Introduction to Computer Science

CSCI 109

Readings

St. Amant, Ch. 4

Andrew Goodney

Fall 2017

"An **algorithm** (pronounced AL-go-rith-um) is a procedure or formula for solving a problem. The word derives from the name of the mathematician, Mohammed ibn-Musa al-Khwarizmi, who was part of the royal court in Baghdad and who lived from about 780 to 850."

Reminders

- HW 1 scores and the rubric will be posted by tomorrow. If you feel there has been a grading error, you must approach one of the TAs in their office hour <u>in person this week</u>. No grade disputes for HW1 will be entertained after the office hours for this week are done. No grade disputes via Piazza or email.
- HW2 has been posted on BB. It is due on 10/2.
- Quiz 1 is today (covers lecture material from 9/21 and 9/28)

Where are we?

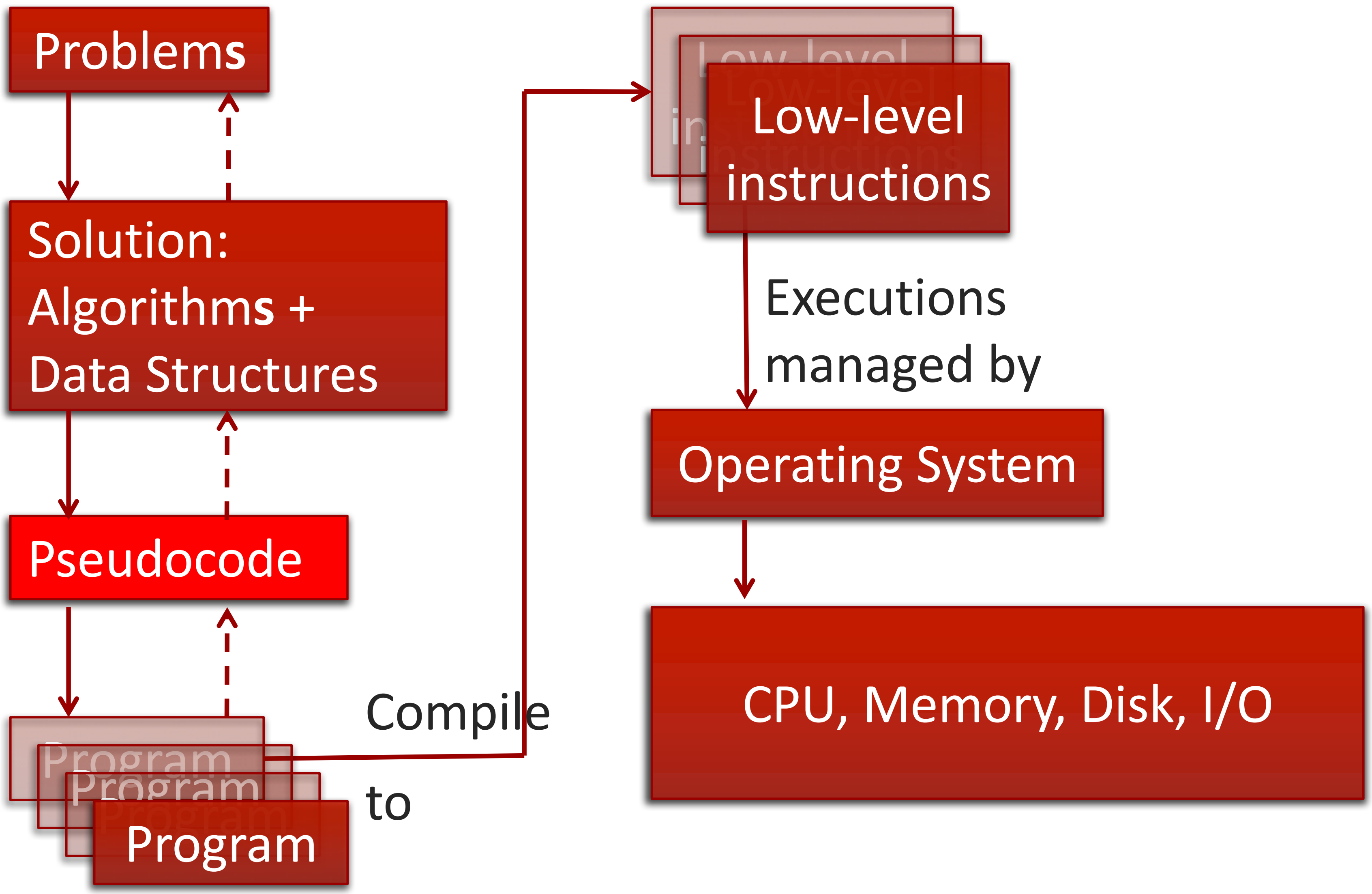
Date	Topic		Assigned	Due	Quizzes/Midterm/Final
21-Aug	Introduction	What is computing, how did computers come to be?			
28-Aug	Computer architecture	How is a modern computer built? Basic architecture and assembly	HW1		
4-Sep	Labor day				
11-Sep	Data structures	Why organize data? Basic structures for organizing data		HW1	
12-Sep	Last day to drop a Monday-only class without a mark of "W" and receive a refund or change to Pass/No Pass or Audit for Session 001				
18-Sep	Data structures	Trees, Graphs and Traversals	HW2		**Quiz 1** on material taught in class 8/21-8/28
25-Sep	More Algorithms/Data Structures	Recursion and run-time			
2-Oct	Complexity and combinatorics	How "long" does it take to run an algorithm.		HW2	**Quiz 2** on material taught in class 9/11-9/25
6-Oct	Last day to drop a course without a mark of "W" on the transcript				
9-Oct	Algorithms and programming	(Somewhat) More complicated algorithms and simple programming constructs			**Quiz 3** on material taught in class 10/2
16-Oct	Operating systems	What is an OS? Why do you need one?	HW3		**Quiz 4** on material taught in class 10/9
23-Oct	Midterm	Midterm			**Midterm** on all material taught so far.
30-Oct	Computer networks	How are networks organized? How is the Internet organized?		HW3	
6-Nov	Artificial intelligence	What is AI? Search, plannning and a quick introduction to machine learning			**Quiz 5** on material taught in class 10/30
10-Nov	Last day to drop a class with a mark of "W" for Session 001				
13-Nov	The limits of computation	What can (and can't) be computed?	HW4		**Quiz 6** on material taught in class 11/6
20-Nov	Robotics	Robotics: background and modern systems (e.g., self-driving cars)			**Quiz 7** on material taught in class 11/13
27-Nov	Summary, recap, review	Summary, recap, review for final		HW4	**Quiz 8** on material taught in class 11/20
8-Dec	Final exam 11 am - 1 pm in SAL 101				**Final** on all material covered in the semester

Data Structures and Algorithms

- A problem-solving view of computers and computing
- Organizing information: sequences and trees
- Organizing information: graphs
- Abstract data types: recursion

Reading:
St. Amant Ch. 4

Overview



Problems
Solution:
Algorithms +
Data Structures
Pseudocode
Program
Compile
to
Low-level
instructions
Executions
managed by
Operating System
CPU, Memory, Disk, I/O

Problem Solving

- Architecture puts the computer under the microscope
- Computers are used to solve problems
- Abstraction for problems
  - How to represent a problem ?
  - How to break down a problem into smaller parts ?
  - What does a solution look like ?
- Two key building blocks
  - Algorithms
  - Abstract data types

Algorithms

- Algorithm: a step by step description of actions to solve a problem
- Typically at an abstract level
- Analogy: clearly written recipe for preparing a meal

"Algorithms are models of procedures at an abstract level we decided is appropriate." [St. Amant, pp. 53]

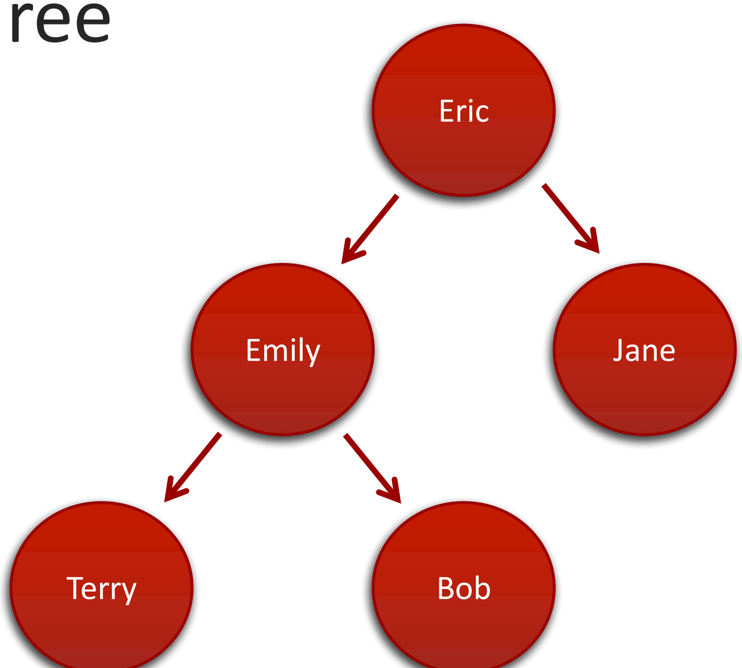
Abstract Data Types

- Models of collections of information
- Typically at an abstract level
- Analogy: spices in your kitchen

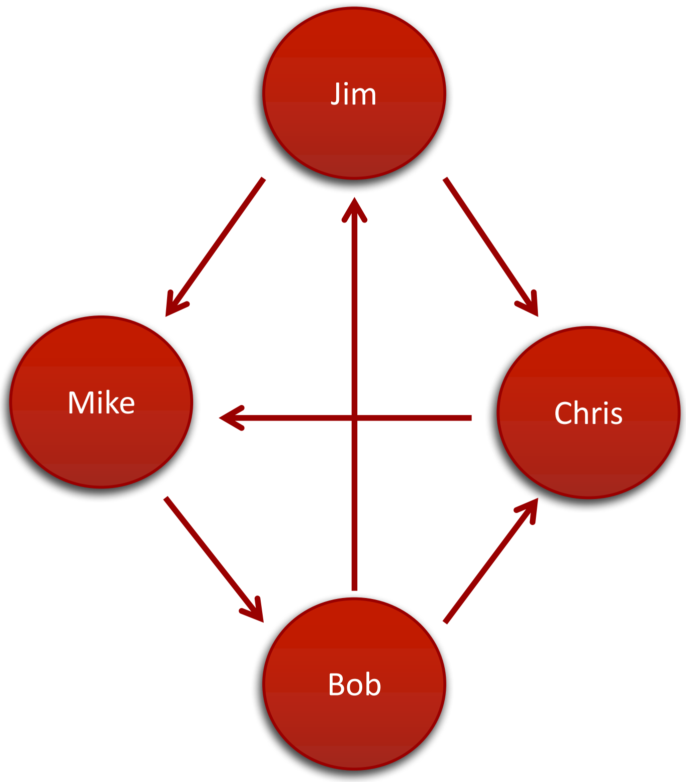
"… describes what can be done with a collection of information, without going down to the level of computer storage." [St. Amant, pp. 53]

Sequences, Trees and Graphs

- Sequence: a list
  - Items are called elements
  - Item number is called the index
- Tree

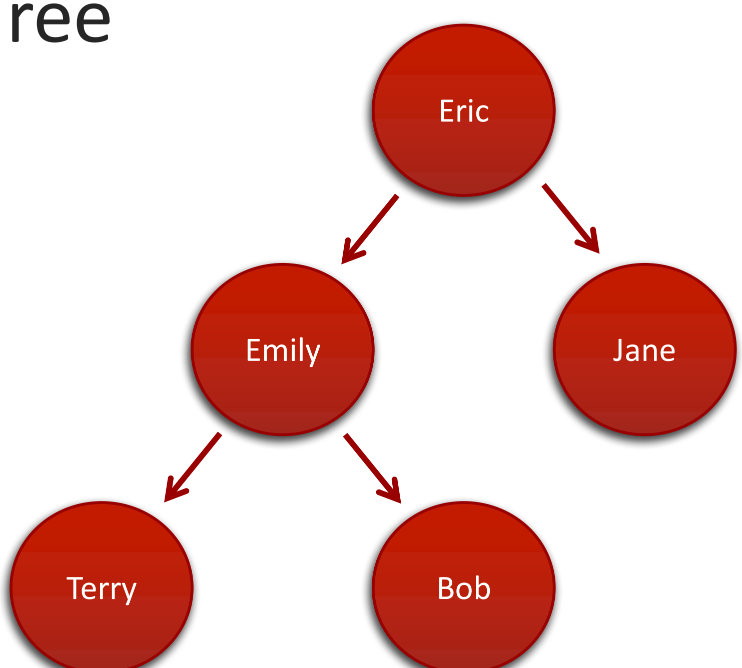


- Graph



Sequences, Trees and Graphs

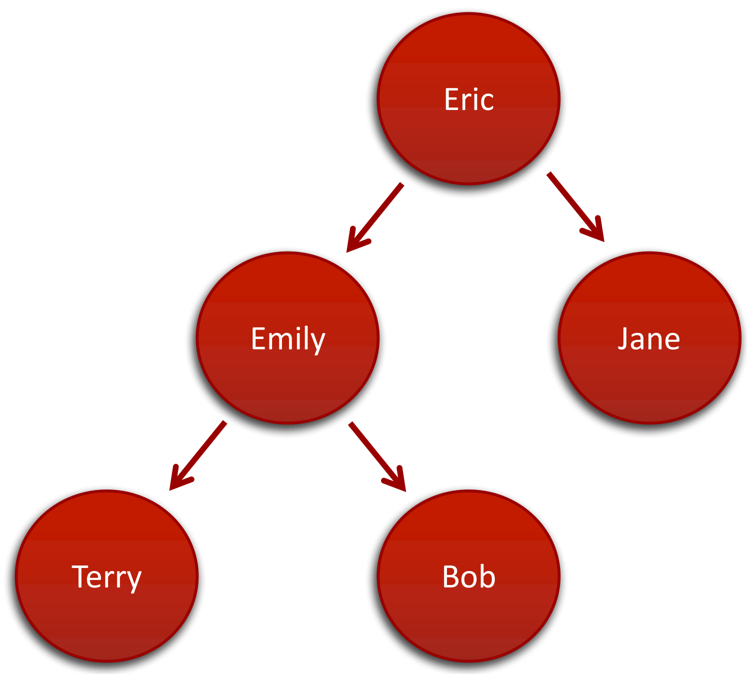
- Sequence: a list
  - Items are called elements
  - Item number is called the index
- Tree



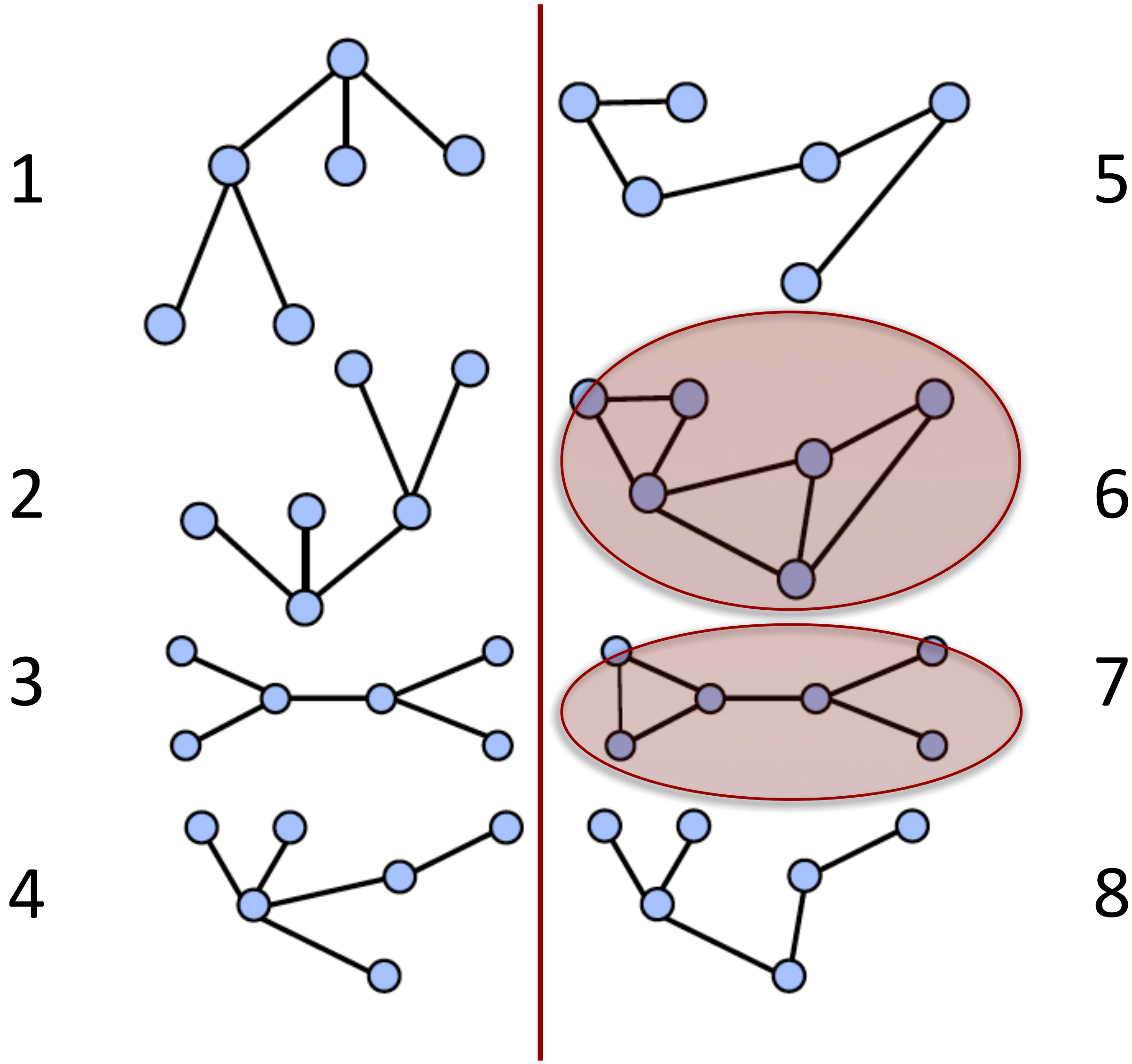
- Lists
  - Searching
    - Unsorted list
    - Sorted list
  - Sorting
    - Selection sort
    - Quicksort
- The notion of a brute force algorithm
- The divide and conquer strategy

Trees

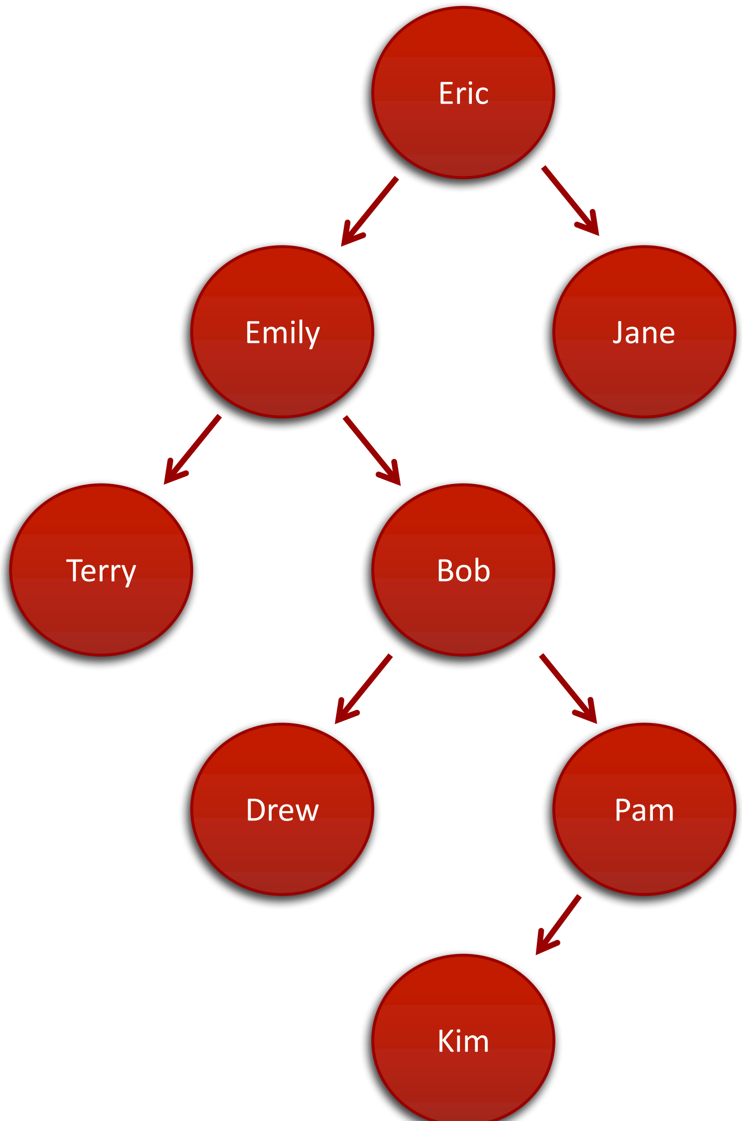
- Each node/vertex has exactly one parent node/vertex
- No loops
- Directed (links/edges point in a particular direction)
- Undirected (links/edges don't have a direction)
- Weighted (links/edges have weights)
- Unweighted (links/edges don't have weights)



Which of these are NOT trees?



Tree Traversal



- Depth first traversal

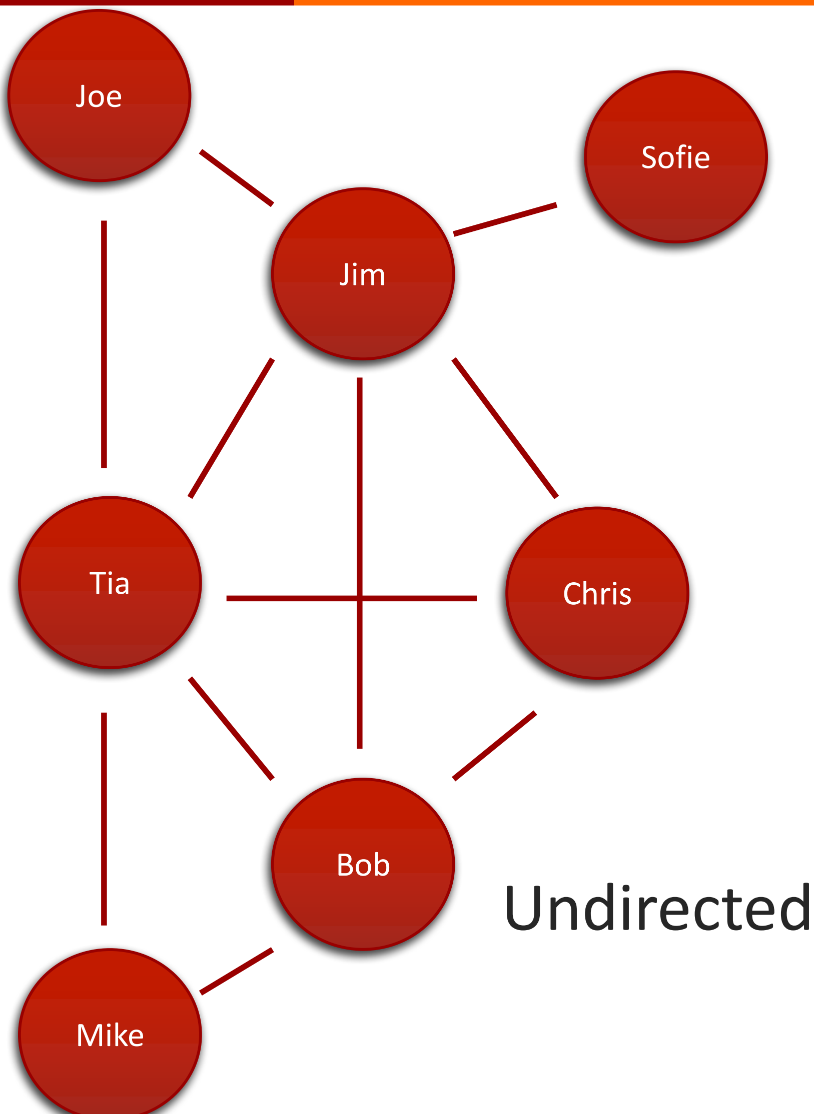
  Eric, Emily, Terry, Bob, Drew, Pam, Kim, Jane

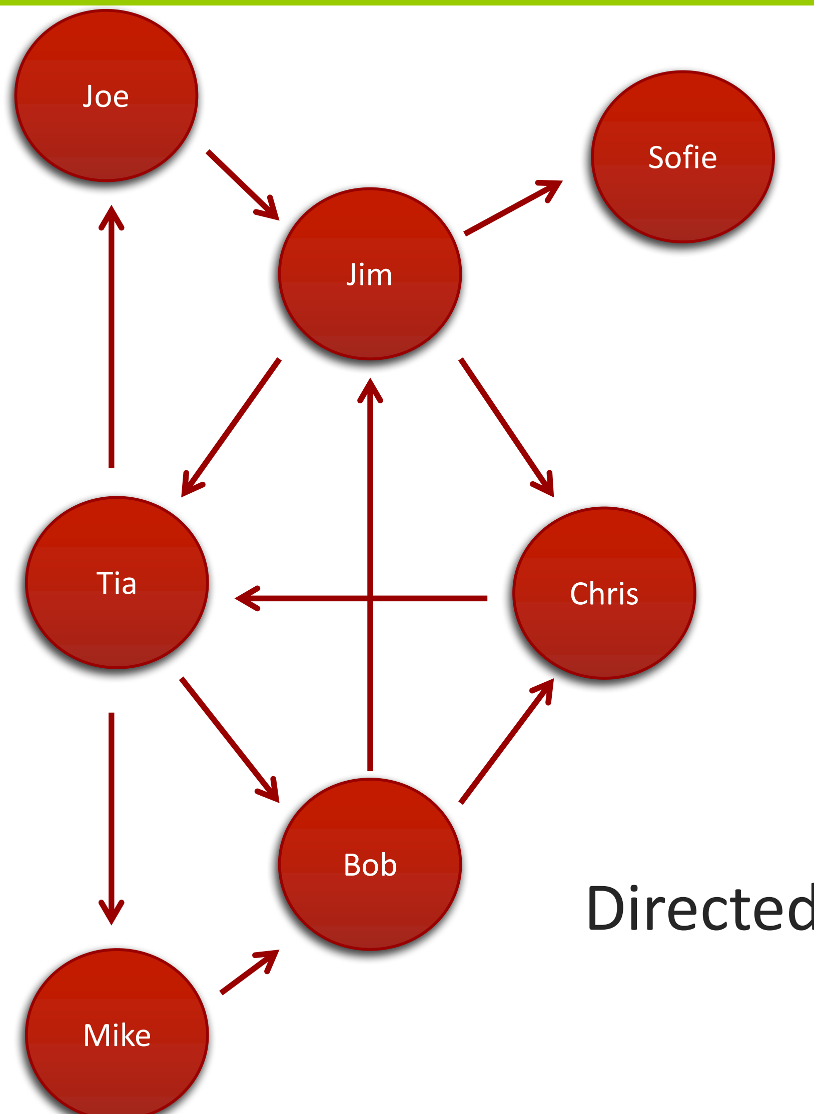
- Breadth first traversal

  Eric, Emily, Jane, Terry, Bob, Drew, Pam, Kim

  Eric, Jane, Emily, Bob, Terry, Pam, Drew, Kim

Graphs: Directed and Undirected

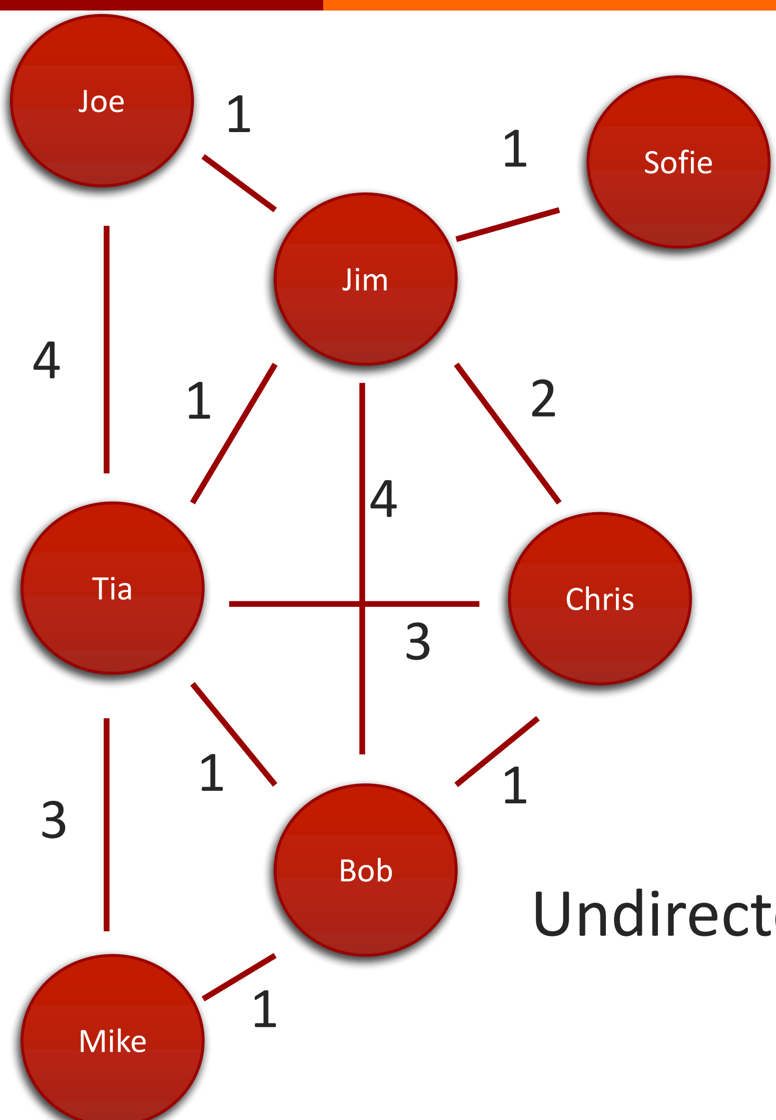




Spanning trees

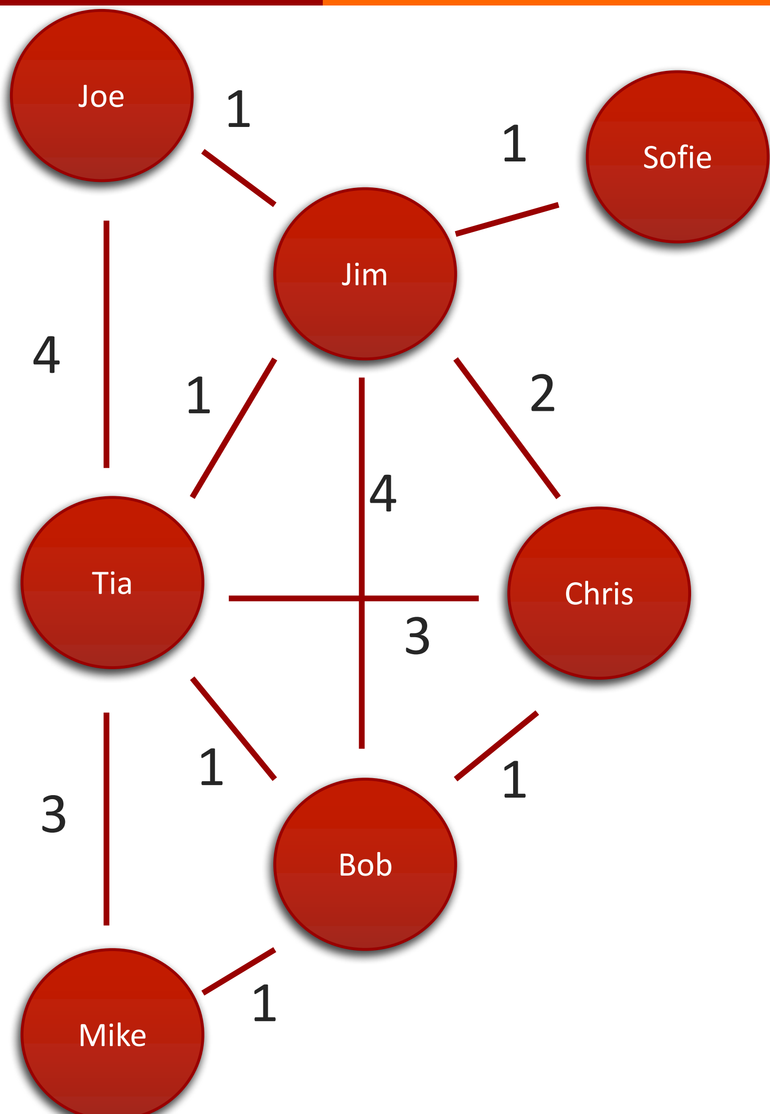
- Spanning tree: Any tree that covers all vertices
- *Minimum* spanning tree (MST): Tree of minimal total edge cost
- Why compute the minimum spanning tree?
  - Minimize the cost of connections between cities
  - Minimize of cost of wires in a layout

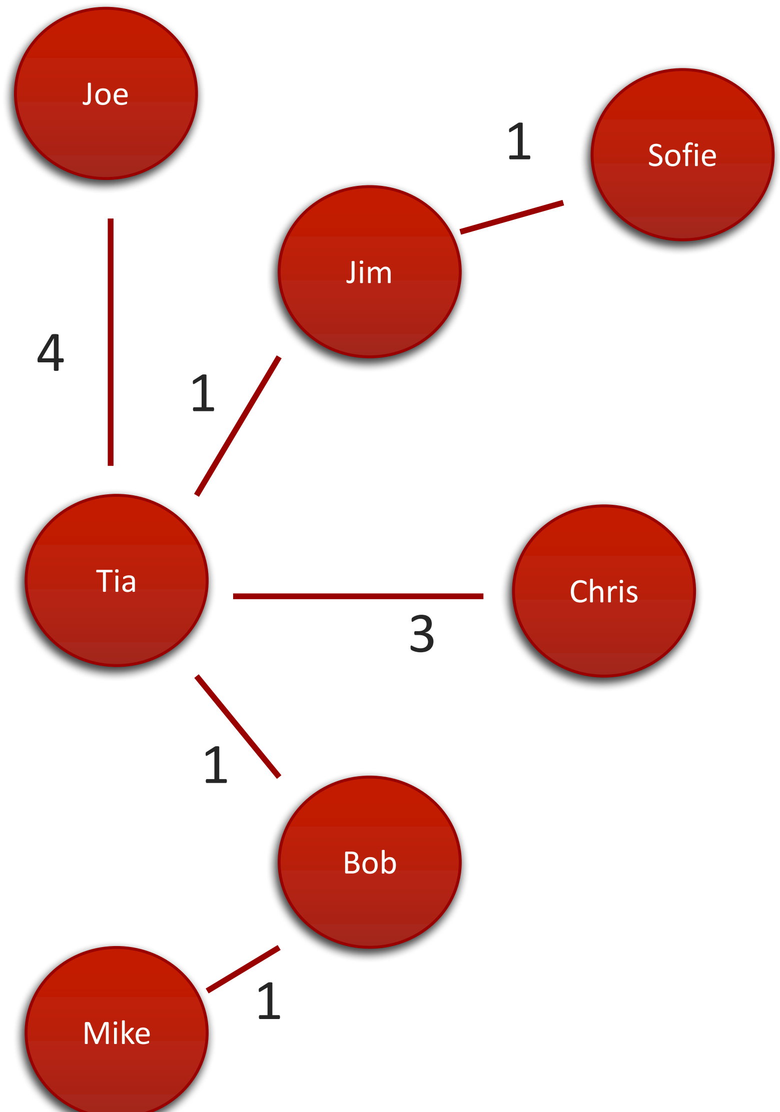
Edge costs, minimum spanning tree



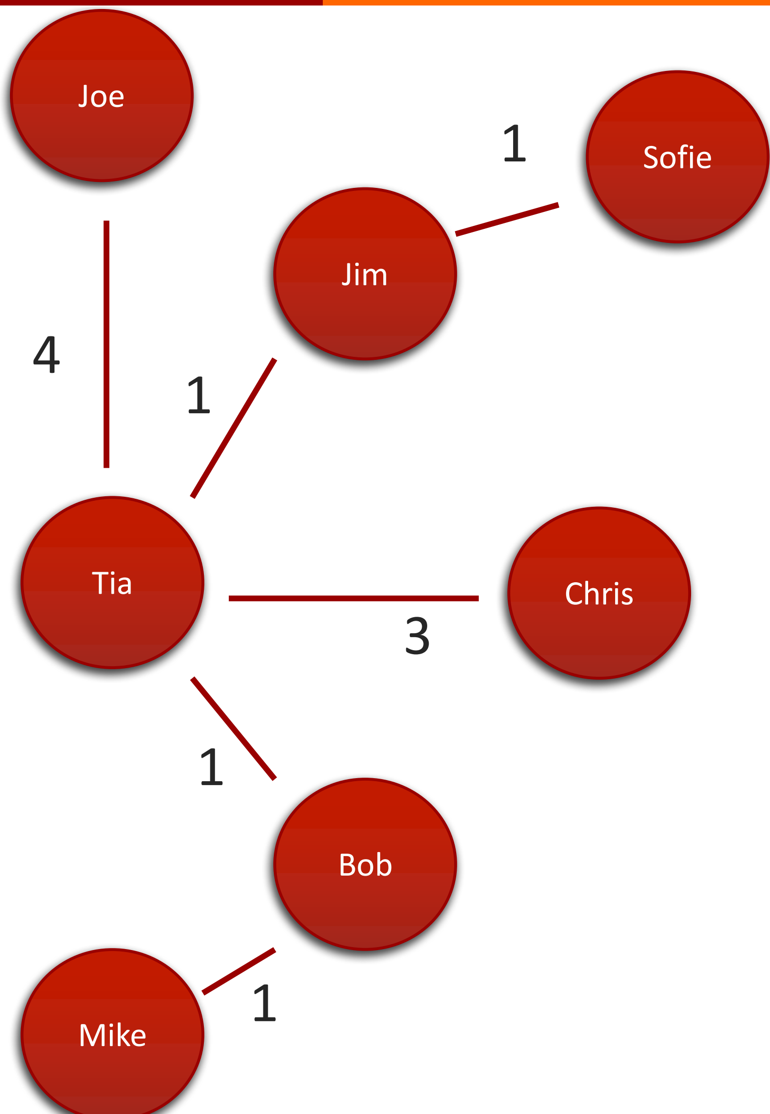
Undirected

Edge costs, minimum spanning tree

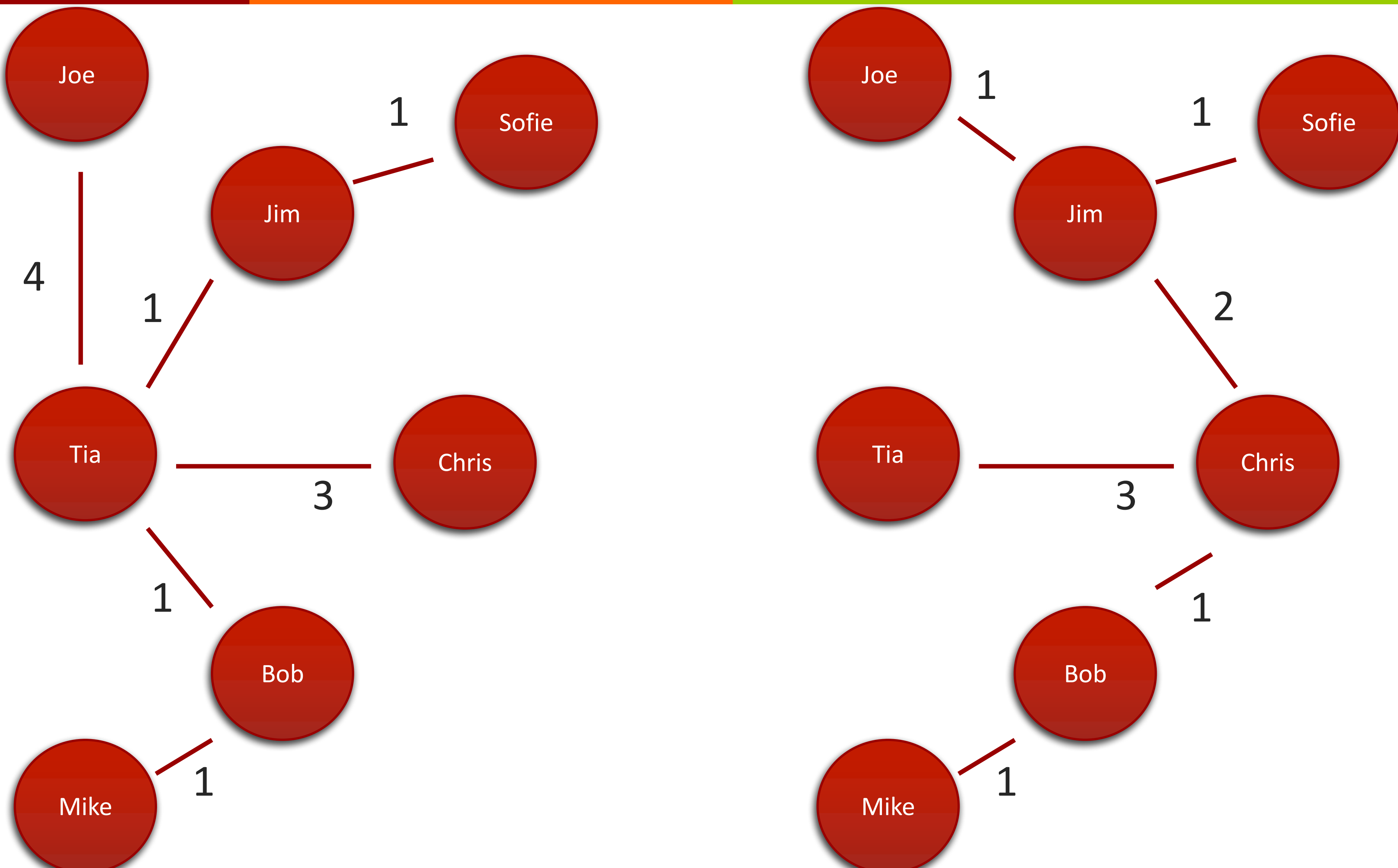




Spanning Trees



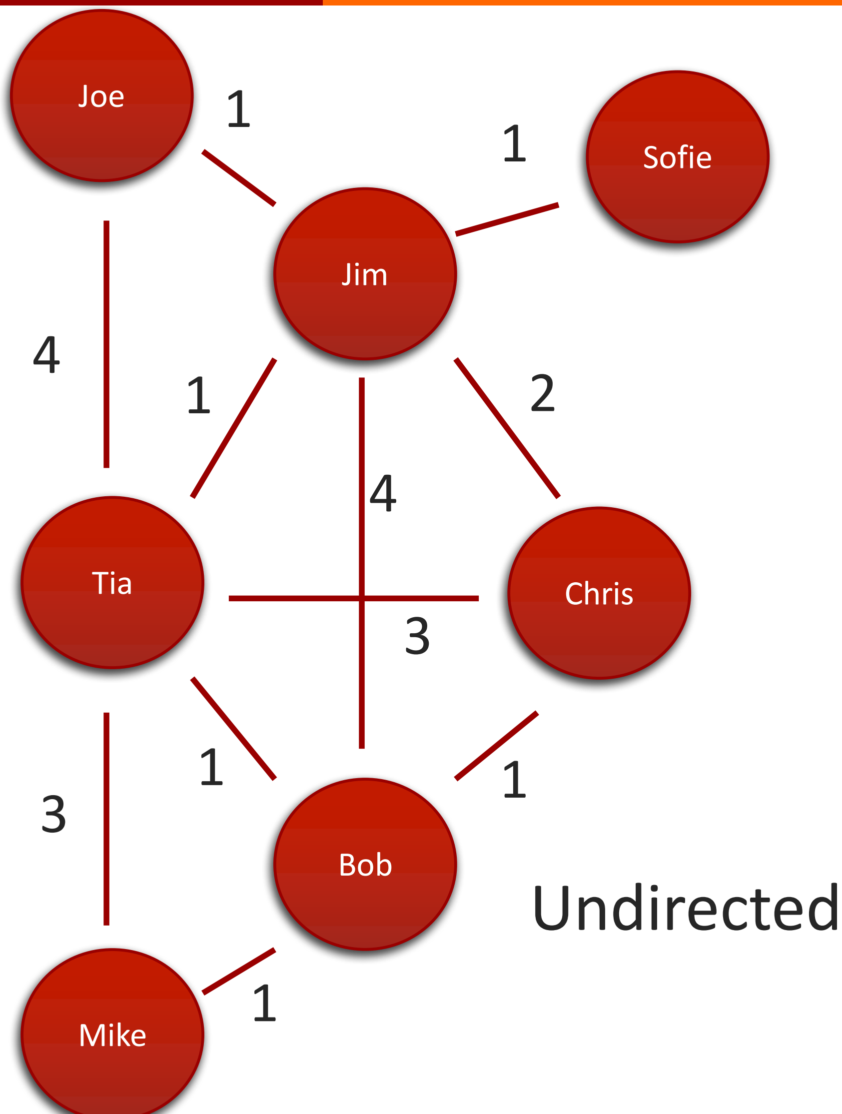
Spanning Trees



Computing the MST

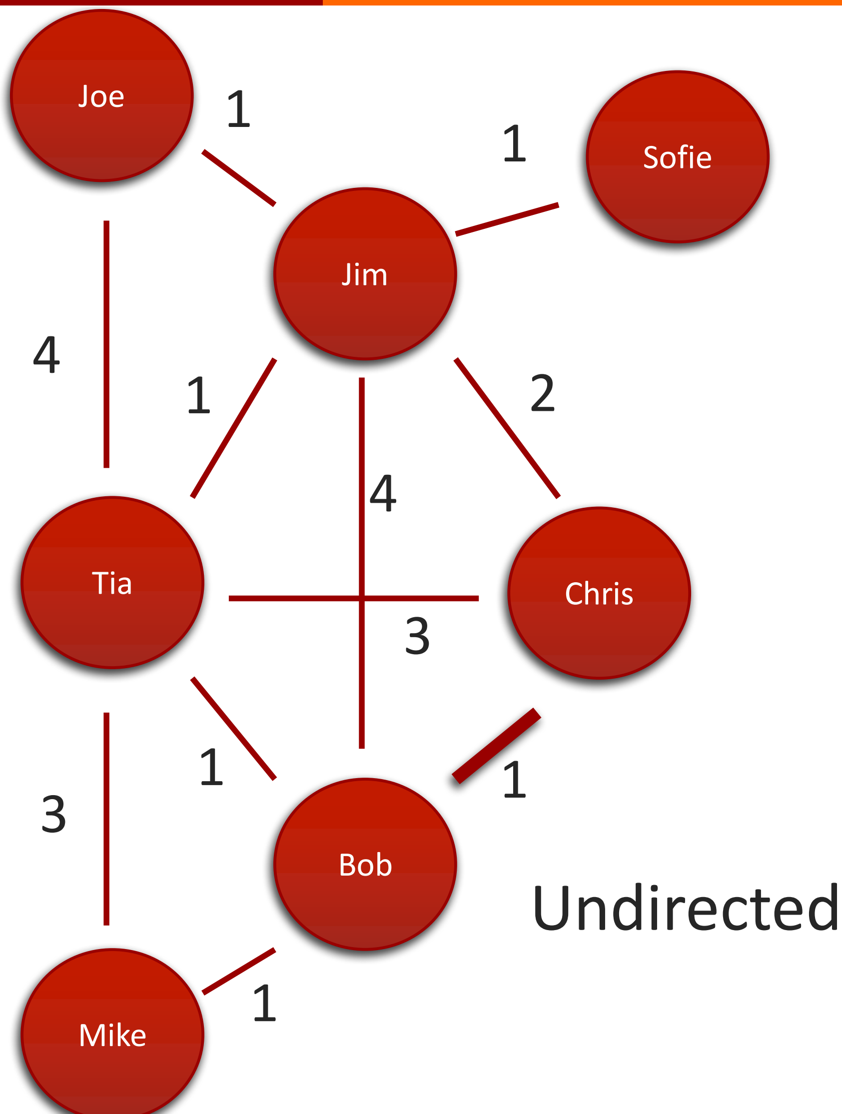
- Two greedy algorithms to compute the MST
  - Prim's algorithm: Start with any node and greedily grow the tree from there
  - Kruskal's algorithms: Order edges in ascending order of cost. Add next edge to the tree without creating a cycle.

Prim's algorithm



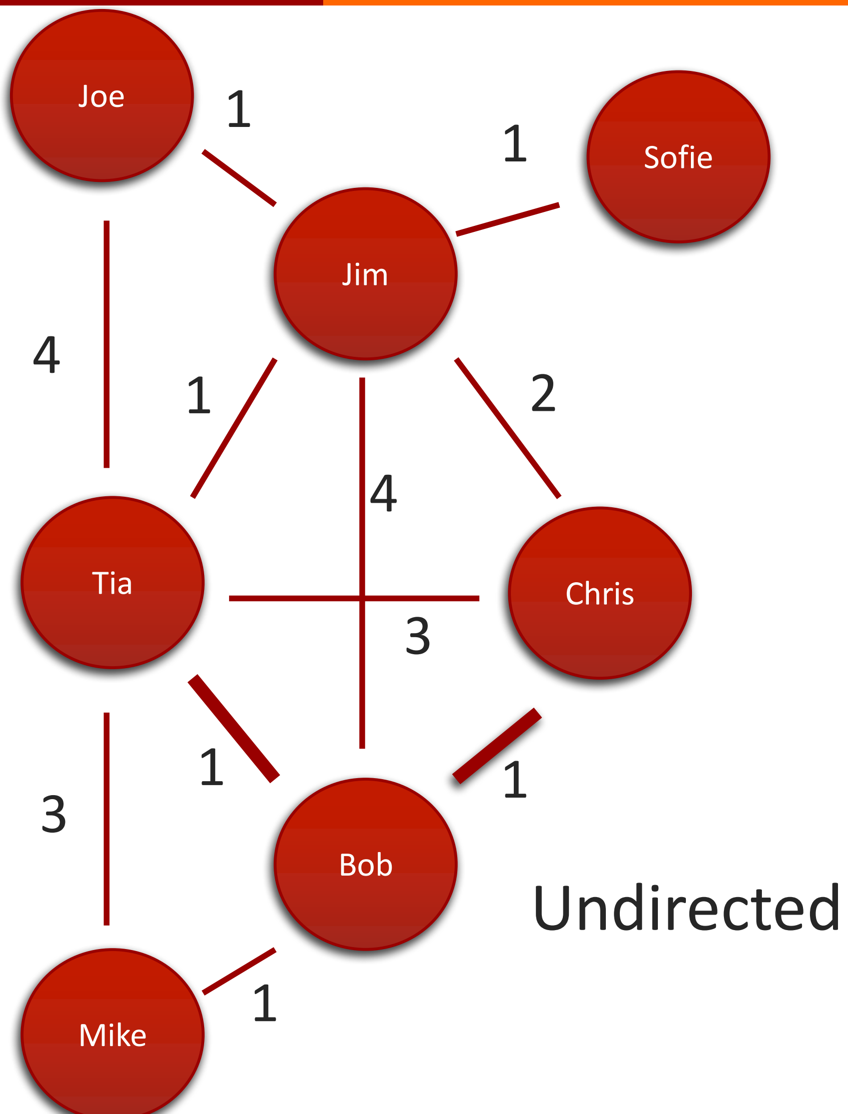
1	Joe-Jim
1	Jim-Sofie
1	Jim-Tia
1	Tia-Bob
1	Chris-Bob
1	Mike-Bob
2	Chris-Jim
3	Tia-Chris
3	Mike-Tia
4	Joe-Tia
4	Jim-Bob

Prim's algorithm



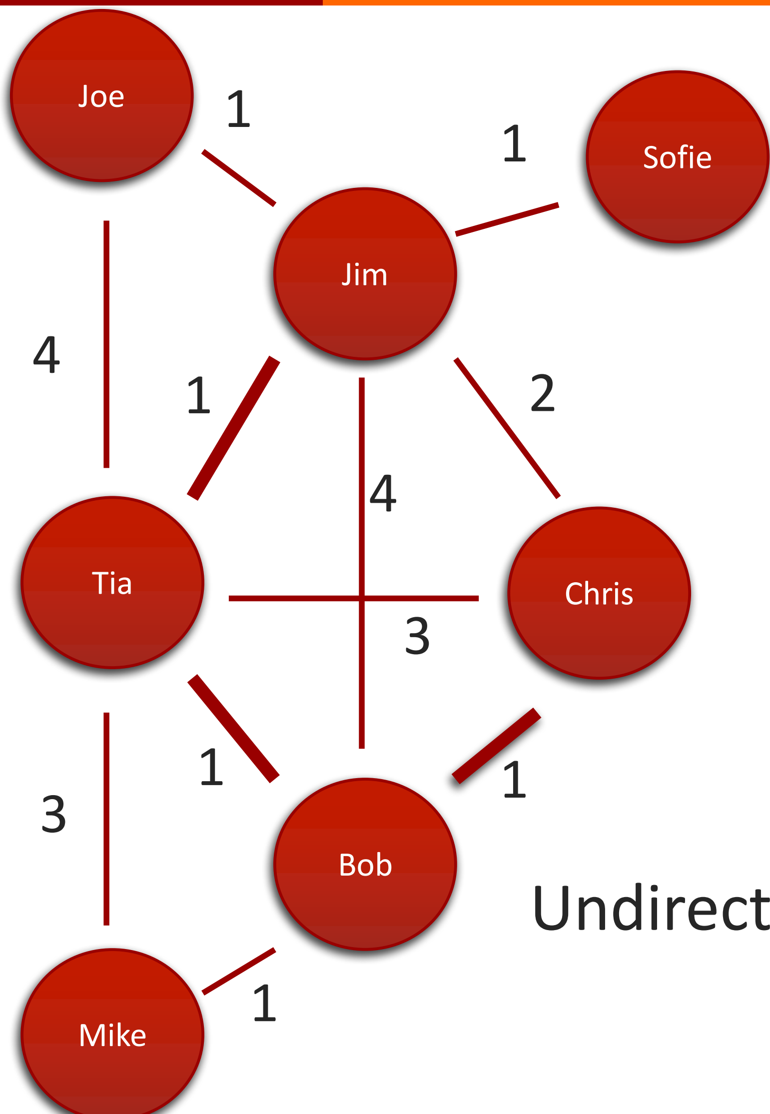
1	Joe-Jim
1	Jim-Sofie
1	Jim-Tia
1	Tia-Bob
1	**Chris-Bob**
1	Mike-Bob
2	Chris-Jim
3	Tia-Chris
3	Mike-Tia
4	Joe-Tia
4	Jim-Bob

Prim's algorithm



1	Joe-Jim
1	Jim-Sofie
1	Jim-Tia
1	**Tia-Bob**
1	**Chris-Bob**
1	Mike-Bob
2	Chris-Jim
3	Tia-Chris
3	Mike-Tia
4	Joe-Tia
4	Jim-Bob

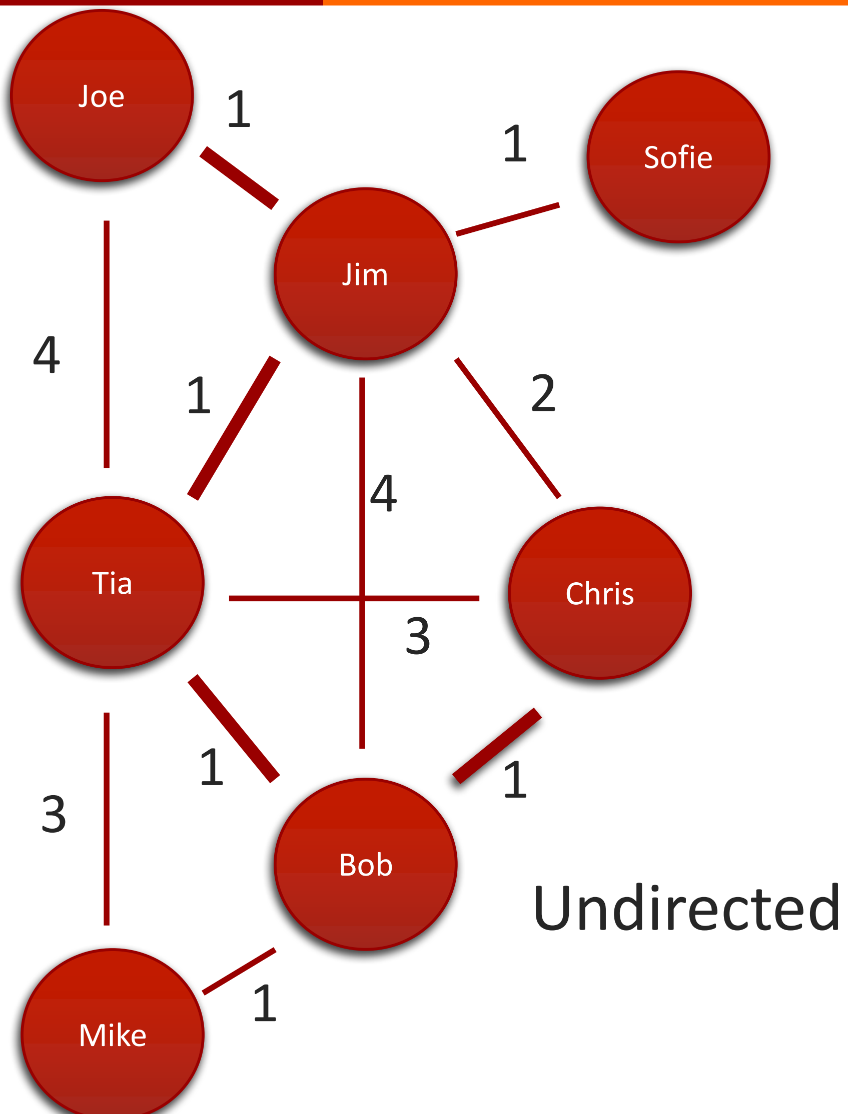
Prim’s algorithm



Undirected

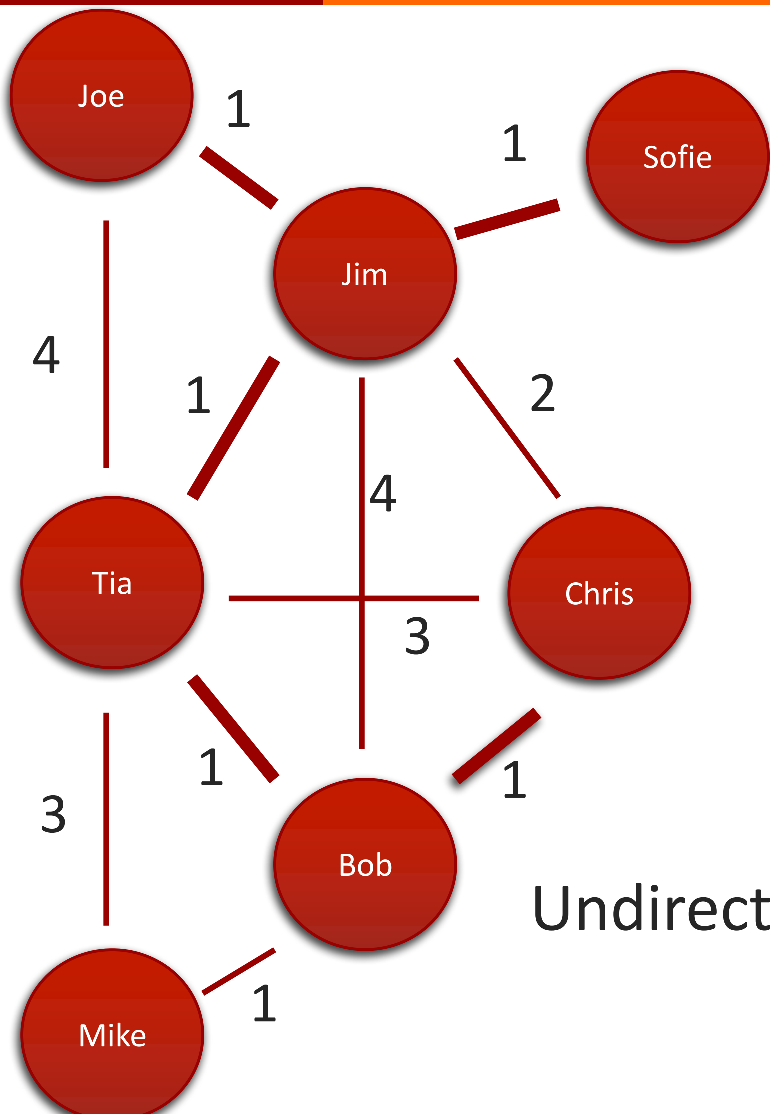
1	Joe-Jim
1	Jim-Sofie
1	**Jim-Tia**
1	**Tia-Bob**
1	**Chris-Bob**
1	Mike-Bob
2	Chris-Jim
3	Tia-Chris
3	Mike-Tia
4	Joe-Tia
4	Jim-Bob

Prim's algorithm



1	**Joe-Jim**
1	Jim-Sofie
1	**Jim-Tia**
1	**Tia-Bob**
1	**Chris-Bob**
1	Mike-Bob
2	Chris-Jim
3	Tia-Chris
3	Mike-Tia
4	Joe-Tia
4	Jim-Bob

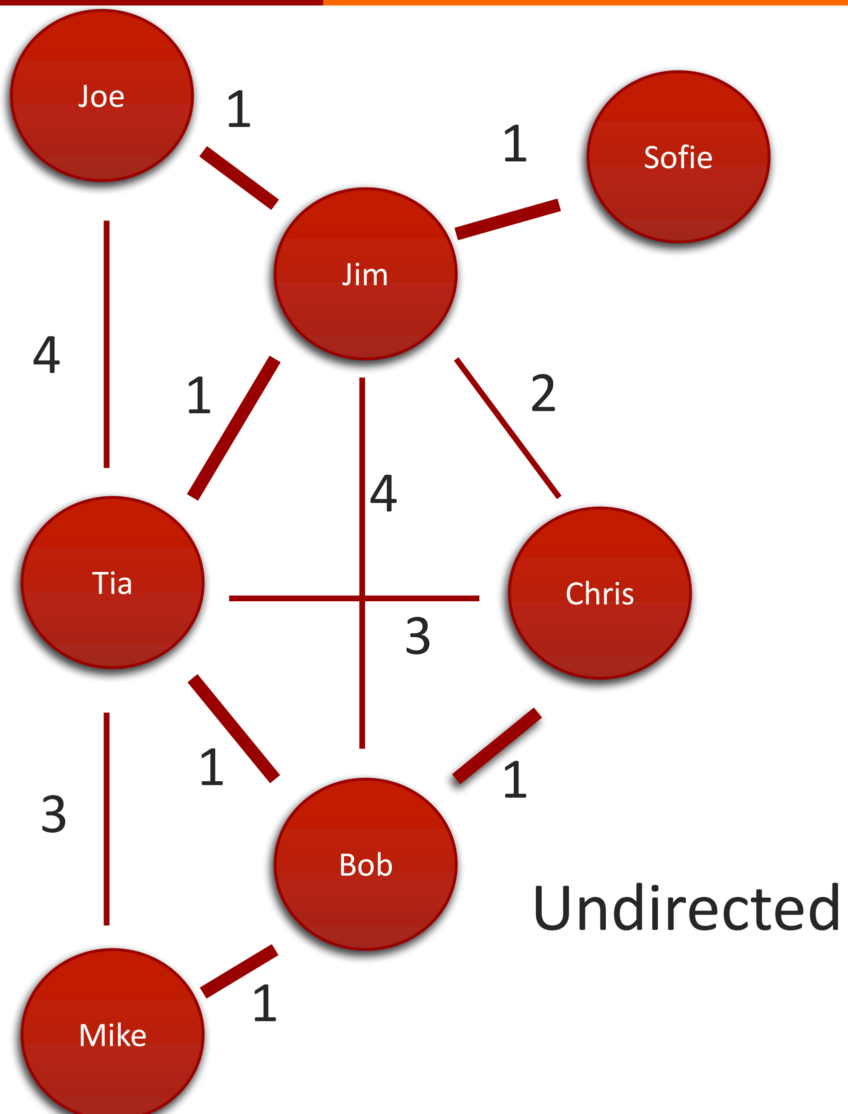
Prim's algorithm



Undirected

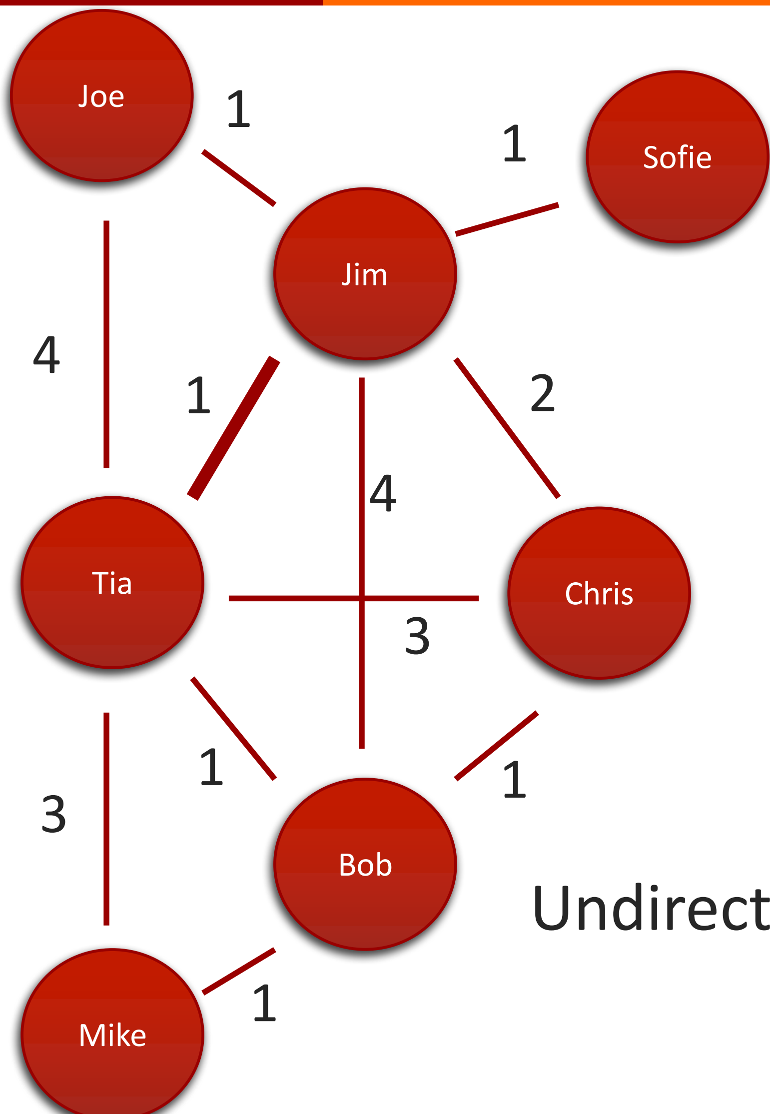
1	**Joe-Jim**
1	**Jim-Sofie**
1	**Jim-Tia**
1	**Tia-Bob**
1	**Chris-Bob**
1	Mike-Bob
2	Chris-Jim
3	Tia-Chris
3	Mike-Tia
4	Joe-Tia
4	Jim-Bob

Prim's algorithm



1	**Joe-Jim**
1	**Jim-Sofie**
1	**Jim-Tia**
1	**Tia-Bob**
1	**Chris-Bob**
1	**Mike-Bob**
2	Chris-Jim
3	Tia-Chris
3	Mike-Tia
4	Joe-Tia
4	Jim-Bob

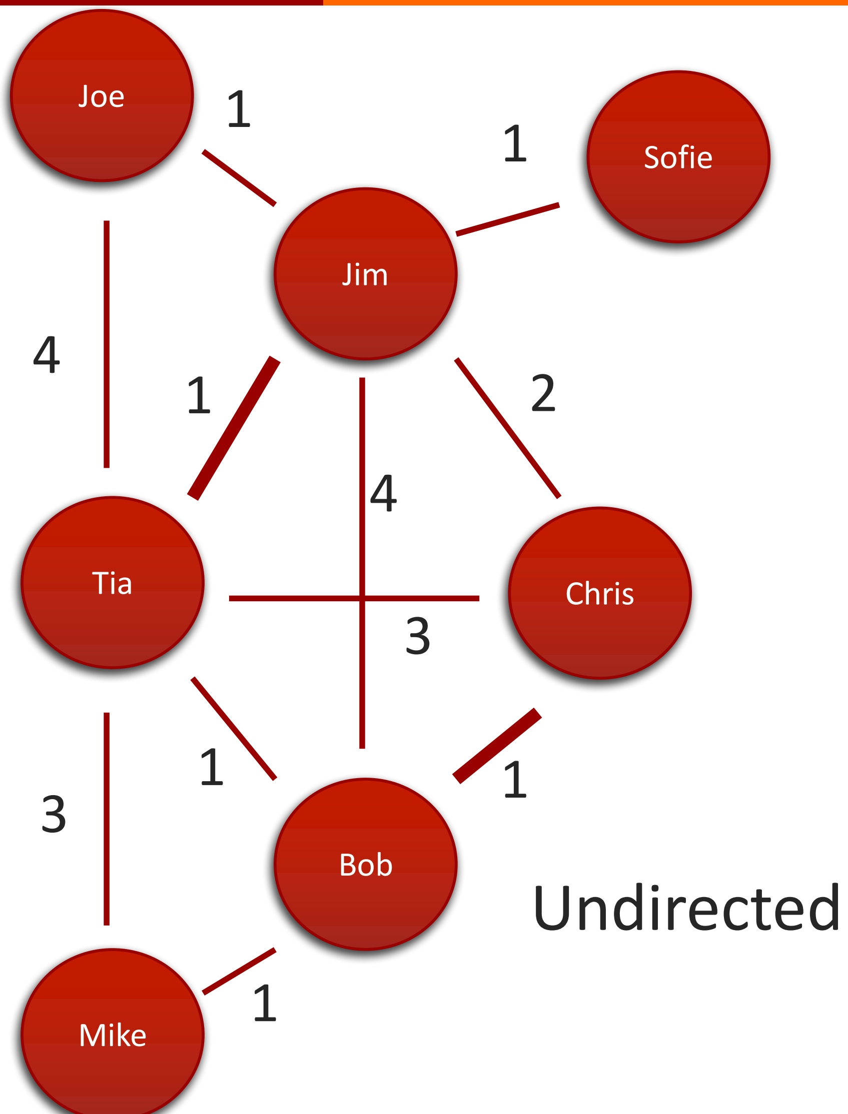
Kruskal's algorithm



Undirected

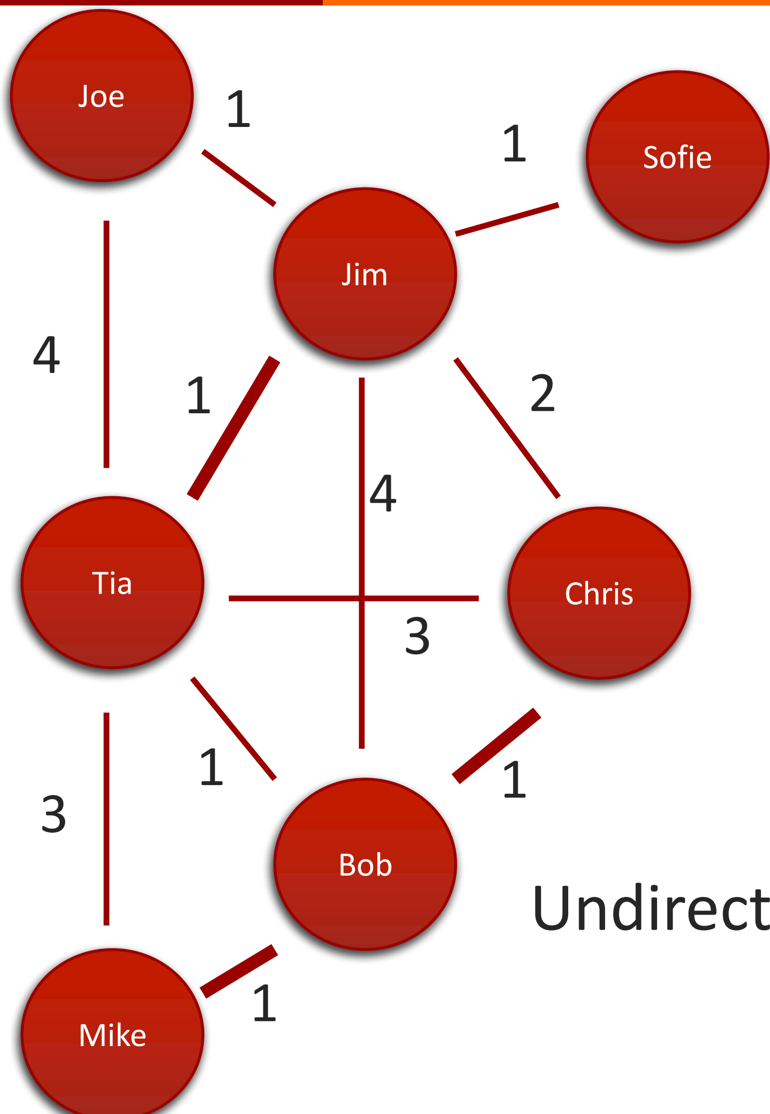
1	Joe-Jim
1	Jim-Sofie
1	**Jim-Tia**
1	Tia-Bob
1	Chris-Bob
1	Mike-Bob
2	Chris-Jim
3	Tia-Chris
3	Mike-Tia
4	Joe-Tia
4	Jim-Bob

Kruskal's algorithm



1	Joe-Jim
1	Jim-Sofie
1	**Jim-Tia**
1	Tia-Bob
1	**Chris-Bob**
1	Mike-Bob
2	Chris-Jim
3	Tia-Chris
3	Mike-Tia
4	Joe-Tia
4	Jim-Bob

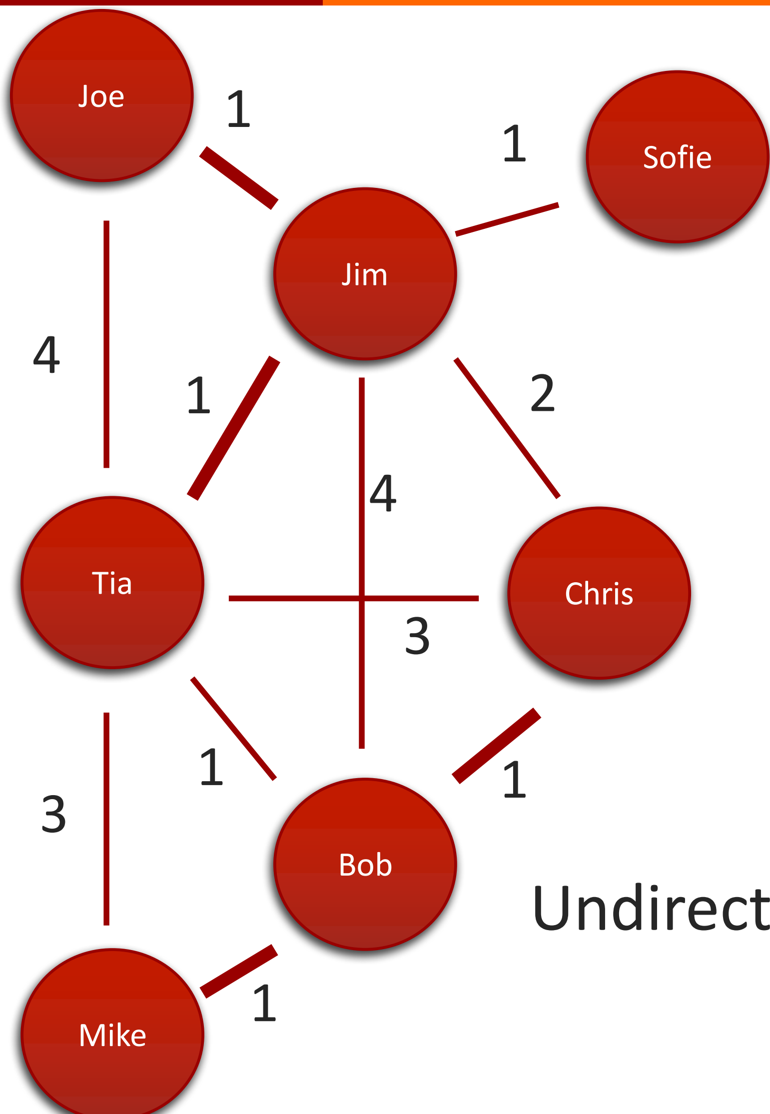
Kruskal’s algorithm



Undirected

1	Joe-Jim
1	Jim-Sofie
1	**Jim-Tia**
1	Tia-Bob
1	**Chris-Bob**
1	**Mike-Bob**
2	Chris-Jim
3	Tia-Chris
3	Mike-Tia
4	Joe-Tia
4	Jim-Bob

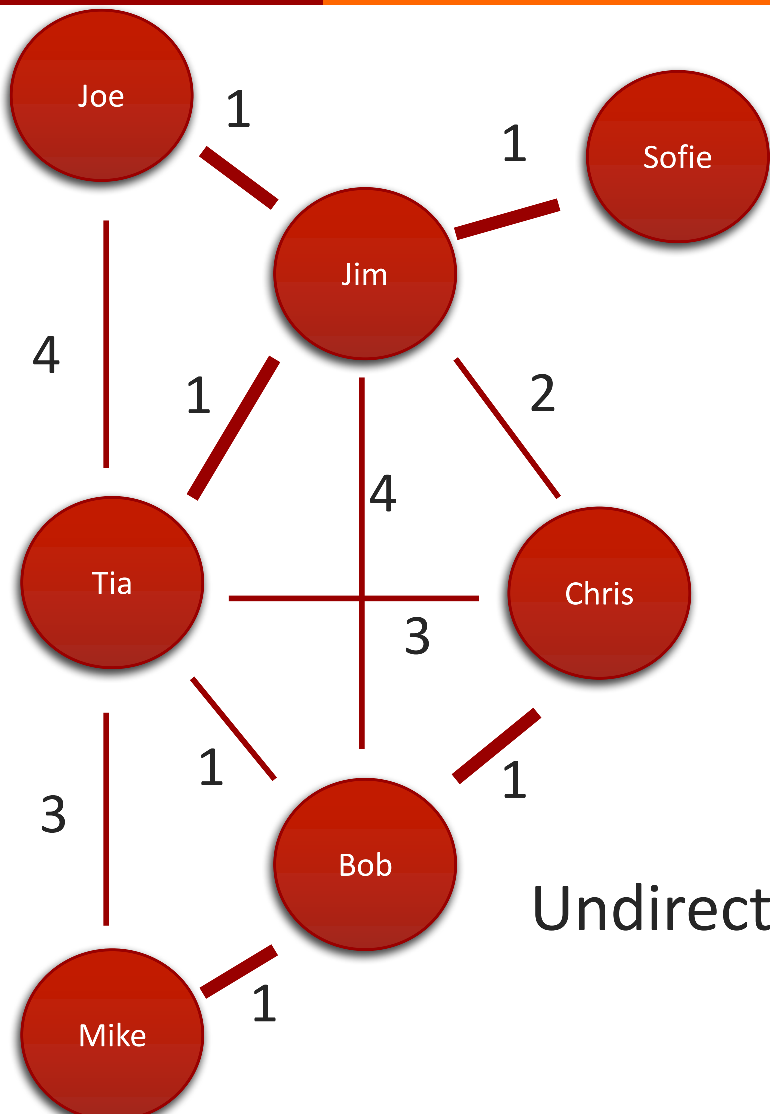
Kruskal's algorithm



Undirected

1	**Joe-Jim**
1	Jim-Sofie
1	**Jim-Tia**
1	Tia-Bob
1	**Chris-Bob**
1	**Mike-Bob**
2	Chris-Jim
3	Tia-Chris
3	Mike-Tia
4	Joe-Tia
4	Jim-Bob

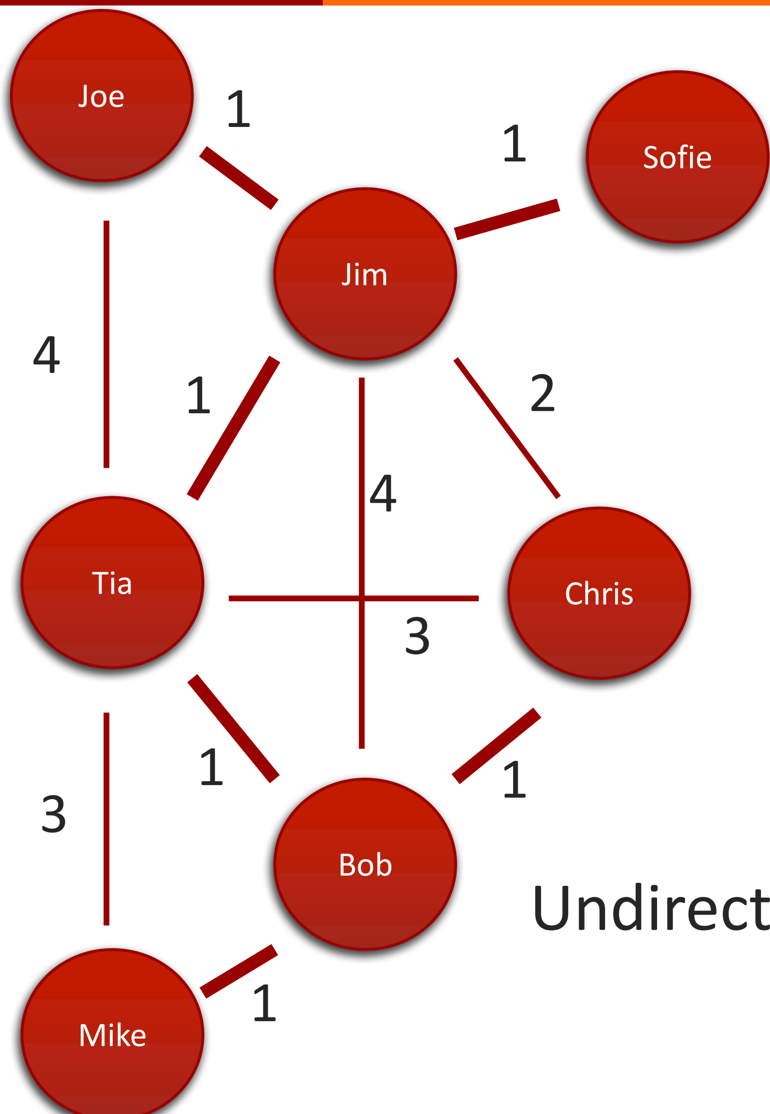
Kruskal's algorithm



Undirected

1	**Joe-Jim**
1	**Jim-Sofie**
1	**Jim-Tia**
1	Tia-Bob
1	**Chris-Bob**
1	**Mike-Bob**
2	Chris-Jim
3	Tia-Chris
3	Mike-Tia
4	Joe-Tia
4	Jim-Bob

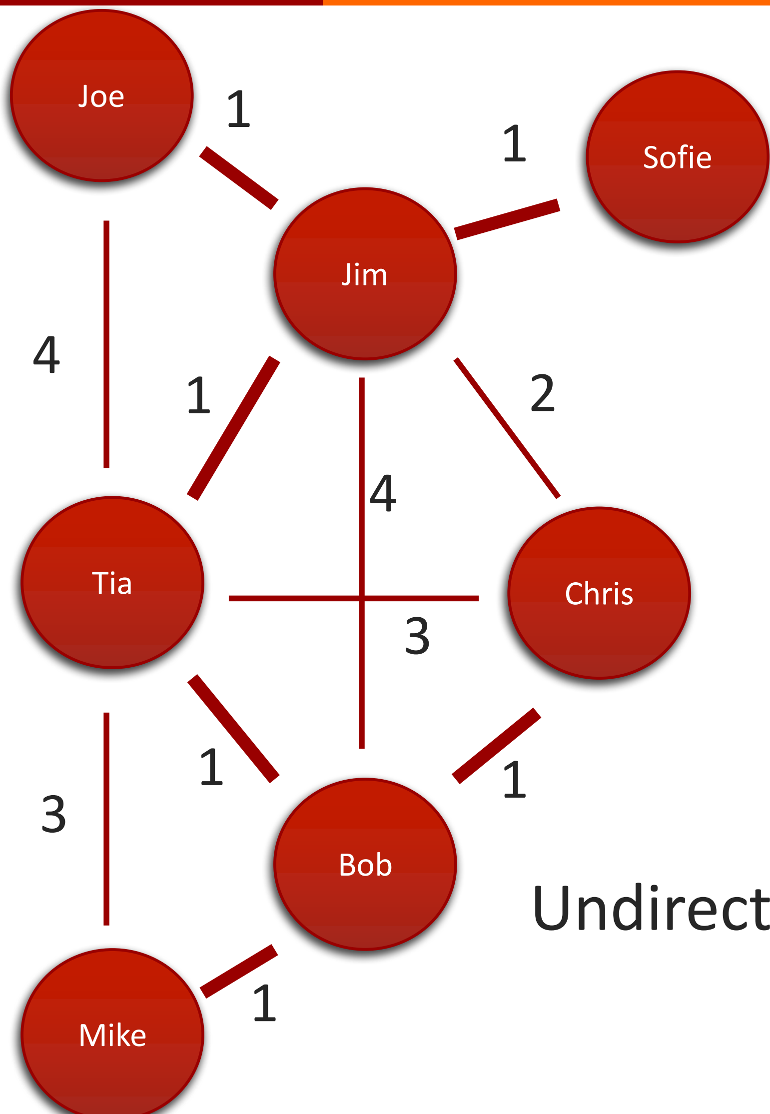
Kruskal's algorithm



Undirected

1	**Joe-Jim**
1	**Jim-Sofie**
1	**Jim-Tia**
1	**Tia-Bob**
1	**Chris-Bob**
1	**Mike-Bob**
2	Chris-Jim
3	Tia-Chris
3	Mike-Tia
4	Joe-Tia
4	Jim-Bob

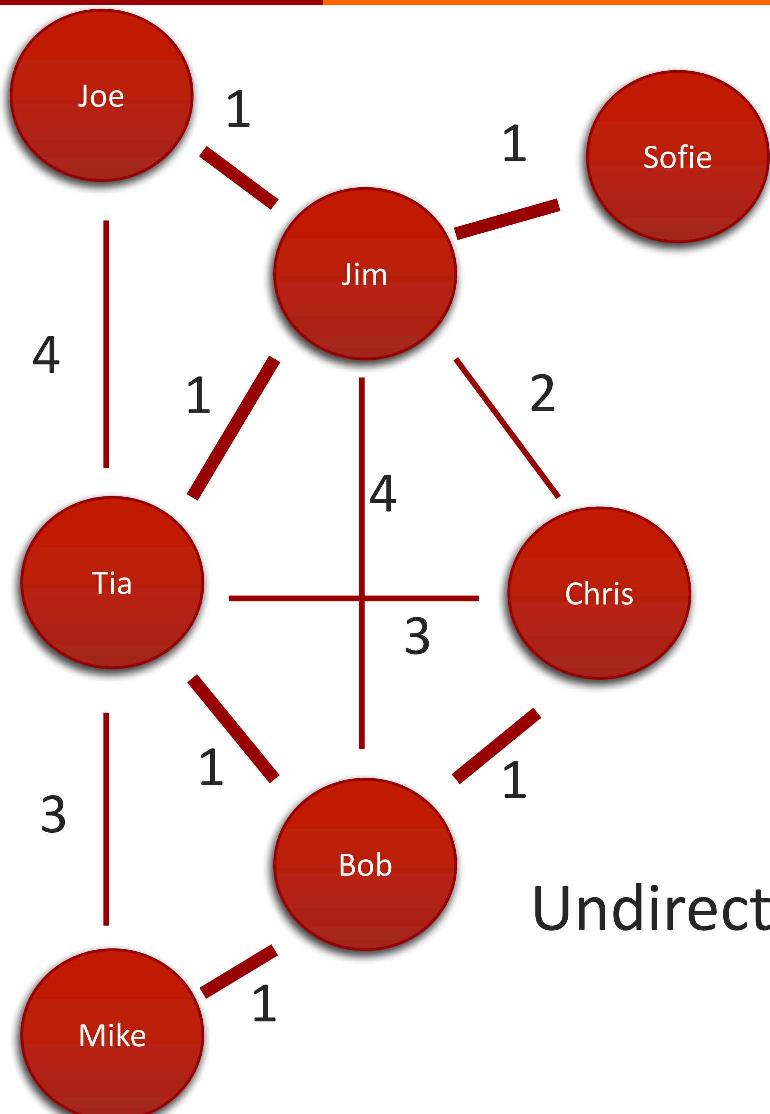
Kruskal's algorithm



Undirected

1	**Joe-Jim**
1	**Jim-Sofie**
1	**Jim-Tia**
1	**Tia-Bob**
1	**Chris-Bob**
1	**Mike-Bob**
2	Chris-Jim
3	Tia-Chris
3	Mike-Tia
4	Joe-Tia
4	Jim-Bob

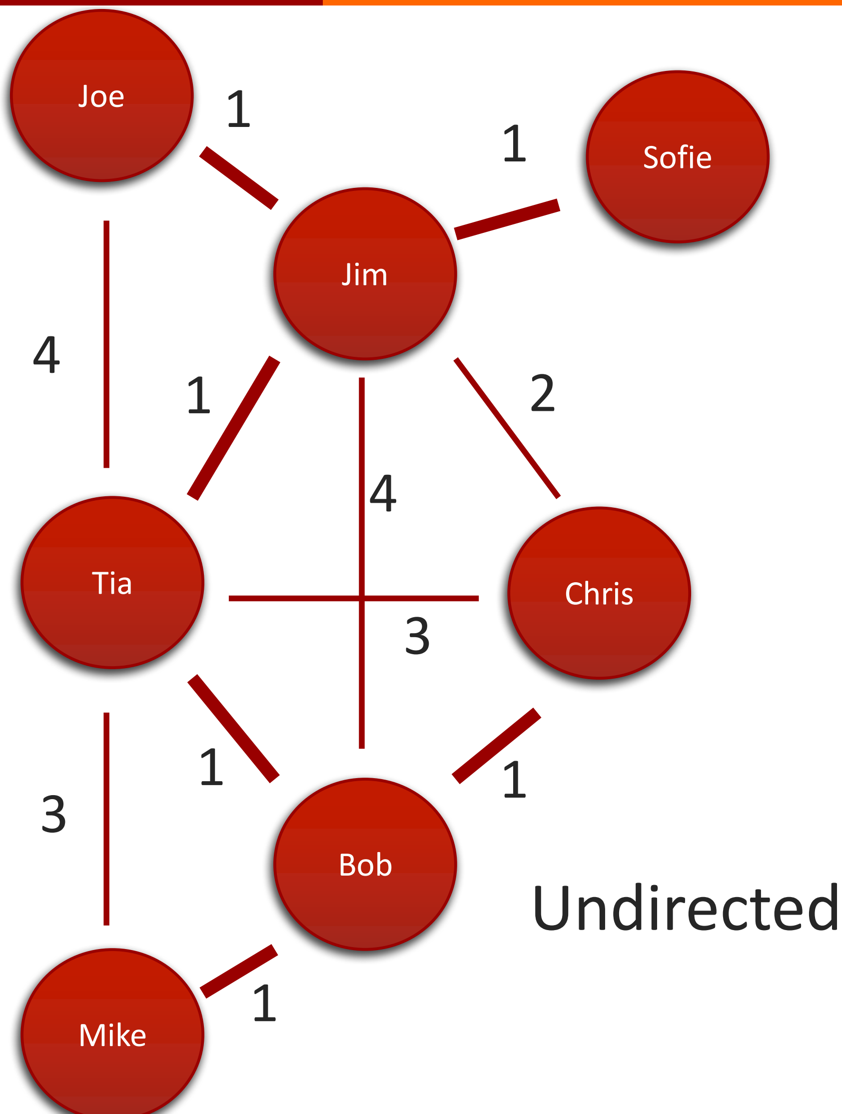
Kruskal's algorithm



Undirected

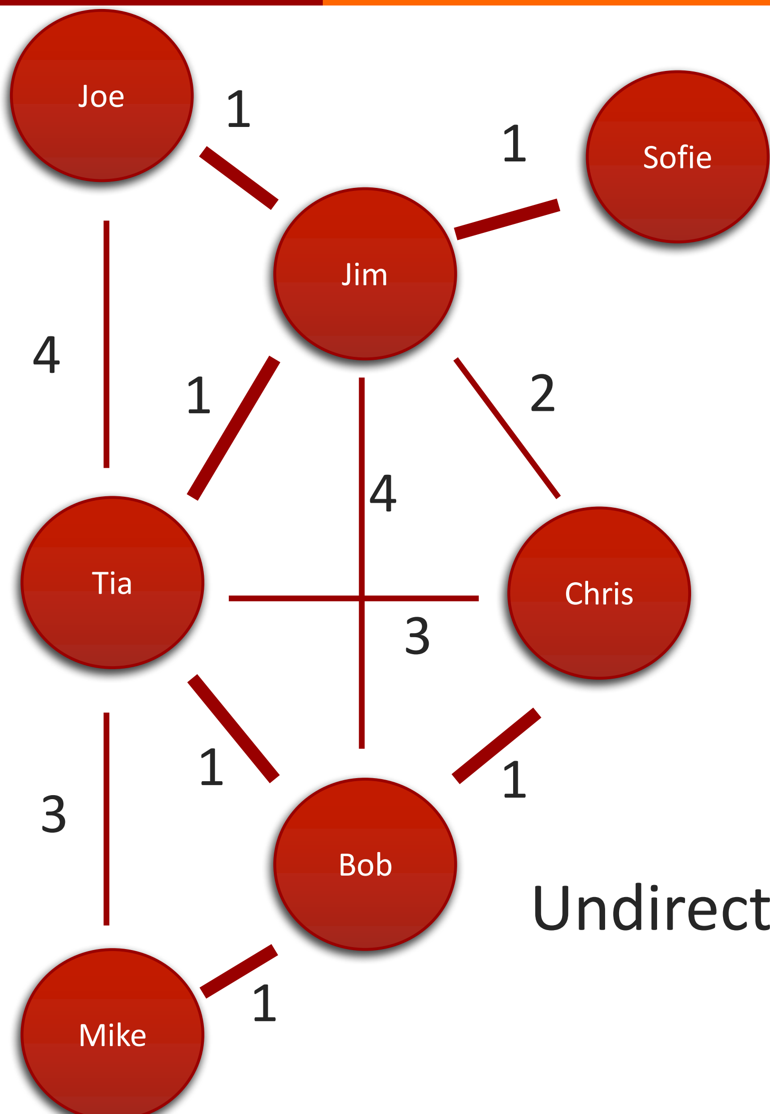
1	**Joe-Jim**
1	**Jim-Sofie**
1	**Jim-Tia**
1	**Tia-Bob**
1	**Chris-Bob**
1	**Mike-Bob**
2	Chris-Jim
3	Tia-Chris
3	Mike-Tia
4	Joe-Tia
4	Jim-Bob

Kruskal's algorithm



1	**Joe-Jim**
1	**Jim-Sofie**
1	**Jim-Tia**
1	**Tia-Bob**
1	**Chris-Bob**
1	**Mike-Bob**
2	Chris-Jim
3	Tia-Chris
3	Mike-Tia
4	Joe-Tia
4	Jim-Bob

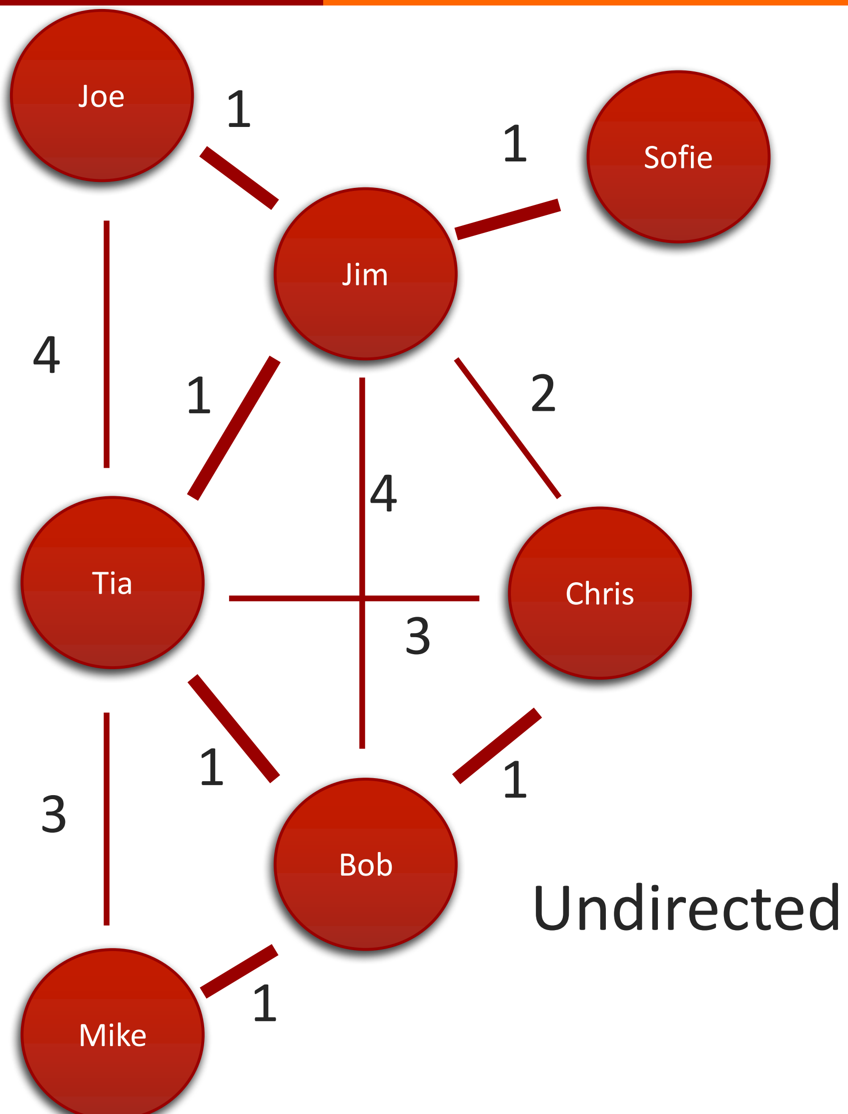
Kruskal’s algorithm



Undirected

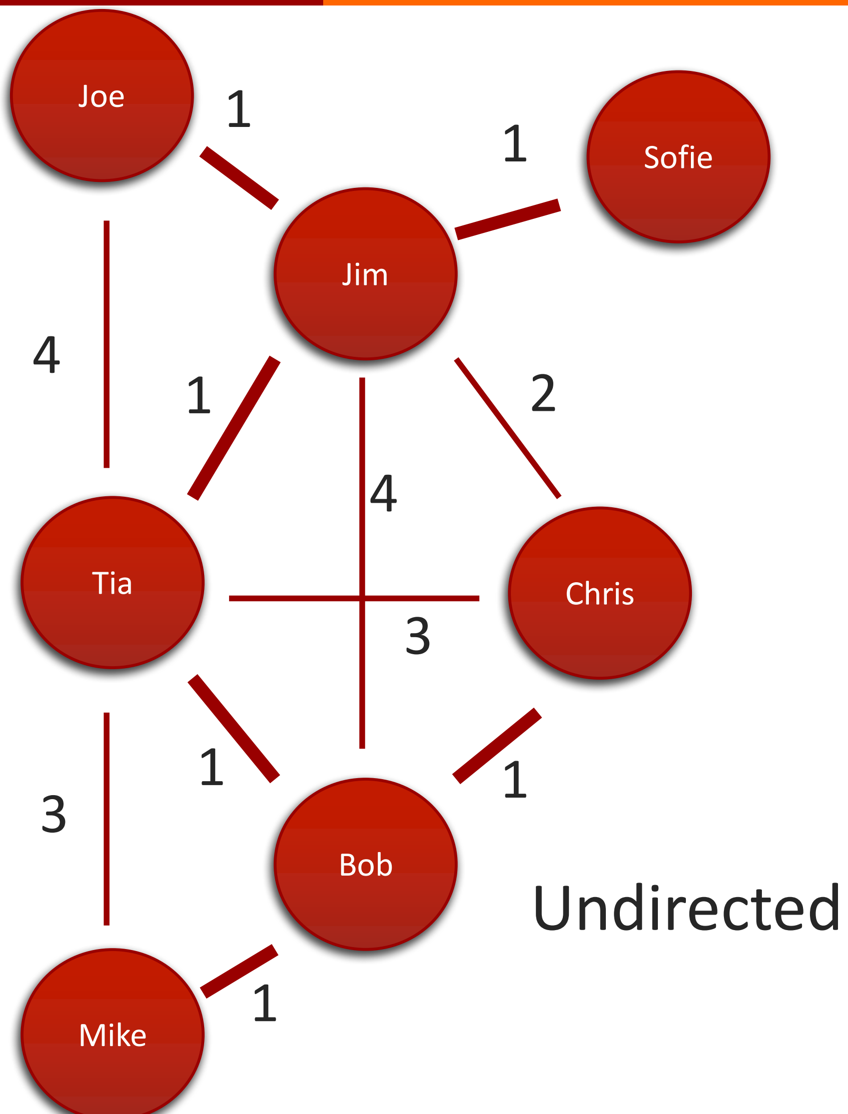
1	**Joe-Jim**
1	**Jim-Sofie**
1	**Jim-Tia**
1	**Tia-Bob**
1	**Chris-Bob**
1	**Mike-Bob**
2	Chris-Jim
3	Tia-Chris
3	Mike-Tia
4	Joe-Tia
4	Jim-Bob

Kruskal's algorithm



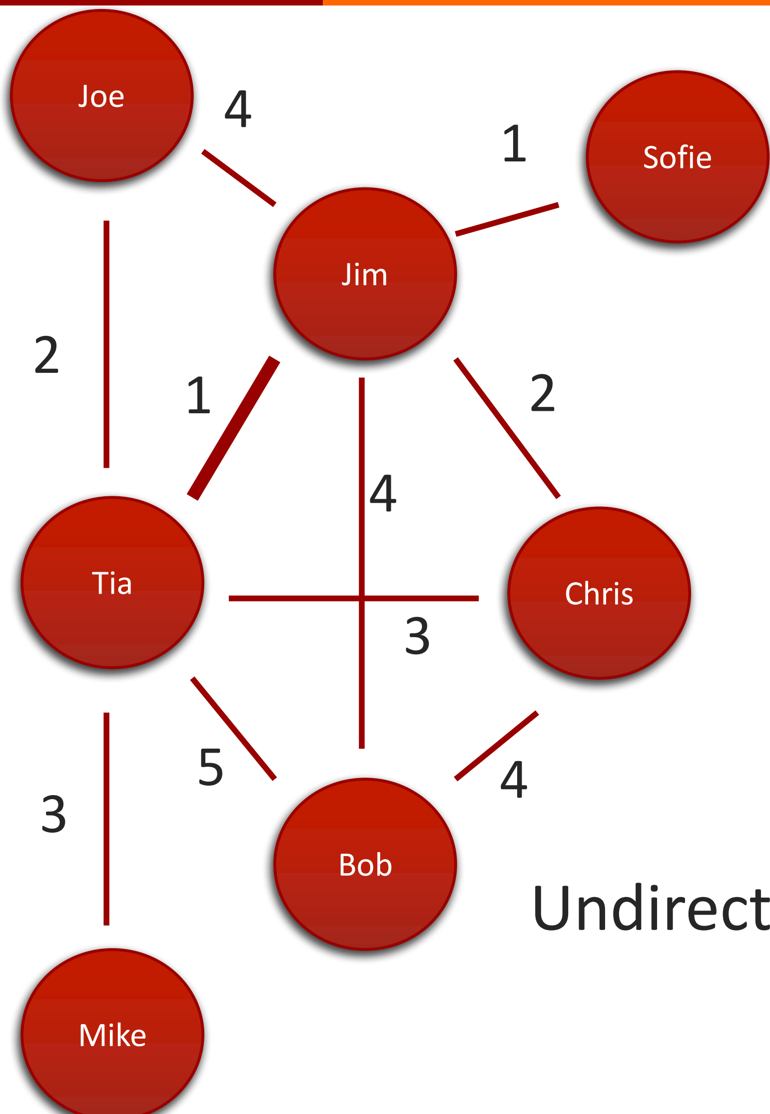
1	**Joe-Jim**
1	**Jim-Sofie**
1	**Jim-Tia**
1	**Tia-Bob**
1	**Chris-Bob**
1	**Mike-Bob**
2	Chris-Jim
3	Tia-Chris
3	Mike-Tia
4	Joe-Tia
4	Jim-Bob

Kruskal's algorithm



1	**Joe-Jim**
1	**Jim-Sofie**
1	**Jim-Tia**
1	**Tia-Bob**
1	**Chris-Bob**
1	**Mike-Bob**
2	Chris-Jim
3	Tia-Chris
3	Mike-Tia
4	Joe-Tia
4	Jim-Bob

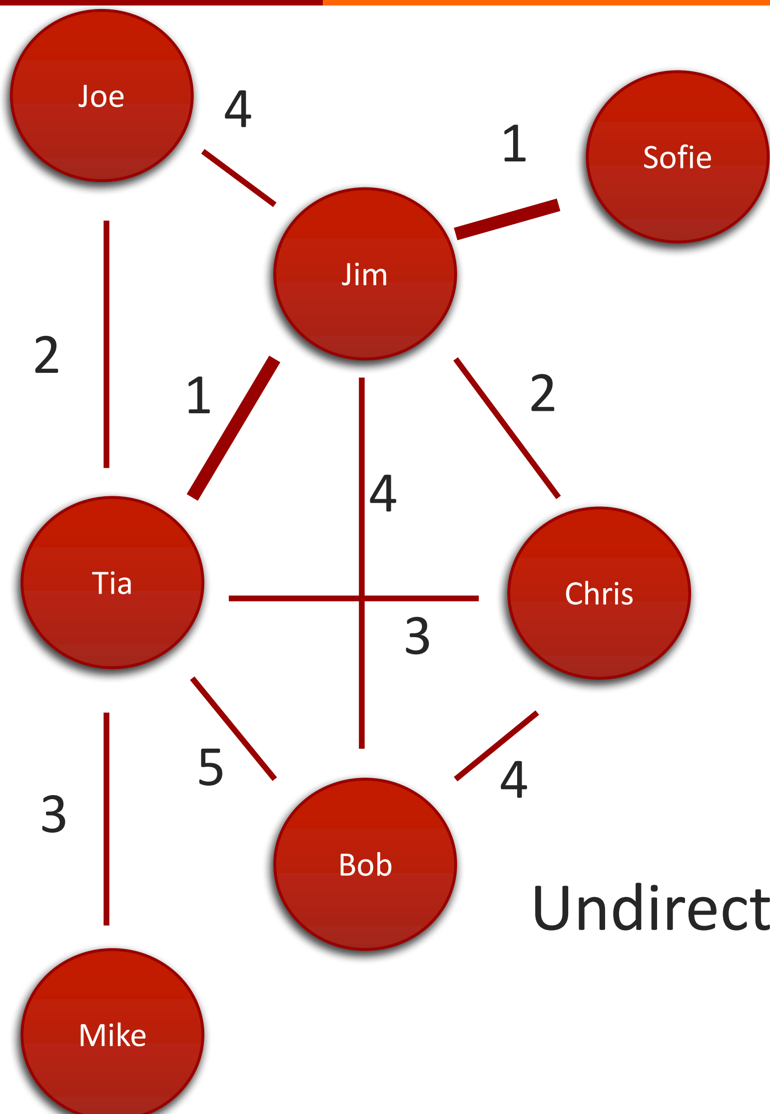
Kruskal’s algorithm example #2



Undirected

1	Jim-Sofie
1	**Jim-Tia**
2	Chris-Jim
2	Joe-Tia
3	Tia-Chris
3	Mike-Tia
4	Jim-Bob
4	Joe-Jim
4	Chris-Bob
5	Tia-Bob

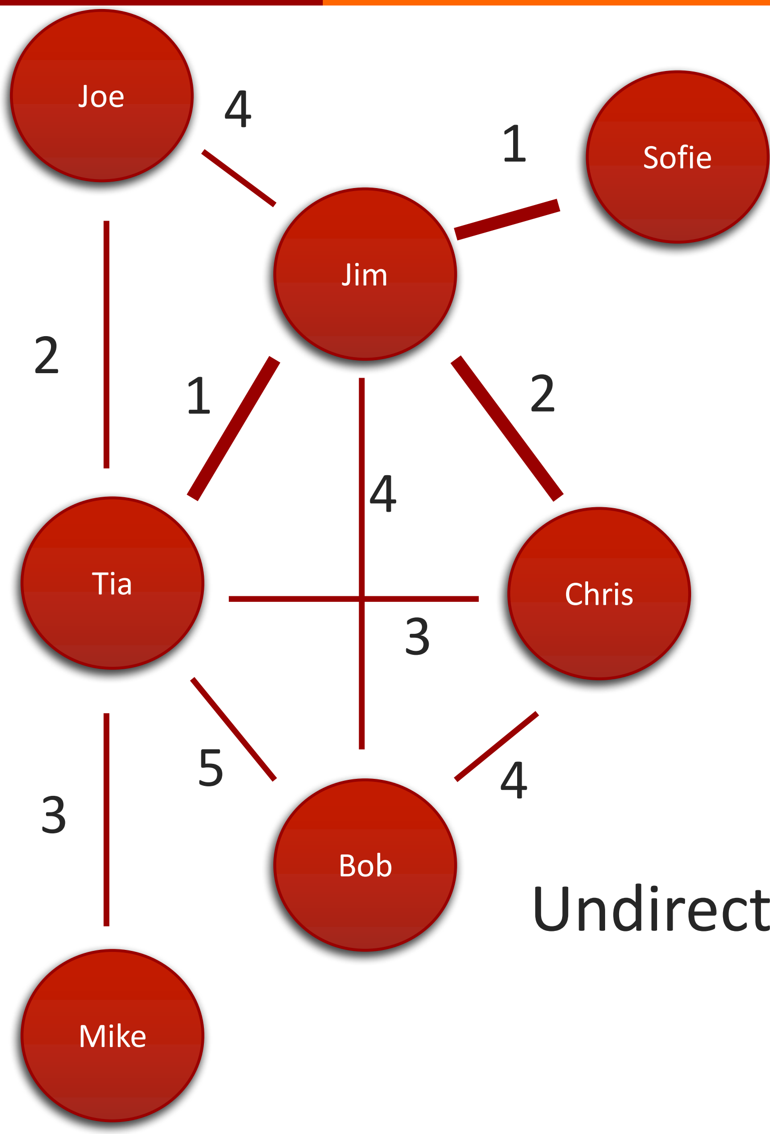
Kruskal's algorithm example #2



Undirected

1	**Jim-Sofie**
1	**Jim-Tia**
2	Chris-Jim
2	Joe-Tia
3	Tia-Chris
3	Mike-Tia
4	Jim-Bob
4	Joe-Jim
4	Chris-Bob
5	Tia-Bob

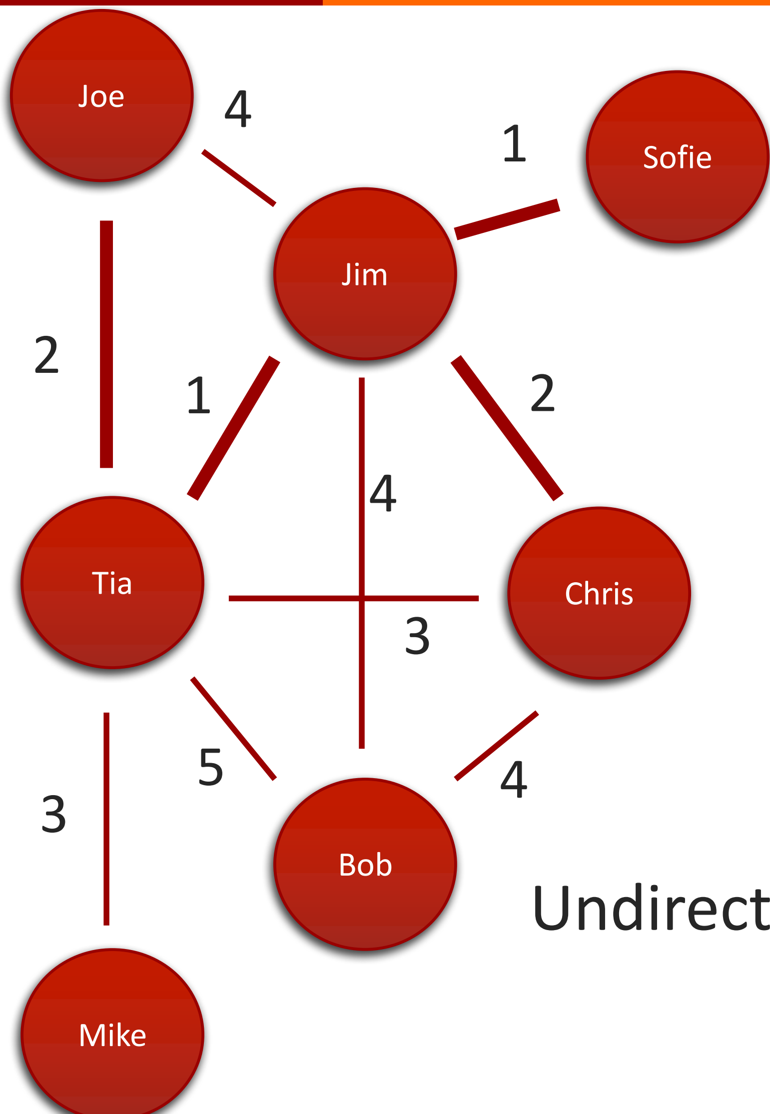
Kruskal’s algorithm example #2



Undirected

1	**Jim-Sofie**
1	**Jim-Tia**
2	**Chris-Jim**
2	Joe-Tia
3	Tia-Chris
3	Mike-Tia
4	Jim-Bob
4	Joe-Jim
4	Chris-Bob
5	Tia-Bob

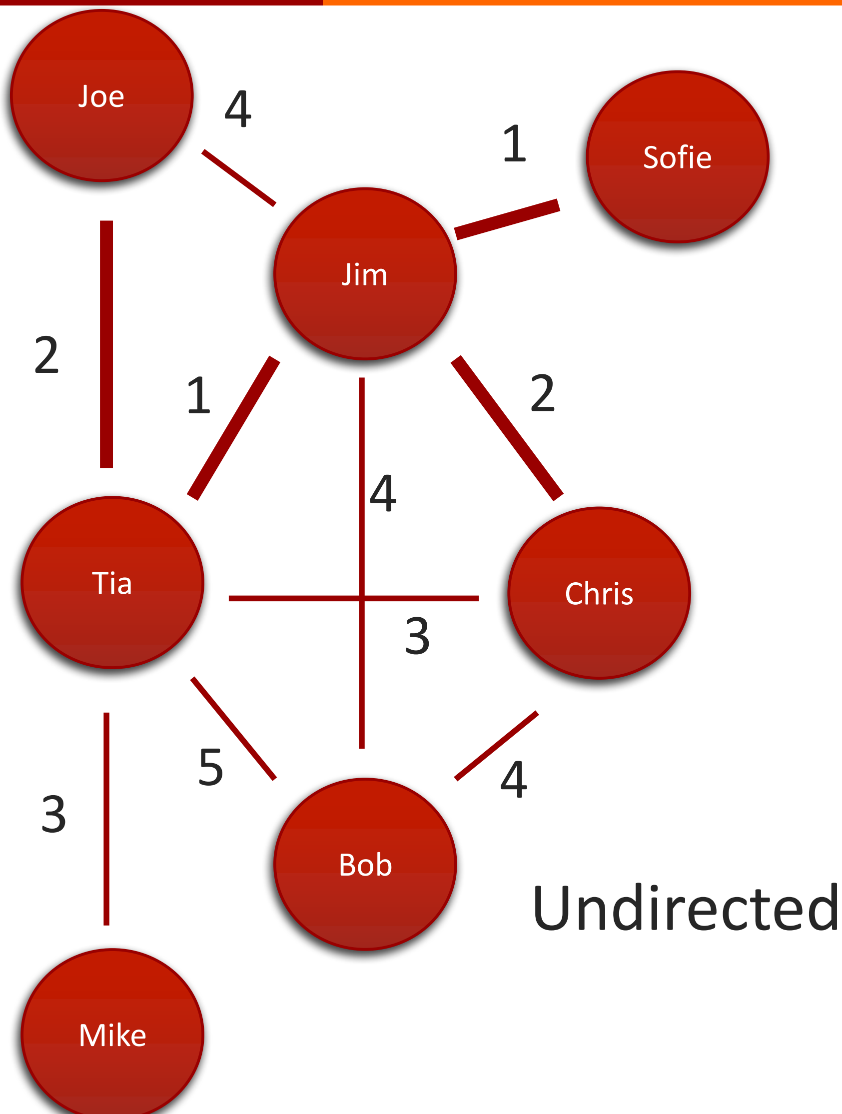
Kruskal's algorithm example #2



Undirected

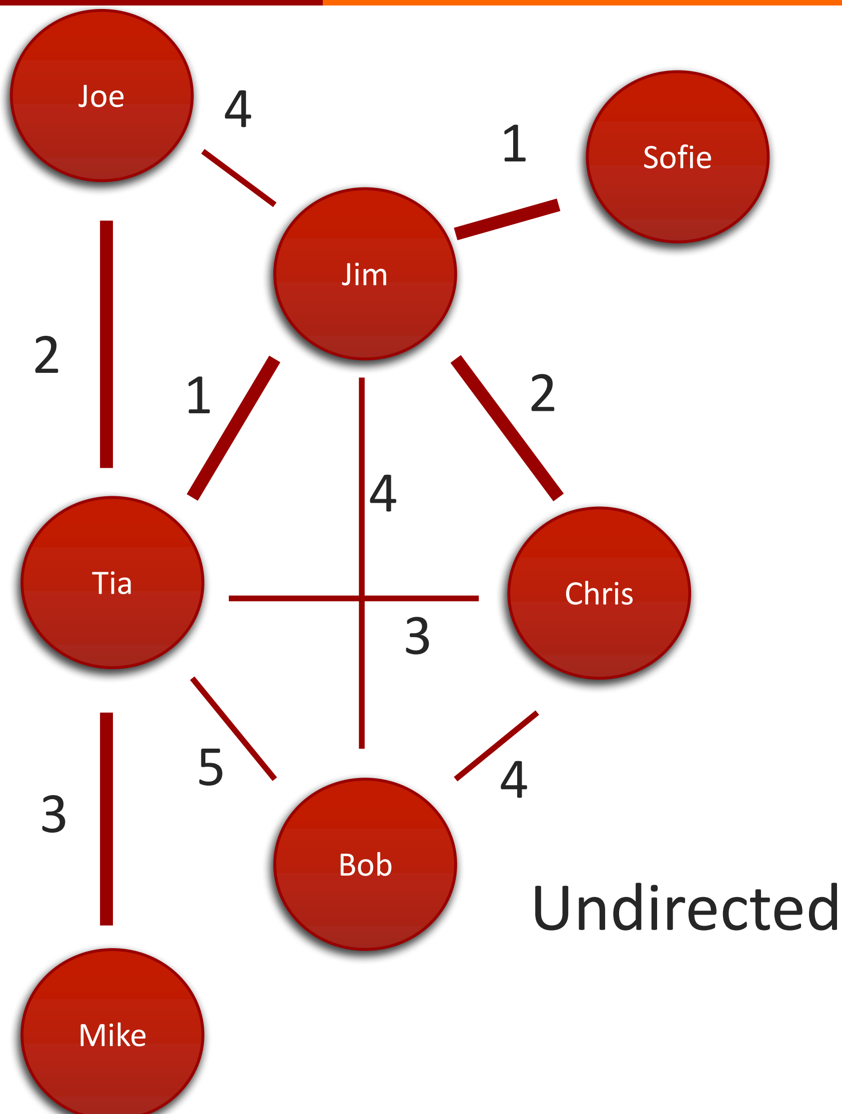
1	**Jim-Sofie**
1	**Jim-Tia**
2	**Chris-Jim**
2	**Joe-Tia**
3	Tia-Chris
3	Mike-Tia
4	Jim-Bob
4	Joe-Jim
4	Chris-Bob
5	Tia-Bob

Kruskal’s algorithm example #2



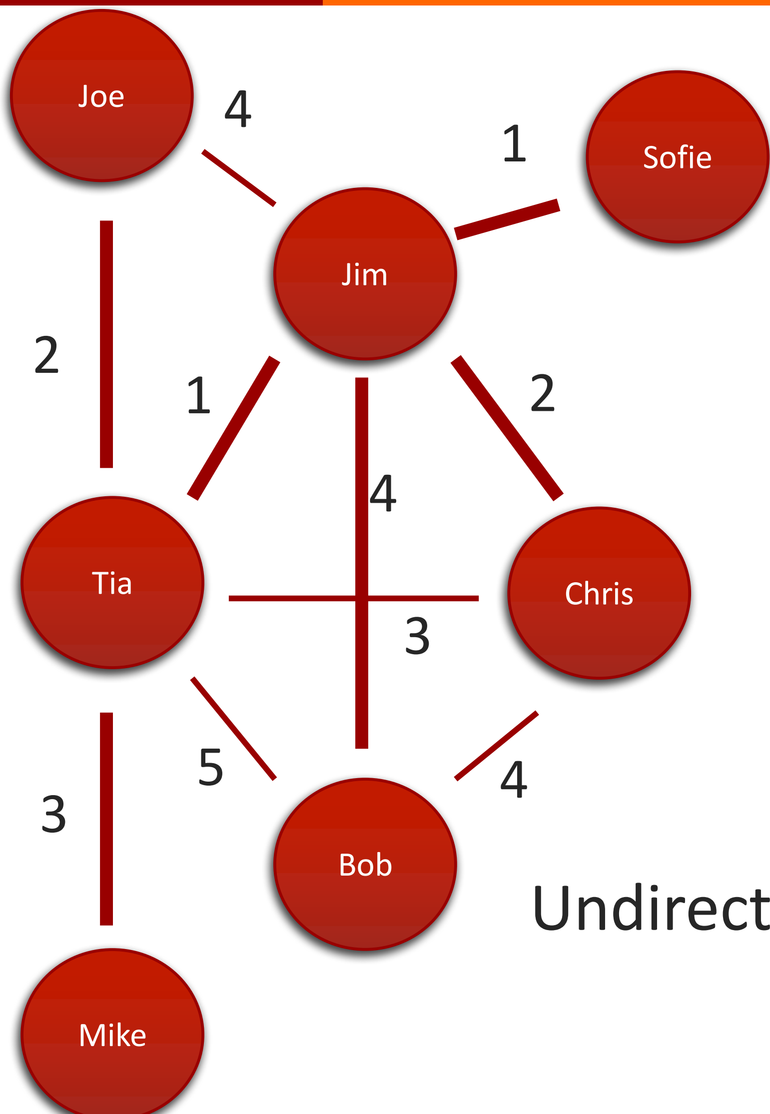
1	**Jim-Sofie**
1	**Jim-Tia**
2	**Chris-Jim**
2	**Joe-Tia**
3	Tia-Chris
3	Mike-Tia
4	Jim-Bob
4	Joe-Jim
4	Chris-Bob
5	Tia-Bob

Kruskal’s algorithm example #2



1	**Jim-Sofie**
1	**Jim-Tia**
2	**Chris-Jim**
2	**Joe-Tia**
3	Tia-Chris
3	**Mike-Tia**
4	Jim-Bob
4	Joe-Jim
4	Chris-Bob
5	Tia-Bob

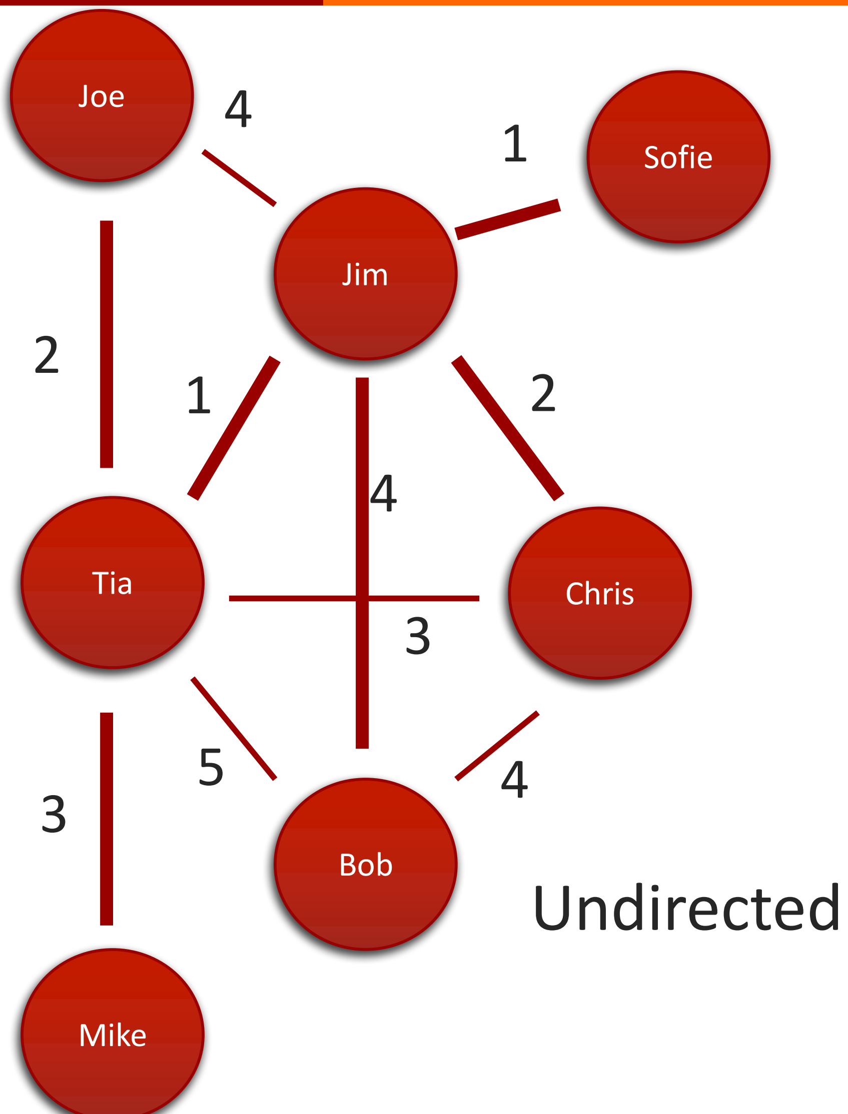
Kruskal's algorithm example #2



Undirected

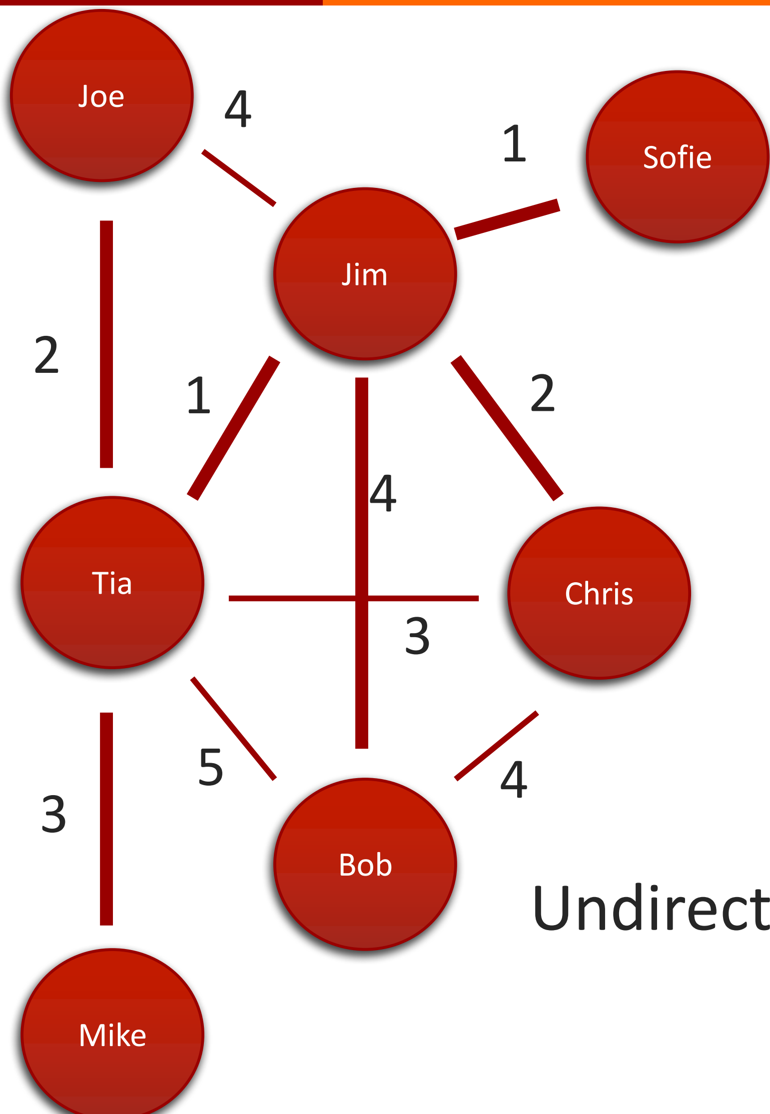
1	**Jim-Sofie**
1	**Jim-Tia**
2	**Chris-Jim**
2	**Joe-Tia**
3	Tia-Chris
3	**Mike-Tia**
4	**Jim-Bob**
4	Joe-Jim
4	Chris-Bob
5	Tia-Bob

Kruskal's algorithm example #2



1	**Jim-Sofie**
1	**Jim-Tia**
2	**Chris-Jim**
2	**Joe-Tia**
3	Tia-Chris
3	**Mike-Tia**
4	**Jim-Bob**
4	Joe-Jim
4	Chris-Bob
5	Tia-Bob

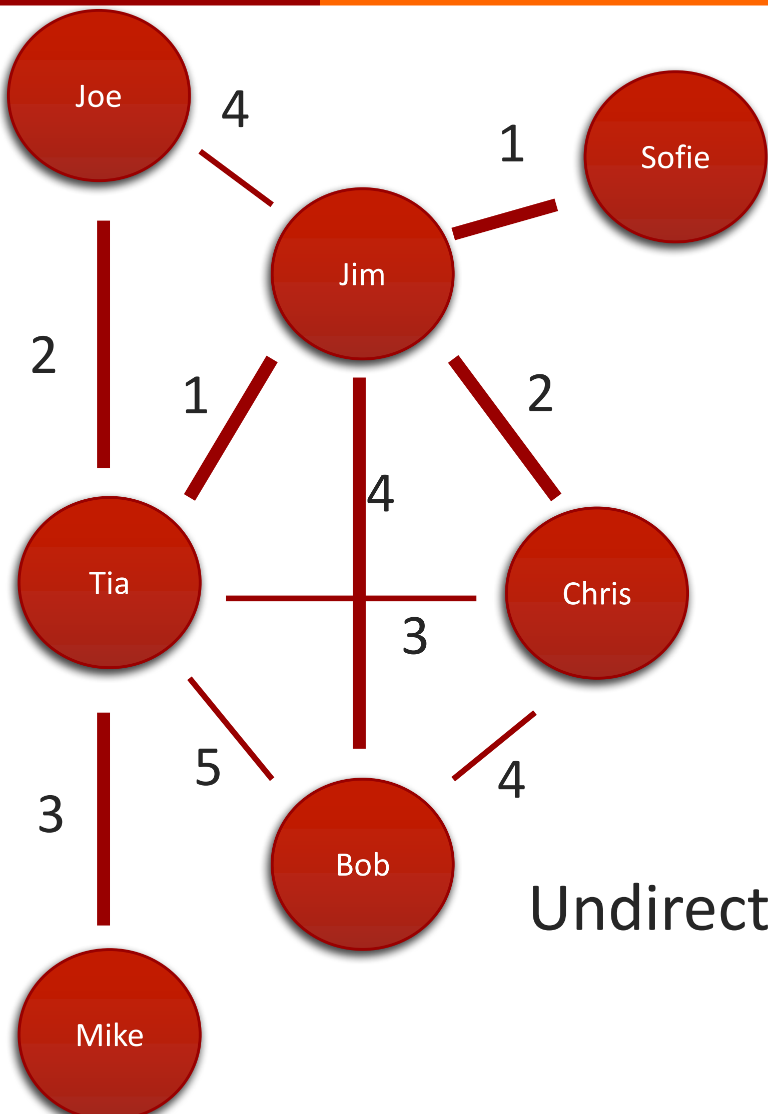
Kruskal's algorithm example #2



Undirected

1	**Jim-Sofie**
1	**Jim-Tia**
2	**Chris-Jim**
2	**Joe-Tia**
3	Tia-Chris
3	**Mike-Tia**
4	**Jim-Bob**
4	Joe-Jim
4	Chris-Bob
5	Tia-Bob

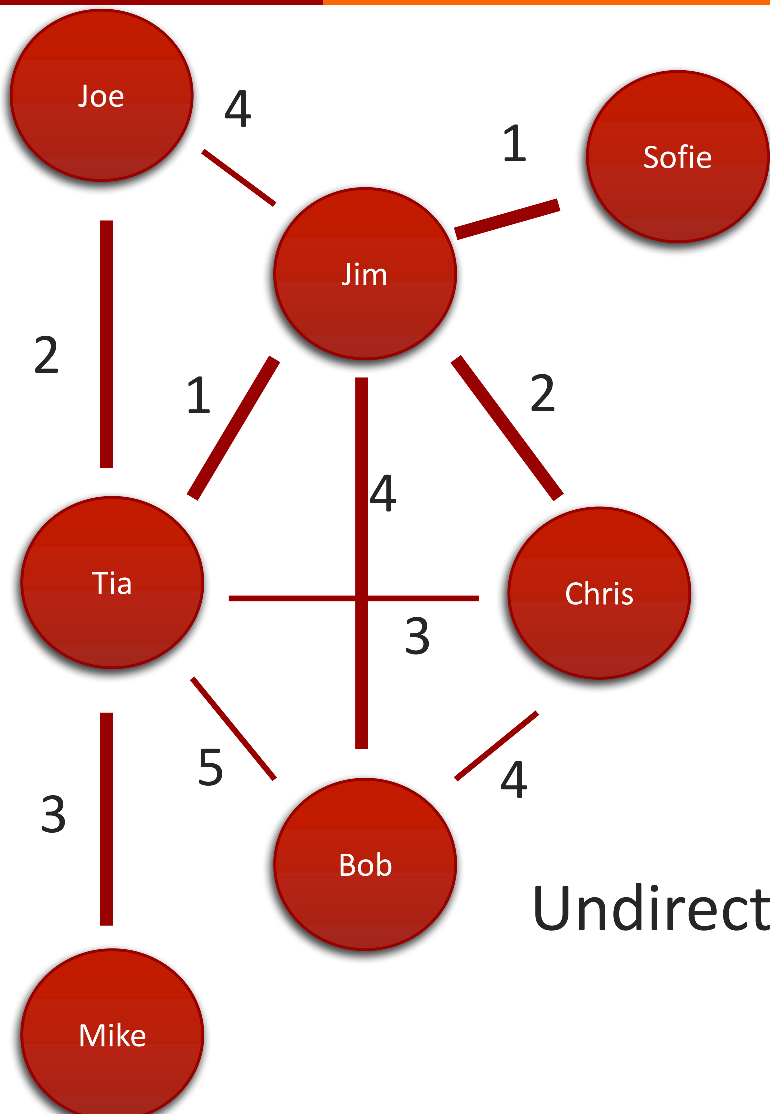
Kruskal’s algorithm example #2



Undirected

1	**Jim-Sofie**
1	**Jim-Tia**
2	**Chris-Jim**
2	**Joe-Tia**
3	Tia-Chris
3	**Mike-Tia**
4	**Jim-Bob**
4	Joe-Jim
4	Chris-Bob
5	Tia-Bob

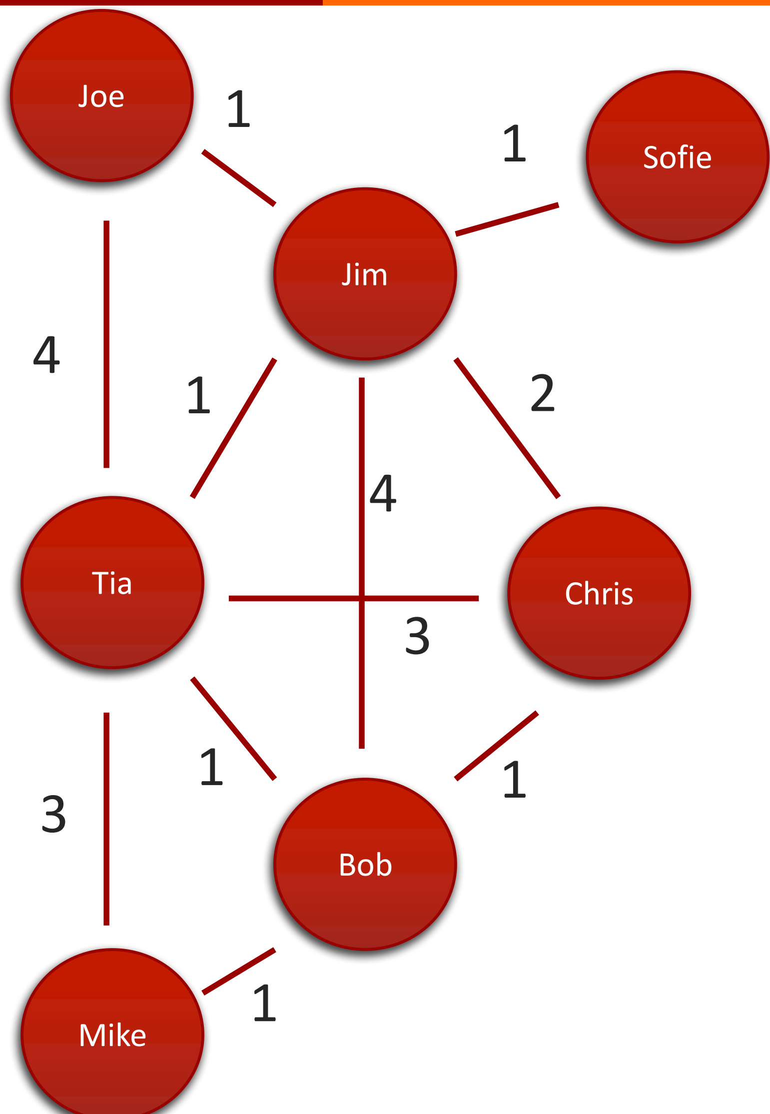
Kruskal's algorithm example #2



Undirected

1	**Jim-Sofie**
1	**Jim-Tia**
2	**Chris-Jim**
2	**Joe-Tia**
3	Tia-Chris
3	**Mike-Tia**
4	**Jim-Bob**
4	Joe-Jim
4	Chris-Bob
5	Tia-Bob

Shortest path

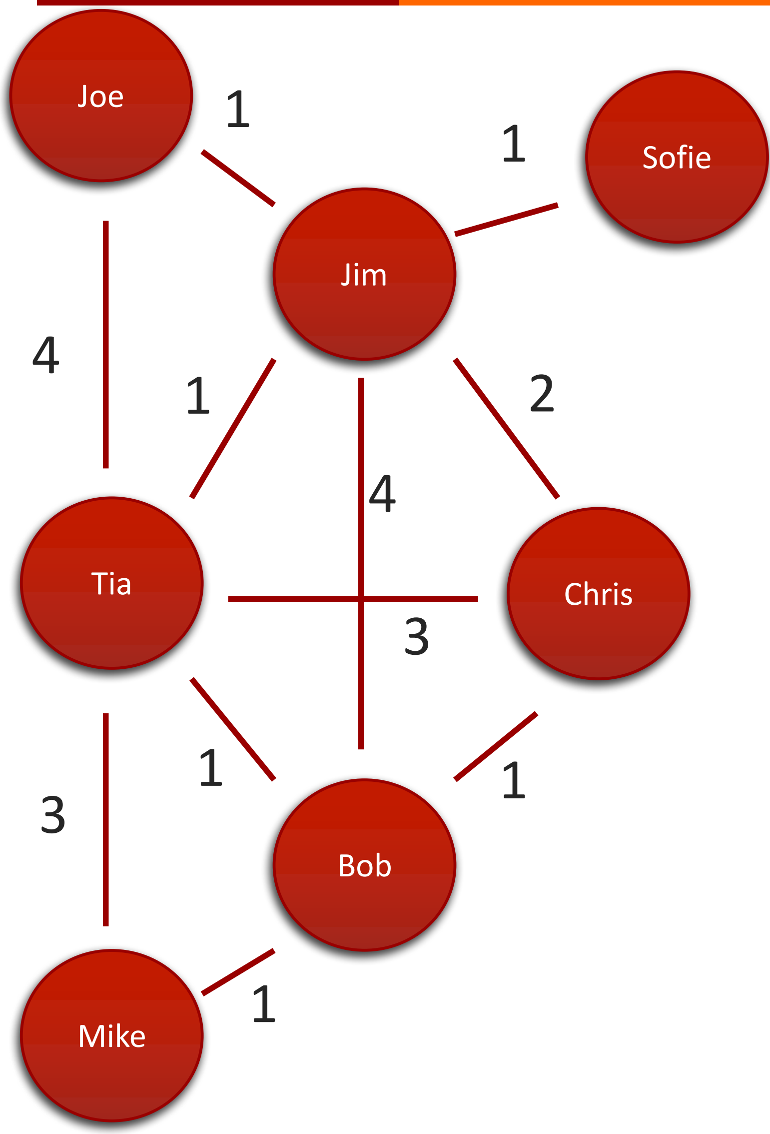


- For a given source vertex (node) in the graph, it finds the path with lowest cost (i.e. the shortest path) between that vertex and every other vertex.
- Say your source vertex is Mike
  - Lowest cost path from Mike to Jim is Mike – Tia – Jim (cost 3)
  - Lowest cost path from Mike to Joe is Mike – Bob – Tia – Jim – Joe (cost 4)
  - …

Dijkstra's algorithm: Basic idea

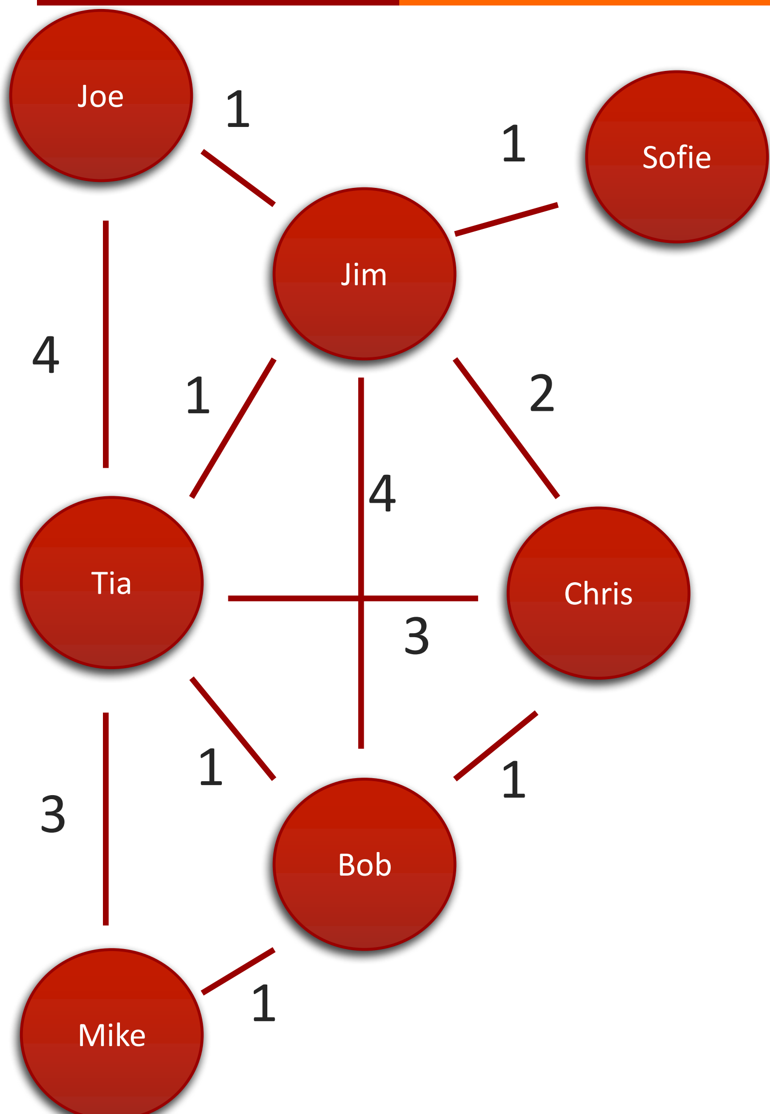
- Fan out from the initial node
- In the beginning the distances to the neighbors of the initial node are known. All other nodes are tentatively infinite distance away.
- The algorithm improves the estimates to the other nodes step by step.
- As you fan out, perform the operation illustrated in this example: if the current node A is marked with a distance of 4, and the edge connecting it with a neighbor B has length 2, then the distance to B (through A) will be $4 + 2 = 6$. If B was previously marked with a distance greater than 6 then change it to 6. Otherwise, keep the current value.

Shortest path from Mike



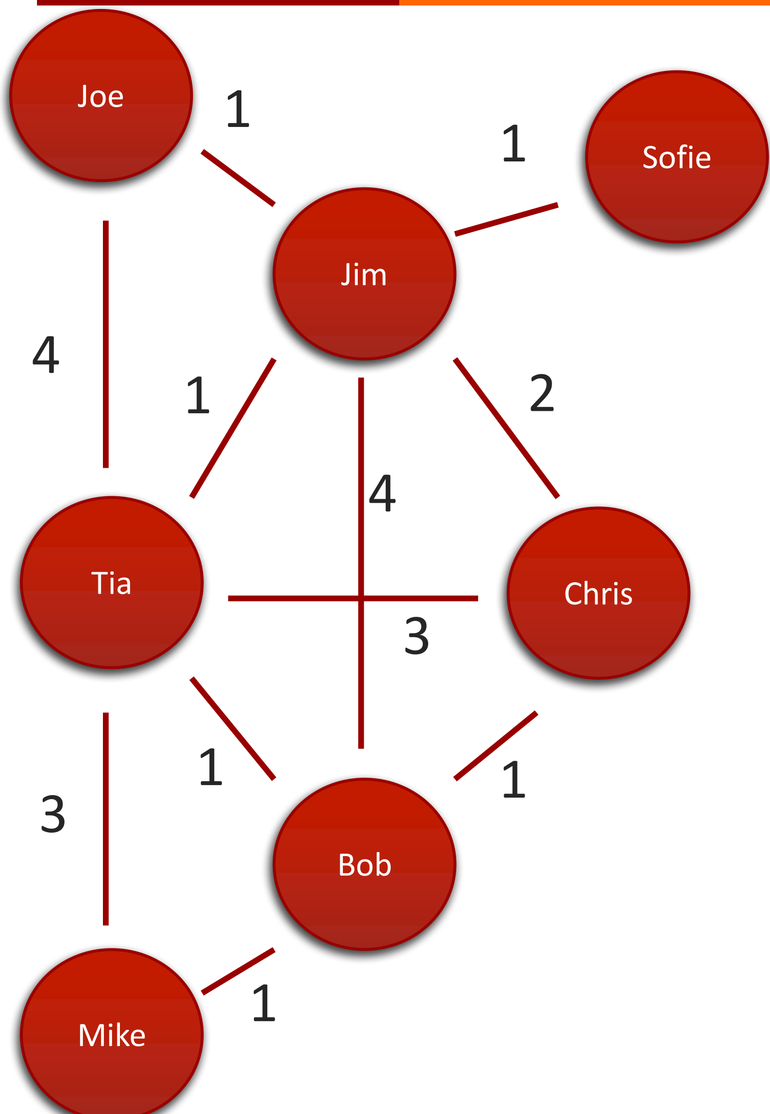
Mike	Bob	Tia	Jim	Chris	Sofie	Joe
0	1	3	∞	∞	∞	∞

Shortest path from Mike



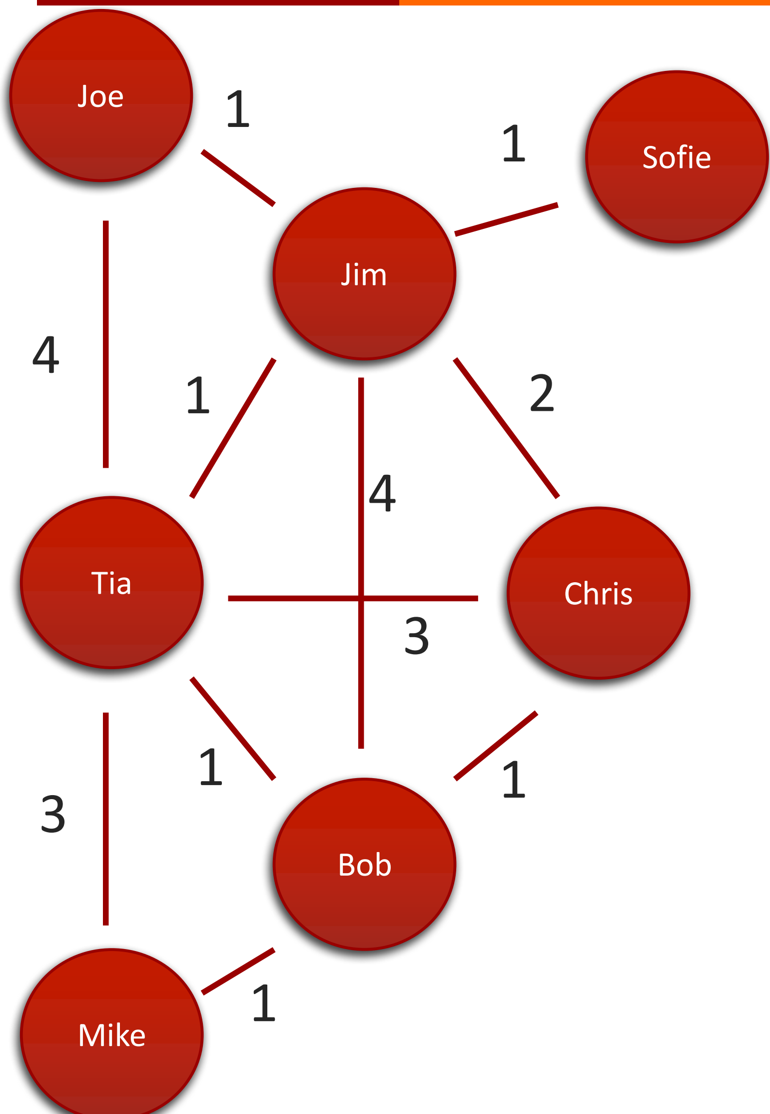
Mike	Bob	Tia	Jim	Chris	Sofie	Joe
0	1	3	∞	∞	∞	∞
0	1	2	5	2	∞	∞

Shortest path from Mike



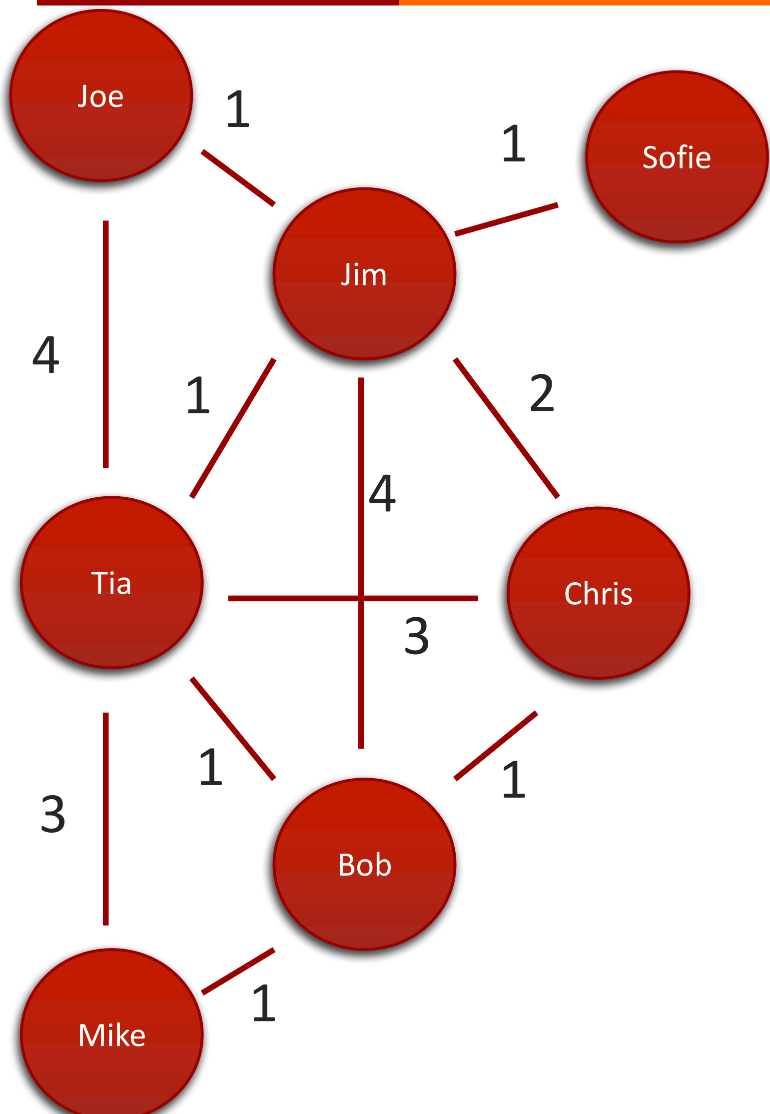
Mike	Bob	Tia	Jim	Chris	Sofie	Joe
0	1	3	∞	∞	∞	∞
0	1	2	5	2	∞	∞
0	1	2	4	2	∞	∞

Shortest path from Mike



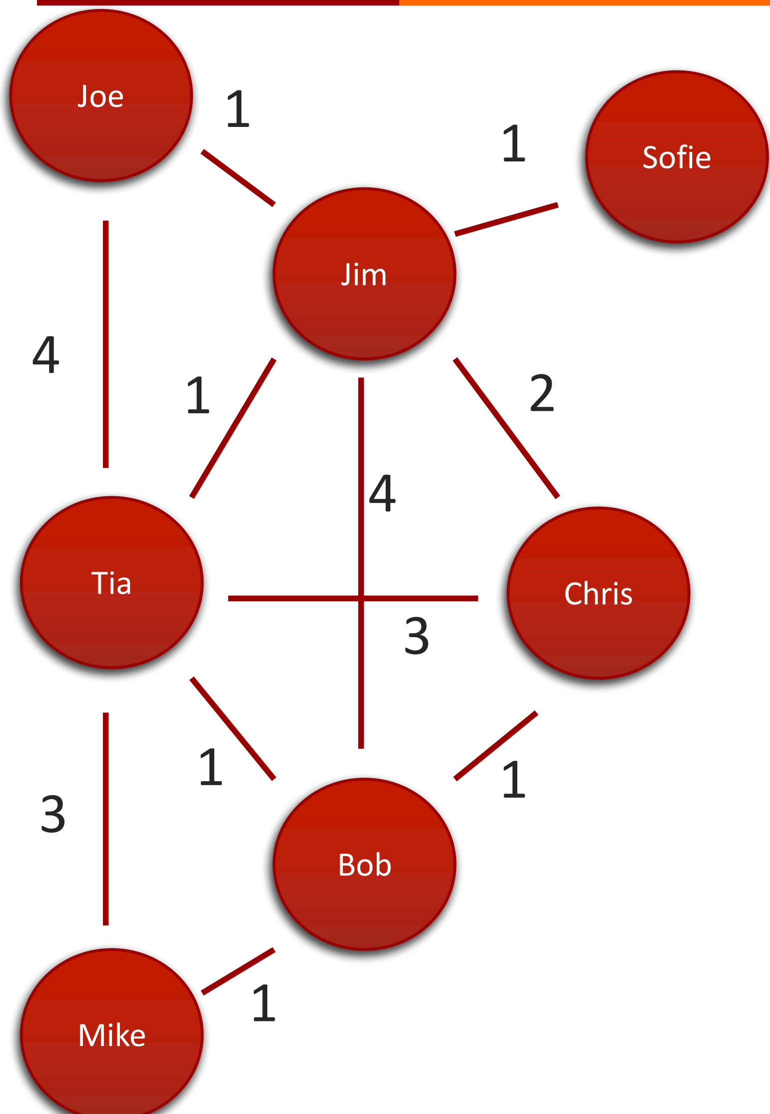
Mike	Bob	Tia	Jim	Chris	Sofie	Joe
0	1	3	∞	∞	∞	∞
0	1	2	5	2	∞	∞
0	1	2	4	2	∞	∞
0	1	2	3	2	∞	6

Shortest path from Mike



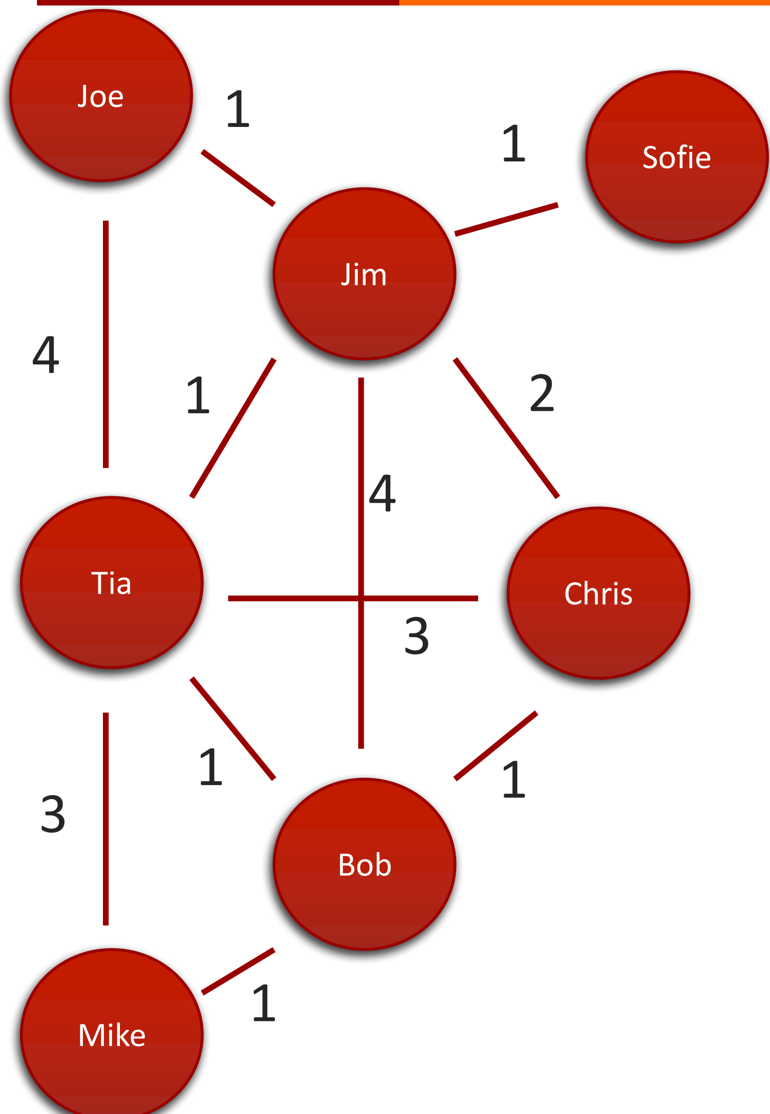
Mike	Bob	Tia	Jim	Chris	Sofie	Joe
0	1	3	∞	∞	∞	∞
0	1	2	5	2	∞	∞
0	1	2	4	2	∞	∞
0	1	2	3	2	∞	6
0	1	2	3	2	4	4

Shortest path from Mike



Mike	Bob	Tia	Jim	Chris	Sofie	Joe
0	1	3	∞	∞	∞	∞
0	1	2	5	2	∞	∞
0	1	2	4	2	∞	∞
0	1	2	3	2	∞	6
0	1	2	3	2	4	4
0	1	2	3	2	4	4

Shortest path from Mike



Mike	Bob	Tia	Jim	Chris	Sofie	Joe
0	1	3	∞	∞	∞	∞
0	1	2	5	2	∞	∞
0	1	2	4	2	∞	∞
0	1	2	3	2	∞	6
0	1	2	3	2	4	4
0	1	2	3	2	4	4
0	1	2	3	2	4	4