Efficient routing for multicast and broadcast

Extra Credit Project - Final Presentation

IT-304 Computer Networks

Prof. Laxminarayana Pillutla

Group members:

201301047 - Yashwant Keswani

201301423 - Devarsh Sheth

201301442 - Akshar Varma

DAIICT, Gandhinagar

Thursday, November 19^{th}, 2015

Brief Description of Problem

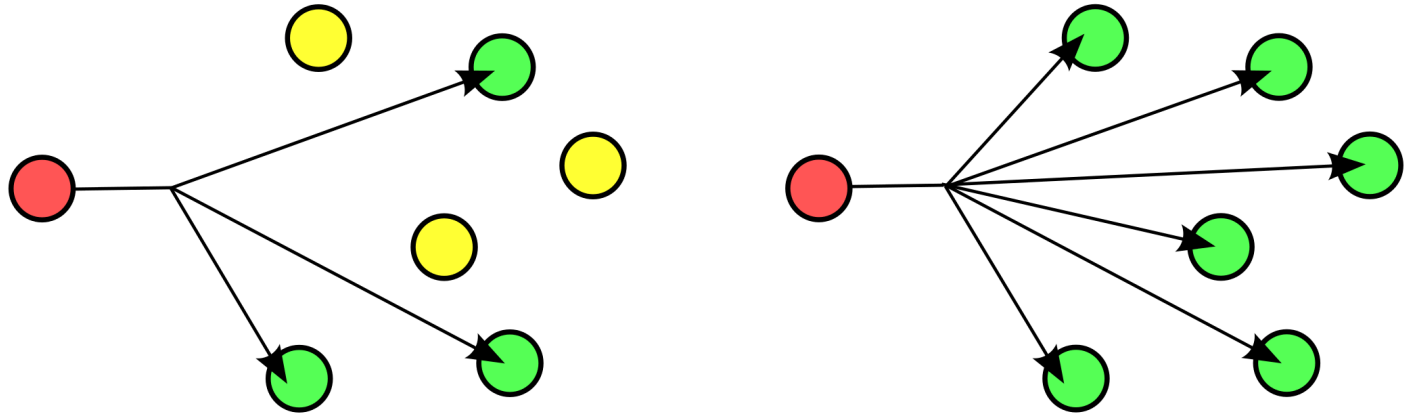

Figure : Multicasting and Broadcasting

One-to-Many and One-to-All communication.

The problem deals with routing packets that are sent to multiple receivers. We will be looking at the parameters associated with the links that effect efficiency of the routing and also the algorithms that opitmize the routes.

Two-pronged Approach

We had a two-pronged approach to solving the problem.

- **Cost Function**:
  1. *Quality of Service (QoS) guarantee.*
  2. *Cost minimization.*

- **Routing Algorithm**:
  1. *Minimum Spanning Tree.*
  2. *Modified (Distributed) Version of Algorithm.*

Cost Function (Approach)

2 Stage Cost Function:

1. *Quality of Service (QoS) guarantee.*
   Used for real-time data, or other high priority data.
   Constraints on Bandwidth, Loss Probability, Propagation Delay.
2. *Cost minimization.*
   Required for every kind of data. This is what makes it optimal.

We implemented a cost function which decides the weights of each link in the network graph.

Link Parameters Used

We consider two physically defined attributes of a link.

- **Bandwidth**: *How much capacity does the link have.*
  The amount of data that can pass through a particular link in unit time.
- **Propagation Delay**: *How long is the link.*
  The amount of time it takes for the unit packet to travel from the sender to the receiver.

We define two attributes that don't have direct physical significance but are results of network operation.

- **Loss Probability**: *How reliable is the link.*
  The probability that a packet sent along this link would be lost.
- **Efficiency Factor**: *How efficiently does the link utilize it's capacity.*
  The throughput of a link is the amount of data that actually passes through the link in unit time. Throughput per bandwidth is efficiency factor.

Cost function (Definition and Usage)

Maximize: Bandwidth (BW) and Efficiency Factor (EF).
Minimize: Loss Probability (LP) and Delay Time (T).

$$C \propto \frac{LP \times T}{BW \times EF}$$

We would periodically be reassigning the cost of the edges using the new values of efficiency factor and the loss probability. Reassignment would also be triggered when there is a change in the network topology. This allows us to improve efficiency.

Results

Table : Increase in system throughput (Mbps).

Iteration Number	Throughput (cost assigned)	Throughput (*NS* default)
1	0.021728	0.000023
2	0.613444	0.000023
3	0.327095	0.000023
4	0.613444	0.000023
5	0.327095	0.000023

There is a clear increase in the system throughput using our cost function as compared to the default manner in which *NS* does the cost assignments.

Algorithm (Approach)

The routing algorithm should have two basic properties.

- *Reach all nodes.*
  Span all nodes in network or in multicast group.
- *Minimize overall cost.*
  We don't worry about cost between pair of nodes but only the overall cost.

These requirements are fulfilled by a minimum spanning tree of the network graph.

Kruskal's Minimum Spanning Tree Algorithm

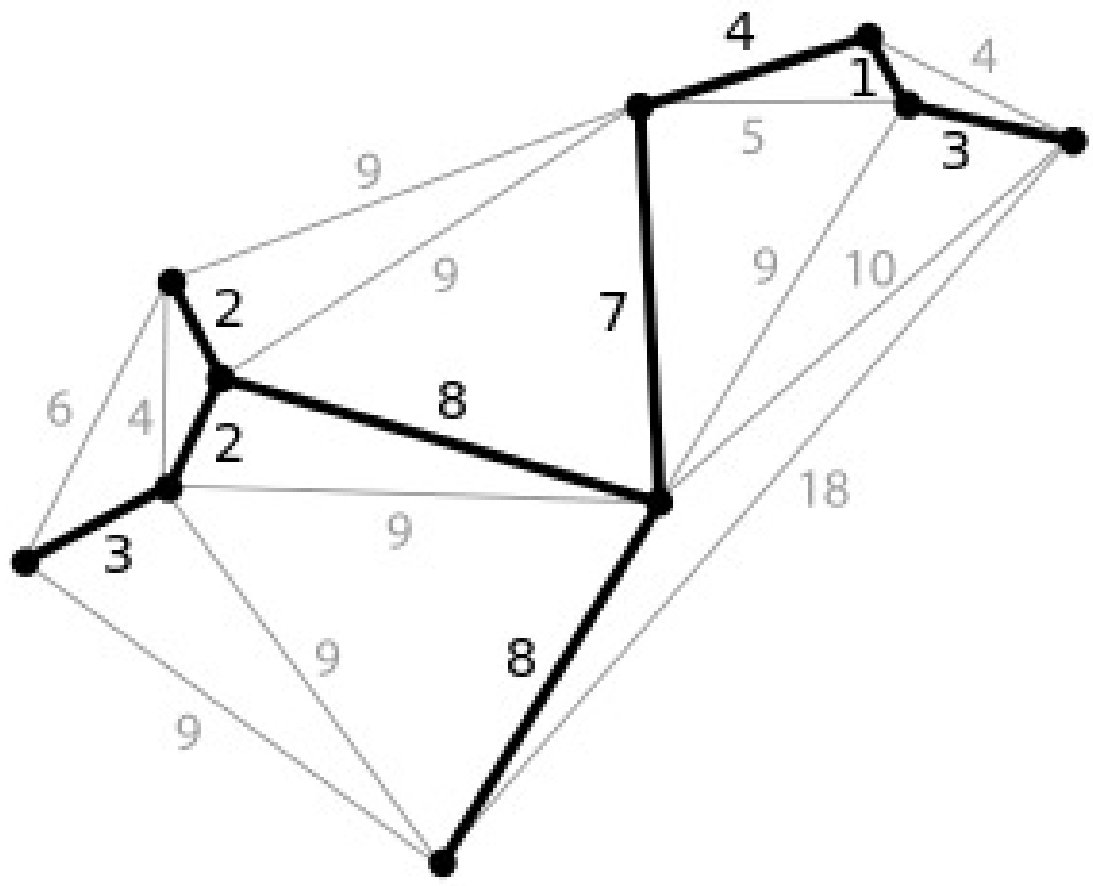

Figure : Minimimum Spanning Tree

We sort the edges by their weights and then keep selecting the smallest weight edge if it doesn't form a loop.

Distributed Minimum Spanning Tree Algorithm (GHS Algorithm)

- *Locally Computable.*
  Each node requires only communicating with neighbors.
  No need to keep or wait for global data. Convergence would be faster.

- *Efficient Algorithm. Distributed in nature.*
  The computation power of each node can be harvested.
  Might prove useful when nodes join/leave multicast group.

- *Hierarchical.*
  Can divide network into subnetworks and make MSTs there.

Our Modified Distributed MST Algorithm

We weren't able to fully code the whole distributed algorithm. We modified the Kruskal's algorithm that we were using to simulate the distributed one. We then used this simplified algorithm to benchmark the running times of both algorithms.

This simplified version still retains the local computability, and heirarchical nature of the GHS Algorithm. It isn't as distributed in nature but has similar features. Since it isn't as distributed, the overhead of having a distributed algorithm isn't present. But even if the overhead had been present, our modified version would be much faster.

Results - Time Comparison Graph

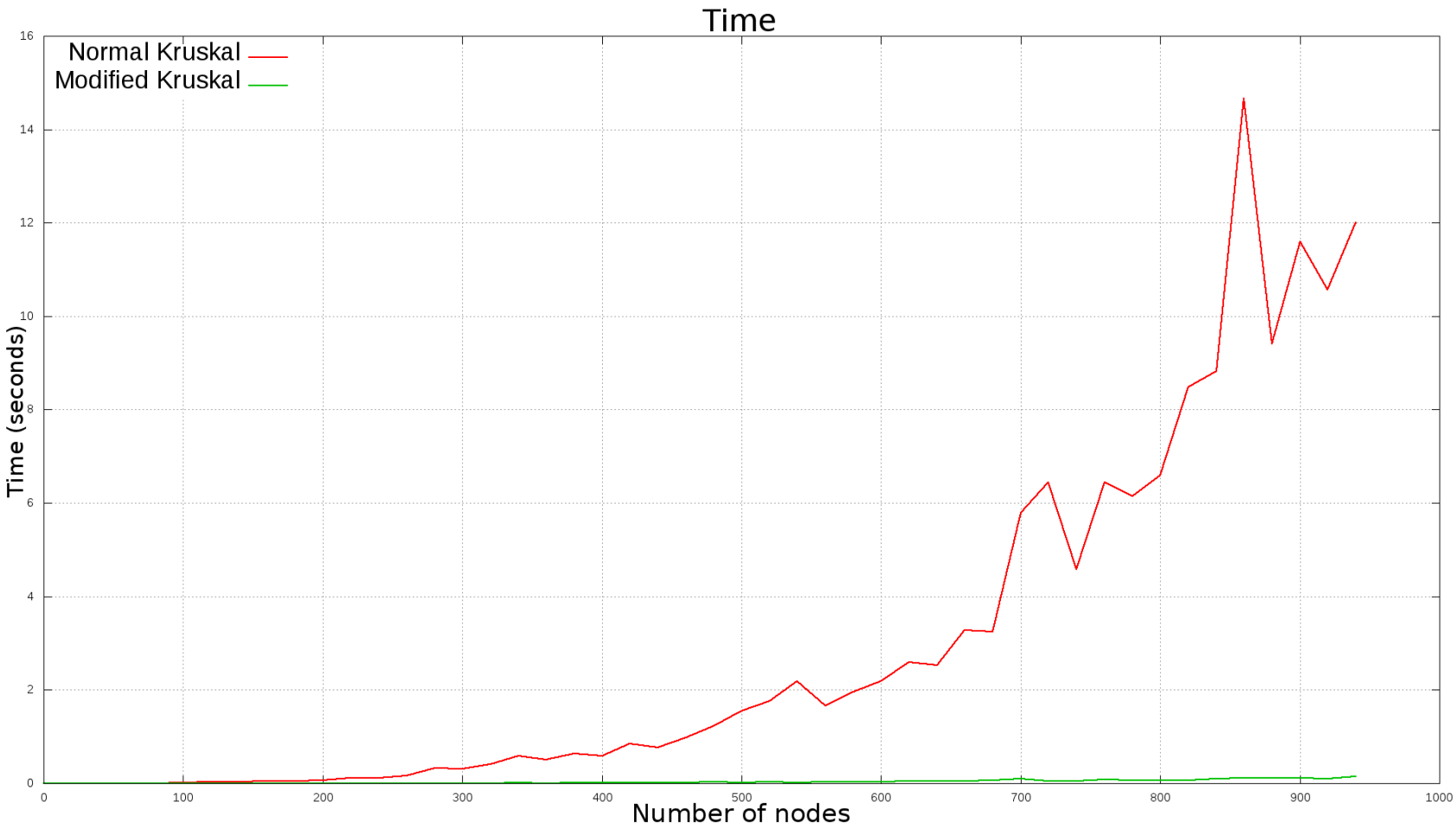

Figure : Number of nodes vs Time

Results - Speedup Graph

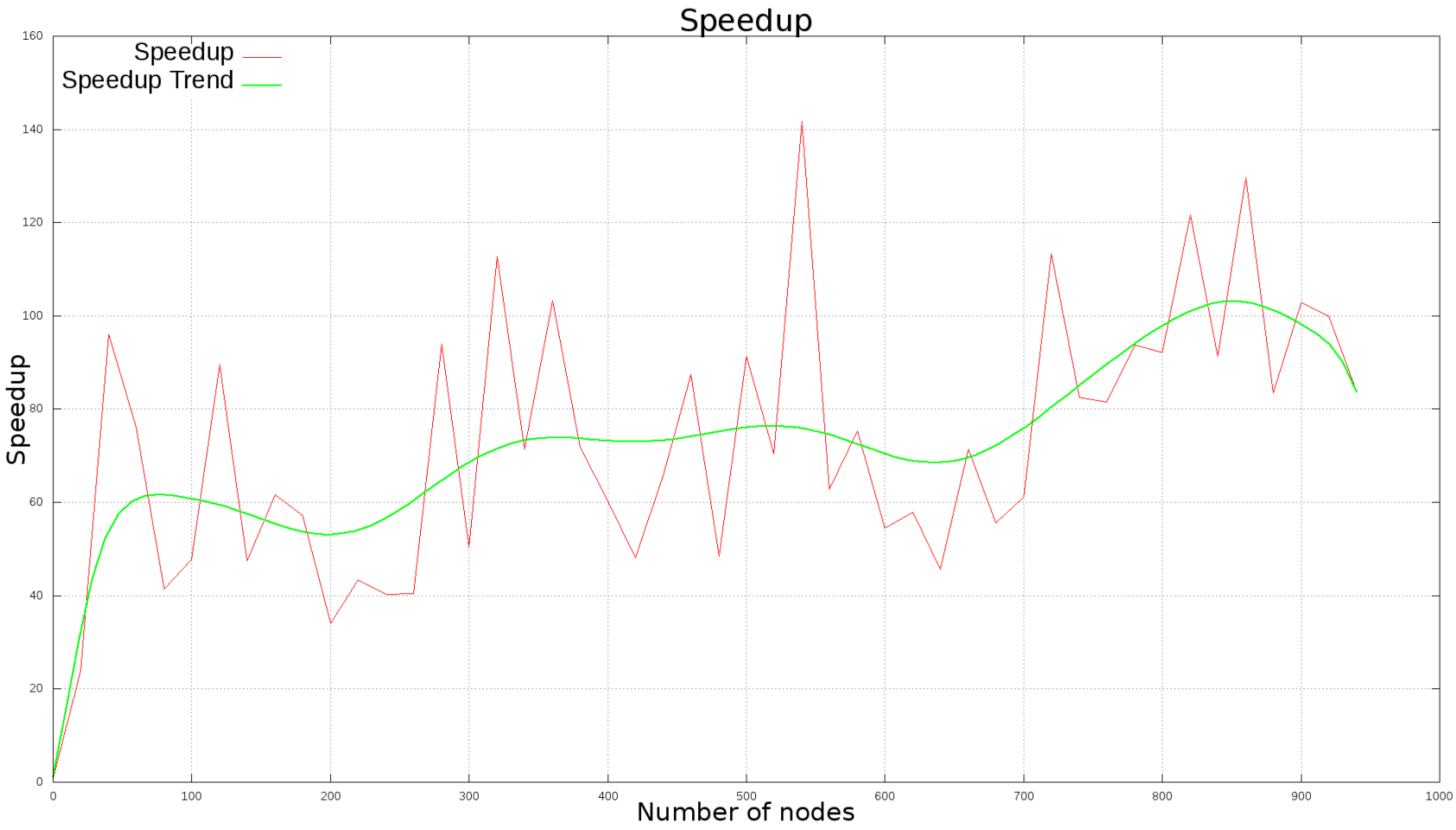

Figure : Number of nodes vs Speedup

Current State and Future Work

- We have implemented the cost function and benchmarked it against the default weights of *NS* and have shown that our cost function does improve efficiency.
- We have compared the working of two different algorithms for routing and have shown that there is a significant improvement in the modified Kruskal's algorithm.
- We have yet to integrate the both in *NS* and benchmark that. This would require extensive work in *NS* apart from what we have done. This can be taken up in the future.

References

- Shigang Chen and K. Nahrstedt. 1998. An overview of quality of service routing for next-generation high-speed networks: problems and solutions. Netwrk. Mag. of Global Internetwkg. 12, 6 (November 1998), 64-79. DOI=http://dx.doi.org/10.1109/65.752646
- Minimum Spanning Tree - Wikipedia: https://en.wikipedia.org/wiki/Minimum_spanning_tree
- Kruskal's Algorithm - Wikipedia: https://en.wikipedia.org/wiki/Kruskal's_algorithm
- Borvka's algorithm - Wikipedia: https://en.wikipedia.org/wiki/Boruvka's_algorithm
- Gallager, Robert G., Pierre A. Humblet, and Philip M. Spira. "A distributed algorithm for minimum-weight spanning trees." ACM Transactions on Programming Languages and systems (TOPLAS) 5.1 (1983): 66-77.
- Distributed Minimum Spanning Tree Algorithm - Wikipedia: https://en.wikipedia.org/wiki/Distributed_minimum_spanning_tree