Applications of Graph Theory

Juan B. Gutierrez, UGA

January 21, 2016

Shortest Path Solutions

A **graph** can have different types of paths (remember, mathematicians need to be very precise):

- **Walk**: Move between vertices without restriction.
- **Trail**: Each edge must occur at most once.
- **Path**: Each vertex must occur at most once.
- **Cycle**: Closed path.

Shortest Path Solutions

Several solutions have been found to find the shortest path in a graph G with m edges and n vertices:

Shortest Path Solutions

Several solutions have been found to find the shortest path in a graph G with m edges and n vertices:

- **Bellman-Ford** (1958): $O(mn)$ from i to j. $O(mn^2)$ from i to every j. $O(mn^3)$ from every i to every j.

Shortest Path Solutions

Several solutions have been found to find the shortest path in a graph G with m edges and n vertices:

- **Bellman-Ford** (1958): $O(mn)$ from i to j. $O(mn^2)$ from i to every j. $O(mn^3)$ from every i to every j.
- **Dijkstra** (1959): $O(n^2)$ from i to every j. $O(n^3)$ from every i to every j. $O(m \log n)$ if an inverted heap is used.

Shortest Path Solutions

Several solutions have been found to find the shortest path in a graph G with m edges and n vertices:

- **Bellman-Ford** (1958): $O(mn)$ from i to j. $O(mn^2)$ from i to every j. $O(mn^3)$ from every i to every j.
- **Dijkstra** (1959): $O(n^2)$ from i to every j. $O(n^3)$ from every i to every j. $O(m \log n)$ if an inverted heap is used.
- **Floyd-Warshall** (1962): $O(n^3)$ to find from every i to every j.

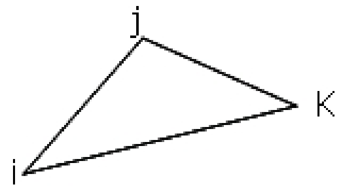
Shortest Path Solutions

Several solutions have been found to find the shortest path in a graph G with m edges and n vertices:

- **Bellman-Ford** (1958): $O(mn)$ from i to j. $O(mn^2)$ from i to every j. $O(mn^3)$ from every i to every j.
- **Dijkstra** (1959): $O(n^2)$ from i to every j. $O(n^3)$ from every i to every j. $O(m \log n)$ if an inverted heap is used.
- **Floyd-Warshall** (1962): $O(n^3)$ to find from every i to every j.
- **Gomory-Hu** (1961!!!): $O(n^3)$ to find from every i to every j plus a tag matrix (very useful as we shall see).

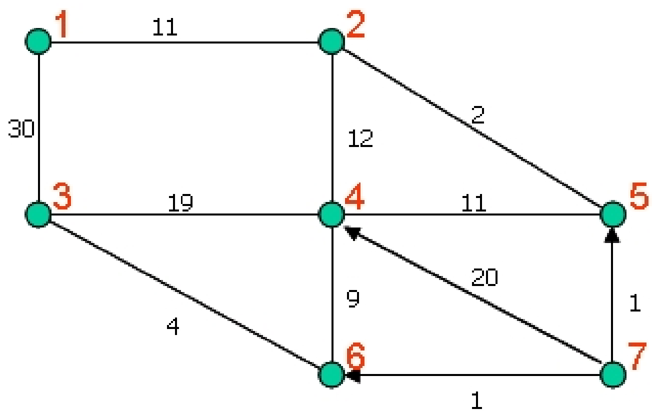
Floyd-Warshall Algorithm

$$c_{ik} = \begin{cases} c_{ij} + c_{jk}, & \text{if } c_{ik} > c_{ij} + c_{jk} \\ c_{ik}, & \text{otherwise} \end{cases}$$



This algorithm works by Bellman's optimality principle (or minimality principle): If $P = 1, 2, ..., i, j$ is a shortest path from 1 to j, then $P : 1 \rightarrow i$ is a shortest path as well.

Route Calculation



--	1	2	3	4	5	6	7
1	0	11	30	Inf	Inf	Inf	Inf
2	11	0	Inf	12	2	Inf	Inf
3	30	Inf	0	19	Inf	4	Inf
4	Inf	12	19	0	11	9	Inf
5	Inf	2	Inf	11	0	Inf	Inf
6	Inf	Inf	4	9	Inf	0	Inf
7	Inf	Inf	Inf	20	1	1	0

--	1	2	3	4	5	6	7
1	0	11	30	23	13	32	Inf
2	11	0	25	12	2	21	Inf
3	30	31	0	13	30	4	Inf
4	23	12	13	0	11	9	Inf
5	13	2	24	11	0	20	Inf
6	32	21	4	9	20	0	Inf
7	14	3	5	10	1	1	0

--	1	2	3	4	5	6	7
1	1	2	3	2	2	2	7
2	1	2	4	4	5	4	7
3	1	6	3	6	6	6	7
4	2	2	6	4	5	6	7
5	2	2	4	4	5	4	7
6	4	4	3	4	4	6	7
7	5	5	6	6	5	6	7