Efficient Algorithms for Selected Problems: Design, Analysis and Implementation

Ye Myat Thein
Institute of Informatics
University of Bonn

Outline

- Background Information
- Overview of Algorithms
- Implementation Aspect
- Program Demonstration
- Summary

Background Information

Flow Network

- A directed graph in which each edge has a capacity to allow the flow of units from source node to destination node. Moreover, each edge has an associated cost to transport the units.
- Minimum cost flow problem finds the cheapest possible way, without violation of capacity requirement, to find the transport of units across the graph so that all nodes are balanced.

Strongly Polynomial Algorithms

- All of the algorithms to solve minimum cost flow problem fall into one of the two basic categories: weakly polynomial and strongly polynomial.
- To be a strongly polynomial algorithms, two constrains must be met.
  - The number of arithmetic operations is polynomially bounded in the number of nodes n, and the number of arcs m.
  - The only arithmetic operations involved are comparisons, additions and subtraction.

Running Time of Minimum Cost Flow Algorithms

Polynomial Algorithms

#	Due to	Year	Running Time
1	Edmonds and Karp	1972	$O((n + m') \log U \, S(n, m, nC))$
2	Rock	1980	$O((n + m') \log U \, S(n, m, nC))$
3	Rock	1980	$O(n \log C \, M(n, m, U))$
4	Bland and Jensen	1985	$O(m \log C \, M(n, m, U))$
5	Goldberg and Tarjan	1987	$O(nm \log (n^2/m) \log (nC))$
6	Goldberg and Tarjan	1988	$O(nm \log n \log (nC))$
7	Ahuja, Goldberg, Orlin and Tarjan	1988	$O(nm \log \log U \log (nC))$

Strongly Polynomial Algorithms

#	Due to	Year	Running Time
1	Tardos	1985	$O(m^4)$
2	Orlin	1984	$O((n + m')^2 \log n \, S(n, m))$
3	Fujishige	1986	$O((n + m')^2 \log n \, S(n, m))$
4	Galil and Tardos	1986	$O(n^2 \log n \, S(n, m))$
5	Goldberg and Tarjan	1987	$O(nm^2 \log n \log(n^2/m))$
6	Goldberg and Tarjan	1988	$O(nm^2 \log^2 n)$
7	Orlin (this paper)	1988	$O((n + m') \log n \, S(n, m))$

$S(n, m)$	$= O(m + n \log n)$	Fredman and Tarjan [1984]
$S(n, m, C)$	$= O(\mathrm{Min}\,(m + n\sqrt{\log C}),$	Ahuja, Mehlhorn, Orlin and Tarjan [1990]
	$(m \log \log C))$	Van Emde Boas, Kaas and Zijlstra[1977]
$M(n, m)$	$= O(\min\,(nm + n^{2+\varepsilon}, nm \log n)$ where ε is any fixed constant.	King, Rao, and Tarjan [1991]
$M(n, m, U)$	$= O(nm \log (\frac{n}{m}\sqrt{\log U} + 2))$	Ahuja, Orlin and Tarjan [1989]

Overview of Algorithms

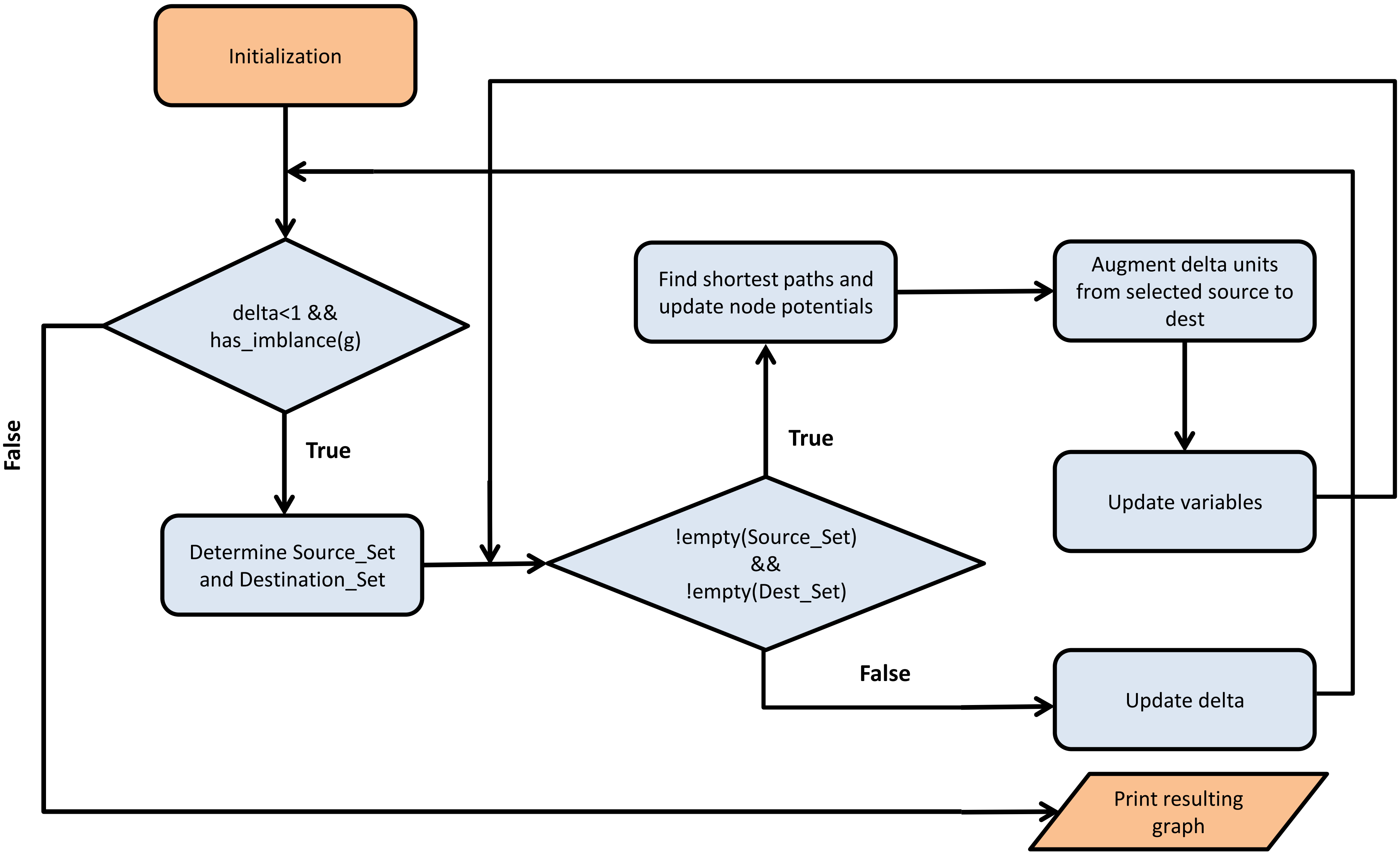
Scaling Algorithms

- Define a certain amount of units in each iteration for transport.
- Well-known algorithms that use scaling to solve min-cost flow problems.
  - Successive shortest path and capacity scaling.
  - Cost scaling.

Edmonds-Karp Scaling Algorithm

- Called Right Hand Side (RHS) scaling.
- Use pseudoflows, node potential, reduced cost, and residual network concepts.
- Find delta-optimality in each iteration.
- The iterations terminate when delta less than 1 or all nodes in the graph are balanced.

Edmonds-Karp Scaling Algorithm



Orlin’s Strongly Polynomial Algorithm

- Extension of RHS algorithm to become strongly polynomial.
- Identify arcs whose flow are so large in the delta scaling phase that they are guaranteed to have positive flow in all subsequent scaling phases.
- Sufficiently large amount is equal to **4*n*delta**.
- So any arc which has flows that is strictly greater than the threshold will be contracted.

Why 4*n*delta

- In modified version of RHS algorithm, the flow in any arc can change at most 2*n*delta units.
- The total number of phases can be 2*n.
- The total change of flow is at most

  2*n*(delta+delta/2+delta/4+…+1)=4*n*delta

- The arcs which have flows that is strictly greater than the threshold are called strongly basic arcs and will be contracted.

Difference between Orlin's Algorithm and RHS Algorithm

- Initial delta selection.
- Contraction before the determining the source set and target set.
- Cost calculation must consider the contracted arcs by expanding them back at the end of the algorithm

Difference between Orlin's Algorithm and RHS Algorithm

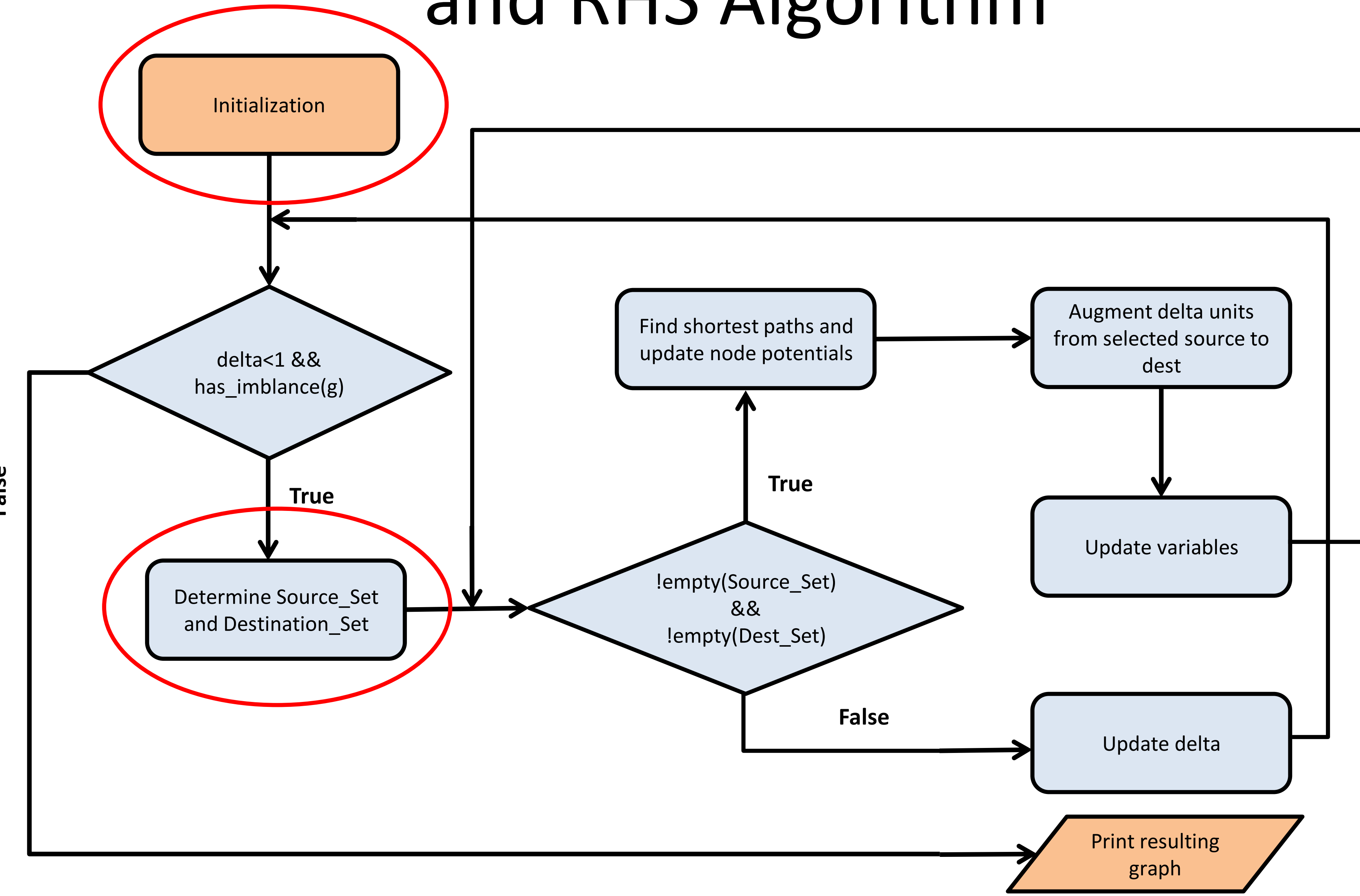
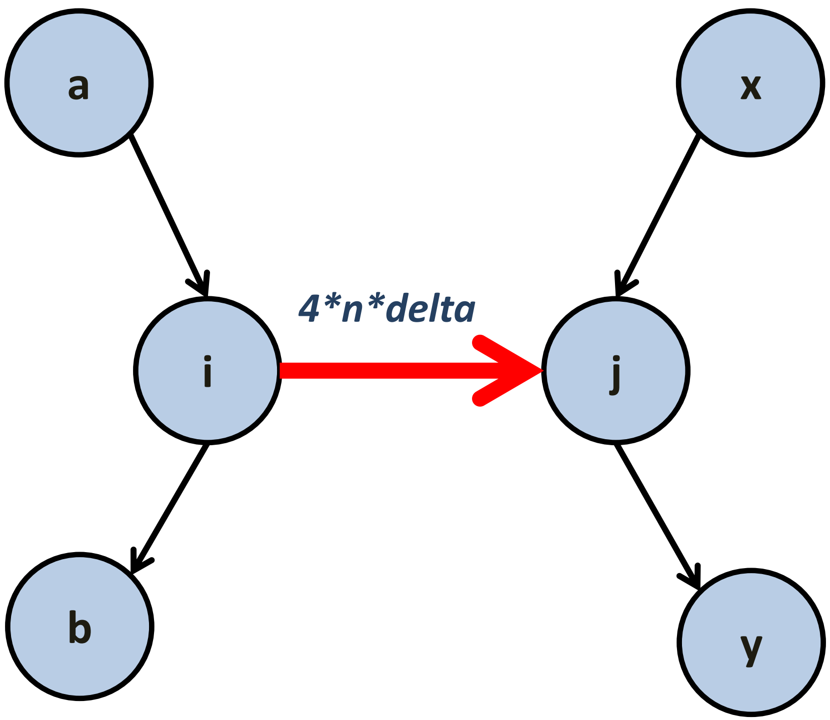
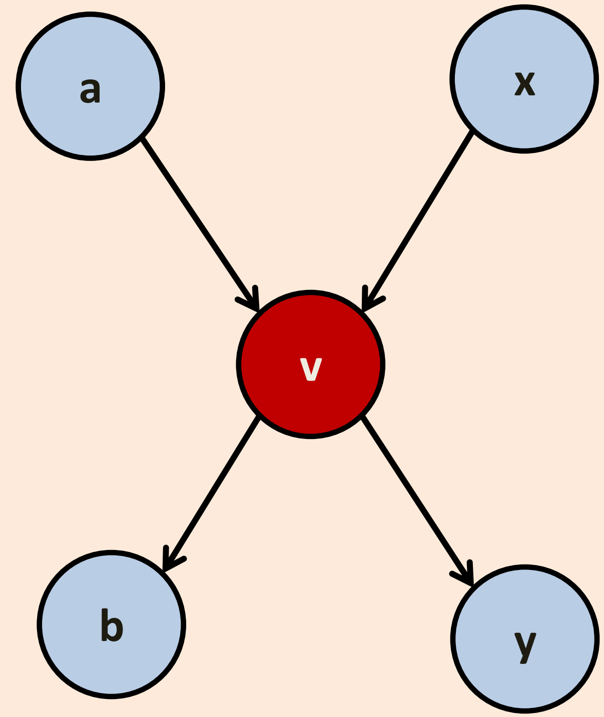


Illustration of Contraction



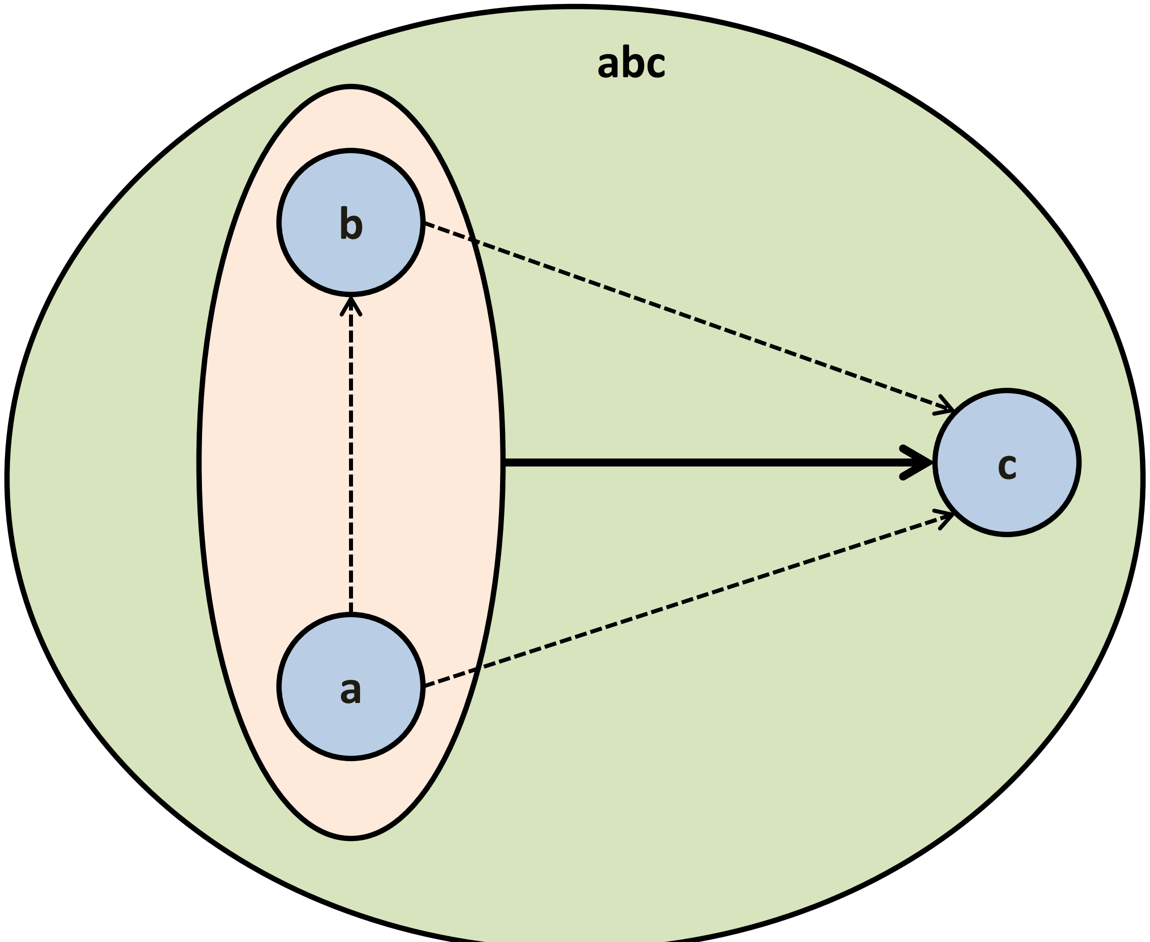


$$b(v) = b(i) + b(j)$$
$$e(v) = e(i) + e(j)$$

Arcs Expansion and Cost Calculation

- For example, let’s say we have the contracted node “abc” which is made up of two nodes: “ab” and “c”.
- The node “ab” is the contraction between node “a” and node “b”.
- When a contracted node is considered for expansion and cost calculation, two-steps-look-back approach is used.
- So, to get the related cost of node “abc”, two contraction backward is looked back.

Arcs Expansion and Cost Calculation



Implementation Aspect

Libraries and Programming Language

- Python programming language due to its clear syntax and availability of scientific libraries.
- NetworkX libray for basic infrasture such as graph, nodes, etc
- Use Bellman Ford algorithm for finding shortest path, and this algorithm is provided by the NetworkX library.
- Modular design, functions are separated logically and can be replaced easily.
- Input module is well-separated from main programs so that new input graph can be specified easily.
- Input is the graph data in text file and the output is the total cost of flows along with the flow information in each iteration. (displayed in the standard output)

Program Demonstration

Summary

- Minimizing the cost of flow on network is a very useful problem in practical applications and there are many well-known algorithms to solve it.
- Two types of algorithms are weakly polynomial and strongly polynomial.
- In my work both types of algorithms are implemented (Edmonds-Karp and Orlin).
- Assumptions are no negative weight and no limitation of capacity, and are made for simplicity in implementation.

Acknowledgement

I have used many related papers from Dr.James B. Orlin and learned the course “Network Optimization” from MIT open courseware to understand concepts and his algorithm.

This is the presentation for lab course at Institute of Informatics, University of Bonn offered by Prof.Dr.Norbert Blum and Mr.Matthias Kretschmer, and would like to thank them for support throughout the course.