Trabalho I – Simulador de Funcionamento do Restaurante Universitário

Genicleito Carvalho Beltrão Gonçalves

Curso de Bacharelado em Ciência da Computação – Universidade Federal da Bahia (UFBA) – Campus de Ondina
Salvador– BA– Brasil

genicleito@gmail.com

Abstract. *This report describes the results of the development work I of discipline MATA58 - Operating Systems, as well as the difficulties encountered in the development of the same and parallel programming experience using threads.*

Resumo. *Este relatório descreve os resultados obtidos no desenvolvimento do Trabalho I da disciplina MATA58 – Sistemas Operacionais, bem como as dificuldades encontradas no desenvolvimento do mesmo e a experiência de programação paralela utilizando threads.*

Introdução

O objetivo desse trabalho é simular o funcionamento de um Restaurante Universitário utilizando Multi-Threads e operações sobre semáforos para sincronização e controle de acesso a regiões criticas. As dificuldade encontradas devem-se à necessidade de sincronizar a execução das threads e definir uma ordem de execução (um estudante não pode entrar na rampa de alimentação sem ter comprado a ficha, por exemplo).

O simulador não apresenta interação com o usuário, uma vez iniciado, o programa cria N threads (onde N é o número de estudantes que chegarão ao RU, gerado aleatoriamente em cada execução). N é menor que duas vezes a quantidade de refeições.

Durante a execução o simulador inicialmente carrega a quantidade de estudantes com o uso do rand() limitando a quantidade máxima a duas vezes a quantidade de refeições que irão ser servidas neste expediente. Esses estudantes são representados no programa por um vetor de threads que chama, para cada thread, a função *estudante*.

Todas as threads iniciadas antes das threads de estudantes serem criadas dormem ao perceber que as respectivas filas (para comprar a ficha e para entrar na rampa de alimentação) estão vazias. Isso é possível graças ao uso de operações sobre semáforos.

A thread responsável pela recepção e venda das fichas aos estudantes executa a função *recepcionista* e assegura que ainda haja refeição para ser servida ao estudante que está comprando a ficha, além de garantir que esteja no horário de funcionamento, informando quando isso não for possível e o RU tiver que ser fechado.

A thread responsável pelo gerenciamento dos estudantes que acessam a rampa de alimentação executa a função *catraca*. Além de gerenciar o acesso, também informa que o simulador pode ser encerrado no momento que o RU já foi fechado e todos os estudantes da fila para rampa já se alimentaram.

Tempo de Simulação

O simulador inicialmente é carregado com um horário de funcionamento pré-definido pelo programador, esse tempo vai sendo decrescido até que chegue a zero e portanto encerre o expediente do simulador, aguardando os estudantes que já estavam dentro do RU terminarem sua alimentação.

É importante ressaltar que, mesmo que as refeições terminem o simulador continua em execução até que seu horário de funcionamento chegue a zero e encerre a simulação (na vida real os estudantes não ficam trancados dentro do RU porque o horário de funcionamento acabou, o serviço é encerrado para novos estudantes apenas).

Dificuldades Encontradas

Mesmo com um pouco de experiência com programação com OpenMP, manipular os jobs mais diretamente, implementar as barreiras, as regiões criticas, a comunicação e sincronização tornam as coisas meio "estranhas" de entender no começo, cicatrizes da programação majoritariamente serial nas disciplinas da grade de Ciência da Computação.

Tive dificuldades também pelo falo de minha dupla desistir da disciplina e eu ficar por conta própria, por conta disso, vários planos e ideias que pretendia por em prática nesse simulador não pude fazê-los pela demanda do tempo.

Roteiro do Trabalho e Conclusões

Coloquei alguns conteúdos relevantes para o desenrolar do simulador em um roteiro que pudesse ser seguido e que facilitasse o acompanhamento do código.

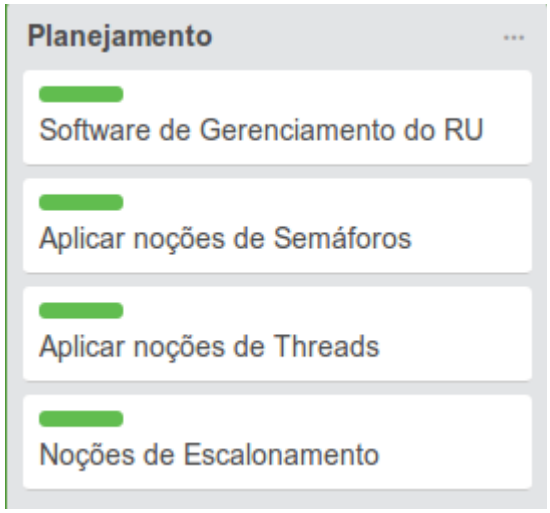

Figura 1. Planejamento

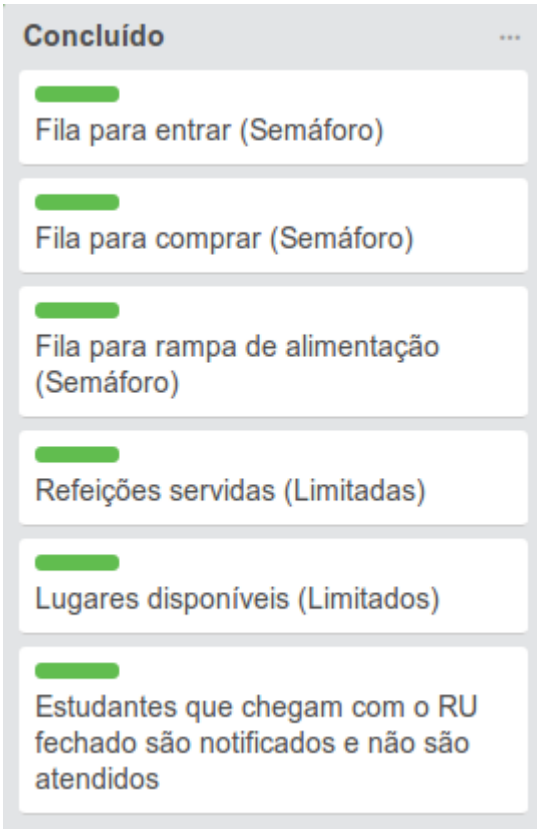

Figura 2. Conclusões na Implementação

Mesmo que possa parecer algo simples, pude praticar neste simulador os conceitos que aprendi até agora na disciplina, especificamente os conceitos de programação paralela com o gerenciamento de vários jobs que realizam múltiplas tarefas mas que precisam se comunicar para evitar sobrescrita de informações importantes e condições de corrida. Além do fato de poder "visualizar" as threads trabalhando em conjunto na versão final do simulador.

Resultados

Foram realizados alguns poucos testes para que pudessem ser representados em forma de tabela e gráfico, para que o comportamento do simulador pudesse ser visualizado.

Tabela 1. Testes para simulação

Horário de funcionamento	Refeições	Estudantes que chegaram*	Estudantes que se alimentaram**	Tempo de Funcionamento (em segundos)**
50	20	14	14	67
50	20	17	17	81
50	20	8	8	46
50	20	38	11	52
150	100	28	28	136
150	100	150	37	182
150	100	167	36	176
150	100	35	35	170
150	10	0	0	150

** Valor gerado aleatoriamente*
*** Resultados da execução*

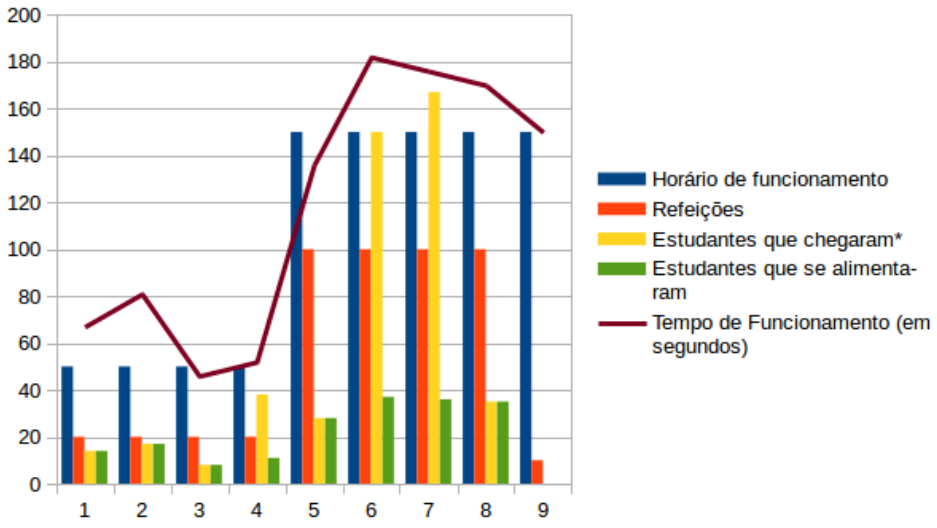

Figura 3. Gráfico de simulação