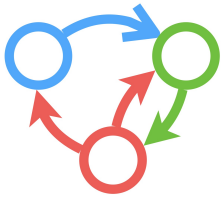
BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

Excel@FIT 2017
http://excel.fit.vutbr.cz

On Improving Adaptive Error-Driven Aggregation of Markov Chains

Roman Andriushchenko*



Abstract
Markov models are widely used in many areas of science and engineering in order to evaluate the probability of certain events of interest. Quantitative analysis of such models typically proceeds through numerical methods or simulation-based evaluation. Since the state space of the models can often be large, several approximation techniques have been proposed. For various systems, level of precision affects the soundness of verification results, so accurate quantification of approximation error is crucial. In this work we focus on adaptively-driven aggregation technique and evaluate its key performance aspects. The key contribution of this work is improving aggregation strategy and the theoretical bounds on the approximation error. Our technique leads to up to 3 orders of magnitude precision improvement over existing methods and allows one to analyse larger models with a higher accuracy.

Keywords: Markov models — probabilistic model checking — approximation techniques — adaptive aggregation

Supplementary Material: *N/A*

*xandri03@stud.fit.vutbr.cz, *Faculty of Information Technology, Brno University of Technology*

1. Introduction

Markov models have been widely used to analyse reliability and performance of computer networks, communication and security protocols [1, 2], and to study various quantitative attributes of biochemical reaction networks [3] as well as molecular devices [4]. Quantitative analysis of time-bounded Markov chains typically proceeds through numerical analysis, via solution of equations yielding the probability of the system residing in a given state at a given time, or simulation-based exploration of its execution paths. There are many situations where highly accurate probability estimates are necessary, for example for reliability analysis in safety-critical systems or for predictive modelling in scientific experiments, but this is difficult to achieve in practice due to the state-space explosion problem [5].

In order to enable the handling of larger state spaces, two types of techniques have been introduced: state aggregation and state-space truncation. State aggregation techniques build a reduced state space using e.g. bisimulation quotient [6]. State-space truncation methods, e.g. fast adaptive uniformisation (FAU) [7, 8], on the other hand, only consider the states whose probability mass is not negligible, while computing the total probability loss.

Imprecise values are known to cause robustness failure, in the sense that the satisfaction of temporal logic formulae is affected by small changes to the probability bound. In [5] there has been proposed a novel adaptive aggregation method for Markov chains

that works over a finite time interval by clustering the state space of a model sequentially in time, with the quality of the current aggregation being quantified and used to derive explicit error bounds. This technique was implemented and evaluated on two case studies of chemical reaction networks and lead to a marked improvement in numerical precision.

1.1 Key contributions

In this work we focus on adaptively-driven aggregation technique proposed in [5] and evaluate its key performance aspects. Based on this analysis, we improve existing method by introducing more efficient approach to model abstraction and derive theoretical error bounds on this approximation. Compared to [5], our technique makes use of average outcoming probability between clusters to define abstract transition matrix, which preserves system dynamics to a higher degree. We then compare both techniques on three case studies and demonstrate that these derived bounds are more accurate by up to 3 orders of magnitude and yield a considerable performance increase.

1.2 Related work

A widely studied model reduction method for Markov models is state aggregation based on (bi-)simulation equivalence [6]. This technique exploits symmetries of a concrete model and performs exact numerical computation but, unfortunately, can only be applied to a specific domain of Markov processes.

An alternative method to deal with large state spaces is truncation, e.g. fast adaptive uniformisation (FAU) [7, 8], where a lower bound on the transient probability distribution of the concrete model is computed, and the total probability mass that is lost due to this truncation is quantified. The efficiency of the truncation techniques depends on the distribution of the significant part of the probability mass over the states, and may result in poor accuracy if this mass is spread out over a large number of states, or whenever the selected window of states does not align with a property of interest.

Related to the state aggregation techniques, [9] presents an algorithm to approximate probability distributions of a Markov process forward in time, which served as an inspiration of the adaptive scheme proposed in [5], where a formal error analysis steers the adaptation. This novel use of derived error bounds allows far greater accuracy and flexibility as it accounts also for the past history of the probability mass within specific clusters. Compared to this method, we make use of average outcoming probability between clusters to define abstract transition matrix and provide theoretical bounds on such approximation.

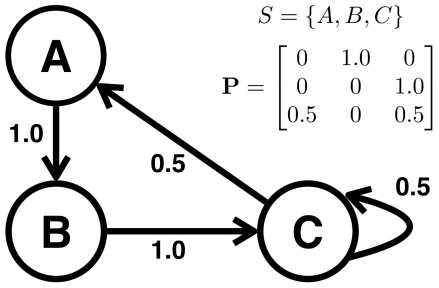


Figure 1. A simple DTMC.

2. State aggregation of Markov models

In this section, we use discrete-time Markov chain (DTMC) to introduce some basic terminology, explain philosophy behind adaptive aggregation techniques and point out differences between existing approach and a new one.

Formally, DTMC is defined as a double $(S, \mathbf{P})$, where

- $S = \{s_1, ..., s_n\}$ is the finite set of states;
- $\mathbf{P} : S \times S \mapsto [0,1]$ is the transition probability matrix, such that $\forall s \in S : \sum_{s' \in S} \mathbf{P}(s,s') = 1$

Figure 1 provides an example of a simple DTMC. The model is initialised via distribution $p_0 : S \mapsto [0,1], \sum_{s \in S} p_0(s) = 1$ and its transient probability distribution at time step $k \geq 0$ is

$$p_{k+1}(s) = \sum_{s' \in S} p_k(s') \mathbf{P}(s', s). \tag{1}$$

We are interested in providing an efficient computation of the vector p_k.

2.1 Adaptive aggregation techniques

Basically, a state aggregation is about clustering adjacent states into larger groups and treating such groups as usual states. Formally, we look for partition $\Phi = \{\varphi_1, ..., \varphi_m\}$, such that $\bigcup_{\varphi \in \Phi} \varphi = S$, $\forall i, j \in [1,m] : i \neq j \Rightarrow \varphi_i \cap \varphi_j = \emptyset$ and $m \ll n$. The latter condition represents our goal to reduce the state space to a reasonable ratio so that performance increase is achieved. Initial probability distribution π_0 of this abstract state space is simply

$$\pi_0(\varphi) = \sum_{s \in \varphi} p_0(s), \tag{2}$$

i.e. sum of all state probabilities within current cluster. Inverse transition from π_k to $\tilde{p}_k$, where $\tilde{p}_k$ represents approximate probability distribution at some time k, is performed the following way:

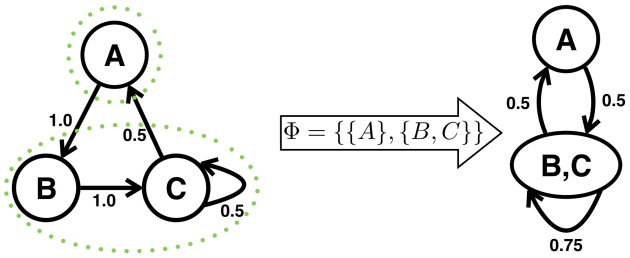


Figure 2. Abstraction that exploits average incoming probabilities may not preserve some properties of a DTMC.

$$\tilde{p}_k(s) = \frac{\pi_k(\varphi)}{|\varphi|}, s \in \varphi, \tag{3}$$

that is, the probability of a cluster is uniformly distributed between the states.

Since we introduced a new abstract state space Φ, abstract transition matrix $\Pi : \Phi \times \Phi \mapsto [0,1]$ must be defined as well. In [5], these transitions are computed the following way:

$$\Pi_{in}(\varphi, \varphi') = \frac{1}{|\varphi'|} \sum_{s \in \varphi} \sum_{s' \in \varphi'} \mathbf{P}(s, s'). \tag{4}$$

The intuition behind such transition matrix is that it encompasses the average *incoming* probability from clusters φ to φ'. Now we can calculate approximate transition probability at any given time $k \geq 0$ similarly as in 1:

$$\pi_{k+1}(\varphi) = \sum_{\varphi' \in \Phi} \pi_k(\varphi') \Pi(\varphi', \varphi). \tag{5}$$

Figure 2 illustrates a potential problem of this method: abstraction built upon a model may not preserve some important properties of a DTMC, namely, transition probabilities from a specific state may not sum to one. This is caused by approximating the transitions between abstract states as average incoming (instead of outcoming) probabilities. In Figure 3 we applied the same abstraction while using 6 to compute abstract transition matrix. This approach preserves all properties of DTMC and leads to considerable precision improvement (see Table 1).

$$\Pi_{out}(\varphi, \varphi') = \frac{1}{|\varphi|} \sum_{s \in \varphi} \sum_{s' \in \varphi'} \mathbf{P}(s, s') \tag{6}$$

The use of 4 in [5] is motivated by the structure of equations for explicit error bounds. Here we will eliminate this flaw by deriving such bounds while using 6 in our abstraction.

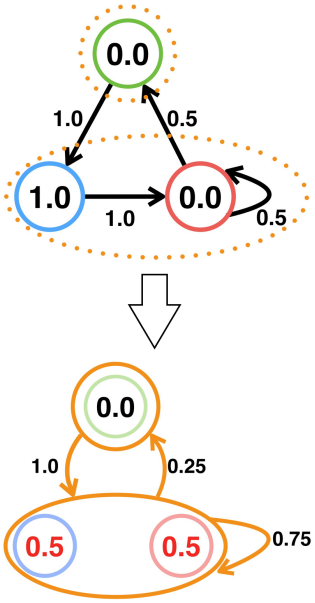


Figure 3. Here the numbers inside the circles denote probability distribution. Aggregation may lead to an error before any iteration is performed.

3. State aggregation error

Let us first examine how an error is produced when we use an abstract model instead of a concrete one. Let $(S, \mathbf{P})$ be DTMC and p_0 be initial probability distribution. Assume that a specific abstraction (Φ, Π) is given, where Π is computed using 6. Probability distribution π_0 of (Φ, Π) is computed using 2. Thus, we have already produced some error by approximating $p_0(s)$ by $\tilde{p}_0(s)$. Figure 3 illustrates this phenomenon.

Transition probabilities Π between clusters also serve as (an approximation of) transition probabilities between concrete states. Let $\tilde{P}$ denote approximate transition probability between states which is computed as

$$\tilde{P}(s, s') = \frac{\Pi(\varphi, \varphi')}{|\varphi'|}, s \in \varphi, s' \in \varphi' \tag{7}$$

Using these transitions instead of "real" ones for transporting probability mass between clusters (i.e. states) naturally generates an error as Figure 4 indicates.

3.1 Theoretical bounds on L_1 norm

Let $e_k(s) = \tilde{p}_k(s) - p_k(s)$ be approximation error for a probability distribution at time k. In order to reason about approximation precision, we use the next metric

$$||e_k||_1 = \sum_{s \in S} |e_k(s)|,$$

that is, an L_1 norm of an error vector. The next theorem explicitly characterises theoretical bound on approximation error.

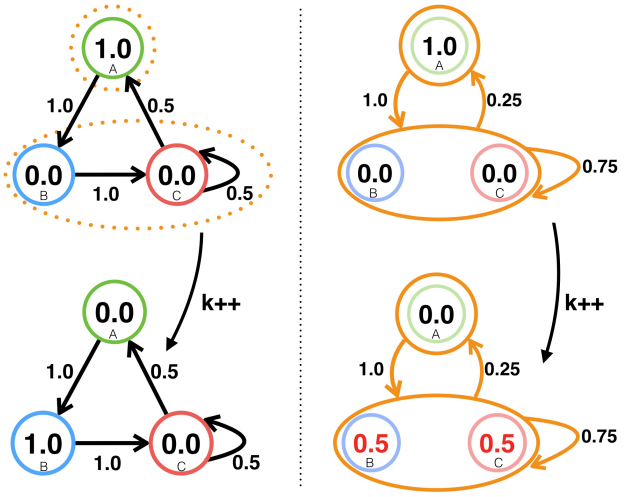


Figure 4. We perform one iteration with a concrete model and its abstraction; state B (C) has effectively lost (gained) some probability mass.

Theorem 1. Consider an abstraction upon $(S,\mathbf{P})$ characterised by (Φ,Π). Let $||e_0||_1$ be aggregation error. Then, for the L_1 norm of the error vector at time k, we obtain

$$||e_{k+1}||_1 \leq ||e_k||_1 + \sum_{\varphi' \in \Phi} \pi_k(\varphi')\tau(\varphi') \tag{8}$$

where

$$\tau(\varphi') = \sum_{\varphi \in \Phi} \sum_{s \in \varphi} \left| \frac{\Pi(\varphi',\varphi)}{|\varphi|} - \frac{\sum_{s' \in \varphi'} \mathbf{P}(s',s)}{|\varphi'|} \right| \tag{9}$$

represents average outgoing transition probability change for the current cluster, that is, the total difference between concrete transition probabilities and abstract ones. The product of this value with the probability of a cluster $\pi_k(\varphi')$ yields the amount of probability mass erroneously transported to other states. Finally, the sum over all clusters of such local error gives us the (upper bound of) total probability difference. The proof of this theorem is presented in Appendix.

3.2 Aggregation strategy

Now we look into 8 in more detail and derive some rules for efficient aggregation strategy. First, initial aggregation error $||e_0||_1$ encompasses state probability change during aggregation and tends to zero if the states with similar probability are clustered together. Second, error accrual in 8 is proportional to both $\pi(\varphi')$ and $\tau(\varphi')$; the latter value is usually more significant for larger cluster, e.g. consider what would the summand

$$\left| \frac{\Pi(\varphi',\varphi)}{|\varphi|} - \frac{\sum_{s' \in \varphi'} \mathbf{P}(s',s)}{|\varphi'|} \right|$$

be evaluated to if $|\varphi| = |\varphi'| = 1$. Thus, we could keep the resulting product low if we avoid using large clusters for states with significant probability.

Also, since the probability distribution over states is not constant over time, 8 implies that after some iterations our abstraction will start to generate a significant amount of error. Therefore, we are forced to adapt our abstraction to the new probability distribution (hence the word 'adaptive'), i.e. reaggregate the model. The resulting algorithm works by clustering the state space sequentially in time, where the quality of each aggregation can be quantified using 8.

3.3 Aggregation scheme

During model aggregation, probability distribution is used to define a maximum cluster size a state can be in. In general, the size of a cluster is inversely proportional to probabilities of concrete states, i.e. states with significant probability are usually aggregated to smaller clusters. In [5], an array of fixed sizes have been used to define aggregation scheme. Here we compute this scheme using the following formula:

$$maxClusterSize(p) = \left\lfloor M - \frac{M-1}{1-e^{-s}}(1-e^{-sp}) \right\rfloor, \tag{10}$$

where p is the probability of a state and M is a global maximum cluster size. For p ranging from 0 to 1, 10 yields sizes from M to 1. Parameter s represents a steepness of a curve and, along with parameter M, may be easily adjusted to achieve greater flexibility.

In [5] the state aggregation is driven by the state spatial locality, i.e. adjacent states are more likely to be clustered together. In our method we use a more general approach where states are aggregated according to their mutual transition probabilities, that is, two states connected by a transition with a high probability are more likely to be in a single cluster. This technique allows us to analyse models with a more complex structure of the underlying state space.

4. Experimental evaluation

The algorithms for both aggregation techniques have been implemented in PRISM (the explicit engine) [10]. We run all experiments on a CentOS 6.5 server with 12x Intel Xeon E5-2640 (6 cores at 2.5 GHz) and 64 GB RAM with all the algorithms being executed sequentially (1 thread). First, we evaluate both techniques on a simple model using neutral aggregation strategy, that is, the one that leads to a similar behaviour for both methods. We compare the accuracy of the empirical (i.e. compared to non-aggregating

simulation) results. For this experiment, we use a discretized version of the Lotka-Volterra model [5]. In the Table 1 we report a maximum value ("max") and an L_1 norm ("L_1") of the empirical error for and old technique exploiting incoming probabilities ("in") as well as for the new approach ("out"). It is clear that using average outgoing probabilities and, therefore, preserving the properties of DTMC, yields an accuracy increase by up to 5 orders of magnitude.

Table 1. Empirical accuracy comparison

		160k	360k	640k	1M
in	max	5.04E-5	7.56E-6	4.09E-6	2.74E-7
	L_1	5.31E-5	5.11E-5	1.22E-4	6.27E-5
out	max	5.02E-10	5.75E-9	7.78E-10	7.03E-10
	L_1	1.00E-9	4.24E-8	4.43E-8	1.68E-7

Next, we evaluate both methods on 3 different case studies with model sizes ranging from 400k to 2M states and compare both acceleration ("acc") and theoretical bounds ("bound") acquired. In order to ensure comparability between the two schemes, aggregation strategy is tuned individually for each model and technique. The goal here is to achieve maximum performance while keeping the theoretical bound relatively low. The models used here are, again, discretized versions of the Lotka-Volterra model, two-component signalling pathway and prokaryotic gene expression [5]. In all cases we can observe a clear improvement both in precision and performance compared to [5].

Table 2. Theoretical bound: Lotka-Volterra model.

		420k	800k	1.2M	1.7M
in	acc	5.46	4.73	4.82	4.82
	bound	4.13E-1	5.36E-1	8.63E-1	1.91E-1
out	acc	6.03	6.35	5.12	5.99
	bound	4.94E-3	9.74E-3	3.67E-3	8.43E-4

Table 3. Theoretical bound: Prokaryotic gene expression.

		850k	1.4M	1.7M	1.9M
in	acc	5.21	4.82	6.65	8.34
	bound	2.53E-1	2.65E-1	8.49E-1	9.85E-1
out	acc	6.48	6.88	8.12	9.71
	bound	5.39E-3	5.04E-3	5.16E-3	6.82E-3

5. Conclusions

In this work we have focused on adaptively-driven aggregation technique for Markov chains and have evaluated its key performance aspects. We then improved this method by introducing a new approach to state aggregation and deriving theoretical error bounds on such approximation. Our technique provides up to 3 orders of magnitude precision improvement, as well as significantly increases simulation performance. This allows one to efficiently analyse larger models with a higher accuracy. Future work will include a deeper study on the properties of Markov chains in order to apply this method for general models. We plan to introduce other metrics to control the error, more specifically, the infinity norm and the probability of satisfying a temporal logic specification. We also plan to apply our approach to the verification and performance analysis of complex safety-critical computer systems, where precision guarantees play a key role.

Table 4. Theoretical bound: Two-component signalling pathway.

		420k	920K	1.3M	1.6M
in	acc	1.59	2.45	2.49	3.27
	bound	3.51E-2	5.47E-2	7.79E-2	9.09E-1
out	acc	1.72	2.48	3.63	5.35
	bound	4.35E-4	5.84E-3	8.27E-4	6.86E-3

Acknowledgements

I would like to thank my supervisor Dr. Milan Češka for his guidance and valuable advices during the writing of this paper and the work on the project.

References

[1] C. Baier, E. M. Hanh, B. R. Haverkort, H. Hermanns, and J.-P. Katoen. Model checking for performability. *Mathematical Structures in Computer Science*, 23(4):751–795, 2013.

[2] Gunter Bolch, Stefan Greiner, Hermann de Meer, and Kishor S. Trivedi. *Queueing Networks and Markov Chains: Modeling and Performance Evaluation with Computer Science Applications*. Wiley-Interscience, New York, NY, USA, 1998.

[3] C. Madsen, C. J. Myers, N. Roehner, C. Winstead, and Z. Zhang. Utilizing stochastic model checking to analyze genetic circuits. In *2012 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*, pages 379–386, May 2012.

[4] Luca Cardelli, Marta Kwiatkowska, and Max Whitby. *Chemical Reaction Network Designs for Asynchronous Logic Circuits*, pages 67–81. Springer International Publishing, Cham, 2016.

[5] Alessandro Abate, Luboš Brim, Milan Češka, and Marta Kwiatkowska. *Adaptive Aggregation of Markov Chains: Quantitative Analysis of Chemical Reaction Networks*, pages 195–213. Springer International Publishing, Cham, 2015.

[6] Kim G. Larsen and Arne Skou. Bisimulation through probabilistic testing. *Information and Computation*, 94(1):1 – 28, 1991.

[7] Frits Dannenberg, Ernst Moritz Hahn, and Marta Kwiatkowska. Computing cumulative rewards using fast adaptive uniformization. *ACM Trans. Model. Comput. Simul.*, 25(2):9:1–9:23, February 2015.

[8] F. Didier, T. A. Henzinger, M. Mateescu, and V. Wolf. Fast adaptive uniformization of the chemical master equation. In *2009 International Workshop on High Performance Computational Systems Biology*, pages 118–127, Oct 2009.

[9] Sadegh Esmaeil Zadeh Soudjani and Alessandro Abate. *Precise Approximations of the Probability Distribution of a Markov Process in Time: An Application to Probabilistic Invariance*, pages 547–561. Springer Berlin Heidelberg, Berlin, Heidelberg, 2014.

[10] M. Kwiatkowska, G. Norman, and D. Parker. PRISM 4.0: Verification of probabilistic real-time systems. In G. Gopalakrishnan and S. Qadeer, editors, *Proc. 23rd International Conference on Computer Aided Verification (CAV'11)*, volume 6806 of *LNCS*, pages 585–591. Springer, 2011.

Appendix

Proof of Theorem 1. Let $t(s',s) = \tilde{\mathbf{P}}(s',s) - \mathbf{P}(s',s)$. Using 1 and the definition of e_k:

$$\begin{aligned} e_{k+1}(s) &= \tilde{p}_{k+1}(s) - p_{k+1}(s) = \sum_{s'\in S} \tilde{p}_k(s')\tilde{\mathbf{P}}(s',s) - p_{k+1}(s) = \sum_{s'\in S} (p_k(s') + e_k(s'))(\mathbf{P}(s',s) + t(s',s)) - p_{k+1}(s) \\ &= \sum_{s'\in S} p_k(s')\mathbf{P}(s',s) + \sum_{s'\in S} e_k(s')\mathbf{P}(s',s) + \sum_{s'\in S} (p_k(s') + e_k(s'))t(s',s) - p_{k+1}(s) \\ &= \sum_{s'\in S} e_k(s')\mathbf{P}(s',s) + \sum_{s'\in S} \tilde{p}_k(s)(\tilde{\mathbf{P}}(s',s) - \mathbf{P}(s',s)) \end{aligned}$$

We are interested in providing an L_1 norm of an error vector, that is, the sum of the absolute values of its components:

$$\begin{aligned} ||e_{k+1}||_1 = \sum_{s\in S} |e_{k+1}(s)| &= \sum_{s\in S} \left| \sum_{s'\in S} e_k(s')\mathbf{P}(s',s) + \sum_{s'\in S} \tilde{p}_k(s)(\tilde{\mathbf{P}}(s',s) - \mathbf{P}(s',s)) \right| \\ &\leq \sum_{s\in S} \left| \sum_{s'\in S} e_k(s')\mathbf{P}(s',s) \right| + \sum_{s\in S} \left| \sum_{s'\in S} \tilde{p}_k(s)(\tilde{\mathbf{P}}(s',s) - \mathbf{P}(s',s)) \right| \end{aligned}$$

Let us inspect the first term:

$$\begin{aligned} \sum_{s\in S} \left| \sum_{s'\in S} e_k(s')\mathbf{P}(s',s) \right| &\leq \sum_{s\in S}\sum_{s'\in S} \left|e_k(s')\right| \mathbf{P}(s',s) = \sum_{s'\in S}\sum_{s\in S} \left|e_k(s')\right| \mathbf{P}(s',s) = \sum_{s'\in S} \left|e_k(s')\right| \sum_{s\in S} \mathbf{P}(s',s) \\ &= \sum_{s'\in S} \left|e_k(s')\right| = ||e_k||_1 \end{aligned}$$

For the second term, running both summations by clusters and then using 3 and 7, we obtain

$$\begin{aligned} \sum_{s\in S} \left| \sum_{s'\in S} \tilde{p}_k(s)(\tilde{\mathbf{P}}(s',s) - \mathbf{P}(s',s)) \right| &= \sum_{\varphi\in\Phi}\sum_{s\in\varphi} \left| \sum_{\varphi'\in\Phi}\sum_{s'\in\varphi'} \tilde{p}_k(s)(\tilde{\mathbf{P}}(s',s) - \mathbf{P}(s',s)) \right| \\ &= \sum_{\varphi\in\Phi}\sum_{s\in\varphi} \left| \sum_{\varphi'\in\Phi}\sum_{s'\in\varphi'} \frac{\pi(\varphi')}{|\varphi'|} \left(\frac{\Pi(\varphi',\varphi)}{|\varphi|} - \mathbf{P}(s',s) \right) \right| \\ &= \sum_{\varphi\in\Phi}\sum_{s\in\varphi} \left| \sum_{\varphi'\in\Phi} \pi(\varphi') \left(\sum_{s'\in\varphi'} \frac{\Pi(\varphi',\varphi)}{|\varphi'||\varphi|} - \sum_{s'\in\varphi'} \frac{\mathbf{P}(s',s)}{|\varphi'|} \right) \right| \\ &= \sum_{\varphi\in\Phi}\sum_{s\in\varphi} \left| \sum_{\varphi'\in\Phi} \pi(\varphi') \left(\frac{\Pi(\varphi',\varphi)}{|\varphi|} - \frac{\sum_{s'\in\varphi'} \mathbf{P}(s',s)}{|\varphi'|} \right) \right| \\ &\leq \sum_{\varphi\in\Phi}\sum_{s\in\varphi}\sum_{\varphi'\in\Phi} \pi(\varphi') \left| \frac{\Pi(\varphi',\varphi)}{|\varphi|} - \frac{\sum_{s'\in\varphi'} \mathbf{P}(s',s)}{|\varphi'|} \right| \\ &= \sum_{\varphi'\in\Phi}\sum_{\varphi\in\Phi}\sum_{s\in\varphi} \pi(\varphi') \left| \frac{\Pi(\varphi',\varphi)}{|\varphi|} - \frac{\sum_{s'\in\varphi'} \mathbf{P}(s',s)}{|\varphi'|} \right| \\ &= \sum_{\varphi'\in\Phi} \pi(\varphi') \sum_{\varphi\in\Phi}\sum_{s\in\varphi} \left| \frac{\Pi(\varphi',\varphi)}{|\varphi|} - \frac{\sum_{s'\in\varphi'} \mathbf{P}(s',s)}{|\varphi'|} \right| \\ &= \sum_{\varphi'\in\Phi} \pi(\varphi')\tau(\varphi') \end{aligned}$$

Finally, by adding both terms:

$$||e_{k+1}||_1 \leq ||e_k||_1 + \sum_{\varphi' \in \Phi} \pi(\varphi')\tau(\varphi')$$

we complete the proof. □