Sistemas Operativos
Projeto C - Doca Carga/Descarga

Helder Correia - 20102556

Curso de Informática, Redes e Multimédia

Universidade dos Açores

2 de Junho de 2012

Conteúdo

Introdução

Este projeto teve como objetivo desenvolver uma doca para carga e descarga de barcos, na linguagem C, e utilizando o conceito de threads/processos, com semáforos e trincos para sincronização.

A doca tem um número (N) limitado de lugares, incluindo 3 ancoradouros, cada um com um lugar para carga e descarga de mercadoria e mais dois em fila de espera. Devem existir mais barcos do que lugares disponíveis na doca ($B > N$), para que seja simulada a situação de ter a doca cheia.

Cada barco tenta entrar na "baía de estacionamento", posteriormente entrando na área do primeiro ancoradouro disponível. Cada ancoradouro recebe o primeiro barco da sua fila para iniciar o processo de descarregar o barco de toda a sua carga, e carregá-lo com uma quantidade aleatória que está armazenada no ancoradouro (inicializada no início), tendo em atenção a capacidade máxima de carga do barco.

A doca fecha quando não há mais carga nos ancoradouros para carregar para os barcos, nem barcos com carga para descarregar. O programa termina indicando a quantidade de carga descarregada em cada ancoradouro.

Desenvolvimento

As constantes com os parâmetros do programa foram definidas logo no início para mais fácil edição.

Duas threads foram criadas para representar cada barco e cada ancoradouro. Foram também criadas duas estruturas que representam um barco e um ancoradouro, o que facilita a comunicação entre as duas threads.

As várias funções do programa foram prototipadas ou declaradas no início para dar maior liberdade na posição das suas definições, tendo como objetivo haver uma maior coerência vertical na leitura do código, onde as funções são definidas depois de serem usadas. Isto torna a leitura do código mais natural, como uma prosa.

Cada ancoradouro tem o seu próprio grupo de semáforos, caso contrário, não seria possível fixar um barco num ancoradouro específico. São usados dois trincos, um para gerir o movimento das cargas e outro para associar e desassociar um barco a um ancoradouro (grua).

O fluxo principal do programa inicia os trincos e semáforos com os seus contadores em valores estratégicos, inicializa cada ancoradouro a um estado inicial criando uma thread para cada um, e inicializa também cada barco com uma quantidade de carga aleatória.

O funcionamento da doca agora depende das threads dos barcos e ancoradouros (`thr_boat` e `thr_anchorage` respetivamente) trabalharem em conjunto para movimentar toda a carga.

Este fluxo de execução fica agora à espera que os barcos retornem, e a seguir todos os ancoradouros. É exibido no ecrã os totais de toda a carga movimentada e o programa encerra.

Funcionamento da thread `thr_anchorage`

A thread de cada ancoradouro inicia dentro de um ciclo que termina quando o ancoradouro fechar, ficando logo de seguida à espera que o primeiro barco se declare pronto a iniciar a carga e descarga.

Quando o próximo barco ficar disponível, é feita a sua descarga e carregamento com base numa quantia aleatória do total por carregar, sem exceder a capacidade máxima.

A seguir, o ancoradouro assinala o fim da carga/descarga, verifica se ainda está em condições para o ancoradouro se manter aberto, e espera pelo próximo barco.

O ancoradouro fecha se não houver mais carga para carregar para os barcos, nem existir carga para descarregar de mais nenhum barco existente.

Funcionamento da thread `thr_boat`

A thread de cada barco inicia dentro de um ciclo que apenas termina quando a doca estiver fechada. A doca fecha quando não houver mais carga em nenhum ancoradouro para carregar para os barcos, nem carga para descarregar de mais nenhum barco.

A seguir, é usado um `sem_trywait()` para tentar entrar na baía de estacionamento (limitada a 15 lugares por defeito). Caso esta esteja cheia, o barco esperará um período aleatório de tempo (por defeito até 6 segundos), antes de voltar a tentar entrar. Este processo também acontece após um barco terminar a sua carga/descarga.

Depois de ter entrado com sucesso na baía, o barco procura um ancoradouro disponível (com outro `sem_trywait()`). Os critérios para entrar na zona de um ancoradouro é haver carga para descarregar ou carga no ancoradouro para carregar, e haver lugar num dos dois lugares de espera. Se estas condições se verificarem, o barco entra na área do ancoradouro e fica à espera pela sua vez de ir para o lugar da grua, que é onde é movimentada a carga.

Inicialmente com o contador do semáforo a 1, a grua aceita um novo barco quando o último sair. Quando o barco segue para a grua, ele fica associado ao ancoradouro e assinala que está pronto para a carga/descarga, ficando à espera que esta termine (execução pela `thr_anchorage`).

Quando terminar, o barco é desassociado do ancoradouro e assinala que o lugar está agora vago para o próximo barco na fila de espera do ancoradouro. Seguindo isto, o barco assinala que um lugar na baía está agora disponível e prossegue a navegar por um período de tempo variável (como descrito no segundo parágrafo desta seção).

Quando o barco retorna, faz nova verificação se a doca se encontra fechada, e em caso positivo encerra o seu fluxo de execução, retornando ao fluxo principal (simulando um retorno ao mar por tempo indeterminado).

Funções de ajuda

Para melhorar a leitura do código e facilitar a reutilização, foram criadas algumas funções de ajuda, descritas sucintamente a seguir.

int anchorage_is_closed(**int** id)
: Verifica se o ancoradouro está fechado devido a não ter carga no ancoradouro para carregar nem mais carga em nenhum barco para descarregar.

int cargo_in_boats()
: Verifica se existe carga em pelo menos um barco.

int dock_is_closed ()

Verifica se a doca está fechada devido a não haver mais carga em nenhum ancoradouro para carregar, nem carga em nenhum barco para descarregar.

int cargo_in_anchorages()

Verifica se existe carga por carregar em pelo menos um ancoradouro.

int enter_anchorage(**int** anchorage_id, boat_t *boat)

Retorna 1 (verdadeiro) caso haja carga no barco para descarregar ou carga no ancoradouro para carregar, e tenha conseguido entrar num dos lugares de espera da área do ancoradouro; 0 (falso) caso contrário.

void wait_for_crane(**int** anchorage_id, boat_t *boat)

Espera até que a grua esteja disponível para iniciar carga/descarga, e associa o barco ao ancoradouro quando isso acontecer.

void leave_crane(**int** anchorage_id, **int** boat_id)

Desassocia o barco do ancoradouro e disponibiliza grua para o próximo barco.

boat_t *boat_at_crane(**int** anchorage_id)

Retorna apontador para o barco associado no momento ao ancoradouro, ou -1 caso não haja nenhum.

void sail(**int** boat_id)

Faz esperar (navega) um número aleatório de segundos.

Conclusão

O projeto tem complexidade suficiente para não ser um trabalho trivial, mas com alguma prática e experiência, vai-se tornando mais fácil. A complexidade maior está em gerir os três ancoradouros de forma independente dentro da baía.

Equilíbrio entre os ancoradouros

Executando o programa, nota-se que o primeiro ancoradouro movimenta cerca de 95% da carga. Isto deve-se ao fato de que quando um barco entra na baía e procura pelo primeiro ancoradouro disponível, a primeira tentativa é sempre o primeiro ancoradouro. Só quando o primeiro está todo ocupado, o barco experimenta o segundo, e assim sucessivamente. Na maior parte do tempo, como a execução é rápida, enquanto um 4º barco procura lugar disponível, o 1º já terminou e libertou o espaço, dando lugar ao 4º na fila de espera.

Esta situação de maior movimentação no primeiro ancoradouro não aumenta o tempo de espera nos restantes ancoradouros e por isso não foi considerado grave nem necessariamente indesejável. No entanto, é algo que com mais tempo poderia ser investigada uma forma de equilibrar o uso dos ancoradouros. A solução, à partida pareceu adicionar complexidade ao programa, e a tentativa foi abandonada.

Encerramento de um ancoradouro

Uma segunda situação que também por falta de tempo não foi melhorada foi a forma como é encerrado um ancoradouro.

Quando um ancoradouro não está a movimentar carga, ele está em estado de espera. Portanto a próxima oportunidade de verificar se ele precisa ser fechado ou não, é depois de iniciar uma carga/descarga. Mas o que acontece quando durante essa verificação ainda existem barcos com carga, mas em menor número que o número de ancoradouros?

Isso significa que haverá dois ancoradouros (exceto o que processar o último barco) que ficarão bloqueados no estado de espera. Seria talvez necessário contar o número de barcos ainda com carga, mas também em que ancoradouro se encontram.

Para poupar tempo, a solução provisória foi fazer uma chamada ao `sem_post()` de cada ancoradouro no `main()` após retornarem todos os barcos, e antes de fazer um `sem_join()` para desbloquear qualquer um que esteja preso.