various

Lecture Notes of Spring 2013

Algorithms I

University of Mannheim
2013

This script originates from the course "‘Algorithms I"’ at the University of Mannheim as lecture notes.

The accuracy of its content is not guaranteed and the author(s) do not assume responsibilitty for possible damages of any kind. This lecture notes are not an official document released by employees of the University of Mannheim, hence those do not assume responsibility, as well.

Many thanks to Ingo Bürk, who initially published the underlying LaTeX template here.

Contents

1
TODO: 1st lecture missing

TODO

Definition 1.1
TODO

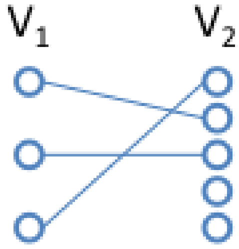
Definition 1.2
TODO

Definition 1.3
TODO

Lemma 1.1
TODO

Definition 1.4
Let $G = (V, E)$ be a graph without loops. If there exists $V_1, V_2 \leq V$ and $V_1 \cup V_2 = V$ such that $V_1 \cap V_2 = \emptyset$ and every edge e has one endpoint in V_1 and the other in V_2, then we call G a bipartite.

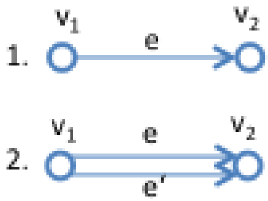
Beispiel



Definition 1.5
A directed graph is a pair $G = (V,E)$ where V is a set of nodes (vertices) and E is a set of edges together with a function $i : E- > VxV$. If $i(e) = (v_1, v_2)$ then v_1 is called start point, v_2 is called end point.

Graphically:

If $i(e) = (v_1, v_2)$ we draw 1.
If $i(e') = (v_1, v_2)$ then this indices a second edge (2.).
If $i(e_1) = i(e_2)$ we call e_1, e_2 parallel.
If $i(e) = (v, v)$ then e is called a directed loop.



$g_{out}(v)$ is the number of edges that have starting point v.
$g_{in}(v)$ is the number of edges with endpoint v.

Lemma 1.2
$\sum\limits_{v \in V} g_{in}(v) = \sum\limits_{v \in V} g_{out}(v)$

Proof: We start with a graph without edges. Then we insert one after the other edges in E. Each edge contributes 1 to both sides of the equation.

Definition 1.6
A directed path is a sequence of edges e_1, e_2... such that the end point of e_i is the start point of $e_1 + 1, i > 1$([NW] +1 seems strange to me, correct?).

A directed path $e_1 \ldots e_k$ is called a (directed) cycle, if the start point of e_1 and the end point of e_k coincide.
A simple (directed) path is a path where every node occurs at most once.
A directed cycle is called simple if every node except for the start and end node occurs at most once.

Definition 1.7
A graph directed or undirected is called simple, if it does not contain parallel edges.

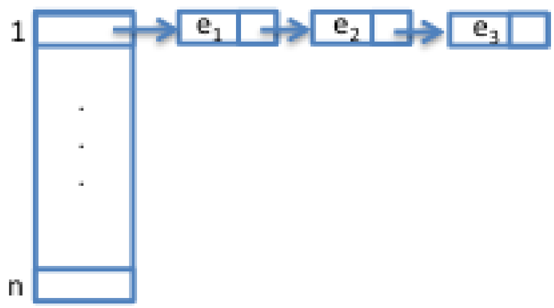
Definition 1.8
A directed graph is called strongly connected if for any pair of nodes (u,v) there is a directd path from u to v.

Let G be a directed graph $G = (V,E)$. $x, y \in V x \sim y$ ([NW] does ~mean are "connected"?) if there is a directed path from x to y and vice versa.

The equivalence classes of this relation $cVxV$ are called strongly connected components. (Analogously: Define connected components for undirected graphs)

We should know how the following terms are defined: reflexivity, symetry, transitivity.

Implementation:
1. Adjacency Lists $V = 1 \ldots n, E = e_1 \ldots e_t$



2. Dynamically changing graphs:
e.g. multi user databses: Nodes ≡ transactions of user; Edges ≡ waiting situations

1	1	2
2	1	3
3	1	2

Graph is used to detect dead locks. Waiting arises when data are locked by a user that modifies these data.
U_1 $write(d)$, $read(d')$
U_2 $read(d)$, $write(d')$

Definition 1.9
An undirected graph is called a tree if it is connected and does not have simple cycles.
Let G be a directed graph, $G=(V,E)$. A node r is called root if every other node can be reached from r via a directed path.
A directed graph is called a tree if it has a root and the underlying undirected graph is a tree.
Let G be a directed graph. A node is called source if $g_{in}(v)=0$. v is called sink if $g_{out}(v)=0$

Lemma 1.3
NW: what was lemma 1.3? the next one was 1.4 in my notes

Lemma 1.4
If $G=(V,E)$ is a directed graph without directed cycles, then tere is always a source and sink.

We use this theorem to detect cycles

Proof: Source (sink analogously): Select an arbitrary node v_1. If v_1 is a source we are done. If it is not, then there must be an edge e_1 leading to it $v_2 \xrightarrow{e_1} v_1$.
If v_2 is a source we are done. If not, there must be an edge e_2 leading to it $v_3 \xrightarrow{e_2} v_2 \xrightarrow{e_1} v_1$. We continue this process. It must stop because there are only finitely many nodes and if a node would appear once more on such a path, there would be a directed cycle.

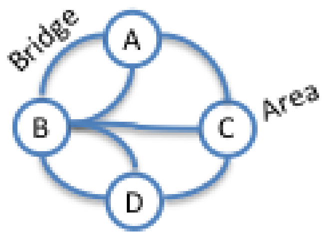
2
Euler Graphs and Hamilton Graphs

TODO

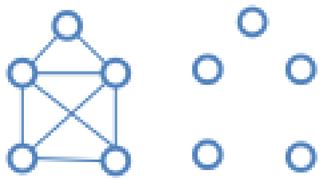
2.1 Euler Graphs

2.1.1 Euler 1736: Königsberger Brückenproblem

Is it possible to do a round walk crossing every bridge exactly once?



Beispiel 2.1



Definition 2.1
Let G be a finite undirected graph. A path $e_1..e_t$ is called a euler path if every edge in E occurs exactly once in the list.
A graph is a euler graph if it has a euler path.

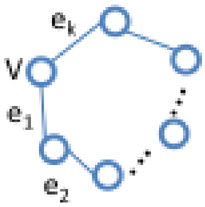
Theorem 2.1
A finite connected graph is a euler graph if and only if:

i) It ha eiter exactly two nodes of odd degree. or

ii) All nodes have even degree.

In the last case the path is a cycle. In the first case no euler path is a cycle. Check is possible in linear time.

Proof: " > " Let $G = (V, E)$ be a graph that has a euler path that is not a cycle. Let $|E| = k$
$\circ \xrightarrow{e_1} \circ \xrightarrow{e_2} ... \circ \xrightarrow{e_k}$ In this path v_1 and v_{k+1} have od degreee and all other nodes have even degree.
Now consider the case teht G has a euler cycle.



Hence every node has even degree.

" < " Let G be a graph with exactly two nodes with odd degree, let this be a and b. We contradict a euler path as follows:
Start at node a and folow an edge ?inktt? on a. $a - -v$ If the node that we currently deal with is neither a nor b then passing this node will use up two edges. If we look at the residual degree at v, it is an even number. If at some point we reach a node which we can enter but not leave it must be b.

3
TODO

「TODO 」