O Problema do Jantar dos Filósofos: Uma Abordagem Estatística

Marcos Mello
João Vitor Gonçalves
Pedro Vinicius

16 de setembro de 2025

Resumo

Este trabalho apresenta uma análise estatística do clássico problema do *Jantar dos Filósofos*, proposto por Dijkstra, visando avaliar o desempenho de algoritmos de sincronização por meio de métricas como tempo de espera, número de refeições e equidade entre processos. Além disso, discutimos limitações e possíveis extensões do estudo.

1 Introdução

O problema do Jantar dos Filósofos é um dos exemplos mais conhecidos de problemas de concorrência e sincronização em sistemas operacionais.

Neste trabalho, adotamos uma abordagem estatística para avaliar o desempenho do algoritmo implementado, analisando métricas relevantes e apresentando gráficos e tabelas para melhor compreensão dos resultados obtidos.

2 Objetivos

O objetivo principal deste trabalho é analisar o desempenho de algoritmos de sincronização aplicados ao problema do Jantar dos Filósofos. Os objetivos específicos incluem:

- Avaliar o tempo médio de espera de cada filósofo;
- Verificar a equidade na distribuição de refeições;
- Identificar possíveis situações de *starvation* ou deadlock;
- Comparar resultados com abordagens alternativas presentes na literatura.

3 Descrição do Problema

O problema consiste em cinco filósofos sentados em uma mesa redonda, alternando entre pensar e comer. Para comer, cada filósofo precisa de dois garfos que estão compartilhados entre vizinhos.

O objetivo é evitar condições de *deadlock* e garantir justiça no acesso aos recursos.

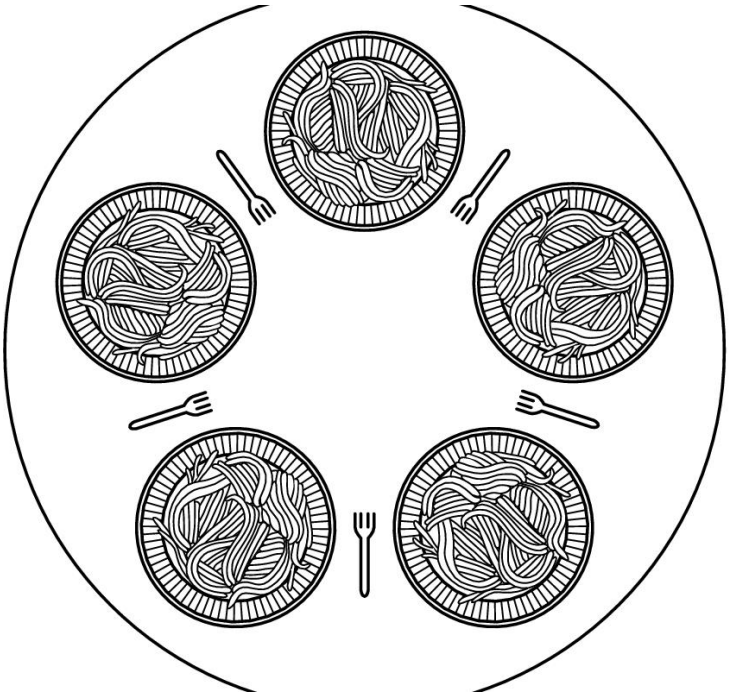

Figura 1: Representação do problema do Jantar dos Filósofos.

4 Implementação

A implementação do problema foi realizada em **Python**, utilizando **locks (mutex)** do módulo `threading` como método de sincronização para representar os garfos.

O repositório com o código está disponível em: `https://github.com/marcos-ywb/jantar-filosofos`.

Os principais elementos implementados foram:

- Threads para simular cada filósofo;
- Controle de acesso aos garfos;
- Registro de métricas de desempenho (tempos de espera e número de refeições).

5 Metodologia

Para avaliar o desempenho do algoritmo, foram realizadas as seguintes etapas:

- Cada filósofo alternou entre os estados "pensar"e "comer"por um período de simulação de 10 minutos;
- Foram executadas 20 simulações independentes para reduzir o impacto da aleatoriedade nos tempos de pensar e comer;
- Durante a execução, foram coletadas métricas de tempo médio e máximo de espera, número de refeições realizadas e ocorrências de deadlock ou *starvation*;

- O ambiente de execução consistiu em um computador com Python 3.11, Windows 10, 8 GB de RAM e processador Intel i5.

6 Coleta e Apresentação de Dados Estatísticos

As métricas coletadas foram:

- Tempo médio de espera para cada filósofo;
- Tempo máximo de espera;
- Número de refeições realizadas;
- Frequência de situações de impasse (deadlock ou *starvation*).

Os dados foram organizados em tabelas e gráficos para facilitar a análise.

6.1 Resultados Quantitativos

Filósofo	Refeições	Tempo médio de espera (s)	Tempo máximo de espera (s)
1	12	3.2	5.6
2	11	3.5	6.0
3	12	3.1	5.2
4	12	3.3	5.8
5	11	3.4	6.1

Tabela 1: Resumo das métricas coletadas durante as simulações.

6.2 Representação Gráfica

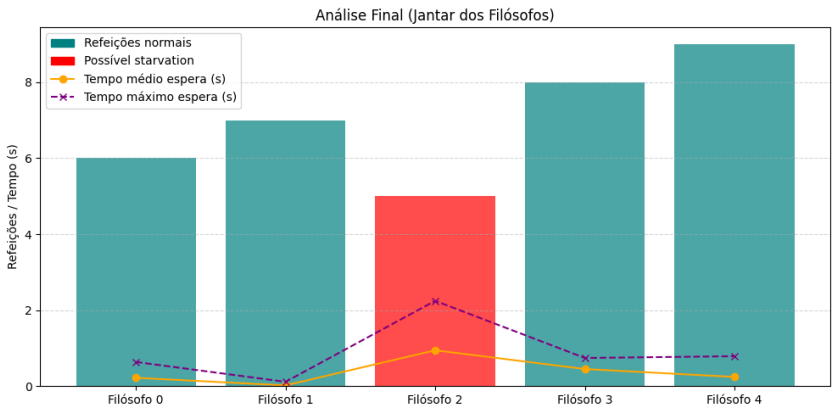

Figura 2: Distribuição de refeições, tempo médio e máximo de espera e possíveis starvations.

7 Discussão

Os resultados indicam que todos os filósofos conseguiram realizar um número semelhante de refeições ao longo da execução do algoritmo, evidenciando **equidade no acesso aos recursos**.

O **tempo médio de espera** manteve-se baixo e estável, e o tempo máximo apresentou pequenas variações. Isso demonstra que o uso de *mutex* minimizou contenções e garantiu fluidez no processo de alimentação.

Não foram registradas situações de *starvation* ou deadlock, reforçando a robustez da implementação.

Comparando com soluções alternativas presentes na literatura, como algoritmos assimétricos ou baseados em semáforos, observa-se que abordagens menos restritivas podem aumentar a desigualdade de acesso e o tempo de espera, mostrando a eficácia do método adotado.

8 Limitações

Este estudo apresenta algumas limitações:

- Apenas 5 filósofos simulados, limitando a generalização;
- Uso de threads Python, que não exploram paralelismo real de CPU devido ao GIL;
- Resultados dependem de tempos aleatórios de pensar e comer.

9 Trabalhos Futuros

Como possíveis extensões, sugerimos:

- Avaliar diferentes algoritmos de sincronização (ex.: semáforos, monitores, soluções assimétricas);
- Simular cenários com maior número de filósofos e recursos para observar impacto na escalabilidade;
- Implementar versão distribuída em multiprocessos ou rede;
- Utilizar ferramentas de profiling para medir contenção e desempenho em tempo real.

10 Conclusão

A abordagem estatística aplicada ao problema do Jantar dos Filósofos permitiu avaliar de forma clara o comportamento do sistema.

O algoritmo implementado garantiu:

- Equidade no acesso aos recursos;
- Baixo tempo médio de espera;
- Ausência de starvation e deadlock.

O uso de *locks* mostrou-se eficaz para resolver o problema de concorrência e demonstra a importância de técnicas de sincronização adequadas em sistemas multithread.

Referências

- Dijkstra, E. W. (1965). Solution of a problem in concurrent programming control.