A Heuristic Algorithm for the K- Clique Problem

Problem Outline

$$\boxed{CLIQUE = \{ < G, k > | \; G \text{ has a clique of size } k\}}$$

Clique

Graph G = (V, E), a subset S of the vertices is a clique if there is an edge between every pair of vertices in S, it also means a subset of vertices in V all connected to each other by edges in E

Size of Clique:

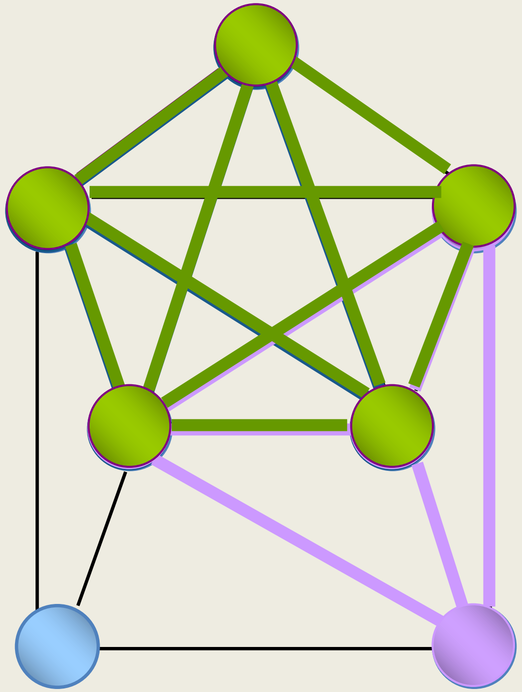
number of vertices it contains

Maximal Clique:

a *clique* cannot be enlarged by adding any more vertices

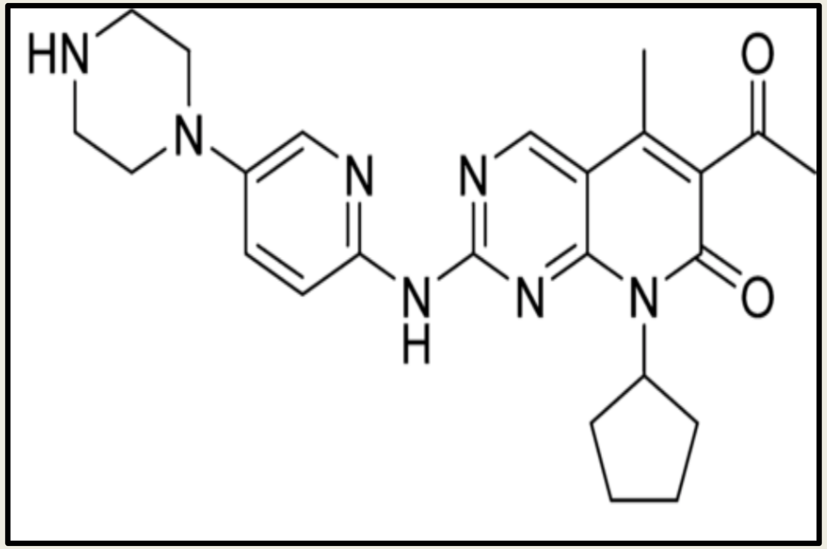
Maximum Clique

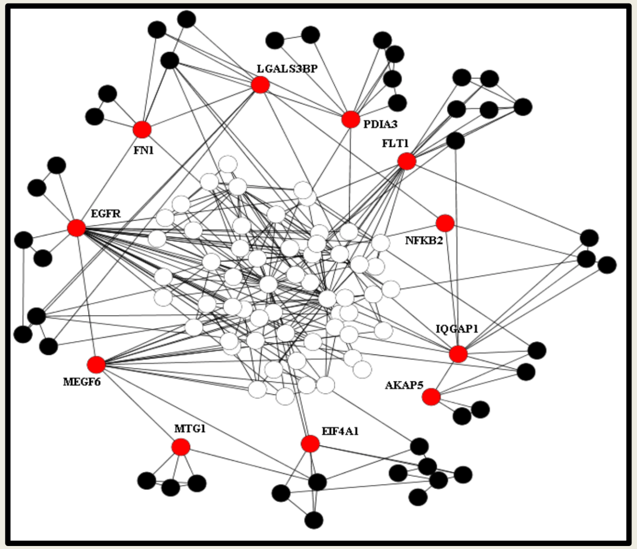
the largest *maximal clique* in the graph



Maximum Clique of Size 5

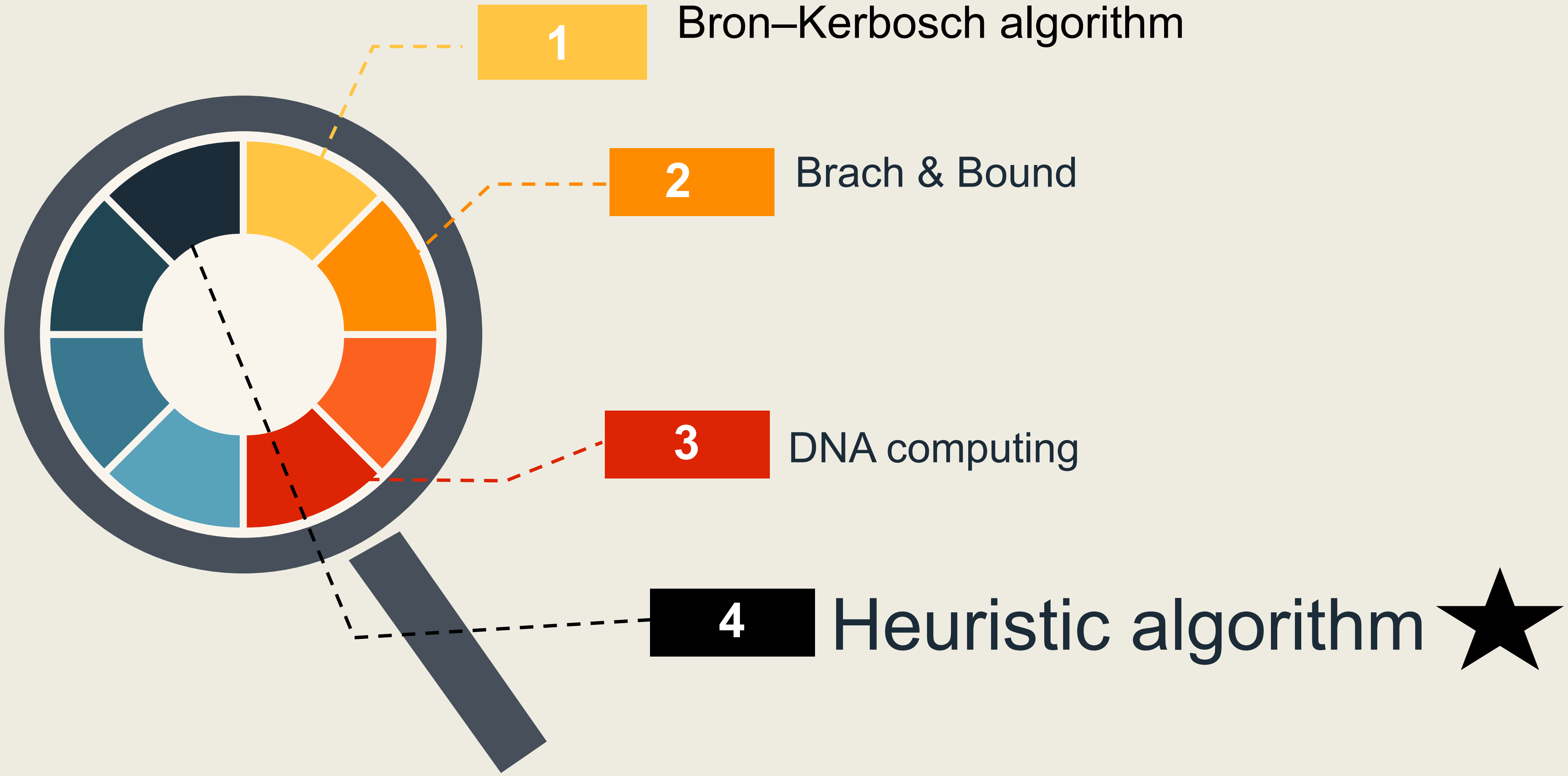
Why is this problem important?





Social Network | **Computational chemistry** | **Bio-information**

Solution Overview



1 Bron–Kerbosch algorithm

2 Brach & Bound

3 DNA computing

4 Heuristic algorithm ★

What is a Heuristic Search?

- A Search Technique that employs a rule that increases the likelihood of finding a solution.
- Domain specific knowledge must be added to improve efficiency.

- **In Our Case:**
  - Node Degree
  - Position relative to other nodes

Solution Details

Test Case Generation

- Generate a random graph of N vertices with a density D.
- Set target clique size k

Preparation

- Remove all vertices with degree < k
- Keep doing this while you can still delete vertices.

Initialization

- For v in V
- v.location = i,i i++

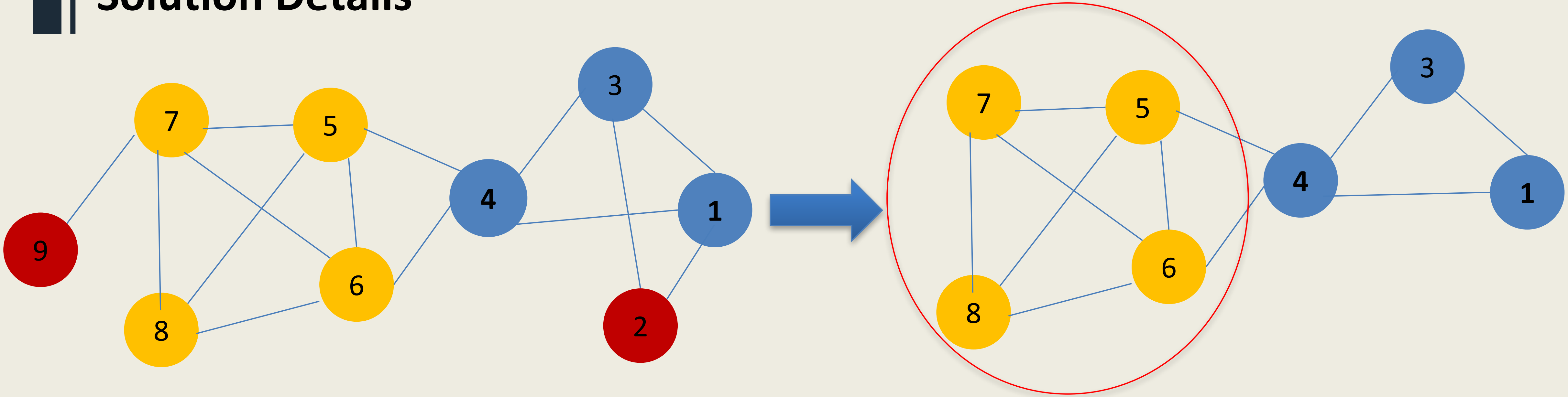
Movement

- For each edge (u,v)
- Move the vertices to the midpoint between them.

Search

- For v in V
- Draw a circle of “unit” distance from v
- Enough points in v?
- All connected?
- Return true
- Return false.

Solution Details



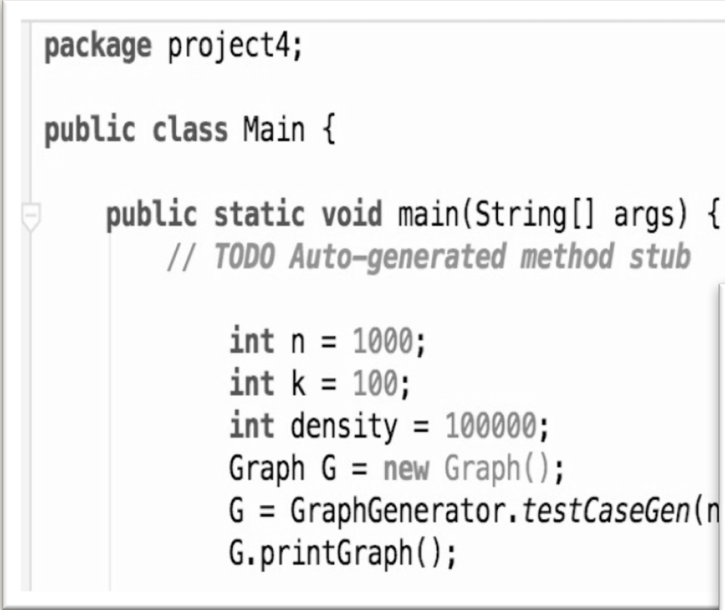
In a clique of size k, each node maintains degree >= k-1
Nodes with degree < k-1 will not be included in the maximum clique

In order to find a clique >3, remove all nodes with degree <=3-1=2

- Remove nodes 2 and 9
- Remove nodes 1 and 3
- Remove node 4

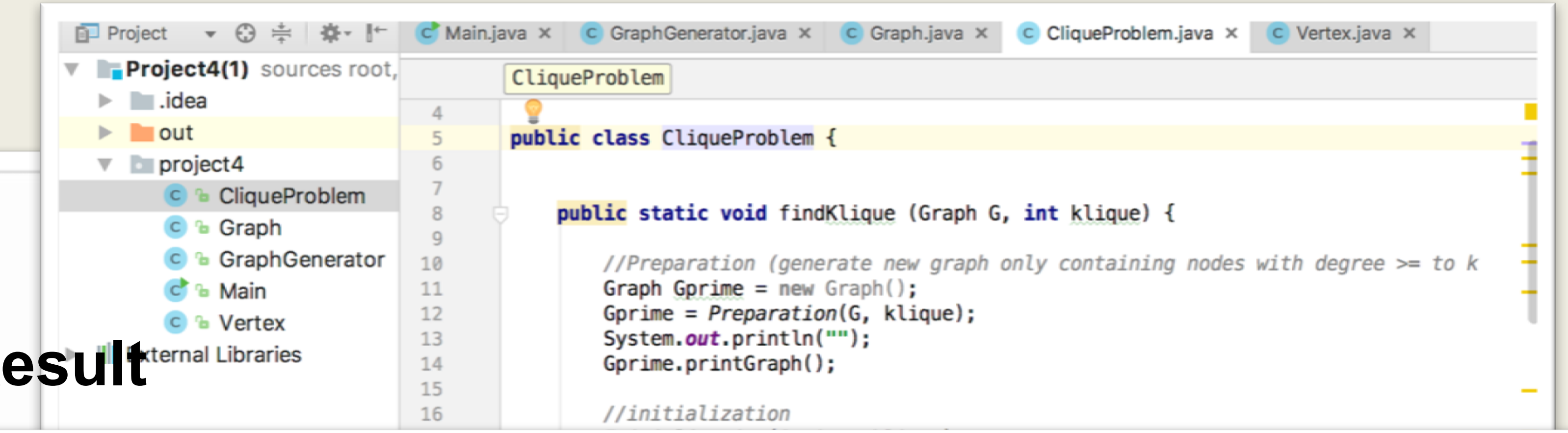
Empirical Results

Inputs



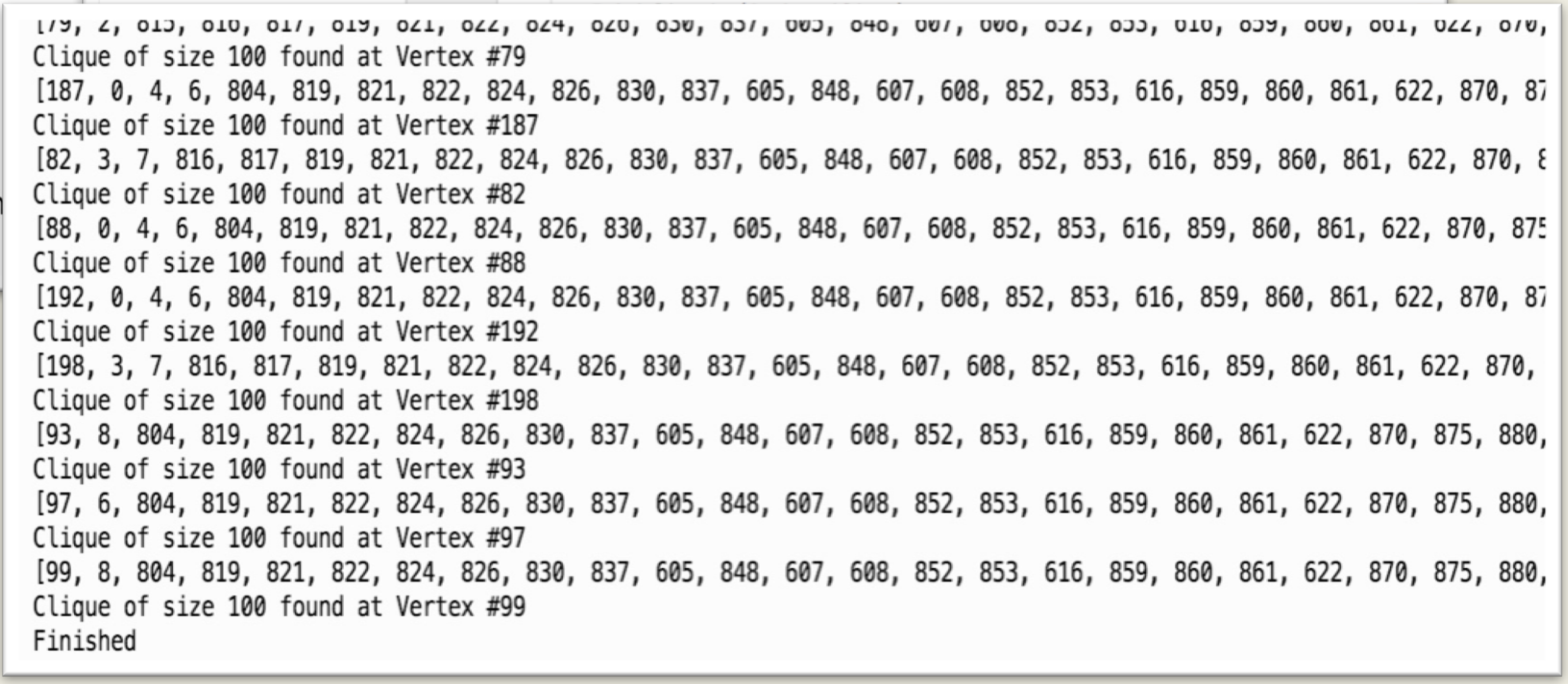
```
package project4;

public class Main {

    public static void main(String[] args) {
        // TODO Auto-generated method stub

            int n = 1000;
            int k = 100;
            int density = 100000;
            Graph G = new Graph();
            G = GraphGenerator.testCaseGen(n
            G.printGraph();
```

Algorithm



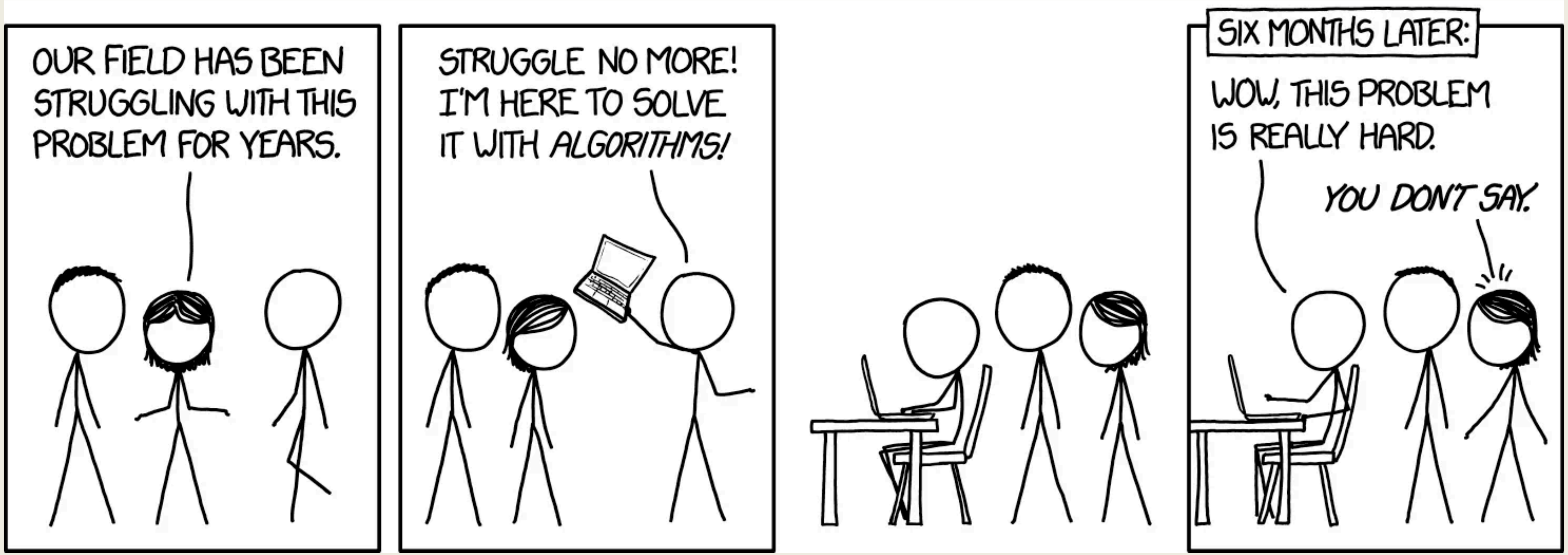
```
public class CliqueProblem {


    public static void findKlique (Graph G, int klique) {

        //Preparation (generate new graph only containing nodes with degree >= to k
        Graph Gprime = new Graph();
        Gprime = Preparation(G, klique);
        System.out.println("");
        Gprime.printGraph();

        //initialization
```

Result



```
[79, 2, 815, 816, 817, 819, 821, 822, 824, 826, 830, 837, 605, 848, 607, 608, 852, 853, 616, 859, 860, 861, 622, 870,
Clique of size 100 found at Vertex #79
[187, 0, 4, 6, 804, 819, 821, 822, 824, 826, 830, 837, 605, 848, 607, 608, 852, 853, 616, 859, 860, 861, 622, 870, 87
Clique of size 100 found at Vertex #187
[82, 3, 7, 816, 817, 819, 821, 822, 824, 826, 830, 837, 605, 848, 607, 608, 852, 853, 616, 859, 860, 861, 622, 870, 8
Clique of size 100 found at Vertex #82
[88, 0, 4, 6, 804, 819, 821, 822, 824, 826, 830, 837, 605, 848, 607, 608, 852, 853, 616, 859, 860, 861, 622, 870, 875
Clique of size 100 found at Vertex #88
[192, 0, 4, 6, 804, 819, 821, 822, 824, 826, 830, 837, 605, 848, 607, 608, 852, 853, 616, 859, 860, 861, 622, 870, 87
Clique of size 100 found at Vertex #192
[198, 3, 7, 816, 817, 819, 821, 822, 824, 826, 830, 837, 605, 848, 607, 608, 852, 853, 616, 859, 860, 861, 622, 870,
Clique of size 100 found at Vertex #198
[93, 8, 804, 819, 821, 822, 824, 826, 830, 837, 605, 848, 607, 608, 852, 853, 616, 859, 860, 861, 622, 870, 875, 880,
Clique of size 100 found at Vertex #93
[97, 6, 804, 819, 821, 822, 824, 826, 830, 837, 605, 848, 607, 608, 852, 853, 616, 859, 860, 861, 622, 870, 875, 880,
Clique of size 100 found at Vertex #97
[99, 8, 804, 819, 821, 822, 824, 826, 830, 837, 605, 848, 607, 608, 852, 853, 616, 859, 860, 861, 622, 870, 875, 880,
Clique of size 100 found at Vertex #99
Finished
```

Limitations


OUR FIELD HAS BEEN STRUGGLING WITH THIS PROBLEM FOR YEARS.
STRUGGLE NO MORE! I'M HERE TO SOLVE IT WITH *ALGORITHMS!*
SIX MONTHS LATER:
WOW, THIS PROBLEM IS REALLY HARD.
YOU DON'T SAY.

References/Citations

1 https://en.wikipedia.org/wiki/Bron–Kerbosch_algorithm

2 https://en.wikipedia.org/wiki/Clique_problem

3 Rossi, R. A., Gleich, D. F., Gebremedhin, A. H., Patwary, M. M. A., & Ali, M. (2013). A fast parallel maximum clique algorithm for large sparse graphs and temporal strong components. *CoRR, abs/1302.6256*.

4 Tomita, E., Tanaka, A., & Takahashi, H. (2006). The worst-case time complexity for generating all maximal cliques and computational experiments. *Theoretical Computer Science*, *363*(1), 28-42.