Faculty of Mathematics and Physics
Charles University in Prague
11th May 2015

C# Made Easy!

Programming II

Workshop 12 – Graph Algorithms

Workshop 12
Outline

1. Test
2. Workshop Finals
3. Graph Algorithms
4. Homework

Questionnaire 1

No Test

Find the test here (no-ads):
https://goo.gl/SgKfgz

Permanent link:
https://docs.google.com/forms/d/1RddJXRSUBC5sPJowEY4Hgg982rmTaVSlgZ2JYB_bL7k/viewform

Time:
As much as you need

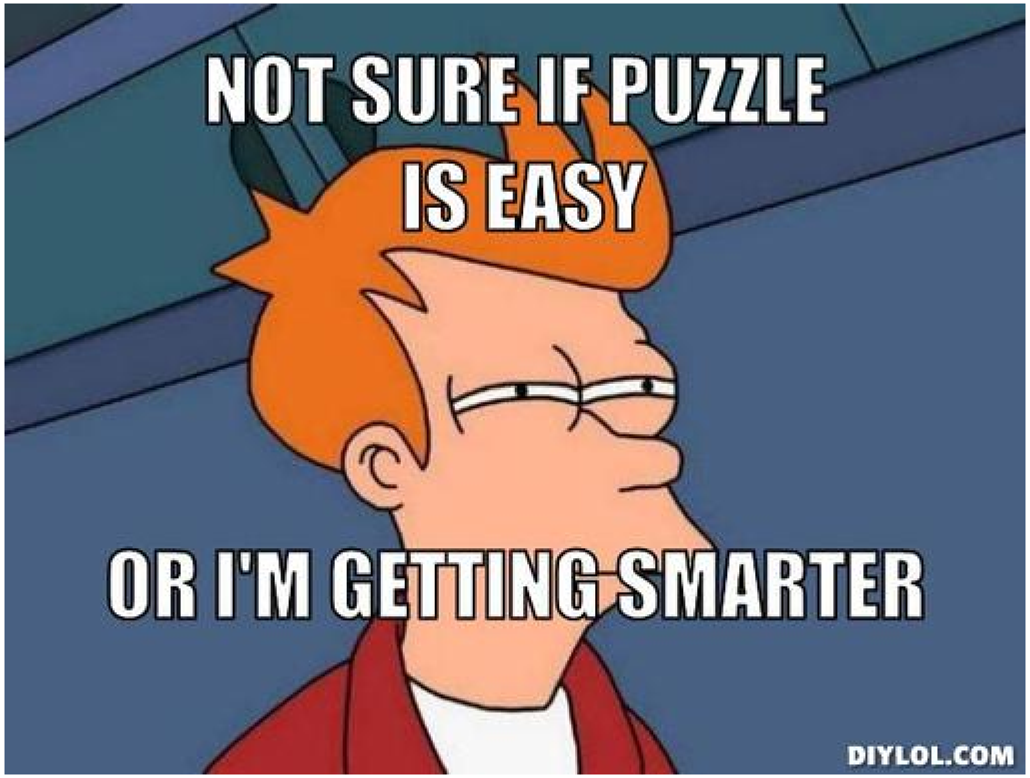
Workshop 12
Finals

You will code the **Final Workshop Test** next week during **Workshop 13, 18.5.2015**

You should revisit:

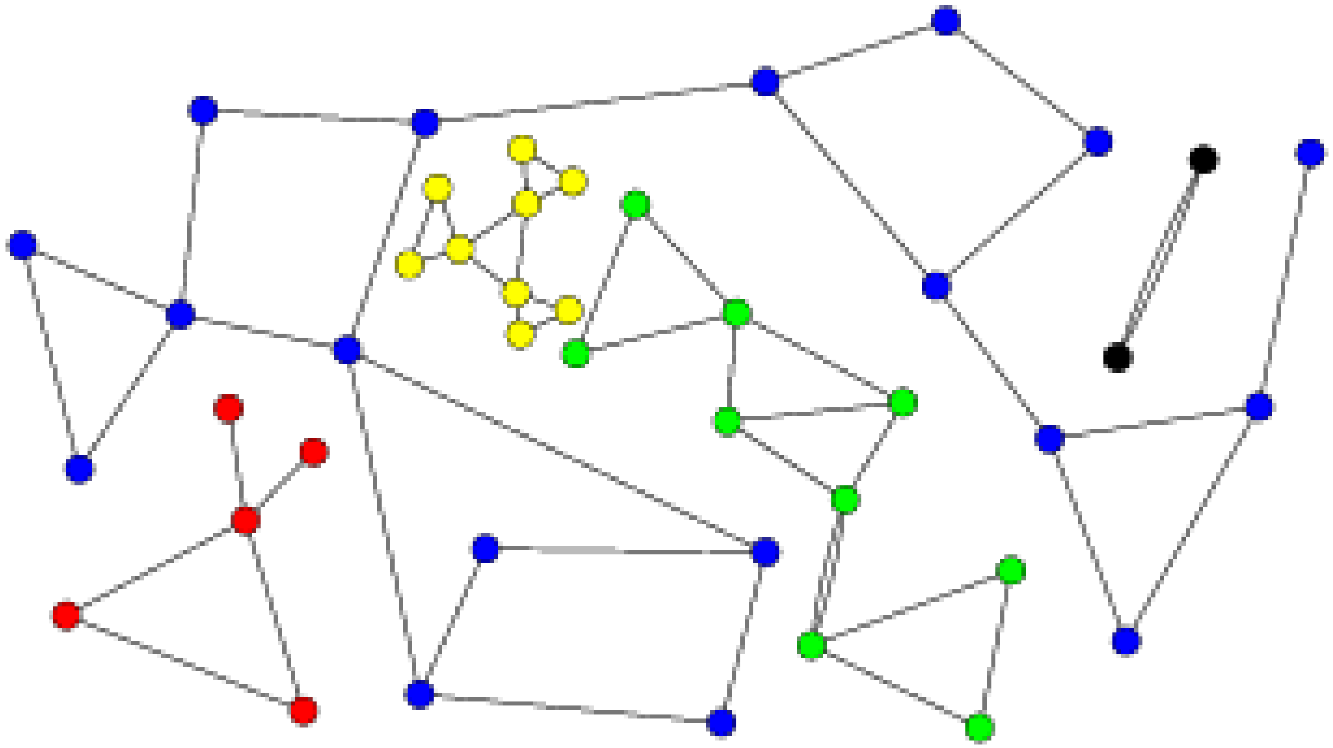
- Recursion
- Graph representation
- DFS, BFS
- Minimax for games

In order to feel like…



Graph Algorithms

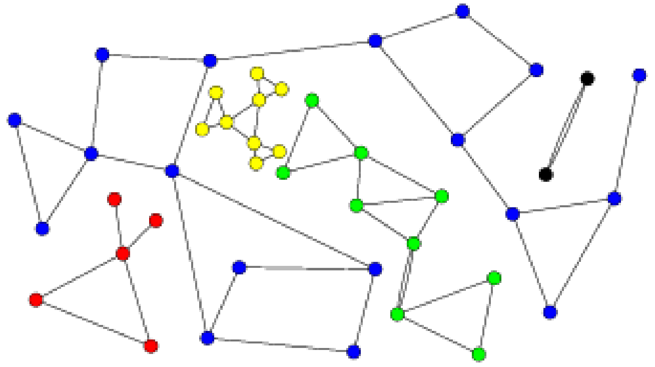
1. Components



Graph Algorithms

1. Components

Algorithm?



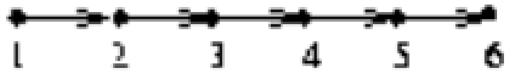
Use BFS or DFS to label nodes of single component, always start from unlabelled node.

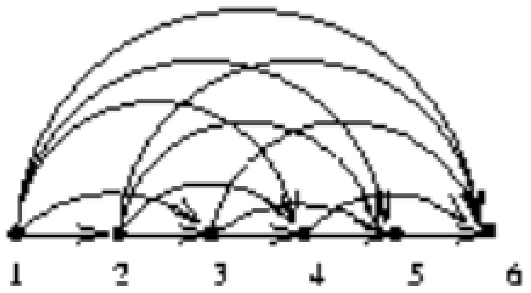
Repeat it as long as there are any unlabelled nodes in the graph.

Complexity?

Graph Algorithms

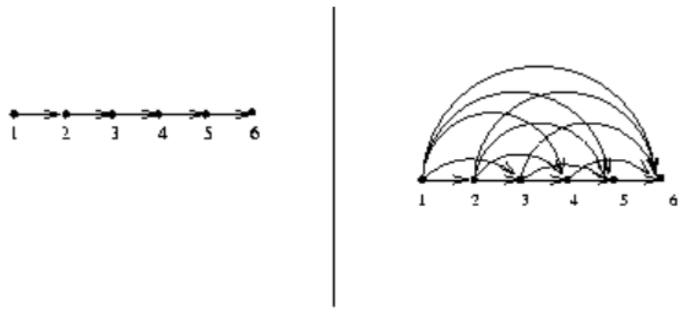
2. Graph transitive closure





Graph Algorithms

2. Graph transitive closure



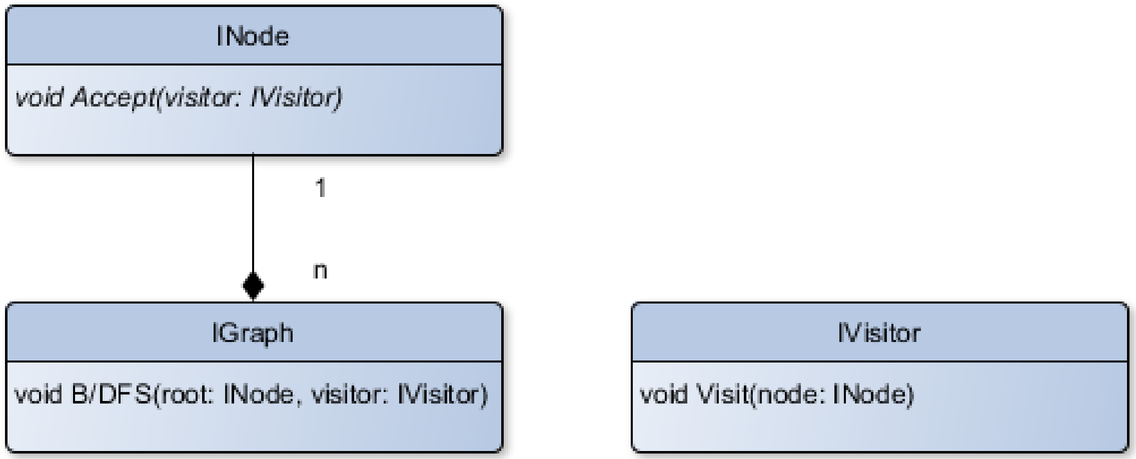
Algorithm?

For every Vertex:
Launch DFS or BFS and introduce new edges
when new vertex is reached.

Complexity?

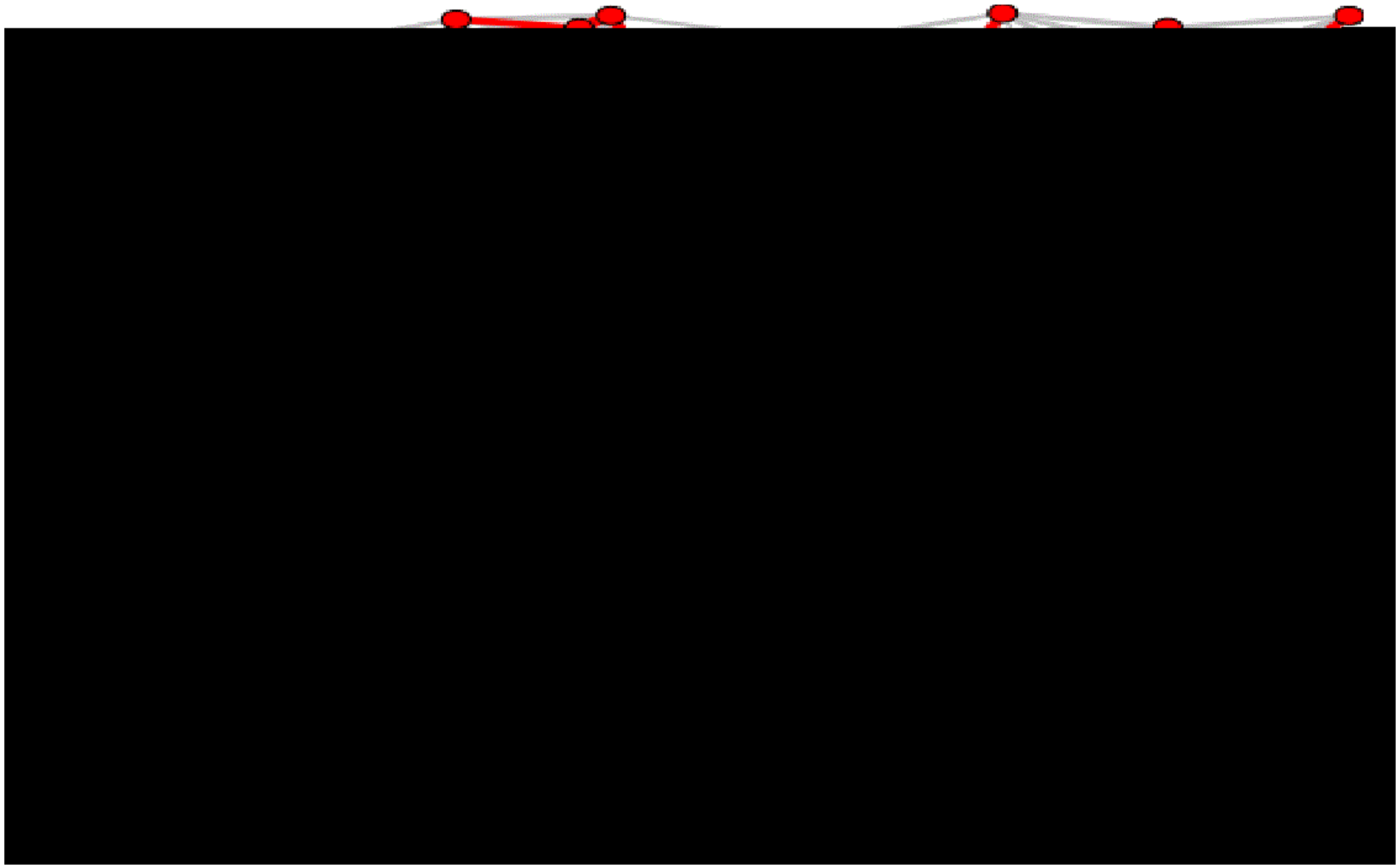
Graph Algorithms
1+2 Implementation?

- We have two graph algorithms using B/DFS … can we somehow split the implementation between “bare” B/DFS and “algorithm internals” ?


INode
void Accept(visitor: IVisitor)
1
n
IGraph
void B/DFS(root: INode, visitor: IVisitor)
IVisitor
void Visit(node: INode)

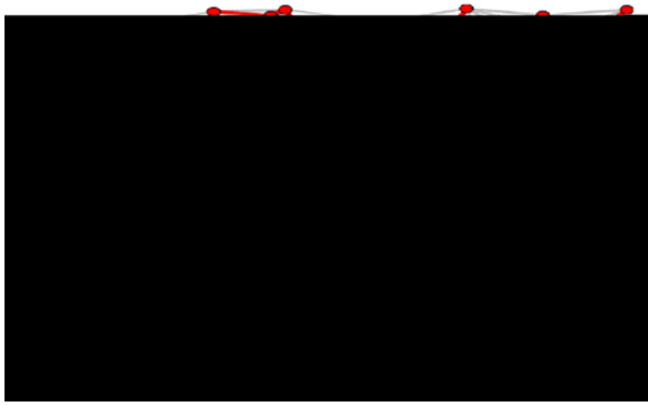
Graph Algorithms

3. Minimum spanning tree



Graph Algorithms

3. Minimum spanning tree



Algorithm?

Kruskal's hungry algorithm:

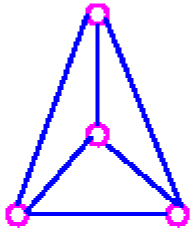
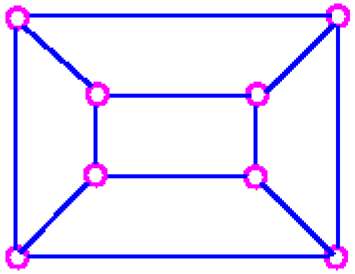
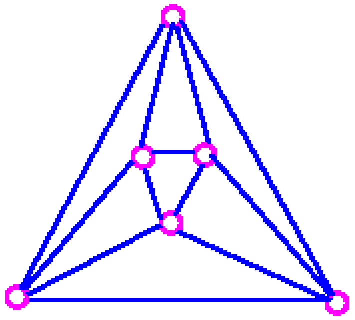
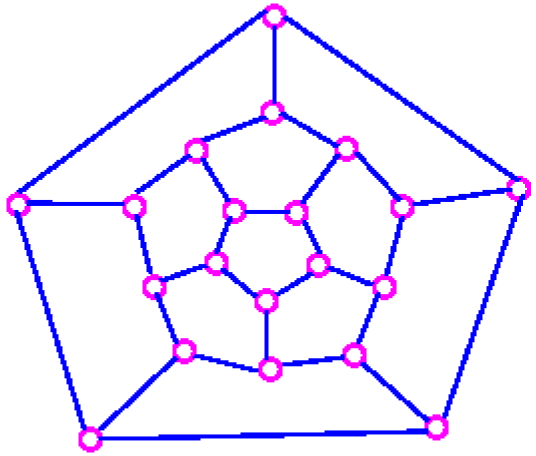
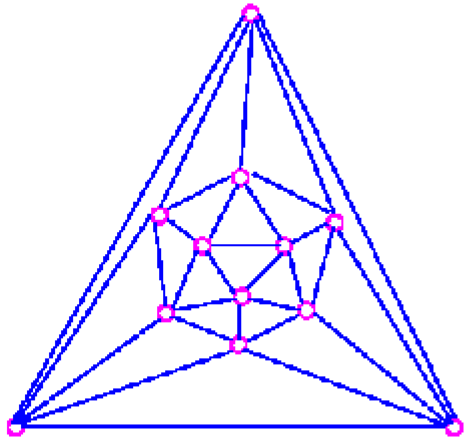
For every component:

1. Order edges according to their value
2. For each edge … add it to the result if it does not form the circle with already included edges

Complexity?

Graph Algorithms

4. Drawing planar graph

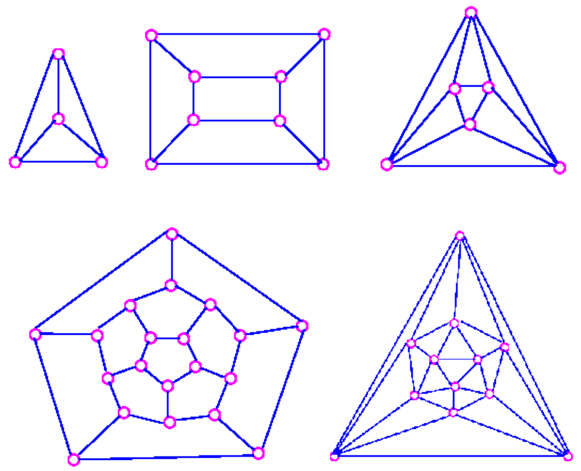






Graph Algorithms

4. Drawing planar graph

Approximate Algorithm?

“Springy”!



Assignment 12
Graph algorithms

- Implement a GUI application that provides visualization of the graph via spring-algorithm

- Provide buttons for computing:
  - Component labeling
  - Graph transitive closure of all components
  - [BONUS] Minimum spanning tree of all components

Assignment 12
Graph algorithms

GRAPH INPUT:

<int> '\n' [<node> ' ' <link> ' ' <node> '\n']+

<node> :	[a-zA-Z]+
<link>:	[<non-oriented-link> | <oriented-link>]
<non-oriented-walk-link>:	'<--(' <int> ')-->'
<oriented-walk-link>:	'--(' <int> ')-->'

Assignment 12
Send me an email

- Email: jakub.gemrot@gmail.com
- Subject: **Programming II – 2015 – Assignment 12**
- Zip up the whole solution and send it
- You WILL NOT find the assignment in CoDex!
- Deadline:
  - **17.5.2015 23:59**
- Points: 10 + 5 (Minimum spanning tree) + 3 (meeting the deadline)

Questions?

I sense a soul in search of answers…

- Sadly, I do not own the patent for perfection (and will never do)
- In case of doubts about the assignment or some other problems don't hesitate to contact me!
  - Jakub Gemrot
    - gemrot@gamedev.cuni.cz