Relatório Projeto Matriz Esparsa

Professor: Francisco da Fonseca Rodrigues

Alunos:

Felipe Scherer Vicentin – 18178

Maria Eduarda Elias Rocha – 18190

Introdução:

O projeto para a criação de matriz esparsa tem como objetivo ensinar o uso de uma estrutura de dados (Lista Ligada) para a criação de outra estrutura de dados (Matriz Esparsa), que é comumente usada por programas mais complexos.

Desenvolvimento:

13/03/2019 – Criação do repositório no GitHub e início da classe Celula e Lista Ligada.

18/03/2019 – Parte visual feita e Celula é finalizada.

19/03/2019 – Alteração, adição e procura de célula, feitos.

24/03/2019 – Todos os métodos de Lista Ligada prontos, exceto somar K a coluna.

03/04/2019 – Somar K a coluna pronto.

04/04/2019 – Alguns ajustes na parte visual.

09/04/2019 – Relatório Finalizado.

Problemas e Soluções encontrados

No início do projeto, decidimos priorizar a velocidade de operações acima de qualquer outra coisa. Para isso, procuramos deixar todos os métodos com o menor número de loops possível, evitando repetições desnecessárias e manipulando o máximo possível os ponteiros existentes em mãos. Contudo, por conta da escolha, a classe ListaLigada ficou, no geral, bem maior que o previsto.

A elaboração e codificação da maior parte dos métodos ocorreu sem muitos problemas, mas os métodos ExisteCelula e SomarNaColuna foram excepcionalmente mais complicados que os demais.

Na primeira tentativa de elaboração do ExisteCelula, fizemos a variável “acima”, que representa a célula vertical com valor logo antes da célula procurada, ser atualizada diversas vezes durante o percurso da variável “atual”. No caso, cada vez que o atual avançava uma posição à direita, “acima” era recalculado, o que retardava muito o processo. Depois que percebemos o engano inicial e a obsolescência dessa atualização repetida de valor, codificamos a versão atual do ExisteCelula, que utiliza poucos loops e nenhum “loop dentro de loop”.

No caso do método SomarNaColuna, iniciamos ele com uma abordagem desafiadora: pretendíamos fazê-lo utilizando operadores de if else e apenas um único loop, que percorre a coluna em questão. Tentamos unir nossas lógicas e chegamos em um resultado muito satisfatório. Em alguns testes, obtivemos uma eficiência enorme. Por exemplo, uma matriz 50000x1, no computador do Dinalva, levava em torno de 15 segundos para somar o valor -22,42 em todas as células e exibir tudo novamente no DataGridView.

As operações com matrizes também deram um certo trabalho, mas com algumas horas de tentativa e erro chegamos em um algoritmo funcional. Nesses métodos, não conseguimos pensar em nenhuma maneira de diminuir o número de loops, infelizmente, então podem ser considerados como os mais lentos. Mesmo assim, os resultados de velocidade não foram ruins. As matrizes que foram testadas, que iam até aproximadamente 10x10, não travavam de modo algum o programa, mesmo em multiplicações.

Optamos ainda em não utilizar atributos que indicam Celulas que são usadas para percorrer a matriz, como “atual” ou “anterior”. Ao invés disso, utilizamos uma funcionalidade muito interessante do C# que é o out em parâmetros. Desse modo, as células que iriam ser usadas para percorrer eram instanciadas no próprio método e passadas por parâmetro aos métodos. Quando tais métodos acabavam, as variáveis passadas assumiram os valores desejados. Além disso, quando a operação acaba, esses ponteiros para percorrer são descartados pelo próprio sistema operacional, o que dá um gasto menor de memória, ainda que quase insignificante.

O uso de um Indexer, visto na classe VetorDado no primeiro semestre, foi uma forma alternativa para retornar o valor de uma célula, visto que facilitaria para o usuário dessa classe usar colchetes, como em uma matriz normal, ao invés de uma método específico, como ValorDe.

Conclusão

Foi um projeto que acrescentou muito na compreensão da matéria de Estrutura de Dados, ensinou que para ter um bom programa que é funcional, o equilíbrio entre o uso de memória e tempo gasto para as operações foi compreendido e percebeu-se que esse equilíbrio é uma das bases de Estrutura de Dados.