The Nsolve File Formats

Peter Buchholz
Informatik IV, University of Dortmund
D-44227 Dortmund, Germany
Email: peter.buchholz@udo.edu

1 Short Introduction to Nsolve

`Nsolve` is a program to determine the stationary solution of hierarchical Markovian models exploiting a hierarchical Kronecker structure of the generator matrix. The model class contains stochastic automata networks (SANs) without functional transition rates and superposed generalized stochastic Petri nets (SGSPNs) which are both described by a hierarchical model with one state in the upper level. A hierarchical model consists of a number of components (LLQPNs) and one high level model (HLQPN). Detailed descriptions of the required matrix structure and some solution methdos can be found in [3, 6, 4]. The `Nsolve` software is part of the APNN-Toolbox and is also available as a standalone program from the author upon request.

`Nsolve` includes iterative solution techniques using the hierarchical Kronecker representation of the matrix and it includes as well iterative techniques using a sparse matrix representation. In the latter case, the flat matrix is first generated out of the hierarchical Kronecker representation. Observe that the implementation of the generation is not very efficient such that solution techniques for sparse matrices should only be used for models with a modest state space size ($\leq 10^5$).

`Nsolve` requires the following files:

- $\langle$ `name` $\rangle$`.conf` configuration file (Description in 2.1)

- $\langle$ `name` $\rangle$`0.mat` HLQPN matrix (Description in 2.2)
- $\langle$ `name` $\rangle$`.spa` HLQPN state space (Description in 2.3)
- $\langle$ `name` $\rangle j$`.mat` LLQPN j matrix (Description in 2.4)
  A file for each LLQPN has to exist. The number of LLQPNs can be found in the configuration file.

All files may include comments. Comments start with # as the first character in a line, then the rest of the line is interpreted as a comment!

The program is called with

```
Nsolve  name
```

2 File Formats

The section introduces the different file formats which are needed for `Nsolve`. The description starts with the configuration file $\langle$ `name` $\rangle$`.conf`. Afterwards the matrix file for the HLQPN (Sect. 2.2), the HLQPN state space (Sec. 2.3) and for the LLQPN matrix file (Sec. 2.4) are described. Additional we introduce the file format for vectors (Sec. 2.5).

2.1 Configuration File

The configuration file includes the following values (plus possibly some comment see the format for comments above):

```
Number of LLQPNs (integer ≥ 0)
Solution method (integer ≥ 0)
Maximum number of iterations (integer ≥ 0)
```

Nr.	Method	Explanation
		Matrix structure test
0	NO SOLUTION	Matrix generation without solution
1	MAT_GEN	Flat matrix generation
		Solution techniques for sparse matrices
2	LU	LU-Factorization ([13] pp. 66ff, [9] pp. 30ff)
11	POWER	Power-Method ([13] pp. 121ff)
12	JOR	JOR ([13] pp. 127f)
13	SOR	SOR ([13] pp. 128f)
21	POWER A/D	Power-Method with Aggregation/Disaggregation ([9], pp. 51ff)
22	JOR A/D	JOR with Aggregation/Disaggregation ([9], pp. 51ff)
23	SOR A/D	SOR with Aggregation/Disaggregation ([9], pp. 51ff)
24	KMS	KMS ([13], pp. 308)
25	ML	Multi-level method ([11])
31	GMRES	GMRES ([13], pp. 198ff)
32	BICGSTAB	BiCGSTAB ([15]))
33	TFQMR	Transpose free variant of QMR ([10]))
61	POWER ILU0	Power-Method with ILU0 Preconditioning ([13], pp. 145)
62	POWER ILUTH	Power-Method with ILUTH Preconditioning ([13], pp. 145)
63	GMRES ILU0	GMRES with ILU0 Preconditioning ([13],pp. 205ff, 145)
64	GMRES ILUTH	GMRES with ILUTH Preconditioning ([13],pp. 205ff, 146)
65	BICGSTAB ILU0	BICGSTAB with ILU0 Preconditioning
66	BICGSTAB ILUTH	BICGSTAB with ILUTH Preconditioning
67	TFQMR ILU0	TFQMR with ILU0 Preconditioning
68	TFQMR ILUTH	TFQMR with ILUTH Preconditioning
		Solution techniques for Kronecker representations
111	POWER	Power-Method ([9] pp. 203)
112	JOR	JOR ([1])
113	SJOR	SJOR ([9] pp. 204f)
114	SOR	Real SOR ([14])
121	POWER A/D	Power-Method with Aggregation/Disaggregation ([2, 3])
122	JOR A/D	JOR with Aggregation/Disaggregation ([2, 3])
123	SJOR A/D	SJOR with Aggregation/Disaggregation ([2, 3])
131	GMRES	GMRES ([13], Spp 198ff)
132	DQGMRES	DQGMRES ([12])
133	ARNOLDI	Arnoldi-Method ([13], pp. 191)
134	CGS	CGS ([13], pp. 222)
135	BICGSTAB	BiCGSTAB ([15]))
136	TFQMR	TFQMR ([10]))
150	PREPOWER	Power with Neumann preconditioning ([6])
151	PREGMRES	GMRES with Neumann preconditioning ([6])
152	PREARNOLDI	Arnoldi with Neumann preconditioning ([6])
154	PRECGS	CGS with Neumann preconditioning ([5])
155	PREBICGSTAB	BICGSTAB with Neumann preconditioning ([5])
156	PRETFQMR	TFQMR with Neumann preconditioning([5])
157	BSOR_BICGSTAB	BICGSTAB with Block SOR preconditioning ([8])
157	BSOR_GMRES	GMRES with Block SOR preconditioning ([8])
157	BSOR_TFQMR	TFQMR with Block SOR preconditioning ([8])
172	STR_BSOR_T	Two level BSOR ([7])

Table 1: Overview of available solution methods

ϵ_1 (real ≥ 0.0)
ϵ_2 (real ≥ 0.0)
ϵ_3 (real ≥ 0.0)
Maximum CPU-time (real ≥ 0.0)
Relaxation-parameter (real ≥ 0.0)
Output overall vector (Boolean)
Output HLQPN vector (Boolean)

for all LLQPNs $1, \ldots,$ Number of LLQPNs
Output LLQPN vector (Boolean)

Initial state HLQPN (integer ≥ -2)

if initial state HLQPN ≥ 0 then
for all LLQPNs $1, \ldots,$Number of LLQPNs
Initial state LLQPN (integer ≥ 0)

Values in the configuration file are interpreted as follows.

Number of LLQPNs
The number of components which form the model.

Solution method
Most of the currently available solution methods are summarized in table 1. The number in the first column of the table has to be used in the configuration file.

Maximal number of iterations
The solution stops, if the number of iterationshais been reached

ϵ_1
The solution stops, if the absolute value if the largest element in the residual vector becomes smaller than ϵ_1 . The residual vector is currently checked every 10 iterations in most methods. Observe that in some methods like the projection methods TFQMR and BICGSTAB the residual can only be estimated during runtime. Thus, this methods may stop too early or too late. In any case, the true residual is computed after termination of the method.

ϵ_2
The value is used for several purposes. In A/D-methods, an A/D step for the HLQPN is performed if the residual norm projected onto the HLQPN state space exceeds ϵ_2. For methods with ILUTH preconditioner, the value is interpreted as the threshold for keeping non-zero elements during the incomplete factorization.

ϵ_3
The value is only used for A/D-methods where an A/D step for a LLQPN is performed, if the residual norm projected onto the LLQPN state exceeds ϵ_3.

CPU time limit
The method stops, if the CPU time of the solution (excluding the time for matrix generation) exceeds the limit.

Relaxation-parameter
Relaxation-parameter for the methods JOR, SJOR, SOR and BSOR. Usually the value should be taken from $(0, 2)$.

Output parameters for solution vectors
Parameter values can be 0 (= no output) or 1 (= output). The first parameter is for the complete solution vector, the second for the solution vector of the HLQPN and then follow the values for all LLQPNs. Solution vectors are written in files. File `name_overall.vec` contains the complete vector, `name0.vec` HLQPN vector and `name`j`.vec` the vector for LLQPN j.

The initial distribution
Different methods exist to define the initial distribution. The first value determines the type of the initial distribution. If the first value is larger or equal to zero, then it is assumed that a single initial state is defined. In this case the following values include the initial states of the LLQPNs. Observe that not every combination of states defines a correct state. If inconsistencies are detected, the program tries to repair the state and prints a warning. The distribution is initialized with 1.0 at the initial state and 0.0 elsewhere. Usually a uniform distribution is recommended as initial distribution because in a hierarchical

model all states are reachable if the state space is generated appropriately (see [4]). A uniform distribution is generated if the first state value is set to -2. In this case the following values are not interpreted. The last possibility to define the initial distribution is to read it from a file. In this case the value in the configuration file has to be set to -1. The name of the file has to be `name_overall.reach`. The vector length has to be equal to the number of states in the state space of the hierarchical model. The vector is normalized to 1 before the iteration starts. With this feature result vectors of one method can be used as initial vectors of another method.

Apart from the parameters in the configuration file, some parameters exist in the source files of the different methods. However, modifications of these parameters requires a modification of the source and recompilation of the program. We tried to set the parameters to reasonable values, but the choices might not be optimal for specific examples.

2.2 HLQPN Matrix File Format

The HLQPN Matrix contains the visible transitions in the HLQPN including the weights/rates at this level. For consistency reason the file contains information about a block structure of the matrix. However, usually the matrix consists of a single block.

```
Number of matrices in the file (integer) (will not be interpreted but should be 1)
Number of partition groups (one partition group is sufficient)
first state of partition group 0 (must be 0!)
...
last state of the last partition group (= Number of states of the HLQPN)

For each row of the HLQPN matrix:
Number of non-zero elements

For each non-zero element
destination state
transition number
rate
(non-zero elements need not be in the right order, but for each
transition number and destination state only one element is allowed)
```

Transition number have to correspond to transition numbers used in the LLQPN matrix files.

2.3 HLQPN State Space File Format

The file includes the correspondence between a HLQPN state and the LLQPN state which form the state. The number of HLQPN states and the number of LLQPNs are not included in the file. The number of HLQPN states is defined in the file with the HLQPN matrix and the number of LLQPNs can be found in the configuration file. These files are read before the HLQPN state file.

```
For each HLQPN state
Macro state number for LLQPN 1, ..., Macro state number for LLQPN J
(where J is the number of LLQPNs)
```

2.4 LLQPN Matrix File Format

A LLQPN is described by a number of matrices. Each matrix includes transitions according to one transition (synchronized) or a set of transitions (local). Transitions are numbered and number 0 is used for local transitions. Synchronized or external visible transitions are numbered with values > 0. Usually the numbers of synchronized transitions are not consecutive if the matrices have been generated with the APNN-toolbox. The numbers of transitions in the LLQPN matrix files correspond to the transition numbers in the HLQPN matrix file. If for a synchronized transition in the HLQPN no matrix is available for a LLQPN, then it is assumed that the corresponding matrix is an identity matrix. Usually matrices in a LLQPN matrix file should be ordered according to the transition number. If the matrices are not ordered, the program `Preproc` can be used to generate an ordered version.

LLQPN matrices are partioned into blocks which are numbered consecutively starting with 0. The block numbers correspond to the macro state numbers used in the HLQPN state file. Thus, if macro state number x appears in the HLQPN state for LLQPN j, then the matrix for LLQPN j has to contain at least $x + 1$ blocks (blocks are numbered from 0).

```
Number of matrices (one local and one for each non-identity synchronized transition)
Number of blocks
first state in the first block (must be 0!)
...
last state in the last block (= number of states of the LLQPN)

For each LLQPN Matrix
transition number

For each row of the matrix
Number of non-zero elements

For each non-zero element
destination state
weight
```

2.5 Vector Files

Two file formats for vectors exist which differ only by an initial integer. We denote the format with the initial integer as extended format. Most times the normal (non-extended) format is used. Values in vectors can be Integer, Real or Boolean (0/1).

```
Integer for classification (only for extended format)
Number of elements
Value of the first element
...
Value of the last element
```

References

[1] P. Buchholz. Numerical solution methods based on structured descriptions of Markovian models. In G. Balbo and G. Serazzi, editors, *Computer Performance Evaluation - Modelling Techniques and Tools*, pages 251–267. Elsevier, 1992.

[2] P. Buchholz. An aggregation/disaggregation algorithm for stochastic automata networks. *Probability in the Engineering and Informational Sciences*, 11(2):229–253, 1997.

[3] P. Buchholz. An adaptive aggregation/disaggregation algorithm for hierarchical Markovian models. *European Journal of Operational Research*, 116(3):85–104, 1999.

[4] P. Buchholz. Hierarchical structuring of superposed GSPNs. *IEEE Transactions on Software Engineering*, 25(2):166–181, 1999.

[5] P. Buchholz. Projection methods for the analysis of stochastic automata networks. In B. Plateau, W. J. Stewart, and M. Silva, editors, *Numerical Solution of Markov Chains (NSMC'99)*, pages 149–168. Prensas Universitarias de Zaragoza, 1999.

[6] P. Buchholz. Structured analysis approaches for large Markov chains. *Applied Numerical Mathematics*, 31(4):375–404, 1999.

[7] P. Buchholz and T. Dayar. Block sor for kronecker structured representations. In *Accepted for Int. Workshop on the Numerical Solution of Markov Chains*, 2003.

[8] P. Buchholz and T. Dayar. Block SOR preconditioned projection methods for Kronecker structured Markovian representations. Technical Report TUD-FI03-05, Department of Computer Science, Dresden University of Technology, 2003. http://www.inf.tu-dresden.de/Fak/berichte.html.

[9] P. Buchholz, J. Dunkel, B. Müller-Clostermann, M. Sczittnick, and S. Zäske. *Quantitative Systemanalyse mit Markovschen Ketten*. Teubner-Texte zur Informatik Band 8. Teubner, 1994.

[10] R. W. Freund and M. Hochbruck. On the use of two QMR for solving singular systems and applications in Markov chain modelling. *Numerical Linear Algebra and Applications*, 1:403–420, 1994.

[11] G. Horton and S. Leutenegger. A multi-level solution algorithm for steady state Markov-chains. *ACM Performance Evaluation Review*, 22:191–200, 1994.

[12] Y. Saad and K. Wu. DQGMRES: a direct quasi-minimal rsidual algorithm based on incomplete orthogonalization. *Numerical Linear Algabra and Applications*, 3:329–343, 1996.

[13] W. J. Stewart. *Introduction to the numerical solution of Markov chains.* Princeton University Press, 1994.

[14] E. Uysal and T. Dayar. Iterative methods based on splittings for stochastic automata networks. *European Journal of Operational Research*, 110(1):166–186, 1998.

[15] H. A. van der Vorst. Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. *SIAM Journal on Scientific Computing*, 13:631–644, 1992.