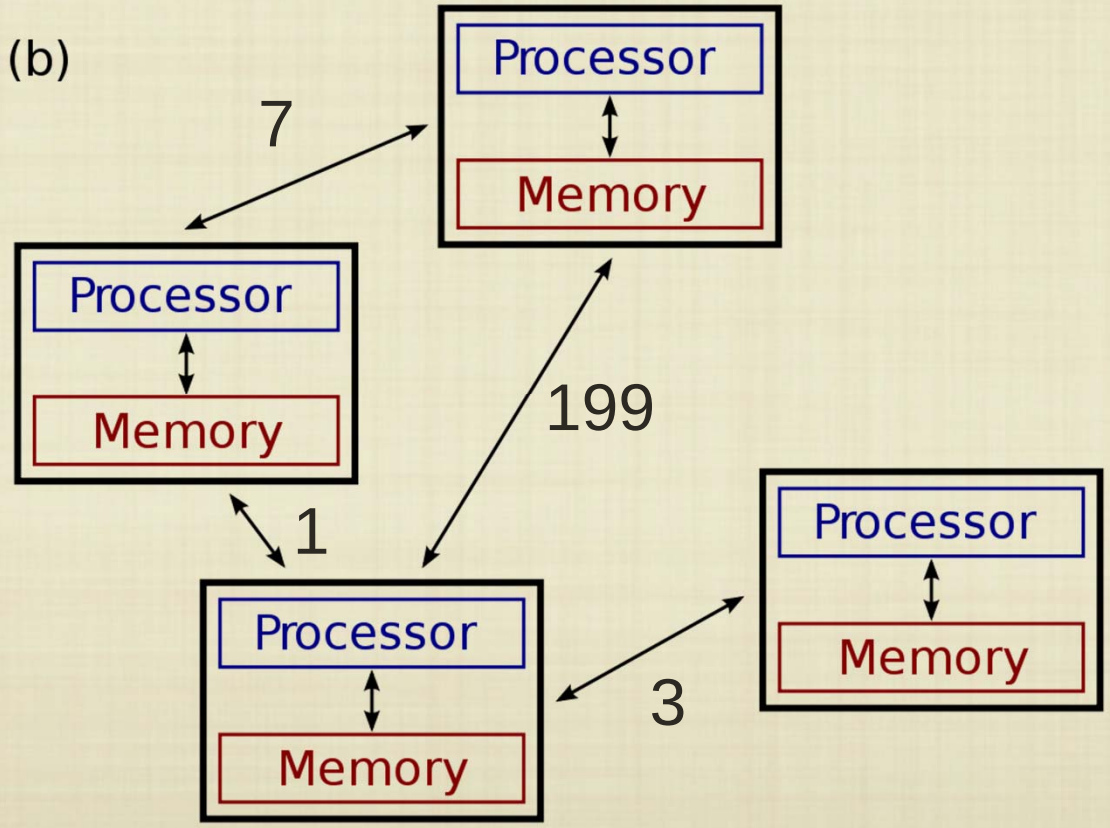
Introduction To Graphs and Networks

Fall 2013

Carola Wenk

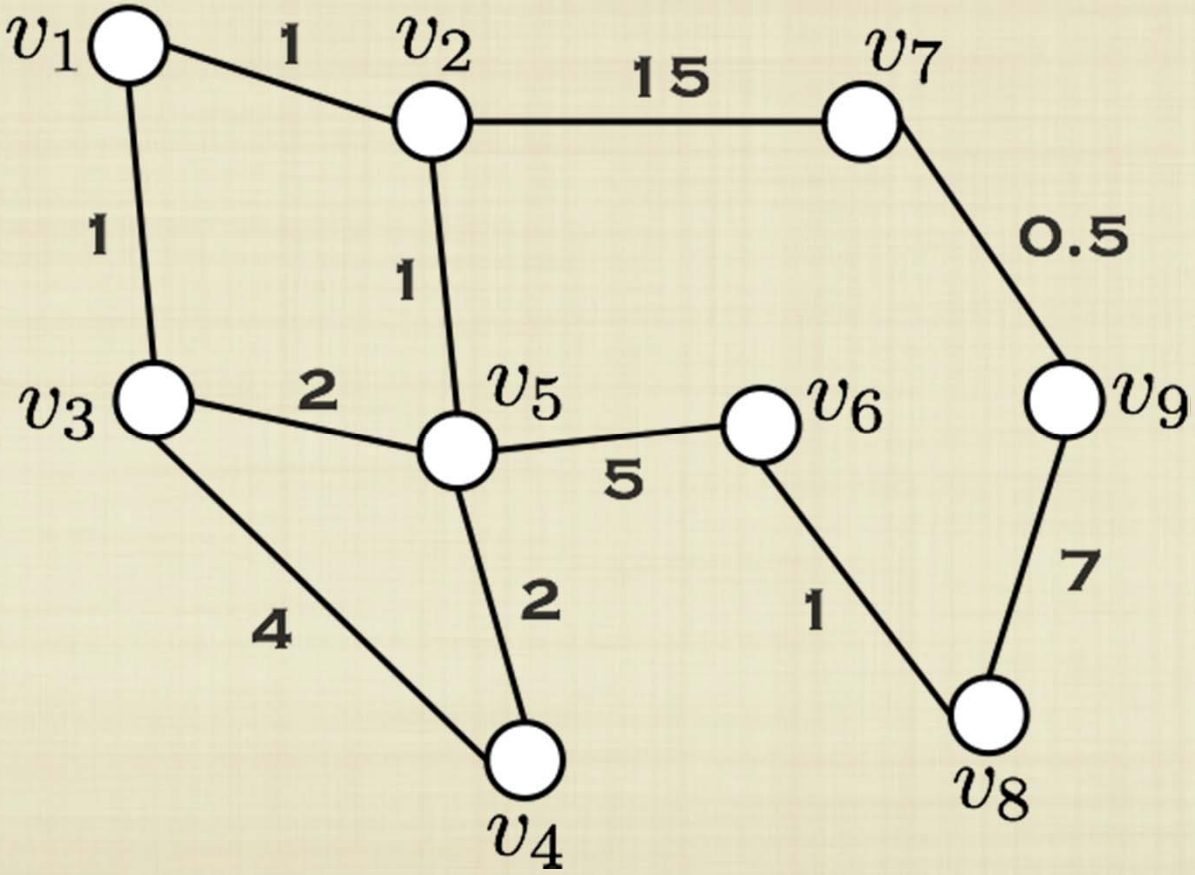
Real-World Routing

- On the Internet, links are essentially “weighted” by factors such as transit time, or cost. The goal is to find the shortest path from one node to another.



Real-World Routing

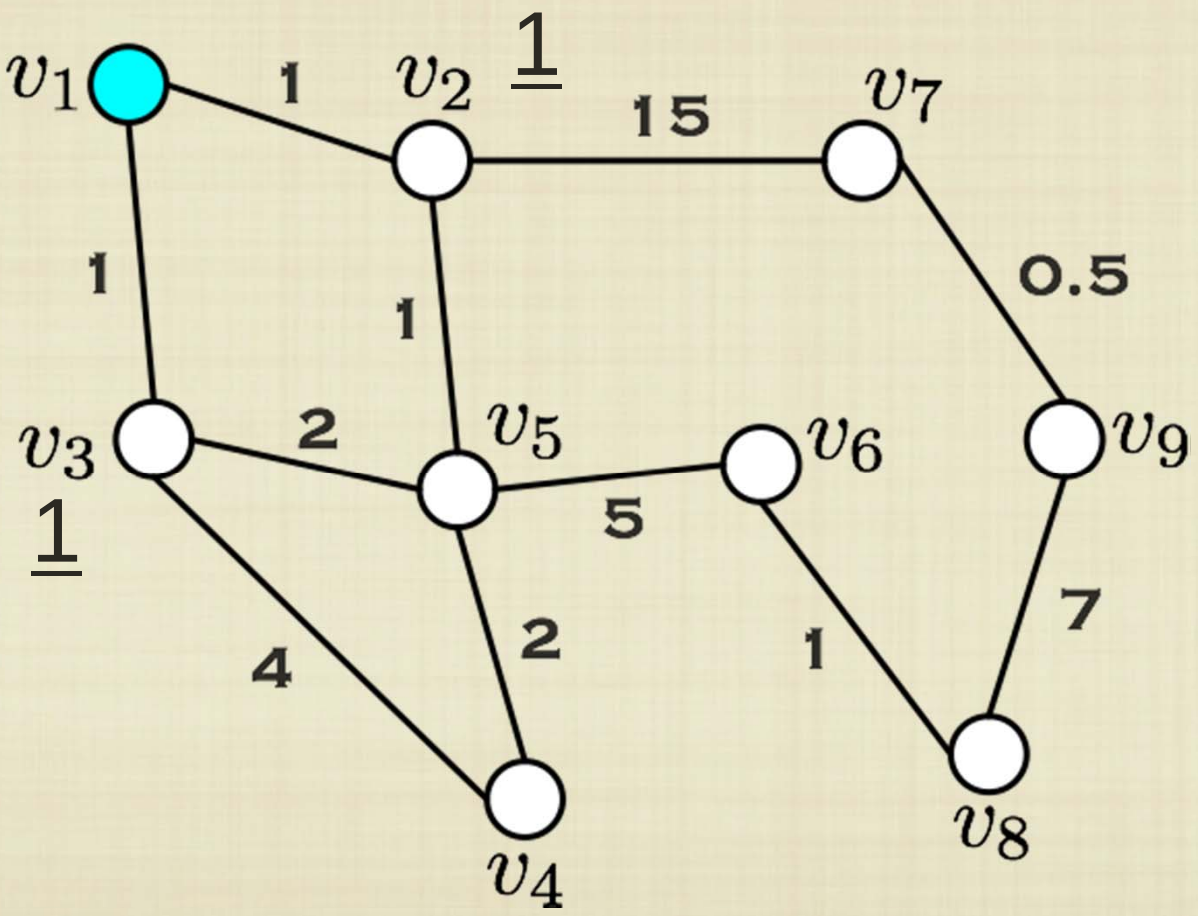
- To solve the routing problem, we must deal with weighted edges:



Dijkstra's algorithm, which is similar to BFS, can be used to find shortest paths from a source.

Real-World Routing

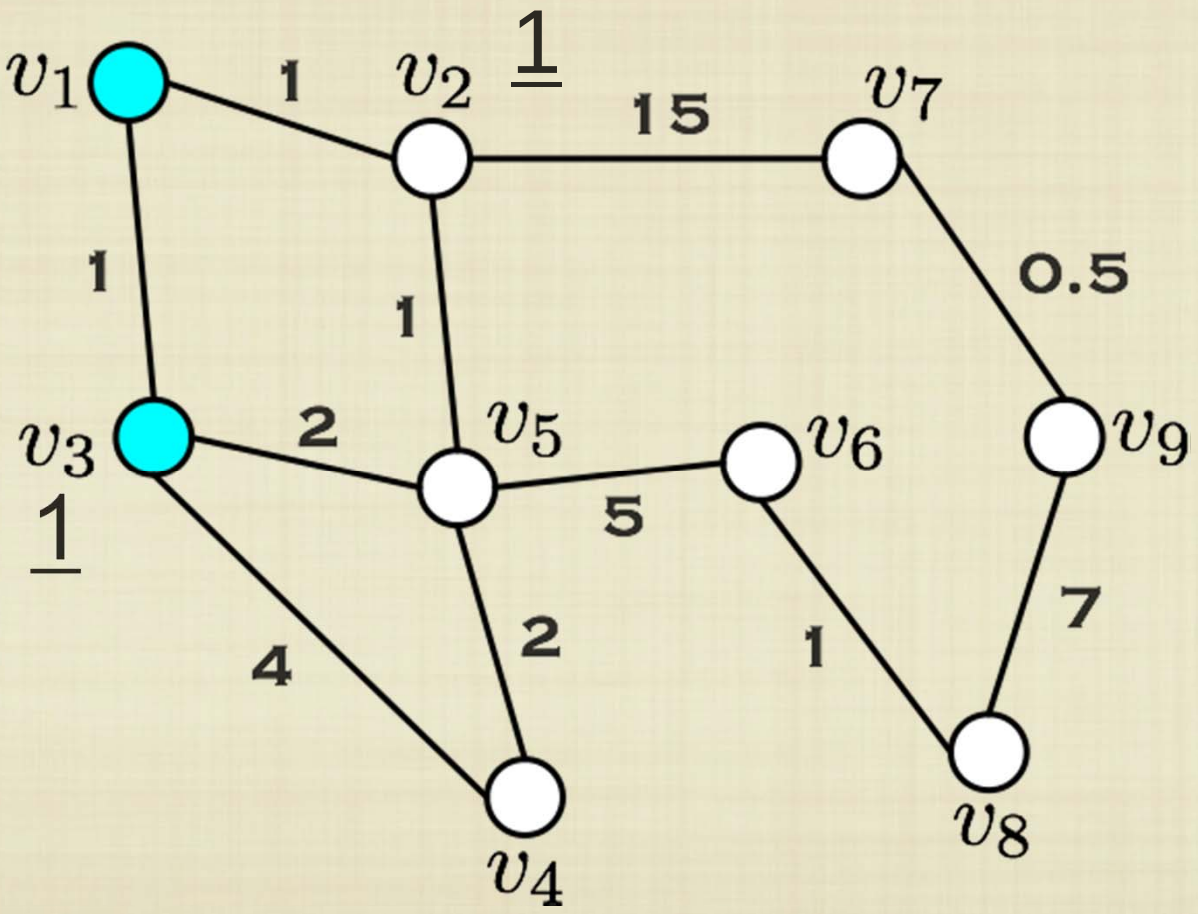
To solve the routing problem, we must deal with weighted edges:



Dijkstra's algorithm, which is similar to BFS, can be used to find shortest paths from a source.

Real-World Routing

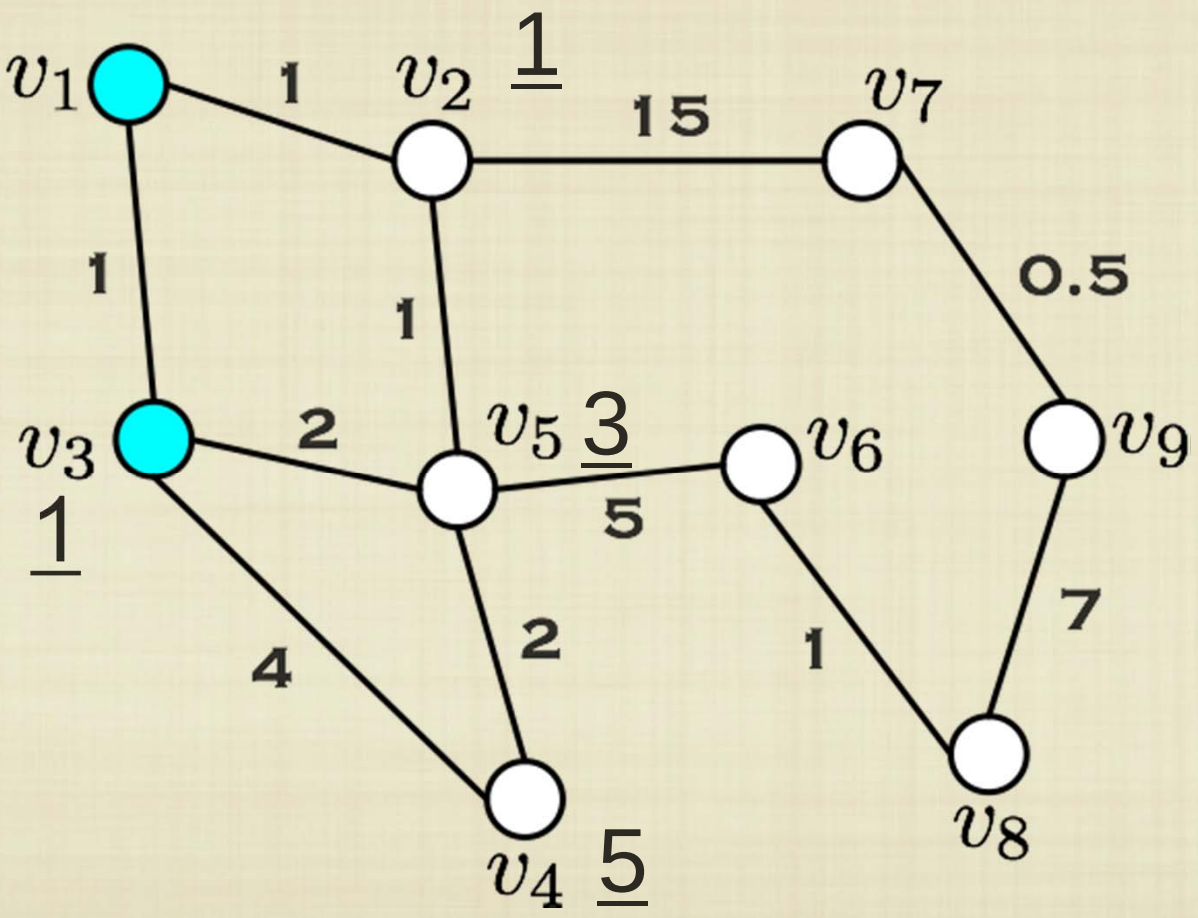
To solve the routing problem, we must deal with weighted edges:



Dijkstra's algorithm, which is similar to BFS, can be used to find shortest paths from a source.

Real-World Routing

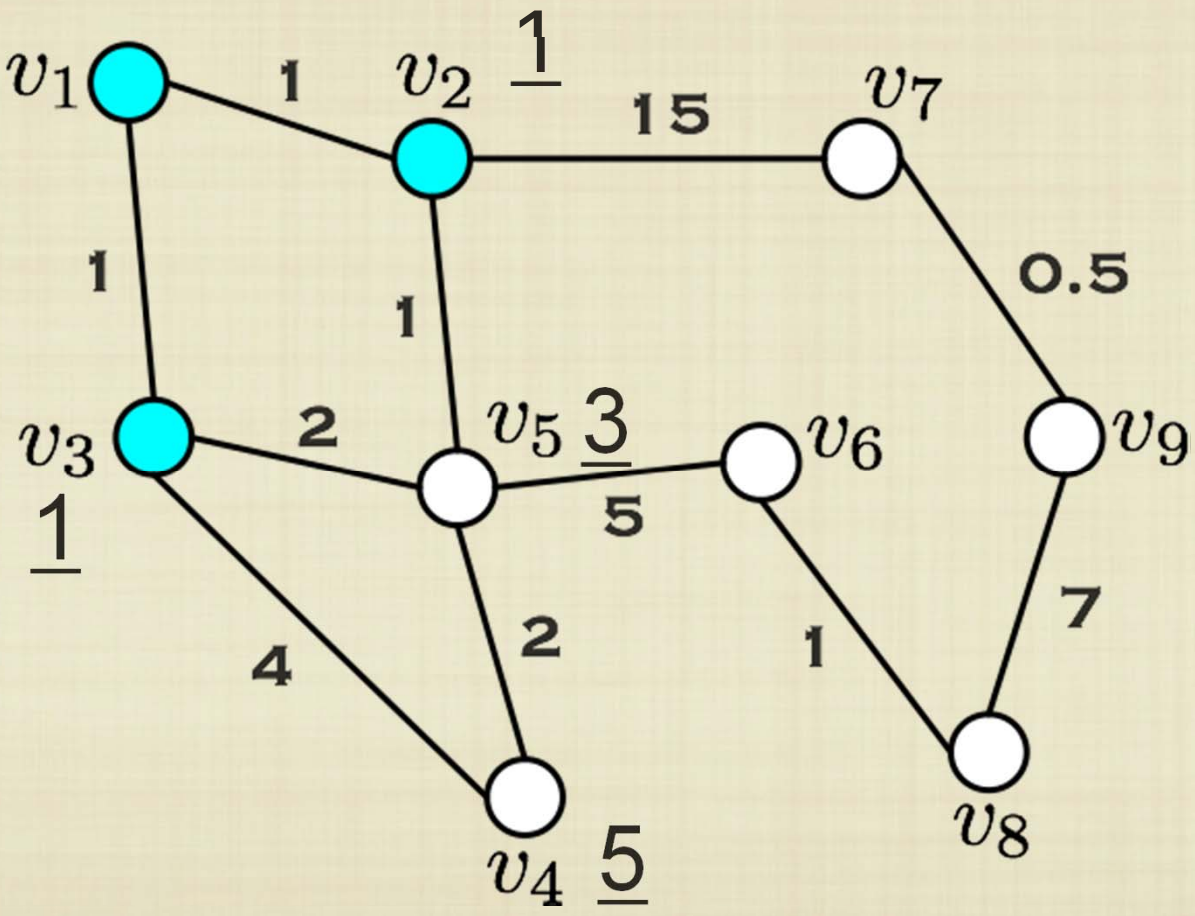
To solve the routing problem, we must deal with weighted edges:



Dijkstra's algorithm, which is similar to BFS, can be used to find shortest paths from a source.

Real-World Routing

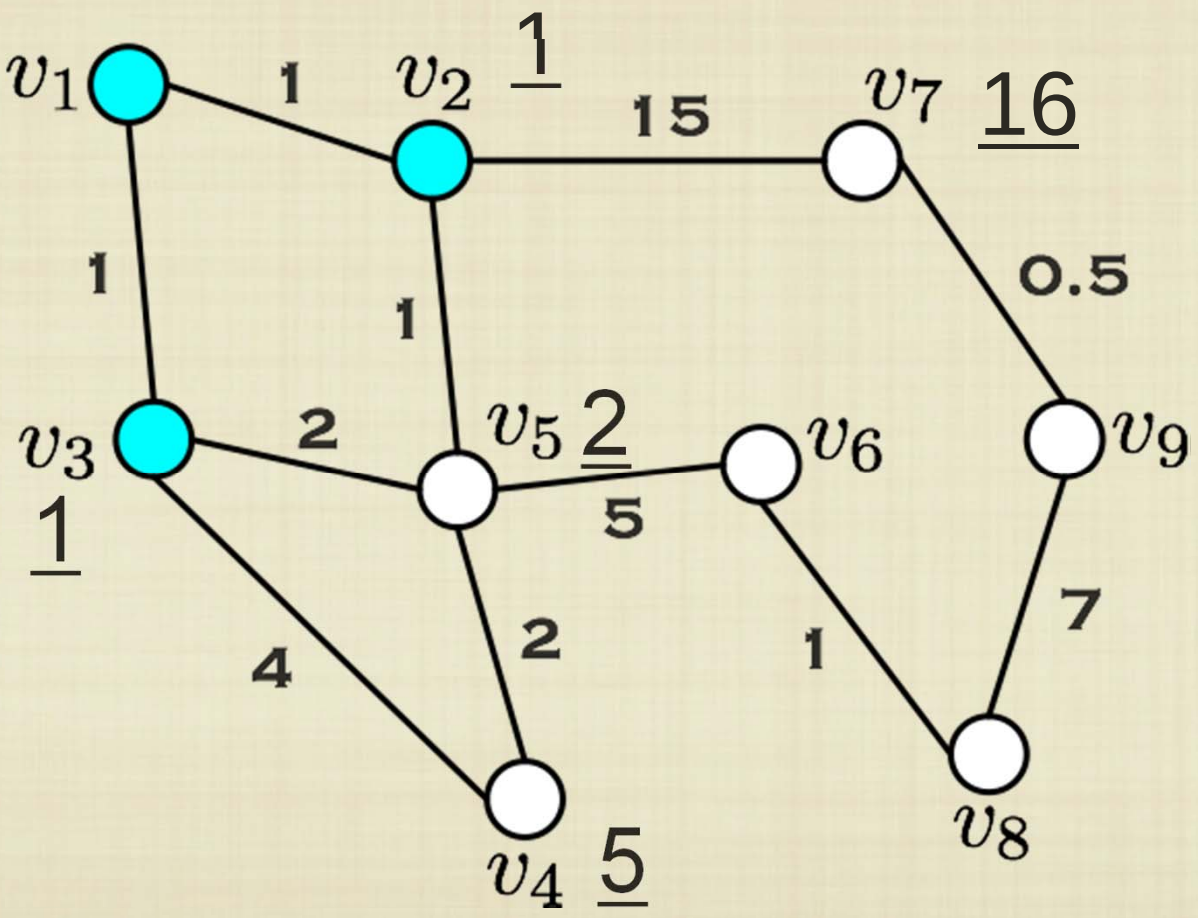
To solve the routing problem, we must deal with weighted edges:



Dijkstra's algorithm, which is similar to BFS, can be used to find shortest paths from a source.

Real-World Routing

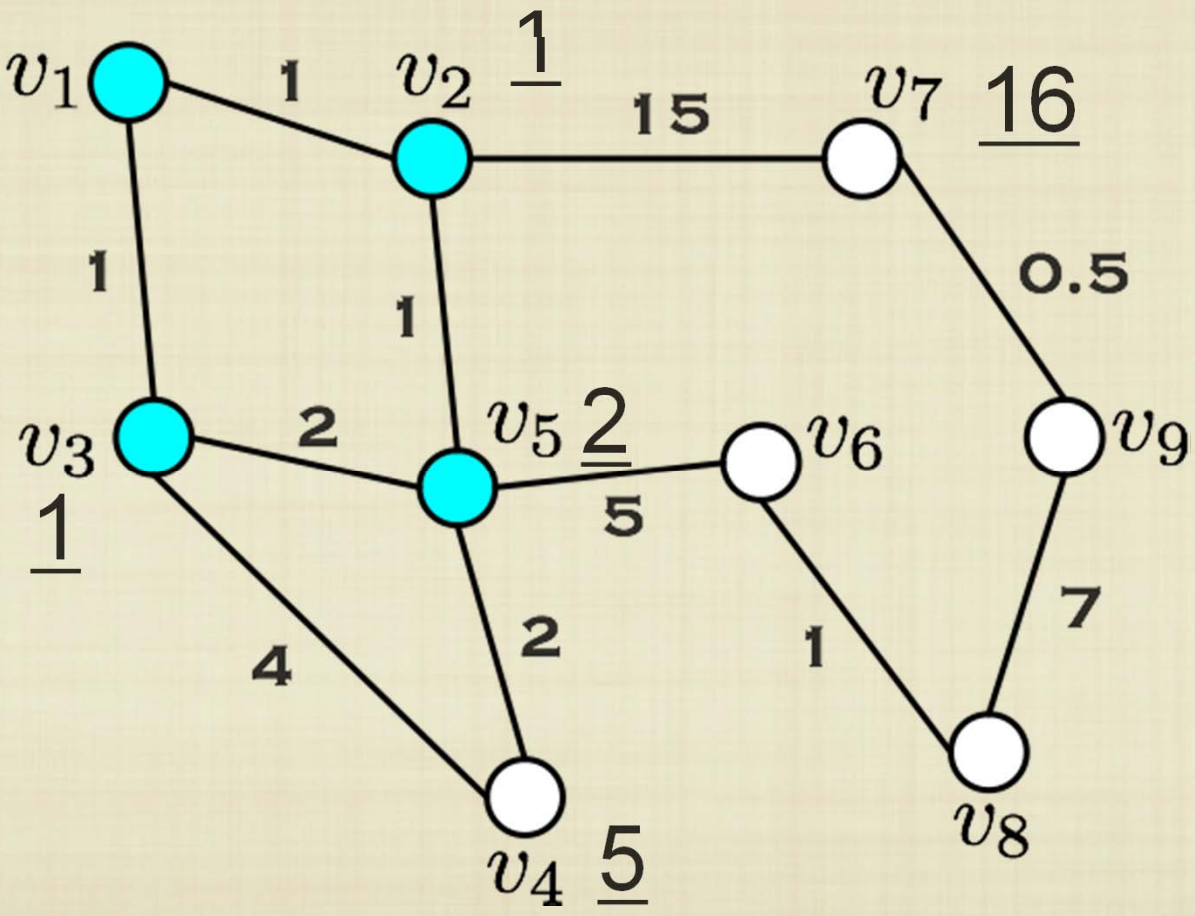
To solve the routing problem, we must deal with weighted edges:



Dijkstra's algorithm, which is similar to BFS, can be used to find shortest paths from a source.

Real-World Routing

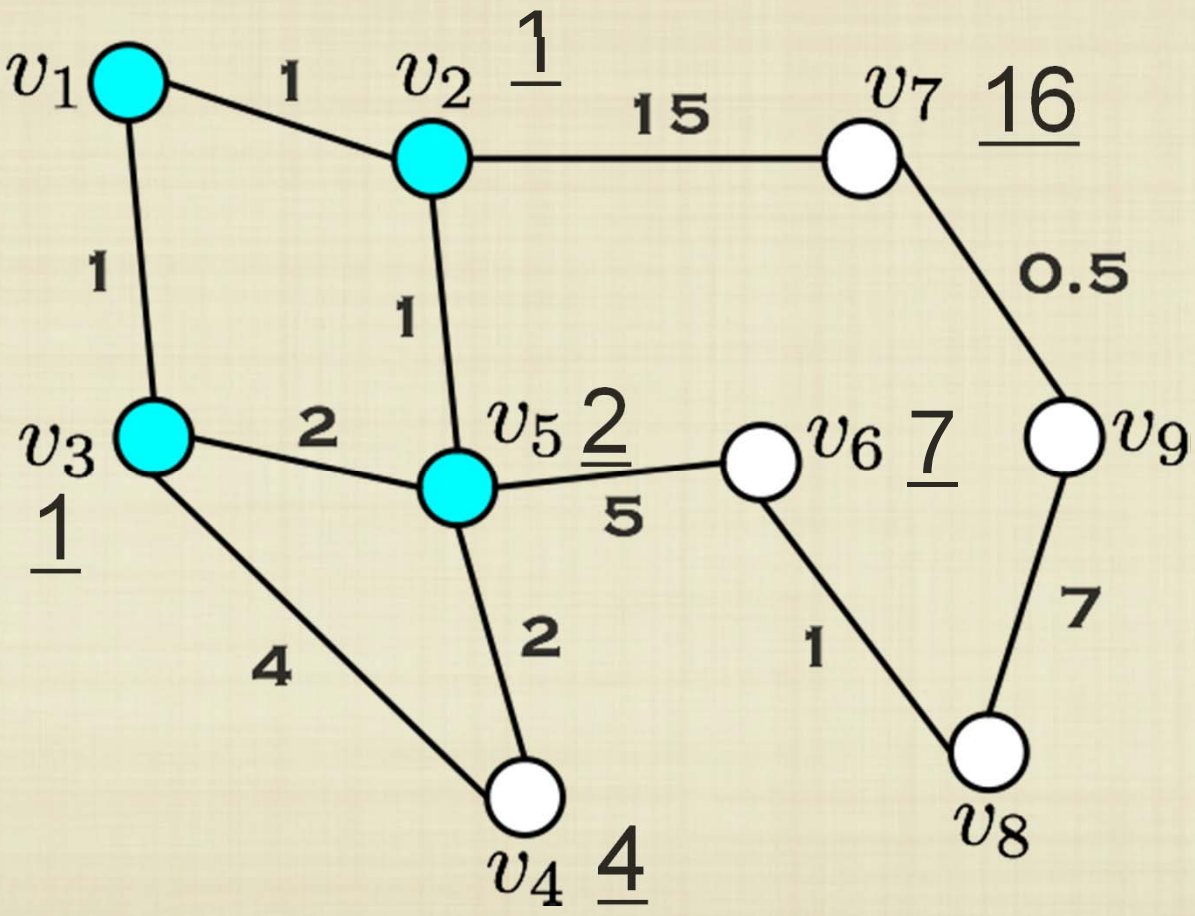
To solve the routing problem, we must deal with weighted edges:



Dijkstra's algorithm, which is similar to BFS, can be used to find shortest paths from a source.

Real-World Routing

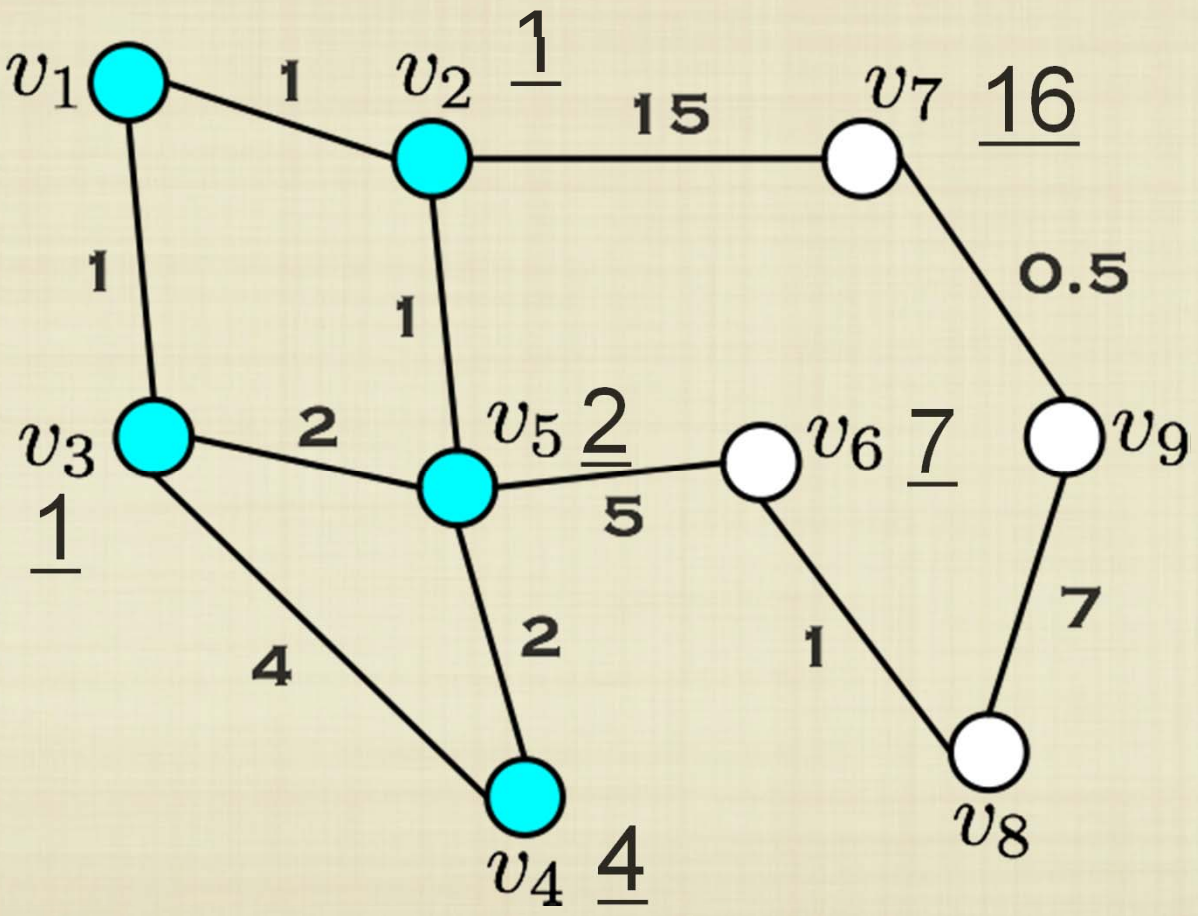
To solve the routing problem, we must deal with weighted edges:



Dijkstra's algorithm, which is similar to BFS, can be used to find shortest paths from a source.

Real-World Routing

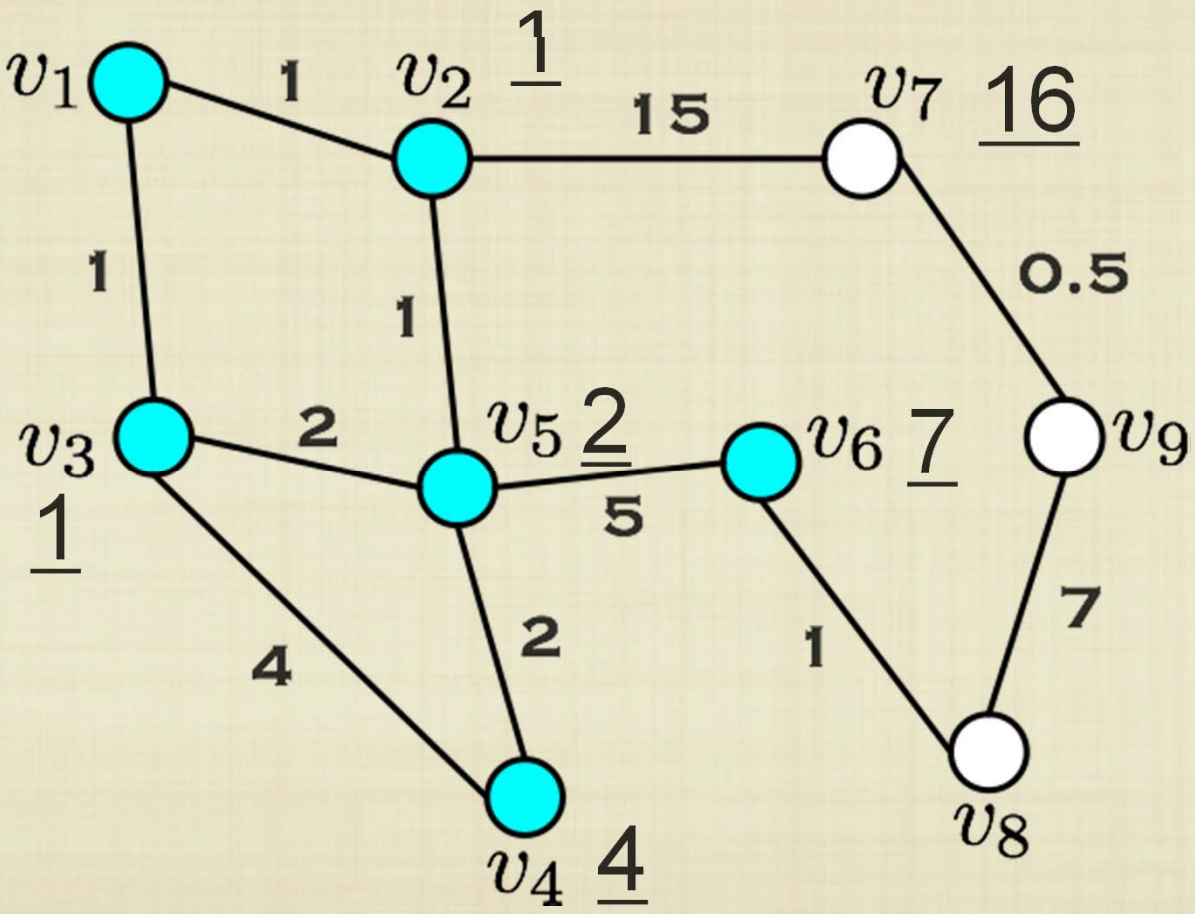
To solve the routing problem, we must deal with weighted edges:



Dijkstra's algorithm, which is similar to BFS, can be used to find shortest paths from a source.

Real-World Routing

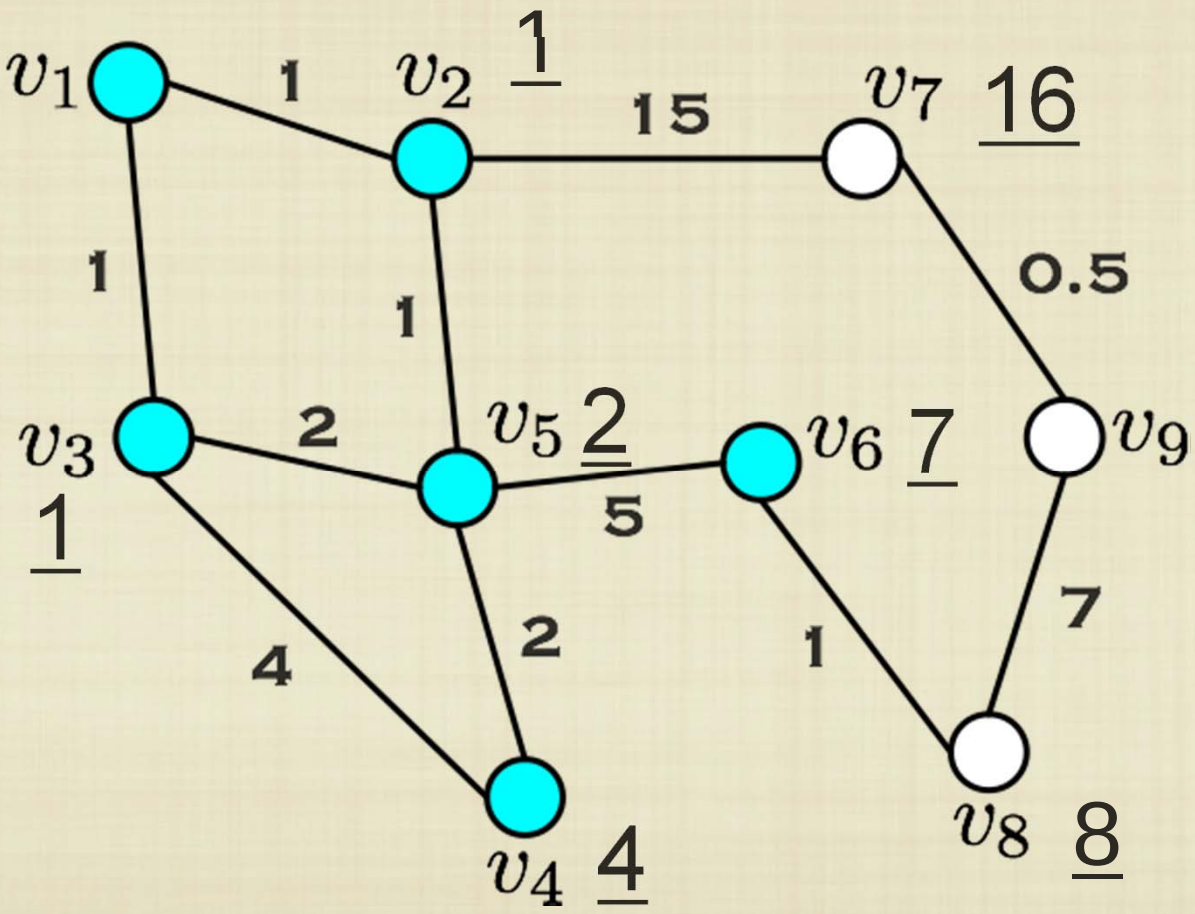
To solve the routing problem, we must deal with weighted edges:



Dijkstra's algorithm, which is similar to BFS, can be used to find shortest paths from a source.

Real-World Routing

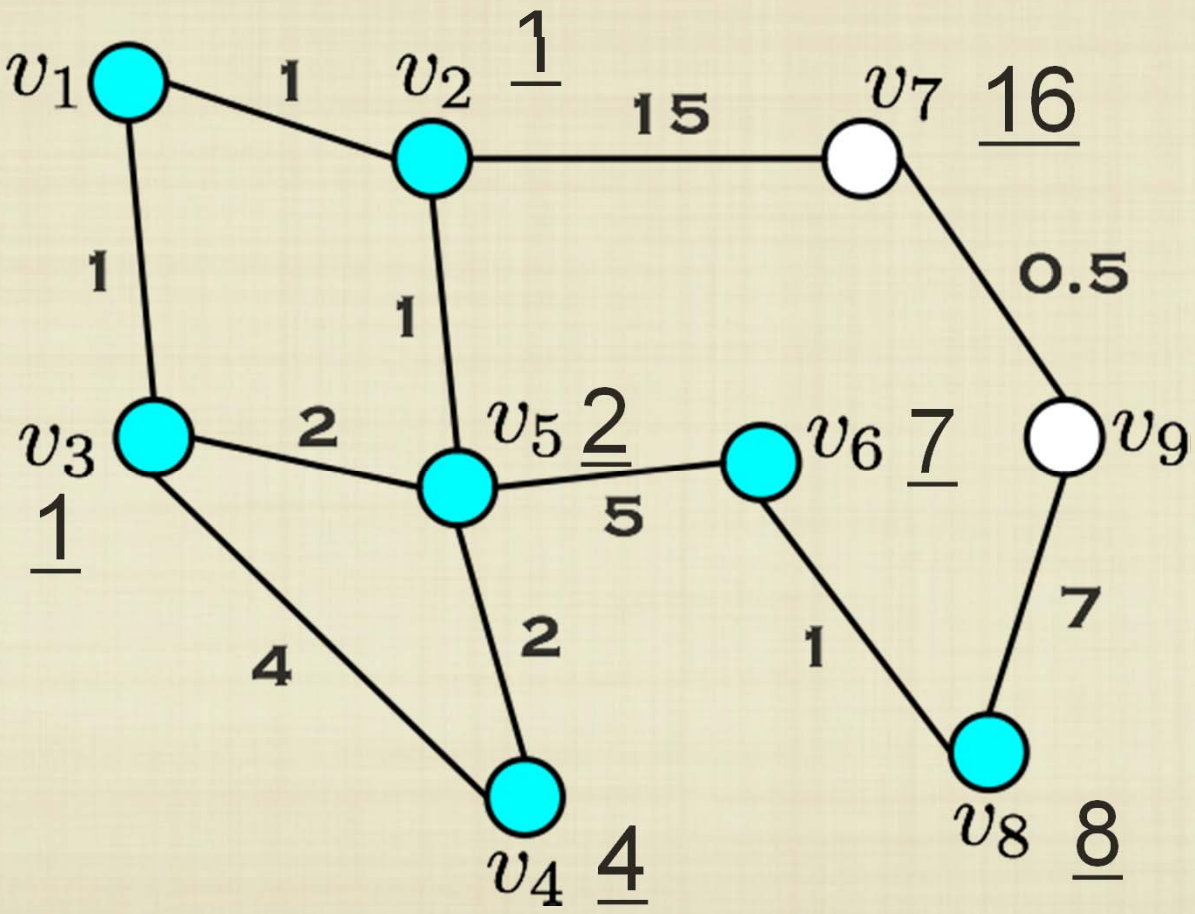
To solve the routing problem, we must deal with weighted edges:



Dijkstra's algorithm, which is similar to BFS, can be used to find shortest paths from a source.

Real-World Routing

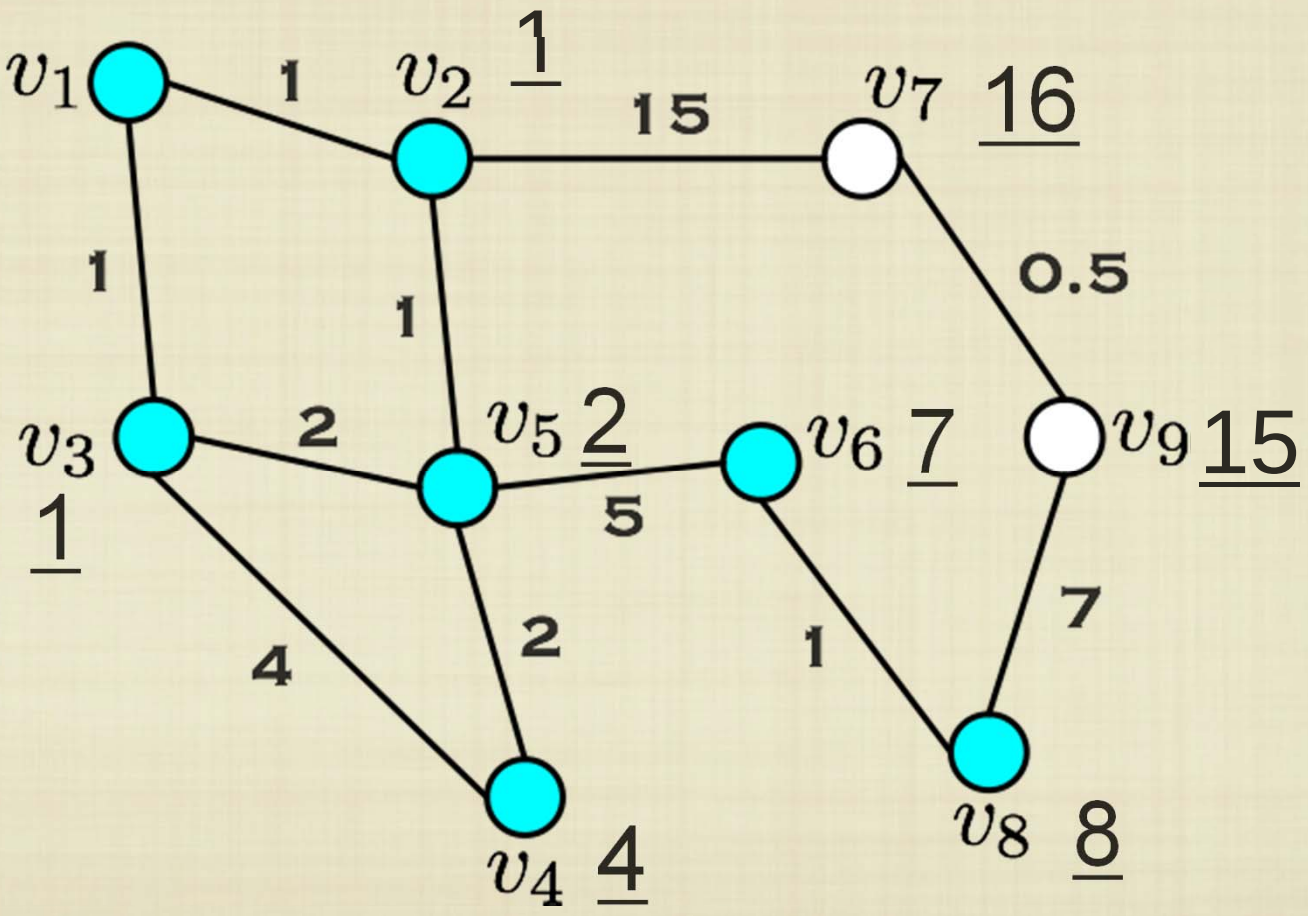
To solve the routing problem, we must deal with weighted edges:



Dijkstra's algorithm, which is similar to BFS, can be used to find shortest paths from a source.

Real-World Routing

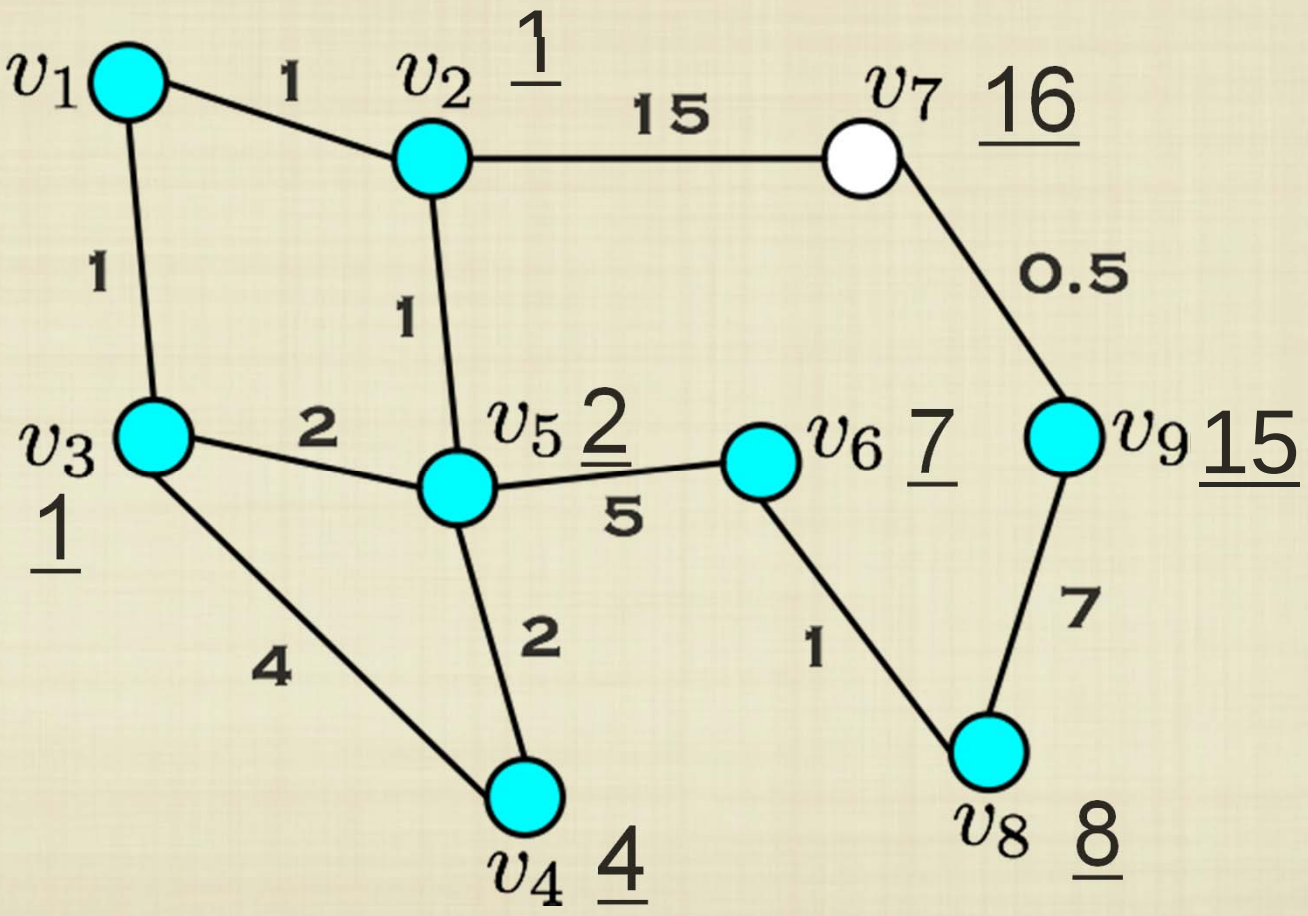
To solve the routing problem, we must deal with weighted edges:



Dijkstra's algorithm, which is similar to BFS, can be used to find shortest paths from a source.

Real-World Routing

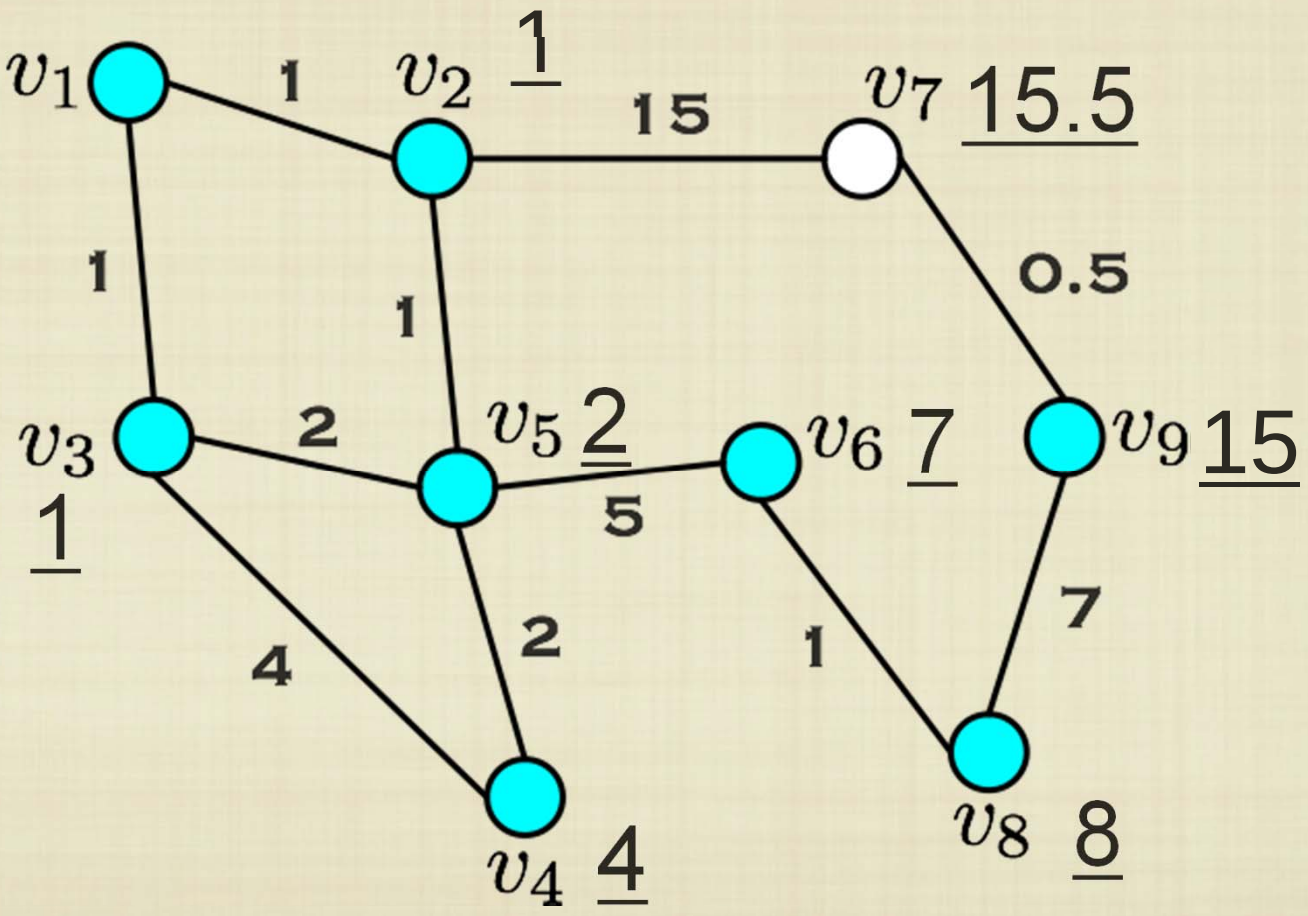
To solve the routing problem, we must deal with weighted edges:



Dijkstra's algorithm, which is similar to BFS, can be used to find shortest paths from a source.

Real-World Routing

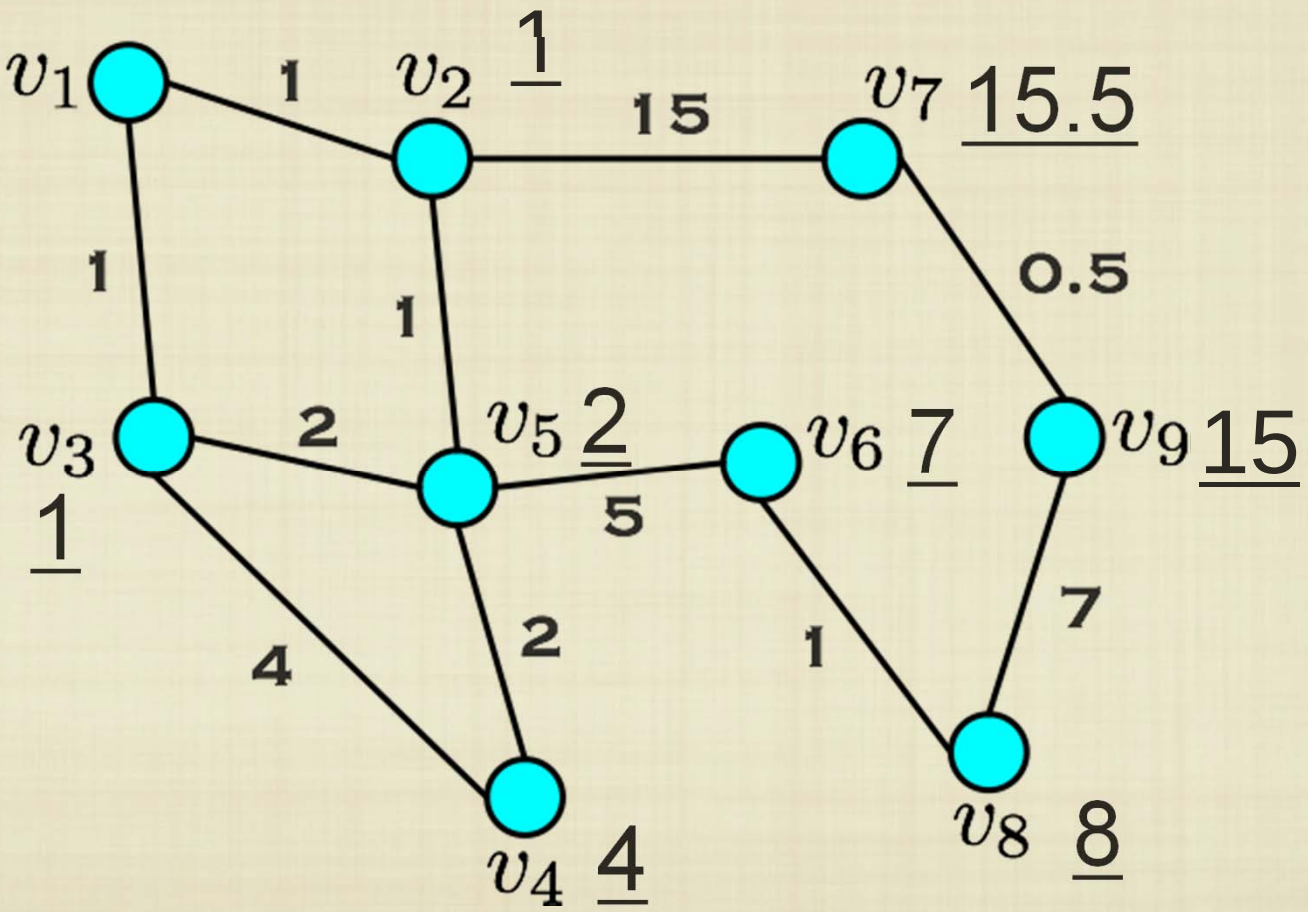
To solve the routing problem, we must deal with weighted edges:



Dijkstra's algorithm, which is similar to BFS, can be used to find shortest paths from a source.

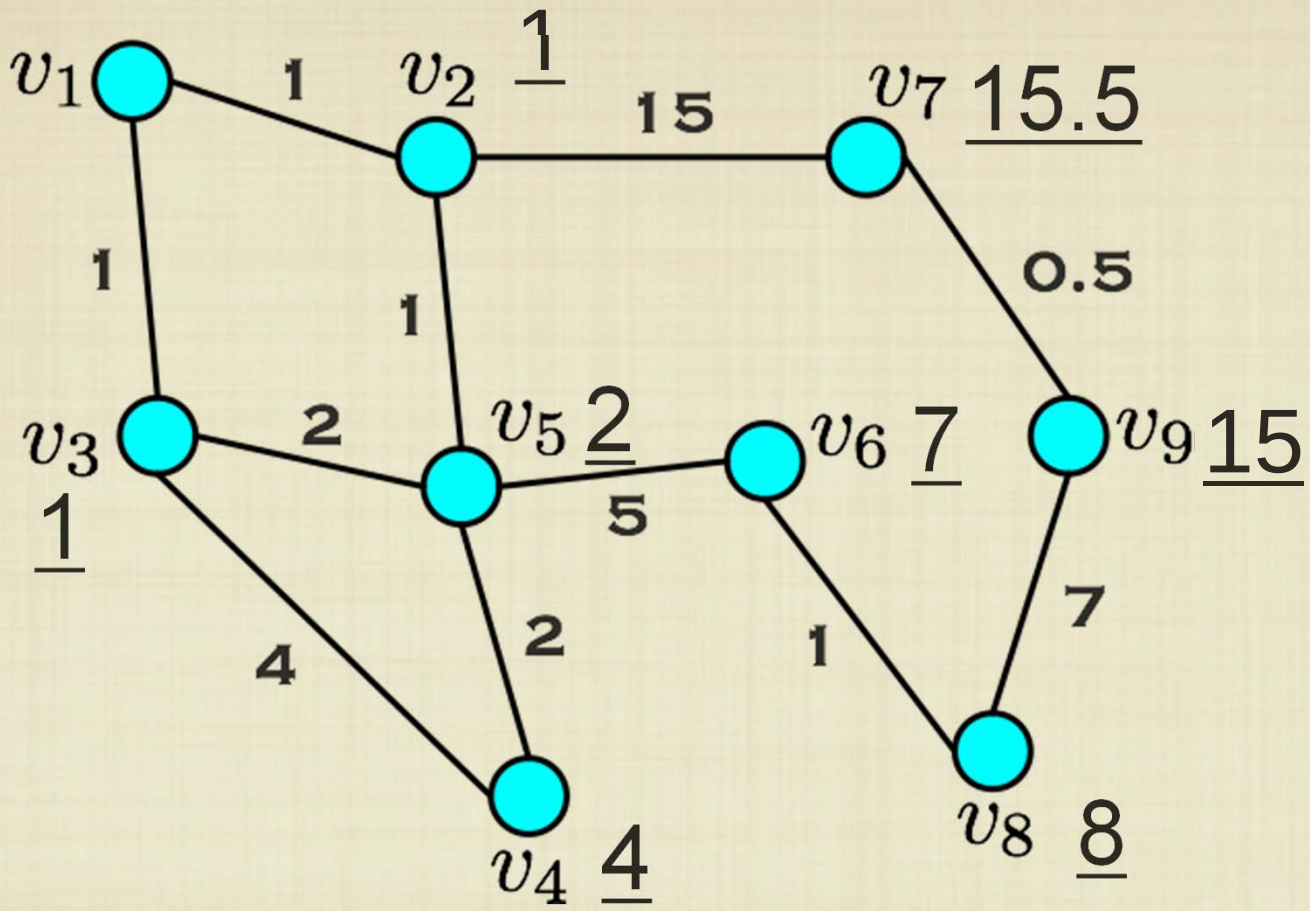
Real-World Routing

To solve the routing problem, we must deal with weighted edges:



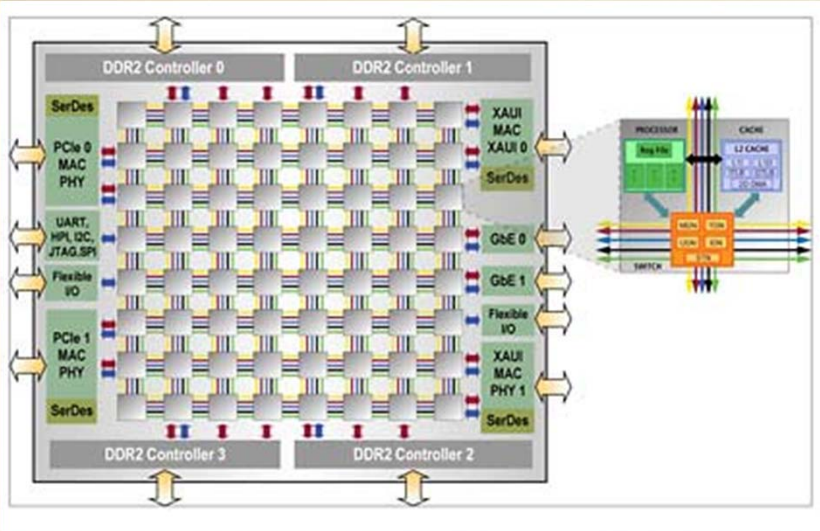
Dijkstra's algorithm is similar to BFS, but must update the distance estimate to any vertex as it progresses.

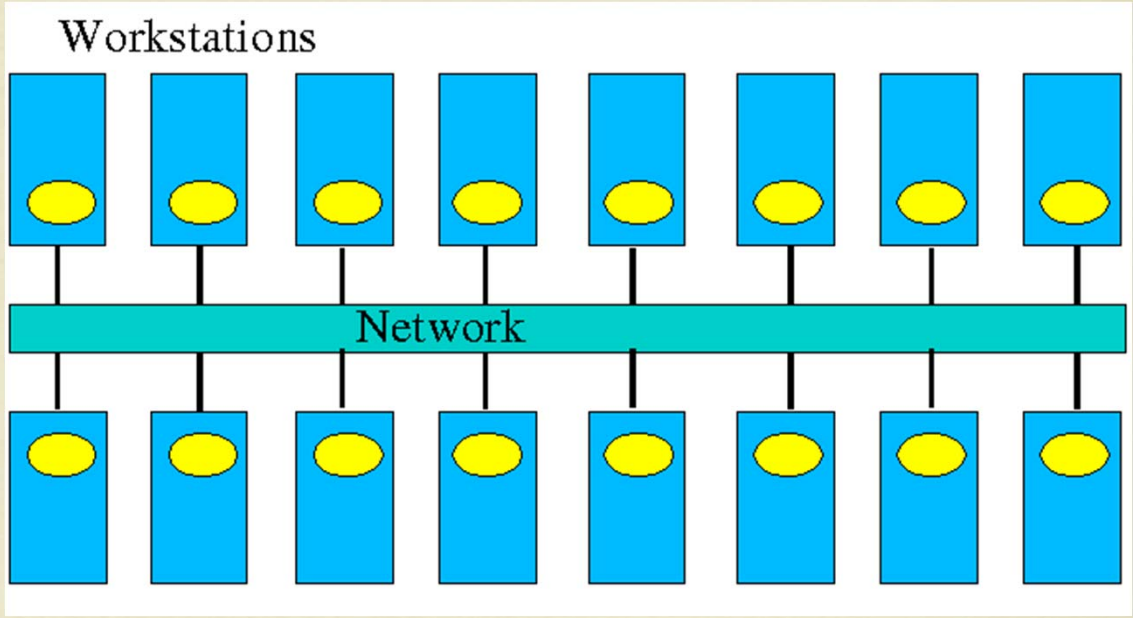
Real-World Routing



Each router computes a “distance-vector” of the shortest paths to a small part of the network, and this is passed along as part of packet communication. The last piece of the puzzle is dealing with changes to connectivity...

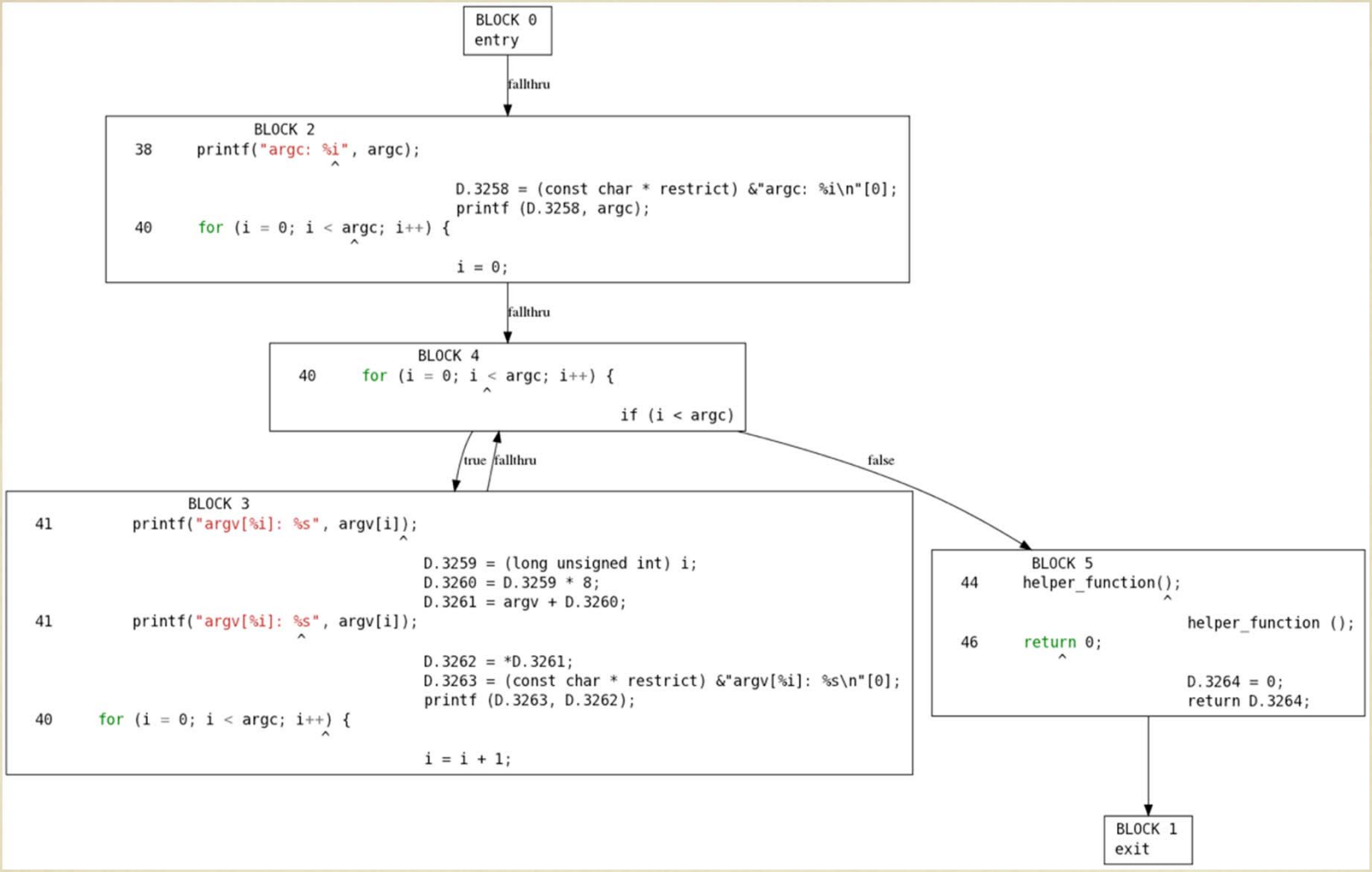
Graphs are Everywhere





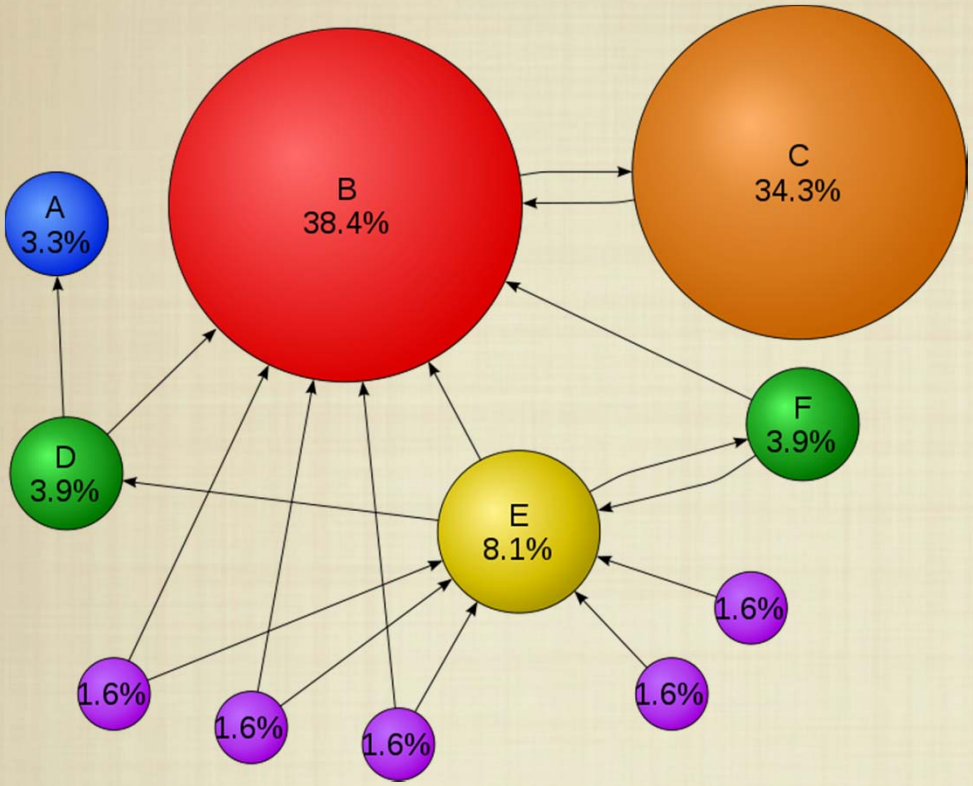
Computation across multiple computers, or cores, can be organized by analyzing the graph of interconnections. By optimizing how programs communicate, we can greatly improve the speed of concurrent computations.

Graphs are Everywhere



Every modern compiler creates a graph of “basic blocks” from a program to analyze it for error checking and optimization.

Graphs are Everywhere



Which webpages have the best information?

Which “friends” have the most relevant content?

“Content graphs” are created from web pages and on social networks. Google makes money by delivering search results relevant to your query. Facebook attracts users by providing the most relevant content generated in your social network.

Google PageRank

- The web graph is slightly different, links are not bidirectional. Web pages have both incoming links and outgoing links.

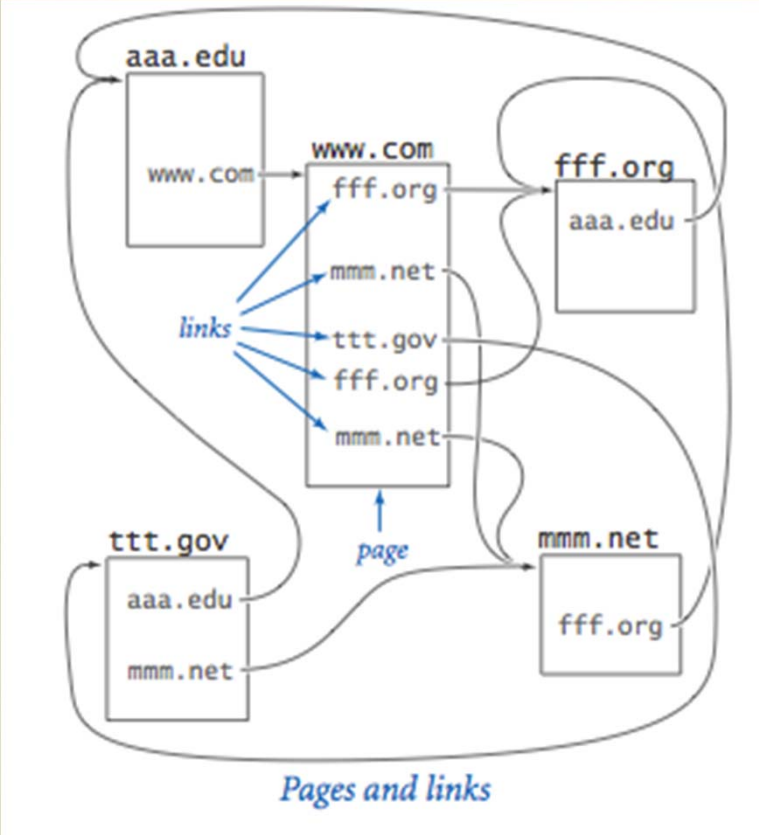
Given a particular search query, we may have tens of thousands of pages with those keywords. Which is the page that we most likely want?

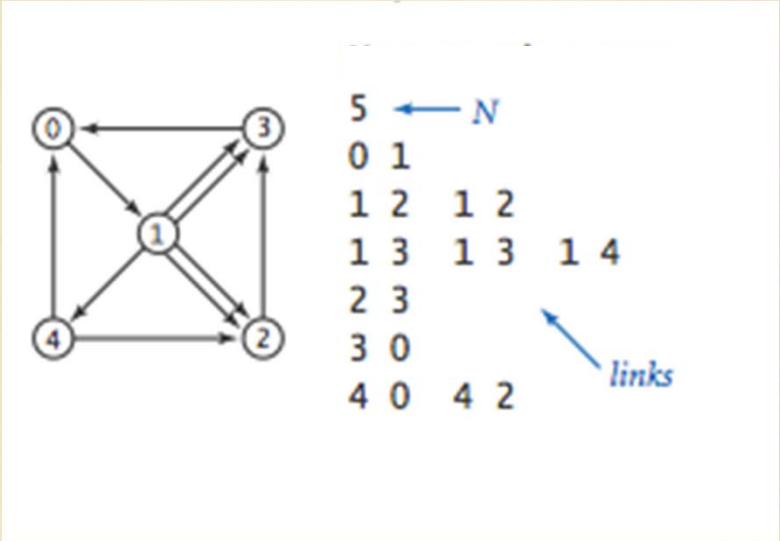
Google PageRank

- The web graph is slightly different, links are not bidirectional. Web pages have both incoming links and outgoing links.

Google “ranks” pages from a relative perspective: if we happened to be following links at random, where would we be likely to end up?

The Web Graph

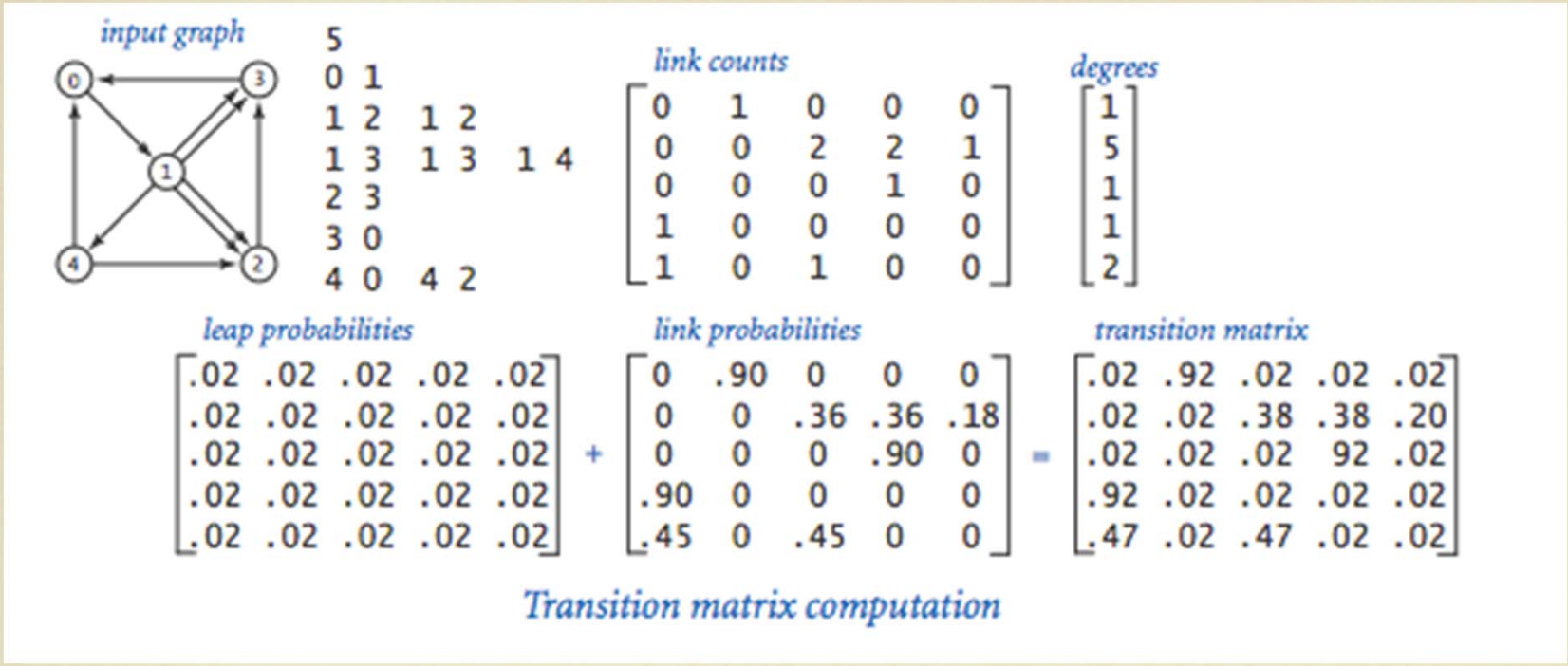




[Sedgewick/Wayne]

Hyperlinked web pages can be represented as a directed graph where we can have multiple links between pages.

Random Web Surfing

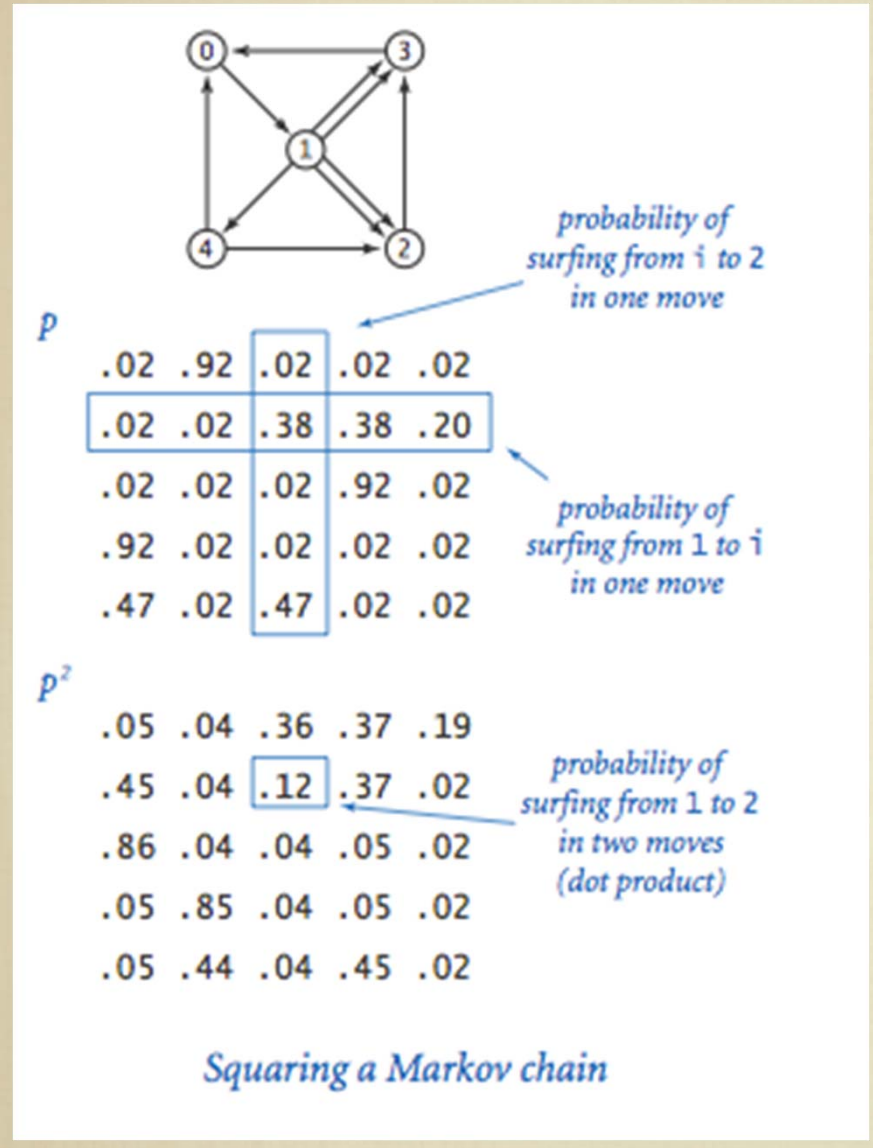


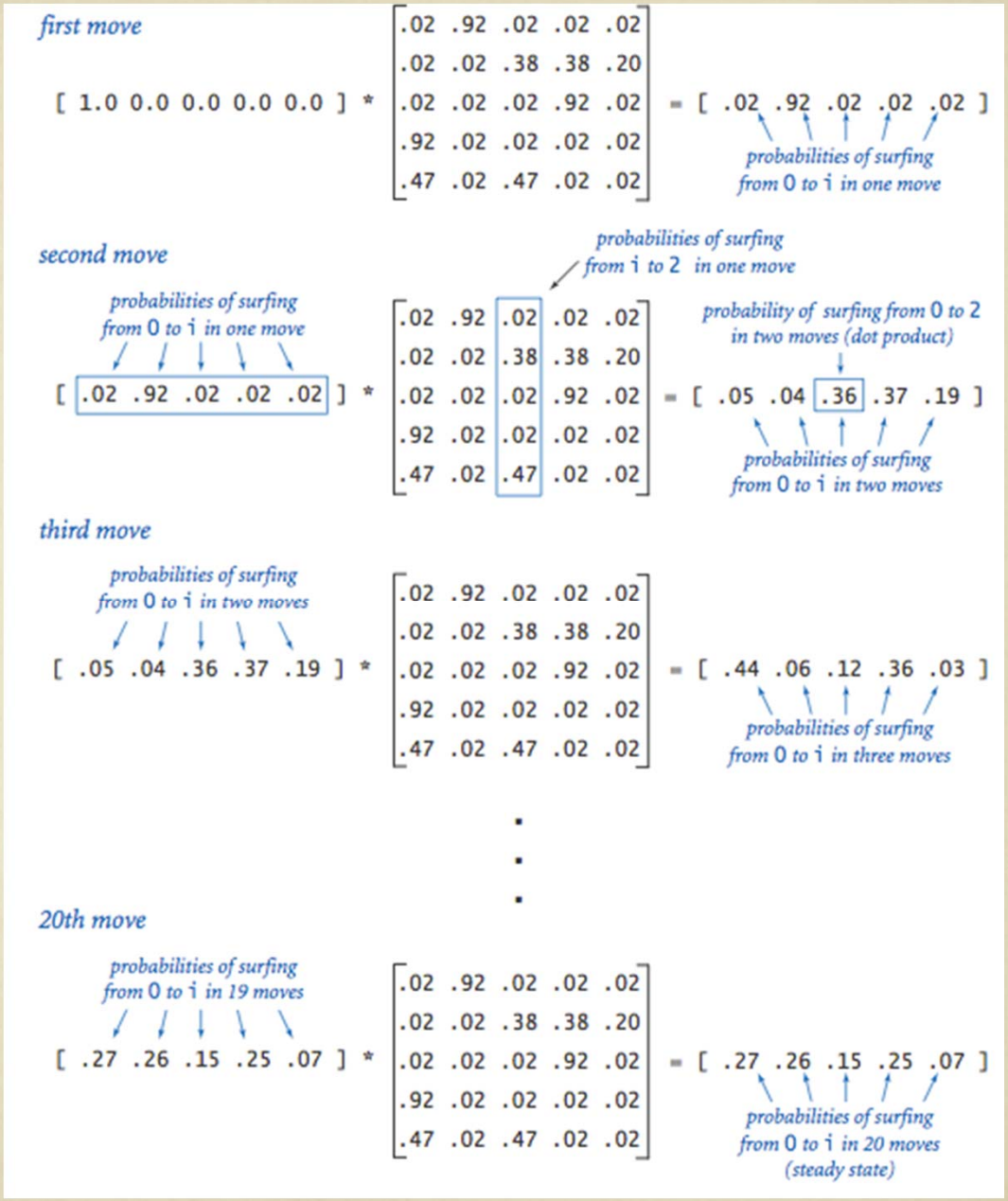
Transition matrix computation

[Sedgewick/Wayne]

The “random surfer model” postulates that we have a 90% chance of clicking on a random outgoing page, an 8% chance of going somewhere totally new, and a 2% chance of staying on the same page. Where will we eventually end up?

Random Web Surfing



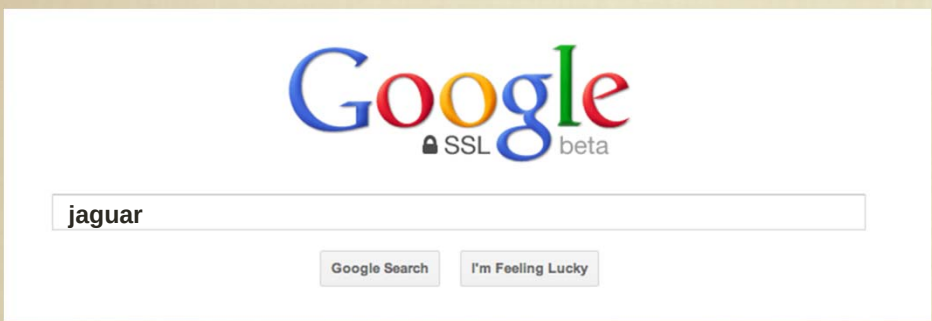


[Sedgewick/Wayne]

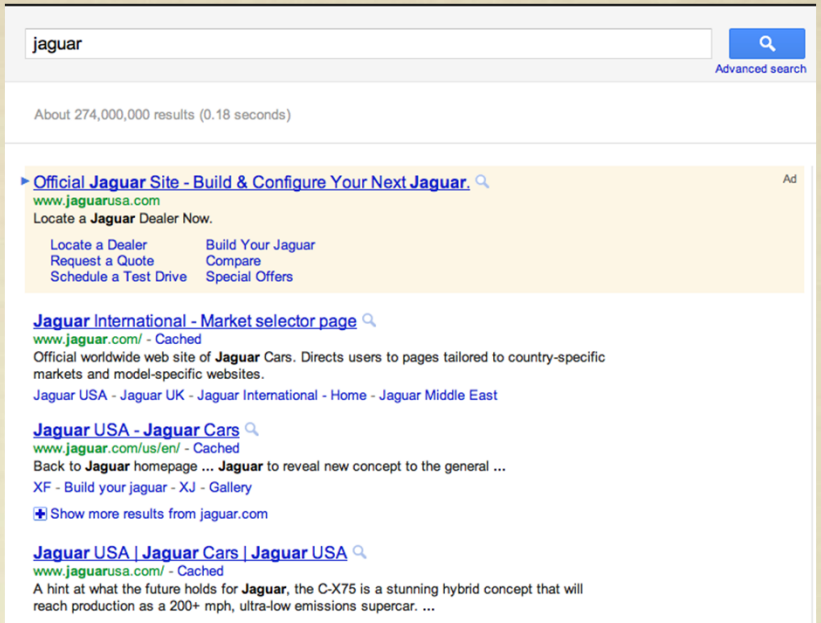
Google PageRank

- How do we calculate the location of our web surfer after an infinite number of clicks?
- We can repeatedly perform vector-matrix multiplication, until the probabilities of being on each page do not change.
- This is the original PageRank score, the current version has many many proprietary tweaks.
- Google computes PageRank offline, as it crawls the web for new pages.

Google in a Nutshell

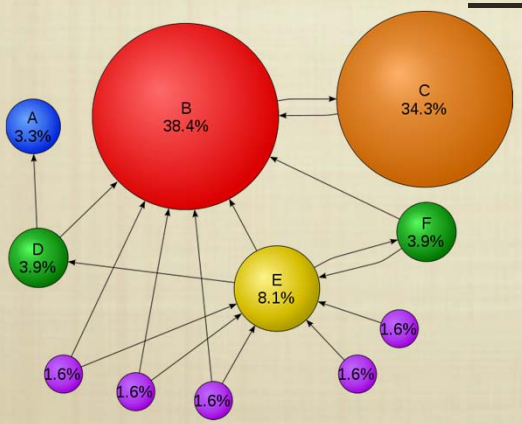


Query



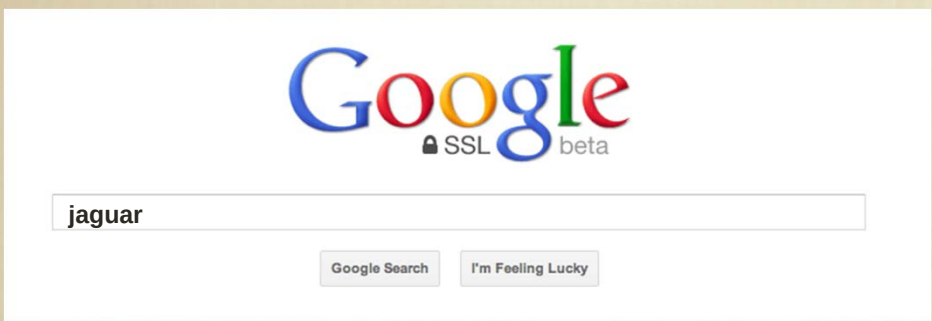
Result

Google Data Center

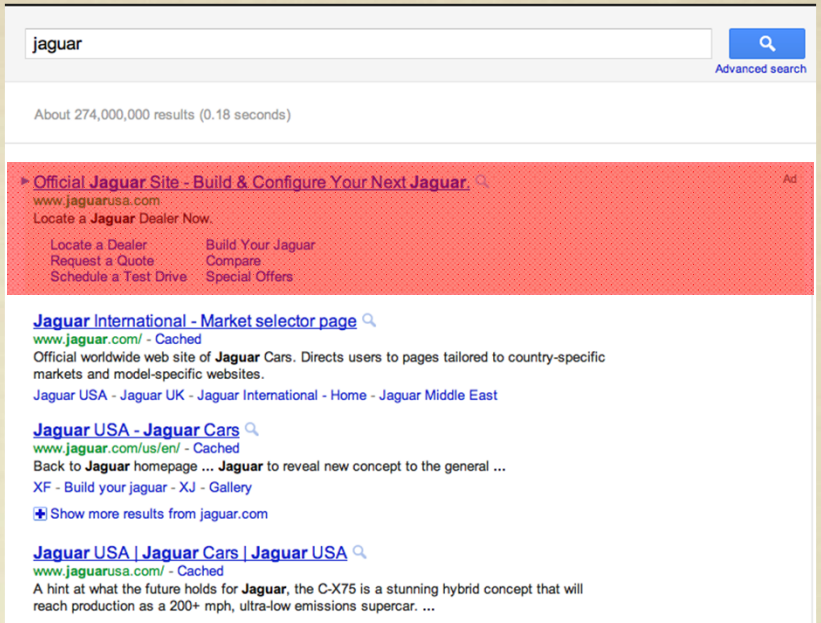


1. Search for query keywords in mined pages.
2. Select a set of “matching” pages and ads.
3. Sort pages by PageRank and return results.

Google in a Nutshell

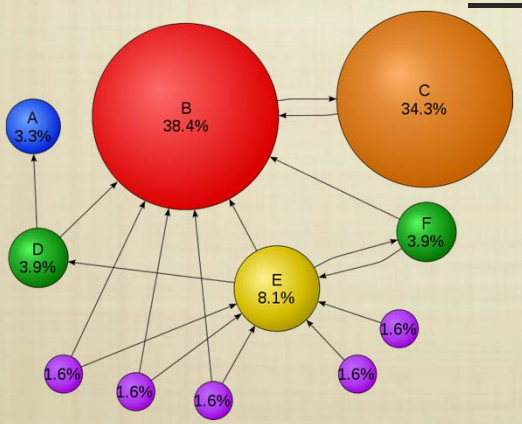


Query



Result

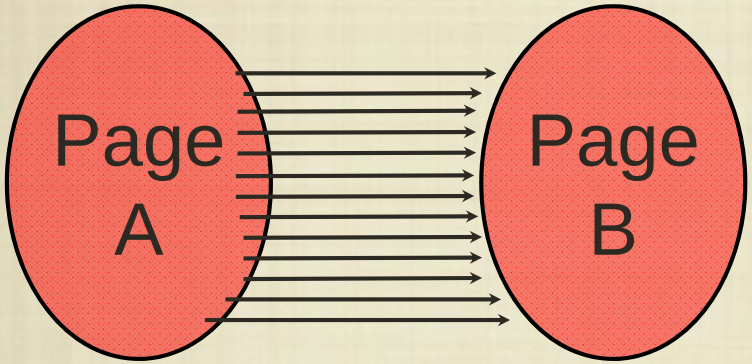
Google Data Center



1. Search for query keywords in mined pages.
2. Select a set of “matching” pages and ads.
3. Sort pages by PageRank and return results.

Increasing PageRank

- To have your webpage have largest PageRank would mean increased visibility. Can we artificially boost a PageRank score?



Create pages that have many links to one another.

Convince big sites to link to your webpage.

Will these work? Are they easy to do?

Small-World Phenomenon

- In the 1960s, psychologist Stanley Milgram performed an experiment to test the chance that two random people know one another through aquaintances:

- 1. Two random individuals in geographically distant cities were chosen and given an information packet about the experiment (that included a roster of names).

- 2. Upon receiving the packet, the subject signs the roster. If they knew the other recipient, they would forward the packet directly. Otherwise, they sign their name to the roster, and forward the packet to someone they thought was more likely to know the recipient.

- 64/296 letters reached their destination, and did so in about 5.5 “steps”. This is where the term “six degrees of separation” comes from.

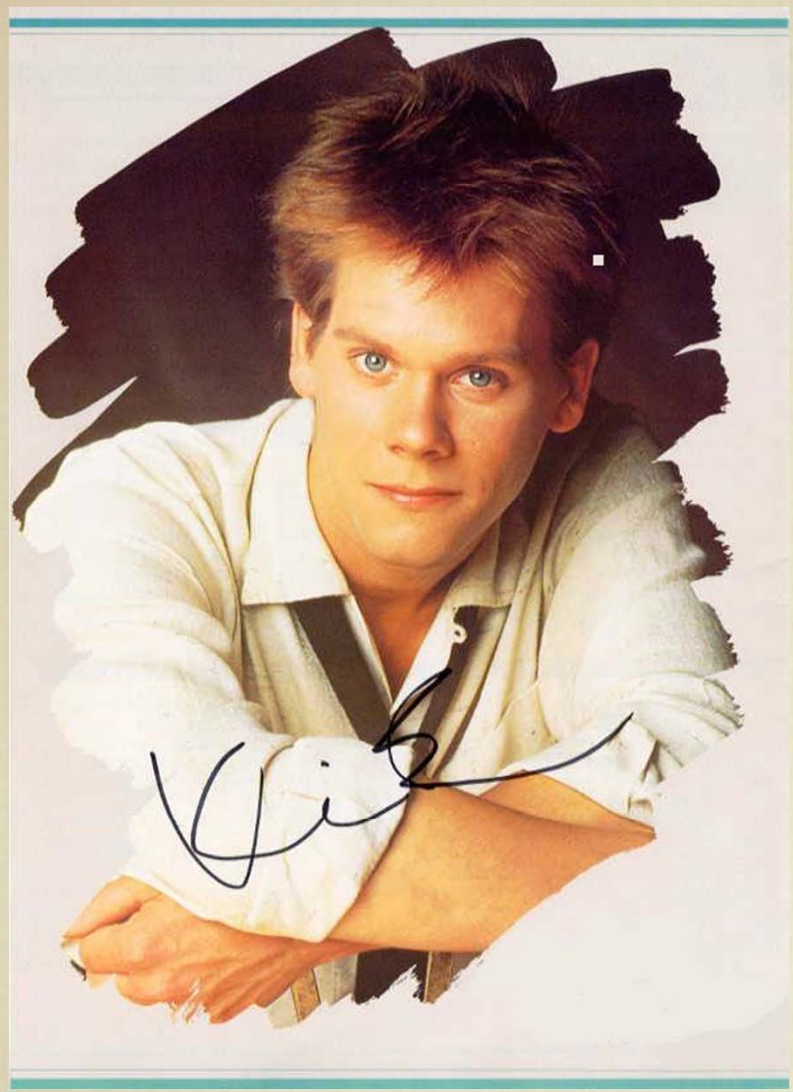
Application to Computer Science

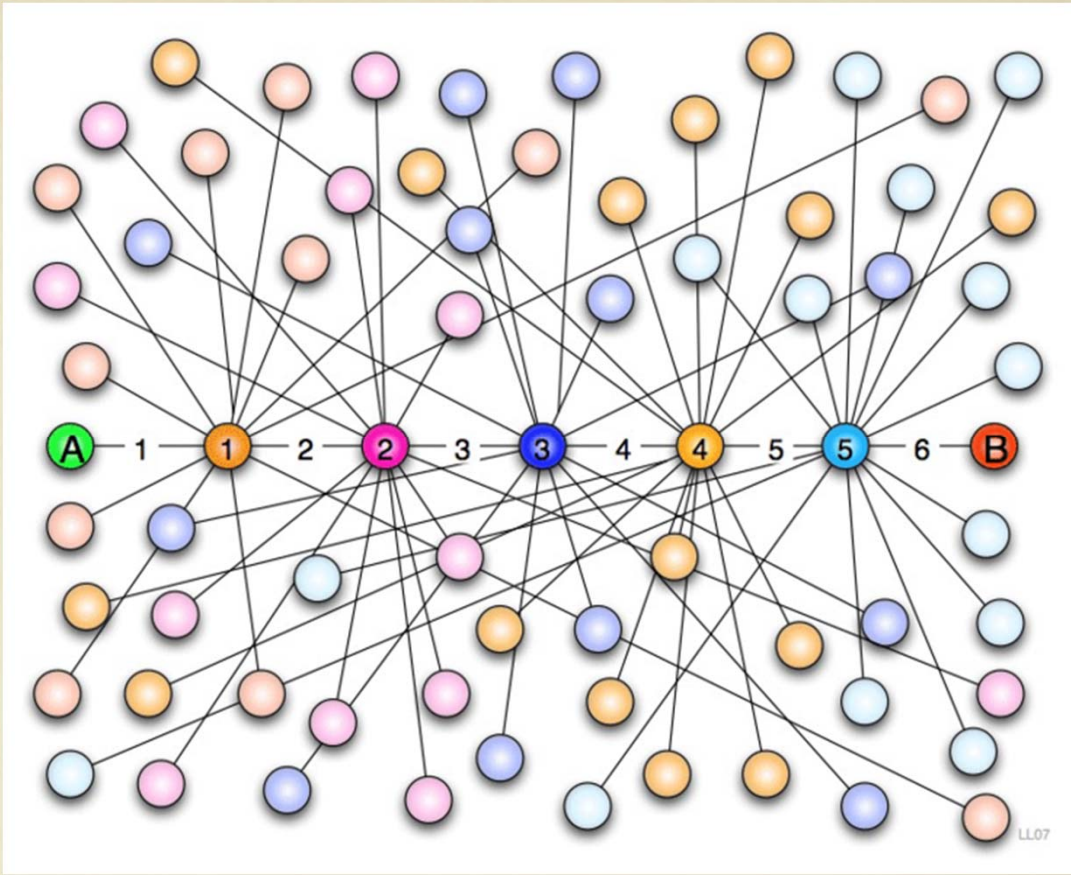
- The small-world phenomenon has two interesting connections to computer science:

- 1. “Oblivious routing”, in which packets are forwarded to a random neighbor reach their destinations surprisingly quickly.

- 2. A graph generated at random exhibits the “small-world” phenomenon.

- These two facts have recently been used by researchers to implement routing protocols and analyze dynamically generated content graphs.





The game “six degrees of Kevin Bacon” apparently works on this same principle.

Recap

- What is a graph? How can we represent it?
- How many edges can an undirected graph (with one edge per pair of vertices) have?
- What does it mean for a graph to be “connected”? What is the definition of a shortest path in a graph?
- What is breadth-first search? What auxiliary data structure does it use, and why?
- How are the problems of network routing, web page ranking and content ranking solved using graphs? In each instance, how is a graph used?