Universidade de São Paulo – USP
Instituto de Ciências Matemáticas e de Computação – ICMC
Departamento de Ciências de Computação – SCC

SCC-5900 – Projeto de algoritmos

Professor Marcelo G. Manzato
mmanzato@icmc.usp.br

Projeto – Algoritmos gulosos (agrupamento)

Data de entrega: 30/05

O objetivo deste projeto é implementar um algoritmo de agrupamento de dados utilizando árvores geradoras mínimas

Este projeto possui duas partes descritas a seguir:

Parte I – Implementação (70% da nota final)

Nesta parte serão implementados dois algoritmos de árvores geradoras mínimas Prim e Kruskal. As implementações devem ser eficientes em termos de complexidade de tempo, conforme discutido em aula:

- Para realizar a implementação do algoritmo de Prim, implemente uma fila de prioridades com uma heap binária. A heap deve organizar as arestas a serem analisadas pelo algoritmo em ordem crescente de peso;
- Para a implementação do algoritmo de Kruskal, implemente uma estrutura de Union-Find conforme descrito em aula. Utilize a estrutura para unir os subconjuntos de vértices encontrados durante a execução do algoritmo.

Organize a sua implementação. Defina um Tipo Abstrato de Dados (TAD) para as estruturas de dados auxiliares. Por exemplo, em C crie módulos para as estruturas de dados e defina uma interface para cada estrutura de dados com as operações definidas pelo TAD em um arquivo header (.h). Separe a definição da interface da implementação das operações criando um arquivo fonte (.c) para cada arquivo header. Em uma linguagem de programação orientada a objetos como C++, crie uma classe para cada estrutura de dados e defina a interface da classe com as operações do TAD.

Faça duas implementações do algoritmo de agrupamento de dados uma utilizando o algoritmo de Prim e a segunda com Kruskal. Esse algoritmo deve receber como entrada um número inteiro *k* que representa o número de clusters. No caso da implementação com Prim, crie a árvore geradora mínima e remova as *k* arestas de maior peso. No caso de Kruskal, induza a árvore geradora mínima até que *k* grupos sejam formados. Mais informações sobre esses algoritmos podem ser encontradas em [3], págs. 157 a 161.

Parte II – Avaliação de Funcionamento (30% da nota final)

Para avaliar o funcionamento do seu algoritmo de agrupamento em um problema simples, vamos utilizar uma base de dados sintética proposta em [1].

A base de dados é ilustrada na Figura 1 e é comporta por pontos em um plano bidimensional.

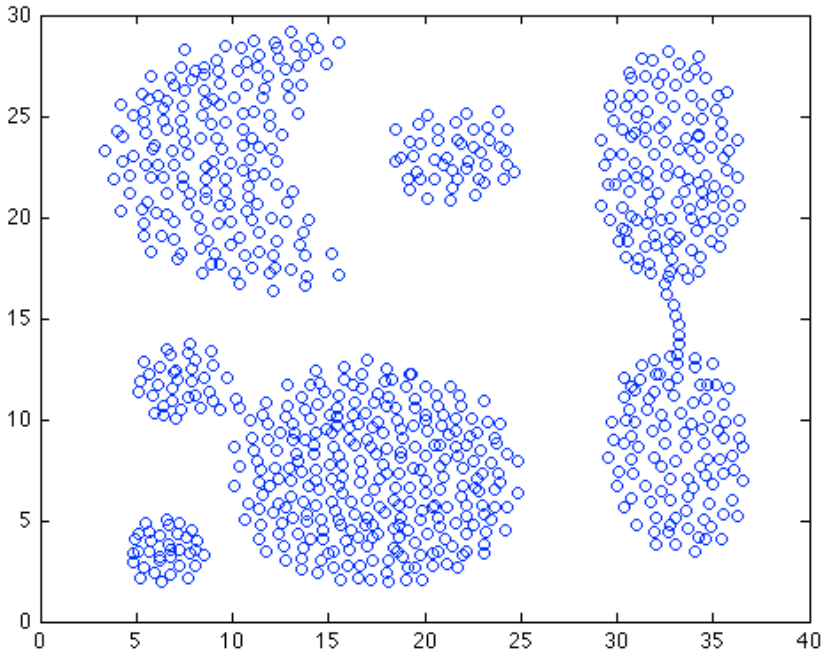

Figura 1: Scatterplot com os dados a serem utilizados no projeto de agrupamento de dados

A base possui um total de 788 pontos armazenados no arquivo texto "dados.txt". O arquivo possui o seguinte formato:

```
15.55   28.65
14.9    27.55
14.45   28.35
14.15   28.8
13.75   28.05
13.35   28.45
13      29.15
13.45   27.5
...     ...
```

Segundo os autores essa base pode ser dividida em 7 grupos. Esses grupos podem ser identificados no arquivo "classes.txt", o qual possui apenas uma sequência de valores inteiros entre 1 e 7. O primeiro valor identifica o grupo do ponto na primeira linha do arquivo "dados.txt", e assim por diante.

```
2
4
2
1
2
5
2
7
...
```

Se associarmos uma cor a cada valor numérico do arquivo “classes.txt” obtemos o resultado ilustrado na Figura 2.

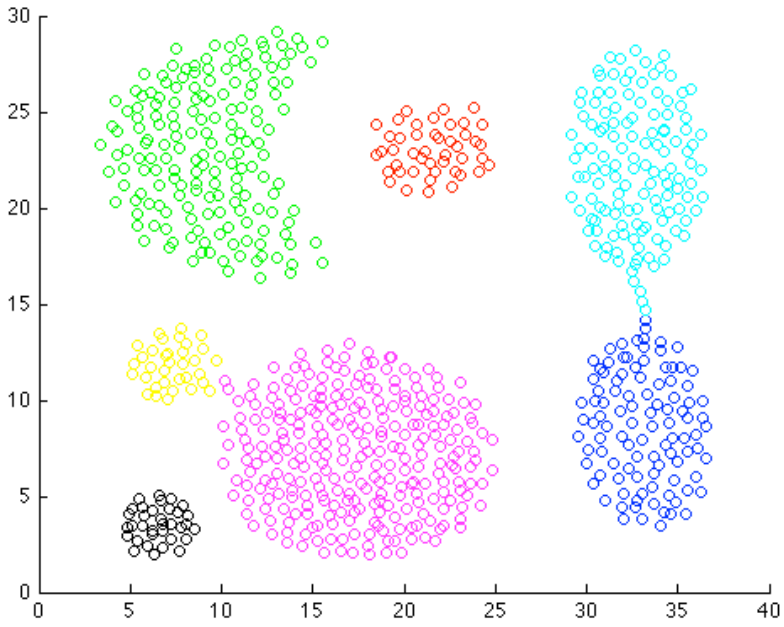

Figura 2: Scatterplot com os grupos sugeridos em [1]

Execute o algoritmo de Prim e Kruskal na base de dados de avaliação e discuta os resultados obtidos. Crie arestas entre todos os pares de pontos ponderadas pela distância Euclidiana. O algoritmo fornece um resultado igual ao sugerido pelos autores do artigo? Se existem diferenças, explique-as utilizando o viés imposto pelo algoritmo implementado. Caso você queira uma medida objetiva de concordância entre os grupos retornados pelo algoritmo implementado e os grupos sugeridos, utilize o Rand index [2].

Faça um curto relatório (2 páginas) explicando a sua implementação e os resultados obtidos na Parte II.

Referências

[1] Gionis, A., H. Mannila, and P. Tsaparas, *Clustering aggregation*. ACM Transactions on Knowledge Discovery from Data (TKDD), 2007. 1(1): p. 1-30.

[2] Rand index. http://en.wikipedia.org/wiki/Rand_index

[3] Kleinberg, J.; Tardos, E. Algorithm Design. Pearson, 2005.