Documentação do Código do Simulador de Elevadores: Modelagem, Estruturas de Dados e Heurísticas

João Lucas Martins Silva Carvalho
Instituto de Ensino Superior - ICEV
Curso de Engenharia de Software
Email: joao.lucas@somosicev.com

João Tiago Lima Carvalho
Instituto de Ensino Superior - ICEV
Curso de Engenharia de Software
Email: joao.tiago@somosicev.com

May 2025

Abstract

Este documento descreve a modelagem e implementação de um sistema de simulação de elevadores inteligentes utilizando Java e estruturas de dados personalizadas. O foco é educacional, promovendo o uso de listas, filas e algoritmos de controle em tempo discreto.

1 Introdução

O Simulador de Elevadores Inteligentes é um projeto acadêmico que visa replicar o comportamento de sistemas de transporte vertical em edifícios. A aplicação é desenvolvida em Java, utilizando princípios da programação orientada a objetos, estruturas de dados manuais e simulação discreta. O objetivo principal é estudar o impacto de diferentes heurísticas no desempenho de um sistema multi-elevador.

2 Modelagem do Sistema

2.1 Diagrama Geral de Classes (Resumo)

- `EntidadeSimulavel` (interface): base para atualização temporal.
- `Andar`, `Elevador`, `CentralDeControle`, `Predio`, `Simulador`: implementam `EntidadeSimulavel`.
- `Pessoa`, `PainelElevador`, `PainelInternoElevador`, `PainelConfig`, `Lista`, `Fila`, `Ponteiro`, `SimuladorGUI`, `TelaConfiguracoes`, `TelaEstatisticas`.

3 Descrição das Principais Classes

- `Simulador`: executa ciclos temporais, gerencia configurações e coordena a simulação.
- `Predio`: estrutura principal contendo os andares e a central de controle.
- `CentralDeControle`: coordena a lógica de alocação dos elevadores com base em heurísticas.
- `Elevador`: executa movimentação, embarque e desembarque de passageiros.
- `Andar`: mantém fila de espera e interações com o painel de chamada.
- `Pessoa`: possui dados como origem, destino, prioridade e tempos.

4 Estruturas de Dados Utilizadas

4.1 Lista Encadeada

Implementação manual com inserção e remoção em ambos os extremos.

- Usada em: andares, elevadores, destinos nos painéis.
- Métodos principais: `inserirFim`, `obterElemento`, `tamanho`, `estaVazia`, `getInicio`.

4.2 Fila (FIFO)

Especialização da lista com inserção no final e remoção no início.

- Usada para: fila de pessoas aguardando elevador e passageiros dentro do elevador.
- Métodos principais: `inserirFim`, `removerInicio`, `tamanho`, `estaVazia`, `getInicio`.

4.3 Ponteiro

Classe que representa um nó da `Lista` ou `Fila`, contendo um objeto e um ponteiro para o próximo.

5 Algoritmos Implementados

5.1 Atualização do Sistema

O método `atualizar(int minutoSimulado)` é invocado a cada ciclo para todos os objetos simuláveis (`Andar`, `Elevador`, `CentralDeControle`).

5.2 Movimento do Elevador

- Se parado: verifica chamadas pendentes e inicia movimentação.
- Se em movimento: atualiza posição com base no tempo por andar e decide parar para embarque/desembarque.

5.3 Heurísticas

- `Ordem de Chegada (1)`: FCFS (First Come, First Served). Elevadores atendem chamadas na ordem em que surgem.
- `Otimizando Tempo (2)`: Escolhe o elevador que minimiza o tempo de espera, priorizando pessoas com necessidades especiais.
- `Otimizando Energia (3)`: Escolhe o elevador mais próximo para reduzir o consumo de energia.

5.4 Estatísticas

- Total de pessoas atendidas (com e sem prioridade).
- Tempo médio e máximo de espera.
- Consumo de energia por elevador e total.
- Média de pessoas por chamada.

6 Interface Gráfica

A interface gráfica foi desenvolvida em Java Swing e exibe:

- A posição e ocupação dos elevadores.
- As filas de espera por andar.
- Estatísticas atualizadas em tempo real (via `TelaEstatisticas`).
- Configurações ajustáveis (via `TelaConfiguracoes`).

7 Considerações Finais

O projeto serve como ferramenta de aprendizado em estruturas de dados e simulação. Sua modularidade permite fácil expansão para incluir novas heurísticas, limites de peso e algoritmos mais avançados.

8 Agradecimentos

Agradecemos ao corpo docente do curso de Engenharia de Software do ICEV pelo suporte técnico e teórico ao longo do desenvolvimento do projeto.

References

[1] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 3rd ed. Cambridge, MA: MIT Press, 2009.

[2] R. Lafore, *Data Structures and Algorithms in Java*, 2nd ed. Indianapolis, IN: Sams Publishing, 2002.

[3] M. T. Goodrich, R. Tamassia, and M. H. Goldwasser, *Data Structures and Algorithms in Java*, 6th ed. Hoboken, NJ: Wiley, 1994.

[4] E. Horowitz, S. Sahni, and D. Mehta, *Fundamentals of Data Structures in C++*, 2nd ed. Hyderabad, India: Universities Press, 2008.

[5] D. E. Knuth, *The Art of Computer Programming, Vol. 1: Fundamental Algorithms*, 3rd ed. Boston, MA: Addison-Wesley, 1997.