MAC0331 - Geometria Computacional

Segundo Projeto - Todas as interseções de uma coleção de círculos

Pedro Gigeck Freire - 10737136

Pequeno relatório sobre a implementação do segundo projeto da matéria, com os desafios, problemas e soluções encontrados ao longo do percurso!

A ABBB

O primeiro passo foi implementar a ABBB. Eu optei por uma árvore rubro-negra e me baseei nesse site https://algorithmtutor.com/Data-Structures/Tree/Red-Black-Trees/, ele já traz uma implementação em python, mas era pouco documentada e um tanto confusa, eu fui reescrevendo o código e mundando o que achava necessário, isso me trouxe algumas dores de cabeça mas deu tudo certo. Minha ABBB está em `\geocomp\common\abbb.py`.

Trabalhando com círculos

Nossa plataforma dispõe de uma classe para trabalhar com discos, então dei uma complementada nela com as funções que eu precisei para os círculos. Inseri minhas modificações no final arquivo `\geocomp\common\disc.py`. Com os discos, conseguíamos apenas desenhar os discos preenchidos, sem indicar a cor do círculo, então fiz as funções para desenhar somente os círculos, *`hilight_circ`* e *`unhilight_circ`*, essas funções já estavam implementadas no `control.py`, então só precisei chamá-las. Outra função gráfica que precisei foi a de desenhar meio-círculos (os arcos que usamos no algoritmo), então criei as funções *`hilight_semi_circle`* e *`unhilight_semi_circle`*, essas não estavam implementadas, eu pensei em usar a função *plot_curve*, mas decidi fazer a minha específica, então criei um *`plot_semi_circle`* em `control.py` e em `tk.py`. Por fim, precisei das funções que detectam interseção: *`intersects`*, que apenas detecta e *`intersection`* que retorna uma lista com o(s) ponto(s) de interseção, nessa última tem um tanto de continhas geométricas chatas.

Enquanto eu fazia essas funções, eu precisava de um jeito de testá-las, então acabei criando um algoritmo de força bruta bem simples em `geocomp\interseçãocirc\brute_force.py`.

O Algoritmo

Depois de preparar o terreno, fiz o passo a passo para incluir o novo algoritmo, criei a pasta interseçãocirc com o arquivo `__init__.py` e a minha implementação em `bent_ott_mod.py` (Bentley e Ottmann Modificado).

A casca externa do algoritmo é muito parecido com o Bentley e Ottmann original, há duas ABBBs, uma para os pontos eventos e outra para os semi-círculos (= arcos). Os pontos eventos são guardados na classe `Node_Point_Circle` e os arcos na `Node_Semi_Circle`, depois de obter os pontos eventos no pré-processamento, processamos: pros pontos da esquerda inserimos os circulos na linha, pros da direita removemos e pros de interseção trocamos a ordem.

Tivemos apenas que tratar um caso degenerado: Quando a interseção não troca a ordem dos arcos. Isso é, a interseção é um ponto de tangência entre os dois círculos, então o ponto ganhou outro atributo, sabendo quais interseções são as gerais e quais são degeneradas (chamei de interseção única)

O teste de esquerda

Para manipular os arcos na linha de varredura, era preciso o teste de esquerda. Para isso, cada arco tem um ponto de referência, assim comparamos um arco com o ponto de referência do outro. No código, explicitei os casos gerais, temos que verificar se o ponto está dentro ou fora do arco, e acima ou abaixo. Houveram, como de costume, casos degenerados:

- O ponto entá bem no meio (na 'linha do equador') do círculo: nesse caso, temos que saber se o ponto faz parte da metade de baixo ou de cima do seu círculo, o que nos fez criar mais um parâmetro.
- O ponto pertence ao arco: nesse caso, estamos tratando um ponto de interseção. Se estivéssemos trabalhando com segmentos, poderiamos trocar o ponto pelo extremo direito do segmento, mas no nosso caso, poderia haver outra interseção, então estariamos entrando no outro arco e o ponto da direita nos diria que estamos fora. Então, ao invés de pegar o ponto do extremo direito, tive que pegar um ponto um pouquinho pra direita (calculando a fórmula do círculo com um x + epsilon).

OBS

Criei alguns outros testes e troquei o nome de DISCOS para circulos. Acabei mudando um pouco a estrutura das pastas, como só estava usando a `tk.py` tirei os outros front-ends, espero que não tenha problema, simplificou um pouco pra mim.