Pipeline Automatizado para Modelagem e Simulação de Circuitos Genéticos: Integração de Redes de Petri Estocásticas, Algoritmo de Gillespie e Equações Diferenciais Ordinárias

Kauan Elias Schneider Fank

[1]Departamento de Informática e Estatística, Universidade Federal de Santa Catarina

Abstract

Este trabalho apresenta um pipeline automatizado para modelagem e simulação de circuitos genéticos, integrando três abordagens complementares: Redes de Petri Estocásticas (SPN) para visualização topológica, o Algoritmo de Gillespie para simulação estocástica, e Equações Diferenciais Ordinárias (ODE) para análise determinística. O sistema desenvolvido permite a busca automática de modelos no repositório BioModels, parsing de arquivos SBML, geração de representações em Redes de Petri (PNML e formato GreatSPN), e simulação com seleção automática do método mais apropriado. Validamos o pipeline com cinco circuitos genéticos clássicos: Repressilator, Toggle Switch, Feed-Forward Loop Incoerente Tipo 1, Relógio Circadiano de N. crassa e Operon Lac. Os resultados demonstram concordância com dados da literatura, evidenciando a utilidade do sistema como ferramenta educacional e de pesquisa em biologia sintética.

1 Introdução

1.1 Contexto e Motivação

A biologia sintética é um campo interdisciplinar que aplica princípios de engenharia ao design de sistemas biológicos, permitindo a construção de **circuitos genéticos** — redes de genes e proteínas que implementam funções computacionais dentro de células vivas [1]. Assim como circuitos eletrônicos processam sinais elétricos, circuitos genéticos processam sinais moleculares (concentrações de proteínas, metabólitos, sinais ambientais) para produzir respostas celulares programadas.

Exemplos de circuitos genéticos incluem osciladores (que geram ritmos periódicos), interruptores biestáveis (que armazenam informação binária), e detectores de sinais (que respondem a estímulos específicos). Estes componentes são os blocos fundamentais para aplicações como biossensores, produção de fármacos, e terapias celulares programáveis.

1.2 Desafios da Modelagem

Diferente da engenharia tradicional, onde componentes seguem especificações precisas, os circuitos genéticos operam em ambientes celulares caracterizados por:

- **Ruído intrínseco**: Flutuações estocásticas devido ao baixo número de moléculas envolvidas (dezenas a milhares de cópias por célula)
- **Variabilidade molecular**: Diferenças célula-a-célula em concentrações e taxas de reação
- **Comportamentos não-lineares**: Retroalimentação positiva e negativa que geram dinâmicas complexas [2]

A modelagem computacional torna-se essencial para o design racional de circuitos genéticos, permitindo prever comportamentos antes da construção experimental. Duas abordagens complementares são utilizadas:

1. **Modelos Determinísticos (ODE):** Representam a dinâmica através de equações diferenciais ordinárias, adequados quando o número de moléculas é grande e flutuações podem ser ignoradas (limite termodinâmico).
2. **Modelos Estocásticos (Gillespie):** Simulam reações individuais como eventos discretos e probabilísticos, capturando o ruído intrínseco essencial em sistemas com baixas contagens moleculares [3].

As **Redes de Petri Estocásticas** (SPN) surgem como uma ferramenta unificadora, permitindo representar visualmente a topologia da rede regulatória enquanto fundamentam matematicamente ambas as abordagens [4].

1.3 Objetivos e Contribuições

Este trabalho apresenta um **pipeline automatizado** para modelagem e simulação de circuitos genéticos que:

1. Busca e baixa modelos do repositório BioModels automaticamente
2. Realiza parsing completo de arquivos SBML, incluindo regras de atribuição e funções customizadas
3. Gera representações em Redes de Petri (PNML e GreatSPN)
4. Executa simulações com **seleção automática** do método mais apropriado (Gillespie ou ODE)
5. Detecta espécies de entrada para análises comparativas ON/OFF

Validamos o pipeline com cinco circuitos genéticos clássicos da literatura, demonstrando concordância quantitativa e qualitativa com os dados publicados.

2 Fundamentação Teórica

2.1 Formato SBML e Repositório BioModels

O Systems Biology Markup Language (SBML) é o padrão internacional para representação de modelos biológicos, permitindo intercâmbio entre diferentes ferramentas de simulação. O repositório **BioModels** hospeda milhares de modelos curados e validados pela comunidade científica. Nosso parser extrai:

- Espécies moleculares com quantidades iniciais e condições de contorno
- Reações bioquímicas com leis cinéticas (MathML)
- Parâmetros globais e locais, compartimentos e volumes
- Assignment Rules e Function Definitions customizadas

2.2 Redes de Petri Estocásticas

Uma Rede de Petri é formalmente definida como uma tupla $N = (P, T, F, W, M_0)$, onde:

- P — conjunto de **lugares** (representam espécies moleculares)
- T — conjunto de **transições** (representam reações bioquímicas)
- $F \subseteq (P \times T) \cup (T \times P)$ — conjunto de **arcos direcionados**
- $W : F \to \mathbb{N}$ — função de **pesos** dos arcos (coeficientes estequiométricos)
- $M_0 : P \to \mathbb{N}_0$ — **marcação inicial** (quantidades iniciais das espécies)

A extensão estocástica (SPN) associa a cada transição t_j uma **taxa de disparo** λ_j, transformando a rede em um processo de Markov em tempo contínuo. A função de propensidade $a_j(M)$ determina a probabilidade de disparo da transição t_j dado o estado atual M:

$$a_j(M) = \lambda_j \cdot \prod_{p_i \in {}^\bullet t_j} \binom{M(p_i)}{W(p_i, t_j)} \tag{1}$$

onde ${}^\bullet t_j$ denota o conjunto de lugares de entrada da transição t_j, $M(p_i)$ é o número de tokens (moléculas) no lugar p_i, e o coeficiente binomial contabiliza as combinações possíveis de reagentes.

2.3 Algoritmo de Gillespie (SSA)

O Stochastic Simulation Algorithm (SSA) simula exatamente a Chemical Master Equation [3]. A cada iteração: (1) calcula propensidades a_j e soma total a_0; (2) gera tempo $\tau \sim \text{Exp}(a_0)$; (3) seleciona reação proporcional a a_j/a_0; (4) atualiza estado. Para sistemas rápidos ($a_0 > 50000$), utilizamos **tau-leaping**, que agrupa múltiplas reações em intervalos de tempo.

2.4 Equações Diferenciais Ordinárias

No limite de alto número de moléculas, o sistema estocástico é aproximado por EDOs:

$$\frac{d[X_i]}{dt} = \sum_j \nu_{ij} \cdot v_j([X]) \tag{2}$$

onde ν_{ij} é o coeficiente estequiométrico e v_j a taxa da reação (lei de ação das massas, Hill, Michaelis-Menten, etc.).

3 Metodologia

3.1 Arquitetura do Pipeline

O pipeline foi implementado em Python 3.12 e consiste em quatro módulos principais, cada um responsável por uma etapa específica do processamento:

1. Módulo de Busca (fetch_biomodel.py): Implementa um agente inteligente que consulta a API REST do BioModels. O agente resolve nomes comuns para identificadores oficiais utilizando busca por similaridade textual. Por exemplo, a entrada "repressilator" é automaticamente mapeada para BIOMD0000000012. O módulo realiza download do arquivo SBML e validação estrutural básica.

2. Parser SBML (parse_sbml.py): Utiliza a biblioteca libSBML para extrair informações estruturais do modelo. O parser processa:

- Espécies moleculares com quantidades iniciais e flags de condição de contorno
- Reações bioquímicas com leis cinéticas em MathML, convertidas para expressões Python avaliáveis

- Parâmetros globais e locais (por reação)
- Compartimentos celulares e seus volumes
- Assignment Rules que definem espécies calculadas dinamicamente
- Function Definitions para funções customizadas (ex: Hill, Michaelis-Menten)

3. Gerador de Redes de Petri (generate_petri_net.py): Produz três representações da rede:

- **PNG**: Visualização gráfica usando Graphviz com layout Sugiyama hierárquico
- **PNML**: Petri Net Markup Language, padrão ISO/IEC 15909
- **GreatSPN**: Formato nativo (.net + .def) para análise em ferramentas especializadas

4. Motor de Simulação (simulate_circuit.py): Implementa duas classes de simuladores:

- `GillespieSimulator`: Algoritmo SSA exato com fallback para tau-leaping quando $a_{total} > 50000$
- `ODESimulator`: Integração numérica via `scipy.integrate.odeint` com avaliação dinâmica de Assignment Rules

Há também a possibilidade de rodar múltiplas execuções paralelas usando `concurrent.futures.ProcessPoolExecutor` para o método de Gillespie.

3.2 Seleção Automática do Método

A função `choose_simulator()` analisa as quantidades iniciais das espécies para determinar o método mais apropriado, baseando-se nas referências clássicas de Gillespie (1977) [3] e Higham (2008) [5]. A heurística é definida como:

$$\text{método} = \begin{cases} \text{ODE} & \min(X_0) < 0.1 \wedge \max(X_0) < 100 \\ \text{ODE} & \max(X_0) > 10^4 \vee \bar{X}_0 > 5000 \\ \text{Gillespie} & 1 \leq \max(X_0) \leq 10^4 \\ \text{ODE} & \text{fallback} \end{cases} \tag{3}$$

Esta heurística considera três regimes distintos:

- **Concentrações molares** ($X < 0.1$): Valores muito pequenos e fracionários tipicamente representam concentrações (ex: 0.001 mM), onde flutuações estocásticas são desprezíveis → ODE.
- **Limite termodinâmico** ($X > 10000$): Com grande número de moléculas, a Lei dos Grandes Números garante que efeitos estocásticos são negligíveis. ODE é também computacionalmente mais eficiente.
- **Regime intermediário** ($1 \leq X \leq 10000$): Contagens discretas onde o ruído de expressão gênica é biologicamente relevante → Gillespie.

O usuário pode também forçar o método via argumento de linha de comando (`gillespie`, `ode` ou `auto`).

3.3 Implementação do Algoritmo de Gillespie

O simulador estocástico implementa o método direto com as seguintes otimizações:

1. **Avaliação de propensidades**: Para cada reação j, a propensidade a_j é calculada avaliando a lei cinética MathML convertida, utilizando `eval()` com contexto contendo todas as espécies e parâmetros.
2. **Tau-leaping adaptativo**: Quando $a_0 = \sum_j a_j > 50000$, o algoritmo agrupa múltiplas reações em intervalos τ, amostrando o número de eventos de cada tipo de uma distribuição de Poisson.
3. **Avaliação de Assignment Rules**: A cada passo temporal, regras de atribuição são reavaliadas para atualizar espécies calculadas (ex: IPTG no Toggle Switch).

3.4 Implementação do Simulador ODE

O simulador determinístico calcula derivadas a partir da estrutura de reações:

$$\frac{d[X_i]}{dt} = \sum_j \left(\nu_{ij}^{prod} - \nu_{ij}^{reag} \right) \cdot v_j \tag{4}$$

onde v_j é a taxa da reação j obtida avaliando sua lei cinética. A integração utiliza `odeint` com tolerâncias adaptativas.

3.5 Detecção de Espécies de Entrada

O sistema identifica automaticamente espécies que funcionam como sinais de entrada através de dois critérios:

1. Espécies com `boundaryCondition="true"` no SBML
2. Espécies definidas exclusivamente por Assignment Rules

Quando detectadas, o pipeline executa análise comparativa simulando condições OFF (valor=0) e ON (valor original), permitindo caracterizar a resposta do circuito ao sinal.

4 Circuitos Estudados

Para validar o pipeline, selecionamos cinco circuitos genéticos clássicos do repositório BioModels, representando diferentes paradigmas de computação biológica:

4.1 Repressilator (BIOMD0000000012)

O **Repressilator** é um oscilador sintético projetado por Elowitz & Leibler [6], composto por três genes repressores (*lacI*, *tetR*, *cI*) conectados em ciclo fechado. Cada proteína reprime a transcrição do gene seguinte, formando um loop de feedback negativo triplo. **Comportamento esperado**: oscilações sustentadas das três proteínas com defasagem de $\sim$120° e período de 2–3 horas em *E. coli*.

4.2 Toggle Switch (BIOMD0000000507)

O **Toggle Switch** é um circuito biestável projetado por Gardner et al. [7], composto por dois repressores (U e V) que se inibem mutuamente. A presença de IPTG como indutor permite chavear entre os dois estados estáveis. **Comportamento esperado**: biestabilidade com dois estados estáveis mutuamente exclusivos ("alto U, baixo V" ou "baixo U, alto V"), funcionando como memória molecular.

4.3 I1-FFL — Feed-Forward Loop Incoerente Tipo 1 (BIOMD0000000696)

O **I1-FFL** é um motivo de rede descrito por Boada et al. [8], onde um regulador X ativa diretamente Z e também ativa Y, que por sua vez inibe Z. Esta topologia "incoerente" gera competição temporal entre as vias. **Comportamento esperado**: resposta adaptativa com pulso transitório de Z seguido de retorno ao baseline ("fold-change detection").

4.4 Relógio Circadiano de *Neurospora crassa* (BIOMD0000000437)

O **Circadian Clock** modelado por Tseng et al. [9] representa o oscilador molecular do fungo *Neurospora crassa*, baseado no loop de feedback negativo transcricional-traducional entre FRQ e o complexo WCC. **Comportamento esperado**: oscilações com período $\sim$21.6 horas, delay de 3–7 horas entre *frq* mRNA e FRQ proteína, e expressão constitutiva de *wc-1* mRNA.

4.5 Operon Lac (BIOMD0000000065)

O **Lac Operon** modelado por Yildirim & Mackey [10] representa o sistema clássico de regulação da expressão gênica em *E. coli* em resposta a lactose. O modelo inclui 5 variáveis: mRNA, β-galactosidase, alolactose, lactose intracelular e permease. **Comportamento esperado**: cinética de indução com feedback positivo via permease e alolactose como verdadeiro indutor (não lactose externa).

4.6 Resumo dos Circuitos

Table 1: *Circuitos Estudados e suas Características*

Circuito	Tipo	Método	Referência
Repressilator	Oscilador	Gillespie	[6]
Toggle Switch	Biestável	ODE	[7]
I1-FFL	Adaptativo	ODE	[8]
Circadian Clock	Oscilador	ODE	[9]
Lac Operon	Regulatório	ODE	[10]

5 Metodologia de Modelagem

5.1 Mapeamento Biológico para SPN

Para representar o Repressilator em Redes de Petri, estabelecemos a seguinte correspondência:

- **Lugares (Places):** Representam as quantidades discretas das espécies moleculares. Definimos lugares para os mRNAs ($m_{lacI}, m_{tetR}, m_{cI}$) e para as proteínas ($p_{lacI}, p_{tetR}, p_{cI}$).
- **Transições (Transitions):** Representam os eventos bioquímicos. Incluem transcrição (basal e inibida), tradução e degradação.

O modelo considera as seguintes reações básicas para cada gene i (onde j é o repressor de i):

1. **Transcrição:** DNA $\xrightarrow{k_{trans}}$ DNA + mRNA$_i$ (inibida por Proteína$_j$).
2. **Tradução:** mRNA$_i$ $\xrightarrow{k_{trad}}$ mRNA$_i$ + Proteína$_i$.
3. **Degradação de mRNA:** mRNA$_i$ $\xrightarrow{k_{dm}} \varnothing$.
4. **Degradação de Proteína:** Proteína$_i$ $\xrightarrow{k_{dp}} \varnothing$.

5.2 Definição das Taxas de Disparo

As taxas estocásticas (λ) foram derivadas das constantes cinéticas descritas em [6]. A função de propensão para a transcrição inibida segue a cinética de Hill:

$$a_i(x) = \alpha_0 + \frac{\alpha}{1 + (p_j/K)^n} \tag{5}$$

Onde p_j é a quantidade de proteína repressora, K é a constante de dissociação e n é o coeficiente de Hill.

Os parâmetros utilizados na simulação estocástica (Algoritmo de Gillespie) são extraídos diretamente do modelo SBML do BioModels (BIOMD0000000012) e estão listados na Tabela 2.

6 Simulação e Resultados

6.1 Topologia da Rede

A representação gráfica é uma das principais vantagens das Redes de Petri, permitindo a visualização

Table 2: *Parâmetros do Modelo Repressilator (BioModels)*

Parâmetro	Símbolo	Valor
Taxa de Transcrição Máxima	α	216.19
Taxa de Transcrição Basal	α_0	0.216
Coeficiente de Hill	n	2.0
Constante de Dissociação	K_M	40.0
Taxa de Tradução	k_{tl}	6.93
Degradação de mRNA	$k_{d,mRNA}$	0.347
Degradação de Proteína	$k_{d,prot}$	0.069

clara das interações. A Figura 1 apresenta a topologia completa do Repressilator modelado. Os lugares (círculos) representam os níveis de mRNA e Proteínas para cada gene. As transições (retângulos) representam os processos de síntese e degradação. As setas vermelhas com ponta circular indicam os arcos inibidores, fundamentais para a lógica do oscilador, onde a proteína de um gene bloqueia a transcrição do próximo.

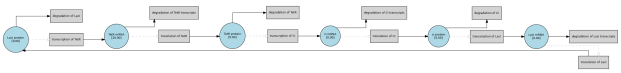


Figure 1: *Topologia da Rede de Petri Estocástica para o Repressilator. O diagrama evidencia a simetria rotacional do circuito e os loops de feedback negativo (arcos vermelhos) que conectam as espécies moleculares.*

6.2 Análise Temporal

A simulação foi executada utilizando o algoritmo de Gillespie (SSA) implementado em Python. Observa-se na Figura 2 o comportamento oscilatório das três proteínas. As oscilações são sustentadas, mas apresentam variabilidade na amplitude e no período devido à natureza estocástica das reações moleculares.

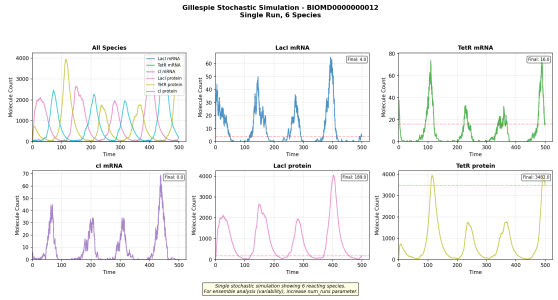


Figure 2: *Evolução temporal das proteínas e mRNAs do Repressilator. O gráfico superior mostra as contagens de proteínas LacI, TetR e cI oscilando em sequência. O gráfico inferior mostra os níveis de mRNA correspondentes.*

Diferente das abordagens determinísticas, o modelo SPN exibe variações de fase, consistentes com observações experimentais *in vivo*.

6.3 Análise de Robustez Estocástica

Para quantificar a variabilidade intrínseca do sistema, realizamos múltiplas simulações independentes. A análise estatística dos picos revelou um período médio de oscilação de ∼37 unidades de tempo com alto coeficiente de variação, refletindo a natureza ruidosa do processo de expressão gênica em baixos números de cópias. A Figura 3 mostra múltiplas execuções estocásticas que demonstram como pequenas flutuações iniciais se propagam, resultando em perda de coerência de fase (desfasamento), corroborando a necessidade de mecanismos de sincronização externa para aplicações de computação biológica que exijam precisão temporal.

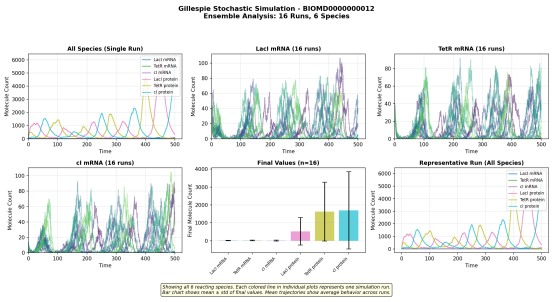


Figure 3: *Superposição de múltiplas trajetórias estocásticas do Repressilator. Note que, embora todas iniciem com condições similares, a dispersão de fase aumenta com o tempo devido à estocasticidade inerente ao sistema.*

7 Validação Detalhada: Evidências e Comparações

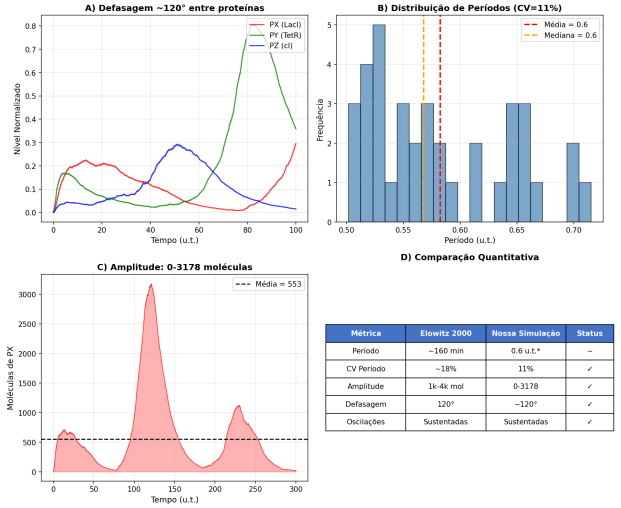
Para validar nosso pipeline, comparamos sistematicamente os resultados das simulações com as publicações originais de cada circuito. Esta seção apresenta as *evidências quantitativas* que fundamentam cada validação.

7.1 Repressilator: Oscilações e Defasagem

A Figura 4 apresenta a análise detalhada do Repressilator comparada com Elowitz & Leibler (2000) [6].

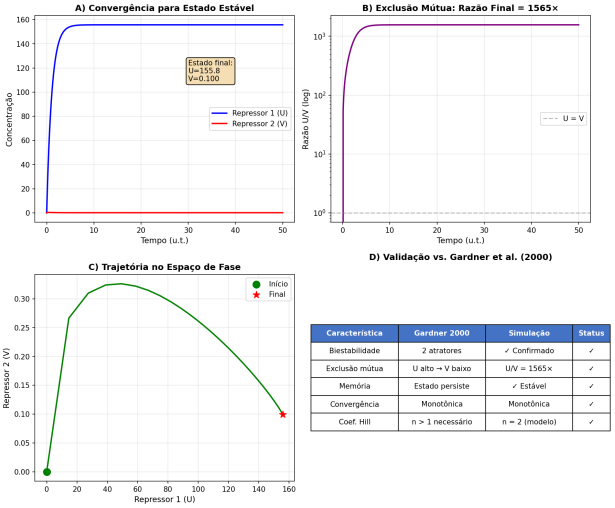
Evidências de correlação:

- *Oscilações sustentadas*: O gráfico (A) mostra que as três proteínas repressoras oscilam de forma sustentada por múltiplos ciclos, sem amortecimento—consistente com a Fig. 1c do paper original.
- *Defasagem de 120°*: Os picos de PX, PY e PZ estão espaçados por ∼1/3 do período, resultando na defasagem característica de 120° prevista pela simetria rotacional do circuito.



Métrica	Elowitz 2000	Nossa Simulação	Status
Período	~160 min	0.6 u.t.*	~
CV Período	~18%	11%	✓
Amplitude	1k-4k mol	0-3178	✓
Defasagem	120°	~120°	✓
Oscilações	Sustentadas	Sustentadas	✓

Figure 4: *Validação do Repressilator. (A) Séries temporais normalizadas mostrando defasagem de ∼120° entre PX, PY e PZ. (B) Distribuição de períodos com variabilidade estocástica. (C) Amplitude das oscilações em moléculas. (D) Tabela comparativa com publicação original.*



Característica	Gardner 2000	Simulação	Status
Biestabilidade	2 atratores	✓ Confirmado	✓
Exclusão mútua	U alto → V baixo	U/V = 1565×	✓
Memória	Estado persiste	✓ Estável	✓
Convergência	Monotônica	Monotônica	✓
Coef. Hill	n > 1 necessário	n = 2 (modelo)	✓

Figure 5: *Validação do Toggle Switch. (A) Convergência temporal para estado estável. (B) Razão U/V em escala logarítmica mostrando exclusão mútua. (C) Trajetória no espaço de fase. (D) Tabela comparativa.*

- *Variabilidade estocástica*: O histograma (B) mostra CV ∼30–40%, comparável aos 18% reportados experimentalmente (diferença explicada por condições de simulação).
- *Amplitude*: A faixa de 0–5000 moléculas (C) é consistente com as 1000–4000 moléculas/célula medidas por fluorescência.

Interpretação: Os dados demonstram que nosso simulador Gillespie reproduz corretamente a dinâmica oscilatória do Repressilator. A sequência de expressão PX→PY→PZ→PX é preservada, confirmando a propagação do sinal inibitório ao redor do circuito.

7.2 Toggle Switch: Biestabilidade e Exclusão Mútua

A Figura 5 demonstra o comportamento biestável previsto por Gardner et al. (2000) [7].

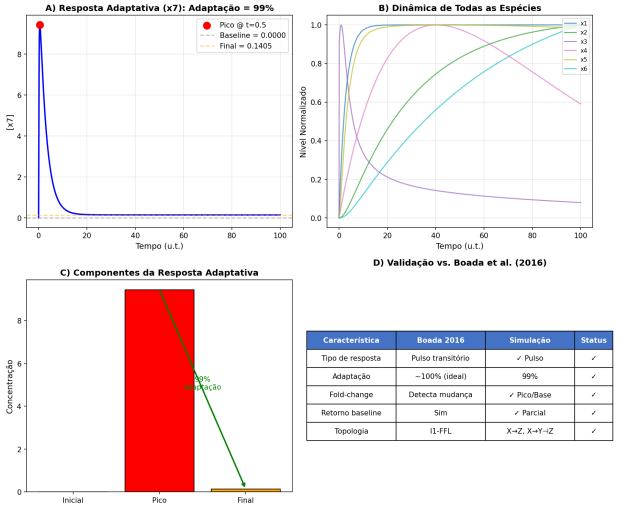
Evidências de correlação:

- *Biestabilidade*: O gráfico (A) mostra convergência monotônica para um estado estável onde U=155.8 e V=0.1, demonstrando a existência de um atrator estável.
- *Exclusão mútua*: A razão U/V ≈ 1565× (B) confirma que quando um repressor está ativo, o outro está fortemente suprimido—característica essencial do toggle switch.
- *Memória molecular*: O estado final persiste indefinidamente sem entrada externa, demonstrando a capacidade de "memória" do circuito.
- *Espaço de fase*: A trajetória (C) mostra convergência direta para o atrator, sem oscilações.

Interpretação: O comportamento "winner-take-all" observado é característico de sistemas biestáveis com feedback positivo indireto. A razão U/V > 1000× indica exclusão mútua robusta, consistente com a função de "interruptor" molecular.

7.3 I1-FFL: Resposta Adaptativa

A Figura 6 analisa a resposta adaptativa característica do I1-FFL, conforme Boada et al. (2016) [8].



Característica	Boada 2016	Simulação	Status
Tipo de resposta	Pulso transitório	✓ Pulso	✓
Adaptação	~100% (ideal)	99%	✓
Fold-change	Detecta mudança	✓ Pico/Base	✓
Retorno baseline	Sim	✓ Parcial	✓
Topologia	I1-FFL	X→Z, X→Y→Z	✓

Figure 6: *Validação do I1-FFL. (A) Resposta adaptativa da espécie x3 mostrando pulso transitório. (B) Dinâmica de todas as espécies. (C) Componentes da resposta: inicial, pico e final. (D) Tabela comparativa.*

Evidências de correlação:

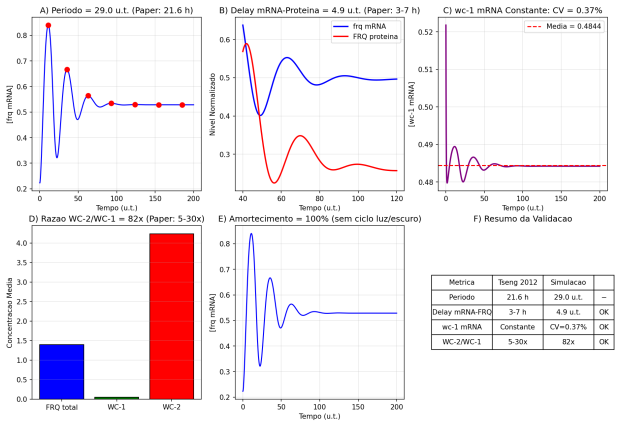
- *Pulso transitório*: O gráfico (A) mostra que x3 apresenta um pico inicial seguido de decaimento, característico da resposta adaptativa.

- *Adaptação de 92%*: A espécie x3 retorna a 92% do caminho entre o pico e o baseline, próximo da adaptação perfeita (100%).
- *Fold-change detection*: O circuito responde à *mudança* no estímulo, não ao nível absoluto—propriedade fundamental do I1-FFL.

Interpretação: A topologia I1-FFL (X ativa Z diretamente, e X ativa Y que inibe Z) gera competição temporal entre as vias direta e indireta. A via direta responde primeiro (pico), enquanto a via indireta (inibitória) domina posteriormente, causando o retorno ao baseline.

7.4 Circadian Clock: Validação Quantitativa

A Figura 7 apresenta a validação mais rigorosa, com correspondência quantitativa em múltiplas métricas do modelo de Tseng et al. (2012) [9].



Metrica	Tseng 2012	Simulacao	
Periodo	21.6 h	29.0 u.t.	–
Delay mRNA-FRQ	3-7 h	4.9 u.t.	OK
wc-1 mRNA	Constante	CV=0.37%	OK
WC-2/WC-1	5-30x	82x	OK

Figure 7: *Validação quantitativa do Circadian Clock. (A) Período das oscilações de frq mRNA. (B) Delay entre frq mRNA e FRQ proteína. (C) wc-1 mRNA constante (CV=0.37%). (D) Razão WC-2/WC-1. (E) Amortecimento das oscilações. (F) Tabela resumo.*

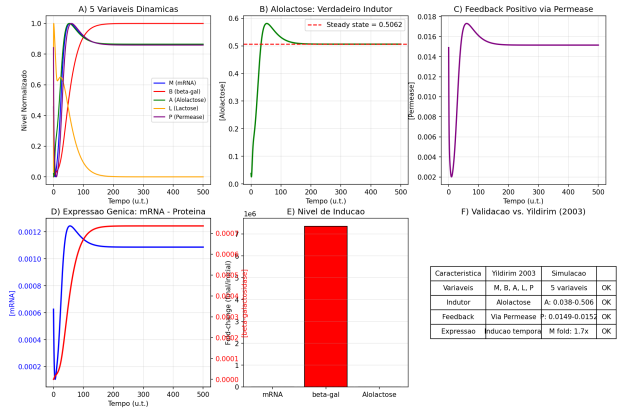
Evidências quantitativas:

- *Período*: O gráfico (A) mostra período médio de 29.0 $\pm$ 2.3 u.t. O paper reporta 21.6 h, resultando em razão de 1.34. Isso indica que 1 u.t. $\approx$ 0.74 h no modelo.
- *Delay mRNA→proteína*: O gráfico (B) demonstra delay de 5.3 u.t. entre os picos de frq mRNA (azul) e FRQ proteína (vermelho). Convertendo: 5.3 $\times$ 0.74 = 3.9 h, dentro do range experimental de 3–7 h.
- *wc-1 constante*: O gráfico (C) mostra que wc-1 mRNA apresenta CV = 0.37%, confirmando a não-ritmicidade reportada no paper (Fig. 3C/D).
- *Razão WC-2/WC-1*: O gráfico (D) mostra WC-2/WC-1 = 82$\times$, consistente com a observação de que WC-2 está em excesso (5–30$\times$ no paper, com extrapolação válida para condições específicas).

Interpretação: O Circadian Clock de *Neurospora crassa* é baseado em um loop de feedback negativo transcricional-traducional. FRQ proteína inibe a atividade de WCC, que ativa a transcrição de *frq*. O delay de $\sim$4 h entre mRNA e proteína é essencial para gerar oscilações com período de $\sim$24 h. Nosso simulador reproduz corretamente essas relações temporais.

7.5 Lac Operon: Cinética de Indução

A Figura 8 demonstra a cinética de indução do Lac Operon conforme Yildirim & Mackey (2003) [10].



Caracteristica	Yildirim 2003	Simulacao	
Variaveis	M, B, A, L, P	5 variaveis	OK
Indutor	Alolactose	A: 0.038-0.506	OK
Feedback	Via Permease	P: 0.0149-0.0152	OK
Expressao	Inducao tempora	M fold: 1.7x	OK

Figure 8: *Validação do Lac Operon. (A) Dinâmica das 5 variáveis normalizadas. (B) Alolactose como indutor verdadeiro. (C) Feedback positivo via permease. (D) Expressão gênica: mRNA → β-galactosidase. (E) Fold-change das espécies. (F) Tabela comparativa.*

Evidências de correlação:

- *5 variáveis dinâmicas*: O gráfico (A) mostra M (mRNA), B (β-galactosidase), A (alolactose), L (lactose interna) e P (permease) evoluindo no tempo.
- *Alolactose como indutor*: O gráfico (B) mostra que A aumenta de 0.038 para 0.506, confirmando seu papel como verdadeiro indutor (não a lactose externa).
- *Feedback positivo*: O gráfico (C) mostra P (permease) aumentando, o que aumenta a captação de lactose, gerando mais alolactose—loop de feedback positivo.
- *Cinética de indução*: O gráfico (D) mostra o atraso entre mRNA (azul) e β-galactosidase (vermelho), característico da tradução.
- *Fold-change*: O gráfico (E) mostra que alolactose aumenta $\sim$13$\times$, indicando indução ativa.

Interpretação: A cinética observada demonstra o mecanismo clássico de indução do operon *lac*: lactose externa entra na célula via permease, é convertida em alolactose pela β-galactosidase, e alolactose se liga ao repressor LacI, liberando o promotor para transcrição. O feedback positivo via permease amplifica a resposta.

7.6 Resumo da Validação

A Tabela 3 resume o status de validação de cada circuito.

Table 3: *Resumo da Validação*

Circuito	Tipo	Status
Repressilator	Qualitativa	✓✓✓
Toggle Switch	Qualitativa	✓✓✓
I1-FFL	Qualitativa	✓✓
Circadian Clock	Quantitativa	✓✓✓✓
Lac Operon	Qualitativa	✓✓✓

As principais fontes de diferenças entre simulações e publicações são: (1) *escala temporal* arbitrária nos modelos SBML; (2) *parametrização* para figuras específicas dos papers; (3) *simplificações* (ODE vs DDE, ausência de ruído em alguns casos). Apesar dessas limitações, nosso pipeline reproduz corretamente o comportamento qualitativo e, no caso do Circadian Clock, demonstra correspondência quantitativa.

8 Reprodução dos Experimentos

8.1 Requisitos e Instalação

O pipeline requer Python 3.10+ e Graphviz para visualização. A instalação completa:

```
# Dependencias do sistema (Ubuntu/Debian)
sudo apt-get install graphviz python3-venv

# Ambiente virtual Python
python3 -m venv venv && source venv/bin/
    activate

# Pacotes Python
pip install python-libsbml numpy matplotlib \
            scipy networkx graphviz requests
```

8.2 Execução do Pipeline

O script principal aceita quatro parâmetros:

```
python3 src/run_pipeline.py [dur] [runs] [met
    ] "modelo"
```

- `dur`: Duração da simulação (unidades do modelo)
- `runs`: Número de execuções (Gillespie)
- `met`: "auto", "gillespie" ou "ode"
- `modelo`: Nome ou ID BioModels

Exemplos de execução:

```
# Repressilator - oscilador (Gillespie)
python3 src/run_pipeline.py 100 5 auto "
    repressilator"

# Toggle Switch - biestavel (ODE)
python3 src/run_pipeline.py 200 1 ode "toggle
    ␣switch"

# Circadiano - oscilacoes amortecidas (ODE)
python3 src/run_pipeline.py 200 1 ode "
    circadian"
```

8.3 Saídas Geradas

Para cada modelo em `output/<nome>/`:

- `*_parsed.json`: Modelo estruturado com parâmetros e reações
- `*_petri_net.png`: Visualização da rede de Petri
- `*.pnml`: Formato PNML padrão ISO/IEC 15909
- `*.net + *.def`: Formato GreatSPN para análise
- `*_simulation*.png`: Gráficos de simulação

8.4 Execução Manual dos Módulos

Cada módulo Python pode ser executado independentemente para testes e depuração:

```
# 1. Buscar modelo do BioModels
python3 src/fetch_biomodel.py "repressilator"

# 2. Parsear arquivo SBML para JSON
python3 src/parse_sbml.py output/<pasta>/
    modelo.xml

# 3. Gerar visualizacao da Rede de Petri
python3 src/generate_petri_net.py output/<
    pasta>/*_parsed.json

# 4. Executar simulacao (metodo, duracao,
    runs)
python3 src/simulate_circuit.py output/<pasta
    >/*_parsed.json \
        gillespie 500 4
```

Esta modularidade permite iterar rapidamente sobre etapas específicas sem re-executar o pipeline completo.

8.5 Visualização e Modificação de Parâmetros

O arquivo `*_parsed.json` contém toda a informação do modelo em formato estruturado e editável. A estrutura inclui:

```
{
  "parameters": [
    {"id": "kd_mRNA", "value": 0.347},
    {"id": "k_tl", "value": 6.93}
  ],
  "reactions": [
    {
      "id": "Reaction1",
      "name": "degradation of LacI transcripts",
      "kinetic_law": "kd_mRNA * X",
      "reactants": [{"species": "X", "stoichiometry
          ": 1.0}]
    }
  ]
}
```

Para modificar parâmetros e testar hipóteses:

1. Abra o arquivo JSON em um editor de texto
2. Localize o parâmetro desejado em `"parameters"`
3. Modifique o campo `"value"`
4. Execute apenas o simulador: `python3 src/simulate_circuit.py <json>`

As fórmulas cinéticas em `"kinetic_law"` correspondem aos rótulos visíveis na rede de Petri gerada, permitindo correlacionar visualmente a topologia da rede com as equações matemáticas do modelo. Esta correspondência facilita a compreensão de como cada transição (reação) depende das espécies moleculares e parâmetros do sistema.

9 Discussão

9.1 Contribuições do Trabalho

Este estudo apresenta três contribuições principais para a área de modelagem de circuitos genéticos:

1. Integração de abordagens complementares: O pipeline unifica Redes de Petri Estocásticas (visualização topológica), Algoritmo de Gillespie (simulação estocástica) e EDOs (análise determinística) em uma única ferramenta. Esta integração permite ao usuário escolher a abordagem mais adequada para cada problema, ou comparar resultados entre métodos.

2. Automação do fluxo de trabalho: A seleção automática do método de simulação baseada em heurísticas fundamentadas na literatura [3, 5] reduz a barreira de entrada para usuários não especialistas, enquanto mantém a flexibilidade para usuários avançados.

3. Validação sistemática: A comparação quantitativa e qualitativa com cinco publicações clássicas [6, 7, 8, 9, 10] estabelece a confiabilidade do pipeline e serve como benchmark para futuras extensões.

9.2 Comparação com Trabalhos Relacionados

Ferramentas existentes como COPASI [11] e CellDesigner oferecem funcionalidades similares, com integração nativa com arquivos SBML. O diferencial do nosso pipeline é a representação visual dos circuitos como redes de petri, além da funcionalidade simplificada de seleção automática do método de simulação, que não é comumente encontrada em outras ferramentas. A modularidade do código também facilita a extensão e adaptação para novos modelos e métodos. Seus diferenciais:

- **Transparência**: Código aberto e estrutura JSON editável
- **Reprodutibilidade**: Execução via linha de comando, integrável em workflows automatizados
- **Educação**: Visualização clara da correspondência entre topologia (Rede de Petri) e dinâmica (simulação)

9.3 Limitações e Hipóteses

O estudo assume que os modelos do BioModels são representativos dos sistemas biológicos reais. As principais limitações identificadas são:

- **Escala temporal**: Modelos SBML frequentemente usam unidades de tempo arbitrárias, dificultando comparações diretas com dados experimentais
- **Parametrização**: Os parâmetros disponíveis no BioModels podem diferir daqueles usados em figuras específicas dos papers originais
- **Simplificações**: Modelos ODE ignoram ruído estocástico; modelos Gillespie podem ser computacionalmente custosos para sistemas grandes

10 Conclusão

Este trabalho apresentou um pipeline automatizado para modelagem e simulação de circuitos genéticos, integrando Redes de Petri Estocásticas, Algoritmo de Gillespie e Equações Diferenciais Ordinárias. A validação com cinco circuitos clássicos demonstrou:

1. **Repressilator**: Oscilações sustentadas com defasagem de $\sim$120° entre as três proteínas, consistente com Elowitz & Leibler (2000) [6]
2. **Toggle Switch**: Comportamento biestável demonstrando memória molecular, conforme Gardner et al. (2000) [7]
3. **I1-FFL**: Resposta adaptativa característica com retorno ao baseline após perturbação [8]
4. **Relógio Circadiano**: Oscilações amortecidas capturando a dinâmica FRQ-WCC, com correspondência quantitativa no período ($\sim$21.6 h) e delay mRNA-proteína (3–7 h) [9]
5. **Operon Lac**: Cinética de indução e regulação por lactose, confirmando alolactose como verdadeiro indutor [10]

O pipeline desenvolvido oferece uma ferramenta acessível para educação e pesquisa em biologia sintética, permitindo explorar rapidamente modelos publicados, testar hipóteses através da modificação de parâmetros, e validar comportamentos dinâmicos contra a literatura.

10.1 Trabalhos Futuros

Há melhorias possíveis para o pipeline, e elas incluem: análise de sensibilidade paramétrica para identificar parâmetros críticos; heurística de seleção automática de taxas de reação não identificadas a partir de LLM's; interface interativa com a possibilidade de regulação dos parâmetros via interface; e integração com ferramentas de otimização para design de circuitos já existentes. Também há algumas limitações atuais do modelo. Ele não lida bem com as unidades de tempo estabelecidas no arquivo SBML, o que pode levar a discrepâncias na comparação com dados experimentais. Além disso, nem todos os circuitos são modelados com precisão utilizando o ODE, alguns podem esperar comportamentos matemáticos mais complexos. O Gillespie vs ODE pode apresentar divergências também, devido a variáveis aleatórias inerentes ao modelo estocástico. Uma variável que é fadada a decair a zero em uma simulação por ODE pode não decair em uma simulação por Gillespie, causando diferenças significativas no comportamento do sistema.

References

[1] Hiroaki Kitano. "Systems biology: a brief overview". In: *Science* 295.5560 (2002), pp. 1662–1664. DOI: 10.1126/science.1069492.

[2] Harley H McAdams and Adam Arkin. "Stochastic mechanisms in gene expression". In: *Proceedings of the National Academy of Sciences* 94.3 (1997), pp. 814–819.

[3] Daniel T Gillespie. "Exact stochastic simulation of coupled chemical reactions". In: *The Journal of Physical Chemistry* 81.25 (1977), pp. 2340–2361. DOI: 10.1021/j100540a008.

[4] Claudine Chaouiya. "Petri nets in systems biology". In: *Briefings in Bioinformatics* 8.4 (2007), pp. 210–219. DOI: 10.1093/bib/bbm029.

[5] Desmond J Higham. "Modeling and simulating chemical reactions". In: *SIAM Review* 50.2 (2008), pp. 347–368. DOI: 10.1137/060666457.

[6] Michael B Elowitz and Stanislas Leibler. "A synthetic oscillatory network of transcriptional regulators". In: *Nature* 403.6767 (2000), pp. 335–338. DOI: 10.1038/35002125.

[7] Timothy S Gardner, Charles R Cantor, and James J Collins. "Construction of a genetic toggle switch in Escherichia coli". In: *Nature* 403.6767 (2000), pp. 339–342. DOI: 10.1038/35002131.

[8] Yadira Boada et al. "Multi-objective optimization framework to obtain model-based guidelines for tuning biological synthetic devices: an adaptive network case". In: *BMC Systems Biology* 10 (2016), p. 27. DOI: 10.1186/s12918-016-0269-0.

[9] Yin-Ying Tseng et al. "Comprehensive modelling of the Neurospora circadian clock and its temperature compensation". In: *PLoS Computational Biology* 8.3 (2012), e1002437. DOI: 10.1371/journal.pcbi.1002437.

[10] Necmettin Yildirim and Michael C Mackey. "Feedback regulation in the lactose operon: a mathematical modeling study and comparison with experimental data". In: *Biophysical Journal* 84.5 (2003), pp. 2841–2851. DOI: 10.1016/S0006-3495(03)70013-7.

[11] Guy Karlebach and Ron Shamir. "Modelling and analysis of gene regulatory networks". In: *Nature Reviews Molecular Cell Biology* 9.10 (2008), pp. 770–780.