Development of a framework for whole-cell model creation

Frederico Chaves Carvalho[1], Paulo Eduardo Ambrósio[1]

[1] Programa de Pós-Graduação em Modelagem Computacional em Ciência e
Tecnologia– Universidade Estadual de Santa Cruz (UESC)
Rodovia Jorge Amado, Km 16, Bairro Salobrinho – Ilhéus – Bahia

fcc073@gmail.com, peambrosio@uesc.br

Abstract. *The use of whole-cell models in research has the potential to be a powerful tool for scientific discovery, allowing researchers to test hypotheses faster than using in vitro or in vivo methods. Such models can be considered the equivalent of Computer Aided Design for Biology. However, given their complexity, it is still difficult to employ them as an instrument in investigations. In order to solve this problem, we are developing a framework with the purpose to guide and help scientists through the process of creating whole-cell models faster, enabling them to use these tools as part of their research. This paper brings details of the early stages of the framework's development process.*

1. Introduction and Objectives

Whole-cell models are computational models that aims to simulate the behavior of living cells by representing all the intracellular biochemical processes, as well as the function of most (or all) genes in the cell. Even though the current models are not perfect, they have proved to be powerful tools with the potential to enable scientist to explore new methods for scientific discoveries in fields such as medicine, biology and bioengineering [Covert, 2014].

The first whole-cell model to consider the role of genes in the lifecycle of the cell was created as part of the E-CELL project [Tomita et al., 1999]. Initiated in 1996, soon after the release of the full genome sequence of *Mycoplasma genitalium*, the team developed a computer model of this bacterium using 127 of the 525 genes, which was enough to simulate a cell capable of performing the basic processes necessary for survival. The model was also capable of reproducing some of the behaviors of the cell, such as the peak for intracellular ATP, followed by a sharp decline, when a starvation process begins.

The most comprehensive whole-cell model to date was presented in 2012, and represents all the 525 genes and most of the molecules in the *M. genitalium* [Karr et al., 2012; Goldberg et al., 2016]. The model was able to predict relevant cell behavior, such as the impact of nonessential single-gene disruption in the growth rate of the cell [Sanghivi et al., 2013]. These observations were important to ratify the possibility of using such models to accelerate research and guide in vitro or in vivo experiments.

Despite the aforementioned successes in the creation of whole-cell models, developing new ones remains a challenge. The first reason is the large amount of data necessary to represent a cell with acceptable accuracy. For instance, one needs to model the proteins and their functions, the metabolic pathways, the compartments, the extracellular environment, etc. Another challenge is the high computational cost for running simulations of increasingly complex models. That is the case of the *Mycoplasma genitalium's* model, which needs approximately 1 core day of an Intel E5520 CPU to complete the simulation [Goldberg et al., 2016].

Moreover, to create and simulate a whole-cell model, one needs to have a strong background in both biology and computer science, which limits the amount of scientists that can use them as a tool for research and prediction. This also restricts the possibility of other uses for whole-cell models, such as in education. To make models accessible to biologists and physicians with average computational knowledge, it would be necessary to simplify the computational part, and one of the ways to do that is to create a software with a user-friendly graphic user interface.

Our goal with this project is to develop a tool that will help the construction of accurate whole-cell models that can be used in research. Therefore, we are creating a framework with graphical user interface that aims to guide the user through the steps of a simplified methodology for model creation. This way, we hope to provide researchers, who otherwise wouldn't be able to build and use whole-cell models, with the possibility of using them as way to accelerate their investigations.

2. Methodology

Bearing in mind that the main objective is to simplify and expedite the creation of accurate whole-cell models, the first step was to define the requirements and specifications for the framework and its GUI. This was done by analyzing two of the most successful models [Karr et al. 2012; Tomita, 2001], their biochemical background and current techniques and paradigms for developing biological models [Fall et al., 2002; Karr et al., 2015; Freddolino and Tavazoie, 2012].

One way to build a whole-cell model is by describing it as a function of chemical species and reactions. By assigning concentrations to each species and rates for each reaction, and creating conditions that determine whether the reaction occurs it is possible to apply a simple ordinary differential equation solver to simulate the behavior of the system. The model can further be refined by adding the function of each gene, protein and organelle. It is possible to do that is by defining each of these components as objects and assigning their behavior to methods associated with them. The complexity, and accuracy of the model can be increased by creating separate compartments or regions within the cell, link reactions to model metabolic pathways, account for the function and structure of the plasma membrane, define the extracellular environment (in terms of concentrations, pH, temperature, osmotic pressure, luminosity, etc.)

With all the above information, we established a simplified model creation methodology, and the framework's interface was built reflecting such procedure and taking into account the need for user-friendliness and versatility. The open source version of Qt Creator was the software used to develop the framework's GUI and its basic functionalities were implemented in Python, due to its versatility and simplicity.

The simulation algorithm used by the most recent model was single-threaded and slow [Goldberg et al., 2016]. In order to remove this bottleneck we are developing a parallel version of two simulation algorithms, which are ODE and Gillespie's method (for stochastic simulation). To ensure that the model runs in the most efficient way, C language is being used to implement these algorithms. This choice serves a double purpose: enhancing the efficiency of the code, and enabling easier parallel processing techniques that will be done using the Open MP and MPI standards.

In order to validate the framework, a model of the bacterium *M. genitalium* will be created and simulated. The results will be compared to experimental data from the literature as well as to the simulation results from the model developed by Karr et al. (2012) as a benchmark.

3. Framework's specifications and requirements

The input information needed for creating a whole-cell model is diverse and scattered through different databases. For instance, the user will find annotated genome sequences and metabolic pathway maps in KEGG database, but will need to look elsewhere for some specific reaction information. While some organism specific databases, such as WholeCellKB can facilitate the work of the user, they are still not common and, most of the time, incomplete. In order to assist the user in the analysis and selection of data, the framework needs to be able import, organize and display information from different databases that can complement each other.

In cases where the available data is not sufficient to create a comprehensive model, the frameworks must allow the user to create reduced models, using only the information available, to simulate part of the processes involved in a cell, such as a single metabolic pathway or a set of reactions. Another important option the framework must provide is to use prediction tools to estimate the missing values and variables, and thus, assist the user in completing the model.

A living cell can behave in a myriad of ways that makes it difficult to describe its behavior using only one algorithm. For this reason, the framework has to offer the choice to use different types of simulation algorithms. In the first version, three choices are available: partial differential equation, ordinary differential equation, and Gillespie's method algorithm for stochastic simulation. The user must also have the possibility of defining compartments or regions of the cell and assign a specific simulation algorithm to each one of them, provided a clear set of inputs and outputs is possible from each compartment.

The framework is being developed to be versatile regarding the cells that can be modelled, meaning that the user can build models for different organisms, not only for the *M. genitalium*. However, because of the added complexity present in eukaryotic cells, such as the presence of internal membranes and the occurrence of splicing and other more sophisticated events, the initial goal is to enable the creation of prokaryote cell models only.

Bearing in mind the high computational demand reported in the current models, the support for parallel processing and use of GPU as coprocessor must be implemented. Figure 1 shows details of the framework's architecture and operation. Note that the parallelization occur in the "Model processing" phase, where simulations in different compartments are run by different threads.

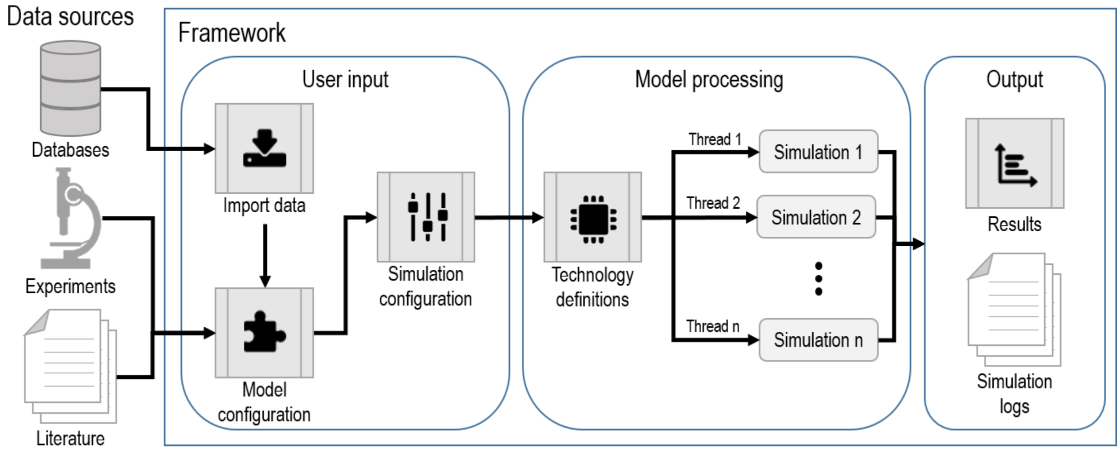


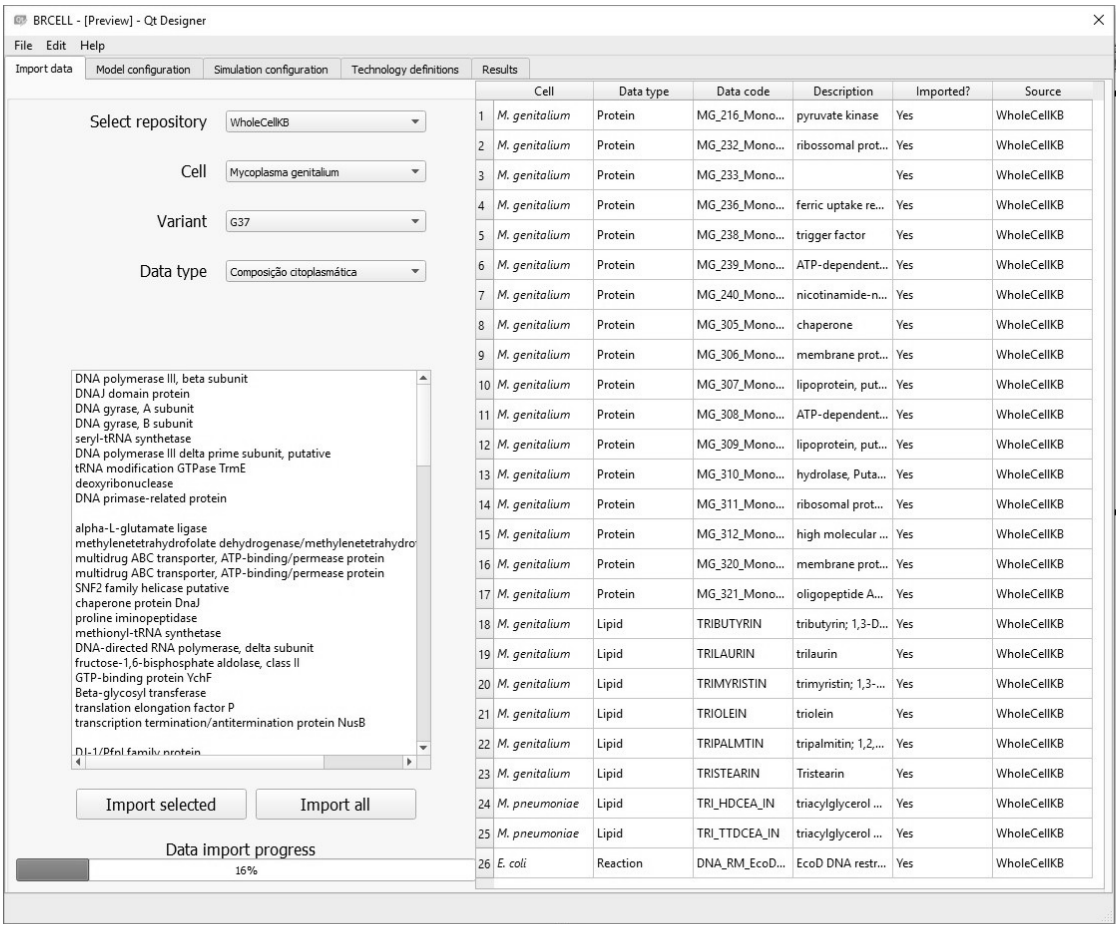
Figure 1. The framework's architecture

4. First results

The first stage of the framework's development process, which is the design of its graphic user interface, is complete. The GUI was designed to reflect the methodology that must be followed to build a whole-cell model that takes into account the genetic information, the molecules inside the cell and the reactions they are involved in, the extracellular environment, and the structure of the cell. To allow the user to go through these steps, the framework is divided into five modules: "Import data", "Model configuration", "Simulation configuration", "Technology definitions" and "Results".

The "Import data" module is implemented and functional, and is currently capable of fetching data from KEGG and WholeCellKB databases. The user can define the data to be imported and used for the model building. Figure 2 shows a preview of how the framework's first module works. It is possible to select the repository from which the data should be imported, the cell/organism, the variant (if more than one is available in the repository), and the data type, which can be the genome sequences, chemical species, reactions or metabolic pathways. The framework creates a separate file containing classes corresponding to every type of information that is allowed to be used for model creation in the framework, and each entry in the table in figure 2 is written in the same file as an object of the corresponding class (data type). These objects will be used alongside the reactions, concentration and positional information (if available) as inputs to the simulation algorithm.

The "Model configuration" module is where the user will use the data imported previously to build the mode and define other important information and parameters for the cell, such as its shape and size, the extracellular environment, membrane structure and composition, concentrations of each species and (optionally) define regions of the cell that has a different behavior (i.e. assign a different simulation algorithm or different concentrations for one or more species). The user is also allowed to manually input new data from other sources (papers, experimental data, information from other databases, etc.), and determine or change the reactions and interactions between each component. Each new piece of information given by the user is also stored as an object in the same file as the imported data, and a new python script is created with the remaining definitions. The module also allows the user to save the model with all the definitions and inserted data at any time.



Select repository: WholeCellKB

Cell: Mycoplasma genitalium

Variant: G37

Data type: Composição citoplasmática

DNA polymerase III, beta subunit
DNAJ domain protein
DNA gyrase, A subunit
DNA gyrase, B subunit
seryl-tRNA synthetase
DNA polymerase III delta prime subunit, putative
tRNA modification GTPase TrmE
deoxyribonuclease
DNA primase-related protein

alpha-L-glutamate ligase
methylenetetrahydrofolate dehydrogenase/methylenetetrahydro
multidrug ABC transporter, ATP-binding/permease protein
multidrug ABC transporter, ATP-binding/permease protein
SNF2 family helicase putative
chaperone protein DnaJ
proline iminopeptidase
methionyl-tRNA synthetase
DNA-directed RNA polymerase, delta subunit
fructose-1,6-bisphosphate aldolase, class II
GTP-binding protein YchF
Beta-glycosyl transferase
translation elongation factor P
transcription termination/antitermination protein NusB

DJ-1/PfpI family protein

Data import progress: 16%

	Cell	Data type	Data code	Description	Imported?	Source
1	*M. genitalium*	Protein	MG_216_Mono...	pyruvate kinase	Yes	WholeCellKB
2	*M. genitalium*	Protein	MG_232_Mono...	ribossomal prot...	Yes	WholeCellKB
3	*M. genitalium*	Protein	MG_233_Mono...		Yes	WholeCellKB
4	*M. genitalium*	Protein	MG_236_Mono...	ferric uptake re...	Yes	WholeCellKB
5	*M. genitalium*	Protein	MG_238_Mono...	trigger factor	Yes	WholeCellKB
6	*M. genitalium*	Protein	MG_239_Mono...	ATP-dependent...	Yes	WholeCellKB
7	*M. genitalium*	Protein	MG_240_Mono...	nicotinamide-n...	Yes	WholeCellKB
8	*M. genitalium*	Protein	MG_305_Mono...	chaperone	Yes	WholeCellKB
9	*M. genitalium*	Protein	MG_306_Mono...	membrane prot...	Yes	WholeCellKB
10	*M. genitalium*	Protein	MG_307_Mono...	lipoprotein, put...	Yes	WholeCellKB
11	*M. genitalium*	Protein	MG_308_Mono...	ATP-dependent...	Yes	WholeCellKB
12	*M. genitalium*	Protein	MG_309_Mono...	lipoprotein, put...	Yes	WholeCellKB
13	*M. genitalium*	Protein	MG_310_Mono...	hydrolase, Puta...	Yes	WholeCellKB
14	*M. genitalium*	Protein	MG_311_Mono...	ribosomal prot...	Yes	WholeCellKB
15	*M. genitalium*	Protein	MG_312_Mono...	high molecular ...	Yes	WholeCellKB
16	*M. genitalium*	Protein	MG_320_Mono...	membrane prot...	Yes	WholeCellKB
17	*M. genitalium*	Protein	MG_321_Mono...	oligopeptide A...	Yes	WholeCellKB
18	*M. genitalium*	Lipid	TRIBUTYRIN	tributyrin; 1,3-D...	Yes	WholeCellKB
19	*M. genitalium*	Lipid	TRILAURIN	trilaurin	Yes	WholeCellKB
20	*M. genitalium*	Lipid	TRIMYRISTIN	trimyristin; 1,3-...	Yes	WholeCellKB
21	*M. genitalium*	Lipid	TRIOLEIN	triolein	Yes	WholeCellKB
22	*M. genitalium*	Lipid	TRIPALMTIN	tripalmitin; 1,2,...	Yes	WholeCellKB
23	*M. genitalium*	Lipid	TRISTEARIN	Tristearin	Yes	WholeCellKB
24	*M. pneumoniae*	Lipid	TRI_HDCEA_IN	triacylglycerol ...	Yes	WholeCellKB
25	*M. pneumoniae*	Lipid	TRI_TTDCEA_IN	triacylglycerol ...	Yes	WholeCellKB
26	*E. coli*	Reaction	DNA_RM_EcoD...	EcoD DNA restr...	Yes	WholeCellKB

Figure 2. A preview of the “Import data” module of the framework

The Simulation configuration module allows the user to choose the simulation algorithm, what information should be monitored and displayed, the desired results output and to be used. In short, this module will allow the user to opt for a simpler simulation for faster results, or a more complete study.

In the Technology definitions module, the user can decide whether parallelization of the model should occur, and what resources should be used. For instance, the user can turn on the use of graphic card as a coprocessor and select which clusters will be used to run the simulation.

The Results module is designed to show the results of the simulation through graphics in real time. It also provides the option to abort the simulation in the event of unexpected model behavior. Once the simulation is complete, the user will have access to a report containing the main results, simulation definitions used and simulation logs.

With the exception of the data importation module, all the other sections of the framework are currently under development and have either partial functionality or no functionality yet.

5. Acknowledgements

The authors of this work acknowledge FAPESB (Fundação de Amparo à Pesquisa do Estado da Bahia) for its scholarship number BOL0029/2018.

References

CARRERA, J.; COVERT, M. W. (2015) “Why Build Whole-Cell Models?” In: Trends in Cell Biology, v. 25, n. 12, p. 719–722, 2015.

COVERT, M. W. (2014) “Simulating a living cell”. Scientific American, v. 310, n. 1, p. 44–51.

FALL, C.P.; MARLAND, E.S.; WAGNER, J.M.; TYSON, J.J. (2002) “Computational cell biology”. Springer.

FREDDOLINO, P. L.; TAVAZOIE, S. (2012) “The dawn of virtual cell biology”. Cell, v. 150, n. 2, p. 248–250.

GOLDBERG, A. P.; CHEW, Y. H.; KARR, J. R. (2016) “Toward Scalable Whole-Cell Modeling of Human Cells”. Proceedings of the 2016 annual ACM Conference on SIGSIM Principles of Advanced Discrete Simulation - SIGSIM-PADS ’16, p. 259–262.

KARR, J. R. et al. (2012) “A whole-cell computational model predicts phenotype from genotype”. Cell, v. 150, n. 2, p. 389–401.

KARR, J. R.; TAKAHASHI, K.; FUNAHASHI, A. (2015) “The principles of whole-cell modeling”, In: Current Opinion in Microbiology, v. 27, p. 18–24.

MACKLIN, D. N.; RUGGERO, N. A.; COVERT, M. W. (2014) “The future of whole-cell modeling”, In: Current Opinion in Biotechnology, v. 28, p. 111–115.

PURCELL, O. et al. (2013) “Towards a whole-cell modeling approach for synthetic biology”. Chaos, v. 23, n. 2, p. 1–8.

SANGHVI, J. C. et al. (2013) “Accelerated discovery via a whole-cell model”. Nature Methods, v. 10, n. 12, p. 1192–1195.

TOMITA, M. (2001) “Whole-cell simulation: A grand challenge of the 21st century, In: Trends in Biotechnology, v. 19, n. 6, p. 205–210.

TOMITA, M. et al. (1999) “E-CELL: Software environment for whole-cell simulation”, In: Bioinformatics, v. 15, n. 1, p. 72–84.