QoS-Constrained Multi-path Routing for High-End Network Applications

Xiaomin Chen, Mohit Chamania, Prof. Admela Jukan[1]
André C. Drummond, Prof. Nelson L. S. da Fonseca[2]

[1] Technische Universität
Carolo-Wilhelmina zu Braunschweig
Germany

[2] Institute of Computing
University of Campinas
Brazil

High-Speed Networks Workshop, 2009

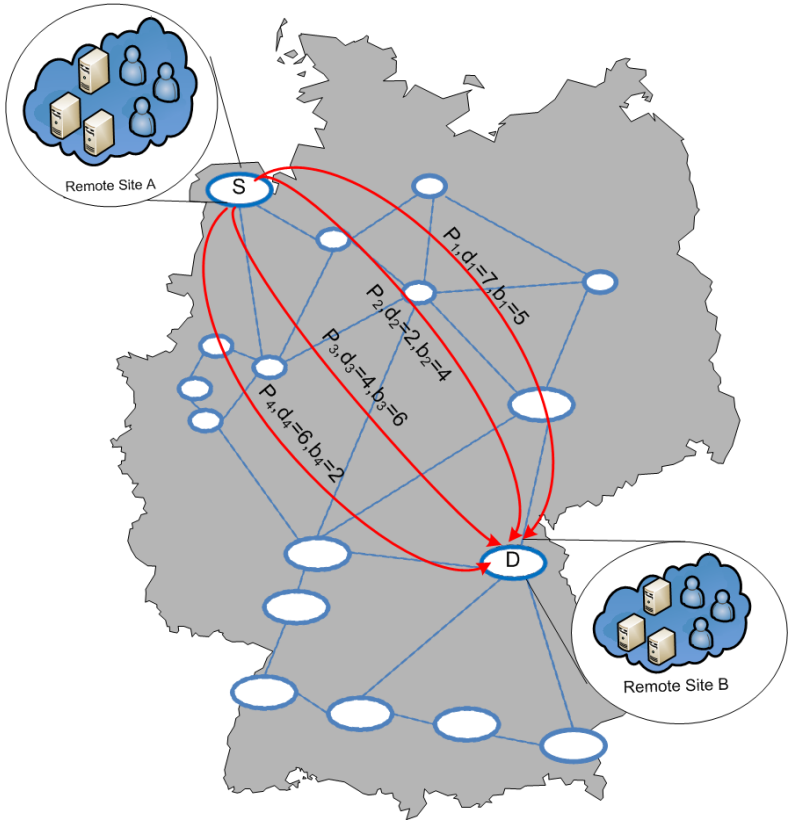
Motivation

- High-end applications have become dependent on global networks, with supercomputing and data-storage facilities widely distributed.
- It is known that multi-path routing can provide more aggregate bandwidth.
- Multiple paths may present differences in the end-to-end delay of each path, which causes jitter at the destination (Differential Delay).
- High speed memory is extremely expensive, and therefore cannot be over-provisioned.

An Example of Data Transmission over Multiple Paths

Request of 40 units from S to D

- Four possible paths P_1,P_2,P_3, and P_4 with delay d and capacity b.
- Single shortest path: P_2
  Delay $40/b_2 + d_2 = 12$.
- Max the achievable bandwidth: P_1 and P_3
  Delay $\max\{20/b_1 + d_1, 20/b_3 + d_3\} = 11$
  Memory $(d_1 - d_3) \cdot b_3 = 18$.
- Min the differential delay: P_1 and P_4
  Delay $\max\{20/b_1 + d_1, 20/b_4 + d_4\} = 16$
  Memory $(d_1 - d_4) \cdot b_4 = 2$.



The Problem to be solved

Given a connection request r with source $s(r)$ and destination $d(r)$, optimize the achievable bandwidth for the requested connection while minimizing the cost incurred by the memory required at the destination due to differential delay among the multiple paths.

Our solution steps

1. Define the first N shortest widest paths in the network.
2. Select a subset of paths that maximize the achievable bandwidth while taking into consideration the buffer (memory) cost.

Algorithm 1: Multipath path computation algorithm

```
Input: G(V, E), (s, d), N
count = 0;
Initialize an Empty Array of Paths pathSet;
For all v_i ∈ V s.t. there exists a link e_j ∈ Efrom s to v_i, create a path from s to v_i and insert in pathSet;
Sort pathSet in decreasing order of bandwidth;
while count < N do
    count = 0;
    for (i = 0 to pathSet.length) do
        if (pathSet[i].destination == d) then
            Increase count by 1 ;
            If(count == N) break;
        end
        else
            path = pathSet[i];
            remove pathSet[i] from pathSet;
            foreach (e_j ∈ E attached to path.destination) do
                Let the vertices at the end of e_j be v_k and path.endVertex;
                if (v_k ∉ path.vertices) then
                    Create a new path by extending path with e_j; Insert new path in pathSet using
                    insertion sort;
                end
            end
        end
    end
    if (no more paths can be extended) then
        break;
    end
end
```

$$\text{Maximize} \sum_{P \in (P)} C_F \cdot \mathbf{t_P} - C_M \cdot M_r$$

$\mathbf{t_P}$ - an integer variable for each path $P \in \mathcal{P}$ denoting the current traffic on the path P.

Subject to:

$$\forall e \in E: \ \mathbf{x_e} = \sum_{P \in \mathcal{P} \wedge e \in P} \mathbf{t_P} \quad (1)$$

$\mathbf{x_e}$ - an integer variable for each link $e \in E$ denoting the current flow on link e.

$$\forall e \in E: \ \mathbf{x_e} \leq c_e \quad (2)$$

$$M(r) = \sum_{P \in \mathcal{P}} \mathbf{t_P}(d_{\tilde{P}} - d_P) \quad (3)$$

$\tilde{P}$ - The path in the solution set with the highest delay.

$$\forall P \in \mathcal{P}: \mathbf{t_P} = 0 \ if \ d_P > d_{\tilde{P}} \quad (4)$$

$$\forall P \in \mathcal{P}: \mathbf{t_P} >= 0 \ if \ d_P <= d_{\tilde{P}} \quad (5)$$

$$\mathbf{t_P} > 0 \quad (6)$$

$$\sum_{P \in (P)} \mathbf{t_P} >= B_{min} \quad (7)$$

$$M_r \leq M_d \quad (8)$$

$$B_{min} = \frac{F}{D_{max}}$$

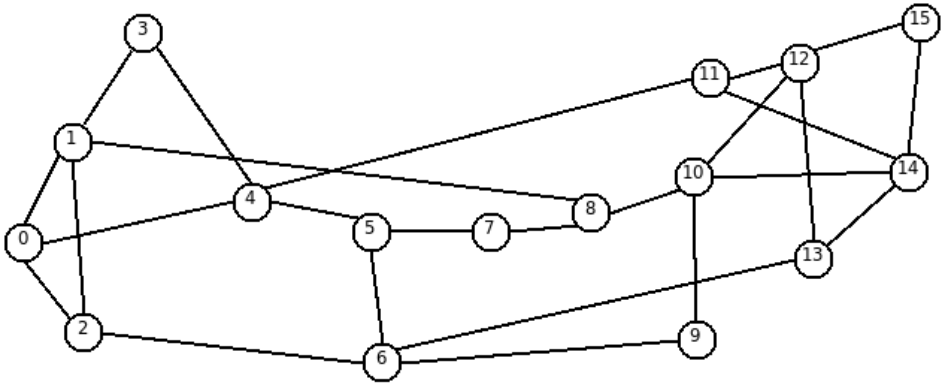
F: total size of the data.
D_{max}: maximum delay.

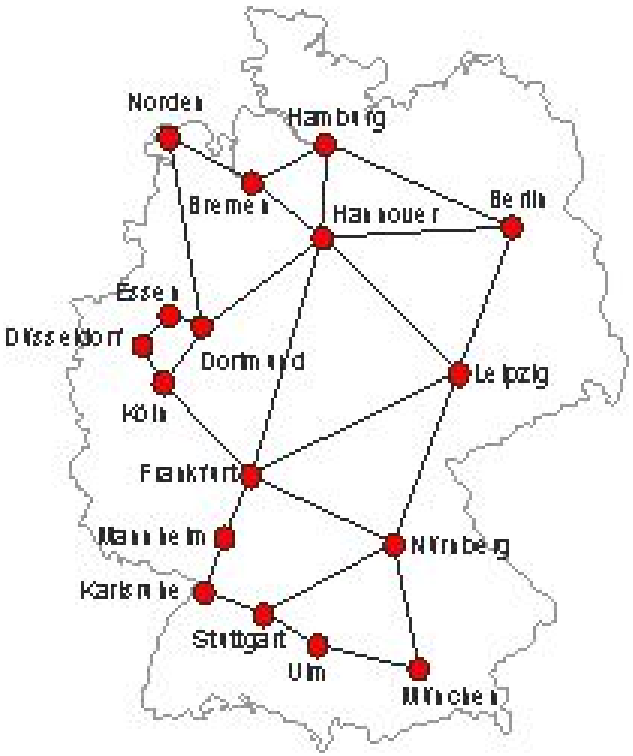
Scenarios

Multi-homing networks with three multi-path multi-domain routing schemes:

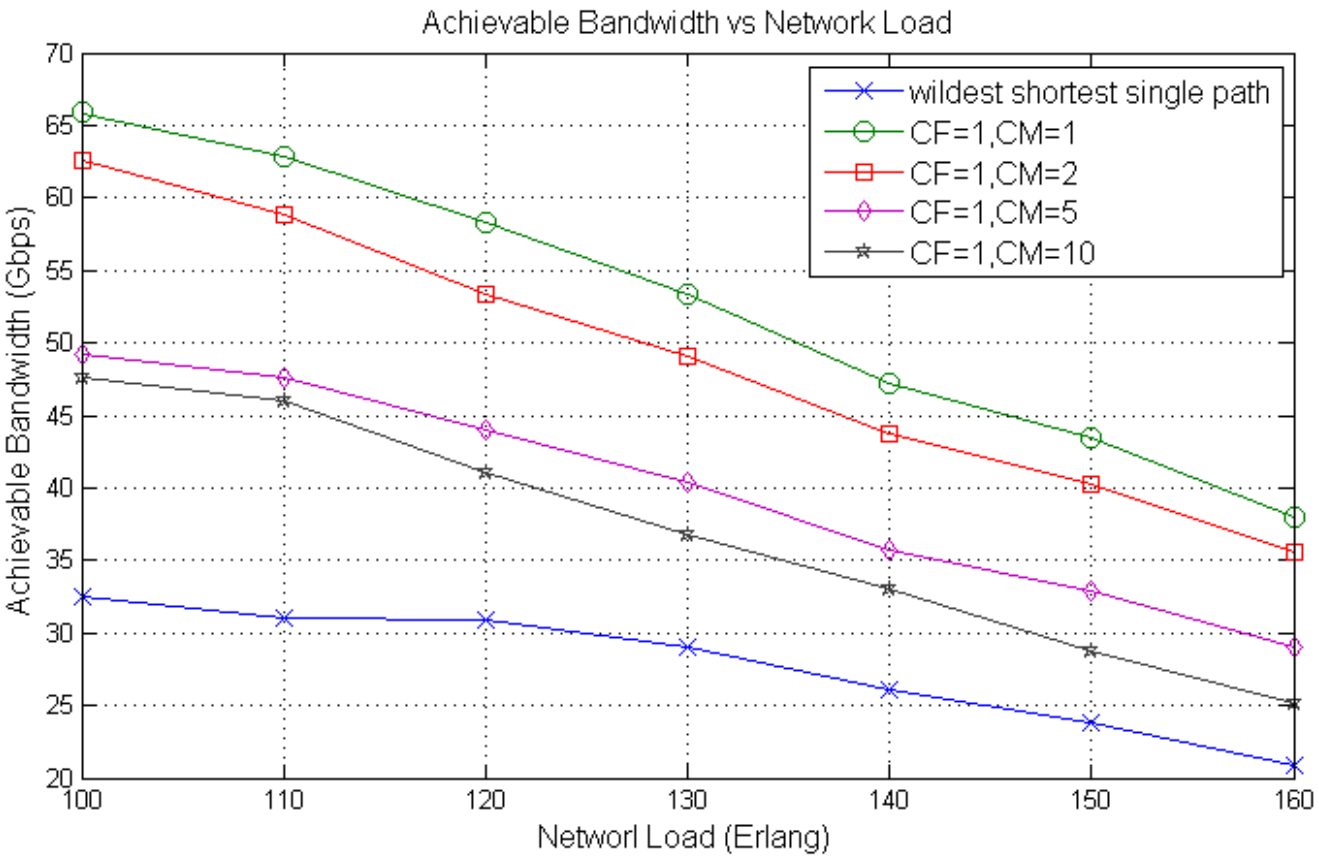
- multi-domain routing with full visibility;
- multi-domain routing with partial visibility;
- multi-path routing in a single preferred domain.

Networks Considered (NSFNet and Germany17)

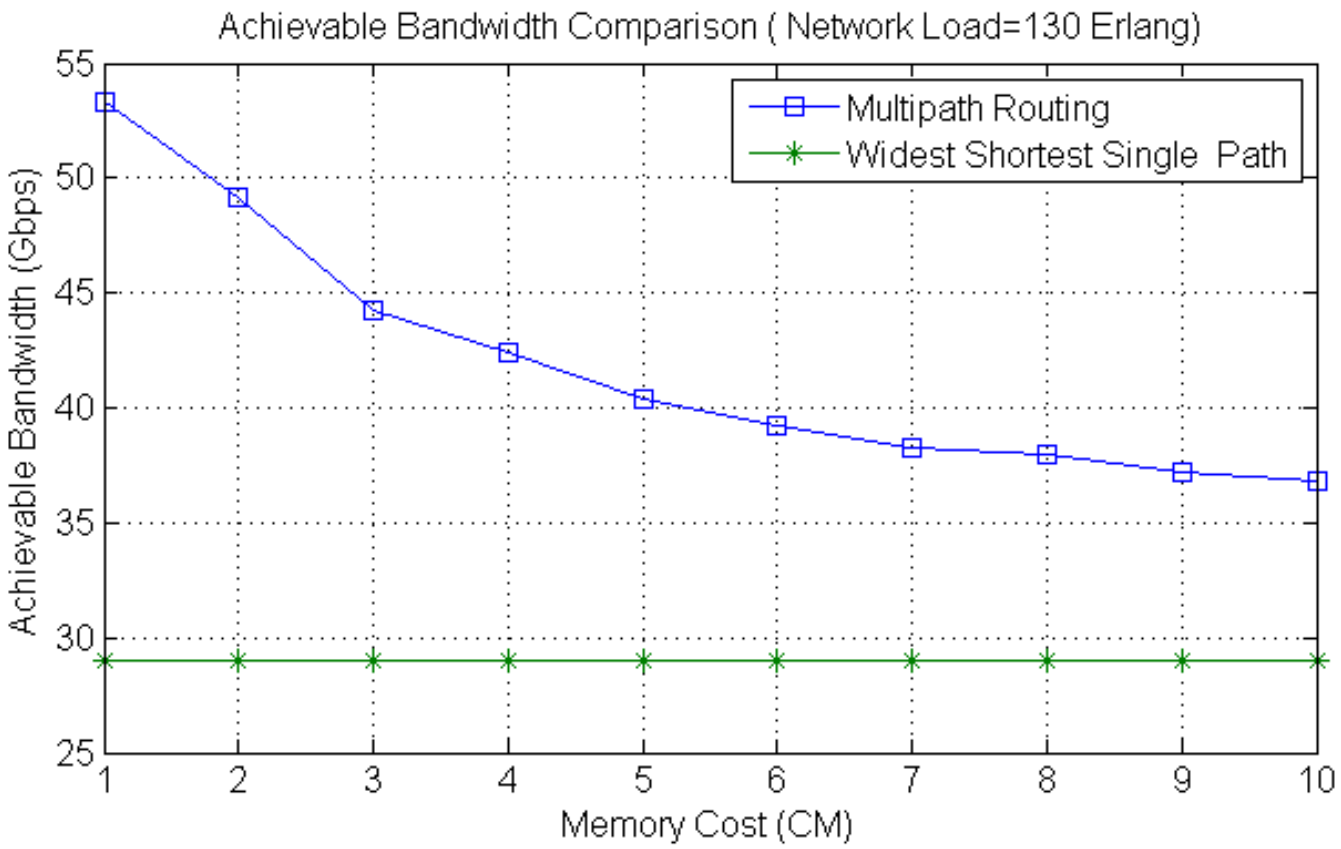




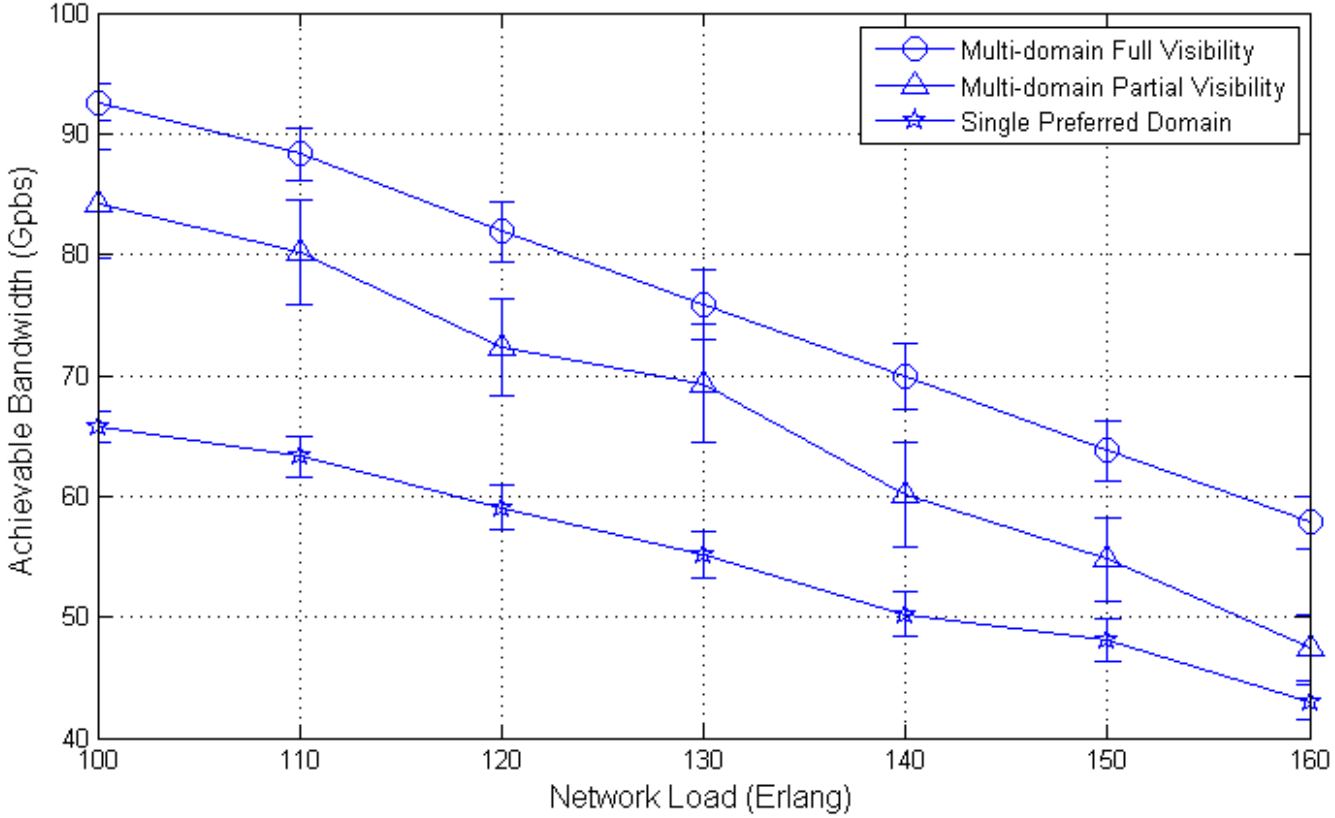
Bandwidth vs Network Load (Germany17)



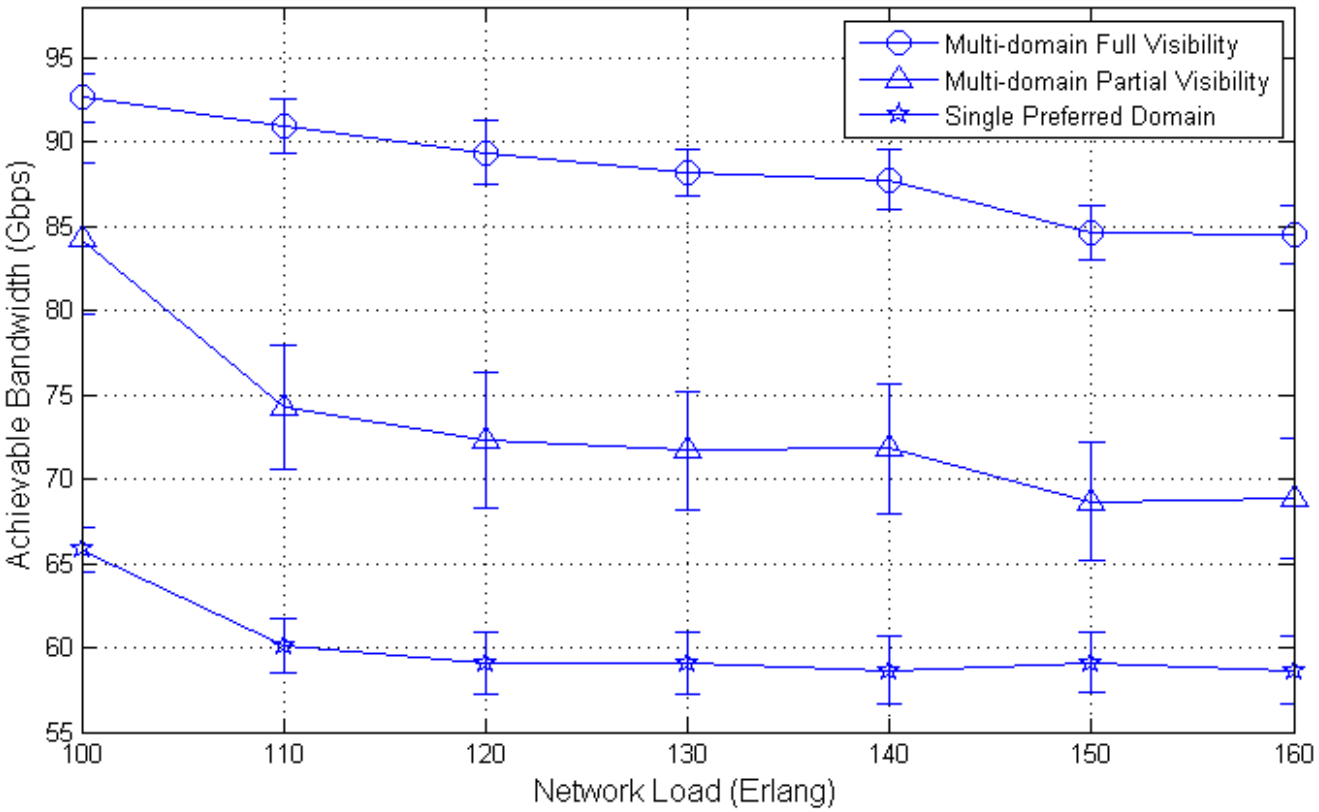
Achievable Bandwidth of Different Algorithms (Germany17)



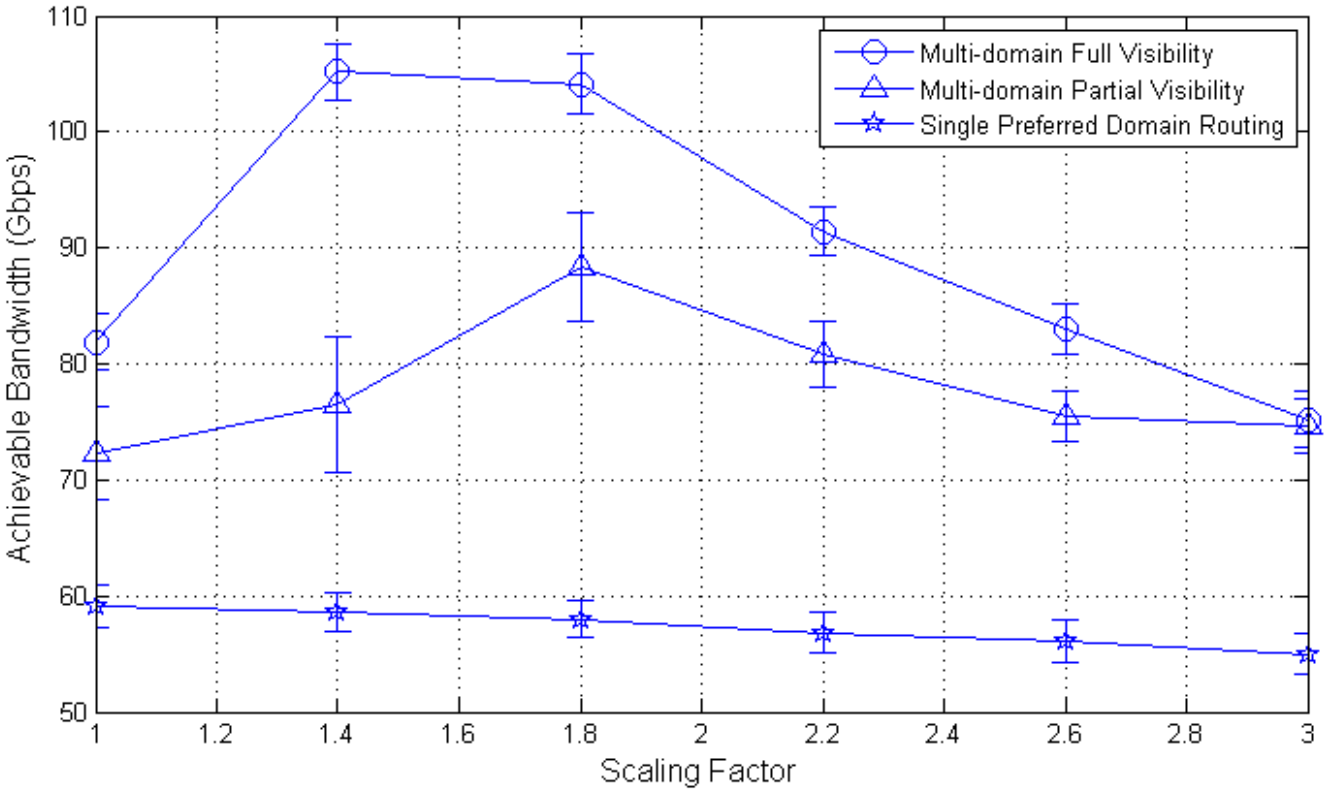
Achievable Bandwidth of Different Schemes With Identical Network Loading In Both Domains (NSFNet and Germany17)



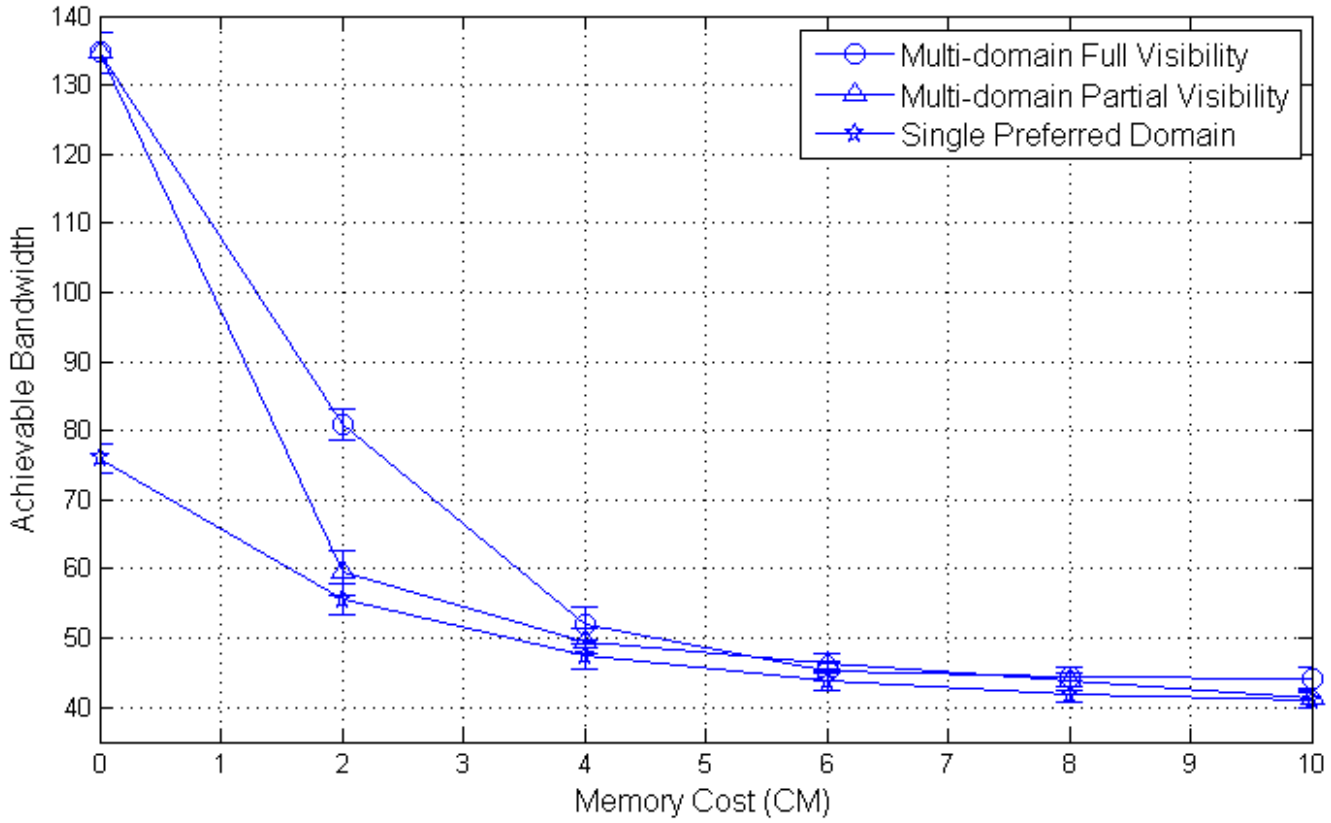
Achievable Bandwidth Of Different Schemes With Skewed Network Loading. The load in the Germany17 is kept constant while the load in the NSFNet is varied



Achievable Bandwidth of Different Schemes vs Delay Scaling Factor. The link delay in the NFSNet is scaled while that of the Germany17 is kept constant



Achievable Bandwidth Of Different Schemes vs Memory Cost (NSFNet and Germany17)



Proposed solutions

- Multi-path routing to maximize bandwidth between a pair of nodes for applications requiring high data transfer rates.

- Multi-path computation algorithm which calculates the first N shortest-widest paths.
- ILP formulation which can maximize the achievable bandwidth while minimizing the memory cost.

Results

- Multi-path routing can achieve significantly higher bandwidth as compared to traditional single path routing.
- multi-domain multi-path routing schemes can achieve significant increase in bandwidth even with partial visibility in a multi-homing environment.

Future Work

- Decrease the complexity of multipath calculation.
- Develop heuristic algorithms to find near-optimal solutions.

Questions?

Thank you!

andred@ic.unicamp.br