Documentação do TP1

Nome: Julia Rafael Correa Abud
Matrícula: 2016058360

Introdução

O problema apresentado era o tempo médio de atendimento de clientes na cantina do ICEx, em 4 horas. Inicialmente, existe uma fila para pegar a ficha de atendimento, uma fila para pegar as bandejas e uma pilha de bandejas. Chegam duas pessoas a cada intervalo de tempo, na fila de fichas, mas apenas uma é atendida. A pessoa atendida chega na fila de bandejas e se houver bandejas disponíveis, a pessoa passa para pegar a comida, que gasta 4 intervalos de tempo.

A solução proposta para melhorar o tempo médio de atendimento de cada cliente é criar mais filas de bandejas.

Código

São três arquivos de código: pilha_fila.c e pilha_fila.h, que contém os TADs de pilha e fila utilizados nesse TP, e o TP1.c que contém o main. São quatro blocos principais dentro do main. O primeiro tem a simulação do caso inicial, o segundo tem a simulação do caso com duas filas de bandejas, o terceiro tem a simulação de uma pilha com 45 bandejas iniciais e o quarto tem a simulação do tempo de reposição das bandejas variando.

Linhas: *9-22:* comportam as declarações das variáveis do TP e das filas e pilhas. A variável *i* é para contagem em estruturas tipo *for, clock* é para a contagem das 4h do problema, *repor* é para o controle do número de bandejas na pilha, *usuarios* para a contagem de quantas pessoas foram atendidas e *restantes* para contar quantas pessoas restaram na fila após as 4h.

26-30: zera o *clock* e empilha as 30 bandejas iniciais.

31-48: inicia a fila com duas pessoas, desenfileira uma pessoa, que recebe a ficha e enfileira essa pessoa na fila de bandejas.

49-67: testa se há bandejas na pilha, se caso tenha, a pessoa pode pegar e seguir para pegar a comida. A pessoa é enfileirada na fila de Atendidos, para a contagem de quantos usuários pegaram a comida.

68-73: verifica de já se passaram 12 intervalos para ocorrer a reposição de bandejas. Caso positivo, verifica se as bandejas podem ser empilhadas, para não ultrapassar o limite de 30. Empilha as novas bandejas e zera o *repor*.

83-94: faz a contagem de usuários em cada fila.

98-104: Imprime os resultados no terminal, e calcula o tempo médio que cada pessoa gastou no atendimento, e acaba a simulação do caso inicial.

Os outros casos possuem uma estrutura bem parecida, só muda o valor das variáveis e a quantidades de filas presentes, e no começo de cada bloco a pilha de bandejas é zerada.

Complexidade

As operações de atribuição tem custo O(1).

CASO INICIAL:

O laço *for(i=0;i<n;i++)* sendo *n* o número de bandejas tem complexidade O(n), pois realizará uma operação de atribuição n vezes.

O laço *while(clock<n)* sendo *n* quantos minutos irá durar a simulação, tem complexidade O(n). Mas ele está aninhado com um *for(i=0;i<n;i++),* em que *n* é o número de bandejas, que também possui complexidade O(n). Portanto, como as atribuições feitas dentro deste laço são O(1), a complexidade total fica como $O(n^2)$.

Temos em seguida três laços *while(!TestaVazia())* que também terá complexidade O(n), sendo *n* a quantidade de itens enfileirados/empilhados.Como O(f(n)) + O(g(n)) = O(max(f(n),g(n))), o algoritmo terá complexidade $O(n^2)$.

OUTROS CASOS:

A complexidade não mudará, pois o que alterado foi apenas os valores de *n* e o número de atribuições.

Resultados e solução

O caso inicial teve um tempo médio de 6 minutos por pessoa. O melhor resultado obtido foi o de duas pilhas de bandejas, que conseguiu atender o dobro de pessoas no mesmo tempo, o que gerou um tempo médio de atendimento de apenas 3 minutos.

Mudanças do tipo aumentar o número de bandejas ou diminuir o tempo de reposição não altera o resultado final, pois gerou o mesmo tempo médio do caso inicial.

A solução então seria aumentar a fila de bandejas, pois assim será possível atender mais pessoas no mesmo intervalo de tempo.