# Algoritmo para a Solução do Problema dos Filósofos

**Sistemas Operacionais - Trabalho Prático 2**

**Gustavo Carvalho Souza**

Objetivo

O trabalho teve por objetivo a implementação de soluções para o problema do jantar dos filósofos. Uma implementação deveria ser utilizando o algoritmo de Djkstra e a outra deveria ser elaborada. Ao final da implementação, uma análise comparativa dos resultados de cada algoritmo deveria ser feita.

Implementação

Para solucionar este problema foi elaborado um algoritmo utilizando monitores que possuía a seguintes características:

* Uma classe filósofo que chamava os procedimentos *pensar(), pegarGarfos(), comer() e retornarGarfos() consecutivamente*;
* No método *pegarGarfos()* uma análise sobre quais garfos estavam disponíveis era realizada, considerando o identificador de um filósofo *n* e o identificador de seu vizinho. Após verificado que o talher estava disponível setava o estatus de faminto, colocava o processo em espera (wait), alterava a disponibilidade de cada talher do identificador para *true*  e o estatus era modificado para comendo;
* No método *retornarGarfos()* um talher do filósofo de identificador n e o talhercorrespondente ao seu vizinho era setado como *false* e acordava todos os segmentos que estavam aguardando no monitor *(notifyAll) ;*
* O tempo que cada filósofo comia e pensava era aleatorizado entre 0,1 e 0,5 segundos.

Resultados e discussões

Após a implementação, as aplicações foram executadas com um tempo de 180 segundos. Na tabela 1 são verificados alguns resultados obtidos da implementação com algoritmo elaborado, tais como número de vezes que cada filósofo comeu, o maior tempo de espera registrado por cada filósofo assim como a maior média em que cada um dos filósofos ficaram sem comer. Veja abaixo a tabela1:

|  |  |  |  |
| --- | --- | --- | --- |
| Filósofo | Vezes em que comeu | Maior Tempo Sem Comer (espera) | Tempo médio Sem Comer |
| Filósofo 1 | 223 vezes | 1511.0 ms | 430.11ms |
| Filósofo 2 | 216 vezes | 1516.0 ms | 406.84 ms |
| Filósofo 3 | 219 vezes | 1705.0 ms | 409.31 ms |
| Filósofo 4 | 216 vezes | 1713.0 ms | 549.92 ms |
| Filósofo 5 | 217 vezes | 1610.0 ms | 530.54 ms |

Tabela : Relação de dados obtidos após a execução do programa que soluciona o problema do jantar dos filósofos, utilizando uma implementação baseada em monitores . Aplicação realizada com um tempo de 180 segundos.

Podemos observar que o algoritmo possibilitou que filósofos diferentes pudessem comer com uma quantidade de vezes próximas aos outros. O tempo sem comer dos filósofos variou de 1,5 a 1,7 segundos, enquanto o tempo médio sem comer variou entre 0,4 e 0,5 segundos.

Na tabela 2 são verificados alguns resultados obtidos da implementação com algoritmo elaborado, tais como número de vezes que cada filósofo comeu, o maior tempo de espera registrado por cada filósofo assim como a maior média em que cada um dos filósofos ficaram sem comer. Veja abaixo a tabela2:

|  |  |  |  |
| --- | --- | --- | --- |
| Filósofo | Vezes em que comeu | Maior Tempo Sem Comer (espera) | Tempo médio Sem Comer |
| Filósofo 1 | 252 vezes | 3024.0 ms | 1173.05 ms |
| Filósofo 2 | 254 vezes | 3212.0 ms | 646.55 ms |
| Filósofo 3 | 267 vezes | 3079.0 ms | 930.0 ms |
| Filósofo 4 | 253 vezes | 3423.0 ms | 479.84 ms |
| Filósofo 5 | 278 vezes | 3044.0 ms | 846.28 ms |

Tabela :Relação de dados obtidos após a execução do programa que soluciona o problema do jantar dos filósofos, utilizando uma implementação com algoritmo de Djkstra. Aplicação realizada com um tempo de 180 segundos.

Com o algoritmo de Djkstra,pode se verificar o quão poderoso ele é, uma vez que possibilitou que filósofos diferentes tivessem comidos comuma quantidade de vezes bem próximos. O tempo sem comer dos filósofos variou de 3,0 a 3,4 segundos, enquanto o tempo médio sem comer variou entre 0,4 e 1,1 segundos. Comparando as duas implementações podemos verificar que ambas coseguiram controlar de forma justa a concorrência pela comida – talvez pela semelhança das abordagens.Um fato curioso é que embora a implementação com o algoritmo de Djkstra tenha possibilitado que filósofos comessem com uma maior quantidade de vezes, este algoritmo obteve o maior tempo de espera por filósofo. A conclusão para isso é que este algoritmo embora permita a concorrência de forma justa, ele permite uma maior repetição do estado de comendo para um filósofo em específico, deixando de privilegiar outros filósofos em determinado tempos seguidos (o que aumenta o tempo de espera) .

Conclusões

Ambos os algoritmos conseguiram exercer uma política de controle para a refeição (jantar) dos filósofos, uma vez que possibilitou de forma justa que filósofos diferentes tivessem valores aproximados na quantidade de vezes em que tinham comido. O algoritmo implementado, de forma indireta se assemelha ao algoritmo de Djsktra, quanto as sequências de de ações: pensar, pegar garfo, comer e liberar garfo. O algoritmo elaborado possui como resultado valores com a mais baixa quantidade de vezes em que os filósofos comeram, assim como o tempo de espera por filósofo. O algoritmo de Djkstra permite que filósofos esteja no estado de comendo um maior número de vezes porém ocasiona um tempo de espera maior por filósofo.