bioRxiv
BIORXIV
Advance Access Publication Date: Day Month Year
Manuscript Category

Systems Biology

Dynamic Flux Balance Analysis Models in SBML

Leandro H. Watanabe [1,†] Matthias König [2,†] and Chris J. Myers [1,*]

[1] Department of Electrical and Computer Engineering, University of Utah, Salt Lake City, 84112, USA
[2] Humboldt-University Berlin, Institute for Theoretical Biology, Institute for Biology, Invalidenstraße 43, 10115 Berlin, Germany

* To whom correspondence should be addressed.
† Equal contribution.

Abstract

Motivation: Systems biology models are typically simulated using a single formalism such as ordinary differential equations (ODE) or stochastic methods. However, more complex models require the coupling of multiple formalisms since different biological concepts are better described using different methods, e.g., stationary metabolism is often modeled using flux-balance analysis (FBA) whereas dynamic changes of model components are better described via ODEs. The coupling of FBA and ODE frameworks results in dynamic FBA models. A major challenge is how to describe such hybrid models coupling multiple frameworks in a standardized way, so that they can be exchanged between tools and simulated consistently and in a reproducible manner.
Results: This paper presents a scheme and implementation for encoding dynamic FBA models in the Systems Biology Markup Language (SBML), thereby allowing to exchange multi-framework computational models between software tools. The paper shows the feasibility of the approach using various example models and demonstrates that different tools are able to simulate the hybrid models and agree on the results. As part of this work, two independent implementations of a multi-framework simulation method for dynamic FBA have been developed supporting such models: `iBioSim` and `sbmlutils`.
Availability: All materials and models are available from https://github.com/matthiaskoenig/dfba. The tools used in this project are freely available: `iBioSim` at http://www.async.ece.utah.edu/ibiosim and `sbmlutils` at https://github.com/matthiaskoenig/sbmlutils/.
Contact: myers@ece.utah.edu

1 Introduction

In systems biology, mathematical modeling has been widely used to describe biological systems (Kitano, 2002). The resulting computational models can be simulated and analyzed *in silico* and allow researchers to make predictions which subsequently can be validated experimentally. Furthermore, such models can provide insights in biological systems that would be difficult to obtain in a wet lab. A key challenge, however, is ensuring that these modeling efforts are reproducible and easily exchanged between research groups such that and results can be validated and existing models can be reused to build more complex models. To achieve these goals, standard model representation formats for the model exchange, such as the Systems Biology Markup Language (SBML) (Hucka *et al.*, 2003) or CellML (Hedley *et al.*, 2001), have been established. Both SBML and CellML have been successfully applied to the encoding of models using a single modeling framework, but the support of multiple framework adds new challenges. This paper addresses this problem by developing a methodology and corresponding implementations to support such hybrid modeling efforts, and it demonstrates the successful exchange and reproducibility of such models between two simulation tools.

1.1 Multi-framework computational models

Various simulation and analysis methods have been developed in systems biology, and depending on the biological question different methods are preferred. Kinetic time-course simulation based on ordinary differential equations (ODE) is often employed to observe the dynamics of the entities in a model over time. Depending on the research question and biological system, such simulations can be either deterministic or non-deterministic (stochastic). Other simulation frameworks are boolean (Thomas, 1973; Kauffman, 1969) models, logical models (Morris *et al.*, 2010) and constraint-based approaches (Bordbar *et al.*, 2014), among others.

Metabolic networks, in particular, are often challenging to model dynamically using ODE approaches because kinetic parameters needed for ODE models are often unavailable (Varma and Palsson, 1994). Hence, steady-state approaches that do not need kinetic information are employed to model metabolism, so called *flux balance analysis* (FBA) (Savinell and Palsson, 1992; Varma *et al.*, 1993) based on constraint-based optimization

© The Author 2017.

assuming steady state. This method only requires the connectivity of the reactions and metabolites along with the respective stoichiometry, an objective function (e.g. cell growth), and additional constraints like flux bounds. The idea is to constrain the model based on the stoichiometry of the reactions and optimize the objective function while satisfying the flux constraints. The advantages of using such method include its efficiency and not requiring any kinetic information.

Biological research questions often require the coupling of different model formalisms. One such recent example is the whole-cell model for the *Mycoplasma genitalium* (Karr *et al.*, 2012) that is encoded using a mixture of boolean networks, stochastic processes, differential equations, and FBA.

1.2 Dynamic flux balance analysis

One disadvantage of FBA is that it cannot express the dynamics of the metabolites since it does not change amounts or concentrations of species, but only provides information about the optimal flux distribution for the given optimization problem. Due to this limitation, the field of *dynamic FBA* (DFBA) (Varma and Palsson, 1994) has emerged, which couples the stationary flux distribution resulting from FBA with the kinetic update of the metabolites taken up or consumed by the FBA network, i.e., the FBA submodel is coupled to a kinetic model (ODE) via a multi-framework approach.

Besides the whole-cell model which uses DFBA as a core module, many DFBA models have been constructed for different metabolic pathways. DFBA has been applied in small-scale examples (Varma and Palsson, 1994; Mahadevan *et al.*, 2002; Luo *et al.*, 2006), over medium-size models (Pizarro *et al.*, 2007; Lequeux *et al.*, 2010; Meadows *et al.*, 2010), and up to genome-scale DFBA applications (Hanly and Henson, 2011; Hjersted *et al.*, 2007). For a recent overview, see Table 1 in (Höffner *et al.*, 2013).

The coupling between FBA and kinetic model parts has hereby be implemented via three main approaches, i.e., *static optimization approach* (SOA), *dynamic optimization approach* (DOA), or *direct approach* (DA) (Gomez *et al.*, 2014). DOA approaches discretize the simulation time and optimize simultaneously over the entire time period by solving a nonlinear programming problem (NLP). The DA approach directly includes the LP solver in the right-hand side of the ordinary differential equations (ODEs). The SOA approach solves the LP at each time step using a Euler forward method assuming constant fluxes over the time step (Gomez *et al.*, 2014). Most of the published DFBA models use the SOA approach, which is relatively simple to implement and not as computationally demanding (see methods algorithm).

1.3 Exchangeability & reproducibility of models

Despite the multitude of published DFBA models, currently no standard for the exchange of such models exists. Existing models are hard-coded in programming code, e.g., the whole-cell model in MATLAB. Hereby, the mathematical models in their respective formalisms are embedded in the script along with the connections between the kinetic and flux balance parts of the models. As a consequence, it is not possible to exchange existing DFBA models between different software tools. Thus, they cannot be reproduced or validated. This is especially problematic in the case of DFBA models because often multiple optima can exist for the FBA model part (and the various time steps), and the resulting DFBA solutions are not unique, but depend on the actual implementation, i.e., how an implementation or solver selects one of the possible solutions. In addition solutions can depend on the selected step size in SOA if the step size is not small enough.

While it is possible to replicate the same scripts in different programming languages, it is unpractical to do so as replication is error prone, requires unnecessary work, needs conversions that can lead to data loss, and most importantly does not solve the underlying problem of exchangability of such models. For these reasons, script replication makes achieving reproducibility difficult and often infeasible. The necessity of an exchange format for DFBA resulted from efforts trying to encode and reproduce the DFBA submodel of the whole-cell model using standards during the whole-cell workshop (Waltemath *et al.*, 2016).

1.4 Model standards

In order to achieve exchangeability and reproducibility of models, standards for the encoding of models have been created. The *de-facto* standard for systems biology models is SBML (Hucka *et al.*, 2003). SBML core elements are used to describe mathematical models of reaction-based networks and provide the means to encode computational models based on ODEs (deterministic and stochastic). SBML uses packages for extending the functionality of the core elements. While SBML is used to encode mathematical models of biological networks, there are different standards for other purposes: the *Simulation Experiment Description Markup Language* (SED-ML) is used for describing simulations (Waltemath *et al.*, 2011), the *Systems Biology Graphical Notation* (SBGN) is used for describing visualizations (Le Novere *et al.*, 2009), and COMBINE Archives are used for exchanging collections of modeling files (Bergmann *et al.*, 2014). The main advantage of using these standards over hard-coding models in code is the ability to exchange models between research groups and reproduce results using various tools that support these standards.

In this work SBML core in combination with the *hierarchical model composition* (comp) package (Smith *et al.*, 2015) and the *flux balance constraints* (fbc) package (Olivier and Bergmann, 2015) is used for describing the multi-framework DFBA models. The comp package is used to construct hierarchical models, providing the means to build built models from submodels and define the interfaces between them. The fbc package is used to encode the FBA submodel consisting of the metabolic network, the flux bounds for the reactions, and an objective function, allowing to perform FBA. In addition, SED-ML is used to describe how each SBML model should be simulated, i.e., provide reproducible example simulation experiments by encoding which simulation algorithm to use and its corresponding parameters, as well as the defining the time course simulations for the DFBA. COMBINE archives are used for the exchange of the encoded models, simulation descriptions and reference solutions.

One of the challenges in current SBML models is the limitation on the expression of models using different formalisms. Although there are several tools that support ODE simulation and FBA, they all support them independently. In order to overcome this challenge, this paper introduces a scheme that allows the coupling of ODE and FBA models. This paper demonstrates that this scheme provides exchangeability and reproducibility by encoding and simulating DFBA models in both `iBioSim` (Madsen *et al.*, 2012) and `sbmlutils` (König, 2017).

2 Methods

2.1 Model encoding

The DFBA models presented in this paper were created in the proposed scheme either using a graphical user interface in `iBioSim` or a script-based approach in `sbmlutils`. For a given model, the TOP, FBA, BOUNDS, and UPDATE submodels were packaged with respective simulation files using SED-ML in COMBINE archives for the exchange between tools. All models and simulation results are available from https://github.com/matthiaskoenig/dfba.

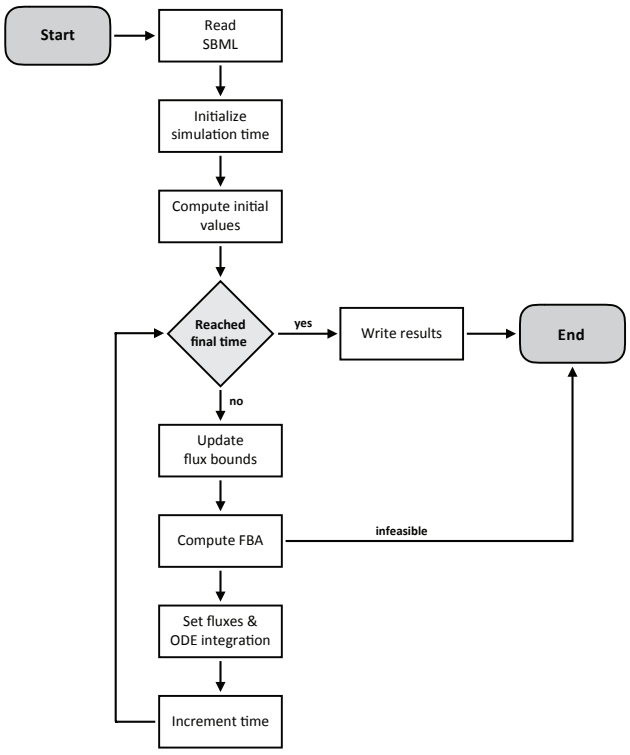


Fig. 1: Overview of the implemented SOA algorithm for DFBA. After initialization of the model, the FBA and kinetic simulations are run in an iterative manner until the simulation end point. In every step, FBA is used to compute the reaction rates of the FBA network. Subsequently, based on the computed FBA rates, the values of the species are updated dynamically. In the SOA approach, FBA fluxes are assumed to be constant within a time step. For a detailed description see the methods section.

2.2 Stationary optimization approach (SOA)

A stationary optimization approach for DFBA was implemented as a simulation algorithm in `iBioSim` and `sbmlutils` following the simulation scheme depicted in Figure 1.

The first step is the initialization of the model. All of the species and parameters in the model are initialized, where each variable's initial value is computed. After the initialization step, the FBA submodel is executed. During the FBA step, reaction fluxes are computed using the initial flux bound values where the flux bounds for the reactions come from the top-level using SBML comp *replacements*. In SBML, replacements of parameters and species indicate the top-level entities are the same entity as the one being replaced. Once the fluxes are computed, they are assigned to parameters using assignment rules on the top-level. These parameters are assigned reaction rates computed as functions of the fluxes.

After computing reaction fluxes, the update step is performed concurrently with the dynamic step by computing the time-evolution of every species in the UPDATE and KINETIC submodels. Species that affect any flux bound in the FBA submodel are updated in the top-level. The new bounds are used in the FBA submodel for the next time step. Simulation time is incremented at the end. If the time limit is reached, then simulation is complete. Otherwise, all of the steps are repeated.

The SOA simulation algorithm has been implemented in `iBioSim` and `sbmlutils`. The `iBioSim` tool uses the structure of (Watanabe and Myers, 2014) for simulation. The `sbmlutils` tool uses *roadrunner* (Somogyi *et al.*, 2015) for the kinetic simulation and *cobrapy* (Ebrahim *et al.*, 2013) to solve the FBA problem. Both `iBioSim` and `sbmlutils` take an SBML file that describes a DFBA model and a SED-ML file that describes the simulation experiment. In the proposed approach, SED-ML is mainly used to indicate which simulation algorithm to use, the time points in which tools should print out the values of the variables, the initial time and the time limit. The SED-ML files provide a minimal simulation experiment to check reproducibility between implementations. The value of each time increment for SOA is defined as a parameter with id dt in the SBML model, which can be overwritten by the SED-ML file for the actual simulation. Ontology terms for the description of DFBA simulation algorithms have been introduced in the Kinetic Simulation Algorithm Ontology (KISAO) (Zhukova *et al.*, 2011) and are used in the SED-ML descriptions, i.e., `KISAO:0000500` (SOA-DFBA).

2.3 Reproducibility between tools

In order to test interoperability based on the proposed scheme, models were built in both the `iBioSim` and the `sbmlutils` tools. Models built in `iBioSim` were then imported into `sbmlutils` and vice-versa to check whether models could be interpreted by both tools consistently. This was done in an iterative manner and resulting issues were solved by clarifying the encoding scheme, e.g., by adding additional rules which resolved ambiguities.

Reproducibility of DFBA models is challenging because there may exist several possible outcomes that satisfy the objective function and constraints of the FBA models. Depending on how a solver and implementation selects one of the multiple optima different trajectories can result from the DFBA simulation. The issue of multiple optima was solved by guaranteeing uniqueness of the solution in every time step based on *Flux Variability Analysis* (FVA) (Mahadevan and Schilling, 2003). FVA gives the possible minimal and maximal fluxes for each reaction in each step of the simulation. If all minimal fluxes are equal to all maximal fluxes for a time point a solution is unique in the time point. If all time points are unique the solution is unique. As a practical note: If the solution is not unique, the addition of additional constraints to the FBA problem allows to make the solution unique.

Reproducibility of the model simulations was tested by comparing the numerical SOA results between the two tools for models with unique solutions (see Supplementary Material S2). Results were assumed as numerical identical if the absolute difference for every time point t_k for all dynamical FBA species in the model c_k was smaller than the tolerance $\epsilon = 1\text{E}{-5}$, i.e.,

$$abs(c_i(t_k)_{sbmlutils} - c_i(t_k)_{ibiosim}) \leq \epsilon \; \forall c_i, t_k$$

In SOA-DFBA it is important that the time steps dt are small enough so that the solution converges against the correct solution, and solutions vary if selected step sizes are too large (e.g. changing the step size in the `toy_wholecell` model from 1.0 to 0.1 resulted in differences in steady state concentrations of up to 10%). Consequently, different step sizes were tested for the models and step size of the simulations were selected, so that smaller step sizes did not change the simulation results.

3 Results

The major result of this work is the creation of the first schema for encoding DFBA in SBML, demonstrating multi-framework computational models to be exchanged and reproduced between tools. In the following the schema and its application to multiple DFBA models is presented.

3.1 Schema for dynamic flux balance analysis

This paper proposes for the first time a schema to encode hybrid models, such as DFBA models, in SBML. The developed schema consisting of rules, guidelines, and additional information is available as Supplementary

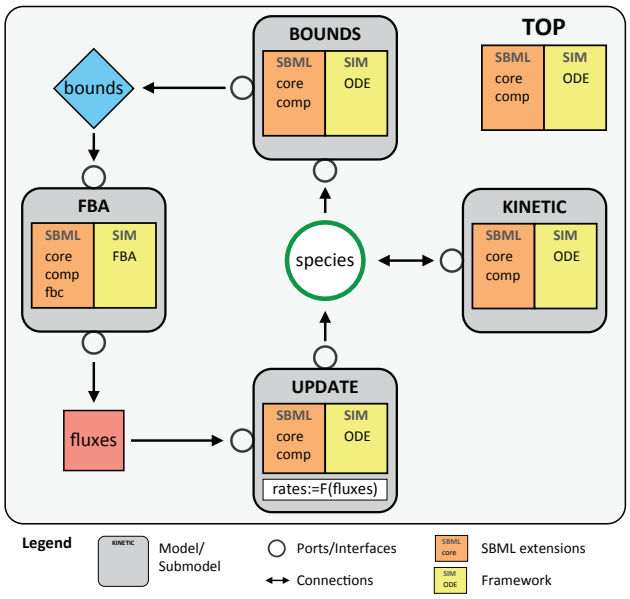


Fig. 2: Overview of schema for encoding DFBA models in SBML. The hierarchical SBML model is composed of a top-level model with four submodels: FBA, BOUNDS, UPDATE, and KINETIC. The individual submodels are connected via ports. The respective SBML packages used are listed in the models, as well as the simulation framework used. The BOUNDS submodel calculates the upper and lower flux bounds based on metabolite availability. The FBA submodel computes the reaction fluxes of the metabolic fbc model using the bounds as constraints. The UPDATE submodel calculates the dynamic update of the dynamic metabolites affected by the FBA model. The rates of change are herby functions of the FBA fluxes. The KINETIC submodel includes all of the other processes in the model, which may affect or be affected by entities in metabolism. The top-level model ties together the different submodels using SBML comp replacements and replacedBy constructs.

Material S1. The latest version of the document is available from https://github.com/matthiaskoenig/dfba/. Proposals, errata, and updates to the schema are managed via the respective issue tracker and releases.

In this section we provide a high-level overview over the underlying concepts used in the schema, followed by application of the schema to encode DFBA models.

The DFBA model is constructed hierarchically using the SBML comp package, separating the hybrid model in different building blocks based on the respective functionality and modeling frameworks (Figure 2). The top-level model is hereby composed of four submodels: (i) a kinetic submodel that computes flux bounds based on the dynamic metabolite availability and ensures that the FBA problem is constrained by the available metabolite resources (BOUNDS submodel); (ii) a FBA submodel that encodes metabolism as a FBA problem (FBA submodel); and (iii) a kinetic submodel that updates the amounts and concentrations of the dynamic metabolites changed via the FBA submodel via consumption or production (UPDATE submodel); (iv) an optional kinetic submodel that represents a dynamic part with all kinetics other than the metabolic pathway, such as DNA transcription, DNA translation, and protein degradation, among others (KINETIC submodel). Alternatively, arbitrary kinetics can be part of the top model.

The top-level model ties together the three different submodels using SBML comp replacements and replacedBy constructs with the interface between the submodels defined via comp ports (which define which model components of the submodels can be connected, i.e, are exposed).

In order to describe the different formalisms of each submodel, the *Systems Biology Ontology* (SBO) is used (Courtot *et al.*, 2011). The SBO defines controlled vocabulary terms used in the systems biology field. The SBO terms are arranged in a taxonomic hierarchy using a tree structure. This allows the grouping of terms that are related to one another. The modeling formalisms of the individual submodels are described using terms on the *modeling framework* branch, where FBA models are described using the *flux balance framework* term, stochastic processes are described using the *non-spatial discrete framework* term, and differential equations are described using the *non-spatial discrete framework* term. Although the terms for stochastic processes and differential equations can be used for describing either stochastic or deterministic simulation methods, these terms were selected because they are the ones that best describes these two formalisms.

In addition to the modeling formalism other key components are annotated in the submodels via SBO terms in the schema, e.g., the upper and lower flux bounds and the exchange reactions in the FBA submodel defining which metabolites can be consumed or produced in the FBA part of the DFBA, or the dynamic species in the top model changed by the FBA submodel. By the means of these annotations the interface between the hybrid submodels can be clearly defined.

All of the interconnections between the submodels are encoded in SBML rather than using an external approach like for instance via SED-ML. The connections between model components are crucial information of the model and should be part of the model encoding. SED-ML is only used to encode which simulation to run with the model. As a consequence, this schema requires only a single hierarchical SBML model and a single SED-ML file.

3.2 Minimal Example (`toy_wholecell`)

In order to illustrate the proposed schema, a simplified example of a whole-cell model was created with a model overview depicted in Figure 3. This figure shows how the different submodels connect with each other in a flat form. The model is available as COMBINE archive in Supplementary Material S3, the Cytoscape visualization as Supplementary Material S4.

This model is constructed hierarchically where a top-level model is created to instantiate different submodels (BOUNDS, UPDATE, and FBA) and make the necessary connections between them. The figure illustrates the structure of each submodel and how each submodel ties in with each other in a flat version of the model once all of the connections are established.

In the example the FBA submodel imports species `A` and convert it via a linear chain of reactions to species `C`. The exchange reactions `EX_A` and `EX_C` contain the rate of consumption and production of the respective species. The TOP model contains assignment rules which assign the fluxes to the parameters `pEX_A` and `pEX_C`, which are used by the UPDATE model to update the dynamic species `A` and `C` via the update reactions `update_A` and `update_C`. The BOUNDS model calculates the bounds of all FBA exchange reactions, i.e., constraining by the availability of the dynamic species, as well as bounds changed by kinetic expressions. In the example the upper bound `ub_R1` of reaction `R1` is changed via a rate rule. Additional kinetics are encoded in the TOP model, i.e., a kinetic conversion of `C` to `C` (these could also be in a separate KINETIC submodel).

In order to validate the exchangability and reproducibility of the model, simulations were performed using the simulation algorithm described in Figure 1 with results depicted in Figure 4. Both implementations resulted in numerically identical results (see 2.3). Importantly, our encoding schema allowed to reproduce the numerical results even if the step sizes were not

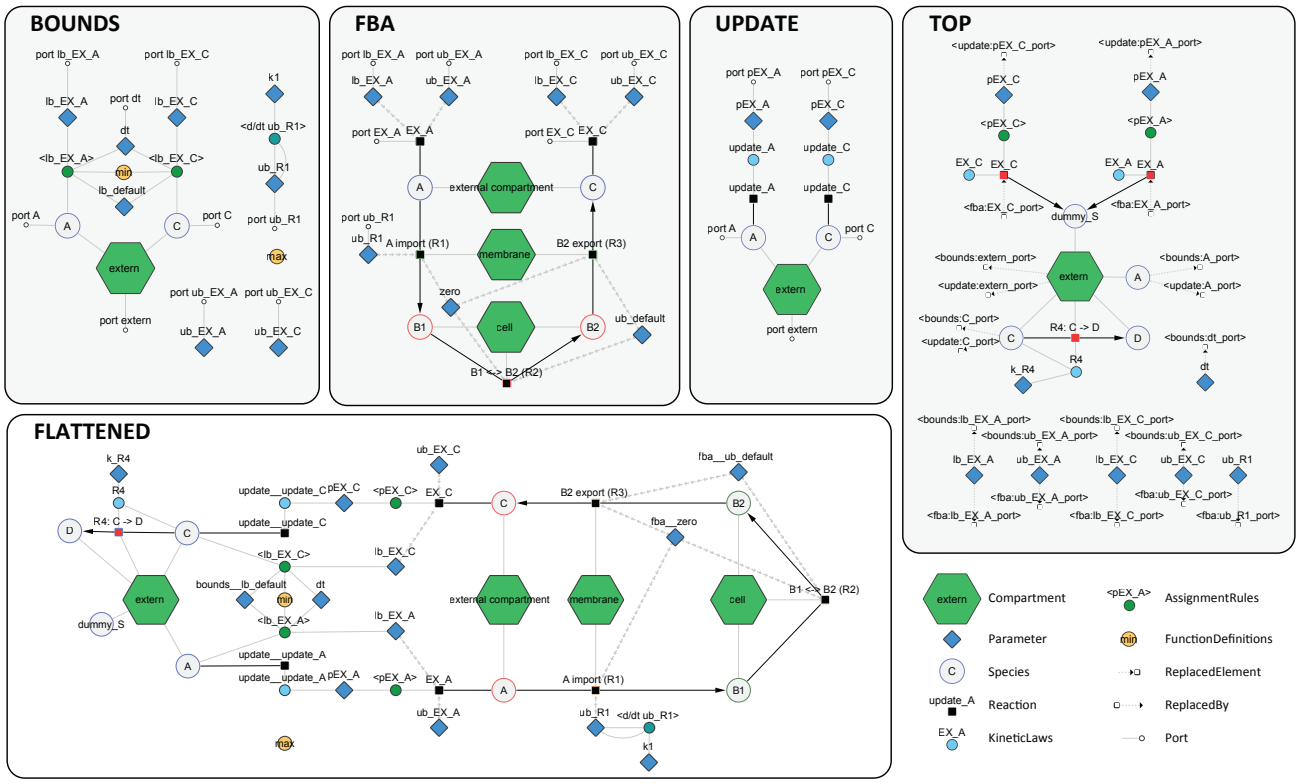
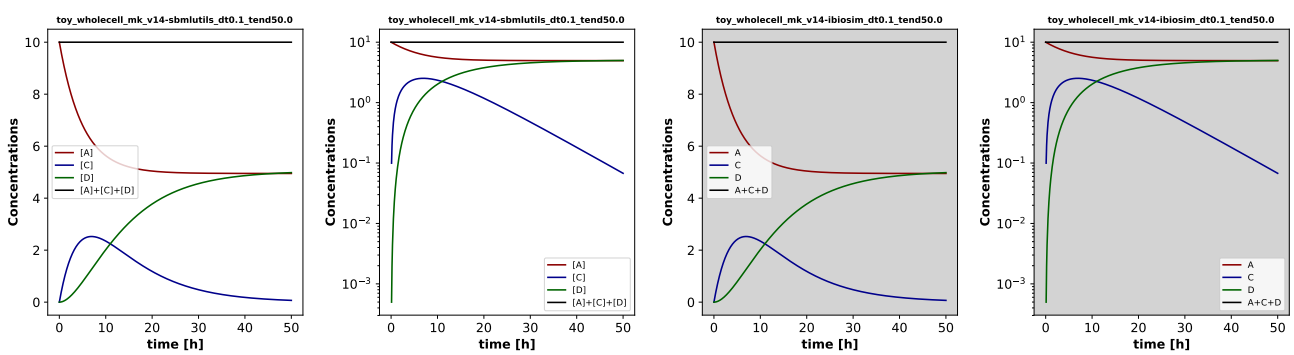


Fig. 3: Detailed schema of the minimal example model (`toy_wholecell`). The figure shows the components in the BOUNDS, FBA and UPDATE submodels. Links between submodel components are based on ports which are connected elements via TOP model replacements (replacedElements and replacedBy). The flattened SBML comp model (FLATTENED) shows the resolved connections between the different submodels after these replacements have been performed. The flattened model can not be simulated because the separation of frameworks is lost in the flattening process. The network visualization are available as interactive graphs in Cytoscape as Supplementary Material S4, which provide additional information and annotation of the components. The figure was created with cy3sbml using the SBML models (König *et al.*, 2012).



(a) Simulation results with `sbmlutils`

(b) Simulation results with `iBioSim`

Fig. 4: DFBA Simulation results for the `toy_wholecell` model in two different tools. This demonstrates that models can be exchanged by different tools using standards and the results can be reproduced when using the same simulation algorithm. Species `A` is converted to `C` via the FBA subnetwork over time. `C` is converted to `D` via the kinetic parts in the top model. Species `A` is not consumed completely because the import of `A` in the FBA subnetwork via `R1` is shut down via a rate rule for the upper flux bound, and a steady state is reached. The model was simulated for 50[h] with a time step `dt` of 0.1[h].

yet small enough to have converged against the correct solution, thereby allowing to test the effects of varying step sizes in a reproducible manner.

In addition to the presented minimal model, a second model of a simplified DFBA glycolysis (`toy_atp`) is available in the supplement (COMBINE archive in Supplementary Material S5, corresponding Cytoscape visualization in Supplementary Material S6).

3.3 Diauxic growth in *E. coli* (`diauxic_growth`)

The next example is an encoding and reproduction of results from a published DFBA model of diauxic growth of *E. coli* (Mahadevan *et al.*, 2002) consisting of four reactions between four metabolites, i.e., glucose ($Glcxt$), oxygen (O_2), acetate (A_c) and biomass (X). The model can

grow either aerobically on acetate ($v1$), aerobically on glucose ($v2$ or $v3$) or anaerobically convert glucose to acetate:

$$v1 : 39.43A_c + 35O_2 \rightarrow X$$
$$v2 : 9.46Glcxt + 12.92O_2 \rightarrow X$$
$$v3 : 9.84Glcxt + 12.73O_2 \rightarrow 1.24A_c + X$$
$$v4 : 19.23Glcxt \rightarrow 12.12A_c + X$$

The kinetic part of the model is described by the following differential equations:

$$\frac{dGlcxt}{dt} = A^{Glcxt}\nu X$$
$$\frac{dA_c}{dt} = A^{A_c}\nu X$$
$$\frac{dO_2}{dt} = A^{O_2}\nu X + k_L a(0.21 - O_2)$$
$$\frac{dX}{dt} = (v1 + v2 + v3 + v4)X$$

where A^{Glcxt}, A^{A_c}, A^{O_2} are the respective rows of each variable in the stoichiometry matrix and $k_L a$ is the mass transfer coefficient of oxygen. For a detailed description see (Mahadevan *et al.*, 2002).

The model is available in Supplementary Material S7, the Cytoscape visualization in Supplementary Material S8.

The results in Figure 5 depict an exponential growth phase using glucose aerobically until running out of glucose, which at this point the cell grows linearly due to oxygen. When both oxygen and glucose run out, the cell growth stagnates. Experimental data from (Varma and Palsson, 1994) is plotted alongside the simulation results. The model is able to capture the behavior observed in the experimental data. The results are equivalent to the models in (Mahadevan *et al.*, 2002).

We hereby showed that our schema is able to encode published DFBA models, resulting in a reproducible and exchangeable model representation between tools.

3.4 *E. coli* core (`ecoli`)

To demonstrate the feasibility of the proposed schema and method for real-world examples of DFBAs, a larger metabolic network for the core metabolism of *E. coli* (Orth *et al.*, 2010) was encoded in the proposed schema and simulated as shown in Figure 6. The model is available as COMBINE archive in Supplementary Material S9. The FBA submodel was downloaded from BiGG (King *et al.*, 2015) (core metabolism of Escherichia coli str. K-12 substr. MG1655) and transformed to an DFBA model in an automatic fashion using `sbmlutils`. BiGG models encode the exchangeable species via annotated exchange reactions which allows and automatic inference of the dynamic species. Only additional information required to run a DFBA simulations are initial concentrations for the species. The automatic encoding of larger scale examples demonstrates the scalability of the proposed encoding approach.

While `sbmlutils` is able to find a solution for the model, `iBioSim` cannot as it runs into an unfeasible solution in the middle of simulation. This captures the well-known problem of DFBA with multiple solutions. The FBA problem is not constrained enough to result in a unique solution and depending on which solution the simulator picks, different solutions and thereby trajectories arise. Despite the existence of multiple solutions, tools and LP solver typically pick solutions deterministically. Hence, single tools can reproduce their own results, but results are irreproducible between different implementation. Without the use of standards, this could never be demonstrated because variations in results could be due to discrepancies in the model, and not in the tool.

4 Discussion

Modularity of models, the ability to encode multi-framework models, and reproducibility of models is indispensable for encoding more complex models in computational biology. In this work we presented such an approach, which allows a clear separation of the different modeling frameworks via comp submodels and defining the interfaces between the submodels. To our knowledge, this paper proposes and implements for the first time an exchangeable and reproducible multi-framework scheme. This scheme for encoding DFBA models in a standard way has been implemented in two different tools, demonstrating the exchangeability and reproducibility of our approach on various examples models like diauxic growth in *E.coli*. `iBioSim` and `sbmlUtils` are freely available for download and offer the necessary infrastructure for anyone to develop DFBA models using the proposed scheme.

Currently, the proposed approach supports the modeling of DFBA models based on the SOA simulation algorithm. Hence, our approach only covers a subset of DFBA algorithms and possible frameworks which could be coupled.

Most DFBA models are stiff and small time steps are required for stability, making the SOA approach computationally expensive. Another disadvantage of the SOA approach is that it requires a sufficiently small fixed time step to give accurate results. Future directions include the exploration of adaptive time steps for executing the DFBA with SOA, alternative DFBA methods, such as DOA or DA, and extending our scheme to encode such models.

Our current is limited to the coupling of ODE and FBA frameworks. Different types of hybrid model, such as a mixture of differential equations, stochastic processes, and boolean models still need further study. The proposed approach of decoupling frameworks via hierarchical model composition could work similarly for other modeling frameworks like boolean models.

So far, only small to medium-size DFBA models have been encoded in our proposed approach. For future work, we will encode genome-scale metabolic models. This would allow us to assess the scalability and performance of the proposed approach.

Acknowledgements

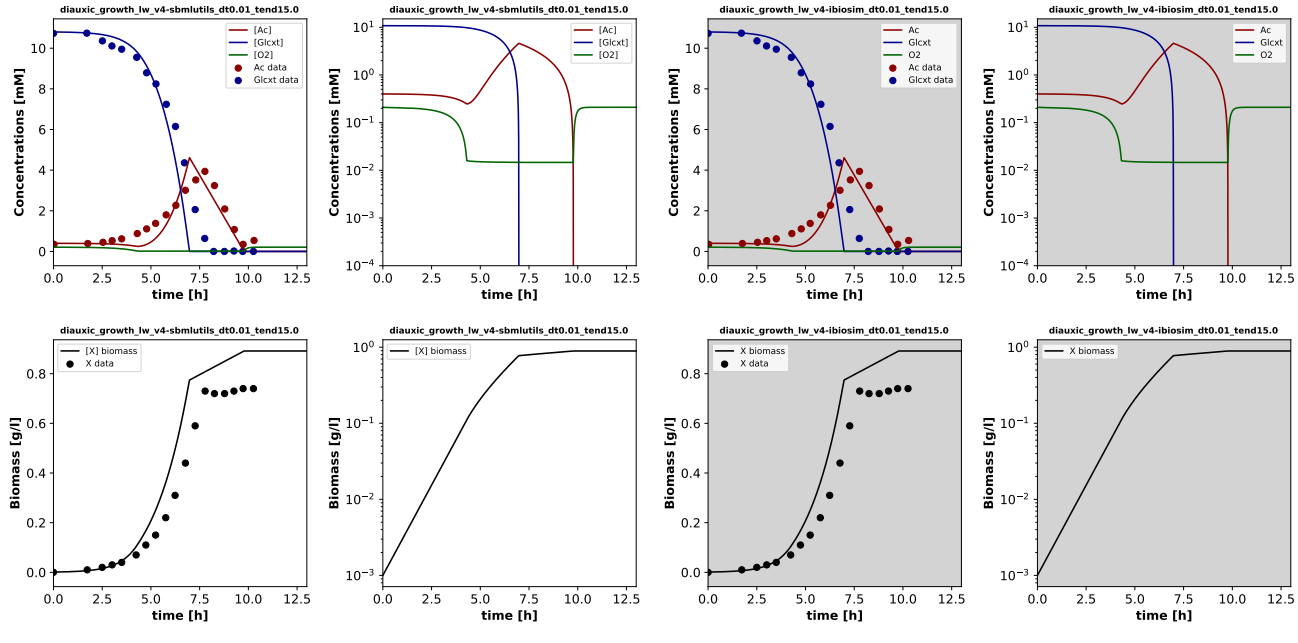
The authors would like to thank Frank Bergmann and Ilya Kiselev for participating in the discussions for this work.

Funding

LW and CM are supported by the National Science Foundation under Grants CCF-1218095 and CCF-1748200. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation. MK is supported by the Federal Ministry of Education and Research (BMBF, Germany) within the research network Systems Medicine of the Liver (LiSyM) (grant number 031L0054).

Supplementary Material

S1 Schema for encoding DFBA in SBML



(a) This is the simulation results for the diauxic growth of *E. Coli* in `sbmlutils`.

(b) This is the simulation results for the diauxic growth of *E. Coli* simulated in `iBioSim`.

Fig. 5: This plot shows the results for the model representing diauxic growth in *E. coli*. The model is able to reproduce the general behavior captured from experiment data. There is an exponential cell growth while glucose is present in the model, but when the cell runs out of glucose, growth slows down and is affected mostly by oxygen. However, when the cell runs out of glucose and oxygen, growth diminishes significantly. The model was simulated for 15[h] with a time step `dt` of 0.01[h].

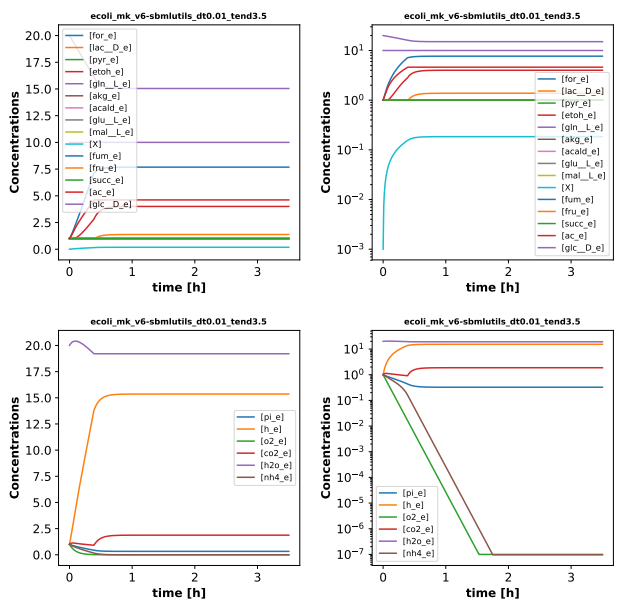


Fig. 6: DFBA simulation results for core metabolism of *E. coli* with sbmlutils. The proposed approach can be used in larger models, such as the *E. coli* model described in the paper. The model is growing aerobically on glucose in the initial phase and reaches a steady state after oxygen is consumed. The model was simulated for 3.5[h] with a time step `dt` of 0.01[h].

S2 Reproducibility results between sbmlutils and iBioSim

S3 `toy_wholecell` Minimal DFBA model COMBINE archive

S4 `toy_wholecell` Minimal DFBA model Cytoscape session file

S5 `toy_atp` Minimal glycolysis DFBA model COMBINE archive

S6 `toy_atp` Minimal glycolysis DFBA model Cytoscape session file

S7 `diauxic` Diauxic DFBA model COMBINE archive

S8 `diauxic` Diauxic DFBA model Cytoscape session file

S9 `ecoli` E.coli core DFBA model COMBINE archive

References

Bergmann, F. T. *et al.* (2014). COMBINE archive and OMEX format: one file to share all information to reproduce a modeling project. *BMC bioinformatics*, **15**(1), 369.

Bordbar, A. *et al.* (2014). Constraint-based models predict metabolic and associated cellular functions. *Nature reviews. Genetics*, **15**(2), 107.

Courtot, M. *et al.* (2011). Controlled vocabularies and semantics in systems biology. *Molecular Systems Biology*, **7**(1).

Ebrahim, A. *et al.* (2013). COBRApy: COnstraints-Based Reconstruction and Analysis for python. *BMC systems biology*, **7**(1), 74.

Gomez, J. A., Höffner, K., and Barton, P. I. (2014). DFBAlab: a fast and reliable MATLAB code for dynamic flux balance analysis. *BMC bioinformatics*, **15**, 409.

Hanly, T. J. and Henson, M. A. (2011). Dynamic flux balance modeling of microbial co-cultures for efficient batch fermentation of glucose and xylose mixtures. *Biotechnology and bioengineering*, **108**(2), 376–385.

Hedley, W. J. *et al.* (2001). A short introduction to CellML. *Philos. T. Roy. Soc. A*, **359**(1783), 1073–1089.

Hjersted, J. L. *et al.* (2007). Genome-scale analysis of saccharomyces cerevisiae metabolism and ethanol production in fed-batch culture. *Biotechnology and bioengineering*, **97**(5), 1190–1204.

Höffner, K. *et al.* (2013). A reliable simulator for dynamic flux balance analysis. *Biotechnology and bioengineering*, **110**(3), 792–802.

Hucka, M. *et al.* (2003). The Systems Biology Markup Language (SBML): a medium for representation and exchange of biochemical network models. *Bioinformatics*, **19**(4), 524–531.

Karr, J. R. *et al.* (2012). A whole-cell computational model predicts phenotype from genotype. *Cell*, (2), 389–401.

Kauffman, S. (1969). Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of Theoretical Biology*, **22**(3), 437 – 467.

King, Z. A. *et al.* (2015). BiGG Models: A platform for integrating, standardizing and sharing genome-scale models. *Nucleic acids research*, **44**(D1), D515–D522.

Kitano, H. (2002). Computational systems biology. *Nature*, **420**(6912), 206–210.

König, M. (2017). matthiaskoenig/sbmlutils: sbmlutils-v0.1.8 10.5281/zenodo.1045519.

König, M. *et al.* (2012). CySBML: a Cytoscape plugin for SBML. *Bioinformatics*, **28**(18), 2402–2403.

Le Novere, N. *et al.* (2009). The Systems Biology Graphical Notation (SBGN). *Nature biotechnology*, **27**(8), 735–741.

Lequeux, G. *et al.* (2010). Dynamic metabolic flux analysis demonstrated on cultures where the limiting substrate is changed from carbon to nitrogen and vice versa. *BioMed Research International*, **2010**.

Luo, R.-Y. *et al.* (2006). Dynamic analysis of optimality in myocardial energy metabolism under normal and ischemic conditions. *Molecular systems biology*, **2**(1).

Madsen, C. *et al.* (2012). Design and test of genetic circuits using iBioSim. *Design and Test of Computers, IEEE*, **29**(3), 32–39.

Mahadevan, R. *et al.* (2002). Dynamic flux balance analysis of diauxic growth in Escherichia coli. *Biophysical journal*, **83**(3), 1331–1340.

Mahadevan, R. and Schilling, C. (2003). The effects of alternate optimal solutions in constraint-based genome-scale metabolic models. *Metabolic engineering*, **5**(4), 264–276.

Meadows, A. L. *et al.* (2010). Application of dynamic flux balance analysis to an industrial Escherichia coli fermentation. *Metabolic engineering*, **12**(2), 150–160.

Morris, M. K. *et al.* (2010). Logic-based models for the analysis of cell signaling networks. *Biochemistry*, **49**(15), 3216–3224. PMID: 20225868.

Olivier, B. G. and Bergmann, F. T. (2015). The Systems Biology Markup Language (SBML) Level 3 Package: Flux Balance Constraints. *Journal of Integrative Bioinformatics*, **12**(2), 269.

Orth, J. *et al.* (2010). Reconstruction and use of microbial metabolic networks: the core Escherichia coli metabolic model as an educational guide. *EcoSal Plus*.

Pizarro, F. *et al.* (2007). Coupling kinetic expressions and metabolic networks for predicting wine fermentations. *Biotechnology and bioengineering*, **98**(5), 986–998.

Savinell, J. M. and Palsson, B. O. (1992). Network analysis of intermediary metabolism using linear optimization. i. development of mathematical formalism. *Journal of theoretical biology*, **154**(4), 421–454.

Smith, L. P. *et al.* (2015). SBML Level 3 package: Hierarchical Model Composition, Version 1 Release 3. *Journal of Integrative Bioinformatics*, **12**(2), 268.

Somogyi, E. T. *et al.* (2015). libRoadRunner: a high performance SBML simulation and analysis library. *Bioinformatics*, **31**(20), 3315–3321.

Thomas, R. (1973). Boolean formalization of genetic control circuits. *Journal of Theoretical Biology*, **42**(3), 563 – 585.

Varma, A. *et al.* (1993). Biochemical production capabilities of Escherichia coli. *Biotechnology and bioengineering*, **42**(1), 59–73.

Varma, A. and Palsson, B. O. (1994). Stoichiometric flux balance models quantitatively predict growth and metabolic by-product secretion in wild-type Escherichia coli W3110. *Applied and environmental microbiology*, **60**(10), 3724–3731.

Waltemath, D. *et al.* (2011). Reproducible computational biology experiments with SED-ML-the Simulation Experiment Description Markup Language. *BMC systems biology*, **5**(1), 1.

Waltemath, D. *et al.* (2016). Toward community standards and software for whole-cell modeling. *IEEE Transactions on Biomedical Engineering*, **63**(10), 2007–2014.

Watanabe, L. H. and Myers, C. J. (2014). Hierarchical stochastic simulation algorithm for SBML models of genetic circuits. *Frontiers in Bioengineering and Biotechnology*, **2**, 55.

Zhukova, A. *et al.* (2011). Kinetic Simulation Algorithm Ontology (KISAO).