ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

MATHICSE

Mathematics Institute of Computational Science and Engineering

School of Basic Sciences - Section of Mathematics

MATHICSE Technical Report

Nr. 26.2015

September 2015

Benchmark problems on stochastic automata networks in tensor train format

Francisco Macedo

http://mathicse.epfl.ch

Phone: +41 21 69 37648

Address:
EPFL - SB - MATHICSE (Bâtiment MA)
Station 8 - CH-1015 - Lausanne - Switzerland

Fax: +41 21 69 32545

BENCHMARK PROBLEMS ON STOCHASTIC AUTOMATA NETWORKS IN TENSOR TRAIN FORMAT

FRANCISCO MACEDO*†

24th September 2015, Version 1.1

Abstract. We present a description of transition rate matrices of models for Stochastic Automata Networks (SAN) in their representation as operators in Tensor Train (TT) format. The collection contains models from real-life applications both taken from references directly or adaptations that still have a realistic interpretation. A classification is given according to the properties of interest of each model. Identifiers based on such properties are provided. A brief description of each model is given. This document both provides an illustration of the big variety of applications associated with Stochastic Automata Network as it also allows testing and tuning algorithms for the solution of relevant associated problems as, for instance, the determination of their steady-state.

1. Introduction. In many areas, from which we highlight Chemical Networks and Queueing Networks, one finds models that are associated with Stochastic Automata Networks, which are Markov Chains with a particular structure. Both developing algorithms and testing software strongly benefit from the availability of collections of such models.

Such collections can be used, among other things, for: tuning algorithms and optimizing its performance across a wide and representative range of problems; testing the correctness of code; measuring its performance and robustness; comparing different algorithms for some measures, among which the main one is typically the computation time.

We combine problems directly extracted from references with others that are adjusted from them while still representing realistic situations. Problems for which something is known about the solution are always particularly attractive. The practice of reproducible research, meaning that research is published in a way that the experiments can be reproduced, is of growing interest and visibility [17, 38, 52]. In this context, the availability of well documented and maintained benchmark collections is fundamental.

Having collections of problems implemented in software have been a particular concern in some fields. Linear algebra and optimization are the first that come to mind. For the first, one can highlight [29], from which most matrices were later incorporated into the MATLAB gallery function. A big collection of sparse matrices from practical applications can be found in [13]. Concerning optimization, we highlight the widely used Cute and Cuter testing environments [7, 26], while other more specialized collections are also available.

The growing interest in finding the stationary distribution of Stochastic Automata Networks (SAN), and more generally the interest for determining some measures of interest associated with such networks, motivates us to create one such collection. The standard problem is the determination of the solution of $Q^T x = 0$, where Q is the transition rate matrix associated with the SAN, as this is the condition on x, for a continuous-time Markov Chain, for the steady-state. Q is what characterizes a model and it can be similar even for the representation of models that have fairly different real life meanings.

*EPF Lausanne, SB-MATHICSE-ANCHP, Station 8, CH-1015 Lausanne, Switzerland, {francisco.macedo}@epfl.ch

†IST, Alameda Campus, Av. Rovisco Pais, 1, 1049-001 Lisbon, Portugal

The interest extends to the study of the transient behaviour of such networks, associated with the dynamics over time, instead of focusing only on the long-term distribution. In the field of Chemical Networks, in particular, this is associated with the Chemical Master Equation (CME), used to describe the time-evolution of a system that can be modelled as being in exactly one of countable number of states at any given time. The equations are usually a set of differential equations for the variation over time of the probabilities that the system occupies each of the different states, where the standard problem is finding $x(t)$ such that $\frac{d}{dt}x(t) = Qx(t)$.

This collection of models, represented by their transition rate matrices, is associated with realistic settings, in their majority taken from existing references. Some of them have been used in numerical experiments in the context of testing algorithms for solving the two problems mentioned above.

There are many important characteristics of the models, which may, in particular, influence the way algorithms perform, and should thus be used to partition the set of models. We provide a classification of each model according to such structural properties.

The collection is implemented in MATLAB, where the matrices are represented in the associated operator for Tensor Train (TT) format as this is the format for which algorithms for finding the stationary state seem to perform the best as recently illustrated in [32], while this then allows much bigger problem sizes to be stored compared to when one has a full matrix representation. It is also illustrated in [30] that the closely related Quantized Tensor Train (QTT) format is very suitable for the problem of the transient behaviour in the context of CME.

The code that allows obtaining the TT representation of the models is available in `http://anchp.epfl.ch/SAN_TT`.

In Sec. 2 we provide the characteristics of interest that will be used to classify the models, as well as a more detailed description of the associated concepts. In Sec. 3 we go through the models one by one and describe them. In Sec. 4 we focus on the utilization of our available implementation of the operators representing the models. Sec. 5 consists of the conclusions.

2. Identifiers. In Tab. 1 we provide a list with the different main characteristics of the models. We then provide some insight on the concepts associated with transition rate matrices of Markov Chains; we move then to the particular case of SAN; we finish with the application to the particular tensor format, Tensor Train (TT) format, on which the representation of the corresponding transition rate matrices is done.

Table 1: Identifiers for the properties of the models.

chemical	queueing	communications	qualitycontrol
reducible			
distinguishable			
productform			
funct	synch	functlocal	
scalable			
chain			

2.1. From general Markov Chains to Stochastic Automata Networks. The **transition rate matrix** of a **Markov Chain** is the matrix Q that contains the rate of transition from state i to state j in the entry $Q(i,j)$. In each row, the diagonal entry is given by, for a Markov Chain with n states associated with a matrix $Q \in \mathcal{R}^{n\times n}$,

$$Q(i,i) = -\sum_{j=1, j\neq i}^{n} Q(i,j), \tag{2.1}$$

negative sum of the off-diagonal elements of the corresponding row. Its absolute value represents the rate of leaving state i as the rate of leaving is the minimum of independent random variables following exponential distributions which is known to also follow an exponential distribution with rate that is the sum of the rates, which can be easily proved using the uniqueness of the survival function of the different distributions. Matrix Q is thus a matrix with sum of rows equal to 0, where all non-diagonal entries are non-negative, while all diagonal entries are non-positive.

If there is only one eigenvalue 0 in matrix Q, at least one such eigenvalue exists given the restriction on the sum of the rows of Q, the associated Markov Chain is called **irreducible** (**reducible**, otherwise). This is of particular interest, for instance, in the context of finding the stationary distribution (steady-state) of the associated Markov Chain as irreducible implies having a unique solution for the problem.

A **Stochastic Automata Network** (SAN) represents a model where there are separate systems that interact between them.

The states of the Markov Chain associated with such network are the combinations of states of the different systems, which implies that there is an exponential growth of the state space dimension with the number of systems.

The high dimensionality of this state space is usually coped with by exploiting the rich structure of the transition rate matrix associated with such networks.

Many alternatives have been tried in the past, in the context of finding the steady-state or studying the transient behaviour.

For the first, one possibility is to use model reduction, in particular with the possibility for a wide range of models of determining **product form solutions**. The basic idea of this reduction is to yield a system for which the stationary distribution factorizes into a product of distributions for the individual processes. This reduced system then allows for a much less expensive numerical treatment. General techniques for arriving at product form solutions are described, e.g., in [34]. Extensive work has been done on finding conditions under which such product form approach applies; see [21, 20] for some recent results. However, its practical range of applicability is limited to specific subclasses of models.

Concerning the transient behaviour, and in particular for the **Chemical Master Equation** (CME), in most cases it cannot be solved explicitly and various Monte Carlo simulation techniques have been used to find approximations of the probability densities by producing either detailed or approximate realizations of each process [24, 23, 25]. However, for many systems, biologically important events may occur rarely, necessitating the generation of a prohibitively large set of realizations to obtain sufficiently precise statistics.

While other techniques exist to deal with the two mentioned problems, they all seem to have significant disadvantages. The approach on which the Tensor-Train (TT) operators are based relates with the observation that the transition rate matrix of a communicating Markov process can often be represented by a short sum of Kronecker

products [43]. This property can then be exploited, for instance, when performing matrix-vector multiplications [35] to reduce the cost of iterative solvers significantly. Complexity scales then linearly with the state space dimension.

In [32, 6, 30] the fact that a vector containing joint probabilities can be naturally rearranged into a tensor is explored. The first two address the problem of the stationary distribution while the last addresses the CME problem. When considering, for example, a network of d communicating finite state Markov processes, an associated vector, for instance for the steady-state or with the probabilities at a given time, has length $n_1 n_2 \cdots n_d$, where n_μ denotes the number of states in the μth process for $\mu = 1, \ldots, d$. Quite naturally, the entries of this vector can be rearranged into an $n_1 \times \cdots \times n_d$ array, defining a dth order tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$.

This then allows us to use established low-rank tensor approximation techniques [27]. In particular the use of matrix product states (MPS) or, equivalently, TT decompositions, is proposed in [32], targeting the computation of the stationary distribution for finite-dimensional communicating Markov processes with development and comparison of different algorithms. TT format is shown to be very well-suited for this problem, thus motivating that we provide in this document proper representations of the transition rate matrix for the different models of interest in this format. In its turn, the closely related Quantized Tensor Train (QTT) format is suitable for the problem associated with the transient behaviour, CME, as observed in [30].

Algorithms require the repeated application of $\mathbf{Q}^T \in \mathbb{R}^{(n_1 \cdots n_d) \times (n_1 \cdots n_d)}$ to a vector in TT decomposition. It is therefore important to represent $\mathbf{Q}$ in a form that allows to perform this operation efficiently.

An *operator TT decomposition*, also called *matrix product operator* (MPO), takes the form

$$\mathbf{Q}_{(i_1,\ldots,i_d),(j_1,\ldots,j_d)} = A_1(i_1, j_1) \cdot A_2(i_2, j_2) \cdots A_d(i_d, j_d), \tag{2.2}$$

where $A_\mu(i_\mu, j_\mu) \in \mathbb{R}^{t_{\mu-1} \times t_\mu}$, and $t_0 = t_d = 1$. This is the TT decomposition applied to $\mathrm{vec}(\mathbf{Q})$, merging each index pair (i_μ, j_μ) in lexicographical order into a single index ranging from 1 to n_μ^2. The ranks of the operator are the entries t_μ, $\mu = 0, ..., d$. The tensor $\tilde{\mathcal{X}}$ resulting from a matrix-vector product with $\mathbf{Q}$ then has a simple TT decomposition as desired.

The transition rate matrix has the general representation

$$\mathbf{Q} = \mathbf{Q}_L + \mathbf{Q}_I,$$

with $\mathbf{Q}_L$ representing local transitions and $\mathbf{Q}_I$ representing interactions. The local part takes the form

$$\mathbf{Q}_L = \sum_{\mu=1}^{d} I_{n_1} \otimes \cdots \otimes I_{n_{\mu-1}} \otimes L_\mu \otimes I_{n_{\mu+1}} \otimes \cdots \otimes I_{n_d}, \tag{2.3}$$

where $L_\mu \in \mathbb{R}^{n_\mu \times n_\mu}$ contains the local transitions in the μth system.

2.2. Detailed definitions and properties. We now explain the different characteristics that are chosen to partition the models, in Tab. 1, which requires going further on some of the previously introduced definitions.

The first row of the table is associated with the applications from which the SAN are extracted. There are four main fields of reference: **Chemical Networks**;

Queueing Networks; **Communications**; **Quality Control**. We use the identifiers chemical, queueing, communications and qualitycontrol, respectively.

Concerning the second row, the concept of **reducible** model was already introduced, implicitly, as we defined **irreducible** model, while a model is **reducible** when it is not **irreducible**. The corresponding identifier is reducible.

For the third row, we note that the different states of a system of a given model are associated with a numerical quantity. In particular, they are generally ordered in increasing way according to this quantity. It represents: a number of customers for Queueing Networks; a number of particles for Chemical Networks; a number of users for Communications; a number of deficiencies for Quality Control. However, there are models for which there are **distinguishable** customers (or particles or users or deficiencies), for which one needs a vector of numbers, instead of just a number, to characterize the state of a particular system. This follows the idea in [11], where such vector follows a decreasing lexicographic ordering. The size of this vector is the number of types of customers plus one, as the first entry is the number of available places in the system, difference between the capacity and the sum of the customers of the different types, while the other entries are the number of customers from the different types. The concept can naturally generalized to the other types of SAN, and if a model has distinguishable customers or particles or users or deficiencies, it gets the identifier distinguishable.

For the fourth row, a model is said to have **product form**, concept introduced before, in case one can represent the interactions of the systems in a way that they are seen as independent for the purpose of the computation of the steady-state. This is possible due to the tendency, as time goes by, for the SAN to become more and more stagnated as it approaches the stationary state. This is particularly interesting as then the exact steady-state can be obtained with clearly less computational effort than using the algorithms that are generally required. They are also relevant to check the correction of the solutions that are obtained using such algorithms. The identifier for this is productform. We add this identifier in the cases where there is at least one reference that explicitly mentions the existence of such model reduction for a particular model. Additionally, this is trivial in the presence of independent systems, implying that these also get the identifier.

For the fifth row, the transitions can be **functional**, meaning that the rates depend on the state of at least one system; and/or **synchronized**, meaning that the transition causes a change in the state of more than one system simultaneously. If a model includes functional transitions, it gets the identifier funct; while if it includes synchronized transitions, it gets the identifier synch. Note that a model can have both types of transitions. As it can be the case that a functional transition is local, meaning that the rates that depend on the state of a system are associated with that same system only, we should have a separate identifier as such functional transitions have no influence on the topology of the network. For these, we use the identifier functlocal.

For the sixth row, identifier scalable is used to denote that the dimension of the problem (number of automata) is a parameter.

For the seventh row, identifier chain is used for the models on which the states associated with the individual systems can only be changed to an immediate neighbour, which means that the topology of each system is one dimensional. In practice, all matrices associated with the systems that take part in the operator TT decomposition will have bandwidth 1. This is particularly important in the context of knowing

if it makes sense, for instance, to apply geometric coarsening when using a tensorized multigrid method as the one proposed in [6] for finding the steady-state.

3. Collection of models. This section contains a description of the models in the collection. The identifiers for the properties of the models are listed inside curly brackets after the name of the model. The models are summarized in Tab. 2.

From here on, '*' represents, in the diagonal of matrices, the negative sum of the off-diagonals of the corresponding row. Entries of vectors are usually identified with round brackets except in some cases where subscripts are used instead to make the notation less heavy. This should be clear from the context.

overflow{queueing, funct, scalable, chain}.

In this model, in the numerical experiments of [12] (in the context of finding the stationary distribution of SAN using well-known CP format for tensors instead of TT - again based on the reshaping of the vector of the states into a tensor), customers can arrive in any of the queues and if a queue is full, they continue searching for one that is not full. The queues are ordered and customers can only try to find a place in the following queues, until reaching the last one. If they are all full, they are dropped from the network.

The local part, L_μ, consists of the local arrivals (if the queue is not full) and departures of customers after they are served

$$L_\mu = \begin{bmatrix} * & \mathrm{arr}(\mu) & 0 & \cdots & 0 \\ \mathrm{dep}(\mu) & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \mathrm{arr}(\mu) \\ 0 & \cdots & 0 & \mathrm{dep}(\mu) & * \end{bmatrix},$$

where 'arr' is the vector containing the arrival rates while 'dep' contains the service rates, for the different queues.

By direct calculation, it can be shown that $\mathbf{Q}$ admits an operator TT decomposition (2.2) with the cores

$$A_1(i_1, j_1) = \begin{bmatrix} L_1(i_1, j_1) & \mathrm{arr}(1) B_1(i_1, j_1) & I_1(i_1, j_1) \end{bmatrix}, \quad A_d(i_d, j_d) = \begin{bmatrix} I_d(i_d, j_d) \\ C_d(i_d, j_d) \\ L_d(i_d, j_d) \end{bmatrix},$$

and

$$A_\mu(i_\mu, j_\mu) = \begin{bmatrix} I_\mu(i_\mu, j_\mu) & 0 & 0 \\ C_\mu(i_\mu, j_\mu) & B_\mu(i_\mu, j_\mu) & 0 \\ L_\mu(i_\mu, j_\mu) & \mathrm{arr}(\mu) B_\mu(i_\mu, j_\mu) & I_\mu(i_\mu, j_\mu) \end{bmatrix}. \quad \mu = 2, \ldots, d-1.$$

for matrices $B_\mu, C_\mu \in \mathbb{R}^{n_\mu \times n_\mu}$, where the first represents the customer finding a full queue while the other one represents the first non-full queue that such customer finds

$$B_\mu = \begin{bmatrix} 0 & \cdots & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & \ddots & 0 & 0 \\ 0 & \cdots & 0 & 1 \end{bmatrix}, \quad C_\mu = \begin{bmatrix} * & 1 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & 1 \\ 0 & \cdots & \cdots & 0 & * \end{bmatrix}.$$

Table 2: Models in the collection.

Name	Description
overflow	Overflow model.
overflowper	Overflow model where a cycle is introduced.
overflowsim	Overflow model where only the next queue can be accessed by a customer arriving at a full one.
overflowpersim	Overflow model where a cycle is introduced and only the next queue can be accessed by a customer arriving at a full one.
backupqueue	Model where if a customer arrives in a full queue, he goes to a backup queue.
kanban	Kanban control model.
kanbanalt	Kanban control model where customers leave if next is full.
kanbanalt2	Kanban control model where customers are only served if next is not full.
indepbirthdeath	Birth-death process on independent systems.
indepMM1queue	M/M/1 queueing on independent systems.
indepMMcqueue	M/M/c queueing on independent systems.
indepbalking	Balking on independent systems.
indeppriority	Priority queues on independent systems.
kanbanalt2batcharr	Kanban control model where customers are only served if next is not full, and first queue has batch arrivals.
kanbanalt2batchdep	Kanban control model where customers are only served if next is not full, and there are batch departures.
kanbanalt2MMc	Kanban control model where customers are only served if next is not full, with more than one server per queue.
kanbanalt2balking	Kanban control model where customers are only served if next is not full, and first queue follows a balking law.
kanbanalt2priority	Kanban control model where customers are only served if next is not full, and customers have different priorities.
cyclequeue	Cyclic queueing network.
compfailure	Component failure.
compfailuregen	Component failure with global repair in four cases.
compfailureorig	Component failure with global repair in case of maximum possible deficiencies.
multiprogramming	Multiprogramming computer system.
impcustomers	Impatient customers.
handoff1class1d	Wireless network with handoff for one class of customers and in 1D space.
handoff1class2d	Wireless network with handoff for one class of customers and in 2D space.

handoff2class1d	Wireless network with handoff for two classes of customers and in 1D space.
handoff	Wireless network with handoff for two classes of customers and in 2D space.
directedmetab	Directed metabolic pathways.
reversiblemetab	Reversible metabolic pathways.
dilutionintermed	Dilution of intermediates.
cyclicmetab	Cyclic metabolic pathways.
endproduct	End-product inhibition.
divergingmetab	Diverging metabolic pathways.
convergingmetab	Converging metabolic pathways - combined fluxes.
convergingmetab2	Converging metabolic pathways - reaction with two fluctuating substrates.
michaelismenten	Michaelis-Menten model.
enzymekineticsI	Enzyme kinetics I.
enzymekineticsII	Enzyme kinetics II.
multiscalechemical	Multiscale chemical network.
enzymaticfutile	Enzymatic futile cycle.
cascadegene	Cascade gene regulatory model.
toggleswitch	Toggle switch.

One can include the negative sum of the off-diagonals inside C_μ, as well as for L_μ, because all other matrices in the corresponding Kronecker product are diagonal. The proof of this statement can be found in Appendix A (more concretely in Th. 2). Note that, in the case of the local part, L_μ, this will hold for all models; recall (2.3). Also note that the theorem will never be possible to use for synchronized transitions as, for this type of interaction, directly from the definition, at least two of the involved matrices of the corresponding Kronecker product will not be diagonal.

The TT ranks t_μ are all 3, independently of d.

Concerning the parameters, they are, in this order: vector 'arr'; vector 'dep'; vector with the maximum capacity of each system $(n-1)$.

One proposed possibility for their values is found in [12] - arrival rates ('arr'): $[1.2, 1.1, 1.0, 0.9, 0.8, 0.7]$; departure rates ('dep'): $[1, 1, 1, 1, 1, 1]$; maximum capacities: $[16, 16, 16, 16, 16, 16]$. These are the default values used in our implementation. We are thus considering a problem of size 17^6.

overflowper{queueing, funct, scalable, chain}.

This model results from an adaptation of the previous one. The first queue is now considered as the queue after the last meaning that a customer that finds the previously called last queue full can then try to join the first, and so on.

For building the TT operator, the dimensions need to be ordered but the one that is chosen to be the first is now irrelevant.

The matrices that are needed to build the operator in TT format are the same as

in the previous model, for a new operator TT decomposition (2.2) with the cores

$$A_1(i_1, j_1) = \begin{bmatrix} L_1(i_1, j_1) & I_1(i_1, j_1) & \mathrm{arr}(1)B_1(i_1, j_1) & C_1(i_1, j_1) \end{bmatrix}, \quad A_d(i_d, j_d) = \begin{bmatrix} I_d(i_d, j_d) \\ L_d(i_d, j_d) \\ C_d(i_d, j_d) \\ B_d(i_d, j_d) \end{bmatrix},$$

$$A_2(i_2, j_2) = \begin{bmatrix} I_2(i_2, j_2) & 0 & 0 & 0 & 0 & 0 \\ L_2(i_2, j_2) & I_2(i_2, j_2) & \mathrm{arr}(2)B_2(i_2, j_2) & 0 & 0 & 0 \\ \mathrm{arr}(1)C_2(i_2, j_2) & 0 & \mathrm{arr}(1)B_2(i_2, j_2) & 0 & C_2(i_2, j_2) & B_2(i_2, j_2) \\ 0 & 0 & 0 & \mathrm{arr}(2)B_2(i_2, j_2) & I_2(i_2, j_2) & \end{bmatrix},$$

for $A_{d-1}(i_{d-1}, j_{d-1})$ we have

$$\begin{bmatrix} I_{d-1}(i_{d-1}, j_{d-1}) & 0 & 0 & 0 \\ L_{d-1}(i_{d-1}, j_{d-1}) & I_{d-1}(i_{d-1}, j_{d-1}) & \mathrm{arr}(d-1)B_{d-1}(i_{d-1}, j_{d-1}) & 0 \\ C_{d-1}(i_{d-1}, j_{d-1}) & 0 & B_{d-1}(i_{d-1}, j_{d-1}) & 0 \\ 0 & 0 & 0 & B_{d-1}(i_{d-1}, j_{d-1}) \\ 0 & 0 & 0 & \mathrm{arr}(d)I_{d-1}(i_{d-1}, j_{d-1}) + \mathrm{arr}(d-1)B_{d-1}(i_{d-1}, j_{d-1}) \\ 0 & 0 & 0 & \mathrm{arr}(d)C_{d-1}(i_{d-1}, j_{d-1}) \end{bmatrix}$$

and

$$A_\mu(i_\mu, j_\mu) = \begin{bmatrix} I_\mu(i_\mu, j_\mu) & 0 & 0 & 0 & 0 & 0 \\ L_\mu(i_\mu, j_\mu) & I_\mu(i_\mu, j_\mu) & \mathrm{arr}(\mu)B_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ C_\mu(i_\mu, j_\mu) & 0 & B_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ 0 & 0 & 0 & B_\mu(i_\mu, j_\mu) & 0 & 0 \\ 0 & 0 & 0 & \mathrm{arr}(\mu)B_\mu(i_\mu, j_\mu) & I_\mu(i_\mu, j_\mu) & 0 \\ 0 & 0 & 0 & 0 & 0 & B_\mu(i_\mu, j_\mu) \end{bmatrix},$$

$\mu = 3, \ldots, d-2$.

Concerning the parameters, they are the same as in overflow - vector 'arr'; vector 'dep'; vector with the maximum capacity of each system $(n-1)$.

The default parameters of our implementation are those from the mentioned model: arrival rates ('arr'): $[1.2, 1.1, 1.0, 0.9, 0.8, 0.7]$; departure rates ('dep'): $[1, 1, 1, 1, 1, 1]$; maximum capacities: $[16, 16, 16, 16, 16, 16]$. We are thus considering a problem of size 17^6.

overflowsim{queueing, funct, scalable, chain}.

The next model is again adapted from overflow. The change is that if the customer finds a full queue when he arrives, he can only try to join the next. If this one is also full, he simply leaves the network.

The matrices that are needed for the construction of the operator are the same as in overflow, for a TT decomposition (2.2) with the cores

$$A_1(i_1, j_1) = \begin{bmatrix} L_1(i_1, j_1) & \mathrm{arr}(1)B_1(i_1, j_1) & I_1(i_1, j_1) \end{bmatrix}, \quad A_d(i_d, j_d) = \begin{bmatrix} I_d(i_d, j_d) \\ C_d(i_d, j_d) \\ L_d(i_d, j_d) \end{bmatrix},$$

and

$$A_\mu(i_\mu, j_\mu) = \begin{bmatrix} I_\mu(i_\mu, j_\mu) & 0 & 0 \\ C_\mu(i_\mu, j_\mu) & 0 & 0 \\ L_\mu(i_\mu, j_\mu) & \mathrm{arr}(\mu)B_\mu(i_\mu, j_\mu) & I_\mu(i_\mu, j_\mu) \end{bmatrix}. \quad \mu = 2, \ldots, d-1.$$

Concerning the parameters, they are the same as in overflow - vector 'arr'; vector 'dep'; vector with the maximum capacity of each system $(n-1)$.

The default parameters of our implementation are those from the mentioned model: arrival rates ('arr'): $[1.2, 1.1, 1.0, 0.9, 0.8, 0.7]$; departure rates ('dep'): $[1, 1, 1, 1, 1, 1]$; maximum capacities: $[16, 16, 16, 16, 16, 16]$. We are thus considering a problem of size 17^6.

overflowpersim {queueing, funct, scalable, chain}.

This model is a combination of the models overflowper and overflowsim. From the first, we allow customers to go to the first queue in case they find the last one full; while from the second we only allow a customer finding a full queue to try entering the next one, meaning that if the next is also full, he leaves the network.

The TT operator that is obtained, associated with the same matrices as in the previous models, is

$$\begin{aligned} A_1(i_1, j_1) &= \begin{bmatrix} L_1(i_1, j_1) & \text{arr}(1)B_1(i_1, j_1) & C_1(i_1, j_1) & I_1(i_1, j_1) \end{bmatrix}, \\ A_d(i_d, j_d) &= \begin{bmatrix} I_d(i_d, j_d) & C_d(i_d, j_d) & \text{arr}(d)B_d(i_d, j_d) & L_d(i_d, j_d) \end{bmatrix}^T, \end{aligned}$$

and

$$A_\mu(i_\mu, j_\mu) = \begin{bmatrix} I_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ C_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ 0 & 0 & I_\mu(i_\mu, j_\mu) & 0 \\ L_\mu(i_\mu, j_\mu) & \text{arr}(\mu)B_\mu(i_\mu, j_\mu) & 0 & I_\mu(i_\mu, j_\mu) \end{bmatrix}, \quad \mu = 2, \ldots, d-1.$$

Concerning the parameters, they are the same as in overflow - vector 'arr'; vector 'dep'; vector with the maximum capacity of each system $(n-1)$.

The default parameters of our implementation are those from the mentioned model: arrival rates ('arr'): $[1.2, 1.1, 1.0, 0.9, 0.8, 0.7]$; departure rates ('dep'): $[1, 1, 1, 1, 1, 1]$; maximum capacities: $[16, 16, 16, 16, 16, 16]$. We are thus considering a problem of size 17^6.

backupqueue {queueing, funct, scalable, chain}.

The next model is again an adaptation of overflow. Customers again try to find a queue that is not full after arriving in a particular one, but the queue they search for if they arrive in a full one is always the same now - the 'backup' queue. If this one is full, they simply leave the network.

The corresponding operator is then, again for the same matrices as for the previous models, where the 'backup' queue is associated with the last dimension while the order of the remaining ones it not relevant,

$$A_1(i_1, j_1) = \begin{bmatrix} L_1(i_1, j_1) & \text{arr}(1)B_1(i_1, j_1) & I_1(i_1, j_1) \end{bmatrix}, \quad A_d(i_d, j_d) = \begin{bmatrix} I_d(i_d, j_d) \\ C_d(i_d, j_d) \\ L_d(i_d, j_d) \end{bmatrix},$$

and

$$A_\mu(i_\mu, j_\mu) = \begin{bmatrix} I_\mu(i_\mu, j_\mu) & 0 & 0 \\ 0 & I_\mu(i_\mu, j_\mu) & 0 \\ L_\mu(i_\mu, j_\mu) & \text{arr}(\mu)B_\mu(i_\mu, j_\mu) & I_\mu(i_\mu, j_\mu) \end{bmatrix}, \quad \mu = 2, \ldots, d-1.$$

As in all models until this point, the ranks are small and independent of d.

Concerning the parameters, they are the same as in overflow - vector 'arr'; vector 'dep'; vector with the maximum capacity of each system $(n-1)$.

The default parameters of our implementation are those from the mentioned model: arrival rates ('arr'): $[1.2, 1.1, 1.0, 0.9, 0.8, 0.7]$; departure rates ('dep'): $[1, 1, 1, 1, 1, 1]$;

maximum capacities: $[16, 16, 16, 16, 16, 16]$. We are thus considering a problem of size 17^6.

kanban{queueing, distinguishable, synch, scalable}.

In this model, used in the experiments of [11, 6], customers arrive in the first queue being then served in sequence until being served in a last one, leaving then the network - for more details about this kind of models see [39]. This is the general structure of the well-known subclass of Tandem Networks.

We assume an infinite source - when the first queue is not full, there is automatically a new arrival. If a customer finishes the service and the next queue is full, he waits in the queue where he was served.

Each system has, as in the previous models, a maximum capacity, and the local transitions are associated with a customer finishing the service, as then this customer stays in the same system but with a different 'label'. In fact, in this model there are two types of customers in each system - already served and waiting to be served.

$B_\mu \in \mathbb{R}^{n_\mu \times n_\mu}$ now represents the departure of a customer that was served already while $C_\mu \in \mathbb{R}^{n_\mu \times n_\mu}$ represents the arrival of that same customer in the next system. As for the local part, $L_\mu \in \mathbb{R}^{n_\mu \times n_\mu}$, it represents the service of a customer, thus resulting in one less customer waiting to be served and one more waiting to move to the next queue (for the last system, there is no next queue and the customer directly leaves the network).

We omit the representation of the matrices as, for the middle dimensions, it is not trivial to represent them for a general maximum capacity now that the states have a non-trivial ordering (based on the one introduced in [11] as noted before) while, for the same reason, their interpretation is also not straight-forward. For a representation of these matrices for a fixed, small, maximum capacity of the systems, the reader is referred to [11].

For the first and last dimensions, the operator is more similar to the ones for the previous models as there is again only one type of customer. While in the first system one has automatic arrivals when the queue is not full implying that the number of waiting customers to be served is always the maximum possible, in the last there are no served customers as they directly leave the network.

For the local part, associated with finished services, one gets

$$L_1 = \begin{bmatrix} * & \mathrm{dep}(1) & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \mathrm{dep}(1) \\ 0 & \cdots & \cdots & 0 & * \end{bmatrix}, \quad L_d = \begin{bmatrix} * & 0 & \cdots & \cdots & 0 \\ \mathrm{dep}(d) & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \mathrm{dep}(d) & * \end{bmatrix}.$$

Note that we get a superdiagonal matrix in the first dimension given that the states are now associated with a decreasing order.

As for the interaction matrices, B_1 represents now, as the other B_μ, $\mu = 2, ..., d$, the departure of a customer. As for C_d, as all other C_μ, C_μ, $\mu = 1, ..., d-1$, as it is involved in a Kronecker product with matrices that are not all diagonal, in particular

B_{d-1}, one cannot now make the diagonal correction inside. We have

$$B_1 = \begin{bmatrix} 0 & \cdots & \cdots & \cdots & 0 \\ 1 & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & 0 \end{bmatrix}, \quad C_d = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & 1 \\ 0 & \cdots & \cdots & \cdots & 0 \end{bmatrix}.$$

The representation is, similarly to overflowsim,

$$A_1(i_1, j_1) = \begin{bmatrix} L_1(i_1, j_1) & \text{move}(1)B_1(i_1, j_1) & \text{move}(1)E_1(i_1, j_1) & I_1(i_1, j_1) \end{bmatrix},$$
$$A_d(i_d, j_d) = \begin{bmatrix} I_d(i_d, j_d) & C_d(i_d, j_d) & F_d(i_d, j_d) & L_d(i_d, j_d) \end{bmatrix}^T,$$

and

$$A_\mu(i_\mu, j_\mu) = \begin{bmatrix} I_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ C_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ F_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ L_\mu(i_\mu, j_\mu) & \text{move}(\mu)B_\mu(i_\mu, j_\mu) & \text{move}(\mu)E_\mu(i_\mu, j_\mu) & I_\mu(i_\mu, j_\mu) \end{bmatrix},$$

$\mu = 2, \ldots, d-1$. Matrices $E_\mu, F_\mu \in \mathbb{R}^{n_\mu \times n_\mu}$ are diagonal matrices containing the negative sum of the off-diagonal entries per row of B_μ and C_μ, respectively, which now have to be done separately, as Th. 2 does not hold, directly implying an increase in the rank of the operator. 'dep' contains again the service rates while 'move' contains the rates of moving to the next queue, for already served customers. This last has only $d-1$ entries as this is not needed for the last system noting again that a customer that finishes service directly leaves the network.

Concerning the parameters, they are, in this order: vector 'dep'; vector 'move'; vector with the maximum capacity of each system.

One possibility is to use those from [10] - service rates ('dep'): $[1, 1, 1, 1]$; moving rates ('move'): $[10, 10, 10]$; maximum capacities: $[10, 10, 10, 10]$. These are the default parameters in our implementation. We are thus considering a problem of size $11 \times 66^2 \times 11$.

kanbanalt{queueing, synch, scalable, chain}.

This model is adapted from the previous one. We consider again a Tandem Network but now the customers leave the network if the service in a system is finished and the next one is full, instead of waiting. This way, if the next queue is not full, once a service is complete, customers can simply move instantaneously. This implies, in particular, that only one type of customer exists.

The local part, which only exists for the first and last systems, is

$$L_1 = \begin{bmatrix} * & \text{dep}(1) & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \text{dep}(1) \\ 0 & \cdots & \cdots & 0 & * \end{bmatrix}, \quad L_d = \begin{bmatrix} * & 0 & \cdots & \cdots & 0 \\ \text{dep}(d+1) & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \text{dep}(d+1) & * \end{bmatrix}.$$

As for the interactions, while B_μ is associated with the synchronized departure of customers, C_μ is associated with receiving them

$$B_\mu = \begin{bmatrix} 0 & \cdots & \cdots & \cdots & 0 \\ 1 & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & 0 \end{bmatrix}, \quad C_\mu = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & 0 & 1 \\ 0 & \cdots & \cdots & 0 & 1 \end{bmatrix}.$$

The operator is

$$A_1(i_1, j_1) = \begin{bmatrix} L_1(i_1, j_1) & \mathrm{dep}(2)B_1(i_1, j_1) & \mathrm{dep}(2)E_1(i_1, j_1) & I_1(i_1, j_1) \end{bmatrix},$$

$$A_d(i_d, j_d) = \begin{bmatrix} I_d(i_d, j_d) & C_d(i_d, j_d) & F_d(i_d, j_d) & L_d(i_d, j_d) \end{bmatrix}^T,$$

and

$$A_\mu(i_\mu, j_\mu) = \begin{bmatrix} I_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ C_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ F_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ 0 & \mathrm{dep}(\mu+1)B_\mu(i_\mu, j_\mu) & \mathrm{dep}(\mu+1)E_\mu(i_\mu, j_\mu) & I_\mu(i_\mu, j_\mu) \end{bmatrix},$$

$\mu = 2, \ldots, d-1$.

'dep' has $d+1$ entries as the first entry is the influx rate, while the remaining entries are, as usual, the ordered service rates for the different systems.

Concerning the parameters, they are: vector 'dep'; vector with the maximum capacities.

The default vector 'dep' in our implementation is $[1, 1, 1, 1, 1, 1, 1]$; and maximum capacities: $[16, 16, 16, 16, 16, 16]$. We are thus considering a problem of size 17^6.

kanbanalt2{queueing, synch, scalable, chain}.

This model is also adapted from kanban, in particular being again associated with a Tandem Network. The difference to kanbanalt is that we avoid having two types of customers by now not allowing a customer to be served unless the next queue is not full, again avoiding, as in the mentioned model, to have served customers that need to wait to move to the next queue.

The operator is the same as in the previous model, with some slight differences in the matrices that compose it. The main difference is in C_μ.

As we change the parameters of the model, it is worth showing the new operator and the matrices that compose it. The departure rates are again in a vector denoted by 'dep'; while the arrival rate, included in the vector 'dep' for the previous model, is now a separate entry denoted by 'arr'.

We get the local part

$$L_1 = \begin{bmatrix} * & \mathrm{arr} & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \mathrm{arr} \\ 0 & \cdots & \cdots & 0 & * \end{bmatrix}, \quad L_d = \begin{bmatrix} * & 0 & \cdots & \cdots & 0 \\ \mathrm{dep}(d) & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \mathrm{dep}(d) & * \end{bmatrix}.$$

For the interaction matrices,

$$B_\mu = \begin{bmatrix} 0 & \cdots & \cdots & \cdots & 0 \\ 1 & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & 0 \end{bmatrix}, \quad C_\mu = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & 1 \\ 0 & \cdots & \cdots & \cdots & 0 \end{bmatrix}.$$

The operator is

$$A_1(i_1, j_1) = \begin{bmatrix} L_1(i_1, j_1) & \mathrm{dep}(1)B_1(i_1, j_1) & \mathrm{dep}(1)E_1(i_1, j_1) & I_1(i_1, j_1) \end{bmatrix},$$
$$A_d(i_d, j_d) = \begin{bmatrix} I_d(i_d, j_d) & C_d(i_d, j_d) & F_d(i_d, j_d) & L_d(i_d, j_d) \end{bmatrix}^T,$$

and

$$A_\mu(i_\mu, j_\mu) = \begin{bmatrix} I_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ C_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ F_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ 0 & \mathrm{dep}(\mu)B_\mu(i_\mu, j_\mu) & \mathrm{dep}(\mu)E_\mu(i_\mu, j_\mu) & I_\mu(i_\mu, j_\mu) \end{bmatrix},$$

$\mu = 2, \ldots, d-1$.

Concerning the parameters, they are: arrival rate 'arr'; vector 'dep'; vector with the maximum capacities.

The default parameters are: 'arr' is 1.2; vector 'dep' is $[1, 1, 1, 1, 1, 1]$; maximum capacities $[16, 16, 16, 16, 16, 16]$. We are thus considering a problem of size 17^6.

indepbirthdeath {chemical, productform, functlocal, scalable, chain}.

The next model is extracted from the numerical experiments of [30]. The idea is to consider a well-known birth-death process on each system, which are assumed to be independent meaning that we only have local transitions.

Birth-death processes are usually characterized by the creation of particles according to linear rates on the number of existing particles in the system. This comes from the assumption that particles have a common rate of reproduction and, in each instant, each of them can reproduce, implying that the rate is the sum of the rates as the minimum of independent random variables following exponential distribution is again exponential with such rate, as noted before. In this model, it is however considered that the creation rate is just a constant.

The operator is, for this context of independence, simple to obtain

$$A_1(i_1, j_1) = \begin{bmatrix} L_1(i_1, j_1) & I_1(i_1, j_1) \end{bmatrix}, \quad A_d(i_d, j_d) = \begin{bmatrix} I_d(i_d, j_d) \\ L_d(i_d, j_d) \end{bmatrix},$$

and

$$A_\mu(i_\mu, j_\mu) = \begin{bmatrix} I_\mu(i_\mu, j_\mu) & 0 \\ L_\mu(i_\mu, j_\mu) & I_\mu(i_\mu, j_\mu) \end{bmatrix}, \quad \mu = 2, \ldots, d-1.$$

This is the simplest possible representation, with all ranks taking the value 2 independently of d.

The involved matrices, denoting the vector with the production rates of particles in the different systems by 'prate' and the vector with the destruction rates by 'drate', are

$$L_\mu = \begin{bmatrix} * & \text{prate}(\mu) & 0 & \cdots & 0 \\ \text{drate}(\mu) & \ddots & \ddots & \ddots & \vdots \\ 0 & 2 \times \text{drate}(\mu) & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \text{prate}(\mu) \\ 0 & \cdots & 0 & (n_\mu - 1) \times \text{drate}(\mu) & * \end{bmatrix}.$$

The degragation rates are of the form (3.3) as it can happen to each particle individually, in each instant, which means that we, again, have a random variable that is the minimum of independent random variables following exponential distribution.

Concerning the parameters, they are: vector 'prate'; vector 'drate'; vector with the maximum capacities.

One can use, for instance, those in [30] - 'prate' is $[1000, 1000, 1000, 1000, 1000]$; 'drate' is $[1, 1, 1, 1, 1]$; maximum capacity is $[4095, 4095, 4095, 4095, 4095]$. In our implementation we have 'prate' as $[1000, 1000, 1000, 1000, 1000, 1000]$; 'drate' as $[1, 1, 1, 1, 1, 1]$; maximum capacities $[16, 16, 16, 16, 16, 16]$. We are thus, again, considering a problem of size 17^6.

indepMM1queue {queueing, productform, scalable, chain}.

We continue with models with independent systems but changing the dynamics associated with each system. The goal is to explore some concepts from queueing theory. The operator will remain the same as in the previous model while the difference is in the matrices L_μ.

We now assume, in each queue, that we have a M/M/1 queueing system; see, for instance, [44, Ch. 2]. There is one server and customers arrive and are served following exponential distributions. As there is only one server, the service rate is the same independently of the number of customers, while we also consider constant arrival rates.

We have, denoting the vector with the arrival rates by 'arr' and the one with the service rates by 'dep',

$$L_\mu = \begin{bmatrix} * & \text{arr}(\mu) & 0 & \cdots & 0 \\ \text{dep}(\mu) & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \text{arr}(\mu) \\ 0 & \cdots & 0 & \text{dep}(\mu) & * \end{bmatrix}.$$

Concerning the parameters, they are: vector 'arr'; vector 'dep'; vector with the maximum capacities.

In our implementation we have: 'arr' as $[1.2, 1.2, 1.2, 1.2, 1.2, 1.2]$; 'dep' as $[1, 1, 1, 1, 1, 1]$; maximum capacities $[16, 16, 16, 16, 16, 16]$. We are thus, again, considering a problem of size 17^6.

indepMMcqueue {queueing, productform, functlocal, scalable, chain}.

We now consider, again for independent systems, a generalization of M/M/1. We consider M/M/c queueing systems now; see [44, Ch. 2]. The difference to

indepMM1queue is that we have c servers instead of just one. This results in an effective rate of service of, again because of what the minimum of independent random variables following exponential distributions is,

$$\begin{cases} m \times \text{ser}, m < c \\ c \times \text{ser}, m \geq c \end{cases},$$

where 'm' denotes the number of customers in the queue and 'ser' denotes the service rate, which is assumed to be common to all servers of a particular queue.

We have

$$L_\mu = \begin{bmatrix} * & \text{arr}(\mu) & 0 & \cdots & \cdots & \cdots & 0 \\ \text{dep}(\mu) & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & 2 \times \text{dep}(\mu) & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & C(\mu) \times \text{dep}(\mu) & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \text{arr}(\mu) \\ 0 & \cdots & \cdots & \cdots & 0 & C(\mu) \times \text{dep}(\mu) & * \end{bmatrix},$$

where 'arr' is the vector of the arrival rates, while 'dep' is the one with the service rates, for the different systems. 'C' is a vector with the number of servers by system.

Concerning the parameters, they are: vector 'arr'; vector 'dep'; vector with the maximum capacities; vector 'C'.

In our implementation we have: 'arr' as $[1.2, 1.2, 1.2, 1.2, 1.2, 1.2]$; 'dep' as $[1, 1, 1, 1, 1, 1]$; maximum capacities $[16, 16, 16, 16, 16, 16]$; 'C' as $[3, 3, 3, 3, 3, 3]$. We are thus, again, considering a problem of size 17^6.

indepbalking {queueing, productform, functlocal, scalable, chain}.

We continue with the sequence of models with independent systems. We now assume that each system suffers from the effect of 'balking'. As in general birth-death processes (even if not in the one considered in indepbirthdeath), the rates of arrival are not constant, varying with the number of customers in the queue. The particularity of 'balking' is that the rate will decrease as the number of customers increases as the concept means that customers are less and less eager to join a queue if they find more customers waiting to be served; see [44, Ch. 2].

Different functions for such decreasing interest can be considered, for instance those in [44, Ch. 2]. We choose the one, from this reference, for which the rate is given by

$$\begin{cases} \text{in}, m < s \\ \frac{1}{m-s+2} \times \text{in}, m \geq s \end{cases}, \tag{3.1}$$

where 's' is a parameter and 'm' again represents the number of customers in the queue. 'in' denotes a constant associated with the arrival rate - its maximal value. This value is multiplied with a factor that is smaller and smaller as the number of customers increases, giving the effective arrival rate.

The local matrices are now

$$L_\mu = \begin{bmatrix} * & \mathrm{arr}(\mu) & 0 & \cdots & \cdots & \cdots & 0 \\ \mathrm{dep}(\mu) & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \mathrm{arr}(\mu) & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \frac{1}{2} \times \mathrm{arr}(\mu) & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \frac{1}{n_\mu - S(\mu)+1} \times \mathrm{arr}(\mu) \\ 0 & \cdots & \cdots & \cdots & 0 & \mathrm{dep}(\mu) & * \end{bmatrix},$$

where 'arr' is the vector with the maximal arrival rates, ('in' in 3.1); 'dep' is the one for service rates; 'S' is a vector containing the different 's', for the different systems, associated with (3.1).

Concerning the parameters, they are: vector 'arr'; vector 'dep'; vector with the maximum capacities; vector 'S'.

In our implementation we have: 'arr' as $[1.2, 1.2, 1.2, 1.2, 1.2, 1.2]$; 'dep' as $[1, 1, 1, 1, 1, 1]$; maximum capacities $[16, 16, 16, 16, 16, 16]$; 'S' as $[3, 3, 3, 3, 3, 3]$. We are thus, again, considering a problem of size 17^6.

indeppriority {queueing, distinguishable, productform, scalable}.

Still in the context of models with independent systems, we focus on the class of priority queues. We assume that there are different types of customers, who get different treatments. This implies using the same procedure for the definition of the states of each system as in kanban.

One can define models where different customers are treated differently in many different ways. We use the setting described in the third example of [42]. We have customers of two types, where a customer of type II is only served if there is no customer of type I waiting, which is the way to impose that these last ones have priority.

The operator is again the same as in the previous models while the matrices L_μ are omitted for the same reason as for kanban.

Concerning the parameters, they are: 'arr', vector of size two with the arrival rates of the customers of the two types, for any system; 'dep', vector of size two with the service rates of the customers of the two types, for any system; vector with the maximum capacities.

In our implementation we have: 'arr' as $[1.2, 1.2]$; 'dep' as $[1, 1]$; maximum capacities $[10, 10, 10, 10]$; We are thus considering a problem of size 66^4.

kanbanalt2batcharr {queueing, synch, scalable}.

We now combine another important concept from queueing theory with the topology from kanbanalt2.

The concept is the one of batch. A batch is associated with the possibility that more than one customer arrives/departs at the same time, for a given system; see, for instance, [45, Ch. 8].

We here assume that such groups of customers are always of the same size, for a fixed system.

In this model, we consider the possibility of batch arrivals.

The idea of combining this concept with kanbanalt2 is simple as the first queue is the only one with arrivals from the exterior in this model and so we simply add the

possibility of an arrival of a batch in that queue.

The TT operator is thus the same as in kanbanalt2, with L_1 changing to

$$L_1 = \begin{bmatrix} * & 0 & \cdots & 0 & \text{arr} & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \text{arr} \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & * \end{bmatrix},$$

where 'arr' denotes the arrival rate of the batch, while the size of the batch is the distance between '*' and 'arr' (either in a row or in a column).

Concerning the parameters, they are: value 'arr'; vector with the service rates, associated with 'dep' in the previous models; vector with the maximum capacities; value for the batch size.

In our implementation we have: 'arr' as 1.2; 'dep' as $[1, 1, 1, 1, 1, 1]$; maximum capacities $[16, 16, 16, 16, 16, 16]$; batch size 3. We are thus, again, considering a problem of size 17^6.

kanbanalt2batchdep {queueing, synch, scalable}.

This model is again based on the combination between kanbanalt2 and the concept of batch, but considering now batch departures instead of arrivals.

We incorporate them by allowing batch departures in all systems. The TT representation is then again the one from kanbanalt2, while matrices B_μ and C_μ change to

$$B_\mu = \begin{bmatrix} * & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 1 & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 1 & 0 & \cdots & 0 & * \end{bmatrix}, \quad C_\mu = \begin{bmatrix} * & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & 1 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & *. \end{bmatrix}.$$

L_d also changes as it is also associated with departures, from the last system,

$$L_d = \begin{bmatrix} * & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \mathrm{dep}(d) & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \mathrm{dep}(d) & 0 & \cdots & 0 & * \end{bmatrix}.$$

'dep' denotes the vector of the service rates; the size of the batch is the distance between '*' and dep(d) for the last dimension, while between '*' and 1 in B_μ and C_μ for the remaining ones.

Concerning the parameters, they are: value for the influx rate, 'arr' in kanbanalt2; vector 'dep'; vector with the maximum capacities; vector with the batch sizes for the different systems.

In our implementation we have: 'arr' as 1.2; 'dep' as $[1, 1, 1, 1, 1, 1]$; maximum capacities $[16, 16, 16, 16, 16, 16]$; batch sizes $[3, 3, 3, 3, 3, 3]$. We are thus, again, considering a problem of size 17^6.

kanbanalt2MMc{queueing, funct, synch, scalable, chain}.

The next models are, as the previous one, based on combining concepts from queueing theory with kanbanalt2. In particular we will use now the concepts of queueing theory that were before considered for networks with independent systems.

We now consider M/M/c queues, as in indepMMcqueue, allowing multiple servers in the different systems.

The TT representation is again the one from kanbanalt2, while some matrices that compose it change. In particular, the ones related with the departures

$$B_\mu = \begin{bmatrix} * & 0 & \cdots & \cdots & \cdots & \cdots & 0 \\ 1 & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & 2 & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & c(\mu) & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & \cdots & 0 & c(\mu) & * \end{bmatrix},$$

where 'c' is now used to denote the vector with the number of servers of each system, instead of 'C' as used for indepMMcqueue, to avoid confusion with the matrices with the same name that compose the TT operator.

L_d also changes

$$L_d = \begin{bmatrix} * & 0 & \cdots & \cdots & \cdots & \cdots & 0 \\ \mathrm{dep}(d) & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & 2\times\mathrm{dep}(d) & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & c(d)\times\mathrm{dep}(d) & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & \cdots & 0 & c(d)\times\mathrm{dep}(d) & * \end{bmatrix},$$

being also associated with departures, even if, in this case, local ones.

Concerning the parameters, they are: value for the influx rate, 'arr' in kanbanalt2; vector 'dep', associated with the departure rates as in the previous models; vector with the maximum capacities; vector 'c'.

In our implementation we have: 'arr' as 1.2; 'dep' as $[1,1,1,1,1,1]$; maximum capacities $[16,16,16,16,16,16]$; 'c' as $[3,3,3,3,3,3]$. We are thus, again, considering a problem of size 17^6.

kanbanalt2balking {queueing, functlocal, synch, scalable, chain}.

We now combine kanbanalt2 with the concept of balking, introduced in indepbalking.

The natural way to combine them is by making the arrivals to the first queue follow the law introduced in indepbalking, from [44, Ch. 2]. The only change to kanbanalt2 is thus in L_1

$$L_1 = \begin{bmatrix} * & \mathrm{arr} & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \mathrm{arr} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \frac{1}{2}\times\mathrm{arr} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \frac{1}{n_1-s+1}\times\mathrm{arr} \\ 0 & \cdots & \cdots & \cdots & \cdots & 0 & * \end{bmatrix},$$

where 's' is the parameter associated with the law for the arrival rates, from (3.1); and 'arr' is the maximal arrival rate, 'in' in (3.1).

The input parameters of the model are thus: value 'arr'; vector for the service rates, associated with 'dep' in the previous models; vector with the maximum capacities; value 's'.

In our implementation we have: 'arr' as 1.2; 'dep' as $[1,1,1,1,1,1]$; maximum capacities $[16,16,16,16,16,16]$; 's' is 3. We are thus, again, considering a problem of size 17^6.

kanbanalt2priority {queueing, distinguishable, synch, scalable}.

We now combine kanbanalt2 with the concept of queues with priorities, based on the type of priority introduced for indeppriority.

The TT operator is changed from the one for kanbanalt2 as the service of a

customer of type I has to be separated from the service of a customer of type II

$$A_1(i_1,j_1) = \begin{bmatrix} L_1(i_1,j_1) \\ \mathrm{dep}(1,1)B1_1(i_1,j_1) \\ \mathrm{dep}(1,2)B2_1(i_1,j_1) \\ E_1(i_1,j_1) \\ I_1(i_1,j_1) \end{bmatrix}^T, \quad A_d(i_d,j_d) = \begin{bmatrix} I_d(i_d,j_d) \\ C1_d(i_d,j_d) \\ C2_d(i_d,j_d) \\ F_d(i_d,j_d) \\ L_d(i_d,j_d) \end{bmatrix}$$

and

$$A_\mu(i_\mu,j_\mu) = \begin{bmatrix} I_\mu(i_\mu,j_\mu) & 0 & 0 & 0 & 0 \\ C1_\mu(i_\mu,j_\mu) & 0 & 0 & 0 & 0 \\ C2_\mu(i_\mu,j_\mu) & 0 & 0 & 0 & 0 \\ F_\mu(i_\mu,j_\mu) & 0 & 0 & 0 & 0 \\ 0 & \mathrm{dep}(\mu,1)B1_\mu(i_\mu,j_\mu) & \mathrm{dep}(\mu,2)B2_\mu(i_\mu,j_\mu) & E_\mu(i_\mu,j_\mu) & I_\mu(i_\mu,j_\mu) \end{bmatrix},$$

$\mu = 2,\ldots,d-1$, where $B1_\mu$ relates with the departure from a customer of the first type while $B2_\mu$ is associated with the departure of a customer of type II. $C1_\mu$ and $C2_\mu$ are associated with the corresponding arrival of a customer of the first type and second type, respectively.

L_d also changes, for the same reason that the global operator changes, as it is associated with services. 'dep' is a matrix containing the rates of departure, on which a particular row is associated with a particular queue while a fixed column is associated with a specific type of customer.

The arrival rates are, as in indeppriority, stored in a vector with two elements, one for each type of customer, and they are needed to define L_1.

E_μ (F_μ) is the matrix associated with the diagonal corrections for $\mathrm{dep}(\mu,1)B1_\mu$ and $\mathrm{dep}(\mu,2)B2_\mu$ ($C1_\mu$ and $C2_\mu$).

The ranks of the operator are directly affected by the fact that there are two types of customers, even if they are again, as in all previous models, independent of d.

Concerning the parameters, they are: vector with the arrival rates; matrix 'dep'; vector with the maximum capacities.

In our implementation we have: vector of arrival rates $[1.2, 1.2]$; 'dep' as $[1,1;1,1;1,1;1,1]$; maximum capacities $[10,10,10,10]$. We are thus considering a problem of size 66^4.

cyclequeue{queueing, reducible, productform, synch, scalable, chain}.

This model is taken from [33, Ch. 7]. The interactions are similar to those in kanbanalt2 but with the difference that now there is no last queue as a customer that is served in the previously called last queue simply proceeds to the previously called first. The definition of first and last gets then lost, as in overflowpersim.

This leads to a closed network in which the total number of customers will remain constant, given that there is no interaction with the exterior.

The TT operator is

$$A_1(i_1,j_1) = \begin{bmatrix} I_1(i_1,j_1) \\ \mathrm{dep}(1)B_1(i_1,j_1) \\ C_1(i_1,j_1) \\ \mathrm{dep}(1)E_1(i_1,j_1) \\ F_1(i_1,j_1) \end{bmatrix}^T, \quad A_d(i_d,j_d) = \begin{bmatrix} C_d(i_d,j_d) \\ F_d(i_d,j_d) \\ I_d(i_d,j_d) \\ \mathrm{dep}(d)B_d(i_d,j_d) \\ \mathrm{dep}(d)E_d(i_d,j_d) \end{bmatrix},$$

$$A_2(i_2,j_2) = \begin{bmatrix} I_2(i_2,j_2) & \mathrm{dep}(2)B_2(i_2,j_2) & \mathrm{dep}(2)E_2(i_2,j_2) & 0 & 0 & 0 \\ 0 & 0 & 0 & C_2(i_2,j_2) & 0 & 0 \\ 0 & 0 & 0 & 0 & I_2(i_2,j_2) & 0 \\ 0 & 0 & 0 & F_2(i_2,j_2) & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & I_2(i_2,j_2) \end{bmatrix},$$

for $A_{d-1}(i_{d-1}, j_{d-1})$ we have

$$\begin{bmatrix} \mathrm{dep}(d-1)B_{d-1}(i_{d-1}, j_{d-1}) & \mathrm{dep}(d-1)E_{d-1}(i_{d-1}, j_{d-1}) & 0 & 0 & 0 \\ 0 & 0 & C_{d-1}(i_{d-1}, j_{d-1}) & 0 & 0 \\ 0 & 0 & F_{d-1}(i_{d-1}, j_{d-1}) & 0 & 0 \\ 0 & 0 & I_{d-1}(i_{d-1}, j_{d-1}) & 0 & 0 \\ 0 & 0 & 0 & I_{d-1}(i_{d-1}, j_{d-1}) & 0 \\ 0 & 0 & 0 & 0 & I_{d-1}(i_{d-1}, j_{d-1}) \end{bmatrix},$$

and

$$A_\mu(i_\mu, j_\mu) = \begin{bmatrix} I_\mu(i_\mu, j_\mu) & \mathrm{dep}(\mu)B_\mu(i_\mu, j_\mu) & \mathrm{dep}(\mu)E_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ 0 & 0 & 0 & C_\mu(i_\mu, j_\mu) & 0 & 0 \\ 0 & 0 & 0 & F_\mu(i_\mu, j_\mu) & 0 & 0 \\ 0 & 0 & 0 & I_\mu(i_\mu, j_\mu) & 0 & 0 \\ 0 & 0 & 0 & 0 & I_\mu(i_\mu, j_\mu) & 0 \\ 0 & 0 & 0 & 0 & 0 & I_\mu(i_\mu, j_\mu) \end{bmatrix},$$

$\mu = 3, \ldots, d-2$. The matrices inside the operator are as in kanbanalt2, but now with B_μ and C_μ being defined for all dimensions as the interaction between last and first dimensions requires the introduction of B_d and C_1, while the local part disappears. In its turn, 'dep' is again the vector with the service (departure) rates while there are no arrival rates in this closed network.

Concerning the parameters, they are: vector 'dep'; vector with the maximum capacities.

In our implementation we have: 'dep' as $[1, 1, 1, 1, 1, 1]$; maximum capacities $[16, 16, 16, 16, 16, 16]$; We are thus, again, considering a problem of size 17^6.

compfailure{qualitycontrol, synch, scalable, chain}.

This model is an adapted version of the model in [14, Ch. 2], which appears in this reference for doing experiments on different techniques that make use of the Kronecker structure of a SAN for finding its steady-state.

Each machine (system) may either be working or not, implying that there are two possible states per system. Each machine can stop working with a certain, as usual, exponential rate; and can be repaired, when not working, with another exponential rate. These are the local transitions. If all machines are not working, there is an extra possibility, synchronized transition, that all machines are repaired at the same time, again with an exponential rate.

As the states reflect the number of deficiencies, it makes sense to associate 0 with a working machine and 1 with a not working machine.

The TT operator is

$$A_1(i_1, j_1) = \begin{bmatrix} L_1(i_1, j_1) \\ \mathrm{repgl} \times B_1(i_1, j_1) \\ \mathrm{repgl} \times E_1(i_1, j_1) \\ I_1(i_1, j_1) \end{bmatrix}^T, \quad A_d(i_d, j_d) = \begin{bmatrix} I_d(i_d, j_d) \\ B_d(i_d, j_d) \\ E_d(i_d, j_d) \\ L_d(i_d, j_d) \end{bmatrix},$$

and

$$A_\mu(i_\mu, j_\mu) = \begin{bmatrix} I_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ 0 & B_\mu(i_\mu, j_\mu) & 0 & 0 \\ 0 & 0 & E_\mu(i_\mu, j_\mu) & 0 \\ L_\mu(i_\mu, j_\mu) & 0 & 0 & I_\mu(i_\mu, j_\mu) \end{bmatrix},$$

for $\mu = 2, \ldots, d-1$, where 'repgl' is the rate of repairing all systems at the same time.

The local matrices are

$$L_\mu = \begin{bmatrix} -\text{fail}(\mu) & \text{fail}(\mu) \\ \text{rep}(\mu) & -\text{rep}(\mu) \end{bmatrix},$$

while the matrices of the interaction are

$$B_\mu = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix},$$

where 'rep' is a vector with the repairing rates for the different systems while 'fail' is a vector with the failing rates.

E_μ has the diagonal corrections associated with B_μ.

Concerning the parameters, they are: vector 'fail'; value 'repgl'; vector 'rep'.

In our implementation the default parameters are: 'fail' as a vector of size 24 with all entries equal to 0.5; 'repgl' is 0.1; 'rep' as a vector of size 24 with all entries equal to 1. We are thus considering a problem of size 2^{24}.

compfailuregen{qualitycontrol, synch, scalable}.

This model is a generalization of compfailure. Each machine has now a general number of deficiencies, not being just about working or not. We assume that the global repair, which repairs all machines at the same time, occurs when the number of deficiencies is 4, 9, 14 or 19, for all machines. It thus only makes sense to consider a maximum possible number of deficiencies of at least 19. After being repaired, the number of deficiencies updates to 0, 5, 10 and 15, respectively. We thus have four synchronized transitions.

The operator is now

$$A_1(i_1, j_1) = \begin{bmatrix} L_1(i_1, j_1) \\ \text{repgl} \times B1_1(i_1, j_1) \\ \text{repgl} \times B2_1(i_1, j_1) \\ \text{repgl} \times B3_1(i_1, j_1) \\ \text{repgl} \times B4_1(i_1, j_1) \\ \text{repgl} \times E1_1(i_1, j_1) \\ \text{repgl} \times E2_1(i_1, j_1) \\ \text{repgl} \times E3_1(i_1, j_1) \\ \text{repgl} \times E4_1(i_1, j_1) \\ I_1(i_1, j_1) \end{bmatrix}^T, \quad A_d(i_d, j_d) = \begin{bmatrix} I_d(i_d, j_d) \\ B1_d(i_d, j_d) \\ B2_d(i_d, j_d) \\ B3_d(i_d, j_d) \\ B4_d(i_d, j_d) \\ E1_d(i_d, j_d) \\ E2_d(i_d, j_d) \\ E3_d(i_d, j_d) \\ E4_d(i_d, j_d) \\ L_d(i_d, j_d) \end{bmatrix},$$

and for $A_\mu(i_\mu, j_\mu)$ we have

$$\begin{bmatrix} I_\mu(i_\mu, j_\mu) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & B1_\mu(i_\mu, j_\mu) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & B2_\mu(i_\mu, j_\mu) & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & B3_\mu(i_\mu, j_\mu) & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & B4_\mu(i_\mu, j_\mu) & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & E1_\mu(i_\mu, j_\mu) & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & E2_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & E3_\mu(i_\mu, j_\mu) & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & E4_\mu(i_\mu, j_\mu) & 0 \\ L_\mu(i_\mu, j_\mu) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & I_\mu(i_\mu, j_\mu) \end{bmatrix},$$

$\mu = 2, \ldots, d-1$.

In the operator above, $B1_\mu$, $B2_\mu$, $B3_\mu$ and $B4_\mu$ represent the synchronizations mentioned before, respectively

$$B1_\mu = B2_\mu = B3_\mu = B4_\mu = \begin{bmatrix} 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\ 1 & 0 & \cdots & \cdots & \cdots & \cdots & 0 \\ 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \end{bmatrix},$$

where the 1 is in the row 5, 10, 15 and 20, respectively.

$E1_\mu$, $E2_\mu$, $E3_\mu$ and $E4_\mu$ stand for the respective diagonal corrections.

'repgl' denotes again the repairing rate associated with the four different synchronized transitions, thus assumed to be, without loss of generality, common to the four interactions.

For the local part, we have

$$L_\mu = \begin{bmatrix} * & \text{fail}(\mu) & 0 & \cdots & 0 \\ \text{rep}(\mu) & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \text{fail}(\mu) \\ 0 & \cdots & 0 & \text{rep}(\mu) & * \end{bmatrix},$$

where again 'fail' is a vector with the rates of failure of the parts, where one assumes that only one specific part can fail at each point in time or the rates would be linear on the number of working parts, as explained before; while 'rep' is a vector with the repairing rates.

All entries of the vector of ranks increase by one per synchronized event, showing how strong an interaction becomes in this context. This is quite severe, and due to the fact that all dimensions are involved in the interaction. The ranks are, however, still independent of d.

Concerning the parameters, they are: vector 'fail'; value 'repgl'; vector 'rep'; vector with the maximum number of deficiencies per system $(n-1)$.

In our implementation the default parameters are: 'fail' as $[0.5, 0.5, 0.5, 0.5, 0.5]$; 'repgl' is 0.1; 'rep' as $[1, 1, 1, 1, 1]$; maximum number of deficiencies $[32, 32, 32, 32, 32]$. We are thus, again, considering a problem of size 33^5.

compfailureorig {qualitycontrol, synch, scalable}.

After two models associated with the one in [14, Ch. 2], we now have the one that is the closest to the original. It is again a generalization of compfailure.

The only synchronized event happens when all parts of all machines have failed. In that case, the repairing occurs on all of them, with all parts working again afterwards.

The operator is the same as in compfailure but the matrices that compose it are different.

For L_μ we have

$$L_\mu = \begin{bmatrix} * & \text{fail}(\mu) & 0 & \cdots & 0 \\ \text{rep}(\mu) & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \text{fail}(\mu) \\ 0 & \cdots & 0 & \text{rep}(\mu) & * \end{bmatrix},$$

where 'fail' and 'rep' are as in the two models above.

As for B_μ,

$$B_\mu = \begin{bmatrix} 0 & \cdots & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & 0 \\ 1 & 0 & \cdots & 0 \end{bmatrix}.$$

Concerning the parameters, they are the same as in compfailuregen - vector 'fail'; value 'repgl'; vector 'rep'; vector with the maximum number of deficiencies per system.

In our implementation the default parameters are, as in the previous model: 'fail' as $[0.5, 0.5, 0.5, 0.5, 0.5, 0.5]$; 'repgl' is 0.1; 'rep' as $[1, 1, 1, 1, 1, 1]$; maximum number of deficiencies $[16, 16, 16, 16, 16, 16]$. We are thus, again, considering a problem of size 17^6.

multiprogramming {communications, reducible, prodform, funct, synch, chain}.

The next model is associated with a time-sharing multiprogramming virtual memory system [9]. It can be found in the numerical experiments of [42] also. A similar version of the model also appears in [33, Ch. 7]. These first and third references concern finding the steady-state in more analytical terms while the second is again an overview of computational techniques.

The network consists of a set of terminals from which a set of users, who are as many as the terminals, generate commands; a CPU; a secondary memory device (SM); and a filing disk (FD). A queue of requests is associated with each device. Once a command is generated, the user at the terminal will remain inactive until the system responds. Symbolically, a user having generated a command enters the CPU queue.

This is a closed network in which the total number of users is denoted by N. The degree of multiprogramming, n, is the sum of the processes in CPU, SM and FD, meaning that the number of processes in the terminal is $N - n$. One can thus extract the degree of multiprogramming from the number of users in the terminals.

Such measure is important as some of the rates are a function of it. In particular, the rate of transition from 'CPU' to 'SM' is modelled in both [42] and [9] as

$$\alpha(\frac{M}{\eta})^k,$$

where α, M and k are constants of the model and η is the mentioned degree of multiprogramming.

The TT operator for this network with 4 dimensions, given the natural ordering of the dimensions that arises from the interactions - 'Terminals', 'CPU', 'SM', 'FD' -

is

$$A_1(i_1,j_1) = \begin{bmatrix} I_1(i_1,j_1) \\ \lambda B_1(i_1,j_1) \\ \lambda E_1(i_1,j_1) \\ C_1(i_1,j_1) \\ F_1(i_1,j_1) \\ \alpha M^k D_1(i_1,j_1) \end{bmatrix}^T, \quad A_4(i_4,j_4) = \begin{bmatrix} I_4(i_4,j_4) \\ \text{mu}(2)B_4(i_4,j_4) \\ \text{mu}(2)E_4(i_4,j_4) \\ C_4(i_4,j_4) \\ F_4(i_4,j_4) \end{bmatrix},$$

$$A_2(i_2,j_2) = \begin{bmatrix} 0 & 0 & 0 & rB_2(i_2,j_2) & rE_2(i_2,j_2) & C_2(i_2,j_2) & F_2(i_2,j_2) \\ C_2(i_2,j_2) & 0 & 0 & 0 & 0 & 0 & 0 \\ F_2(i_2,j_2) & 0 & 0 & 0 & 0 & 0 & 0 \\ cB_2(i_2,j_2) & 0 & 0 & 0 & 0 & 0 & 0 \\ cE_2(i_2,j_2) & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & B_2(i_2,j_2) & E_2(i_2,j_2) & 0 & 0 & 0 & 0 \end{bmatrix},$$

and

$$A_3(i_3,j_3) = \begin{bmatrix} I_3(i_3,j_3) & 0 & 0 & 0 & 0 \\ C_3(i_3,j_3) & 0 & 0 & 0 & 0 \\ F_3(i_3,j_3) & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & I_3(i_3,j_3) & 0 \\ 0 & 0 & 0 & 0 & I_3(i_3,j_3) \\ \text{mu}(1)B_3(i_3,j_3) & I_3(i_3,j_3) & 0 & 0 & 0 \\ \text{mu}(1)E_3(i_3,j_3) & 0 & I_3(i_3,j_3) & 0 & 0 \end{bmatrix}.$$

λ , r, c and vector 'mu' are parameters of the model associated with the rates of transition between systems. λ is the rate of leaving 'Terminals' to 'CPU'; r is the rate of transition from 'CPU' to 'FD'; c is the rate from 'CPU' to 'Terminals'; vector 'mu' has two entries that are associated with leaving 'SM' and 'FD' to 'Terminals'.

In coherence with the usual notation, matrices B_μ are associated with the loss of a user, while then C_μ is associated with the corresponding arrival in a different system. One adds the notation D_1 for a matrix associated with an interaction that is both functional and synchronized, but where this matrix is associated only with the functional part as the rates depend on the state of this system but this is not the system from which the user leaves, which is different from all previous similar interactions as the state from which the user leaves was always the one on which the rates depended.

In particular, we have

$$B_1 = \begin{bmatrix} 0 & \cdots & \cdots & \cdots & 0 \\ 1 & \ddots & \ddots & \ddots & \vdots \\ 0 & 2 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & (n_1-1) & 0 \end{bmatrix}, \quad B_\mu \begin{bmatrix} 0 & \cdots & \cdots & \cdots & 0 \\ 1 & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & 0 \end{bmatrix},$$

for $\mu = 2, \ldots, 4$, while

$$D_1 = \begin{bmatrix} (n_1-1)^{-k} & 0 & \cdots & 0 \\ 0 & (n_1-2)^{-k} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 1 \end{bmatrix}.$$

E_μ (F_μ) are then the corresponding diagonal corrections of B_μ (C_μ).

Concerning the parameters, they are: N; λ; c; r; M; α; k; vector 'mu'.

In [42], some of the values are fixed throughout the experiments. We have $\alpha = 0.01$, $M = 128$, $k = 1.5$, $r = 500\times10^{-3}$, $c = 20\times10^{-3}$, vector 'mu' is $[\frac{1}{5\times10^{-3}}, \frac{1}{30\times10^{-3}}]$, $\lambda = \frac{1}{10}$. For N, they consider 20 and 50. In [9], parameters are discussed in more detail. They note that k is supposed to be between 1.5 and 2.5 according to [5], while close to 1 according to [46]. Later, in the numerical experiments, they also fix some of the parameters. In particular, they use $\lambda = \frac{1}{10}$, $c = 1$, $r = 50 \times 10^{-3}$, $\alpha = 0.01$. The remaining ones are tested with different values. They try 128 and 256 for M; all integers between 1 and 30 for N; for k they end up trying 1.5, 2 and 2.5; for 'mu', they use $[\frac{1}{5\times10^{-3}}, \frac{1}{30\times10^{-3}}]$, $[\frac{1}{10\times10^{-3}}, \frac{1}{30\times10^{-3}}]$ and $[\frac{1}{5\times10^{-3}}, \frac{1}{60\times10^{-3}}]$, after also mentioning the pair $[\frac{1}{10\times10^{-3}}, \frac{1}{50\times10^{-3}}]$ before the experiments. We note that the range of the parameters is similar for the two references. In this sequence, we use in our implementation, as default values, some of the parameters above: $N = 50$, $\lambda = \frac{1}{10}$, $c = 20 \times 10^{-3}$, $r = 500 \times 10^{-3}$, $M = 128$, $\alpha = 0.01$, $k = 1.5$, vector 'mu' is $[\frac{1}{5\times10^{-3}}, \frac{1}{30\times10^{-3}}]$. We are thus considering a problem of size 51^4.

impcustomers{queueing, funct, synch, chain}.

The next model concerns the effect of impatient customers on a computerized telephone exchange [8], being included in the numerical experiments of [42].

A customer requests a service. He will wait a certain time until getting a response and if he does not start being served in that period, either he leaves or he waits for some time before joining the queue again. We thus have two queues. One, associated with the second dimension of the TT operator, represents the number of customers that are waiting for the service, associated with the special processing task that is required by the customers. After giving up on waiting, the customer may join another station, a queue, associated with the first dimension of the operator, where he will stay for a certain period, going then again back to the main station to try again; or he simply leaves without going through this 'stand-by' queue.

The customers in the first system, 'stand-by' queue, are said to be in their 'thinking time'. The ones in the second one are directly waiting for the processing task that they are requesting.

The TT representation of the transition rate matrix is now

$$A_1(i_1, j_1) = \begin{bmatrix} I_1(i_1, j_1) & \lambda B_1(i_1, j_1) & \lambda E_1(i_1, j_1) & C_1(i_1, j_1) & F_1(i_1, j_1) \end{bmatrix},$$
$$A_2(i_2, j_2) = \begin{bmatrix} L_2(i_2, j_2) & C_2(i_2, j_2) & F_2(i_2, j_2) & \tau h B_2(i_2, j_2) & \tau h E_2(i_2, j_2) \end{bmatrix}^T,$$

where h, τ and λ are parameters of the model - h is the probability of moving to the first system given that the customer leaves the second system as he is tired of waiting ($1 - h$ is thus the conditional probability that he leaves the network instead), which happens with rate τ; λ is the rate for each customer to finish the 'thinking time', going then back to the main system.

One has

$$B_1 = \begin{bmatrix} 0 & \cdots & \cdots & \cdots & 0 \\ 1 & \ddots & \ddots & \ddots & \vdots \\ 0 & 2 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & (n_1 - 1) & 0 \end{bmatrix}, \quad B_2 \begin{bmatrix} 0 & \cdots & \cdots & \cdots & 0 \\ 1 & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & 0 \end{bmatrix},$$

for the matrices of the interactions; while for the local part

$$L_2 = \begin{bmatrix} * & \text{arr} & 0 & \cdots & 0 \\ \tau(1-h)+\text{mu} & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \text{arr} \\ 0 & \cdots & 0 & \tau(1-h)+\text{mu} & * \end{bmatrix},$$

where 'mu' is the rate of service while 'arr' is the arrival rate of customers. The total rate of departure is $\tau(1-h)+\text{mu}$ as customers may also leave the system, without having been served, with rate $\tau(1-h)$.

There is only local part in the second dimension as the first system does not interact with the exterior.

E_μ and F_μ are, as usual, the diagonal corrections for the terms associated with B_μ and C_μ, respectively.

The parameters are: value 'arr'; value 'mu'; τ; h; λ; vector with the maximum capacities for the two dimensions.

The parameters that are used in [42], and argued to be realistic by the authors, are: 0.6 for 'arr'; 1 for 'mu'; 0.05 for τ; 0.85 for h; 5 for λ; $[10, 220]$ for the vector of maximum capacities. They note that the mode sizes should be chosen big enough in order for the probability of saturation to be below 1×10^{-10} as the idea is to simulate the case where the capacities are infinite, which is the real situation. We use, as default values, the mentioned parameters, except for the maximum capacities, changed to $[1024, 1024]$. The problem size is thus 1025^2.

handoff1class1d {communications, synch, scalable, chain}.

The next models, mainly based on [50, 3], while approximations are obtained for analysing such networks in [4], consider a group of base stations covering some geographical area where each base station, with which mobile users communicate, is referred to as a cell. A base station is responsible for the bandwidth management concerning mobile phones in its cell. The bandwidth that is being used is what characterizes the state of a cell, while the dimensions of the tensor represent the different cells. New calls are initiated in the different cells and they may be transferred to another cell in case the phones move to the geographical area of that other cell.

The general model in the references consider distinguishable customers who can: move to a different cell; finish the call in a particular cell with a certain rate associated with the expected duration of the call; or simply leave the network by moving to a geographical region that is not covered by any cell of the network. The possibility of moving between cells introduces the concept of 'handoff'. Customers stay in the same cell for an exponentially distributed period. The rate of having a new call is the same for all cells, but can be different among different types of customers. Concerning the required bandwidth per call, one assumes the simplifying hypothesis that it is one unit for all types of customers.

We first consider a model with only one type of customer and where one assumes a topology in one dimension for the geographical area. We assume, without loss of generality, that the probability of moving in any direction is the same, meaning that the probability of leaving to each neighbour is simply $\frac{1}{2}$.

The corresponding TT operator is

$$A_1(i_1, j_1) = \begin{bmatrix} \frac{1}{2}\gamma B_1(i_1, j_1) & \frac{1}{2}\gamma E_1(i_1, j_1) & C_1(i_1, j_1) & F_1(i_1, j_1) & I_1(i_1, j_1) \end{bmatrix},$$

$$A_d(i_d, j_d) = \begin{bmatrix} I_d(i_d, j_d) & C_d(i_d, j_d) & F_d(i_d, j_d) & \frac{1}{2}\gamma B_d(i_d, j_d) & \frac{1}{2}\gamma E_d(i_d, j_d) \end{bmatrix}^T,$$

$$A_2(i_2, j_2) = \begin{bmatrix} \frac{\text{mu}+0.5\gamma}{0.5\gamma} I_2(i_2, j_2) + C_2(i_2, j_2) & 0 & 0 & 0 & 0 & 0 \\ -\frac{\text{mu}+0.5\gamma}{0.5\gamma} I_2(i_2, j_2) + F_2(i_2, j_2) & 0 & 0 & 0 & 0 & 0 \\ \text{mu} \times I_2(i_2, j_2) + \frac{1}{2}\gamma B_2(i_2, j_2) & 0 & 0 & 0 & 0 & 0 \\ \text{arr} \times I_2(i_2, j_2) + \frac{1}{2}\gamma E_2(i_2, j_2) & 0 & 0 & 0 & 0 & 0 \\ L_2(i_2, j_2) & \frac{1}{2}\gamma B_2(i_2, j_2) & \frac{1}{2}\gamma E_2(i_2, j_2) & C_2(i_2, j_2) & F_2(i_2, j_2) & I_2(i_2, j_2) \end{bmatrix},$$

for $A_{d-1}(i_{d-1}, j_{d-1})$ we get

$$\begin{bmatrix} I_{d-1}(i_{d-1}, j_{d-1}) & C_{d-1}(i_{d-1}, j_{d-1}) & F_{d-1}(i_{d-1}, j_{d-1}) & \frac{1}{2}\gamma B_{d-1}(i_{d-1}, j_{d-1}) & \frac{1}{2}\gamma E_{d-1}(i_{d-1}, j_{d-1}) & L_{d-1}(i_{d-1}, j_{d-1}) \\ 0 & 0 & 0 & 0 & 0 & \text{arr} I_{d-1}(i_{d-1}, j_{d-1}) + \frac{1}{2}\gamma B_{d-1}(i_{d-1}, j_{d-1}) \\ 0 & 0 & 0 & 0 & 0 & \text{arr} I_{d-1}(i_{d-1}, j_{d-1}) + \frac{1}{2}\gamma E_{d-1}(i_{d-1}, j_{d-1}) \\ 0 & 0 & 0 & 0 & 0 & \frac{\text{mu}+0.5\gamma}{0.5\gamma} I_{d-1}(i_{d-1}, j_{d-1}) + C_{d-1}(i_{d-1}, j_{d-1}) \\ 0 & 0 & 0 & 0 & 0 & \frac{\text{mu}+0.5\gamma}{0.5\gamma} I_{d-1}(i_{d-1}, j_{d-1}) + F_{d-1}(i_{d-1}, j_{d-1}) \end{bmatrix}^T,$$

and

$$A_\mu(i_\mu, j_\mu) = \begin{bmatrix} I_\mu(i_\mu, j_\mu) & 0 & 0 & 0 & 0 & 0 \\ C_\mu(i_\mu, j_\mu) & 0 & 0 & 0 & 0 & 0 \\ F_\mu(i_\mu, j_\mu) & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{2}\gamma B_\mu(i_\mu, j_\mu) & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{2}\gamma E_\mu(i_\mu, j_\mu) & 0 & 0 & 0 & 0 & 0 \\ L_\mu(i_\mu, j_\mu) & \frac{1}{2}\gamma B_\mu(i_\mu, j_\mu) & \frac{1}{2}\gamma E_\mu(i_\mu, j_\mu) & C_\mu(i_\mu, j_\mu) & F_\mu(i_\mu, j_\mu) & I_\mu(i_\mu, j_\mu) \end{bmatrix},$$

$\mu = 3, \ldots, d-2$. γ is the rate of leaving a cell, associated with 'handoff' effect, either to another cell or out of the geographical area covered by the cells, in which case the transition is local; 'mu' is the rate associated with the end of a call and 'arr' is the rate of having a new call, for any cell.

The local part is

$$L_1 = L_d = \begin{bmatrix} * & \text{arr} & 0 & \cdots & 0 \\ \text{dep} + \frac{1}{2}\gamma & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \text{arr} \\ 0 & \cdots & 0 & \text{dep} + \frac{1}{2}\gamma & * \end{bmatrix}, \quad L_\mu \begin{bmatrix} * & \text{arr} & 0 & \cdots & 0 \\ \text{dep} & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \text{arr} \\ 0 & \cdots & 0 & \text{dep} & * \end{bmatrix},$$

$\mu = 2, \ldots, d-1$. L_1 and L_d are not part of the final operator as one can write them as linear combinations of the remaining matrices of the corresponding cores, thus reducing the associated ranks, which is a technique that appeared first in [31].

The matrices of the local part are different in the first and last dimensions as they have the particularity of having the exterior of the network as immediate neighbour, implying that, with probability $\frac{1}{2}$, a customer that leaves a cell goes out of the network.

As for the matrices of the interactions, they are the usual ones

$$B_\mu = \begin{bmatrix} 0 & \cdots & \cdots & \cdots & 0 \\ 1 & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & 0 \end{bmatrix}, \quad C_\mu = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & 1 \\ 0 & \cdots & \cdots & \cdots & 0 \end{bmatrix},$$

while E_μ and F_μ are the corresponding diagonal corrections, respectively.

The parameters are: value 'arr'; value 'dep'; γ; vector of maximum capacities of the different cells.

In our implementation the default parameters are: 'arr' is 1.5; 'dep' is 1; $\gamma = 0.1$; maximum capacities $[8, 8, 8, 8, 8, 8, 8]$. We are thus considering a problem of size 9^7.

handoff1class2d {communications, synch, chain}.

The next model generalizes the previous one to a two-dimensional geography. As in [50, 3], cells with shape of hexagons are considered, meaning that each cell will have six neighbours now instead of two.

We omit the representation of the operator as it requires too much space even for the smallest d that makes sense to consider, $d = 7$, which is the value that we use in our implementation, as the ranks of the TT operator are significantly larger than for all previous models, which is coherent with the lack of suitability of the topology to the format.

The matrices associated with the local part are

$$L_\mu = \begin{bmatrix} * & \text{arr} & 0 & \cdots & 0 \\ \text{dep} + \frac{1}{2}\gamma & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \text{arr} \\ 0 & \cdots & 0 & \text{dep} + \frac{1}{2}\gamma & * \end{bmatrix} \text{ or } L_\mu = \begin{bmatrix} * & \text{arr} & 0 & \cdots & 0 \\ \text{dep} & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \text{arr} \\ 0 & \cdots & 0 & \text{dep} & * \end{bmatrix},$$

depending on if the dimension is associated with a cell that has contact with the exterior, first case; or not, second case.

Matrices associated with the interactions, B_μ and C_μ (and thus also the diagonal corrections E_μ and F_μ), are the same as in the previous model.

The parameters are, as in the previous model: value 'arr'; value 'dep'; γ; vector of maximum capacities of the different cells.

In our implementation the default parameters are, also as in the previous model: 'arr' is 1.5; 'dep' is 1; $\gamma = 0.1$; maximum capacities $[8, 8, 8, 8, 8, 8, 8]$. We are thus considering a problem of size 9^7.

handoff2class1d {communications, distinguishable, synch, scalable}.

The next model consists of generalizing handoff1class1d to allow, instead of a two-dimensional geography as in the last model, having two types of customers.

The operator is given by

$$A_1(i_1, j_1) = \begin{bmatrix} L_1(i_1, j_1) \\ \frac{1}{2}\gamma(1)B1_1(i_1, j_1) \\ \frac{1}{2}\gamma(2)B2_1(i_1, j_1) \\ E_1(i_1, j_1) \\ C1_1(i_1, j_1) \\ C2_1(i_1, j_1) \\ F_1(i_1, j_1) \\ I_1(i_1, j_1) \end{bmatrix}^T, \quad A_d(i_d, j_d) = \begin{bmatrix} I_d(i_d, j_d) \\ C1_d(i_d, j_d) \\ C2_d(i_d, j_d) \\ F_d(i_d, j_d) \\ \frac{1}{2}\gamma(1)B1_d(i_d, j_d) \\ \frac{1}{2}\gamma(2)B2_d(i_d, j_d) \\ E_d(i_d, j_d) \\ L_d(i_d, j_d) \end{bmatrix},$$

and for $A_\mu(i_\mu, j_\mu)$, $\mu = 2, \ldots, d-1$, we have

$$\begin{bmatrix} I_\mu(i_\mu,j_\mu) & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ C1_\mu(i_\mu,j_\mu) & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ C2_\mu(i_\mu,j_\mu) & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ F_\mu(i_\mu,j_\mu) & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{2}\gamma(1)B1_\mu(i_\mu,j_\mu) & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{2}\gamma(2)B2_\mu(i_\mu,j_\mu) & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ E_\mu(i_\mu,j_\mu) & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ L_\mu(i_\mu,j_\mu) & \frac{1}{2}\gamma(1)B1_\mu(i_\mu,j_\mu) & \frac{1}{2}\gamma(2)B2_\mu(i_\mu,j_\mu) & E_\mu(i_\mu,j_\mu) & C1_\mu(i_\mu,j_\mu) & C2_\mu(i_\mu,j_\mu) & F_\mu(i_\mu,j_\mu) & I_\mu(i_\mu,j_\mu) \end{bmatrix},$$

where *gamma* is now a vector with two entries containing the rates associated with 'handoff' for the two types of customers.

All matrices that compose the operator are defined similarly to handoff1class1d with the difference that one needs to distinguish the calls according to the involved type of customer, which results in the matrices $B1_\mu$ and $B2_\mu$ to replace B_μ; and $C1_\mu$ and $C2_\mu$ to replace C_μ. They are omitted for the same reason as in kanban.

Similarly to kanbanalt2priority, E_μ (F_μ) are now associated with the diagonal corrections for $\frac{1}{2}\gamma(1)B1_\mu$ and $\frac{1}{2}\gamma(2)B2_\mu$ ($C1_\mu$ and $C2_\mu$).

The parameters are the same as in the previous two models except that 'arr', 'dep' and γ are now vectors with two entries. Then there is again the vector with the maximum capacities.

We consider the default parameters: 'arr' is $[1.5, 1.5]$; 'dep' is $[1, 1]$; $\gamma = [0.1, 0.1]$; maximum capacities $[3, 3, 3, 3, 3, 3, 3]$. We are thus considering a problem of size 10^7.

handoff{communications, distinguishable, synch}.

We now combine the previous two models to get a more general version. We consider both two types of customers and the two-dimensional geometry. The main references are again [50, 3, 4].

As in handoff1class2d, the operator is omitted, for the same reason. It involves the same matrices that compose the operator for the previous model.

The parameters are the same as in the previous model: vectors of two entries 'arr', 'dep' and γ; vector with the maximum capacities.

We consider the default parameters, as in the previous model: 'arr' is $[1.5, 1.5]$; 'dep' is $[1, 1]$; $\gamma = [0.1, 0.1]$; maximum capacities $[3, 3, 3, 3, 3, 3, 3]$. We are thus considering a problem of size 10^7.

directedmetab{chemical, productform, funct, synch, scalable, chain}.

The next models are extracted from [36]. The general idea of such models is that the different states are associated with the distribution of molecule numbers of metabolites, for different types of topologies of the links between the nodes. Each dimension of the operator represents a node.

The first model that we consider consists of an incoming flux of substrate molecules being converted, through a series of enzymatic reactions, into a product flux [37, Ch. 3]. Each reaction takes as precursor the product of the preceding reaction.

The incoming flux, associated with the first node of the pathway, is constant. As for the set of reactions that convert a substrate of a given type in one of the next type, the rates are a function of the number of particles in the first. The rate, for substrate i, is

$$\frac{v_i m_i}{m_i + K_i - 1}, \tag{3.2}$$

where v_i and K_i are constants, while m_i is the number of particles in substrate i.

The TT operator is very similar to the one from kanbanalt2 even if they model totally different applications, as the topologies are very similar. The operator is

$$A_1(i_1,j_1) = \begin{bmatrix} L_1(i_1,j_1) & B_1(i_1,j_1) & E_1(i_1,j_1) & I_1(i_1,j_1) \end{bmatrix}, \quad A_d(i_d,j_d) = \begin{bmatrix} I_d(i_d,j_d) \\ C_d(i_d,j_d) \\ F_d(i_d,j_d) \\ L_d(i_d,j_d) \end{bmatrix},$$

and

$$A_\mu(i_\mu,j_\mu) = \begin{bmatrix} I_\mu(i_\mu,j_\mu) & 0 & 0 & 0 \\ C_\mu(i_\mu,j_\mu) & 0 & 0 & 0 \\ F_\mu(i_\mu,j_\mu) & 0 & 0 & 0 \\ 0 & B_\mu(i_\mu,j_\mu) & E_\mu(i_\mu,j_\mu) & I_\mu(i_\mu,j_\mu) \end{bmatrix}, \quad \mu = 2,\ldots,d-1.$$

The local part of the operator above is

$$L_1 = \begin{bmatrix} * & c & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & c \\ 0 & \cdots & \cdots & 0 & * \end{bmatrix}, \quad L_d = \begin{bmatrix} * & 0 & \cdots & \cdots & 0 \\ \frac{v_d}{K_d} & \ddots & \ddots & \ddots & \vdots \\ 0 & \frac{2v_d}{K_d+1} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \frac{(n_d-1)v_d}{n_d+K_d-2} & * \end{bmatrix},$$

where 'c' denotes the influx constant rate.

For the interaction matrices,

$$B_\mu = \begin{bmatrix} 0 & \cdots & \cdots & \cdots & 0 \\ \frac{v_\mu}{K_\mu} & \ddots & \ddots & \ddots & \vdots \\ 0 & \frac{2v_\mu}{K_\mu+1} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & \frac{(n_\mu-1)v_\mu}{n_\mu+K_\mu-2} & 0 \end{bmatrix}, \quad C_\mu = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & 1 \\ 0 & \cdots & \cdots & \cdots & 0 \end{bmatrix}.$$

The local part is associated with the production of particles, in the first node; and their degeneration, in the last node. B_μ represents the loss of a particle, which is associated with a new particle in the next node. This next node is the one associated with C_μ.

The diagonal corrections associated with B_μ and C_μ are in E_μ and F_μ, respectively.

The parameters are: vector 'v'; vector 'K'; influx rate 'c'; vector with the maximum capacity of each system.

Some restrictions are considered in [36]. They state that entries of 'K' should take values between 10^3 and 10^4; entries in 'v' should be between c and $10c$. They mention that $d = 10$ is a realistic value. As for the experiments that are done, v_1 is tested as $1.1c$ and $5c$, for an example with $d = 6$. In our implementation, we use the default parameters: 'v' is $[0.1, 0.1, 0.1, 0.1, 0.1, 0.1]$; 'K' is $[1000, 1000, 1000, 1000, 1000, 1000]$; $c = 0.01$; maximum capacities $[16, 16, 16, 16, 16, 16]$. The problem size is thus 17^6.

reversiblemetab{chemical, funct, synch, scalable, chain}.

The next model is again taken from [36].

The difference to the previous model is that the particles can also be converted in the opposite direction now - from one node to the previous.

The rates of the interactions again follow (3.2) but now each substrate has two associated values for v_i and K_i, one for the forward and one for the backward transformation. In this context, we split 'v' into 'vfor' and 'vback', where the first will contain the 'v' associated with the forward transformations while the second will contain the ones for the backward transformations. 'Kfor' and 'Kback' are similarly associated with 'K'.

The operator is now

$$A_1(i_1,j_1) = \begin{bmatrix} L_1(i_1,j_1) \\ B1_1(i_1,j_1) \\ E1_1(i_1,j_1) \\ C_1(i_1,j_1) \\ F_1(i_1,j_1) \\ I_1(i_1,j_1) \end{bmatrix}^T, \quad A_d(i_d,j_d) = \begin{bmatrix} I_d(i_d,j_d) \\ C_d(i_d,j_d) \\ F_d(i_d,j_d) \\ B2_d(i_d,j_d) \\ E2_d(i_d,j_d) \\ L_d(i_d,j_d) \end{bmatrix}$$

and

$$A_\mu(i_\mu,j_\mu) = \begin{bmatrix} I_\mu(i_\mu,j_\mu) & 0 & 0 & 0 & 0 & 0 \\ C_\mu(i_\mu,j_\mu) & 0 & 0 & 0 & 0 & 0 \\ F_\mu(i_\mu,j_\mu) & 0 & 0 & 0 & 0 & 0 \\ B2_\mu(i_\mu,j_\mu) & 0 & 0 & 0 & 0 & 0 \\ E2_\mu(i_\mu,j_\mu) & 0 & 0 & 0 & 0 & 0 \\ 0 & B1_\mu(i_\mu,j_\mu) & E1_\mu(i_\mu,j_\mu) & C_\mu(i_\mu,j_\mu) & F_\mu(i_\mu,j_\mu) & I_\mu(i_\mu,j_\mu) \end{bmatrix},$$

$\mu = 2, \ldots, d-1$.

The local part is now represented as

$$L_1 = \begin{bmatrix} * & c & 0 & \cdots & 0 \\ \frac{\mathrm{vback}_1}{\mathrm{Kback}_1} & \ddots & \ddots & \ddots & \vdots \\ 0 & \frac{2\mathrm{vback}_1}{\mathrm{Kback}_1+1} & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & c \\ 0 & \cdots & 0 & \frac{(n_1-1)\mathrm{vback}_1}{n_1+\mathrm{Kback}-2} & * \end{bmatrix}, \; L_d = \begin{bmatrix} * & 0 & \cdots & \cdots & 0 \\ \frac{\mathrm{vfor}_d}{\mathrm{Kfor}_d} & \ddots & \ddots & \ddots & \vdots \\ 0 & \frac{2\mathrm{vfor}_d}{\mathrm{Kfor}_d+1} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \frac{(n_d-1)\mathrm{vfor}_d}{\mathrm{Kfor}_d+n_d-1} & * \end{bmatrix}.$$

As for the matrices for the interactions, we need to break B_μ (and thus E_μ) from the previous model into two

$$B1_\mu = \begin{bmatrix} 0 & \cdots & \cdots & \cdots & 0 \\ \frac{v(1)_\mu}{K(1)_\mu} & \ddots & \ddots & \ddots & \vdots \\ 0 & \frac{2v(1)_\mu}{K(1)_\mu+1} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & \frac{(n_\mu-1)v(1)_\mu}{K(1)_\mu+n_\mu-2} & 0 \end{bmatrix}, \; B2_\mu = \begin{bmatrix} 0 & \cdots & \cdots & \cdots & 0 \\ \frac{v(2)_\mu}{K(2)_\mu} & \ddots & \ddots & \ddots & \vdots \\ 0 & \frac{2v(2)_\mu}{K(2)_\mu+1} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & \frac{(n_\mu-1)v(2)_\mu}{K(2)_\mu+n_\mu-2} & 0 \end{bmatrix},$$

while C_μ (and F_μ) is unchanged.

The parameters are: vector 'vback'; vector 'Kback'; vector 'vfor'; vector 'Kfor'; influx rate 'c'; vector with the maximum capacity of each system.

Same restrictions are again considered in [36], with 'Kback' and 'Kfor' having the restrictions from 'K' while 'vback' and 'vfor' have the restrictions from 'v', recalling the previous model. As for their experiments, they again consider vfor_1 taking the values $1.1c$ and $5c$; vfor_3 is either $8.4c$ or $6.9c$; vback_4 is either $1.6c$ or $3.7c$; Kfor_3 is either 2500 or 8000; Kback_4 is either 7700 or 3700. They again use $d = 6$. In our implementation, we use the default parameters: 'vback' and 'vfor' as $[0.1, 0.1, 0.1, 0.1, 0.1, 0.1]$; 'Kback' and 'Kfor' as $[1000, 1000, 1000, 1000, 1000, 1000]$; $c = 0.01$; maximum capacities $[16, 16, 16, 16, 16, 16]$. The problem size is thus 17^6.

dilutionintermed {chemical, funct, synch, functlocal, scalable, chain}.

This model is again from [36].

One now considers possible catabolism of intermediates or dilution due to growth. This makes the flux a conserved quantity throughout the pathway and is the basis of the flux-balance analysis [18].

In this context, this model generalizes directedmetab considering now that flux is not conserved, allowing particles to be degraded with a certain rate. Such degradation has a rate that is linear on the number of particles in a node, implying that, for system i, we have the rate

$$\text{cte} \times m_i, \tag{3.3}$$

where 'cte' is a constant while m_i is the number of particles of type i.

The TT operator is now

$$A_1(i_1, j_1) = \begin{bmatrix} L_1(i_1, j_1) & B_1(i_1, j_1) & E_1(i_1, j_1) & I_1(i_1, j_1) \end{bmatrix}, \quad A_d(i_d, j_d) = \begin{bmatrix} I_d(i_d, j_d) \\ C_d(i_d, j_d) \\ F_d(i_d, j_d) \\ L_d(i_d, j_d) \end{bmatrix},$$

and

$$A_\mu(i_\mu, j_\mu) = \begin{bmatrix} I_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ C_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ F_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ \beta L_\mu(i_\mu, j_\mu) & B_\mu(i_\mu, j_\mu) & E_\mu(i_\mu, j_\mu) & I_\mu(i_\mu, j_\mu) \end{bmatrix}. \quad \mu = 2, \ldots, d-1,$$

where β is the constant 'cte' in (3.3), considered to be the same for all systems.

The matrices associated with the local part are now

$$L_1 = \begin{bmatrix} * & c & 0 & \cdots & 0 \\ \beta & \ddots & \ddots & \ddots & \vdots \\ 0 & 2\beta & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & c \\ 0 & \cdots & 0 & (n_1 - 1)\beta & * \end{bmatrix}, \quad L_d = \begin{bmatrix} * & 0 & \cdots & \cdots & 0 \\ \beta + \frac{v_d}{K_d} & \ddots & \ddots & \ddots & \vdots \\ 0 & 2\beta + \frac{2v_d}{K_d+1} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & (n_d - 1)\beta + \frac{(n_d-1)v_d}{n_d+K_d-2} & * \end{bmatrix}$$

and

$$L_\mu = \begin{bmatrix} * & 0 & \cdots & \cdots & 0 \\ 1 & \ddots & \ddots & \ddots & \vdots \\ 0 & 2 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & (n_\mu - 1) & * \end{bmatrix},$$

$\mu = 2, \ldots, d-1$.

As for the interaction matrices, they are those from directedmetab as this degradation is a local event, thus affecting only the local part.

The parameters are: vector 'v'; vector 'K'; parameter β; influx rate 'c'; vector with the maximum capacity of each system.

The restrictions on the parameters, in [36], are the same as before, for directedmetab, on 'K' and 'v'. As for the experiments, they again consider v_1 taking the values $1.1c$ and $5c$; they consider the ratio $\frac{\beta}{c} = \frac{1}{100}$; their concretization is again defined for $d = 6$. In our implementation, we use the default parameters: 'v' is $[0.1, 0.1, 0.1, 0.1, 0.1, 0.1]$; 'K' is $[1000, 1000, 1000, 1000, 1000, 1000]$; $\beta = 0.0001$; $c = 0.01$; maximum capacities $[16, 16, 16, 16, 16, 16]$. The problem size is thus 17^6.

cyclicmetab{chemical, reducible, productform, funct, synch, scalable, chain}.

This model is again from [36].

Based on directedmetab, one adds the possibility that the last metabolite is converted into the first, introducing a cycle, similar to the one in cyclequeue. As in that model, local transitions are removed, meaning that no contact with the exterior of the network exists - the network is closed.

The TT operator is

$$A_1(i_1, j_1) = \begin{bmatrix} I_1(i_1, j_1) & B_1(i_1, j_1) & C_1(i_1, j_1) & E_1(i_1, j_1) & F_1(i_1, j_1) \end{bmatrix},$$
$$A_d(i_d, j_d) = \begin{bmatrix} C_d(i_d, j_d) & F_d(i_d, j_d) & I_d(i_d, j_d) & B_d(i_d, j_d) & E_d(i_d, j_d) \end{bmatrix}^T,$$

$$A_2(i_2, j_2) = \begin{bmatrix} I_2(i_2, j_2) & B_2(i_2, j_2) & E_2(i_2, j_2) & 0 & 0 & 0 \\ 0 & 0 & 0 & C_2(i_2, j_2) & 0 & 0 \\ 0 & 0 & 0 & 0 & I_2(i_2, j_2) & 0 \\ 0 & 0 & 0 & F_2(i_2, j_2) & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & I_2(i_2, j_2) \end{bmatrix},$$

$$A_{d-1}(i_{d-1}, j_{d-1}) = \begin{bmatrix} B_{d-1}(i_{d-1}, j_{d-1}) & E_{d-1}(i_{d-1}, j_{d-1}) & 0 & 0 & 0 \\ 0 & 0 & C_{d-1}(i_{d-1}, j_{d-1}) & 0 & 0 \\ 0 & 0 & F_{d-1}(i_{d-1}, j_{d-1}) & 0 & 0 \\ 0 & 0 & I_{d-1}(i_{d-1}, j_{d-1}) & 0 & 0 \\ 0 & 0 & 0 & I_{d-1}(i_{d-1}, j_{d-1}) & 0 \\ 0 & 0 & 0 & 0 & I_{d-1}(i_{d-1}, j_{d-1}) \end{bmatrix},$$

and

$$A_\mu(i_\mu, j_\mu) = \begin{bmatrix} I_\mu(i_\mu, j_\mu) & B_\mu(i_\mu, j_\mu) & E_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ 0 & 0 & 0 & C_\mu(i_\mu, j_\mu) & 0 & 0 \\ 0 & 0 & 0 & F_\mu(i_\mu, j_\mu) & 0 & 0 \\ 0 & 0 & 0 & I_\mu(i_\mu, j_\mu) & 0 & 0 \\ 0 & 0 & 0 & 0 & I_\mu(i_\mu, j_\mu) & 0 \\ 0 & 0 & 0 & 0 & 0 & I_\mu(i_\mu, j_\mu) \end{bmatrix}.$$

$\mu = 3, \ldots, d-2$.

As for the matrices, they are the ones from directedmetab, with the difference that now B_μ and C_μ are defined for all dimensions, in particular there exist B_d and C_1, while there are no local matrices.

The parameters are: vector 'v', as in directedmetab; vector 'K', as in directedmetab; vector with the maximum capacity of each system.

In our implementation, we use the default parameters: 'v' is $[0.1, 0.1, 0.1, 0.1, 0.1, 0.1]$; 'K' is $[1000, 1000, 1000, 1000, 1000, 1000]$; maximum capacities $[16, 16, 16, 16, 16, 16]$. The problem size is thus 17^6.

endproduct{chemical, funct, synch, scalable, chain}.

This mode is again from [36].

We now take into consideration that many biosynthesis pathways couple between supply and demand by a negative feedback [37, Ch. 3], meaning that the end-product inhibits the first reaction in the pathway or the transport of its precursor. This implies that the flux is reduced when the end-product builds up.

The idea is to add this effect to directedmetab.

The influx rate of the network, associated with the first system, is thus a decreasing function of the number of particles in the last substrate.

Such decreasing function is

$$c_0[1 + (\frac{m_d}{K_I})^h]^{-1}, \tag{3.4}$$

where c_0 is a constant associated with the influx (maximal influx); m_d is the end-product quantity; K_I is a constant associated with the dissociation between the first enzyme and the last substrate; h is a constant - Hill coefficient describing the cooperation in the interaction between the first enzyme and the last substrate.

The corresponding operator is

$$A_1(i_1, j_1) = \begin{bmatrix} I_1(i_1, j_1) & B_1(i_1, j_1) & C_1(i_1, j_1) & E_1(i_1, j_1) \end{bmatrix}, \; A_d(i_d, j_d) = \begin{bmatrix} C_d(i_d, j_d) \\ F_d(i_d, j_d) \\ I_d(i_d, j_d) \\ c_0 D_d(i_d, j_d) \\ L_d(i_d, j_d) \end{bmatrix},$$

$$A_2(i_2, j_2) = \begin{bmatrix} I_2(i_2, j_2) & B_2(i_2, j_2) & E_2(i_2, j_2) & 0 & 0 \\ 0 & 0 & 0 & C_2(i_2, j_2) & 0 \\ 0 & 0 & 0 & 0 & I_2(i_2, j_2) \\ 0 & 0 & 0 & F_2(i_2, j_2) & 0 \end{bmatrix},$$

$$A_{d-1}(i_{d-1}, j_{d-1}) = \begin{bmatrix} B_{d-1}(i_{d-1}, j_{d-1}) & E_{d-1}(i_{d-1}, j_{d-1}) & 0 & 0 & I_{d-1}(i_{d-1}, j_{d-1}) \\ 0 & 0 & C_{d-1}(i_{d-1}, j_{d-1}) & 0 & 0 \\ 0 & 0 & F_{d-1}(i_{d-1}, j_{d-1}) & 0 & 0 \\ 0 & 0 & I_{d-1}(i_{d-1}, j_{d-1}) & 0 & 0 \\ 0 & 0 & 0 & I_{d-1}(i_{d-1}, j_{d-1}) & 0 \end{bmatrix},$$

and

$$A_\mu(i_\mu, j_\mu) = \begin{bmatrix} I_\mu(i_\mu, j_\mu) & B_\mu(i_\mu, j_\mu) & E_\mu(i_\mu, j_\mu) & 0 & 0 \\ 0 & 0 & 0 & C_\mu(i_\mu, j_\mu) & 0 \\ 0 & 0 & 0 & F_\mu(i_\mu, j_\mu) & 0 \\ 0 & 0 & 0 & I_\mu(i_\mu, j_\mu) & 0 \\ 0 & 0 & 0 & 0 & I_\mu(i_\mu, j_\mu) \end{bmatrix},$$

$\mu = 3, \ldots, d-2$, where c_0 is the corresponding constant in (3.4).

The matrices are those from directedmetab, except for D_d, whose name is coherent with D_1 in multiprogramming, associated with the new interaction between last and

first systems, functional but not synchronized,

$$D_d = \begin{bmatrix} [1+(\frac{0}{K_I})^h]^{-1} & 0 & \cdots & 0 \\ 0 & [1+(\frac{1}{K_I})^h]^{-1} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & [1+(\frac{n_d-1}{K_I})^h]^{-1} \end{bmatrix},$$

where again the constants are the ones from (3.4).

The parameters are: vector 'v'; vector 'K'; entry K_I; value c_0; value h; vector with the maximum capacity of each system.

The restrictions on 'v' and 'K' are again those from directedmetab. In the experiments, they consider $K_I = 1000$, for a concretization again defined for $d = 6$. Our default parameters are: 'v' is $[0.1, 0.1, 0.1, 0.1, 0.1, 0.1]$; 'K' is $[1000, 1000, 1000, 1000, 1000, 1000]$; $K_I = 1000$; $c_0 = 0.01$; $h = 1$; maximum capacities $[16, 16, 16, 16, 16, 16]$. The problem size is thus 17^6.

divergingmetab {chemical, funct, synch, scalable, chain}.

This model is again from [36].

Many metabolites serve as substrates for several pathways. In such cases, different enzymes can bind to the substrate and each catabolizes a first reaction in a different pathway.

Within our scheme, this can be modelled by allowing one of the metabolites to be converted into one of two metabolites. The rates will depend on v_i and K_i through (3.2) again but, for the particular metabolite that can be converted in two different ones, there are two values for each of these variables, one for each possible reaction.

We add such effect to the basic configuration from directedmetab, in the sense that the main pathway and the two ones in which it splits all follow this model.

Finding an optimal ordering of the dimensions is not totally intuitive. We follow the natural ordering until reaching the metabolite that can be converted in two different ones, after which, when the splitting into two linear pathways happens, we go through all the systems in the first of the two pathways, doing then the same with the second.

For the described ordering of the systems, the operator is

$$A_1(i_1, j_1) = \begin{bmatrix} L_1(i_1, j_1) & B_1(i_1, j_1) & E_1(i_1, j_1) & I_1(i_1, j_1) \end{bmatrix}, \quad A_d(i_d, j_d) = \begin{bmatrix} I_d(i_d, j_d) \\ C_d(i_d, j_d) \\ F_d(i_d, j_d) \\ L_d(i_d, j_d) \end{bmatrix},$$

$$A_\mu(i_\mu, j_\mu) = \begin{bmatrix} I_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ C_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ F_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ 0 & B_\mu(i_\mu, j_\mu) & E_\mu(i_\mu, j_\mu) & I_\mu(i_\mu, j_\mu) \end{bmatrix}, \quad \mu = 2, \ldots, \mathrm{sp} - 1,$$

$$A_{\mathrm{sp}}(i_{\mathrm{sp}}, j_{\mathrm{sp}}) = \begin{bmatrix} I_{\mathrm{sp}}(i_{\mathrm{sp}}, j_{\mathrm{sp}}) & 0 & 0 & 0 & 0 & 0 \\ C_{\mathrm{sp}}(i_{\mathrm{sp}}, j_{\mathrm{sp}}) & 0 & 0 & 0 & 0 & 0 \\ F_{\mathrm{sp}}(i_{\mathrm{sp}}, j_{\mathrm{sp}}) & 0 & 0 & 0 & 0 & 0 \\ 0 & B1_{\mathrm{sp}}(i_{\mathrm{sp}}, j_{\mathrm{sp}}) & E1_{\mathrm{sp}}(i_{\mathrm{sp}}, j_{\mathrm{sp}}) & B2_{\mathrm{sp}}(i_{\mathrm{sp}}, j_{\mathrm{sp}}) & E2_{\mathrm{sp}}(i_{\mathrm{sp}}, j_{\mathrm{sp}}) & I_\mu(i_{\mathrm{sp}}, j_{\mathrm{sp}}) \end{bmatrix},$$

$$A_\mu(i_\mu, j_\mu) = \begin{bmatrix} I_\mu(i_\mu, j_\mu) & 0 & 0 & 0 & 0 & 0 \\ C_\mu(i_\mu, j_\mu) & 0 & 0 & 0 & 0 & 0 \\ F_\mu(i_\mu, j_\mu) & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & I_\mu(i_\mu, j_\mu) & 0 & 0 \\ 0 & 0 & 0 & 0 & I_\mu(i_\mu, j_\mu) & 0 \\ 0 & B_\mu(i_\mu, j_\mu) & E_\mu(i_\mu, j_\mu) & 0 & 0 & I_\mu(i_\mu, j_\mu) \end{bmatrix},$$

$\mu = \mathrm{sp}+1, \ldots, \mathrm{sp}+\mathrm{br}-2$. For $A_{\mathrm{sp+br-1}}(i_{\mathrm{sp+br-1}}, j_{\mathrm{sp+br-1}})$ we have

$$\begin{bmatrix} I_{\mathrm{sp+br-1}}(i_{\mathrm{sp+br-1}}, j_{\mathrm{sp+br-1}}) & 0 & 0 & 0 \\ C_{\mathrm{sp+br-1}}(i_{\mathrm{sp+br-1}}, j_{\mathrm{sp+br-1}}) & 0 & 0 & 0 \\ F_{\mathrm{sp+br-1}}(i_{\mathrm{sp+br-1}}, j_{\mathrm{sp+br-1}}) & 0 & 0 & 0 \\ 0 & I_{\mathrm{sp+br-1}}(i_{\mathrm{sp+br-1}}, j_{\mathrm{sp+br-1}}) & 0 & 0 \\ 0 & 0 & I_{\mathrm{sp+br-1}}(i_{\mathrm{sp+br-1}}, j_{\mathrm{sp+br-1}}) & 0 \\ L_{\mathrm{sp+br-1}}(i_{\mathrm{sp+br-1}}, j_{\mathrm{sp+br-1}}) & 0 & 0 & I_{\mathrm{sp+br-1}}(i_{\mathrm{sp+br-1}}, j_{\mathrm{sp+br-1}}) \end{bmatrix};$$

$$A_\mu(i_\mu, j_\mu) = \begin{bmatrix} I_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ C_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ F_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ 0 & B_\mu(i_\mu, j_\mu) & E_\mu(i_\mu, j_\mu) & I_\mu(i_\mu, j_\mu) \end{bmatrix}, \quad \mu = \mathrm{sp+br}, \ldots, d-1,$$

where 'sp' is used to denote the last system before the splitting, while 'br' is used to indicate the one where the first of the two linear pathways of the splitting ends. The second pathway thus starts in system $\mathrm{sp}+\mathrm{br}$.

While most of the matrices in the operator are again as in directedmetab, we need to separate the usual B_{sp} into $B1_{\mathrm{sp}}$ and $B2_{\mathrm{sp}}$

$$B1_{\mathrm{sp}} = \begin{bmatrix} 0 & \cdots & \cdots & \cdots & 0 \\ \frac{v_{\mathrm{sp}}}{K_{\mathrm{sp}}} & \ddots & \ddots & \ddots & \vdots \\ 0 & \frac{2v_{\mathrm{sp}}}{K_{\mathrm{sp}}+1} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & \frac{(n_{\mathrm{sp}}-1)v_{\mathrm{sp}}}{n_{\mathrm{sp}}+K_{\mathrm{sp}}-2} & 0 \end{bmatrix}, \quad B2_{\mathrm{sp}} = \begin{bmatrix} 0 & \cdots & \cdots & \cdots & 0 \\ \frac{v_{\mathrm{d+1}}}{K_{\mathrm{d+1}}} & \ddots & \ddots & \ddots & \vdots \\ 0 & \frac{2v_{\mathrm{d+1}}}{K_{\mathrm{d+1}}+1} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & \frac{(n_{\mathrm{sp}}-1)v_{\mathrm{d+1}}}{n_{\mathrm{sp}}+K_{\mathrm{d+1}}-2} & 0 \end{bmatrix}.$$

'v' and 'K' have now length $d+1$ as there are two entries for 'sp' dimension for each. As we cannot have two entries for v_{sp} and K_{sp}, we add the rate associated with the first pathway in v_{sp} and K_{sp} while adding the rates for the second pathway in the end of each vector, in $v_{\mathrm{d+1}}$ and $K_{\mathrm{d+1}}$.

The corresponding diagonal corrections are in $E1_{\mathrm{sp}}$ and $E2_{\mathrm{sp}}$, respectively.

Note that there are now two dimensions associated with ends of pathways, $\mathrm{sp}+\mathrm{br}-1$ and d, implying that we get an extra local transition, represented in $L_{\mathrm{sp+br-1}}$, which is just the same as L_d but with adjusted subscripts.

The ranks are greater than in directedmetab because of the interaction that exists between dimensions sp and $\mathrm{sp}+\mathrm{br}$, which suggests that it is optimal to make the first pathway the smallest of the two in order to minimize the distance between the mentioned dimensions, as the increased ranks are those between them.

The parameters are: vector 'v'; vector 'K'; influx rate 'c'; vector with the maximum capacity of each system; value of 'sp'; value of 'br'.

The restrictions on 'K' and 'v' are again the same as in directedmetab, in [36]. In the experiments, v_1 takes again the values $1.1c$ and $5c$; they consider $K_{\mathrm{sp}} =$

810 and $K_{d+1} = 370$; 'sp' is mentioned in the caption of the figure of the experiments to be 4 but in the actual image it seems to be 3; depending on this, 'br' is 3 or 4, respectively; their concretization is now defined for $d = 9$. In our implementation, we use the default values: 'v' is $[0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1]$; 'K' is $[1000, 1000, 1000, 1000, 1000, 1000, 1000]$; $c = 0.01$; 'sp' and 'br' are 2; maximum capacities $[16, 16, 16, 16, 16, 16]$. The problem size is thus 17^6.

convergingmetab {chemical, productform, funct, synch, scalable, chain}.

This model is again from [36].

Two pathways result in the synthesis of the same product, with two linear pathways merging into one. It is assumed that the different metabolites in the combined pathway, namely the two pathways producing the product and a pathway catabolizing the final product, remain decoupled.

This is combined with directedmetab in the sense that the two linear pathways that converge, as well as the one after these two converge, follow this model.

The ordering of the dimensions is again not trivial to define. We decide to first go through the first linear pathway; then we go through the second; then we follow the natural linear pathway starting from the dimension where those two merge.

The representation of the operator is

$$A_1(i_1, j_1) = \begin{bmatrix} L_1(i_1, j_1) & B_1(i_1, j_1) & E_1(i_1, j_1) & I_1(i_1, j_1) \end{bmatrix}, \quad A_d(i_d, j_d) = \begin{bmatrix} I_d(i_d, j_d) \\ C_d(i_d, j_d) \\ F_d(i_d, j_d) \\ L_d(i_d, j_d) \end{bmatrix},$$

$$A_\mu(i_\mu, j_\mu) = \begin{bmatrix} I_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ C_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ F_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ 0 & B_\mu(i_\mu, j_\mu) & E_\mu(i_\mu, j_\mu) & I_\mu(i_\mu, j_\mu) \end{bmatrix}, \quad \mu = 2, \ldots, \mathrm{br} - 1,$$

$$A_{\mathrm{br}}(i_{\mathrm{br}}, j_{\mathrm{br}}) = \begin{bmatrix} I_{\mathrm{br}}(i_{\mathrm{br}}, j_{\mathrm{br}}) & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & I_{\mathrm{br}}(i_{\mathrm{br}}, j_{\mathrm{br}}) & 0 \\ 0 & 0 & 0 & 0 & 0 & I_{\mathrm{br}}(i_{\mathrm{br}}, j_{\mathrm{br}}) \\ L_{\mathrm{br}}(i_{\mathrm{br}}, j_{\mathrm{br}}) & B_{\mathrm{br}}(i_{\mathrm{br}}, j_{\mathrm{br}}) & E_{\mathrm{br}}(i_{\mathrm{br}}, j_{\mathrm{br}}) & I_\mu(i_{\mathrm{br}}, j_{\mathrm{br}}) & 0 & 0 \end{bmatrix},$$

$$A_\mu(i_\mu, j_\mu) = \begin{bmatrix} I_\mu(i_\mu, j_\mu) & 0 & 0 & 0 & 0 & 0 \\ C_\mu(i_\mu, j_\mu) & 0 & 0 & 0 & 0 & 0 \\ F_\mu(i_\mu, j_\mu) & 0 & 0 & 0 & 0 & 0 \\ 0 & B_\mu(i_\mu, j_\mu) & E_\mu(i_\mu, j_\mu) & I_\mu(i_\mu, j_\mu) & 0 & 0 \\ 0 & 0 & 0 & 0 & I_\mu(i_\mu, j_\mu) & 0 \\ 0 & 0 & 0 & 0 & 0 & I_\mu(i_\mu, j_\mu) \end{bmatrix},$$

$\mu = \mathrm{br} + 1, \ldots, \mathrm{sp} - 2,$

$$A_{\mathrm{sp}-1}(i_{\mathrm{sp}-1}, j_{\mathrm{sp}-1}) = \begin{bmatrix} I_{\mathrm{sp}-1}(i_{\mathrm{sp}-1}, j_{\mathrm{sp}-1}) & 0 & 0 & 0 \\ C_{\mathrm{sp}-1}(i_{\mathrm{sp}-1}, j_{\mathrm{sp}-1}) & 0 & 0 & 0 \\ F_{\mathrm{sp}-1}(i_{\mathrm{sp}-1}, j_{\mathrm{sp}-1}) & 0 & 0 & 0 \\ 0 & B_{\mathrm{sp}-1}(i_{\mathrm{sp}-1}, j_{\mathrm{sp}-1}) & E_{\mathrm{sp}-1}(i_{\mathrm{sp}-1}, j_{\mathrm{sp}-1}) & I_{\mathrm{sp}-1}(i_{\mathrm{sp}-1}, j_{\mathrm{sp}-1}) \\ 0 & I_{\mathrm{sp}-1}(i_{\mathrm{sp}-1}, j_{\mathrm{sp}-1}) & 0 & 0 \\ 0 & 0 & I_{\mathrm{sp}-1}(i_{\mathrm{sp}-1}, j_{\mathrm{sp}-1}) & 0 \end{bmatrix},$$

$$A_\mu(i_\mu, j_\mu) = \begin{bmatrix} I_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ C_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ F_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ 0 & B_\mu(i_\mu, j_\mu) & E_\mu(i_\mu, j_\mu) & I_\mu(i_\mu, j_\mu) \end{bmatrix}, \quad \mu = \mathrm{sp} \ldots, d - 1,$$

where 'br' corresponds to the dimension where the second pathway starts while 'sp' is the dimension where the two pathways merge.

The matrices are the same as in directedmetab, except that there are now two dimensions associated with influxes as there are two linear pathways

$$L_1 = \begin{bmatrix} * & c(1) & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & c(1) \\ 0 & \cdots & \cdots & 0 & * \end{bmatrix}, \quad L_{\mathrm{br}} = \begin{bmatrix} * & c(2) & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & c(2) \\ 0 & \cdots & \cdots & 0 & * \end{bmatrix},$$

where now 'c', associated again with the constant influx, is a vector with two entries, with one influx rate for each pathway.

The ranks are increased, as in the previous model, from directedmetab, now due to the interaction between the first dimension and dimension 'sp'.

The parameters are: vector 'v'; vector 'K'; vector with the two influx rates 'c'; vector with the maximum capacity of each system; value of 'br'; value of 'sp'.

The same restrictions, in [36], as for directedmetab are considered on 'K' and 'v'. Experiments are done for v_1 being $1.1 \cdot c(1)$ or $5 \cdot c(1)$, again; then $c(2) = \frac{c(1)}{2}$; 'br' is 4 while 'sp' is 7; they consider $d = 9$. We consider the default parameters: 'v' is $[0.1, 0.1, 0.1, 0.1, 0.1, 0.1]$; 'K' is $[1000, 1000, 1000, 1000, 1000, 1000]$; $c = [0.01, 0.01]$; 'br' is 2; 'sp' is 4; maximum capacities $[16, 16, 16, 16, 16, 16]$. The problem size is thus 17^6.

convergingmetab2 {chemical, funct, synch, scalable, chain}.

The next model is a variant of the previous one, again taken from [36].

We have again two pathways resulting in the synthesis of one product. The difference is motivated by the fact that some reactions in a biosynthesis pathway involve side-reactants.

One thus considers that two products of two linear pathways serve as precursors for one reaction, meaning that one needs both products to generate the one associated with the dimension where the pathways converge.

The representation of the operator is, for the same ordering of the dimensions as in the previous model,

$$A_1(i_1, j_1) = \begin{bmatrix} L_1(i_1, j_1) & k(1)B_1(i_1, j_1) & k(1)E_1(i_1, j_1) & I_1(i_1, j_1) \end{bmatrix},$$
$$A_d(i_d, j_d) = \begin{bmatrix} I_d(i_d, j_d) & C_d(i_d, j_d) & F_d(i_d, j_d) & k(d)L_d(i_d, j_d) \end{bmatrix}^T,$$

$$A_\mu(i_\mu, j_\mu) = \begin{bmatrix} I_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ C_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ F_\mu(i_\mu, j_\mu) & 0 & 0 & 0 \\ 0 & k(\mu)B_\mu(i_\mu, j_\mu) & k(\mu)E_\mu(i_\mu, j_\mu) & I_\mu(i_\mu, j_\mu) \end{bmatrix}, \quad \mu = 2, \ldots, \mathrm{br}{-}1,$$

$$A_{\mathrm{br}}(i_{\mathrm{br}}, j_{\mathrm{br}}) = \begin{bmatrix} I_{\mathrm{br}}(i_{\mathrm{br}}, j_{\mathrm{br}}) & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & I_{\mathrm{br}}(i_{\mathrm{br}}, j_{\mathrm{br}}) & 0 \\ 0 & 0 & 0 & 0 & 0 & I_{\mathrm{br}}(i_{\mathrm{br}}, j_{\mathrm{br}}) \\ L_{\mathrm{br}}(i_{\mathrm{br}}, j_{\mathrm{br}}) & k(\mathrm{br})B_{\mathrm{br}}(i_{\mathrm{br}}, j_{\mathrm{br}}) & k(\mathrm{br})E_{\mathrm{br}}(i_{\mathrm{br}}, j_{\mathrm{br}}) & I_\mu(i_{\mathrm{br}}, j_{\mathrm{br}}) & 0 & 0 \end{bmatrix},$$

$$A_\mu(i_\mu, j_\mu) = \begin{bmatrix} I_\mu(i_\mu, j_\mu) & 0 & 0 & 0 & 0 & 0 \\ C_\mu(i_\mu, j_\mu) & 0 & 0 & 0 & 0 & 0 \\ F_\mu(i_\mu, j_\mu) & 0 & 0 & 0 & 0 & 0 \\ 0 & k(\mu)B_\mu(i_\mu, j_\mu) & k(\mu)E_\mu(i_\mu, j_\mu) & I_\mu(i_\mu, j_\mu) & 0 & 0 \\ 0 & 0 & 0 & 0 & I_\mu(i_\mu, j_\mu) & 0 \\ 0 & 0 & 0 & 0 & 0 & I_\mu(i_\mu, j_\mu) \end{bmatrix},$$

$\mu = \mathrm{br}+1, \ldots, \mathrm{sp}-2$,

$$A_{\mathrm{sp}-1}(i_{\mathrm{sp}-1}, j_{\mathrm{sp}-1}) = \begin{bmatrix} I_{\mathrm{sp}-1}(i_{\mathrm{sp}-1}, j_{\mathrm{sp}-1}) & 0 & 0 & 0 \\ C_{\mathrm{sp}-1}(i_{\mathrm{sp}-1}, j_{\mathrm{sp}-1}) & 0 & 0 & 0 \\ F_{\mathrm{sp}-1}(i_{\mathrm{sp}-1}, j_{\mathrm{sp}-1}) & 0 & 0 & 0 \\ 0 & 0 & 0 & I_{\mathrm{sp}-1}(i_{\mathrm{sp}-1}, j_{\mathrm{sp}-1}) \\ 0 & k(\mathrm{sp}-1)B_{\mathrm{sp}-1}(i_{\mathrm{sp}-1}, j_{\mathrm{sp}-1}) & 0 & 0 \\ 0 & 0 & k(\mathrm{sp}-1)E_{\mathrm{sp}-1}(i_{\mathrm{sp}-1}, j_{\mathrm{sp}-1}) & 0 \end{bmatrix},$$

$$A_{\mu}(i_{\mu}, j_{\mu}) = \begin{bmatrix} I_{\mu}(i_{\mu}, j_{\mu}) & 0 & 0 & 0 \\ C_{\mu}(i_{\mu}, j_{\mu}) & 0 & 0 & 0 \\ F_{\mu}(i_{\mu}, j_{\mu}) & 0 & 0 & 0 \\ 0 & k(\mu)B_{\mu}(i_{\mu}, j_{\mu}) & k(\mu)E_{\mu}(i_{\mu}, j_{\mu}) & I_{\mu}(i_{\mu}, j_{\mu}) \end{bmatrix}, \quad \mu = \mathrm{sp}\ldots, d-1,$$

where the meaning of 'br' and 'sp' is the same as in the previous model.

One important difference, emphasized in [36], to the previous model is that the rate for the laws has to be linear on the quantity of particles, of the form (3.3), instead of following (3.2). This is justified by the fact that unless we consider such form of the rates, because of the particular transition that converts two products into one, there is no steady-state.

We will thus need a vector of constants 'k', which replaces the vectors 'v' and 'K' from directedmetab as input, which will contain in position i, the constant, for system i, 'cte', from (3.3).

The matrices that get changed from the ones in the previous model are

$$L_d = \begin{bmatrix} * & 0 & \cdots & \cdots & 0 \\ 1 & \ddots & \ddots & \ddots & \vdots \\ 0 & 2 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & (n_d - 1) & * \end{bmatrix}, \quad B_{\mu} = \begin{bmatrix} * & 0 & \cdots & \cdots & 0 \\ 1 & \ddots & \ddots & \ddots & \vdots \\ 0 & 2 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & (n_{\mu} - 1) & * \end{bmatrix}.$$

The parameters are: vector 'k'; vector with two influx rates, as in the previous model, 'c'; vector with the maximum capacity of each system; value of 'br'; value of 'sp'.

We consider the default parameters: 'k' is $[0.0001, 0.0001, 0.0001, 0.0001, 0.0001, 0.0001]$; $c = [0.01, 0.01]$; 'br' is 2; 'sp' is 4; maximum capacities $[16, 16, 16, 16, 16, 16]$. The problem size is thus 17^6.

michaelismenten{chemical, reducible, productform, funct, synch, functlocal, chain}.

This model is taken from [36] again. It can be also found in [37, Ch. 1] or [19].

It is a well established model in the community of Chemical Networks.

It is possible nowadays to track the enzymatic turnover of substrate to product at the single-molecule level [53], and to study instantaneous metabolite concentration in the living cell. To describe this fluctuation mathematically, one models the cell as a reaction vessel characterized by the number of substrate molecules (S) and enzymes (E). A single molecule of S can bind to a single enzyme E with a certain rate that is a function of the number of molecules of each of the products that react, and form a complex, SE. This complex, in its turn, can unbind; or convert S into a product, P.

In a metabolic pathway, the number of substrate molecules is not kept fixed; rather, these molecules are synthesized or imported from the environment.

The set of reactions is summarized in a very clear scheme in [36]. The ordering of the dimensions follows naturally from such scheme - S, E, SE, P.

The corresponding operator is

$$A_1(i_1,j_1) = \begin{bmatrix} k(1)B_1(i_1,j_1) & C_1(i_1,j_1) & I_1(i_1,j_1) & k(1)E_1(i_1,j_1) & F_1(i_1,j_1) \end{bmatrix},$$
$$A_4(i_4,j_4) = \begin{bmatrix} I_4(i_4,j_4) & C_4(i_4,j_4) & F_4(i_4,j_4) \end{bmatrix}^T,$$

$$A_2(i_2,j_2) = \begin{bmatrix} B_2(i_2,j_2) & 0 & 0 & 0 & 0 & 0 & \frac{k(4)}{k(1)}I_2(i_2,j_2) \\ 0 & C_2(i_2,j_2) & 0 & 0 & 0 & 0 & cI_2(i_2,j_2) \\ 0 & 0 & C_2(i_2,j_2) & F_2(i_2,j_2) & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & E_2(i_2,j_2) & 0 & \frac{k(4)}{k(1)}I_2(i_2,j_2) \\ 0 & 0 & 0 & 0 & 0 & F_2(i_2,j_2) & cI_2(i_2,j_2) \end{bmatrix},$$

$$A_3(i_3,j_3) = \begin{bmatrix} C_3(i_3,j_3) & 0 & 0 \\ k(2)B_3(i_3,j_3) & 0 & 0 \\ 0 & k(3)B_3(i_3,j_3) & 0 \\ 0 & 0 & k(3)E_3(i_3,j_3) \\ F_3(i_3,j_3) & 0 & 0 \\ k(2)E_3(i_3,j_3) & 0 & 0 \\ I_3(i_3,j_3) & 0 & 0 \end{bmatrix}.$$

Matrix L_1, representing the connections between the first system and the exterior, is

$$L_1 = \begin{bmatrix} * & c & 0 & \cdots & 0 \\ k(4) & \ddots & \ddots & \ddots & \vdots \\ 0 & 2k(4) & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & c \\ 0 & \cdots & 0 & (n_1-1)k(4) & * \end{bmatrix},$$

where 'c' is the importing rate of particles S. This matrix does not appear in the final representation of the operator as we can write it as a linear combination of other matrices, which allows having a compressed representation, again using a previously mentioned technique from [31].

While C_μ, and thus F_μ, its diagonal correction, are the same as in directedmetab, we now have, for B_μ,

$$B_\mu = \begin{bmatrix} 0 & \cdots & \cdots & \cdots & 0 \\ 1 & \ddots & \ddots & \ddots & \vdots \\ 0 & 2 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & (n_\mu - 1) & 0 \end{bmatrix}.$$

A new vector 'k' appears inside the operator and it is associated with the rates of most reactions, except the local one associated with the importation of particles, which now stop following (3.2). Instead, one has this vector of constants, 'k', where each entry is associated with one reaction. The rates are given by

$$k_j \times_{i \in N} m_i, \tag{3.5}$$

where k_j is the constant associated with reaction j while the values in the set N are the number of the dimensions, in the chosen ordering, associated with systems that lose one particle in that reaction. m_i is the number of particles of type i.

The order in which the reactions are considered, in the context of vector 'k' in (3.5), is the following

$$\begin{aligned} S + E &\to SE; \\ SE &\to S + E; \\ SE &\to P + E; \\ S &\to \emptyset. \end{aligned}$$

Such scheme with the existing reactions also helps understanding the chosen ordering for the dimensions.

We additionally need the constant rate, denoted by 'c', associated with the influx transition

$$\emptyset \to S.$$

The associated Markov Chain is reducible as there are quantities that will remain constant.

The parameters are: vector 'k'; influx rate 'c'; vector with maximum capacities of the systems.

We consider the default parameters: 'k' as $[0.001, 0.001, 0.001, 0.001]$ and $c = 0.01$. Maximum capacities are $[64, 64, 64, 64]$, for a problem size 65^4.

enzymekineticsl {chemical, productform, funct, synch, functlocal, chain}.

This model is taken from [2], which focused on illustrating conditions under which product-form exists. It can also be found in [37, Ch. 2].

This model is very similar to michaelismenten. The dimensions are the same and the existing reactions are only very slightly changed.

Three reactions are added, including the connection between E and the exterior that allows the resulting Markov Chain to now be irreducible.

The reactions are ordered, in the context of vector 'k' in (3.5), as follows

$$\begin{aligned} S + E &\to SE; \\ SE &\to S + E; \\ SE &\to P + E; \\ P + E &\to SE; \\ S &\to \emptyset; \\ E &\to \emptyset. \end{aligned}$$

'c' is now a vector with two entries as it has the constant rates for two influx transitions, in the following order

$$\begin{aligned} \emptyset &\to S; \\ \emptyset &\to E. \end{aligned}$$

The operator is now

$$A_1(i_1,j_1)=\begin{bmatrix}k(1)B_1(i_1,j_1)\\ C_1(i_1,j_1)\\ I_1(i_1,j_1)\\ k(1)E_1(i_1,j_1)\\ F_1(i_1,j_1)\end{bmatrix}^T,\quad A_4(i_4,j_4)=\begin{bmatrix}I_4(i_4,j_4)\\ C_4(i_4,j_4)\\ F_4(i_4,j_4)\\ k(4)B_4(i_4,j_4)\\ k(4)E_4(i_4,j_4)\end{bmatrix},$$

$$A_2(i_2,j_2)=\begin{bmatrix}B_2(i_2,j_2) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{k(5)}{k(1)}I_2(i_2,j_2)\\ 0 & C_2(i_2,j_2) & 0 & 0 & 0 & 0 & 0 & 0 & c(1)I_2(i_2,j_2)\\ 0 & 0 & C_2(i_2,j_2) & F_2(i_2,j_2) & B_2(i_2,j_2) & E_2(i_2,j_2) & 0 & 0 & L_2(i_2,j_2)\\ 0 & 0 & 0 & 0 & 0 & 0 & E_2(i_2,j_2) & 0 & \frac{k(5)}{k(1)}I_2(i_2,j_2)\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & F_2(i_2,j_2) & c(1)I_2(i_2,j_2)\end{bmatrix},$$

$$A_3(i_3,j_3)=\begin{bmatrix}C_3(i_3,j_3) & 0 & 0 & 0 & 0\\ k(2)B_3(i_3,j_3) & 0 & 0 & 0 & 0\\ 0 & k(3)B_3(i_3,j_3) & 0 & 0 & 0\\ 0 & 0 & k(3)E_3(i_3,j_3) & 0 & 0\\ 0 & 0 & 0 & C_3(i_3,j_3) & 0\\ 0 & 0 & 0 & 0 & F_3(i_3,j_3)\\ F_3(i_3,j_3) & 0 & 0 & 0 & 0\\ k(2)E_3(i_3,j_3) & 0 & 0 & 0 & 0\\ I_3(i_3,j_3) & 0 & 0 & 0 & 0\end{bmatrix}.$$

While the matrices for the interactions are the same, the local part is now

$$L_1=\begin{bmatrix}* & c(1) & 0 & \cdots & 0\\ k(5) & \ddots & \ddots & \ddots & \vdots\\ 0 & 2k(5) & \ddots & \ddots & 0\\ \vdots & \ddots & \ddots & \ddots & c(1)\\ 0 & \cdots & 0 & (n_1-1)k(5) & *\end{bmatrix},\ L_2=\begin{bmatrix}* & c(2) & 0 & \cdots & 0\\ k(6) & \ddots & \ddots & \ddots & \vdots\\ 0 & 2k(6) & \ddots & \ddots & 0\\ \vdots & \ddots & \ddots & \ddots & c(2)\\ 0 & \cdots & 0 & (n_2-1)k(6) & *\end{bmatrix}.$$

The first is again not present in the final representation of the operator just as L_1 in the previous model.

The parameters are: vector 'k'; vector 'c'; vector with maximum capacities of the systems.

We consider the default parameters: 'k' as $[0.001, 0.001, 0.001, 0.001, 0.001, 0.001]$ and $c = [0.01, 0.01]$. Maximum capacities are $[64, 64, 64, 64]$, for a problem size, again, 65^4.

enzymekineticsII {chemical, reducible, productform, funct, synch, functlocal, chain}.

This model is again taken from [2], while it can be also again found in [37, Ch. 2].

This model is similar to michaelismenten and enzymekineticsI.

In particular, the difference to enzymekineticsI is only that S is now not connected with the exterior.

The operator is now

$$A_1(i_1,j_1)=\begin{bmatrix}k(1)B_1(i_1,j_1)\\ C_1(i_1,j_1)\\ I_1(i_1,j_1)\\ k(1)E_1(i_1,j_1)\\ F_1(i_1,j_1)\end{bmatrix}^T,\quad A_4(i_4,j_4)=\begin{bmatrix}I_4(i_4,j_4)\\ C_4(i_4,j_4)\\ F_4(i_4,j_4)\\ k(4)B_4(i_4,j_4)\\ k(4)E_4(i_4,j_4)\end{bmatrix},$$

$$A_2(i_2,j_2)=\begin{bmatrix} B_2(i_2,j_2) & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & C_2(i_2,j_2) & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & C_2(i_2,j_2) & F_2(i_2,j_2) & B_2(i_2,j_2) & E_2(i_2,j_2) & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & E_2(i_2,j_2) & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & F_2(i_2,j_2) \end{bmatrix},$$

$$A_3(i_3,j_3)=\begin{bmatrix} C_3(i_3,j_3) & 0 & 0 & 0 & 0 \\ k(2)B_3(i_3,j_3) & 0 & 0 & 0 & 0 \\ cI_3(i_3,j_3) & k(3)B_3(i_3,j_3) & 0 & 0 & 0 \\ cI_3(i_3,j_3) & 0 & k(3)E_3(i_3,j_3) & 0 & 0 \\ k(5)I_3(i_3,j_3) & 0 & 0 & C_3(i_3,j_3) & 0 \\ k(5)I_3(i_3,j_3) & 0 & 0 & 0 & F_3(i_3,j_3) \\ F_3(i_3,j_3) & 0 & 0 & 0 & 0 \\ k(2)E_3(i_3,j_3) & 0 & 0 & 0 & 0 \end{bmatrix},$$

where 'c' denotes the importing rate of E - local transition. As for vector 'k', it is associated with the same ordering of the reactions as in enzymekineticsl after removing the one involving the one associated with the now removed connection of S with the exterior (previous fifth entry).

The operator changes quite significantly compared with enzymekineticsl given that L_1 stops existing while the corresponding L_2, represented as

$$L_2=\begin{bmatrix} * & c & 0 & \cdots & 0 \\ k(5) & \ddots & \ddots & \ddots & \vdots \\ 0 & 2k(5) & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & c \\ 0 & \cdots & 0 & (n_2-1)k(5) & * \end{bmatrix},$$

can be, using again the technique from [31] to reduce the ranks, now for a middle dimension but which works the same way, omitted from the final representation of the operator.

The matrices inside the operator above are the same as for michaelismenten.

The parameters are: vector 'k'; influx rate 'c'; vector with maximum capacities of the systems.

We consider the default parameters: 'k' as $[0.001, 0.001, 0.001, 0.001, 0.001]$ and $c = 0.01$. Maximum capacities are $[64, 64, 64, 64]$, for a problem size, again, 65^4.

multiscalechemical {chemical, reducible, productform, funct, synch, functlocal, chain}.

This model is again from [2].

It is associated with six chemical species, with interactions that have again rates following (3.5).

The dimensions are associated with: enzymes E1 and E2; substrates A and S; resulting product P; and a final type C.

We make the same type of representation of the interactions as for the three

models above, ordered again according to vector 'k' associated with (3.5),

$$\begin{aligned} E2 + A &\to E1; \\ E1 &\to E2 + A; \\ E1 + S &\to C; \\ C &\to E1 + S; \\ C &\to P + E1; \\ E2 &\to \emptyset. \end{aligned}$$

In the sequence of the interactions above, the ordering of the dimensions is chosen as: E2, A, E1, S, C, P.

There is then the additional transition, associated with the local part, concerning the importation, with constant rate 'c', of E2

$$\emptyset \to E2.$$

The operator is

$$A_1(i_1, j_1) = [k(1)B_1(i_1, j_1) \;\; C_1(i_1, j_1) \;\; I_1(i_1, j_1) \;\; k(1)E_1(i_1, j_1) \;\; F_1(i_1, j_1)],$$
$$A_6(i_6, j_6) = \begin{bmatrix} I_6(i_6, j_6) & C_6(i_6, j_6) & F_6(i_6, j_6) \end{bmatrix}^T,$$

$$A_2(i_2, j_2) = \begin{bmatrix} B_2(i_2, j_2) & 0 & 0 & 0 & 0 & \frac{k(6)}{k(1)} I_2(i_2, j_2) \\ 0 & C_2(i_2, j_2) & 0 & 0 & 0 & cI_2(i_2, j_2) \\ 0 & 0 & I_2(i_2, j_2) & 0 & 0 & 0 \\ 0 & 0 & 0 & E_2(i_2, j_2) & 0 & \frac{k(6)}{k(1)} I_2(i_2, j_2) \\ 0 & 0 & 0 & 0 & F_2(i_2, j_2) & cI_2(i_2, j_2) \end{bmatrix},$$

$$A_3(i_3, j_3) = \begin{bmatrix} C_3(i_3, j_3) & 0 & 0 & 0 & 0 \\ k(2)B_3(i_3, j_3) & 0 & 0 & 0 & 0 \\ 0 & k(3)B_3(i_3, j_3) & C_3(i_3, j_3) & k(3)E_3(i_3, j_3) & F_3(i_3, j_3) \\ F_3(i_3, j_3) & 0 & 0 & 0 & 0 \\ k(2)E_3(i_3, j_3) & 0 & 0 & 0 & 0 \\ I_3(i_3, j_3) & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$A_4(i_4, j_4) = \begin{bmatrix} I_4(i_4, j_4) & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & B_4(i_4, j_4) & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & C_4(i_4, j_4) & 0 & 0 & I_4(i_4, j_4) & 0 \\ 0 & 0 & 0 & E_4(i_4, j_4) & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & F_4(i_4, j_4) & 0 & I_4(i_4, j_4) \end{bmatrix},$$

$$A_5(i_5, j_5) = \begin{bmatrix} I_5(i_5, j_5) & 0 & 0 \\ C_5(i_5, j_5) & 0 & 0 \\ k(4)B_5(i_5, j_5) & 0 & 0 \\ F_5(i_5, j_5) & 0 & 0 \\ k(4)E_5(i_5, j_5) & 0 & 0 \\ 0 & k(5)B_5(i_5, j_5) & 0 \\ 0 & 0 & k(5)E_5(i_5, j_5) \end{bmatrix}.$$

We note that, once again, the matrix for the local part, L_1, can be omitted from the final representation.

All matrices inside the operator are as in michaelismenten.

The parameters are: vector 'k'; influx rate 'c'; vector with maximum capacities of the systems.

We consider the default parameters: 'k' as $[0.001, 0.001, 0.001, 0.001, 0.001, 0.001]$ and $c = 0.01$. Maximum capacities are $[16, 16, 16, 16, 16, 16]$, for a problem size 17^6.

enzymaticfutile{chemical, reducible, synch, chain}.

This model was first introduced in [47], where the steady-state problem is highlighted; it was later used in the numerical experiments of [30].

Futile cycles are composed of two metabolic pathways that work in opposite directions, meaning that the products of one pathway are the precursors for the other and vice versa. This biochemical network structure results in no net production of molecules and often results only in the dissipation of energy as heat [49]. Nevertheless, there is an abundance of known pathways that use this motif.

The stochastic model consists of six chemical species (six systems) and six reactions. The species are: X and X^*, representing forward substrate and product; E_+ and E_-, for the forward and reverse enzymes, respectively.

We again use the same type of representation of the interactions, ordered according to vector 'k' from (3.5)

$$\begin{aligned} X + E_+^f &\to E_+^b; \\ E_+^b &\to X + E_+^f \\ X^* + E_-^f &\to E_-^b; \\ E_-^b &\to X^* + E_-^f; \\ E_+^b &\to X^* + E_+^f; \\ E_-^b &\to X + E_+^b. \end{aligned}$$

The network is closed.

The way to optimally order the dimensions is again not trivial. In the end, based again on the set of existing interactions above, we choose the ordering: E_+^f, E_+^b, X, X^*, E_-^f, E_-^b.

For the choice above, one gets the operator

$$\begin{aligned} A_1(i_1, j_1) &= \begin{bmatrix} I_1(i_1, j_1) & k(1)B_1(i_1, j_1) & k(1)E_1(i_1, j_1) & C_1(i_1, j_1) & F_1(i_1, j_1) \end{bmatrix}, \\ A_6(i_6, j_6) &= \begin{bmatrix} C_6(i_6, j_6) & F_6(i_6, j_6) & B_6(i_6, j_6) & E_6(i_6, j_6) & I_6(i_6, j_6) \end{bmatrix}^T, \end{aligned}$$

$$A_2(i_2, j_2) = \begin{bmatrix} I_2(i_2, j_2) & 0 & 0 & 0 & 0 \\ 0 & C_2(i_2, j_2) & 0 & 0 & 0 \\ 0 & 0 & F_2(i_2, j_2) & 0 & 0 \\ 0 & 0 & 0 & B_2(i_2, j_2) & 0 \\ 0 & 0 & 0 & 0 & E_2(i_2, j_2) \end{bmatrix},$$

$$A_3(i_3, j_3) = \begin{bmatrix} I_3(i_3, j_3) & k(6)C_3(i_3, j_3) & k(6)F_3(i_3, j_3) & 0 & 0 & 0 \\ 0 & 0 & 0 & B_3(i_3, j_3) & 0 & 0 \\ 0 & 0 & 0 & E_3(i_3, j_3) & 0 & 0 \\ 0 & 0 & 0 & k(2)C_3(i_3, j_3) & I_3(i_3, j_3) & 0 \\ 0 & 0 & 0 & k(2)F_3(i_3, j_3) & 0 & I_3(i_3, j_3) \end{bmatrix},$$

$$A_4(i_4,j_4)=\begin{bmatrix} k(3)B_4(i_4,j_4) & k(3)E_4(i_4,j_4) & k(4)C_4(i_4,j_4) & k(4)F_4(i_4,j_4) & 0 \\ 0 & 0 & I_4(i_4,j_4) & 0 & 0 \\ 0 & 0 & 0 & I_4(i_4,j_4) & 0 \\ 0 & 0 & 0 & 0 & I_4(i_4,j_4) \\ 0 & 0 & 0 & 0 & k(5)C_4(i_4,j_4) \\ 0 & 0 & 0 & 0 & k(5)F_4(i_4,j_4) \end{bmatrix},$$

$$A_5(i_5,j_5)=\begin{bmatrix} B_5(i_5,j_5) & 0 & 0 & 0 & 0 \\ 0 & E_5(i_5,j_5) & 0 & 0 & 0 \\ 0 & 0 & C_5(i_5,j_5) & 0 & 0 \\ 0 & 0 & 0 & F_5(i_5,j_5) & 0 \\ 0 & 0 & 0 & 0 & I_5(i_5,j_5) \end{bmatrix}.$$

All matrices inside the operator are again as in michaelismenten.

The parameters are: vector 'k'; vector with maximum capacities of the systems.

One has the reference parameters from the experiments in [30], based on [48], where the vector 'k' is $[40, 10^4, 200, 100, 10^4, 5000]$ while maximum capacities are $[3, 3, 127, 127, 3, 3]$. We consider the default parameters 'k' as in the mentioned reference while maximum capacities are $[16, 16, 16, 16, 16, 16]$, in order to have a problem size 17^6 again.

cascadegene{chemical, funct, scalable, chain}.

This model can be found in the numerical experiments of [28, 1, 16]. All references concern the solution of the CME. All references test algorithms that make use of the Kronecker structure of the problem, in particular with the last using TT format through the well-established algorithm for solving linear systems that was later adapted in [32] to a least squares formulation with the goal of finding the steady-state.

It is similar to directedmetab in topology and mainly differs in the fact that the interactions do not include losing a particle of one type while forming one from the next type, but the rate still depends on the quantity of particles of the first. This basically means that the interactions are still functional but not synchronized.

The function that determines the rate of creation for system $\mu = 2, \ldots, d$ is

$$\frac{\beta m_{\mu-1}}{\beta m_{\mu-1}+\gamma}, \tag{3.6}$$

where β and γ are constants of the model and $m_{\mu-1}$ is the number of particles of type $\mu - 1$.

Note that the law above is very similar to (3.2).

The possibility of local degradation, as in dilutionintermed, is also considered, with constant δ taking again the role of 'cte' in 3.3.

For the influx, the rate is the constant α_0.

The operator is

$$A_1(i_1,j_1)=\begin{bmatrix} L_1(i_1,j_1) & cB_1(i_1,j_1) & I_1(i_1,j_1) \end{bmatrix}, \quad A_d(i_d,j_d)=\begin{bmatrix} I_d(i_d,j_d) \\ C_d(i_d,j_d) \\ L_d(i_d,j_d) \end{bmatrix},$$

and

$$A_\mu(i_\mu,j_\mu)=\begin{bmatrix} I_\mu(i_\mu,j_\mu) & 0 & 0 \\ C_\mu(i_\mu,j_\mu) & 0 & 0 \\ L_\mu(i_\mu,j_\mu) & \mathrm{arr}(\mu)B_\mu(i_\mu,j_\mu) & I_\mu(i_\mu,j_\mu) \end{bmatrix}. \quad \mu = 2, \ldots, d-1.$$

The local part is

$$L_1 = \begin{bmatrix} * & \alpha_0 & 0 & \cdots & 0 \\ \delta & \ddots & \ddots & \ddots & \vdots \\ 0 & 2\delta & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \alpha_0 \\ 0 & \cdots & 0 & (n_1-1)\delta & * \end{bmatrix}, \quad L_d = \begin{bmatrix} * & 0 & \cdots & \cdots & 0 \\ \delta + \frac{\beta}{\beta+\gamma} & \ddots & \ddots & \ddots & \vdots \\ 0 & 2\delta + \frac{2\beta}{2\beta+\gamma} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & (n_d-1)\delta + \frac{(n_d-1)\beta}{(n_d-1)\beta+\gamma} & * \end{bmatrix}$$

and

$$L_\mu = \begin{bmatrix} * & 0 & \cdots & \cdots & 0 \\ 1 & \ddots & \ddots & \ddots & \vdots \\ 0 & 2 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & (n_\mu - 1) & * \end{bmatrix},$$

$\mu = 2, \ldots, d-1$.

For the interaction matrices,

$$B_\mu = \begin{bmatrix} \frac{\beta}{\beta+\gamma} & 0 & \cdots & 0 \\ 0 & \frac{2\beta}{2\beta+\gamma} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \frac{(n_\mu-1)\beta}{(n_\mu-1)\beta+\gamma} \end{bmatrix}, \quad C_\mu = \begin{bmatrix} * & 1 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & 1 \\ 0 & \cdots & \cdots & 0 & * \end{bmatrix},$$

where B_μ are now diagonal matrices and one can thus include the diagonal corrections inside C_μ in the sequence of Th. 2, in Appendix A. This is more generally because the interactions are not associated with synchronized transitions as noted in the context of overflow.

The operator is

$$A_1(i_1, j_1) = \begin{bmatrix} L_1(i_1, j_1) & B_1(i_1, j_1) & I_1(i_1, j_1) \end{bmatrix}, \quad A_d(i_d, j_d) = \begin{bmatrix} I_d(i_d, j_d) \\ C_d(i_d, j_d) \\ L_d(i_d, j_d) \end{bmatrix},$$

and

$$A_\mu(i_\mu, j_\mu) = \begin{bmatrix} I_\mu(i_\mu, j_\mu) & 0 & 0 \\ C_\mu(i_\mu, j_\mu) & 0 & 0 \\ \beta L_\mu(i_\mu, j_\mu) & B_\mu(i_\mu, j_\mu) & I_\mu(i_\mu, j_\mu) \end{bmatrix}. \quad \mu = 2, \ldots, d-1.$$

The parameters are: δ; β; γ; α_0; vector with maximum capacities.

In the experiments of all references, the parameters are the same as [28] is followed by the other two: $\delta = 0.07$; $\beta = 1$; $\gamma = 5$; $\alpha_0 = 0.7$; and maximum capacities 2^{12}. The difference is only in the number of dimensions as [28] considers $d = 3$ while the other two references consider $d = 20$. We use the parameters in the mentioned references, but for maximum capacities $[16, 16, 16, 16, 16, 16]$, giving the usual problem size 17^6.

toggleswitch{chemical, funct, chain}.

This model is taken from [22, 40].

It can also be found in the numerical experiments of [30], and also of [28, 1, 51, 15] but with slightly different laws for the functional transitions for these last ones. All experiments are again associated with the solution of the CME.

It models a synthetic gene-regulatory circuit designed to produce bi-stability over a wide range of parameter values. The network is composed of two repressors and two constitutive promoters arranged in a feedback loop meaning that each promoter is inhibited by the repressor transcribed by the opposing promoter. If the concentration of one repressor is high, this lowers the production rate of the other repressor, keeping its concentration low. This allows a high rate of production of the original repressor, thereby stabilizing its high concentration.

The stochastic model consists of two species (two systems) and four reactions.

The rate of creation of a species of the first type is

$$\frac{\alpha_1}{1+m_2^{\beta}},$$

where α_1 and β are constants of the model and m_2 is the quantity of particles of the second type.

The rate of creation of a particle of the second type is

$$\frac{\alpha_2}{1+m_1^{\gamma}},$$

where α_2 and γ are constants of the model and m_1 denotes the quantity of particles of the first type.

As for the rates of local degeneration, they follow the usual linear law (3.3), from which 'cte' is here δ_i, for particles of type i.

The operator is

$$A_1(i_1,j_1) = \begin{bmatrix} \delta_1 L_1(i_1,j_1) & I_1(i_1,j_1) & B_1(i_1,j_1) & C_1(i_1,j_1) \end{bmatrix}, \quad A_2(i_2,j_2) = \begin{bmatrix} I_2(i_2,j_2) \\ \delta_2 L_2(i_2,j_2) \\ C_2(i_2,j_2) \\ B_2(i_2,j_2) \end{bmatrix}.$$

The local part concerns the degeneration of particles

$$L_\mu = \begin{bmatrix} * & 0 & \cdots & \cdots & 0 \\ 1 & \ddots & \ddots & \ddots & \vdots \\ 0 & 2 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & (n_\mu - 1) & * \end{bmatrix}.$$

The remaining matrices are given by

$$B_1 = \begin{bmatrix} \frac{\alpha_2}{1+0^\gamma} & 0 & \cdots & 0 \\ 0 & \frac{\alpha_2}{1+1^\gamma} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \frac{\alpha_2}{1+(n_1-1)^\gamma} \end{bmatrix}, \quad B_2 = \begin{bmatrix} \frac{\alpha_1}{1+0^\gamma} & 0 & \cdots & 0 \\ 0 & \frac{\alpha_1}{1+1^\beta} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \frac{\alpha_1}{1+(n_2-1)^\beta} \end{bmatrix},$$

where, as in the previous model, B_μ is a diagonal matrix as it is associated with a functional but not synchronized transition, allowing C_μ to be the same as in the mentioned model, in particular including the needed diagonal corrections, again in the sequence of Th. 2, in Appendix A.

The parameters are: β; γ; α; δ; vector with maximum capacities of the systems.

The parameters used in [40] are: $\beta = 1$; $\gamma = 1$; $\alpha = [25, 30]$; $\delta = [1, 1]$; maximum capacities of $[64, 88]$. They then try two more sets of combinations (we only present the changed parameters): $\alpha = [80, 100]$, maximum capacities of $[120, 200]$; and $\beta = 2.5$, $\alpha = [100, 25]$, maximum capacities of $[176, 96]$. In [30] they consider: $\beta = 2.5$; $\gamma = 1.5$; $\alpha = [5000, 1600]$; $\delta = [1, 1]$; maximum capacities of $[8191, 4095]$. The parameters, thus, do not seem too strict in general. We choose to use as default values in our implementation the first combination - $\beta = 1$; $\gamma = 1$; $\alpha = [25, 30]$; $\delta = [1, 1]$ - combined with maximum capacities $[1024, 1024]$, for a problem size 1025^2.

4. Utilization of our implementation. Our implementation of the operators for the models in this collection is done in MATLAB. It is available in `http://anchp.epfl.ch/SAN_TT`.

The function for a particular model is saved in a script with the name that is specified in Tab. 2.

The input parameters for the different models are as detailed in the end of the more detailed description of the corresponding model, where we also add an indication of what parameters to use, in MATLAB notation, based on the references of the models, in case they exist. Default parameters are also specified as the user has two options - using the default parameters (by calling the desired model without input parameters); or choosing them, by giving them as input. As for the output, it is a cell $d \times 1$ with row i consisting of a representation of the core i of the TT operator. We now exemplify for model overflow:

```
>> a = overflow([1.2,1.1,1.0,0.9,0.8,0.7],[1,1,1,1,1,1],[16,16,16,16,16,16]);
>> b = overflow;
>> a
a =
    [17x17x3 double]
    [ 4-D    double]
    [ 4-D    double]
    [ 4-D    double]
    [ 4-D    double]
    [17x17x3 double]
>> b
b =
    [17x17x3 double]
    [ 4-D    double]
    [ 4-D    double]
    [ 4-D    double]
    [ 4-D    double]
    [17x17x3 double]
```

For the use of functionality from the TT-Toolbox [41], one needs to apply the function tt_matrix to this resulting cell, while in previous versions of this toolbox this cell would be immediately read as a TT operator:

```
>> tena = tt_matrix(a);
>> tenb = tt_matrix(b);
>> tena
```

```
tena is a 6-dimensional TT-matrix, ranks and mode sizes:
r(1)=1   n(1)=17 m(1) = 17
r(2)=3   n(2)=17 m(2) = 17
r(3)=3   n(3)=17 m(3) = 17
r(4)=3   n(4)=17 m(4) = 17
r(5)=3   n(5)=17 m(5) = 17
r(6)=3   n(6)=17 m(6) = 17
r(7)=1
>> tenb
tenb is a 6-dimensional TT-matrix, ranks and mode sizes:
r(1)=1   n(1)=17 m(1) = 17
r(2)=3   n(2)=17 m(2) = 17
r(3)=3   n(3)=17 m(3) = 17
r(4)=3   n(4)=17 m(4) = 17
r(5)=3   n(5)=17 m(5) = 17
r(6)=3   n(6)=17 m(6) = 17
r(7)=1
```

One can now check that the obtained operators are the same, using the default parameters ('tena') or explicitly giving them as input ('tenb'):

```
>> norm(tena-tenb)/norm(tena)
ans =
     1.610351863521918e-15
```

Note that it is then possible to apply function full to obtain the full transition rate matrix, which must however be done carefully as, in particular when the mode sizes and the number of dimensions are relevant for testing algorithms, this will easily lead to memory problems:

```
>> full(tena);
Error using  *
Out of memory. Type HELP MEMORY for your options.
Error in tt_tensor/full (line 25)
  a=a*cr;
Error in tt_matrix/full (line 18)
a=full(tt1);
```

Even for the models with the identifier scalable, the number of dimensions is not an input of the models. It is chosen implicitly through other input parameters, in particular through the size of some vectors. In the example above, for overflow, the size of the vector with the maximum capacities, for instance, defines the number of dimensions.

5. Conclusions. This collection of models shows the big variety of applications associated with SAN and we provide such representative problems for testing, with implementations of all operators, in TT format, for the transition rate matrix of the introduced models, in MATLAB. We hope that it can be smoothly used for developing, testing and comparing algorithms for the associated relevant problems as, for instance, finding the steady-state of the models, which is a very important problem in the context of SAN, in particular because many measures of interest about the models are extracted from their steady-state. The introduction of identifiers for characterizing the different models, based on relevant characteristics, makes it easier for the user to extract specialized subsets of the collection for use in numerical experiments.

Acknowledgements. I thank Daniel Kressner (EPF Lausanne) for helpful discussions. I thank Christian Mazza (University of Fribourg) for contributing to the collec-

tion by providing the references for most of the included models related with Chemical Networks. I also thank Vladimir Kazeev (ETH Zürich) for helping to understand the dynamics of the models taken from [30].

Appendix A.
Proof concerning the diagonal correction of matrices associated with the TT operators for some particular models.

We here prove that it is possible to include, in the context of a Kronecker product, the diagonal correction (2.1) inside that same Kronecker product, in case the involved matrices verify a particular property. This property is that all except for one are diagonal matrices (in the context, as it is the case for such SAN models, of square matrices).

First of all, it is clear from the definition of Kronecker product that the term associated with the diagonal correction that is needed, negative sum of the off-diagonals of the corresponding row - (2.1), can be represented, starting with a Kronecker product between only two matrices, let us call them $A1$ and $A2$, by

$$-\text{diag}[(A1 \otimes A2)(e_{A1} \otimes e_{A2})], \tag{A.1}$$

where e_{A1} and e_{A2} denote the vectors of all ones with the number of rows of $A1$ and $A2$, respectively. In its turn, 'diag' represents the MATLAB operation of defining a diagonal square matrix from a vector, by defining the diagonal with this vector.

A well-known property for Kronecker products is the following: for any matrices $A1$, $A2$, $A3$ and $A4$, where the dimensions of $A1$ and $A3$ match, as well as those of $A2$ and $A4$,

$$(A1 \otimes A2)(A3 \otimes A4) = (A1 \times A3) \otimes (A2 \times A4). \tag{A.2}$$

We also need the following property, which can be easily derived directly from the definition of Kronecker product, for matrices $A1$, $A2$ and $A3$, where the first two need to have the same sizes:

$$A \otimes C - B \otimes C = (A - B) \otimes C \tag{A.3}$$

We can now prove the desired property, for the case of two matrices.

LEMMA 1. *Given two square matrices A and D, where D is diagonal,*

$$A \otimes D - \mathit{diag}[(A \otimes D)(e_A \otimes e_D)] = (A - \mathit{diag}(A \times e_A)) \otimes D.$$

Proof.

$$A \otimes D - \text{diag}[(A \otimes D)(e_A \otimes e_D)] \tag{A.4}$$
$$= A \otimes D - \text{diag}(A \times e_A) \otimes \text{diag}(D \times e_D) \tag{A.5}$$
$$= A \otimes D - \text{diag}(A \times e_A) \otimes D \tag{A.6}$$
$$= (A - \text{diag}(A \times e_A)) \otimes D. \tag{A.7}$$

From (A.4) to (A.5), we use (A.2). From (A.5) to (A.6), we use the fact that $D \times e_D$ is a vector with the entries in the diagonal of D, implying that 'diag', as defined before, applied to that, gives D again. From (A.6) to (A.7), we use (A.3). □

The result we want to prove is the generalization of the above, for not necessarily only two matrices involved in the Kronecker products.

Again from the definition of Kronecker product, the term associated with the diagonal correction that is needed, now for a general Kronecker product between diagonal matrices A_1, A_2, ..., A_N, is given by

$$-\text{diag}[(A_1 \otimes ... \otimes A_N)(e_{A_1} \otimes ... \otimes e_{A_N})]. \tag{A.8}$$

We now prove the desired result.

THEOREM 2. *Given N square matrices A_1, ..., A_N, where at most one of them is not diagonal, A_i, one can write*

$$A_1 \otimes ... \otimes A_N - \text{diag}[(A_1 \otimes ... \otimes A_N)(e_{A_1} \otimes ... \otimes e_{A_N})] = A_1 \otimes ... \otimes (A_i - \text{diag}(A_i \times e_{A_i})) \otimes ... \otimes A_N.$$

Proof.

From the definition of Kronecker product, it follows immediately that $A_1 \otimes ... \otimes A_{i-1}$ is a diagonal matrix. The same holds for $A_{i+1} \otimes ... \otimes A_N$.

The Kronecker product of the matrices is thus something of the form

$$D_* \otimes A_i \otimes D_{**},$$

where D_* and D_{**} are diagonal matrices.

We thus need to prove that

$$D_* \otimes A_i \otimes D_{**} - \text{diag}[(D_* \otimes A_i \otimes D_{**})(e_{D_*} \otimes e_{A_i} \otimes e_{D_{**}})] = D_* \otimes (A_i - \text{diag}(A_i \times e_{A_i})) \otimes D_{**}.$$

One just needs to apply Lemma 1 twice. Applying it to the term in the left side, in particular to the non-diagonal matrix $(D_* \otimes A_i)$ and diagonal matrix D_{**}, one gets

$$\{(D_* \otimes A_i) - \text{diag}[(D_* \otimes A_i)(e_{D_*} \otimes e_{A_i})]\} \otimes D_{**}.$$

The lemma is then applied to D_*, diagonal, and A_i, non-diagonal, and the result immediately follows.

□

Note that the last application of Lemma 1 is not direct as we have the diagonal matrix in the first position but the result is clearly also true for this case as the adaptation of the proof is straight-forward.

The proof above assumes that the dimension associated with the non-diagonal matrix is not the first or the last, as it is assumed to have matrices to its left and right, but the proof is similar and even abbreviated for such two cases as one then only needs to apply the lemma once.

REFERENCES

[1] A. Ammar, E. Cueto, and F. Chinesta, *Reduction of the chemical master equation for gene regulatory networks using proper generalized decompositions*, Int. J. Numer. Meth. Biomed. Engng., 28 (2012), pp. 960–973.

[2] David F. Anderson, Gheorghe Craciun, and Thomas G. Kurtz, *Product-form stationary distributions for deficiency zero chemical reaction networks*, Bulletin of Mathematical Biology, 72 (2010), pp. 1947–1970.

[3] Nelson Antunes, Christine Fricker, Philippe Robert, and Danielle Tibi, *Analysis of loss networks with routing*, Annals of Applied Probability, 16 (2006), pp. 2007–2026.

[4] N. Antunes, A. Pacheco, and R. M. Rocha, *A Markov renewal based model for wireless networks*, Queueing Systems, 40 (2002), pp. 247 – 281.

[5] L. A. Belady and C. J. Kuehner, *Dynamic space-sharing in computer systems*, Commun. ACM, 12 (1969), pp. 282–288.

[6] M. Bolten, K. Kahl, and S. Sokolović, *Multigrid methods for tensor structured Markov chains with low rank approximation*, ArXiv e-prints, (2014).

[7] I. Bongartz, A. R. Conn, Nick Gould, and Ph. L. Toint, *Cute: Constrained and unconstrained testing environment*, ACM Trans. Math. Softw., 21 (1995), pp. 123–160.

[8] P. Boyer, A. Dupuis, and A. Khelladi, *A simple model for repeated calls due to time-outs*, 1988.

[9] A. Brandwajn, *A model of a time sharing virtual memory system solved using equivalence and decomposition methods*, Acta Informat., 4 (1974/75), pp. 11–47.

[10] Peter Buchholz, *Structured analysis approaches for large Markov chains*, Appl. Numer. Math., 31 (1999), pp. 375–404.

[11] ———, *An adaptive decomposition approach for the analysis of stochastic petri nets*, in Proceedings of the 2002 International Conference on Dependable Systems and Networks, DSN '02, Washington, DC, USA, 2002, IEEE Computer Society, pp. 647–656.

[12] ———, *Product form approximations for communicating markov processes*, Perform. Eval., 67 (2010), pp. 797–815.

[13] Timothy A. Davis and Yifan Hu, *The university of florida sparse matrix collection*, ACM Trans. Math. Softw., 38 (2011), pp. 1:1–1:25.

[14] Tugrul Dayar, *Analyzing Markov chains using Kronecker products: theory and applications*, Springer Science & Business Media, 2012.

[15] Peter Deuflhard, Wilhelm Huisinga, Tobias Jahnke, and Michael Wulkow, *Adaptive discrete galerkin methods applied to the chemical master equation*, SIAM Journal on Scientific Computing, 30 (2008), pp. 2990–3011.

[16] S. V. Dolgov and D. V. Savostyanov, *Alternating minimal energy methods for linear systems in higher dimensions. Part II: Faster algorithm and application to nonsymmetric systems*, arXiv preprint 1304.1222, 2013.

[17] David L. Donoho, Arian Maleki, Inam Ur Rahman, Morteza Shahram, and Victoria Stodden, *Reproducible research in computational harmonic analysis*, Computing in Science and Engineering, 11 (2009), pp. 8–18.

[18] Jeremy S. Edwards, Markus Covert, and Bernhard Palsson, *Metabolic modelling of microbes: the flux-balance approach*, Environmental microbiology, 4 (2002), pp. 133–140.

[19] Brian P English, Wei Min, Antoine M Van Oijen, Kang Taek Lee, Guobin Luo, Hongye Sun, Binny J Cherayil, SC Kou, and X Sunney Xie, *Ever-fluctuating single enzyme molecules: Michaelis-menten equation revisited*, Nature Chemical Biology, 2 (2005), pp. 87–94.

[20] Jean-Michel Fourneau, *Product form steady-state distribution for stochastic automata networks with domino synchronizations.*, in EPEW, Nigel Thomas and Carlos Juiz, eds., vol. 5261 of Lecture Notes in Computer Science, Springer, 2008, pp. 110–124.

[21] J. M. Fourneau, B. Plateau, and W. J. Stewart, *An algebraic condition for product form in stochastic automata networks without synchronizations*, Perform. Eval., 65 (2008), pp. 854–868.

[22] Timothy S Gardner, Charles R Cantor, and James J Collins, *Construction of a genetic toggle switch in escherichia coli*, Nature, 403 (2000), pp. 339–342.

[23] Michael A Gibson and Jehoshua Bruck, *Efficient exact stochastic simulation of chemical systems with many species and many channels*, The journal of physical chemistry A, 104 (2000), pp. 1876–1889.

[24] Daniel T Gillespie, *A general method for numerically simulating the stochastic time evolution of coupled chemical reactions*, Journal of computational physics, 22 (1976), pp. 403–434.

[25] ———, *Approximate accelerated stochastic simulation of chemically reacting systems*, The Jour-

nal of Chemical Physics, 115 (2001), pp. 1716–1733.
[26] Nicholas I. M. Gould, Dominique Orban, and Philippe L. Toint, *Cuter and sifdec: A constrained and unconstrained testing environment, revisited*, ACM Trans. Math. Softw., 29 (2003), pp. 373–394.
[27] L. Grasedyck, D. Kressner, and C. Tobler, *A literature survey of low-rank tensor approximation techniques*, GAMM-Mitt., 36 (2013), pp. 53–78.
[28] Markus Hegland, Conrad Burden, Lucia Santoso, Shev MacNamara, and Hilary Booth, *A solver for the stochastic master equation applied to gene regulatory networks*, J. Comput. Appl. Math., 205 (2007), pp. 708–724.
[29] Nicholas J. Higham, *Algorithm 694: A collection of test matrices in matlab*, ACM Trans. Math. Softw., 17 (1991), pp. 289–305.
[30] Vladimir Kazeev, Mustafa Khammash, Michael Nip, and Christoph Schwab, *Direct solution of the chemical master equation using quantized tensor trains*, PLOS Computational Biology, (2014).
[31] Vladimir A. Kazeev and Boris N. Khoromskij, *Low-rank explicit QTT representation of the Laplace operator and its inverse*, SIAM J. Matrix Anal. Appl., 33 (2012), pp. 742–758.
[32] Daniel Kressner and Francisco Macedo, *Low-rank tensor methods for communicating markov processes*, in Quantitative Evaluation of Systems, Gethin Norman and William Sanders, eds., vol. 8657 of Lecture Notes in Computer Science, Springer International Publishing, 2014, pp. 25–40.
[33] V.G. Kulkarni, *Modeling and Analysis of Stochastic Systems*, Chapman & Hall/CRC Texts in Statistical Science, Taylor & Francis, 1996.
[34] V. G. Kulkarni, *Introduction to modeling and analysis of stochastic systems*, Springer Texts in Statistics, Springer, New York, second ed., 2011.
[35] Amy N. Langville and William J. Stewart, *The kronecker product and stochastic automata networks*, Journal of Computational and Applied Mathematics, 167 (2004), pp. 429 – 447.
[36] Erel Levine and Terence Hwa, *Stochastic fluctuations in metabolic pathways*, Proceedings of the National Academy of Sciences, 104 (2007), pp. 9224–9229.
[37] Gerhard Michal, *Biochemical Pathways: An Atlas of Biochemistry and Molecular Biology*, Wiley-Spektrum, 1 ed., Dec. 1998.
[38] Ian M. Mitchell, Randall J. LeVeque, and Victoria Stodden, *Reproducible research for scientific computing: Tools and strategies for changing the culture*, Computing in Science and Engineering, 14 (2012), pp. 13–17.
[39] Debasis Mitra and Isi Mitrani, *Analysis of a kanban discipline for cell coordination in production lines: Ii. stochastic demands*, Oper. Res., 39 (1991), pp. 807–823.
[40] Brian Munsky and Mustafa Khammash, *The finite state projection approach for the analysis of stochastic noise in gene networks*, Automatic Control, IEEE Transactions on, 53 (2008), pp. 201–214.
[41] I. V. Oseledets, *TT-Toolbox Version 2.2*, 2012. Available at `https://github.com/oseledets/TT-Toolbox`.
[42] Bernard Philippe, Youcef Saad, and William J. Stewart, *Numerical methods in markov chain modelling*, Operations Research, 40 (1996), pp. 1156–1179.
[43] Brigitte Plateau, Jean-Michel Fourneau, and Kuei-Hsiang Lee, *Peps: A package for solving complex markov models of parallel systems*, in Modeling Techniques and Tools for Computer Performance Evaluation, Ramon Puigjaner and Dominique Potier, eds., Springer US, 1989, pp. 291–305.
[44] N.U. Prabhu, *Foundations of Queueing Theory*, International Series in Operations Research & Management Science, Springer US, 1997.
[45] Sheldon M. Ross, *Introduction to probability models*, Harcourt/Academic Press, Burlington, MA, seventh ed., 2000.
[46] Jerome H. Saltzer, *A simple linear model of demand paging performance*, Commun. ACM, 17 (1974), pp. 181–186.
[47] M. Samoilov, S. Plyasunov, and A. P. Arkin, *Stochastic amplification and signaling in enzymatic futile cycles through noise-induced bistability with oscillations*, Proc. Natl. Acad. Sci. U.SA., 102 (2005), pp. 2310–2315.
[48] Michael Samoilov, Sergey Plyasunov, and Adam P. Arkin, *Stochastic amplification and signaling in enzymatic futile cycles through noise-induced bistability with oscillations*, Proceedings of the National Academy of Sciences of the United States of America, 102 (2005), pp. 2310–2315.
[49] Jörg Schwender, John Ohlrogge, and Yair Shachar-Hill, *Understanding flux in plant metabolic networks*, Current Opinion in Plant Biology, 7 (2004), pp. 309–317.
[50] Moshe Sidi and David Starobinski, *New call blocking versus handoff blocking in cellular*

networks, Wirel. Netw., 3 (1997), pp. 15–27.
[51] Paul Sjöberg, Per Lötstedt, and Johan Elf, *Fokker–planck approximation of the master equation in molecular biology*, Computing and Visualization in Science, 12 (2009), pp. 37–50.
[52] Victoria Stodden, Friedrich Leisch, and Roger D. Peng, eds., *Implementing Reproducible Research*, The R Series, Chapman and Hall/CRC, Apr. 2014.
[53] X Sunney Xie and H Peter Lu, *Single-molecule enzymology*, Journal of Biological Chemistry, 274 (1999), pp. 15967–15970.

Recent publications:

MATHEMATICS INSTITUTE OF COMPUTATIONAL SCIENCE AND ENGINEERING
Section of Mathematics
Ecole Polytechnique Fédérale
CH-1015 Lausanne

14.2015 ASSYR ABDULLE, TIMOTHÉE POUCHON:
A priori error analysis of the finite element heterogenenous multiscale method for the wave equation in heterogenenous media over long time

15.2015 ANDREA MANZONI, STEFANO PAGANI:
A certified reduced basis method for PDE-constrained parametric optimization problems by an adjoint-based approach

16.2015 SIMONE DEPARIS, DAVIDE FORTI, ALFIO QUARTERONI:
A fluid-structure interaction algorithm using radial basis function interpolation between non-conforming interfaces

17.2015 ASSYR ABDULLE, ONDREJ BUDAC:
A reduced basis finite element heterogeneous multiscale method for Stokes flow in porous media

18.2015 DANIEL KRESSNER, MICHAEL STEINLECHNER, BART VANDEREYCKEN:
Preconditioned low-rank Riemannian optimization for linear systems with tensor product structure

19.2015 ALESSANDRO S. PATELLI, LUCA DEDÈ, TONI LASSILA, ANDREA BARTEZZAGHI, ALFIO QUARTERONI:
Isogeometric approximation of cardiac electrophysiology models on surfaces: an accuracy study with application to the human left atrium

20.2015 MATTHIEU WILHELM, LUCA DEDÈ, LAURA M. SANGALLI, PIERRE WILHELM:
IGS: an IsoGeometric approach for Smoothing on surfaces

21.2015 SIMONE DEPARIS, DAVIDE FORTI, PAOLA GERVASIO, ALFIO QUARTERONI:
INTERNODES: an accurate interpolation-based method for coupling the Galerkin solutions of PDEs on subdomains featuring non-conforming interfaces

22.2015 ABDUL-LATEEF HAJI-ALI, FABIO NOBILE, LORENZO TAMELLINI, RAÙL TEMPONE:
Multi-index stochastic collocation for random PDEs

23.2015 SIMONE BRUGIAPAGLIA, FABIO NOBILE, STEFANO MICHELETTI, SIMONA PEROTTO:
A theoretical study of COmpRessed SolvING for advection-diffusion-reaction problems

24.2015 ANA ŠUŠNJARA, NATHANAËL PERRAUDIN, DANIEL KRESSNER, PIERRE VANDERGHEYNST:
Accelerate filtering on graphs using Lanczos mehtod

25.2015 FRANCESCO BALLARIN, ELENA FAGGIANO, SONIA IPPOLITO, ANDREA MANZONI, ALFIO QUARTERONI, GIANLUIGI ROZZA, ROBERTO SCROFANI:
Fast simulation of patient-specific haemodynamics of coronary artery bypass grafts based on a Pod-Galerkin method and a vascular shape parametrization

26.2015 FRANCISCO MACEDO:
Benchmark problems on stochastic automata networks in tensor train format