Documentação Trabalho Prático 01 AEDS2

Fila, pilhas e complexidade.

Introdução:

O objetivo deste trabalho consiste em simular o funcionamento da Cantina do ICEx a partir de algumas simplificações das suas situações e parametrização das suas condições. Sendo seu funcionamento durante o almoço separado em três partes principais, que no ponto de vista de um consumidor seriam: Comprar uma ficha; Pegar uma bandeja, talheres e prato; Ser servido do almoço.

Para esse funcionamento, Existem três filas: Uma para comprar a ficha; Uma para pegar uma bandeja, prato e talheres; E mais uma em que é servido o almoço. Estas duas últimas podem também serem vistas como uma só.

Existem também as transições entre cada fila, cada uma levando um tempo para ocorrer, senso este tempo simplificado no problema. E também existe uma pilha de bandejas, pratos e talheres.

Dessa forma, utilizando estas simplificações, é pedido que se calculasse o tempo médio que uma pessoa é servida durante o almoço. Para isso, foram utilizadas TADs do tipo fila e pilha para representarem as filas de pessoas e pilhas de bandejas, enquanto que cada transição de filas é tratada.

Desenvolvimento:

Para calcular o tempo que cada pessoa gasta para ser servida, bastaria guardar o tempo de chegada de cada pessoa, e comparar ao tempo de saída desta mesma pessoa.

Durante o desenvolvimento do método para o cálculo, percebi que não era de extrema necessidade que fossem utilizadas filas e pilhas.

As pilhas de bandejas poderiam ser convertidas apenas para inteiros, já que entre uma bandeja e outra não há diferenciação, e o que realmente importa é a quantidade disponível na pilha.

Já as pessoas, são diferenciáveis, mas apenas pelo seu tempo de chegada, mas isso pode ser facilmente transformada em uma equação da forma:

Seja N(t) quantas pessoas entraram na fila em um tempo “t”, e “A” quantas pessoas chegam por unidade de tempo.

Pegando a inversa temos que...

Como estamos tratando de tempo e quantidade de pessoa como inteiros temos que:

Sendo t(N) o tempo de entrada da N-ésima pessoa a ser servida.

Dessa forma, se assumirmos que o processo no geral é justo, não é mais necessário guardar o valor do tempo de cada pessoa, apenas quantas pessoas já saíram da fila. Isso então também torna as pessoas indistinguíveis, o que transforma cada fila também em um inteiro, sendo apenas importante a quantidade de pessoas nela.

Diminuindo então bastante a complexidade de espaços do programa.

Mas como o tema do TP é “Filas, pilhas e complexidade”, considerei importante a implementação utilizando esses TADs.

Devido às alterações no programa, decidi também fazer um único programa mais complexo, que recebe as alterações como parâmetros, facilitando os testes.

Implementação:

Para a implementação do programa, foram necessários implementar os Tipos Abstratos de Dados Fila e Pilha, ambas implementadas nos arquivos “TADs.c” e “TADs.h” Utilizando como base os *slides* disponibilizados pelo professor Adriano. As principais alterações foram a criação de métodos para a desalocação de memória em cada TAD, alteração no tipo de métodos e parâmetros recebidos por eles. As Pilhas não continham nada além dos Apontadores dentro de suas células, e as Filas continham um inteiro junto ao apontador.

No “main.c”:

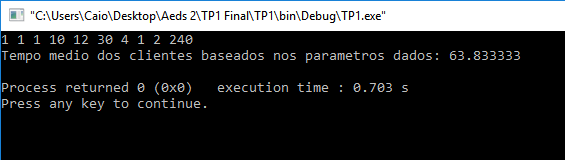
O cálculo do tempo médio gasto para servir cada cliente da cantina, é então devolvido pela função “tempoMedioCantina()” que recebe dez parâmetros de acordo com as possíveis alterações cabíveis à simulação, são eles:

* **int** numFilasDeComprasDeFicha: Seria o número de filas para a compra de fichas, cada uma associada a um caixa.
* **int** numFilasDeBandejas: Seria o número de filas para pegar uma bandeja, prato e talheres. O primeiro de cada fila pega uma bandeja a cada instante de tempo caso tenham bandejas disponíveis.
* **int** numPilhasDeBandejas: Seria o número de pilhas de bandejas, sendo cada uma reposta independentemente.
* **int** reposicaoDeBandejas: É a quantidade de bandejas que são repostas por vez.
* **int** tempoDeReposicao: É o tempo necessário para repor as bandejas.
* **int** limiteDeBandejasPorPilha: É a quantidade máxima de bandejas permitidas em uma pilha.
* **int** numDeAlimentos: É o número de alimentos diferentes a serem servidos. (Esta alteração não foi requisitada no TP1).
* **int** baiasDeAlimentos: É a quantidade de filas onde é servido o almoço. (Esta alteração não foi requisitada no TP1, mas era um fator extremamente limitante no desempenho da cantina).
* **int** clientesEntrandoPorMinuto: É a taxa com que os clientes entram nas filas de compra de fichas. (Esta alteração não foi requisitada no TP1).
* **int** tempoDeSimulacao: É o tempo total do serviço dado em minutos. (Esta alteração não foi requisitada no TP1).

O programa main lê os dez inteiros utilizando a função scanf() e imprime o resultado do cálculo de acordo com esses parâmetros. Sendo que caso seja escrito:

*“1 1 1 10 12 30 4 1 2 240”*

O programa calculará para o caso inicial descrito pelo enunciado do TP1, sendo a resposta “63.833333”, como pode ser visto na figura abaixo:



O programa apenas considera aqueles que já foram servidos como base para a média, aqueles que estão ainda esperando nas filas quando o tempo de simulação é atingido são desconsiderados.

Para manter uma justiça no funcionamento, de forma que o primeiro a entrar é o primeiro a sair, as transições de filas são marcadas por uma alternância quanto as filas em que as pessoas entram.

Para respeitar o tempo gasto nas transições entre filas (comprar ficha, e pegar bandeja) as filas são tratadas na ordem inversa, ou seja, primeiro a fila de servir, segundo a de pegar bandeja e terceiro, a de comprar ficha.

E ao final da parte iterativa, é verificado se as pilhas de bandejas devem ser repostas.

Análise:

A função “tempoMedioCantina()” tem a parte com maior complexidade, a parte iterativa, que pode ser dividida em quatro partes:

* Tratamento das filas onde é servido o almoço, com complexidade:  
  O(baiasDeAlimentos \* numDeAlimentos).
* Tratamento das filas para pegar as bandejas, com complexidade:  
  O(numFilasDeBandejas).
* Tratamento para as filas para comprar a ficha, com complexidade:  
  O(clientesEntrandoPorMinuto + numFilasDeCompraDeFicha).
* Tratamento para a reposição de bandejas, com complexidade:  
  O(numPilhasDeBandejas \* reposicaoDeBandejas / tempoDeReposicao).

Tendo uma complexidade total:  
O( tempoDeSimulação \* ((baiasDeAlimentos \* numDeAlimentos)+ numFilasDeBandejas + clientesEntrandoPorMinuto + numFilasDeCompraDeFicha +(numPilhasDeBandejas \* reposicaoDeBandejas / tempoDeReposicao)).

Resultados:

* Caso Inicial:   
  Entrada: “1 1 1 10 12 30 4 1 2 240”.Saída: “63.833333 minutos”.
* Caso Inicial com as alterações: 2 filas de fichas, uma de bandejas e 2 pilhas de bandejas;  
  Entrada: “2 1 1 10 12 30 4 1 2 240”.  
  Saída: “64.5 minutos”.
* Caso Inicial com as alterações: 1 de ficha, duas de bandejas, 1 pilha de bandeja com 40 bandejas.  
  Entrada: “1 2 1 10 12 40 4 1 2 240”.  
  Saída: “63.243363 minutos”
* Caso Inicial com as alterações: 2 de ficha, 2 de bandejas, 2 pilha de bandeja, 2 baias de alimentos.  
  Entrada: “2 2 2 10 12 30 4 2 2 240”.  
  Saída: “9.833333 minutos”
* Caso Inicial com as alterações: 1 pessoa por minuto  
  Entrada: “9.833333 minutos”.  
  Saída: “63.243363 minutos”
* Caso Inicial com as alterações: 1 pessoa por minuto e 12 bandejas repostas por vez.  
  Entrada: “1 1 1 12 12 30 4 1 1 240”.  
  Saída: “6 minutos”
* Caso Inicial com as alterações: 1 pessoa por minuto e 6 bandejas repostas por vez.  
  Entrada: “1 1 1 6 12 30 4 1 1 240”.  
  Saída: “36 minutos”
* Caso Inicial com as alterações: 2 horas de tempo de simulação.  
  Entrada: “1 1 1 10 12 30 4 1 2 120”.  
  Saída: “34.5 minutos”
* Caso Inicial com as alterações: 1 hora de tempo de simulação.  
  Entrada: “1 1 1 10 12 30 4 1 2 60”.  
  Saída: “19.5 minutos”
* Caso Inicial com as alterações: meia hora de tempo de simulação.  
  Entrada: “1 1 1 10 12 30 4 1 2 30”.  
  Saída: “12 minutos”

Essas alterações nos permitem avaliar o impacto individual de cada parametrização, e o impacto que cada uma tem.

Dessa forma podemos observar que alguns fatores tem mais impacto que outros, e com a mudança no tempo, podemos ver a evolução do tempo médio ao decorrer da simulação.

Conclusão:

No geral, o caso com média baixa é o que as filas, pilhas e baias de comida equiparam ou são maiores que a quantidade de pessoas que entram por minuto da fila, e quanto maior a razão entre a quantidade de bandejas repostas e o tempo gasto por reposição, menos pessoas se acumularão devido a não existência de bandejas nas pilhas, sendo os atrasos agravados pelo tempo de simulação.

O melhor caso é aquele em que:

* O número de filas e baias é igual à taxa de entrada de clientes por minuto. (numFilasDeCompraDeFicha = numFilasDeBandejas = baiasDeAlimentos = clientesEntrandoPorMinuto);
* A razão entre a quantidade de bandejas repostas e o tempo de reposição é igual à taxa de entrada de clientes por minuto e a quantidade de pilhas. ((reposicaoDeBandejas/ tempoDeReposicao) = (clientesEntrandoPorMinuto/ numPilhasDeBandejas));
* Dessa forma a média sempre será “numDeAlimentos + 2” independente do tempo. Sem gastar recursos extras.

Referências e Agradecimentos:

Slides apresentados pelo professor Adriano.

Meu amigo e ex-Professor Lucas Alves por me ajudar a debugar o programa.

E as discussões com meu colega de classe Catrinque.