Desenvolvimento de um software científico para a modelagem e simulação computacional

Dissertação de Mestrado

Mestrado em Ciência da Computação

Brenno Lemos Melquiades dos Santos

Orientador: Prof. Alexandre B. Pigozzo

24 de março de 2024

Universidade Federal de São João del-Rei

UFSJ

Sumário

1 Introdução

Introdução

- O desenvolvimento de modelos computacionais requer um conjunto de etapas: estudo do problema, formulação de hipóteses, construção, implementação e simulação do modelo.
- Uma das etapas mais desafiadoras é a implementação.
  - Requer o conhecimento de programação, estrutura de dados, bibliotecas, etc;
  - Um erro na implementação pode comprometer todo o trabalho.
- Questão científica:
  - É possível que ferramentas de software automatizem etapas do processo de modelagem computacional?

- Para responder a questão anterior, foi desenvolvido um software para auxiliar a implementação e simulação de modelos de Equações Diferenciais Ordinárias (EDOs) com o objetivo de automatizar certas etapas do processo de modelagem.
- A partir de uma representação visual de um modelo, o software é capaz de gerar o código que implementa o modelo, simulá-lo e até exportar gráficos com os resultados.

UFSJ

Sumário

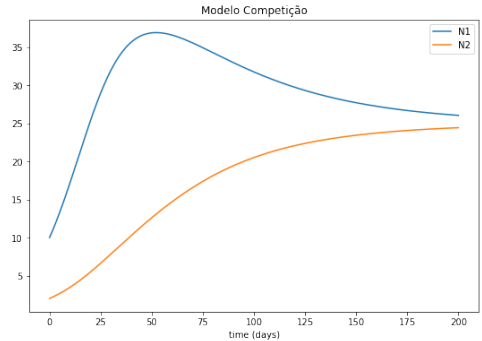
2 Referencial teórico

Equações Diferenciais Ordinárias (EDOs)

- Usadas para estudar o comportamento populacional ao longo do tempo;
- Diversas aplicações em várias áreas do conhecimento;
- Cada equação descreve a concentração de uma população diferente;

$$\frac{dN_1}{dt} = r_1.N_1(1 - W_{11}.N_1 - W_{21}.N_2)$$

$$\frac{dN_2}{dt} = r_2.N_2(1 - W_{22}.N_2 - W_{12}.N_1) \quad (1)$$



EDO — Modelo Predador-Presa

2 Referencial teórico

Um modelo clássico da literatura é o modelo Predador-Presa. Este modelo descreve o comportamento de duas populações, H e P, que possuem uma uma relação de predação entre si.

$$\begin{aligned} \frac{dH}{dt} &= r.H - a.H.P \\ \frac{dP}{dt} &= b.H.P - m.P \end{aligned} \tag{2}$$

Na equação, temos que

H	Presa
P	Predador
r	Taxa de reprodução da presa
m	Taxa de mortalidade dos predadores
a	Taxa de predação
b	Taxa de reprodução dos predadores

.

Programação visual

- É uma maneira do usuário programar a máquina por meio de elementos gráficos que abstraem instruções do computador.
- Os elementos podem representar múltiplas operações por vez, com o objetivo de facilitar a programação.
- Exemplos: GRAIL, Scratch, Logisim, Blender.
- Um exemplo comum de aplicação envolve editores baseados em nós, que representam operações complexas de forma natural, combinando um conjunto de entradas para gerar uma ou mais saídas.

- GRaIL — 1968:

Geração de código

- Recebe como entrada uma Representação Intermediária (RI) e gera como saída um código na linguagem alvo (por exemplo, Python).
- Aplicações da RI:
  - Separar o *front-end* do compilador do *back-end*;
  - Permitir que sejam realizadas otimizações independente de máquina ou otimizações independente da linguagem alvo;
  - Para facilitar a tradução e geração do código alvo.
- Exemplo: LLVM IR, usada originalmente para compilar códigos em C/C++ pelo Clang, e agora também usada na compilação de códigos em Rust.

Geração de código baseada em *templates*

- Um *template* é um esqueleto (uma estrutura) que serve de referência para todo o processo de geração de código.
- Em sua essência, *templates* são arquivos com marcadores especiais que podem ser substituídos por outros valores dinamicamente.
- A presença de estruturas de controle de fluxo como condicionais e laços de repetição permitem a escrita de *templates* legíveis e de fácil manutenção.

Geração de código baseada em *templates*


```
def system(t: np.float64, y: np.ndarray, *constants) -> np.ndarray:
    {% for arg in populations -%}
        {{- arg.name }}, {%- endfor %} = y

    {%- if constants %}
    {% for arg in constants -%}
        {{- arg.name }}, {%- endfor %} = constants
    {% endif -%}

    {% for pop in populations %}
    {%- set comp = model.arguments[equations[pop.name].argument] %}
    d{{ pop.name }}_dt = {{ display_composite(comp) }}
    {%- endfor %}

    # ...
```

UFSJ

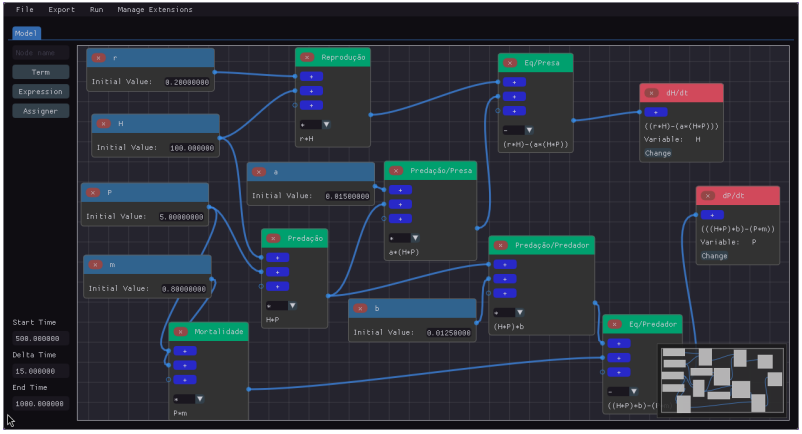
Sumário

3 Software para modelagem e simulação

Visão geral do software

3 Software para modelagem e simulação

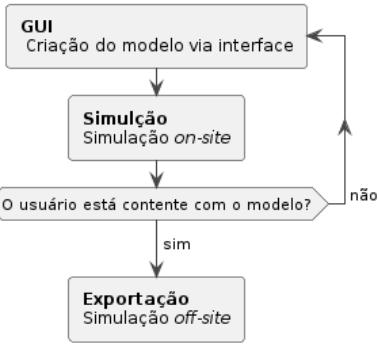
- O software desenvolvido apresenta uma GUI contendo um editor baseado em nós.
- Os nós representam partes das equações, como parâmetros, populações e expressões



Visão geral do software

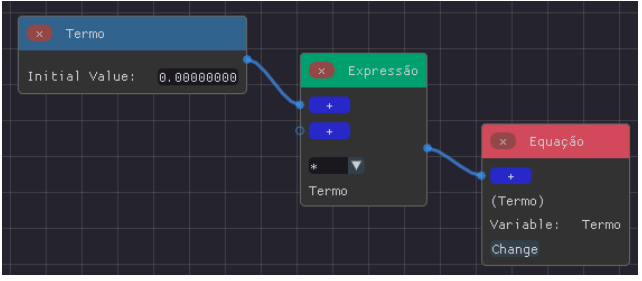
3 Software para modelagem e simulação

Após construído o modelo, o usuário poderá simulá-lo diretamente pela GUI, exportar um PDF dos resultados, ou exportar um código de Python equivalente para o modelo desenhado



Representação visual

Tipo de nó	Usado para representar	Exemplo
Termo	Variáveis, parâmetros e constantes.	H, P, a, b, r, m
Expressão	Expressões matemáticas que compõem as equações.	$r.H$, $a.H.P$, $r.H - a.H.P$;
Equação	O lado direito de uma EDO.	$dH/dt = \ldots$



UFSJ

Sumário

4 Aplicações do software

UFSJ

Sumário

5 Conclusões e trabalhos futuros

Conclusões e trabalhos futuros

- Neste trabalho, foi desenvolvido um software para automatizar a implementação e simulação de modelos computacionais baseados em EDOs.
- A construção das equações do modelo matemático é auxiliada pela representação visual que foi criada permitindo que o usuário acompanhe a construção de todas as expressões e como elas estão sendo combinadas para formar o sistema de EDOs.
- Através da GUI, é possível ver as entradas e operações de cada expressão, os sinais de cada entrada, quais expressões fazem parte de uma determinada EDO, entre outras coisas.

Conclusões e trabalhos futuros

- Como limitações do trabalho, destaca-se:
  - A representação visual apresenta uma limitação na qual tornar-se mais difícil entender um modelo complexo com muitos nós e ligações.
  - Não foi realizada uma avaliação de usabilidade do software.

Conclusões e trabalhos futuros

- Como trabalhos futuros, destaca-se:
  - Geração de código e simulação de modelos estocásticos;
  - Ajuste de parâmetros;
  - Análise de sensibilidade de parâmetros;
  - Geração de código e simulação de Equações Diferenciais Parciais (EDPs);
  - Desenvolvimento de uma versão Web do software.

UFSJ

Desenvolvimento de um software científico para a modelagem e simulação computacional