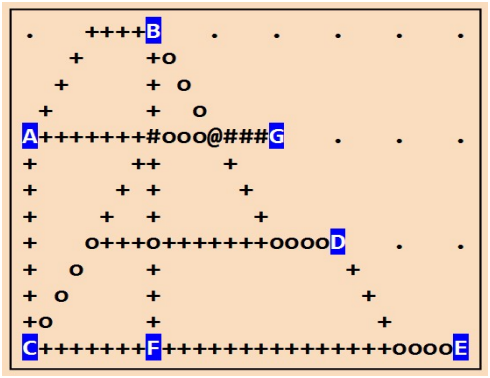
Code / Project : CME1252 / 1
Year / Semester : 2024-2025 Spring Semester
Duration : 7 weeks



Project: Graph Generator

The aim of the project is to develop a graph application on simple graphs.

General Information

In this application, user can generate random simple graphs and make some operations on them. Simple graphs are undirected, have no parallel edges or self loops. The application has a main graph, secondary graph and 9 depot graphs. Each graph has a board with 25*37 squares. Each board has 7*10 specific points (dotted) which can be used for vertices. Letters of the alphabet represent vertices of the graph. The first n letters are used for an n-node graph.

Random Graph Generation

User defines some properties of a random simple graph, then the program generates relation matrix R.

1. Number of Vertices: n
2. Degrees of Vertices: User can give a Degree Sequence or specify degree intervals.

Example:

Number of vertices/nodes/points = 6

Degree Sequence Method: 3, 3, 2, 2, 2, 2 (Each number represents the degrees/connections of a vertex)

Degree Intervals Method:
Min. degree: 3
Max. degree: 5 (Average degree: 4)

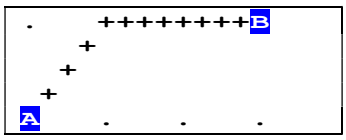
Calculation of Other Matrices

Program calculates R^2, R^3, ... , R^n, R* and R_{min} matrices. R matrix gives directly connected nodes (1 step away). R^2 matrix gives points exactly 2 steps away. R^n matrix gives points exactly n steps away. R* matrix gives all connected points. R_{min} matrix gives the minimum number of steps required for going from point a to point b.

Drawing Graph

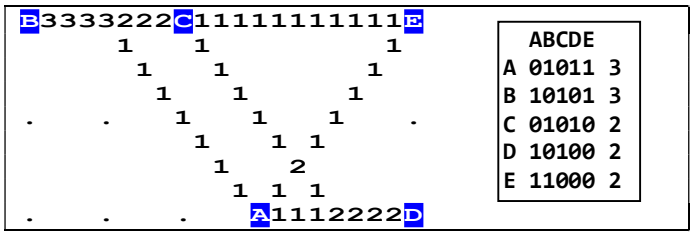
Graphs consists of points/vertices and lines/edges.

Line Drawing Method:



Line from A (x1,y1) to B (x2,y2):
---
1. Change x and y to minimize delta x and delta y (diagonal line segment).
2. Change x or y (straight line segment).

Graph Drawing Method:
1. Determine random vertex coordinates in specific grid pattern (dotted 7*10 squares).
2. Draw lines of the graph with respect to R matrix (use only upper triangle (fromPoint < toPoint)).



penalty=1019

Drawing Modes (Key:D) :

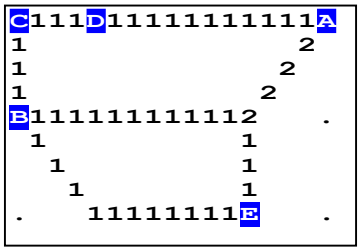
---

0. Use 123456789 for lines. Number represents crossings/overlaps.
1. Use +o#@ (1,2,3,4 or more) for lines.
2. Use only + for lines.

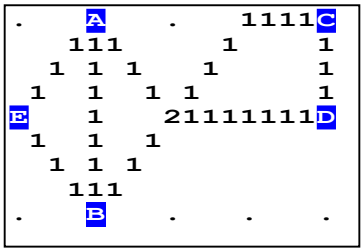
Improving Graph Drawing

Improvement means minimizing the line crossings/overlaps in a graph.

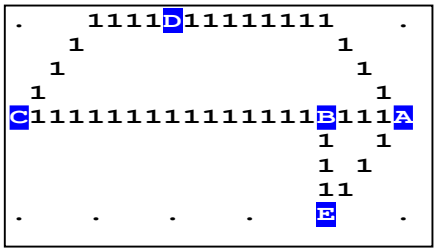
```
Line-line overlaps   = 1 penalty point
Line-vertex overlaps = 1000 penalty points
Total penalty point  = (Number of line overlaps)  +  1000 * (Line-Vertex overlaps)
```



penalty=4



penalty=1



penalty=0

Menus

There are 3 menus. All operations are for the main graph unless stated otherwise.

1.Graph Generation Menu **(Key: Z)**
2.Graph Test Menu **(Key: X)**
3.Graph Transfer Menu **(Key: C)**

Graph Generation Menu

1. Generate graph
2. Set degrees with Degree Sequence
3. Set degrees with mindegree and maxdegree
4. Calculate R^2, R^3, ... , R^n, R* and R_{min} matrices

Graph Generation Menu parameters:
1. Number of vertices: ? 4
2. Degree Sequence: ? 3,2,2,1
3. mindegree: ? 1 maxdegree: ? 3

Graph Test Menu

1. Connected?
2. Contains C_3?
3. Isolated vertices?
4. Complete graph (K_n)?
5. Bipartite?
6. Complete bipartite ($K_{m,n}$)?
7. Cycle graph (C_n)?
8. Wheel graph (W_n)?
9. Star graph (S_n)?
10. Isomorphic? (main graph and secondary graph)

Possible Answers To Graph Tests:
1. Yes/No
2. Yes. Vertices: A,C,D
3. B,D
4. Yes/No
5. Yes. V1={A,C} V2={B,D,E,F}
6. Yes. V1={B,C,D} V2={A}
7. Yes/No
8. Yes. Center:B
9. Yes. Center:D

Graph Transfer Menu (or Keys)

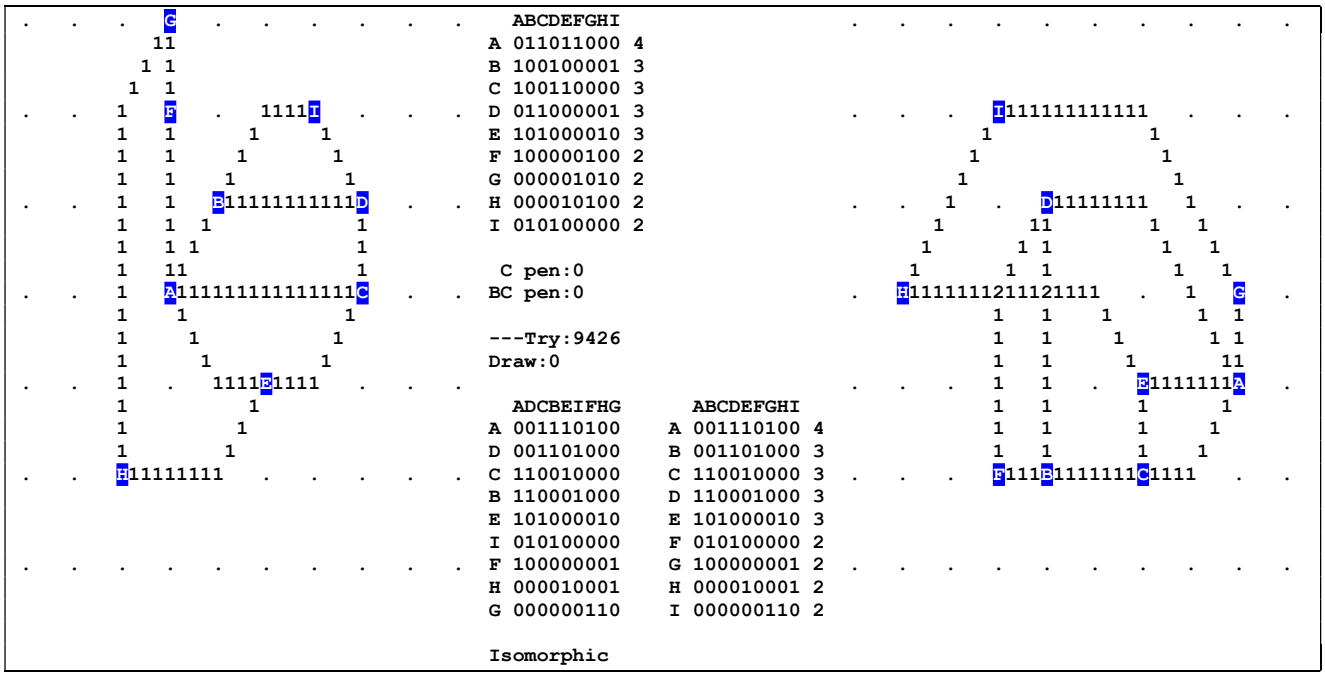
1. Copy main graph to secondary graph **(Key: G)**
2. Copy secondary graph to main graph **(Key: H)**
3. Load a graph file ("graph1.txt") to main graph **(Key: L)**
4. Save main graph to a file ("graph1.txt") **(Key: S)**
5. Copy main graph to a depot graph (1-9) **(Keys: QWE RTY UIO)**
6. Copy a depot graph to main graph **(Keys: 123 456 789)**

Sample graph file:

```
4
A 0 0
B 4 4
C 0 4
D 4 0
0110
1010
1101
0010
```

Graph file format:
NumberOfVertices
VertexID coordx coordy
VertexID coordx coordy
...
RelationMatrix

Sample Isomorphism Screen



Screen Arrangement:

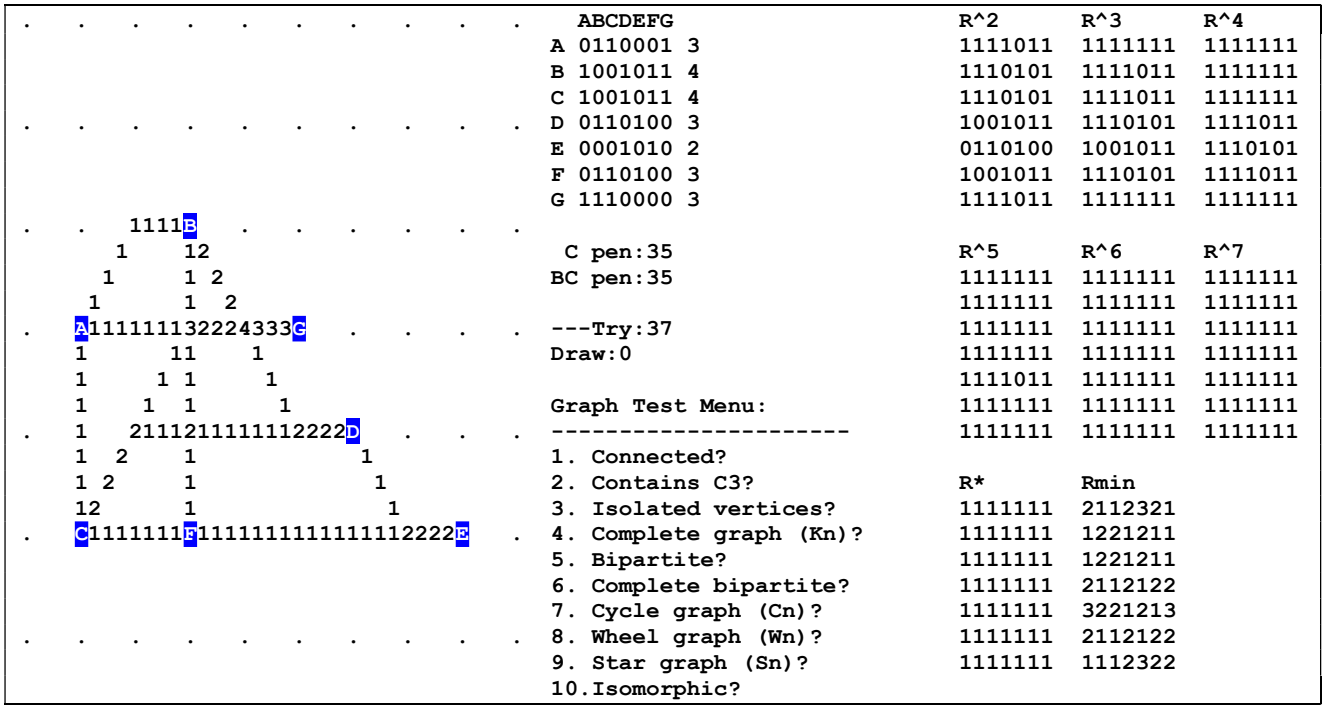
Left side: Main graph (for all operations)
Right side: Secondary graph (for isomorphism) or info field.
Center area: Info field

C pen: Crossing/overlap penalty point (for main graph)
BC pen: Best crossing/overlap penalty point (for main graph)

Try: Graph drawing improvement trial number
Draw: Current drawing mode

Upper left matrix: R matrix for main graph
Lower right matrix: R matrix for secondary graph
Lower left matrix: Altered R matrix for main graph
Isomorphic: Isomorphism test result

Screen for Calculating R^2, R^3, ... , R^n, R* and R_{min} matrices, then Displaying Graph Test Menu



Producing Permutations for Isomorphism

Permutations for 5 elements (ordered):

0. 12345	6. 13245	12. 14235	18. 15234
1. 12354	7. 13254	13. 14253	19. 15243
2. 12435	8. 13425	14. 14325	20. 15324
3. 12453	9. 13452	15. 14352	...
4. 12534	10. 13524	16. 14523	118. 54312
5. 12543	11. 13542	17. 14532	119. 54321

Question: How can we find the 83rd permutation?

Step 1. Finding factorial base number for 83

```
(int)(83/4!) = (int)(83/24) = 3              ( ---> first digit:   3 )
Remaining: 83 - 3*24 = 83 - 72 = 11

(int)(11/3!) = (int)(11/6) = 1              ( ---> second digit:  1 )
Remaining: 11 - 1*6 = 11 - 6 = 5

(int)(5/2!) = (int)(5/2) = 2                ( ---> third digit:   2 )
Remaining: 5 - 2*2 = 5 - 4 = 1

(int)(1/1!) = (int)(1/1) = 1                ( ---> fourth digit:  1 )
Remaining: 1 - 1*1 = 1 - 1 = 0
```

Factorial base number: 3121

Step 2. Finding the 83rd permutation by using the factorial base number

```
for 3: (1,2,3,4,5)        ( ---> first digit of 83rd permutation:  4 )
        0 1 2 3

for 1: (1,2,3,5)          ( ---> second digit of 83rd permutation: 2 )
        0 1

for 2: (1,3,5)            ( ---> third digit of 83rd permutation:  5 )
        0 1 2

for 1: (1,3)              ( ---> fourth digit of 83rd permutation: 3 )
        0 1

       (1)                ( ---> fifth digit of 83rd permutation:  1 )
        0
```

83rd permutation: 42531

Suggested Weekly Program

1. Discussing and designing solution alternatives. Designing classes.
   Creating the necessary variables, structures. Screen.
   Determining team leader. Planning task distribution and scheduling.
2. Generating graph. Calculating R^2, R^3, ... , R^n, R* and R_{min} matrices.
3. Drawing the graph. Graph Transfer Menu operations.
4. Graph Test Menu (Operation: 1,2,3,4). (---First Evaluation Week---)

5. Improving the Graph. Graph Test Menu (Operation: 5,6,7,8,9).
6. Producing permutations for isomorphism. Isomorphism test.
7. Remaining parts of the application. Testing/Debugging. (---Final Evaluation Week---)

First Evaluation: 21.3.2025 **Report:** 21.3.2025	**Final Evaluation:** 11.4.2025 (presentation **in English**: powerpoint+poster) **Report:** 11.4.2025