Guilherme Augusto Pangratz - 15159554

TDA Área

Resumo

O TDA área foi implementado de acordo com a abordagem tradicional para implementação, escondendo da aplicação as propriedades da estrutura e até mesmo partes da estrutura. Todas as operações foram implementadas .

Estruturas

As estruturas do TDA encontram-se no arquivo area_priv.c. São duas ao todo, uma para descrever as celulas, e uma para descrever a lista em sí. Esiste também uma função privada para comparar células e verificar qual delas possui a maior chave, apesar de implementada, acabei não utilizando-a.

Na apresentação do problema havia uma estrutura item que contia então o valor da chave, porem acabei por usar na celula apenas um inteiro com ochave, o que acabou economizando uma porção de acessos ao longo do código, sem causar nenhum problema, já que o problema pedia que as chaves fossem inteiros.

Operações

As operações do TDA fundamentais para o TDA foram todas declarada no arquivo area.h que é a interface do TDA, e foram descritas no arquivo area.c, que contem a implementação de todas as funções declaradas em area.h e em area_priv.h.

A abordagem que utilizei para criação foi a de retornar um ponteiro de Area para o usuario na hora de usar a função que cria e inicializa a área. Dessa maneira com algumas alterações bem simples é possível alterar a estrutura area e a operação cria_area_vazia para que o usuario possa definir a quantidade de celulas que ele deseja em sua area. Para isso basta substituir o vetor itens por um ponteiro para celulas e depois alocar memoria para este ponteiro depois de alocar memoria para a área. Não utilizei esta abordagem devido ao fato de o problema ter sido apresentado com a macro TAMCELS definindo a quantidade de celulas em area.

Funcionamento

A parte mais complexa da implementação é a que trata da inserção de dados, já que eles devem ser colocados em ordem. Apesar disso não se trata de nada muito difícil. Como a conexão entre as células acorre através dos valores armazenados em cursores que indicam qual a posição do elemento anterior e posterior daquela chave dentro da lista, a maneira mais simples para ordenar os valores consultar todos os valores em alguma ordem comparando o tamanho das chaves até encontrar a posição em que o elemento deve ser

inserido. Este elemento pode ser então ligado aos seus vizinhos através de seus cursores, e então podemos ligar os respectivos vizinhos a ele. Esse processo deve ser O(n), sendo n a quantidade de celulas na lista.

Como só ocorre a remoção do menor ou do menor valor da área de cada vez, a lista ordenada possibilita que uma operação bem simples faça isso com O(1). Basta desligar a extremidade da estrutura correspondente ao maior ou menor valor.

o maior desafio desta abordagem de cursores em um vetor de tamanho conhecido é o gerenciamento dos valores dos cursores de cada célula e da própria estrutura área. Deve-se prestar muita atenção ao contador no processo de incersão e remoção, e também aos valores “padrões” que cada cursor ou contador recebe quando a área é inicializada, esvaziada, ou completamente preenchida, para que não se perca a ligação e a ordem entre os elementos, e também para o sistema de celulas vazias interligadas funcionar.

Como optei por usar ponteiros de área na hora de criar a área, foi necessária uma operação extra para destruir a área, limpando a memória que foi alocada e endereçada pelo ponteiro de área na hora da sua criação.

Considerações finais

O TDA área assemelha-se muito com uma implementação de lista duplamente encadeada e oferece essa abordagem interessante do uso de cursores que indicam posições em um vetor para manter a ordem desejada para os elementos.