We use state independent repair rate β. (Note that the applied modeling approach can handle state dependent repair rates with the same computational complexity.) Job arrival is also allowed during repair. The computing performance of the server slightly decreases with the number of jobs under execution (e.g., due to the swapping of jobs). r_a ($0 \leq r_a \leq 1$, $r_a \sim 1$) is the portion of the computing power that is utilized for job execution when there is only one job in the server. Suppose the presence of class 1 jobs increases the checkpointing rate, the portion of useful work maintained at a system failure increases with the number of class 1 jobs. α_a and α_b are used to represent the load independent and load dependent part of the useful work ratio, respectively.

Having these Markovian assumptions one can easily model a wide range of service discipline schemes. We consider weighted processor sharing with state dependent weights. Our service discipline assigns a predefined portion of the computing power, ϕ_1 ($0 < \phi_1 < 1$) and $\phi_2 = 1 - \phi_1$, to jobs of class 1 and class 2, respectively. Jobs of the same class are executed at the same speed. If there are only jobs of one class in the system, the whole computing capacity will be utilized by that class. As a special case of this service discipline we obtain the preemptive priority service discipline when ϕ_2 tends to 0. In this case class 1 jobs are executed with the whole computing power of the server as long as there are class 1 jobs in the system.

Based on this system behavior the performance of the considered computing system is analyzed using the partial loss Markov reward model defined in Table 1. The state space of the CTMC is characterized by the number of class 1 and class 2 jobs in the system and the operational condition of the system. The operational condition can be one of the following three: *Good*, *To_fail* and *Repair*. We need to distinguish between the operational states that are followed by another operational state (*Good*) and the operational states that are

State space description	
n_1 : 0 **To** N_1	#class 1 jobs
n_2 : 0 **To** N_2	#class 2 jobs
$\{Good,\ To\_fail,\ Repair\}$	operational condition
$n_1 + n_2 \leq N_{MAX}$	
Underlying CTMC	
$(n_1, n_2, Good) \rightarrow (n_1 + 1, n_2, Good) = p\ \lambda_1$	class 1 job arrival
$(n_1, n_2, Good) \rightarrow (n_1 + 1, n_2, To\_fail) = q\ \lambda_1$	
$(n_1, n_2, Repair) \rightarrow (n_1 + 1, n_2, Repair) = \lambda_1$	
$(n_1, n_2, Good) \rightarrow (n_1, n_2 + 1, Good) = p\ \lambda_2$	class 2 job arrival
$(n_1, n_2, Good) \rightarrow (n_1, n_2 + 1, To\_fail) = q\ \lambda_2$	
$(n_1, n_2, Repair) \rightarrow (n_1, n_2 + 1, Repair) = \lambda_2$	
$(n_1, n_2, Good) \rightarrow (n_1 - 1, n_2, Good) = p\ \dfrac{\phi_1 n_1}{\phi_1 n_1 + \phi_2 n_2} \mu_1$	class 1 job departure
$(n_1, n_2, Good) \rightarrow (n_1 - 1, n_2, To\_fail) = q\ \dfrac{\phi_1 n_1}{\phi_1 n_1 + \phi_2 n_2} \mu_1$	
$(n_1, n_2, Good) \rightarrow (n_1, n_2 - 1, Good) = p\ \dfrac{\phi_2 n_2}{\phi_1 n_1 + \phi_2 n_2} \mu_2$	class 2 job departure
$(n_1, n_2, Good) \rightarrow (n_1, n_2 - 1, To\_fail) = q\ \dfrac{\phi_2 n_2}{\phi_1 n_1 + \phi_2 n_2} \mu_2$	
$(n_1, n_2, To\_fail) \rightarrow (n_1, n_2, Repair) = \omega_a + \omega_b(n_1 + n_2)$	failure
$(n_1, n_2, Repair) \rightarrow (n_1, n_2, Good) = p\ \beta$	repair
$(n_1, n_2, Repair) \rightarrow (n_1, n_2, To\_fail) = q\ \beta$	
Reward and loss structure	
$r(n_1, n_2, Good) = r_a^{n_1+n_2}$ if: $n_1 + n_2 > 0$	reward rate
$r(0, 0, Good) = 0$	
$r(n_1, n_2, To\_fail) = r_a^{n_1+n_2}$ if: $n_1 + n_2 > 0$	
$r(0, 0, To\_fail) = 0$	
$r(n_1, n_2, Repair) = 0$	
$\alpha(n_1, n_2, Good) = 1$	useful work ratio
$\alpha(n_1, n_2, To\_fail) = \alpha_a + \alpha_b \dfrac{n_1}{n_1 + n_2}$ if: $n_1 + n_2 > 0$	
$\alpha(0, 0, To\_fail) = 0$	
$\alpha(n_1, n_2, Repair) = 0$	

Table 1: The partial loss Markov reward model of the computing system

followed by a failure (*To_fail*), because there is no work loss at the departure from a *Good* state while there is some work loss at the departure from a *To_fail* state. The probability of moving to the *Good* and *To_fail* condition (i.e., p and q, respectively) are calculated based on the number of jobs in the *destination* state. For $0 < n_1+n_2 < N_{MAX}$ & $n_1 < N_1$ & $n_2 < N_2$:

$$q = 1 - p = \frac{\omega_a + \omega_b(n_1+n_2)}{\lambda_1 + \lambda_2 + \frac{\phi_1 n_1 \mu_1}{\phi_1 n_1 + \phi_2 n_2} + \frac{\phi_2 n_2 \mu_2}{\phi_1 n_1 + \phi_2 n_2} + \omega_a + \omega_b(n_1+n_2)} ,$$

for $n_1 = n_2 = 0$:

$$q = 1 - p = \frac{\omega_a}{\lambda_1 + \lambda_2 + \omega_a} ,$$

for $n_1 + n_2 = N_{MAX}$ or $n_1 < N_1$ & $n_2 < N_2$:

$$q = 1 - p = \frac{\omega_a + \omega_b(n_1+n_2)}{\frac{\phi_1 n_1 \mu_1}{\phi_1 n_1 + \phi_2 n_2} + \frac{\phi_2 n_2 \mu_2}{\phi_1 n_1 + \phi_2 n_2} + \omega_a + \omega_b(n_1+n_2)} ,$$

for $n_1 + n_2 < N_{MAX}$ and $n_1 = N_1$ & $n_2 < N_2$:

$$q = 1 - p = \frac{\omega_a + \omega_b(n_1+n_2)}{\lambda_2 + \frac{\phi_1 n_1 \mu_1}{\phi_1 n_1 + \phi_2 n_2} + \frac{\phi_2 n_2 \mu_2}{\phi_1 n_1 + \phi_2 n_2} + \omega_a + \omega_b(n_1+n_2)} ,$$

and for $n_1 + n_2 < N_{MAX}$ and $n_1 < N_1$ & $n_2 = N_2$:

$$q = 1 - p = \frac{\omega_a + \omega_b(n_1+n_2)}{\lambda_1 + \frac{\phi_1 n_1 \mu_1}{\phi_1 n_1 + \phi_2 n_2} + \frac{\phi_2 n_2 \mu_2}{\phi_1 n_1 + \phi_2 n_2} + \omega_a + \omega_b(n_1+n_2)} .$$

The following set of system parameters were used for the numerical evaluation:

- state space: $N_1 = 3, N_2 = 4, N_{MAX} = 6$;
- job arrival and computing requirement [1/hours]: $\lambda_1 = 0.4, \lambda_2 = 0.4, \mu_1 = 2, \mu_2 = 1$;
- resource sharing between class 1 and class 2 jobs: $\phi_1 = 2/3, \phi_2 = 1/3$;
- failure and repair parameters [1/hours]: $\omega_a = 0.3, \omega_b = 0.03, \beta = 2$;
- overhead parameter: $r_a = 0.98$;
- work loss parameters: $\alpha_a = 0.6, \alpha_b = 0.05$.

The system performance was evaluated with two initial probability distributions (Figure 3). In the first case the system starts from stationary state, and in the second case the system starts from state $(0, 0, Good)$ with probability 1. The case when the system starts from state $(0, 0, Good)$ was evaluated by the method presented in section 3 and the case of stationary background CTMC was evaluated with both methods (section 3 and 4). The accuracy of the prs reward analysis method, which is applied in both cases, was 10^{-6}. The numerical integration of the first method was computed over 100 equidistant points. The numerical results obtained for the stationary case were practically identical, hence there are only two curves depicted in Figure 3.

Based on the stationary analysis of the prs Markov reward model with reduced reward rates, (Q, R_α), (10) we have $\lim_{t\to\infty} E(B(t))/t = 0.4718$, and $\lim_{t\to\infty} Var(B(t))/t = 0.0548$. Each pair of mean and variance curves in Figure 3 tends to the respective limit. The mean curve associated with the stationary background process starts from the stationary accumulation rate of the prs Markov reward model with original reward rates, (Q, R), (10).

The detailed analysis of a slightly larger partial loss Markov reward model of the same example with stationary initial distribution and with $N_1 = 10, N_2 = 20, N_{MAX} = \infty, \lambda_1 = 0.5, \lambda_2 = 0.5$ results in the curves in Figure 4. It can be seen that the transition from the initial to the final $E(B(t))/t$ value takes place between 0.1 and 10 hours, and the $Var(B(t))/t$ curve has a peak in this range. That is the range where the effect of the reward loss at the first state transition turns up. The peak of the $Var(B(t))/t$ curve is sharper for the small system.

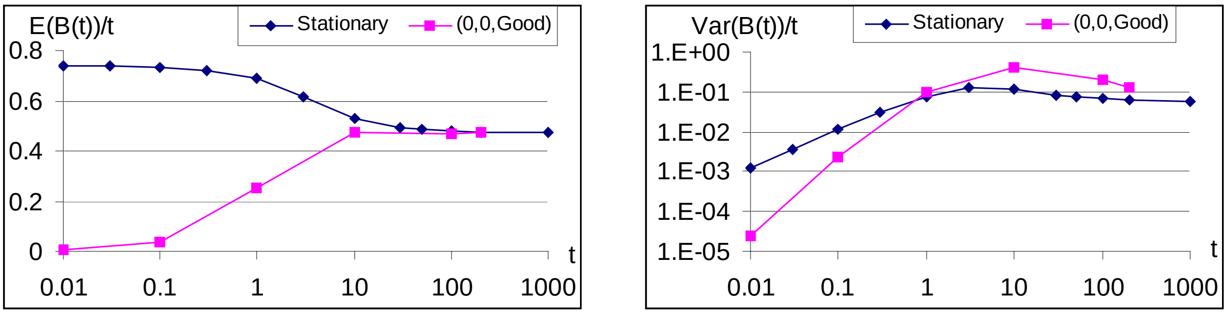


Figure 3: Moments of computing system performance (57 state model)

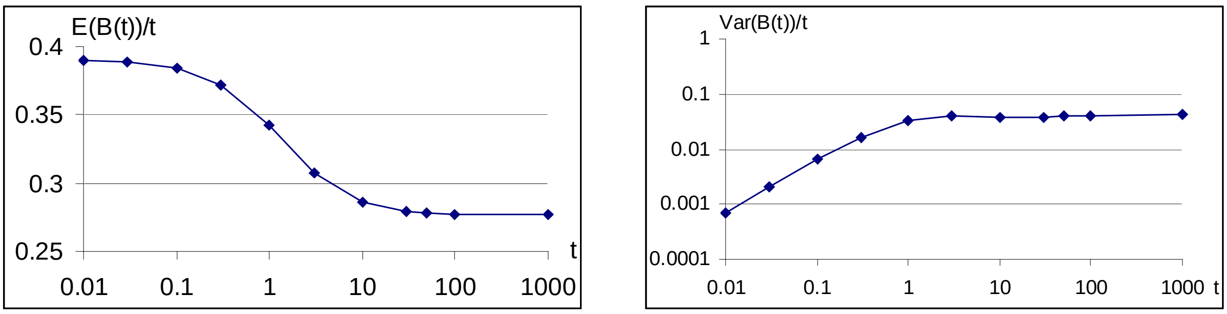


Figure 4: Moments of computing system performance (1386 state model)

6 Conclusion

The paper presents two new numerical analysis methods for the analysis of partial incremental loss Markov reward models. The first one is applicable with any general initial probability distribution, but it is computationally more intensive. It can be applied for models with ∼100 states. The second one is applicable only for partial loss models with stationary background CTMC, but it is computationally more effective. It can be applied for models with $\sim 10^6$ states. To demonstrate the applicability of partial loss reward models and the numerical properties of the proposed analysis methods a computing system executing long running batch programs is analyzed. Numerical results show that the proposed method is stable with the considered range of model parameters. We used only a simple implementation of the method presented in section 3. With a sophisticated implementation of the same mode (e.g., with intelligent numerical integration) one can further enhance the applicability of the method.

Acknowledgement

The authors thank the careful review of the referees. They helped to improve the presentation of the paper and to remove an annoying numerical error from the example.

References

[1] A. Bobbio, V. G. Kulkarni, and M. Telek. Partial loss in reward models. In *2nd Int. Conf. on Mathematical Methods in Reliability*, pages 207–210, Bordeaux, France, July 2000.

[2] A. Bobbio and M. Telek. Task completion time. In *Proceedings 2nd International Workshop on Performability Modelling of Computer and Communication Systems (PMCCS2)*, 1993.

[3] A. Brock. An analysis of checkpointing. *ICL Tech. Journal*, 1(3), 1979.

[4] K.M. Chandy, J.C. Browne, C.W. Dissly, and W.R. Uhrig. Analytic models for rollback and recovery strategies in database systems. *IEEE Trans. on Software Engineering*, SE-1(1):100–110, 1975.

[5] E. de Souza e Silva, H.R. Gail, and R. Vallejos Campos. Calculating transient distributions of cumulative reward. In *Proceedings ACM/SIGMETRICS Conference*, Ottawa, 1995.

[6] L. Donatiello and V. Grassi. On evaluating the cumulative performance distribution of fault-tolerant computer systems. *IEEE Trans. on Computers*, 1991.

[7] E. Gelenbe and D. Derochette. Performance of rollback recovery systems under intermittent failures. *Commun. ACM*, 21(6):493–499, 1978.

[8] V.G. Kulkarni, V.F. Nicola, and K. Trivedi. On modeling the performance and reliability of multi-mode computer systems. *The Journal of Systems and Software*, 6:175–183, 1986.

[9] V.G. Kulkarni, V.F. Nicola, and K. Trivedi. The completion time of a job on a multimode system. *Advances in Applied Probability*, 19:932–954, 1987.

[10] V.G. Kulkarni, V.F. Nicola, and K. Trivedi. Effects of checkpointing and queueing on program performance. *Stochastic models*, 4(6):615–648, 1990.

[11] J.F. Meyer. On evaluating the performability of degradable systems. *IEEE Trans. on Computers*, C-29:720–731, 1980.

[12] H. Nabli and B. Sericola. Performability analysis: a new algorithm. *IEEE Trans. on Computers*, 45:491–494, 1996.

[13] V.F. Nicola, R. Martini, and P.F. Chimento. The completion time of a job in a failure environment and partial loss of work. In *2nd Int. Conf. on Mathematical Methods in Reliability (MMR'2000)*, pages 813–816, Bordeaux, France, July 2000.

[14] S. Rácz and M. Telek. Performability analysis of Markov reward models with rate and impulse reward. In *Int. Conf. on Numerical Solution of Markov Chains*, pages 169–187, Zaragoza, Spain, 1999.

[15] M. Telek and A. Pfening. Performance analysis of Markov Regenerative Reward Models. *Performance Evaluation*, 27&28:1–18, 1996.

[16] M. Telek and S. Rácz. Numerical analysis of large Markovian reward models. *Performance Evaluation*, 36&37:95–114, Aug 1999.

A Proof of Theorem 1

Conditioning on H, the sojourn time in the initial state i, we have:

$$P_{ij}(t, w|\, H=\tau) \;=\; \begin{cases} \delta_{ij}\; U_w(w - r_i t) & \text{if} : \tau > t \\ \displaystyle\sum_{k\in S, k\neq i} \frac{q_{ik}}{q_i}\, P_{kj}(t-\tau, w - \alpha_i \tau r_i) & \text{if} : \tau < t \end{cases} \tag{11}$$

where $q_i = -q_{ii}$, $U(\cdot)$ is the unit step function and δ_{ij} is the Kronecker delta (if $i = j$ then $\delta_{ij} = 1$, otherwise $\delta_{ij} = 0$). Taking the Laplace-Stieltjes transform with respect to w ($\rightarrow v$), $Re(v) \geq 0$:

$$P^{\sim}_{ij}(t, v|\, H=\tau) \;=\; \begin{cases} \delta_{ij}\; e^{-v r_i t} & \text{if} : \tau > t \\ \displaystyle\sum_{k\in S, k\neq i} \frac{q_{ik}}{q_i}\, e^{-v r_i \alpha_i \tau}\, P^{\sim}_{kj}(t-\tau, v) & \text{if} : \tau < t \end{cases} \tag{12}$$

Unconditioning with respect to H, based on the sojourn time distribution in state i, $(1 - e^{-q_i t})$, results in:

$$P^{\sim}_{ij}(t, v) \;=\; \delta_{ij}\; e^{-v r_i t}\; e^{-q_i t} + \sum_{k\in S, k\neq i} \int_{\tau=0}^{t} q_{ik}\; e^{-v r_i \alpha_i \tau}\; e^{-q_i \tau}\, P^{\sim}_{kj}(t-\tau, v)\; d\tau \tag{13}$$

Taking the Laplace transform with respect to t ($\rightarrow s$), $Re(s) \geq 0$, results in:

$$\begin{aligned}
&P^{*\sim}_{ij}(s, v) \;=\; \int_{t=0}^{\infty} e^{-st} P^{\sim}_{ij}(t, v) dt \;= \\
&\int_{t=0}^{\infty} e^{-st}\; \delta_{ij}\; e^{-(v r_i + q_i)t}\; dt + \sum_{k\in S, k\neq i} \int_{t=0}^{\infty} e^{-st} \int_{\tau=0}^{t} q_{ik}\; e^{-(v r_i \alpha_i + q_i)\tau}\, P^{\sim}_{kj}(t-\tau, v)\; d\tau\; dt \;= \\
&\delta_{ij}\; \frac{1}{s + v r_i + q_i} + \sum_{k\in S, k\neq i} \int_{\tau=0}^{\infty} e^{-s\tau}\; q_{ik}\; e^{-(v r_i \alpha_i + q_i)\tau} \int_{t=\tau}^{\infty} e^{-s(t-\tau)}\, P^{\sim}_{kj}(t-\tau, v)\; dt\; d\tau \;= \\
&\delta_{ij}\; \frac{1}{s + v r_i + q_i} + \sum_{k\in S, k\neq i} \int_{\tau=0}^{\infty} q_{ik}\; e^{-(s + v r_i \alpha_i + q_i)\tau}\; d\tau\; P^{*\sim}_{kj}(s, v) \;= \\
&\delta_{ij}\; \frac{1}{s + v r_i + q_i} + \sum_{k\in S, k\neq i} \frac{q_{ik}}{s + v r_i \alpha_i + q_i}\; P^{*\sim}_{kj}(s, v)
\end{aligned} \tag{14}$$

Equation (14), in matrix form, results in the theorem.

B Proof of Theorem 2

The left hand side of eq. (9) can be rewritten as

$$\underline{\gamma}(sI+vR_\alpha-Q)^{-1}D(s,v)\underline{h^T} = \underline{\gamma}(sI+vR_\alpha-Q)^{-1}(sI+vR_\alpha-Q_D)\,(sI+vR-Q_D)^{-1}\underline{h^T} \quad (15)$$

For the evaluation of the right hand side of eq. (9), we use the partitioned form of matrices I', R', Q'. That is

$$(sI'+vR'-Q') = \begin{array}{|c|c|}\hline sI+vR-Q_D & -\overleftarrow{Q}+Q_D \\ \hline 0 & sI+vR_\alpha-\overleftarrow{Q} \\ \hline \end{array}\,, \quad (16)$$

and

$$(sI'+vR'-Q')^{-1} = $$

$$\begin{array}{|c|c|}\hline (sI+vR-Q_D)^{-1} & (sI+vR-Q_D)^{-1}(\overleftarrow{Q}-Q_D)(sI+vR_\alpha-\overleftarrow{Q})^{-1} \\ \hline 0 & (sI+vR_\alpha-\overleftarrow{Q})^{-1} \\ \hline \end{array}\,. \quad (17)$$

Using the special structure of the initial vector $\underline{\pi'}$ we have:

$$\begin{aligned}
&\underline{\pi'}(sI'+vR'-Q')^{-1}\underline{h'^T} = \\
&\underline{\gamma}(sI+vR-Q_D)^{-1}\left[I+(\overleftarrow{Q}-Q_D)(sI+vR_\alpha-\overleftarrow{Q})^{-1}\right]\underline{h^T} = \\
&\underline{\gamma}(sI+vR-Q_D)^{-1} \\
&\quad\left[(sI+vR_\alpha-\overleftarrow{Q})(sI+vR_\alpha-\overleftarrow{Q})^{-1}+(\overleftarrow{Q}-Q_D)(sI+vR_\alpha-\overleftarrow{Q})^{-1}\right]\underline{h^T} = \\
&\underline{\gamma}(sI+vR-Q_D)^{-1}(sI+vR_\alpha-Q_D)\,(sI+vR_\alpha-\overleftarrow{Q})^{-1}\underline{h^T} =
\end{aligned} \quad (18)$$

Let Γ be the diagonal matrix of the stationary probabilities, i.e., $\Gamma = \mathrm{diag}\langle\gamma_i\rangle$. Using this diagonal matrix $\underline{\gamma} = \underline{h}\,\Gamma$ and from eq. (6) $\overleftarrow{Q} = \Gamma^{-1}Q^T\Gamma$. In the following steps the diagonal matrices Γ, R_α, $(sI+vR-Q_D)$ and $(sI+vR_\alpha-Q_D)$ are commuted if necessary:

$$\underline{h}\;\Gamma\;(sI+vR-Q_D)^{-1}(sI+vR_\alpha-Q_D)\;(sI+vR_\alpha-\Gamma^{-1}Q^T\Gamma)^{-1}\underline{h^T} =$$

$$\underline{h}\left[\Big((sI+vR_\alpha-\Gamma^{-1}Q^T\Gamma)^{-1}\Big)^T\Big(\Gamma\;(sI+vR-Q_D)^{-1}(sI+vR_\alpha-Q_D)\Big)^T\right]^T\underline{h^T} = \dots$$

The external transpose vanishes due to the multiplication by $\underline{h}$ from left and $\underline{h^T}$ from right and the second internal transpose also vanishes because it contains a diagonal matrix. In the first internal transpose we interchange the order of transpose and inversion:

$$\begin{aligned}
&\underline{h}\Big((sI+vR_\alpha-\Gamma^{-1}Q^T\Gamma)^T\Big)^{-1}\;\Gamma\;(sI+vR-Q_D)^{-1}(sI+vR_\alpha-Q_D)\;\underline{h^T} = \\
&\underline{h}\Big(sI+vR_\alpha-\Gamma\;Q\Gamma^{-1}\Big)^{-1}\Big(\Gamma^{-1}\Big)^{-1}\;(sI+vR-Q_D)^{-1}(sI+vR_\alpha-Q_D)\;\underline{h^T} = \\
&\underline{h}\Big(s\Gamma^{-1}+v\Gamma^{-1}R_\alpha-Q\Gamma^{-1}\Big)^{-1}\;(sI+vR-Q_D)^{-1}(sI+vR_\alpha-Q_D)\;\underline{h^T} = \\
&\underline{h}\;\Gamma\;(sI+vR_\alpha-Q)^{-1}(sI+vR_\alpha-Q_D)\;(sI+vR-Q_D)^{-1}\;\underline{h^T}
\end{aligned} \quad (19)$$

The theorem is given by the equivalence of (15) and (19).