Introduction to Bioinformatics

Ernane Rosa Martins
(Organizador)

Atena Editora
2019

Ernane Rosa Martins
(Organizador)

Introduction to Bioinformatics

Atena Editora
2019

Editora Chefe: Profª Drª Antonella Carvalho de Oliveira
Diagramação e Edição de Arte: Geraldo Alves e Natália Sandrini
Revisão: Os autores

Dados Internacionais de Catalogação na Publicação (CIP)
(eDOC BRASIL, Belo Horizonte/MG)

I61 Introduction to bioinformatics [recurso eletrônico] / Organizador Ernane Rosa Martins. – Ponta Grossa (PR): Atena Editora, 2019.

Formato: PDF
Requisitos de sistema: Adobe Acrobat Reader
Modo de acesso: World Wide Web
Inclui bibliografia.
ISBN 978-85-7247-113-8
DOI 10.22533/at.ed.138191202

1. Bioinformática. 2. Inteligência artificial. I. Martins, Ernane Rosa.

CDD 570.285

Elaborado por Maurício Amormino Júnior – CRB6/2422

APRESENTAÇÃO

A bioinformática é um campo interdisciplinar, que busca analisar, interpretar e processar dados biológicos, com foco na aplicação de técnicas computacionais intensivas, tais como: métodos computacionais, teoria de grafos, inteligência artificial, algoritmos matemáticos, reconhecimento de padrões, mineração de dados, algoritmos de aprendizado de máquina, processamento de imagens e simulação computacional. Como um campo interdisciplinar, a bioinformática combina diversas áreas do conhecimento, como: engenharia, matemática, física, química, estatística, ciência da computação e biologia, entre outras.

A coletânea "*Introduction to bioinformatics*" é um livro composto por 6 capítulos que abordam assuntos atuais, tais como: o adenocarcinoma gástrico que é uma malignidade com elevada incidência e mortalidade no mundo; o vírus zika (VZIK) que é um Arbovirus que pertence à família Flaviviridae; As H + -ATPases que são proteínas integrais da membrana plasmática que têm a capacidade de utilizar a energia química da hidrólise de ATP para expulsar os prótons para o ambiente extracelular, atuando na manutenção da homeostase iônica e transporte de solutos; o vírus da família Geminiviridae que tem sido intensamente estudado devido à gravidade das doenças causadas em várias culturas importantes como: feijão, algodão, milho, tomate e mandioca.

Espero que os capítulos deste livro possam contribuir efetivamente na disseminação dos conhecimentos relevantes da bioinformática, proporcionando uma visão ampla sobre este campo de conhecimento.

Assim, desejo a todos uma excelente leitura.

Ernane Rosa Martins

SUMÁRIO

CAPÍTULO 6

DESENVOLVIMENTO DE FRAMEWORK PARA CRIAÇÃO DE MODELOS COMPUTACIONAIS DE CÉLULA COMPLETA

Frederico Chaves Carvalho
Programa de Pós-graduação em Modelagem Computacional em Ciência e Tecnologia – Universidade Estadual de Santa Cruz
Ilhéus – Bahia

Paulo Eduardo Ambrósio
Programa de Pós-graduação em Modelagem Computacional em Ciência e Tecnologia – Universidade Estadual de Santa Cruz
Ilhéus – Bahia

RESUMO: Propostas para a criação modelos computacionais de células completas que levem em consideração a função dos genes são relativamente recentes. Tais modelos buscam representar todos os processos bioquímicos intracelulares de maneira a fornecer uma maneira rápida e eficiente de obter resultados simulados confiáveis e comparáveis aos obtidos com métodos *in vitro* ou *in vivo*. O desenvolvimento desses modelos tem ajudado a consolidar o conhecimento atual da biologia, e fornecido subsídios para avanços científicos mais rápidos em áreas como a medicina e a bioengenharia. No entanto, a alta complexidade envolvida em sua construção é uma das principais barreiras para sua popularização. Visando simplificar e acelerar o processo de produção de novos modelos, nos propomos a criar um framework que represente a formalização da metodologia seguida pelos principais modelos atuais, e guie o usuário no processo de criação e simulação dos modelos.

PALAVRAS-CHAVE: Desenvolvimento de software, Modelagem computacional, Modelos de célula completa, Ferramenta computacional.

ABSTRACT: Attempts to create whole-cell computational models that take into consideration the function of genes are relatively recent. Such models aim to represent all the intracellular biochemical processes in a way that allows fast and efficient achievement of fast and reliable results, comparable to the ones obtained from in vitro or in vivo experiments. The development of these models has been important not only to consolidate the current knowledge of Biology, but also as a way to provide subsidy for faster scientific advancements in areas such as medicine and bioengineering. However, the high complexity involved in their construction is one of the main barriers to their popularization. To simplify and accelerate the process of model building, we are creating a framework as a tool that guides the user through the methodology used to create and simulate the main current whole-cell models.

KEYWORDS: Software development, Computer modelling, Whole-cell models, Computational tool.

1 I INTRODUÇÃO

O desenvolvimento de ferramentas computacionais especializadas é um passo importante para garantir a acessibilidade e popularização de determinadas metodologias e técnicas que, de outra maneira, não seriam possíveis de serem utilizadas pelo público que mais se beneficiaria de seu uso.

Nas engenharias, por exemplo, é comum encontrar ferramentas computacionais que podem ser utilizadas para acelerar e aperfeiçoar avanços científicos. Softwares como AutoCAD, ANSYS e COMSOL são frequentemente utilizados como auxiliares no processo de projeto e investigação, não só conferindo mais celeridade, como também reduzindo os custos envolvidos. Isso ocorre pois tais softwares permitem a obtenção de resultados através de simulações computacionais, dispensando a necessidade de construção de protótipos físicos, ou realização de experimentos laboratoriais onerosos.

Por outro lado, quando observamos os métodos utilizados pela biologia e áreas afins, como a farmacologia e a medicina, percebemos que existem poucas alternativas tecnológicas com a mesma capacidade. Apesar da existência de técnicas como virtual screening e sequenciamento, por exemplo, partes importantes das ciências biológicas ainda se sustentam puramente em observações e experimentos laboratoriais para produzir resultados e avançar.

Nesse contexto, modelos computacionais tem se mostrado bastante promissores quando utilizados como ferramentas para auxiliar no processo de investigação científica na Biologia (FALL et al. 2002). Seu uso permite que cientistas sejam capazes de testar hipóteses e observar fenômenos que, de outra forma, seriam inviáveis ou impossíveis de serem observados. Em particular, os modelos computacionais de célula completa são de especial interesse para investigações que buscam compreender melhor o funcionamento de células isoladas, que podem ser desde simples bactérias até células humanas, importantes para pesquisas sobre o câncer, por exemplo.

Modelos de célula completa são modelos computacionais que buscam prever o fenótipo (características observáveis e comportamentos de um organismo) através do conjunto de dados genotípicos (código genético) e bioquímicos de uma célula (KARR et al. 2012). Para tanto, esses modelos são criados a partir da representação computacional de diversos comportamentos celulares, que ocorrem em várias escalas, desde as interações entre as menores moléculas até a forma de estruturas maiores, como a membrana plasmática.

As tentativas de criação de modelos computacionais que representam sistemas biológicos não são recentes. Ainda na década de 1950 (TURING, 1952), Allan Turing criou o primeiro modelo biológico a ser representado matematicamente e calculado utilizando um computador. Tratava-se do modelo que ilustrava suas teorias a respeito das bases químicas da morfogênese. Este modelo, assim como a maioria dos que o seguiram, utilizavam puramente equações químicas e matemáticas para representar aspectos importantes do sistema descrito. Tal abordagem resultava na criação de

modelos simplificados, mas que descreviam os sistemas com precisão suficiente para testar hipóteses e realizar observações.

O primeiro modelo de célula completa a levar em consideração o papel dos genes no ciclo de vida de uma célula foi o E-CELL (TOMITA, 1999). O Projeto iniciado em 1996, na Universidade de Keio (Japão) tinha como objetivo modelar a bactéria *Mycoplasma genitalium*, cujo genoma havia sido sequenciado em 1995 (FRASER et al. 1995). Apesar de considerar o papel dos genes no fenótipo da bactéria, o modelo representava apenas 127 dos 525 genes da *M. genitalium*, o que foi o suficiente para que o modelo pudesse replicar alguns comportamentos celulares conhecidos, como o rápido pico na concentração de ATP intracelular, que ocorre quando a célula inicia um processo de morte por falta de alimento.

Cerca de 16 anos depois, o segundo modelo da mesma bactéria foi criado por um grupo de pesquisadores da Universidade de Stanford, nos Estados Unidos (KARR et al. 2012). O novo modelo representa a função de todos os genes da *M. genitalium*, bem como a quase totalidade de seus fenômenos bioquímicos, sendo capaz de replicar computacionalmente uma série de comportamentos celulares, alguns até então pouco estudados.

Para a construção do modelo, a equipe precisou reunir e analisar dados experimentais disponíveis em aproximadamente 900 artigos científicos, extraindo cerca de 1900 parâmetros que foram utilizados na modelagem (HAYES, 2013). O modelo final é constituído de 28 submodelos, que descrevem distintos processos celulares, cada um com inputs e outputs específicos. Um dos maiores desafios nessa abordagem inovadora foi a integração dos submodelos em um único modelo, já que nem todos os módulos possuíam a mesma natureza, podendo eles ser matemáticos, booleanos, probabilísticos, baseados em regras, etc. (KARR et al. 2012).

Por se tratar de um modelo extremamente complexo e que abrange eventos que ocorrem em várias dimensões simultaneamente, é natural que o modelo seja computacionalmente custoso. Nesse aspecto, vale ressaltar que as primeiras simulações do modelo necessitaram de aproximadamente 1 dia para completar (GOLDBERG, CHEW, KARR, 2016). Uma versão paralelizada do algoritmo de simulação foi capaz de reduzir esse tempo para cerca de 9 horas, quando executado por um cluster (HAYES, 2013).

Testes conduzidos utilizando o modelo confirmaram seu potencial como ferramenta para aceleração de pesquisas científicas. Através da realização de experimentos in silico, a mesma equipe que construiu o modelo foi capaz de predizer efeitos de parâmetros cinéticos e de determinadas enzimas sobre o ciclo de vida da célula, podendo retardar ou limitar o crescimento celular. Tais predições foram posteriormente confirmadas e validadas através de experimentos laboratoriais (SANGHIVI et al. 2013).

Tais êxitos alcançados levam a crer que, uma vez que modelos suficientemente precisos sejam criados, será possível utilizá-los em diversas áreas do conhecimento que possuem interface com a biologia. Por exemplo, através do uso de modelos de

célula completa, farmacêuticos poderão desenvolver medicamentos mais eficientes, e personalizados para o genoma de cada paciente; e bioengenheiros serão capazes de projetar microrganismos para funções específicas, como a produção de biocombustíveis ou detecção de doenças.

Apesar do sucesso alcançado com o modelo de 2012, e as potencialidades do uso de modelos computacionais de células completas nas pesquisa científica, a popularização de tais modelos ainda parece ser uma realidade distante. O alto volume e heterogeneidade dos dados, as dificuldades em traduzi-los em algoritmos computacionais e a falta de ferramentas projetadas especificamente para tal finalidade são alguns dos principais desafios que têm afastado diversos cientistas da construção desses modelos.

Tendo em mente tais barreiras, uma das possíveis soluções é o desenvolvimento de uma ferramenta computacional que que possa ser utilizada com facilidade por pesquisadores sem conhecimentos avançados de computação. É desejável que a ferramenta criada seja o mais fiel possível à metodologia utilizada para a construção do modelo construído por Karr et al. (20012), uma vez que este representa o estado da arte dos modelos de célula completa. Dessa maneira, seria possível produzir outros modelos com qualidade semelhante, e em menor tempo, contanto que haja disponibilidade de dados completos a respeito da célula a ser modelada.

Quanto aos dados, vale ressaltar que a popularização e os constantes avanços nas técnica de high-throughput tem propiciado um significativo aumento na quantidade de informações disponíveis a respeito das mais diversas células e processos celulares. Atualmente, diversos bancos de dados contam com conjuntos de dados suficientemente completos para desenvolver modelos mais simples.

Neste capítulo, descrevemos o processo de criação de um framework para desenvolvimento de modelos de célula completa, cujo objetivo é permitir que cientistas que possam se beneficiar com o uso desses modelos tenham condições de criá-los de maneira rápida e fácil, sem necessitar de conhecimentos avançados de computação, ou de uma equipe interdisciplinar para isso.

Nas seções subsequentes serão detalhadas as bases teóricas por trás da criação de modelos de células completas, a arquitetura e especificações do framework, e por fim, os primeiros resultados alcançados, bem como os próximos passos no desenvolvimento.

2 | BASES TEÓRICAS DO FRAMEWORK

Quando se estuda uma célula, por mais simples que esta seja, é possível identificar que trata-se de uma composição de diversos subsistemas que atuam em conjunto de maneira a possibilitar todas as funções que sustentam a vida da célula. Cada um desses subsistemas possui suas peculiaridades e componentes específicos,

que podem ou não estar presentes em outros subsistemas. Das interações entre distintos subsistemas surgem ainda propriedades emergentes, que devem ser levadas em consideração quando se modela uma célula.

A heterogeneidade observada entre os subsistemas tem reflexos na quantidade e qualidade dos dados disponíveis sobre cada um deles. Por exemplo, já existem métodos experimentais que permitem identificar e quantificar com precisão concentrações de determinados compostos em uma célula, e suas constantes de velocidade de reação. Por outro lado, ainda é um desafio identificar com alta precisão, e de maneira quantitativa, a ação de fatores de transcrição.

Para traduzir essa complexidade inerente às características celulares em linguagem computacional, é necessário recorrer à utilização de diversos paradigmas diferentes de modelagem (COVERT, 2006). Enquanto as reações entre moléculas podem ser modeladas como um sistema de equações diferenciais ordinárias, redes reguladoras de genes, que representam a ação de fatores de transcrição, são melhor representadas computacionalmente através de modelos booleanos.

É possível classificar os diferentes tipos de modelos que podem ser utilizados para tal finalidade em três grandes grupos: modelos baseados em redes; modelos baseados em regras; e modelos estatísticos (KLIP et al., 2016). Cada um desses grupos requer diferentes tipos de input e geram outputs distintos, fazendo-os mais ou menos apropriado como solução para modelar determinados sistemas.

Os modelos baseados em redes, por exemplo, são apropriados quando os sistemas sendo modelados podem ser descritos como um conjunto de nós interconectados por relações bem definidas. Um exemplo disso são redes metabólicas, que possuem metabólitos como nós, e reações químicas como conexões entre os nós. Para representa-las, podemos utilizar sistemas de equações diferenciais ordinárias, equações diferenciais parciais ou equações diferenciais estocásticas, por exemplo

Para ilustrar esse tipo de modelo, podemos utilizar a seguinte reação:

$$A + B \underset{k_1}{\overset{k_2}{\rightleftarrows}} C$$

Para a reação acima, podemos elaborar a seguinte equação diferencial ordinária, que representa a taxa com a qual a concentração de C (representada por [C]) varia com o tempo:

$$\frac{d[C]}{dt} = k_1[A][B] - k_2[C]$$

O mesmo procedimento pode ser repetido para encontrarmos a variação das concentrações das espécies A e B nesse mesmo sistema. É importante ressaltar que, nesse caso, estamos aproximando o modelo através da consideração de que este representa uma solução bem homogeneizada, onde não há gradientes de concentração, ou desigualdade na distribuição espacial das moléculas. Se quisermos

incluir dados relativos à espacialidade nos modelos, o que pode ser útil ao levar em consideração fenômenos como a difusão, por exemplo, podemos recorrer ao uso de equações diferenciais parciais para isso, como a equação abaixo:

$$\frac{\partial C}{\partial t} = D\nabla^2 C$$

Os modelos baseados em redes podem ainda ser do tipo booleanos, redes de Petri, ou estocásticos.

O segundo grupo de modelos são os modelos baseados em regras, ou modelos baseados em agentes. Assim como os primeiros, estes possuem elementos claramente definidos, porém, a relação entre eles ocorre através de um conjunto de regras, e não reações ou interações que podem ser descritas por equações. Esse tipo de modelo é de grande utilidade quando se quer modelar sistemas de sinalização celular levando em consideração tanto dados temporais como espaciais, por exemplo.

Por fim, os modelos estatísticos, incluídos no último grupo, são modelos mais apropriados para trabalhar com modelos criados a partir de grande volume de dados experimentais. Esses modelos tem se tornado mais comuns e importantes à medida que o volume de dados se torna maior, já que através deste tipo de modelagem é possível estabelecer relações de causalidade e frequências de ocorrência de determinados eventos, por exemplo.

Do ponto de vista computacional, há ainda um outro desafio relativo não à construção, mas à simulação de modelos de célula completa. Por se tratar de modelos de alta demanda computacional, a construção de um modelo não otimizado pode resultar em tempos de simulação elevados. Para resolver este problema em potencial, além da otimização do modelo e do algoritmo de simulação, uma outra técnica pode ser utilizada: a paralelização dos algoritmos.

Paralelizar um algoritmo significa permitir que sua execução seja dividida entre diferentes unidades de processamento. Tendo em mente que a maioria dos computadores atuais possuem algum grau de descentralização em suas unidades de processamento, essa técnica ganha ainda mais importância. Além disso, através da aplicação de técnicas de paralelização, permite-se que os modelos possam ser simulados eficientemente em um cluster, acelerando ainda mais a obtenção de resultados em simulações de maior complexidade.

Diversos padrões estão disponíveis para auxiliar o processo de paralelização de algoritmos. Dentre eles, os mais conhecidos, e utilizados, hoje são MPI e OpenMP. Nas últimas duas décadas, uma nova técnica de paralelização vem ganhando bastante notoriedade, que é o uso de placas gráficas (GPU) como coprocessadores. Essa técnica também pode ser aplicada de maneira a auxiliar na simulação de modelos de célula completa. Para a utilização de GPUs, os principais padrões atuais são CUDA (restritos e otimizados para placas gráficas da marca NVIDIA), e o OpenCL (irrestritos, porém com menor otimização em alguns casos).

É importante observar que, apesar de existirem formas de utilizar os padrões supracitados com qualquer linguagem de programação, esses foram desenvolvidos para implementação utilizando C ou Fortran, de maneira que a utilização dessas linguagens geralmente resulta em códigos com maior grau de eficiência.

3 I ARQUITETURA E FUNCIONAMENTO DO FRAMEWORK

Para alcançar o objetivo de ser uma ferramenta de fácil utilização para cientistas que não possuem conhecimentos avançados de programação, o framework foi planejado para guiar o usuário de forma simples e natural pelas diversas etapas do processo de criação de um modelo de célula completa.

Sua interface gráfica foi projetada de maneira a refletir a metodologia adotada como padrão para construção dos modelos de célula completa (KARR et al. 2012). Com a finalidade de estabelecer uma sequência lógica a ser seguida durante a criação de um modelo utilizando o framework, este foi dividido em 5 módulos. São eles:

- Inserção de Dados
- Modelagem
- Definições de Simulação
- Definições de Tecnologia
- Visualização de Resultados

Visando melhorar a usabilidade, a interface foi desenvolvida de maneira a permitir que o usuário possa alternar entre os módulos como achar conveniente, podendo voltar e inserir novos dados caso necessário, por exemplo. Além disso, tendo em vista o tempo necessário para a construção de um modelo, o usuário pode salvar o estado atual do modelo a qualquer momento, e continuar posteriormente a construção.

A Figura 1 mostra um esquema simplificado da arquitetura do Framework. Nela, as setas denotam o fluxo dos dados desde suas fontes (experimentos, bancos de dados e literatura) até a obtenção dos resultados finais. Na seção “Processamento”, é possível observar também a forma como a paralelização, cujos parâmetros serão definidos no módulo “Definições de Tecnologia”.

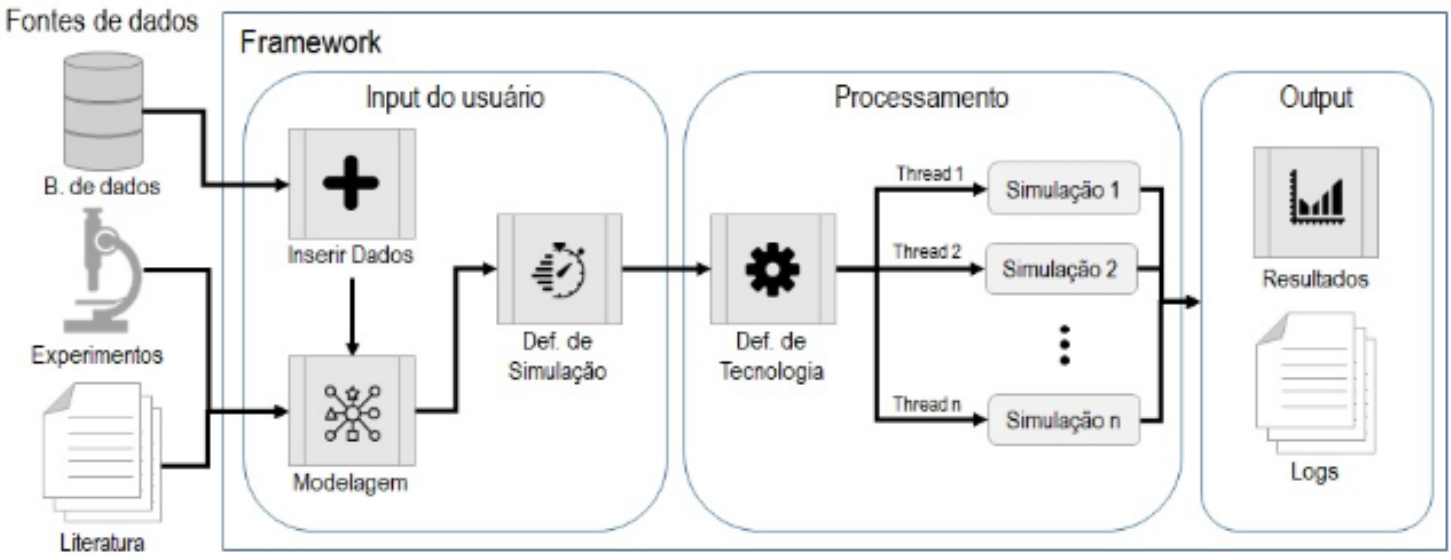

Figura 1: Arquitetura e fluxo de dados no framework

As funcionalidades básicas da interface (clique de botões, mudança de aba, inserção de dados, etc.) estão sendo desenvolvidos na linguagem Python. Essa escolha deve-se principalmente à simplicidade com que o código pode ser desenvolvido utilizando-se de bibliotecas disponíveis nesta linguagem, como numpy, pandas e matplotlib. Além disso, como a interface gráfica em si está sendo desenvolvida em Qt, Python oferece algumas soluções interessantes à organização dos arquivos, como a possibilidade de criar um arquivo à parte para a implementação das funções da interface, não alterando assim o arquivo principal, que pode ser atualizado sem necessidade de reescrita total do código.

Por não se tratar de um ponto crítico para o desempenho ou para o tempo total de simulação do modelo desenvolvido, não há necessidade de implementar técnicas de paralelização às funcionalidades básicas da interface. Por fim, a opção por essa linguagem permite implementar visualização de gráficos dinâmicos de maneira mais simples, através de extensões disponíveis na biblioteca matplotlib, que permitem integrá-la ao Qt.

Buscando melhorar a experiência do usuário com o framework, a tradução das informações biológicas em código é feita automaticamente, à medida que o usuário procede pelas etapas da modelagem. Para isso, foi implementada uma abordagem simples na qual o framework trabalha com a construção de 5 arquivos durante a construção do modelo e simulação do modelo.

O primeiro arquivo consiste em uma lista de classes, com atributos e métodos predefinidos. Cada tipo de input esperado (e permitido ao usuário) é representado por uma classe. Por exemplo, a classe “Proteínas” tem como atributos: “sequência”, “concentração”, “localização” (opcional), “gradiente” (opcional), “meia vida”, “estrutura 3D” (opcional), etc. Os métodos pré-definidos são “degradação”, “reação” “interação” “transporte ativo” “mutação aleatória” (estatisticamente ativado) e “difusão”. Usuários mais avançados podem editar este arquivo de maneira a definir novos métodos que

reflitam comportamentos desejados, além dos pré-definidos.

O segundo arquivo guarda todos os dados fornecidos pelo usuário como objetos das classes apropriadas. Esse arquivo é escrito à medida que o usuário fornece informações ao programa, nos módulos “Inserir Dados” e “Modelagem”. Sempre que o usuário decidir criar uma nova molécula, ele preencherá um formulário, e ao salvar, as informações entradas serão devidamente organizadas nesse arquivo.

O terceiro arquivo corresponde à reunião de todos os algoritmos que podem ser aplicados para a simulação do modelo construído. O(s) algoritmo(s) utilizado(s) dependerão das entradas fornecidas pelo usuário no módulo “Definições de Simulação”. Do ponto de vista do desempenho em geral da simulação, este é o arquivo mais importante. Sua função dele é processar todos os inputs de acordo com as opções estabelecidas no módulo “Definições de Tecnologia”, onde o usuário poderá controlar, até certo ponto, o grau de paralelização do modelo criado, além de introduzir informações a respeito da máquina ou cluster que processará a simulação, permitindo assim que os pontos críticos de paralelismo, já pré-definidos nos algoritmos sejam otimizados.

Por padrão, a paralelização se dará da seguinte maneira: cada submodelo será inicializado como um processo, e cada uma das reações, regras ou interações entre os componentes será processado por uma thread diferente, quando possível. Caso o usuário opte por utilizar também placas gráficas para o processamento, submodelos adjacentes (que possuem maior grau de interação entre si) poderão ser atribuídos ao mesmo processo, e serão processados por threads da mesma GPU.

O quarto arquivo contém os outputs da simulação. Nele, os dados referentes ao andamento da simulação serão gravados a cada passo concluído, que representam intervalos de tempo definidos pelo usuário. Como outputs, o usuário poderá definir no módulo “Definições de Simulação”, e visualizá-los no módulo “Resultados”. A estrutura do arquivo permite que o usuário tenha acesso a toda a evolução temporal da simulação, e visualize eventos celulares importantes com maior precisão.

O quinto arquivo, contendo os registros (logs) da simulação serão salvos em um arquivo à parte, e poderão ser consultados através de uma seção da interface. Neste arquivo, o usuário terá acesso a informações complementares da simulação, como as definições, parâmetros de entrada selecionados, critérios de parada e alertas de possíveis erros e sua provável origem.

4 I PRIMEIROS RESULTADOS

Uma vez definida a metodologia para criação de modelos e os requisitos destacados na seção anterior, o desenvolvimento do framework foi iniciado pela construção de sua interface gráfica. No estágio atual de desenvolvimento, todos os módulos da interface gráfica já estão desenhados, porém somente os dois primeiros

estão funcionais.

O primeiro módulo, denominado “Inserção de dados” (Figura 2) permite que o usuário forneça todos os dados necessários à criação do modelo. Com o intuito de facilitar o processo de inserção de dados, o usuário pode optar por importar dados diretamente de bancos de dados. Todos os dados inseridos ficam registrados para posterior uso no módulo “Modelagem”.

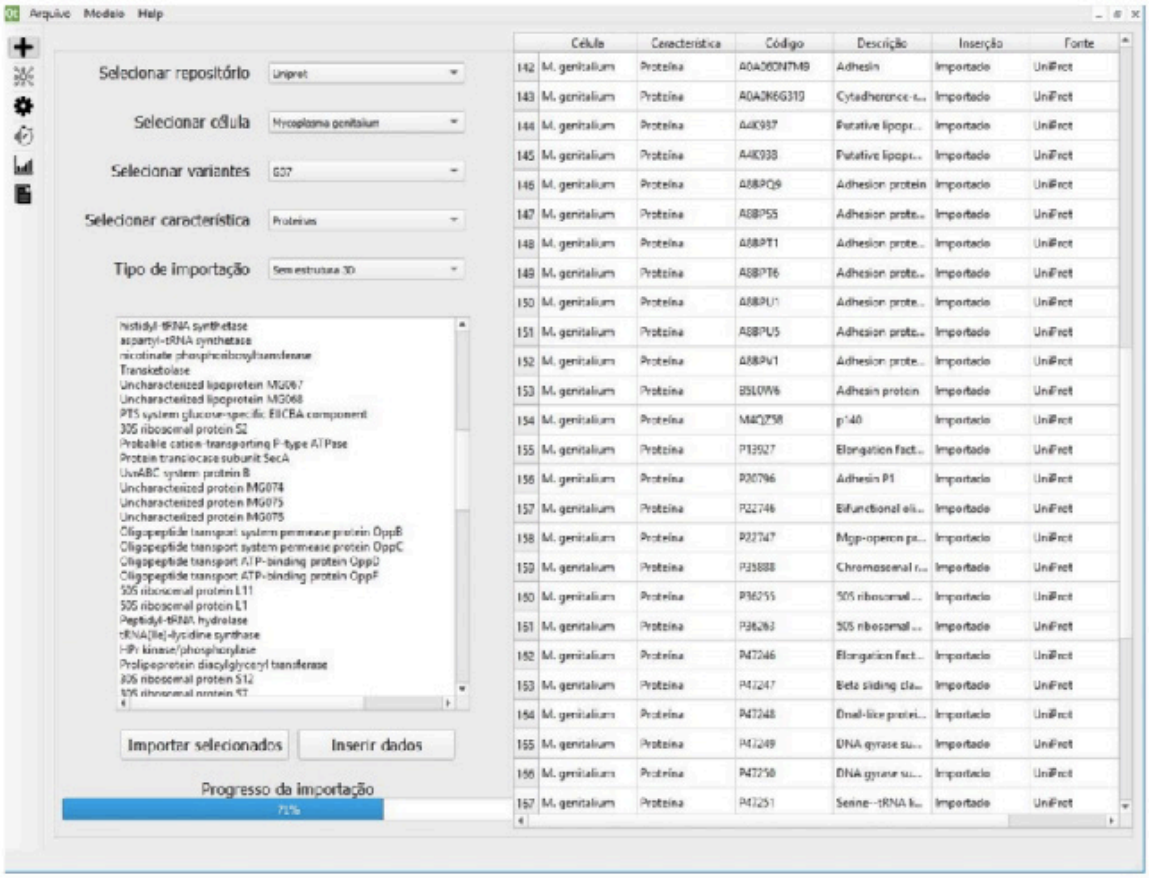

Figura 2: Exemplo da utilização da função "importar dados" do módulo 1. Essa função atualmente permite que usuário obtenha dados para construção do modelo diretamente de bancos de dados suportados (UniProt e WholeCellKB). Para importar dados, o usuário deverá selecionar o repositório, o tipo de célula, a variante (caso mais de um tipo esteja disponível, os parâmetros e o tipo de importação. Esta última opção permite escolher entre importar dados relativos à estrutura 3D, quando disponíveis, dados de interação, e sequencias, o que facilita o processo de modelagem.

O módulo “Modelagem” (Figura 3) permite que o usuário construa seu modelo utilizando os dados inseridos no módulo anterior. A interface foi projetada de maneira possibilitar que o modelo seja criado visualmente, onde o usuário define o tipo de modelo, os elementos participantes e as relações entre eles. Dessa forma é possível construir submodelos utilizando o paradigma de modelagem apropriado, devendo o usuário definir as regras de integração dos submodelos para a obtenção de um modelo final coerente.

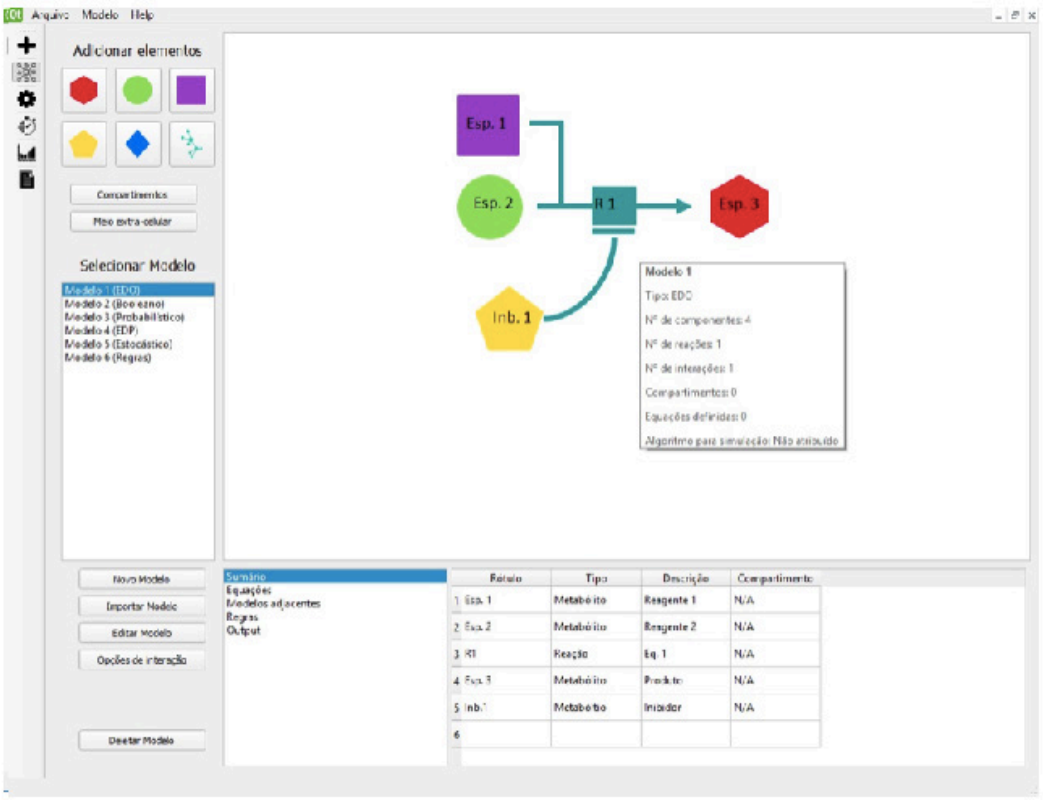

Figura 3: Exemplo de utilização do módulo "Modelagem". Neste módulo, o usuário pode utilizar formas geométricas para representar componentes do sistema modelado, e estabelecer a relação entre eles através de equações ou regras. Neste exemplo, o modelo criado é do tipo EDO, e representa uma reação genérica sob a ação de um inibidor Clicando sobre qualquer uma das espécies, inibidor ou reação representada, o usuário pode alterar concentrações, parâmetros ou equações que governam a reação, por exemplo. Clicando sobre as linhas é possível estabelecer as relações entre as espécies representadas, de maneira a gerar equações pré-definidas automaticamente, agilizando o processo de criação de modelos.

O terceiro módulo (“Definições de Simulação”) permite que o usuário defina os algoritmos a serem utilizados, a resolução da simulação (definição da escala de tempo e passo), variáveis a serem monitoradas, outputs, etc. Neste módulo o usuário pode ainda definir eventos que deverão ocorrer quando determinadas condições forem atendidas, a exemplo de perturbações devem ser introduzidas após a decorrência de certo tempo. Dessa forma, será possível simular experimentos com maior precisão dos resultados.

O quarto módulo permite definir e analisar critérios de paralelização, como quantidade de processos ou threads utilizados, como eles serão utilizados, e se a simulação deve ou não utilizar placas gráficas como coprocessadores, quando disponíveis. Por se tratar de um módulo que exige conhecimentos avançados, todas as configurações vem com valores pré-definidos recomendáveis que, permitem que a simulação seja paralelizada de forma genérica para reduzir os tempos de simulação de maneira razoável. Usuários avançados poderão ajustar critérios que exigem maior conhecimento de computação de alto desempenho, como balanceamento de carga, alocação de memória e divisão das threads e processos.

O último módulo (“Resultados”) permite analisar os resultados graficamente. Nele o usuário pode ter acesso a toda a evolução temporal das variáveis previamente

selecionadas, de maneira que o efeito de todos os eventos definidos possam ser visualizados tanto visual quanto quantitativamente. Em suma, esse módulo tem o objetivo de permitir uma análise aprofundada dos resultados da simulação.

5 I PRÓXIMOS PASSOS

Além da implementação das funcionalidades dos módulos restantes, as próximas etapas do desenvolvimento do framework incluem melhorias no suporte a diferentes bancos de dados, suporte a modelos SBML, criação e validação de um primeiro modelo de célula completa utilizando o framework, e publicação do framework.

Uma vez implementadas todas as funções do framework, será necessário proceder à validação do framework e de modelos criados por esta ferramenta. Para isso, um primeiro modelo será desenvolvido e simulado com o framework, e seus resultados comparados ao modelo atual da bactéria *Mycoplasma Genitalium* e a dados provenientes de literatura, que não tenham sido utilizados na construção do modelo.

Após validado, o framework será disponibilizado através da internet com licença Open Source, juntamente com sua documentação e manual, de maneira a permitir que pesquisadores nas áreas de biologia, farmácia, química, medicina, biomedicina, etc. tenham acesso a esta ferramenta em suas pesquisas, e possam produzir modelos computacionais utilizando-os como auxiliares em suas investigações.

Levando em consideração que nem todas as informações de interesse podem ser encontradas em um único, ou poucos bancos de dados, é desejável que o framework seja capaz de importar dados de uma maior variedade de repositórios. Dessa forma, o usuário poderá obter dados de maneira mais simples e rápida, acelerando ainda mais o processo de construção de modelos.

Outro aspecto importante é a possibilidade de compartilhar modelos, estejam eles prontos ou em desenvolvimento. Dessa maneira, será possível a criação de modelos de forma colaborativa. Por ser um formato já estabelecido, é interessante que os framework implemente uma forma de traduzir os arquivos referentes aos dados do modelo, e suas configurações, em um único arquivo SMBL, que possa ser publicado em plataformas de compartilhamento de modelos, por exemplo.

6 I AGRADECIMENTOS

Os autores deste trabalho agradecem à FAPESB (Fundação de Amparo à Pesquisa do Estado da Bahia) pelo financiamento do projeto, através da concessão da bolsa BOL0029/2018.

Os ícones utilizados para a produção dos diagramas e do Framework foram produzidos pelos usuários “Freepik” e “geotatah”, e disponibilizados gratuitamente em flaticon.com.

REFERÊNCIAS

COVERT, M. W. **Integrated Regulatory and Metabolic Models**. In: KRIETE, A.; EILS, R. Computational System Biology. San Diego: Elsevier Inc., 2006. cap. 10.

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

FALL, C.P. et al. **Computational Cell Biology**. New York: Springer, 2002.

FRASER, C. M. *et al.* **The Minimal Gene Complement of Mycoplasma genitalium**. Science, v. 270, n. 5235, p. 397–404, 1995.

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

GOLDBERG, A. P.; CHEW, Y. H.; KARR, J. R. **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, 2016.

HAYES, B. **Imitation of life**. American Scientist, v. 101, n. 1, p. 10–15, 2013.

HOOPS, S. *et al.* **COPASI - A COmplex PAthway SImulator**. Bioinformatics, v. 22, n. 24, p. 3067–3074, 2006.

HUCKA, M., et al. **The systems biology markup language (SBML)**: a medium for representation and exchange of biochemical network models. Bioinformatics, v. 19, n. 4, p. 524–531, 2003.

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

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

KLIPP, E.; LIEBERMEISTER, W.; WIERLING, C.; AXEL, K. **Systems Biology**. 2. ed. Weinheim: Wiley-VCH, 2012.

SZIGETI, B. et al. **A blueprint for human whole-cell modeling**. Current Opinion in Systems Biology, v. 7, p. 8–15, 2018.

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

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

TURING, A. M. **The chemical basis of morphogenesis**. Bulletin of Mathematical Biology, v. 237, n. 641, p. 37–72, 1952.

WALTEMATH, D. et al. **Toward Community Standards and Software for Whole-Cell Modeling**. IEEE Transactions on Biomedical Engineering, v. 63, n. 10, p. 2007–2014, 2016.

SOBRE O ORGANIZADOR

ERNANE ROSA MARTINS Doutorado em andamento em Ciência da Informação com ênfase em Sistemas, Tecnologias e Gestão da Informação, na Universidade Fernando Pessoa, em Porto/Portugal. Mestre em Engenharia de Produção e Sistemas pela PUC-Goiás, possui Pós-Graduação em Tecnologia em Gestão da Informação pela Anhanguera, Graduação em Ciência da Computação pela Anhanguera e Graduação em Sistemas de Informação pela Uni Evangélica. Atualmente é Professor de Informática do Instituto Federal de Educação, Ciência e Tecnologia de Goiás - IFG (Câmpus Luziânia), ministrando disciplinas nas áreas de Engenharia de Software, Desenvolvimento de Sistemas, Linguagens de Programação, Banco de Dados e Gestão em Tecnologia da Informação. Pesquisador do Núcleo de Inovação, Tecnologia e Educação (NITE).

Agência Brasileira do ISBN
ISBN 978-85-7247-113-8