

**Graphs And Algorithms**

Peeking into Computer Science

Peeking into Computer Science

Jalal Kawash

© Jalal Kawash 2010

- Mandatory: Chapter 3 – Sections 3.1 & 3.2




**Reading Assignment**

Peeking into Computer Science

© Jalal Kawash 2010

2




**Graphs**  
Abstraction of Data

3

At the end of this section, you will be able to:

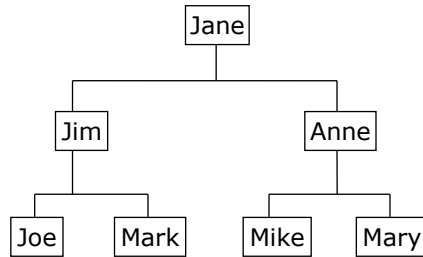
1. Define directed and undirected graphs
2. Use graphs to model data
3. Use graph terminology
4. Represent graphs using adjacency matrices

 **Objectives**

*Peeking into Computer Science* © Jalal Kawash 2010

4

- To show relationships (people, objects, locations etc.)



## JT's Extra: What Is A Graph Used For?

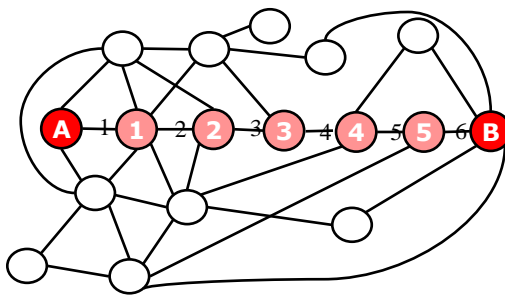


Peeking into Computer Science

© Jalal Kawash 2010

5

- Visualizing connections e.g., "6 degrees of separation"



## JT's Extra: What Is A Graph Used For?



Peeking into Computer Science

© Jalal Kawash 2010

6

- They can be used anytime that the relationships between some entities must be visualized.
- Examples:
  - A few from will be used in the following slides:
    - [http://www.graph-magics.com/practic\\_use.php](http://www.graph-magics.com/practic_use.php)

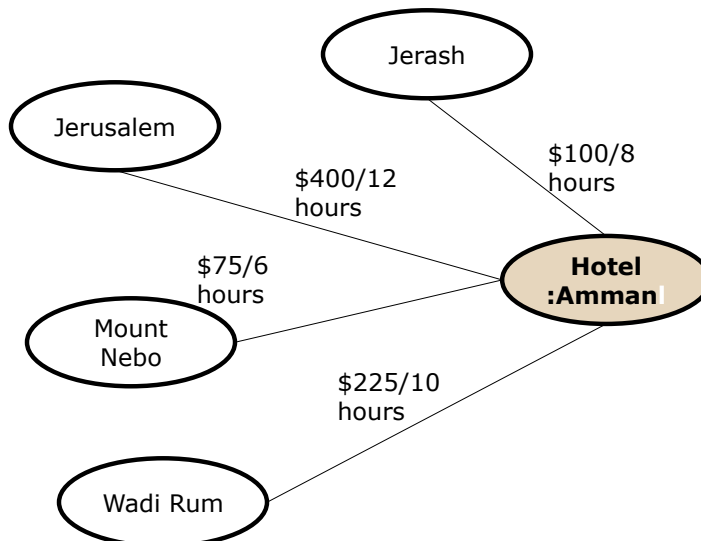
## JT's Extra: Practical Applications (Graphs)



Peeking into Computer Science

© Jalal Kawash 2010

7



## JT's Extra Example: JT's Vacation

Peeking into Computer Science

© Jalal Kawash 2010

8

- Logistics:
  - Making sure that you deliver your items to every street within a part of the city. You want to cover every street but at the same time minimize travel time.
  - Telecommuincations: Find the cheapest way to connect communication stations (TV, telephone, computer network) so that a station is connected to any other (directly, or through intermediate stations).



## JT's Extra: Other Examples<sup>1</sup>

<sup>1</sup> [http://www.graph-magics.com/practic\\_use.php](http://www.graph-magics.com/practic_use.php)  
Peeking into Computer Science © Jalal Kawash 2010

9

- (More logistics)
  - A warehouse should be placed in a city (a region) so that the sum of shortest distances to all other points (regions) is minimal. This is useful for lowering the cost of transporting goods from a warehouse to clients.



## JT's Extra: Other Examples<sup>1</sup> (2)

<sup>1</sup> [http://www.graph-magics.com/practic\\_use.php](http://www.graph-magics.com/practic_use.php)  
Peeking into Computer Science © Jalal Kawash 2010

10

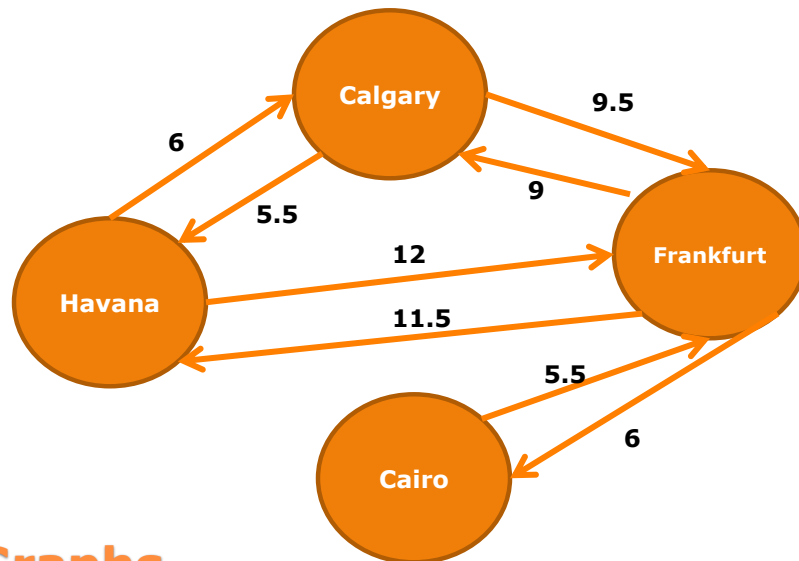
- Solving everyday problems by abstracting details:
  - In this case the problem is represented by (abstracted into) a graph.
  - Other types of problems can be abstracted into other structures such as trees (later section).
- Removing extraneous details through abstraction may make the problem easier to solve (or even make the seemingly impossible possible).



## JT's Extra: Why Learn Graph Theory?

Peeking into Computer Science

© Jalal Kawash 2010



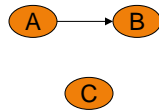
## Graphs

Peeking into Computer Science

© Jalal Kawash 2010

12

- To graphically show the relations in a set
- Example graph:
  - $S = \{A, B, C\}$
- A graph can show the relationships between elements



## JT's Extra: How Will Graphs Be Used In This Section



Peeking into Computer Science

© Jalal Kawash 2010

- A graph  $G$  is defined as:

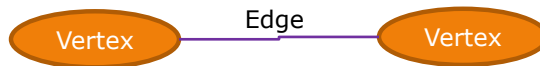
$$G = (V, E)$$

where

$V(G)$  is a set of vertices

$E(G)$  is a set of edges (pairs of vertices)

JT's note: Vertices are the 'things' being connected, edges are the connectors.



- Directed Graph: edges are one-way
- Undirected Graph: edges are two-way
- Labeled Graphs: edges have weights

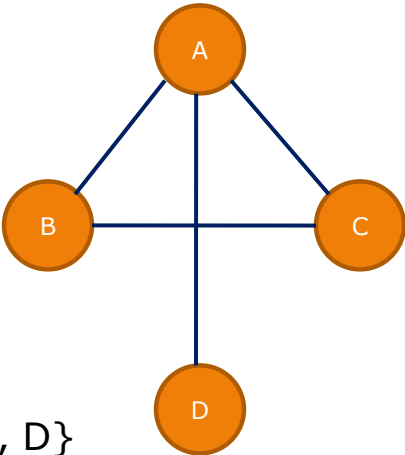


## Definition

Peeking into Computer Science

© Jalal Kawash 2010

14

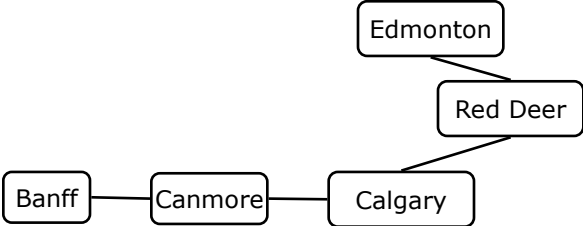


- $V = \{A, B, C, D\}$
- $E = \{\{A,B\}, \{A,C\}, \{A,D\}, \{B,C\}\}$

## An undirected graph

*Peeking into Computer Science* © Jalal Kawash 2010 15

- Edges are two-way
  - Alberta towns, cities and the highways that connect them.

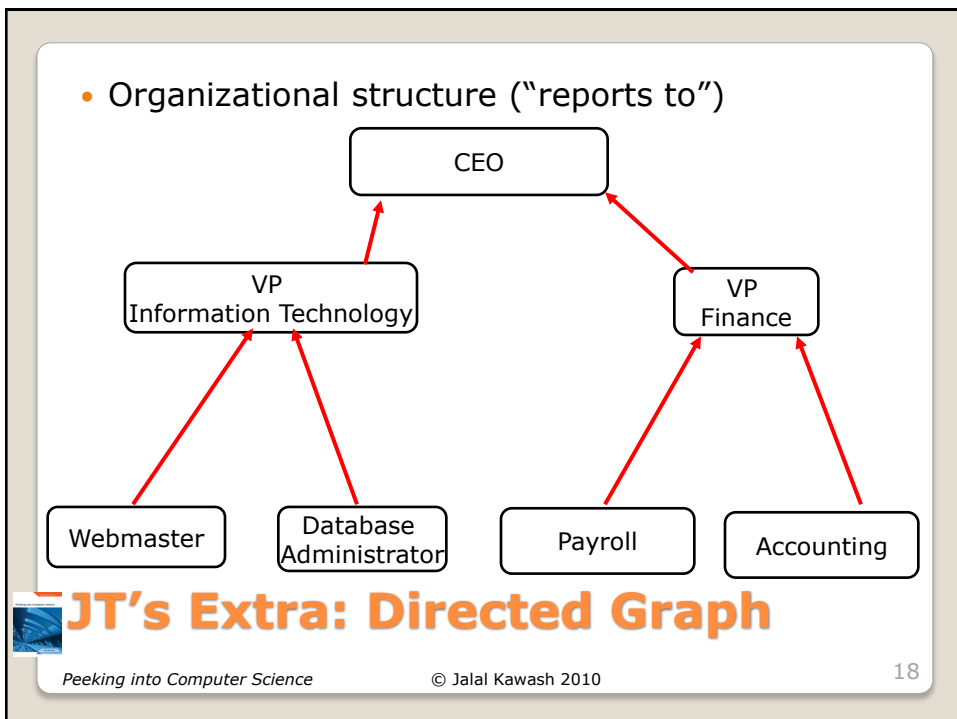
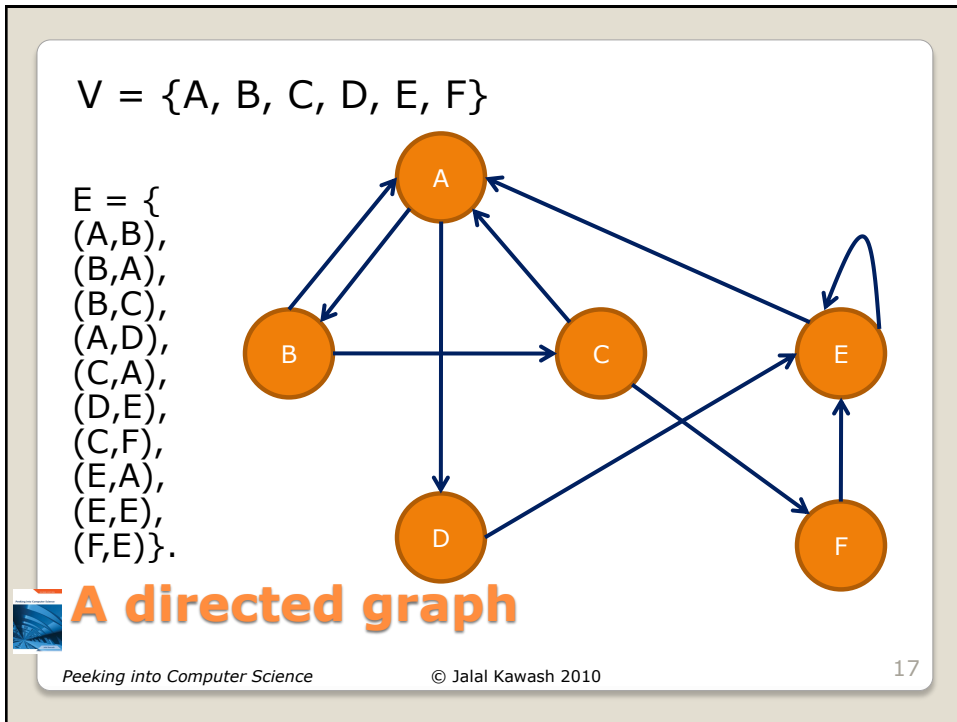


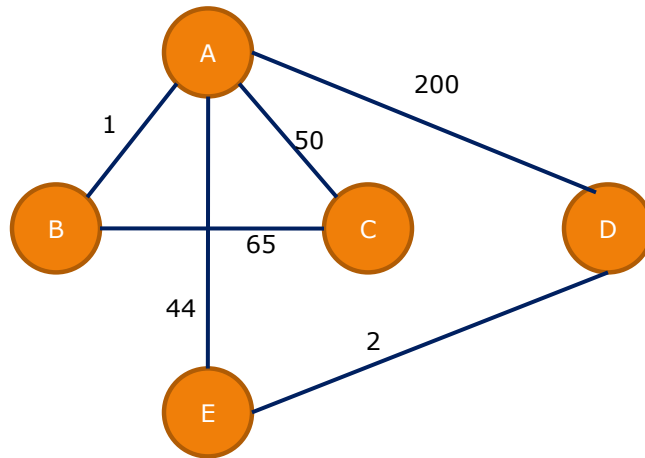
$V = \{\text{Edmonton, Red Deer, Calgary, Canmore, Banff}\}$   
 $E = \{\{\text{Banff, Canmore}\},$   
 $\{\text{Calgary, Canmore}\},$   
 $\{\text{Calgary, Red Deer}\},$   
 $\{\text{Edmonton, Red Deer}\}\}$

## JT' Extra: Undirected Graphs

*Peeking into Computer Science* © Jalal Kawash 2010 16







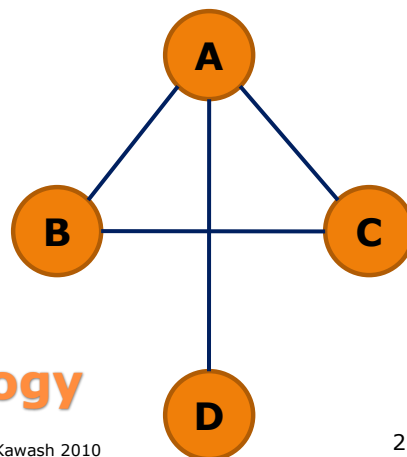
## A labeled undirected graph

Peeking into Computer Science

© Jalal Kawash 2010

19

- **Adjacent** vertices: connected by an edge
- A and B are adjacent
- D and C are not



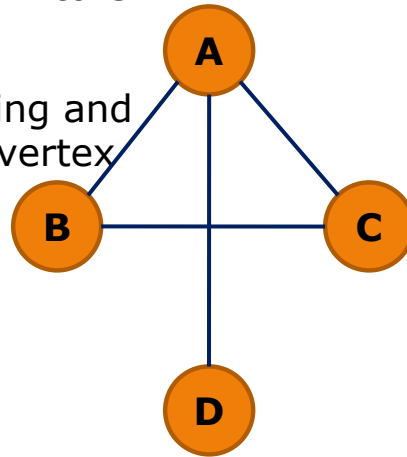
## Graph Terminology

Peeking into Computer Science

© Jalal Kawash 2010

20

- There is a **Path** from A to C
  - One path: A to B to C
- **Cycle** = a path starting and ending with the same vertex



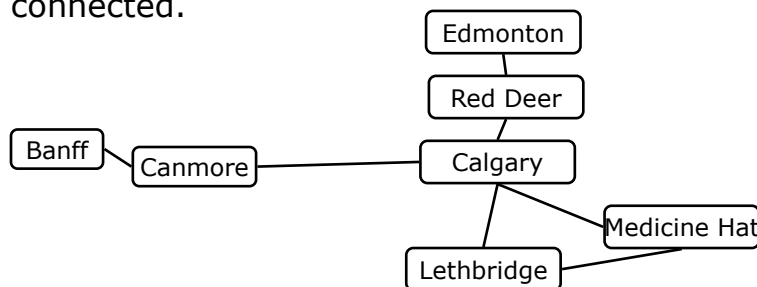
## Graph Terminology

Peeking into Computer Science

© Jalal Kawash 2010

21

- There is a path between vertices if they are: 1) directly connected by an edge or 2) indirectly connected.



- Red Deer and Lethbridge are not adjacent but there is at least one path from Red Deer to Lethbridge.

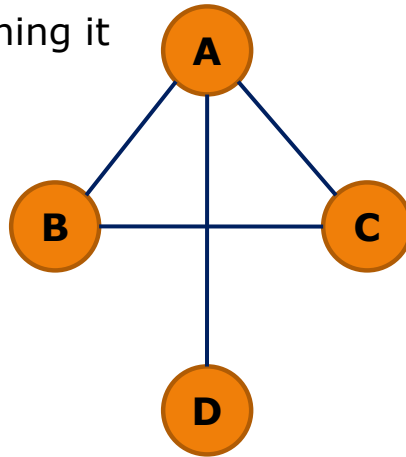
## JT's Extra: Path Example

Peeking into Computer Science

© Jalal Kawash 2010

22

- **Degree** of a vertex =  
Number of edges touching it  
degree of C = 2  
degree of A = 3



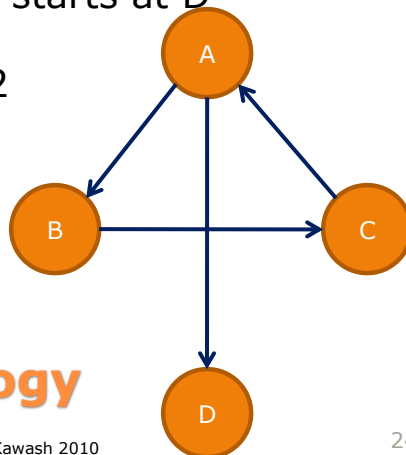
## Graph Terminology

Peeking into Computer Science

© Jalal Kawash 2010

23

- A is adjacent **to** B
- B is adjacent **from** A
- There is a **Path** from A to C
- There is no path that starts at D
- **In-degree** of A is 1
- **Out-degree** of A is 2



## Graph Terminology

Peeking into Computer Science

© Jalal Kawash 2010

24

- The World Wide Web itself can be visualized as a directed graph.
  - Vertex = web page, Edge = link between pages.



## The WWW Is A Graph

Peeking into Computer Science

© Jalal Kawash 2010

26

The screenshot shows a web browser window displaying the faculty home page of James Tam. The page has a navigation bar with links for Home, Teaching, Research, and About me. Below the navigation bar, there is a section for contact information, a photo, and a section for current teaching. Red arrows point from the navigation links to the corresponding sections on the page, illustrating the structure of the web page as a graph.

**The faculty home page of James Tam**

Navigation links: Home, Teaching, Research, About me

Contact information:

- Office:JCT 707
- Email: tam@cpsc.ucalgary.ca
- Phone:(403) 210-9455

My schedule

Current teaching (Fall 2007 - Summer 2008)

Fall 2009

- CPSC 203 (University of Calgary, Faculty of Nursing Qatar - go to Blackboard for course information)

Winter 2010

- CPSC 212
- CPSC 231

Spring 2010

- CPSC 203

Research work

1. Tam, J. and Grendling, S. (2006) A Framework for Asynchronous Change Awareness in Collaborative Documents and Workspaces. *International Journal of Human Computer Studies*, 64(7), p583-591. Elsevier. This paper is an expansion of our earlier CHI2004 conference paper. [Abstract](#) [Paper](#)
2. Sherja, L. and Tam, J. (2005) Developing Character Personas and Scenarios: Vital Steps in Theoretical Performance-Based Usability Design. *Credibility and Cognition '05*. ACM (April 21-15, London, UK). [Abstract](#) [Paper](#)
3. Tam, J. and Grendling, S. (2004) A Framework for Asynchronous Change Awareness in Collaborative Co-Constructed Documents. *CHI2004: 2 International Workshop on Groupware, Lecture Notes in Computer Science (LNCS Number 3198)*, Springer Verlag (September 5-9, San Carlos, Costa Rica). [Abstract](#) [Paper](#) [Presentation](#)
4. Nagano, C. (2004) *Educating Engineers on Linked Lists*. An undergraduate research project. [Abstract](#) [Paper](#)
5. Sherja, L. (2004) *Weapons trainer for "Messaging in the Metaverse"*. The performance for an M.F.A. thesis. [http://www.metaverse.com/theses/links.html](#)
6. Luo, E. (2003) *Stencils: An undergraduate research project*. [Abstract](#) [Paper](#)
7. Fahn, S. (2003) *Geometry: Mapping Data in the Elementary World*. An undergraduate research project. [Abstract](#) [Paper](#)
8. Sarikaya, S. (2003) *Zhina: JDR*. An undergraduate research project. [Abstract](#) [Paper](#)
9. Tam, J. (2002) *Supporting Change Awareness in Visual Workspaces*. M.Sc. thesis, Department of Computer Science, University of Calgary, Alberta, February. [Abstract](#)
10. Tam, J., McCallister, L., Miano, F. and Grendling, S. (2000) *Change Awareness in Software Engineering Using Two Dimensional Graphical Design and Development Tools*. Report 2000-070, 20. Department of Computer Science, University of Calgary, Alberta, Canada, October. [Abstract](#) [Paper](#)



## The WWW Is A Graph (2)

Peeking into Computer Science

© Jalal Kawash 2010

27

- Visualizing the layout of a page as a graph can be useful in web design.
  - Are there sufficient connections between pages?
  - Do the connections (links) make sense to visitor?
  - Although it should be possible to reach any page without too much clicking (rule of thumb is 3), there are some pages that should always be accessible e.g., home page, contact page etc.

## JT's Extra: Applying Graphs To Web Design

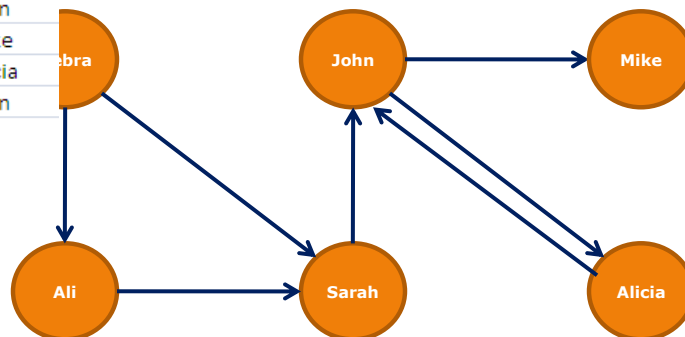


Peeking into Computer Science

© Jalal Kawash 2010

28

	A	B
1	Debra	Ali
2	Debra	Sarah
3	Ali	Sarah
4	Sarah	John
5	John	Mike
6	John	Alicia
7	Alicia	John



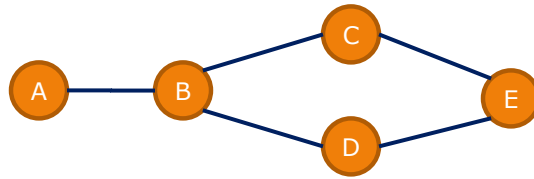
## Graph Representation



Peeking into Computer Science

© Jalal Kawash 2010

29



	A	B	C	D	E
A	0	1	0	0	0
B	1	0	1	1	0
C	0	1	0	0	1
D	0	1	0	0	1
E	0	0	1	1	0



## Graph Representation

Peeking into Computer Science

© Jalal Kawash 2010

30

- Graphs Up To This Point:
  - Do not allow for multiple edges between any pair of vertices.
- Multi-graphs are a type of graph that allows for these cases.

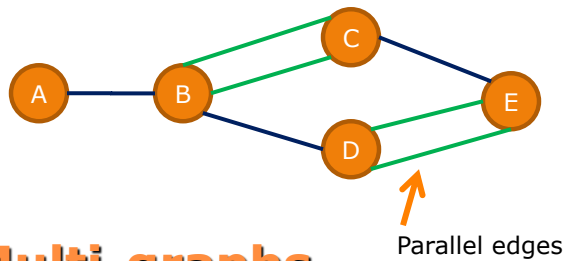


## JT's Extra: Repeated Connections

Peeking into Computer Science

© Jalal Kawash 2010

- In a multi-graph the set of edges is a multi-set (JT: recall a multi-set is a set that allows for duplications)
- Edges can be repeated
- Graphs are special cases of multi-graphs



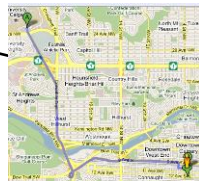
## Multi-graphs

Peeking into Computer Science

© Jalal Kawash 2010

32

- Example: path finding when alternatives are possible.




## JT's Extra: Application Multi-graphs

Peeking into Computer Science

© Jalal Kawash 2010

33





## Euler Paths

34

At the end of this section, you will be able to:

1. Understand how graphs are used to represent data
2. Appreciate the ability of graphs to lead to generalized conclusions
3. Define Euler tours and paths
4. Identify under which conditions an Euler circuit or path exists in a graph
5. Understand why such conditions are required
6. Learn the Euler circuit algorithm

## Objectives

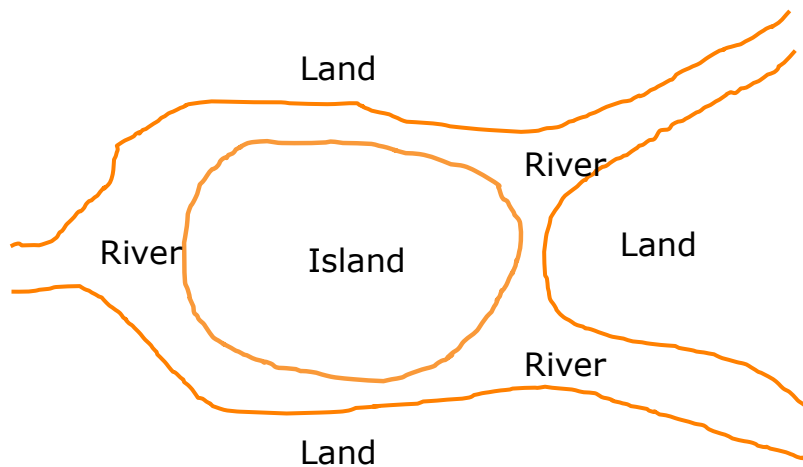
- Today called Kaliningrad in the Russian Republic
- In 1736 was in Prussia
- The town had seven bridges on the Pregel River

## **Konigsberg**

Peeking into Computer Science

© Jalal Kawash 2010

36

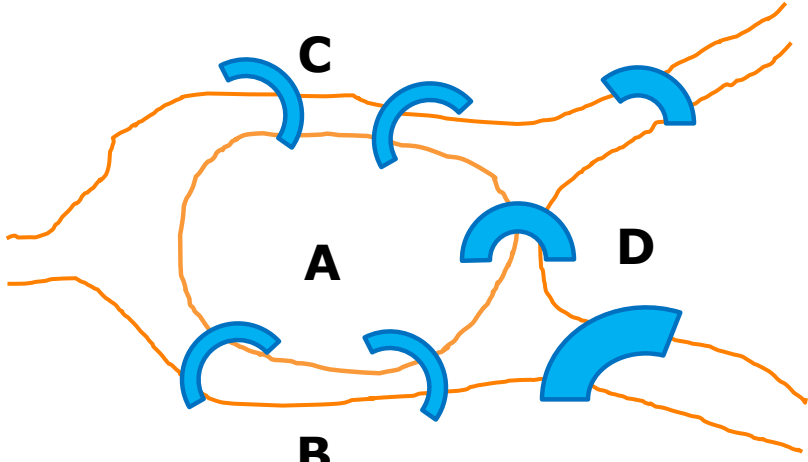


## **Konigsberg Layout**

Peeking into Computer Science

© Jalal Kawash 2010

37



**Seven Bridges**

*Peeking into Computer Science* © Jalal Kawash 2010 38

- People in 1736 wondered if it is possible to take the following walk in town:
- Start at some location in town
- Travel across all bridges without crossing any bridge twice
- End the walk where it started
- They wrote to Swiss Mathematician Leonhard Euler for help

**Konigsberg's Walk**

*Peeking into Computer Science* © Jalal Kawash 2010 39

- 1707 –1783
- The greatest mathematician of the 18th century and one of the greatest of all time
- a pioneering Swiss mathematician and physicist
- important discoveries in graph theory
- introduced much of the modern mathematical terminology and notation

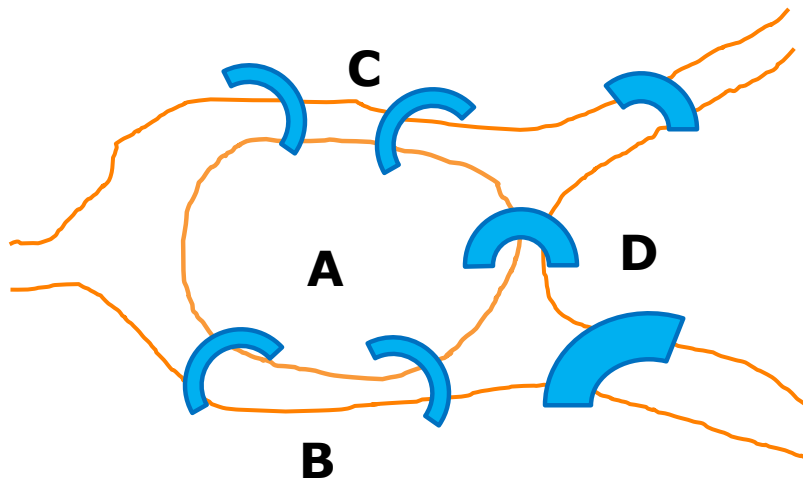


## Leonhard Euler

Peeking into Computer Science

© Jalal Kawash 2010

40

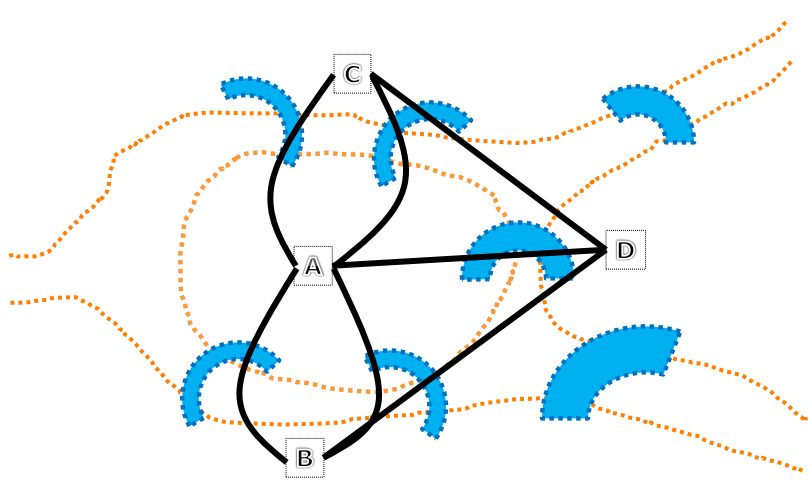


## Königsberg Multigraph

Peeking into Computer Science

© Jalal Kawash 2010

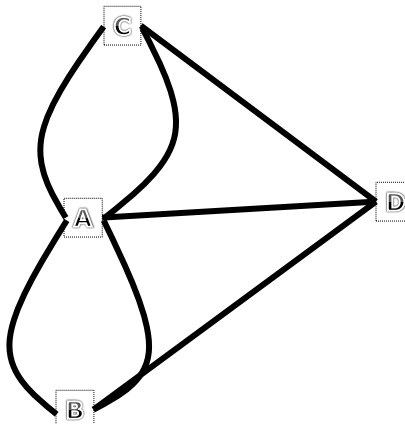
41



The diagram shows a Königsberg multigraph with four vertices labeled A, B, C, and D. Vertex A is in the center, B is below it, C is above it, and D is to the right. There are two curved edges between A and B, two between A and C, and two between A and D. Additionally, there are two curved edges between B and C. The graph is overlaid with several blue arcs and orange dotted lines that represent a path or a set of paths.

**Königsberg multigraph**

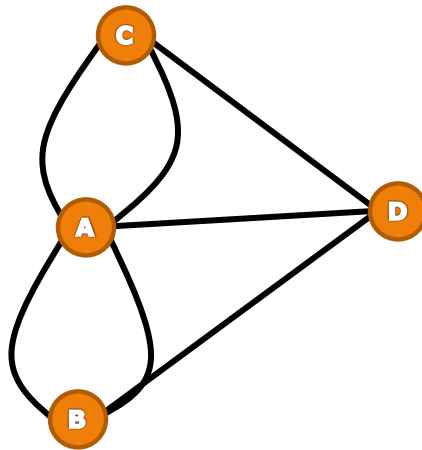
Peeking into Computer Science © Jalal Kawash 2010 42



The diagram shows the same Königsberg multigraph as in the previous slide, with four vertices labeled A, B, C, and D. There are two curved edges between A and B, two between A and C, and two between A and D. Additionally, there are two curved edges between B and C.

**Königsberg multigraph**

Peeking into Computer Science © Jalal Kawash 2010 43



## Konigsberg multigraph

Peeking into Computer Science

© Jalal Kawash 2010

44

- Is there a path through this graph that
  - Starts at a vertex
  - Ends in the same vertex
  - Passes through every edge once
  - Does not cross an edge more than once?
- Called an **Euler Circuit**
- Such a walk is impossible on any graph as long as the graph has one vertex with an *odd* degree



## Euler's Response

Peeking into Computer Science

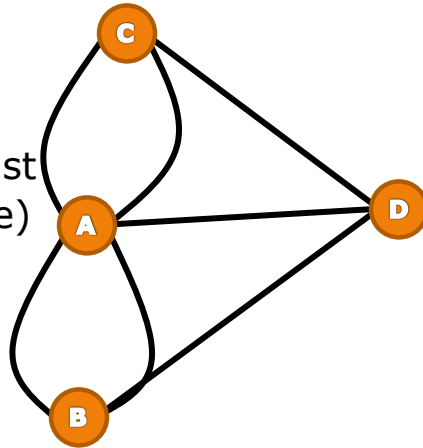
© Jalal Kawash 2010

45

- There is at least one vertex of an odd degree
- Does not work (the start vertex must have an even degree)



Success!



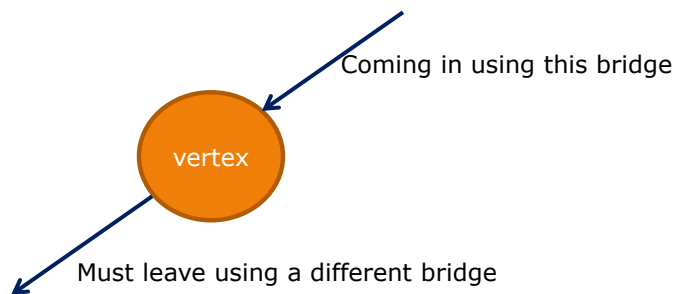
## Euler's Response

Peeking into Computer Science

© Jalal Kawash 2010

46

- To cross every bridge once



- Each vertex must have an even degree

## Even-Degree Vertex (JT: Not Start/End)

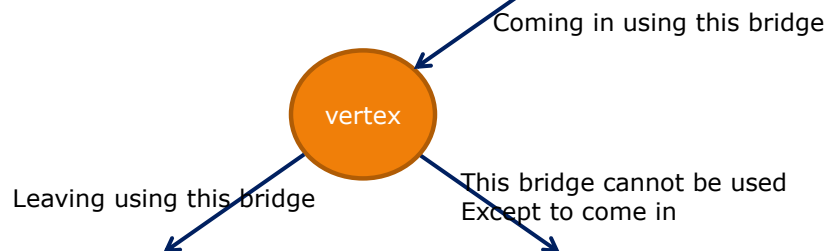


Peeking into Computer Science

© Jalal Kawash 2010

47

- Must come in using some bridge
- Must leave using a different bridge

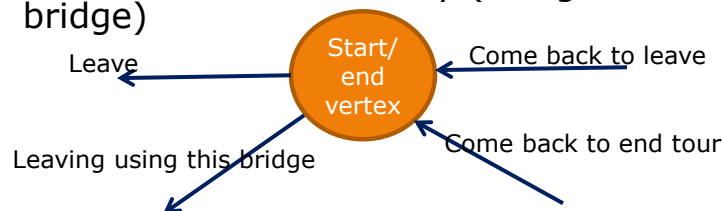


- We're stuck

## Odd-Degree Vertex (JT: Not Start/End)



- Must leave using some bridge
- Either we come back to leave again (need two new bridges)
- Or we come back to stay (using another bridge)



- Cannot be of an odd degree: we need another bridge to come back



## The Starting/Ending Vertex

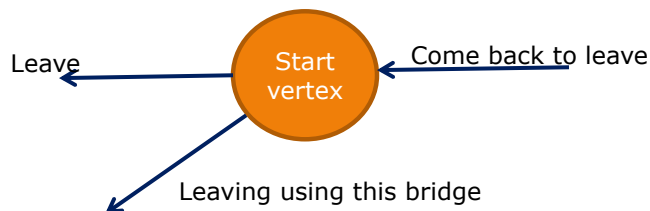


- Is there a path through this graph that
  - Starts at some vertex
  - Ends at a (possibly **different**) vertex
  - Passes through every edge once
  - Does not cross an edge more than once?
- Called a **Euler Path**
- Note that if there is a circuit, then there is a path
- If not, then the path is necessarily not a circuit



## An Easier Problem

- The starting vertex must have an odd degree

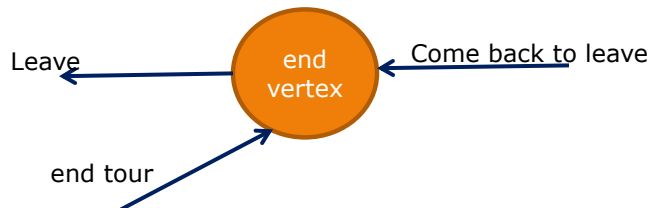


- **Never** come back to stay



## Euler Path – Start Vertex

- The end vertex must have an odd degree



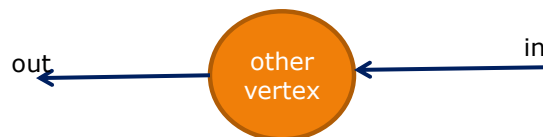
## Euler Path – End Vertex

Peeking into Computer Science

© Jalal Kawash 2010

52

- All other vertices must have an even degree



- Every **in** must have a matching **out** on "new" bridges

## Other Vertices

Peeking into Computer Science

© Jalal Kawash 2010

53

- Start and end vertices must have odd degrees
- Every other vertex must have an even degree
- There is no Euler Path for the Königsberg graph

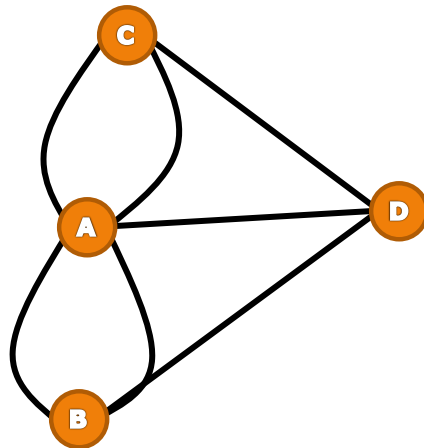
## Euler (non-cycle) Path Requirements



Peeking into Computer Science

© Jalal Kawash 2010

54



## No Euler Path for this Graph

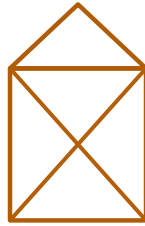


Peeking into Computer Science

© Jalal Kawash 2010

55

- Can you draw this shape with the rules:
  - Draw **continuously**, cannot lift pen from one position to another
  - Draw each line **once**, cannot let pen run on top of an already drawn line

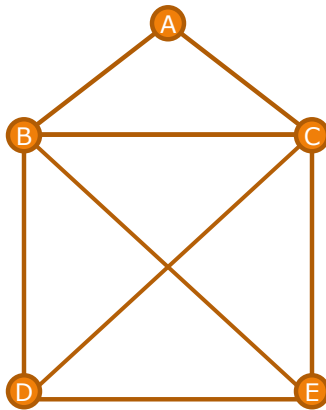


## A Similar Problem

Peeking into Computer Science

© Jalal Kawash 2010

56



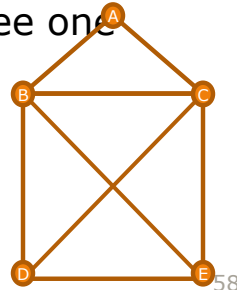
## Graph Representation

Peeking into Computer Science

© Jalal Kawash 2010

57

- Is there a Euler Circuit
  - No (some vertices have odd degrees)
- Is there a Euler Path
  - Yes (exactly two vertices have odd degrees)
- Path must start at an odd-degree vertex and ends at another odd-degree one



## Graph Representation

Peeking into Computer Science

© Jalal Kawash 2010

58

- Given a graph  $G$  (all vertices have even degrees):
1. Construct a circuit  $c$ , starting and ending at arbitrary vertex in  $G$
  2. Remove all the edges of  $c$  from  $G$
  3. Repeat until  $G$  has no edges:
    - a) Construct a circuit  $c'$  in  $G$  that starts (ends) in a vertex  $v$  that is in  $c$
    - b) Add  $c'$  to  $c$  at  $v$
    - c) Remove all the edges of  $c'$  from  $G$

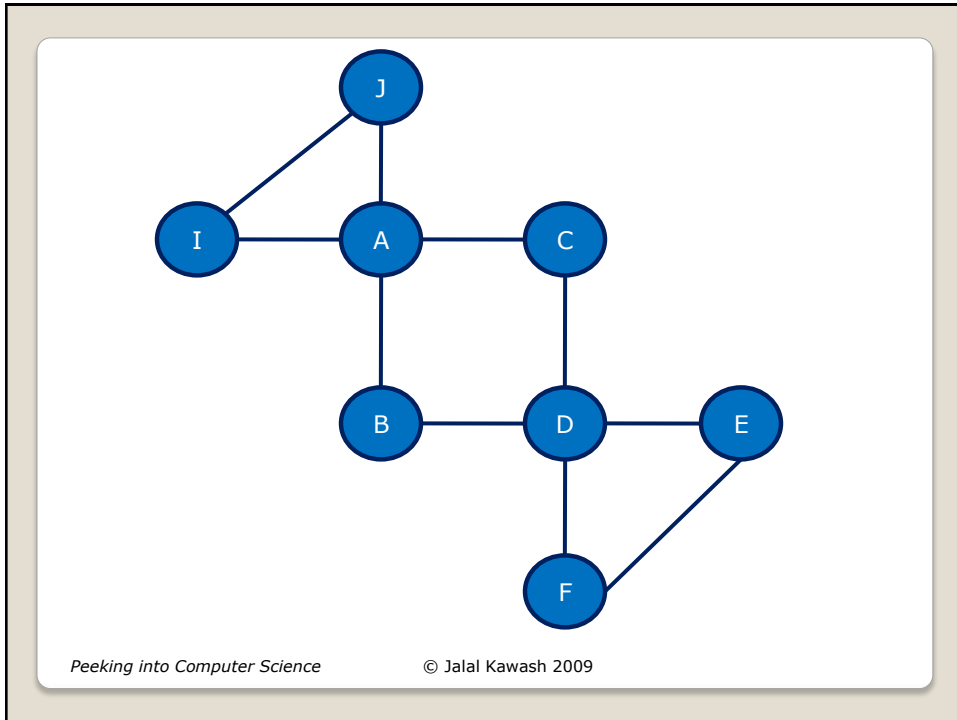


## Euler Circuit Algorithm

Peeking into Computer Science

© Jalal Kawash 2010

59



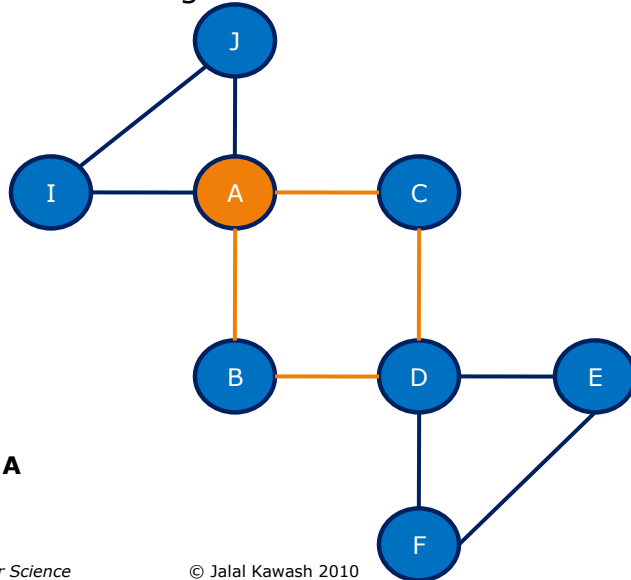
1. Construct a circuit  $c$ , starting and ending at arbitrary vertex in  $G$

Note: other choices for  $c$  are possible

**$c$ : A, B, D, C, A**

Peeking into Computer Science © Jalal Kawash 2010 61

2. Remove all the edges of  $c$  from  $G$



$c$ : A, B, D, C, A

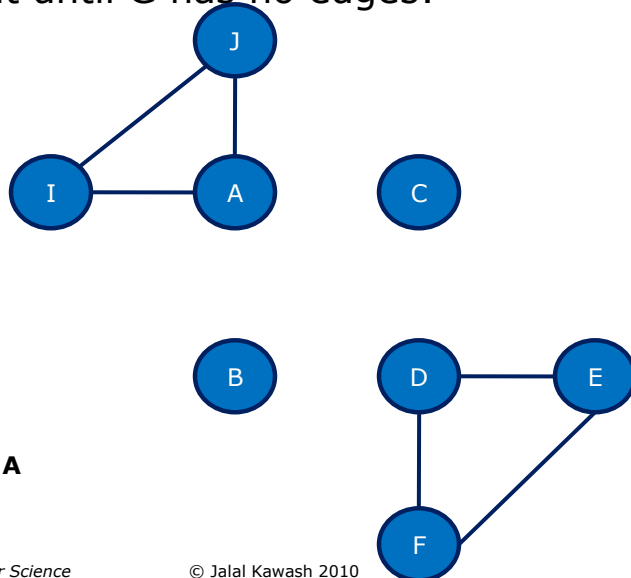


Peeking into Computer Science

© Jalal Kawash 2010

62

3. Repeat until  $G$  has no edges:



$c$ : A, B, D, C, A

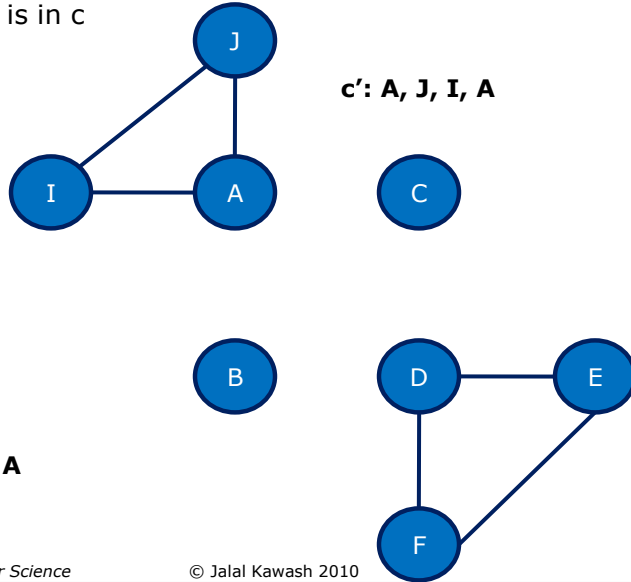


Peeking into Computer Science

© Jalal Kawash 2010

63

- a) Construct a circuit  $c'$  in  $G$  that starts (ends) in a vertex  $v$  that is in  $c$



$c$ : A, B, D, C, A

$c'$ : A, J, I, A

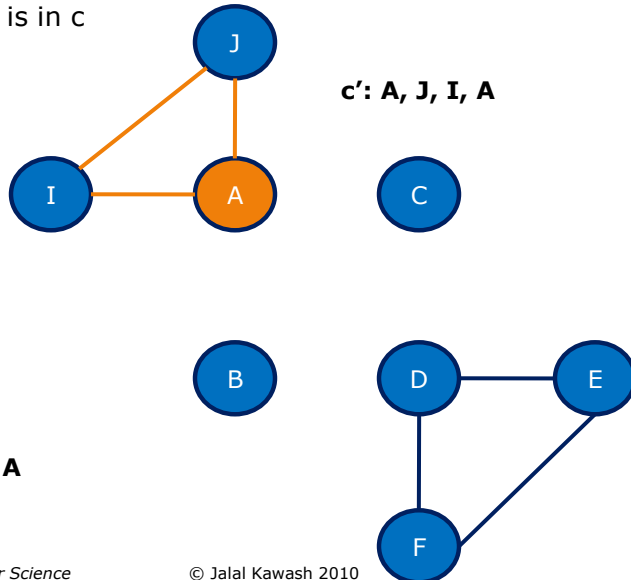


Peeking into Computer Science

© Jalal Kawash 2010

64

- a) Construct a circuit  $c'$  in  $G$  that starts (ends) in a vertex  $v$  that is in  $c$



$c$ : A, B, D, C, A

$c'$ : A, J, I, A



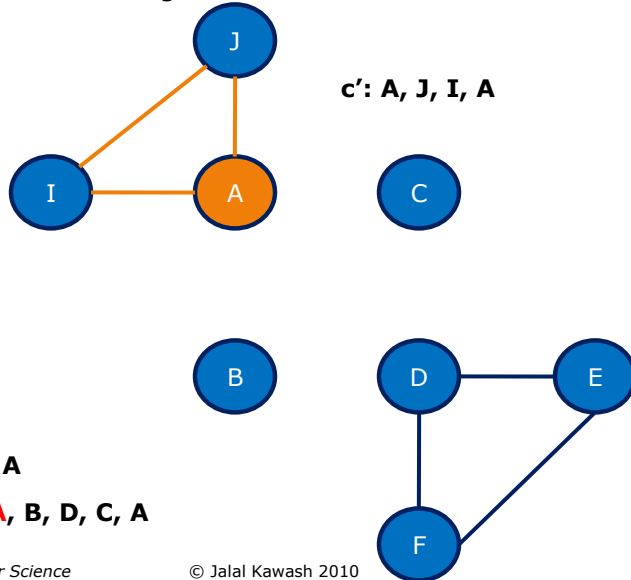
Peeking into Computer Science

© Jalal Kawash 2010

65



- a) Add  $c'$  to  $c$  at  $v$   
 b) Remove all the edges of  $c'$  from  $G$

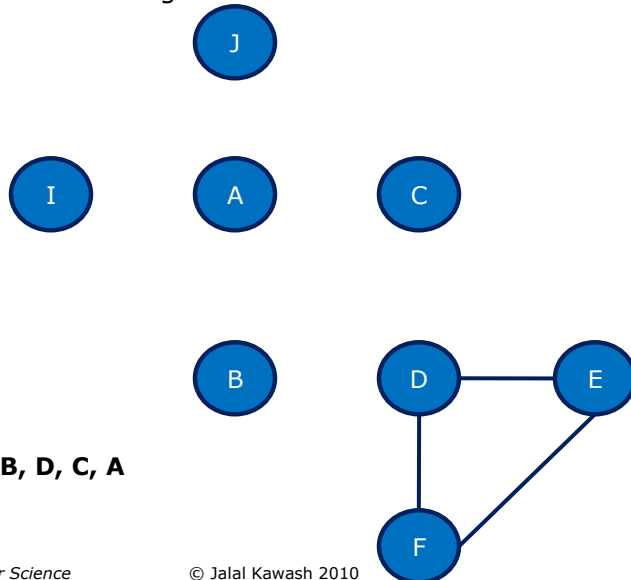


Peeking into Computer Science

© Jalal Kawash 2010

66

- a) Add  $c'$  to  $c$  at  $v$   
 b) Remove all the edges of  $c'$  from  $G$



Peeking into Computer Science

© Jalal Kawash 2010

67

Repeat a), b), and c)

**c': D, F, E, D**

**An Euler circuit**  
**c: A, J, I, A, B, D, F, E, D, C, A**

**c: A, J, I, A, B, D, C, A**

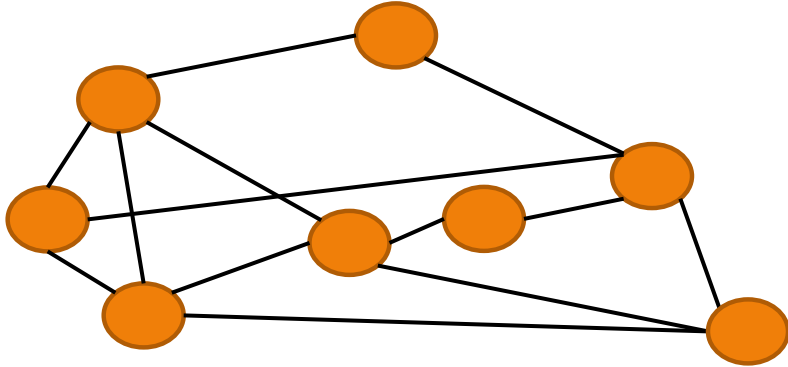
**c: A, J, I, A, B, D, C, A**

*Peeking into Computer Science* © Jalal Kawash 2010 68

The graph has no more edges, stop

**An Euler circuit**  
**c: A, J, I, A, B, D, F, E, D, C, A**

*Peeking into Computer Science* © Jalal Kawash 2010 69



**JT's Extra: Euler Cycle Or Path?**

*Peeking into Computer Science* © Jalal Kawash 2010 70