Utility Maximizing Routing to Data Centers

M. Sarwat, J. Shin and S. Kapoor
(Presented by J. Shin)

Sep 26, 2011

Outline

1. Problem Definition - Data Center Allocation
2. How to construct Data Center Allocation
3. Mathematical Model
4. Greedy Approach - single server (GA-I)
5. Greedy Approach - multiple server (GA-II)
6. Auction Algorithm (Auction)
7. Simulation Results : utility, congestion, exe. time
8. Conclusion

Problem Definition

- Objective : Maximize the utility of users.
- Constraint :
  1. *Supply Constraint* : Each source (or data/service center) i has a *supply* of goods, bounded by a capacity function a_i.
  2. *Budget Constraint* : Each sink (or user) j has a bound on the incoming set of data, specified by a *budget* b_j.

Problem Definition - Example

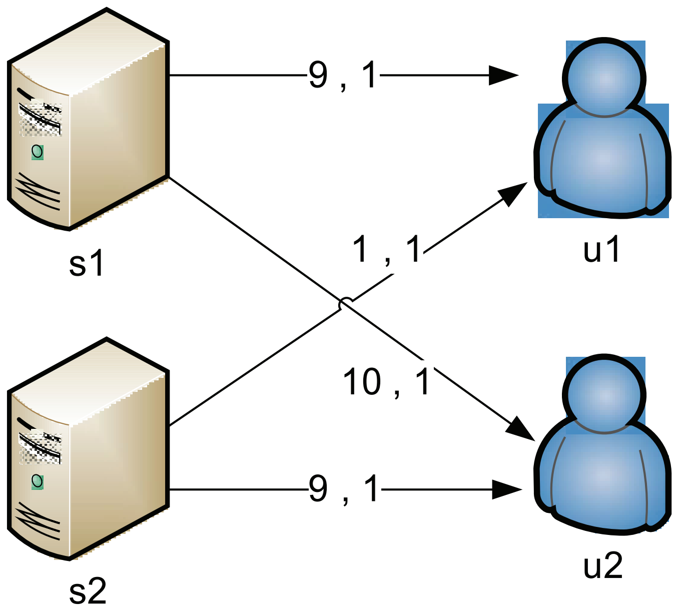

Figure 1: One instance of Data Center Allocation Problem

Network Topology for Simulation

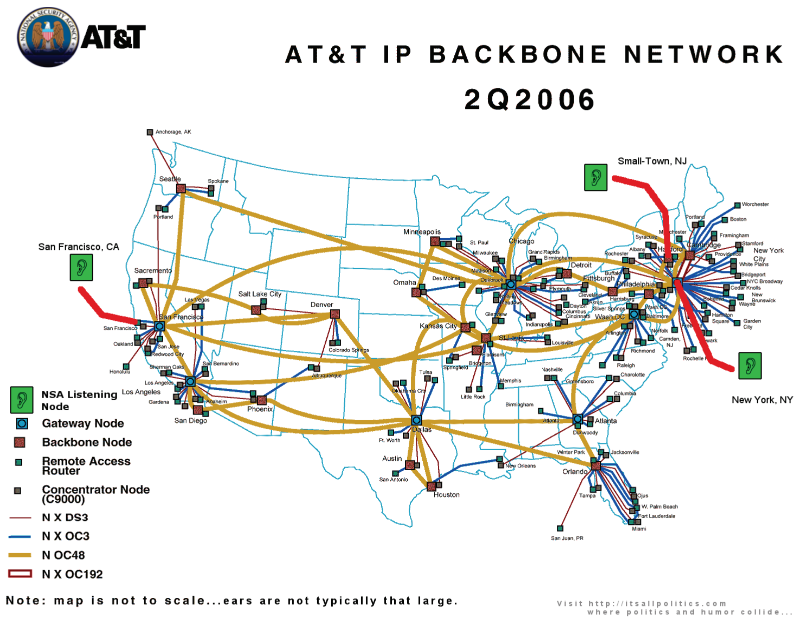

Figure 2: AT&T Network Topology

To Data Center Allocation Problem

- Convert the original problem into Data Center Allocation Problem
- Defined on a Weighted Bipartite Graph.
- Given a pair of data center i and user j, add edge e_{ij} between i and j with p_{ij} (p_{ij} represents the price of utilizing the path). One such measure is the number of hops between i and j.
- u_{ij} represents the benefit of using the path. One such benefit is the minimum capacity amongst of edges on the path.

Mathematical Model - Primal

P: Maximize $\sum_{i \in S, j \in T} u_{ij} f_{ij}$ subject to

$$\sum_{j \in T} f_{ij} \leq a_i \quad \forall i \in S \qquad (1)$$

$$\sum_{i \in S} p_{ij} f_{ij} \leq b_j \quad \forall j \in T \qquad (2)$$

$$f_{ij} \geq 0 \quad \forall i \in S, j \in T \qquad (3)$$

- u_{ij}, p_{ij} : utility and price using i by j
- f_{ij} : amount of data from i to j
- a_i, b_j : capacity function of i and budget function of j

Mathematical Model - Dual

The dual of (P) is as follows denoted by (D).

D: Minimize $\sum_{i \in S} \alpha_i a_i + \sum_{j \in T} \beta_j b_j$ subject to

$$\alpha_i \geq u_{ij} - p_{ij}\beta_j \quad \forall i \in S, j \in T \qquad (4)$$

$$\alpha_i \geq 0 \quad \forall i \in S \qquad (5)$$

$$\beta_j \geq 0 \quad \forall j \in T \qquad (6)$$

α_i and β_j represent dual variables.

GA-I - High Level Idea

- *Objective :* Selects a data center which maximizes their utility.
- *Constraint:* Each user uses at most a single path.

1: Let us start with $j = 1$ in a round robin way.
2: Pick i s.t. utility is maximized.
3: Calculate current amount of data from i.
4: Calculate budget of user j.
5: Assign i to j under those constraints.

The algorithm iterates at most n_u times.

GA-I - Pseudocode

for $j = 1 \to n_u$ **do**

  $i := \max_k u_{kj}/p_{kj}$

  $t_f := \sum_k f_{ik}$

  $t_p := \sum_k p_{kj} f_{kj}$

  $x := \min(a_i - t_f, (b_j - t_p)/p_{ij})$

end for

GA-II - High Level Idea

- *Objective :* Selects a data center maximizing their utility.
- *Constraint:* Each user is able to access multiple servers.

1: Find (i, j) s.t. maximizing utility.
2: Calculate current amount of data from i.
3: Calculate budget of user j.
4: Assign i to j under those constraints.

The algorithm may iterate for all pairs.

GA-II - Pseudocode

$\forall i, j : visited_{ij} := false$

for $i = 1 \to n_d$ **do**

  for $j = 1 \to n_u$ **do**

    $(s_i, t_j) := \arg\max_{(i,j)|visited_{ij}=false} u_{ij}/p_{ij}$

    $visited_{s_i t_j} := true$

    $f_s := \sum_k f_{s_i k}$ and $p_t := \sum_k p_{kt_j} f_{kt_j}$

    $f_{s_i,t_j} := \min(a_{s_i} - f_s, (b_{t_j} - p_t)/p_{s_i t_j})$

  end for

end for

Greedy Approach is Not Enough!

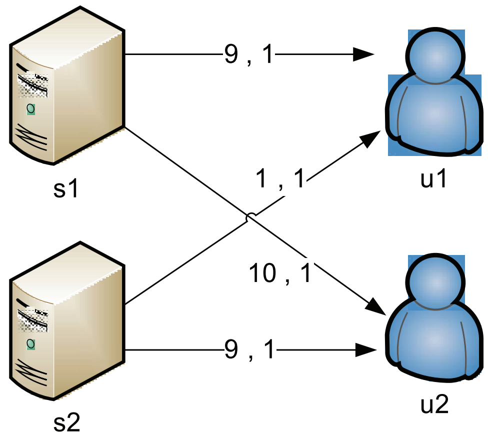

Figure 3: A counter example of Auction vs. GA-I

Let $a_1 = a_2 = 1$, $b_1 = b_2 = 1$, $p_{11} = p_{12} = p_{21} = p_{22} = 1$.
Let $u_{11} = u_{22} = 9$, $u_{12} = 10$ and $u_{21} = 1$.

- $\sum_{i=1}^{2} \sum_{j=1}^{2} u_{ij} f_{ij} = 11$ vs. $\sum_{i=1}^{2} \sum_{j=1}^{2} u_{ij} f_{ij} = 18$

What is Auction?

An auction algorithm is an intuitive method for solving the classical assignment problem.
Suppose there are n agents with budget e_i and m goods.
At each iteration, an agent will increment his bid to acquire his preferred good, and it terminates when each agent are satisfied.

- Users bid up to their own budget allowed to maximize their utility.
- Data centers assigns a user an edge with available supply (capacity).

Auction - 1

```
while ∃i s.t. α_i > 0 and Σ_j f_ij < a_i do
    Pick j : max_j(u_ij − p_ij β_j)
    if Σ_i p_ij f_ij = b_j then
        Find a source i' : y_i'j = β'_j
        Shift flow from i' to i
    else
        Sink is unsaturated, push max. flow possible
        Under supply and demand constraint
    end if
end while
```

while $\exists i$ s.t. $\alpha_i > 0$ and $\sum_j f_{ij} < a_i$ **do**

    Pick $j : \max_j(u_{ij} - p_{ij}\beta_j)$

    if $\sum_i p_{ij} f_{ij} = b_j$ **then**

        Find a source $i' : y_{i'j} = \beta'_j$

        Shift flow from i' to i

    else

        Sink is unsaturated, push max. flow possible

        Under supply and demand constraint

    end if

end while

Auction - 2

if $\beta_j = 0$ **then**

  $\beta_j \leftarrow \epsilon \max_i u_{ij}/p_{ij}$

else if $\forall i : f_{ij} > 0, y_{ij} = \beta_j$ **then**

  $\beta'_j \leftarrow \beta_j$

  $\beta_j \leftarrow \beta_j(1 + \epsilon)$

else

  //no update required

end if

Simulation Results - Environment

- Environment : A Intel Core 2 Duo(2 GHZ) processor, 2 GB memory and Windows 32-bit operating system.
- Compiler : Visual Studio 2008 on a Windows operating system.
- Parameters : Given a_i, b_j, p_{ij}, u_{ij} and $\epsilon = 0.001$.
- Output : average 10 different instances of the same parameter(utility,congestion,exeTime)

Simulation Results - E-I

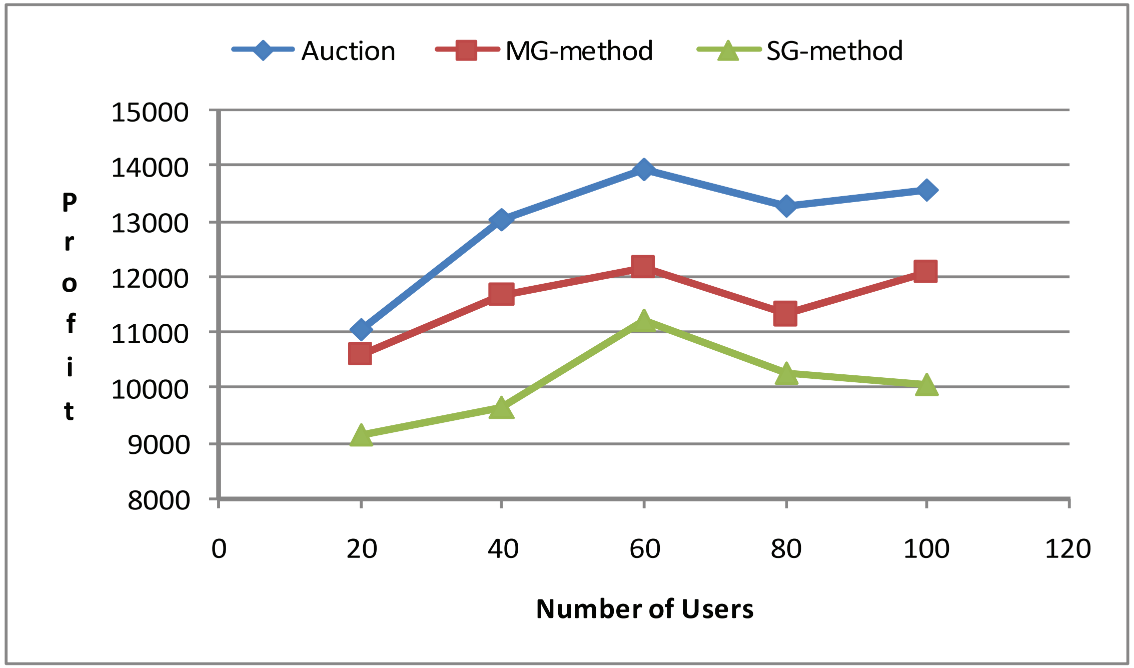

Figure 4: Auction vs. GA-II vs. GA-I -Utility

Simulation Results - E-I

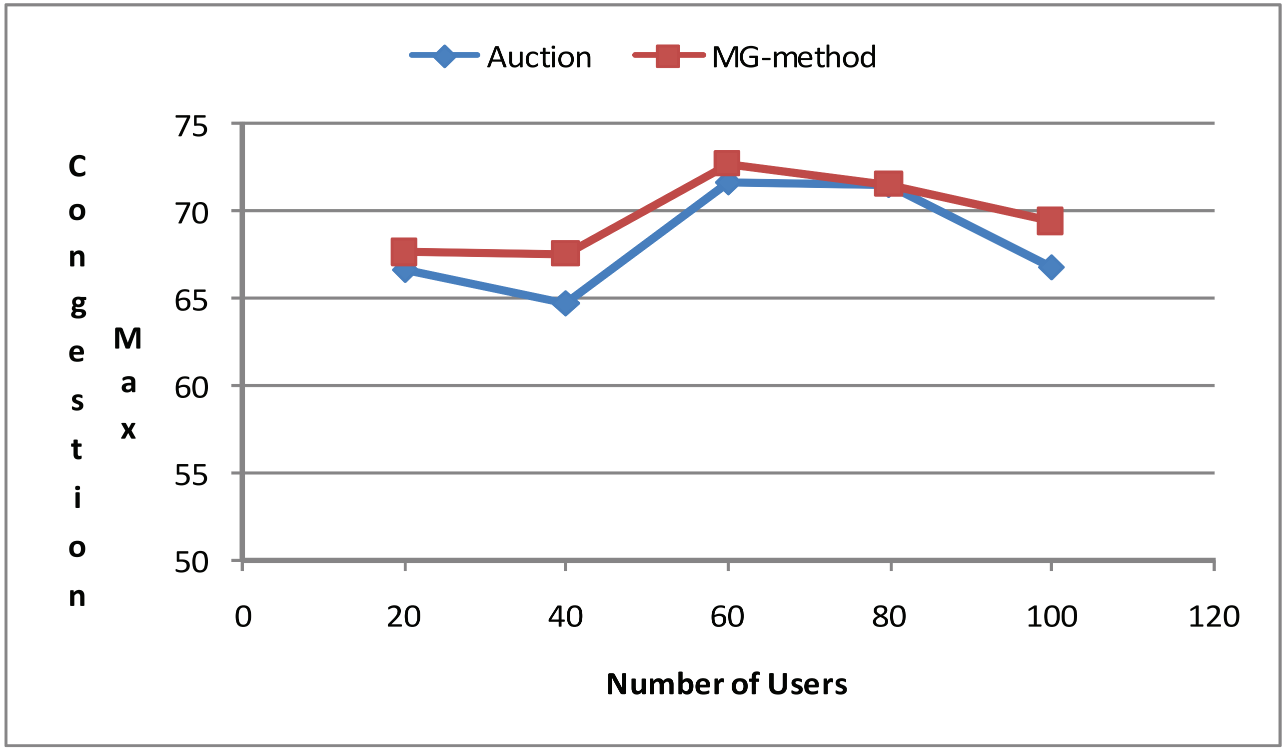

Figure 5: Auction vs. GA-II - Congestion

Simulation Results - E-I

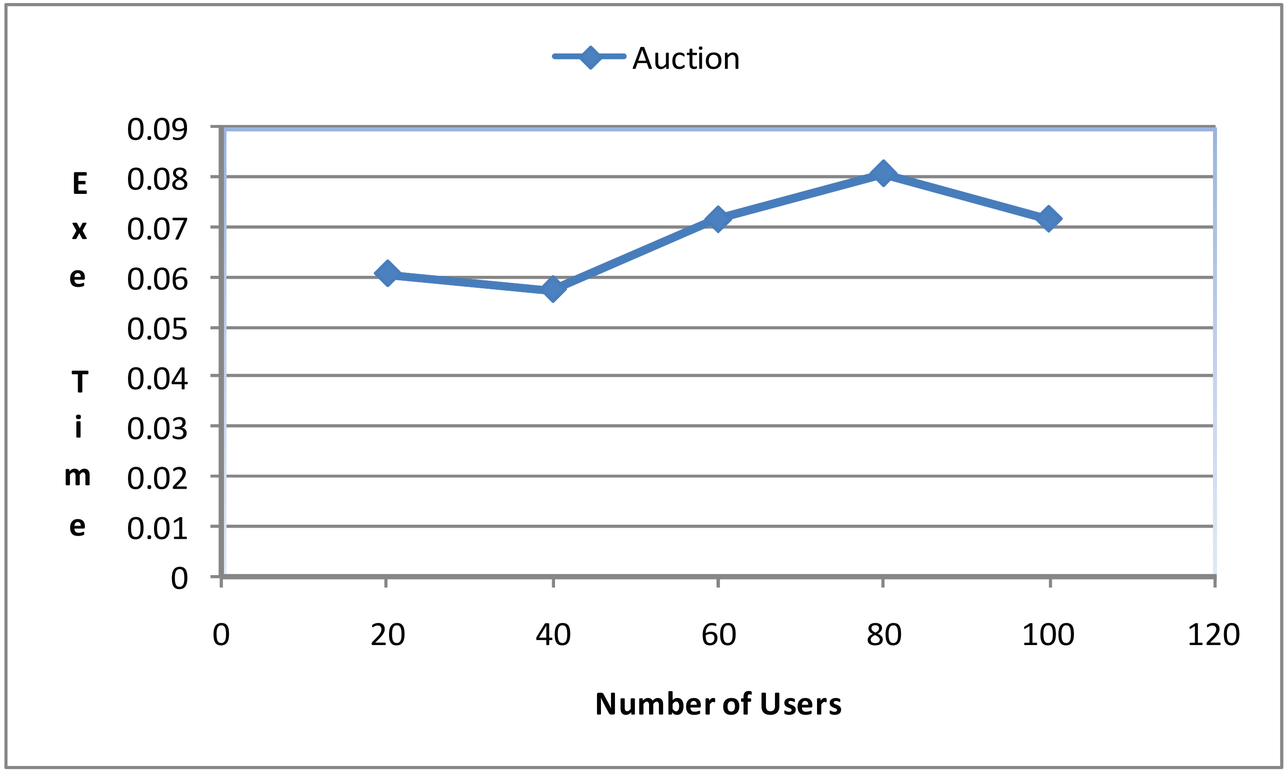

Figure 6: Execution Time of Auction in Seconds

Simulation Results - E-II

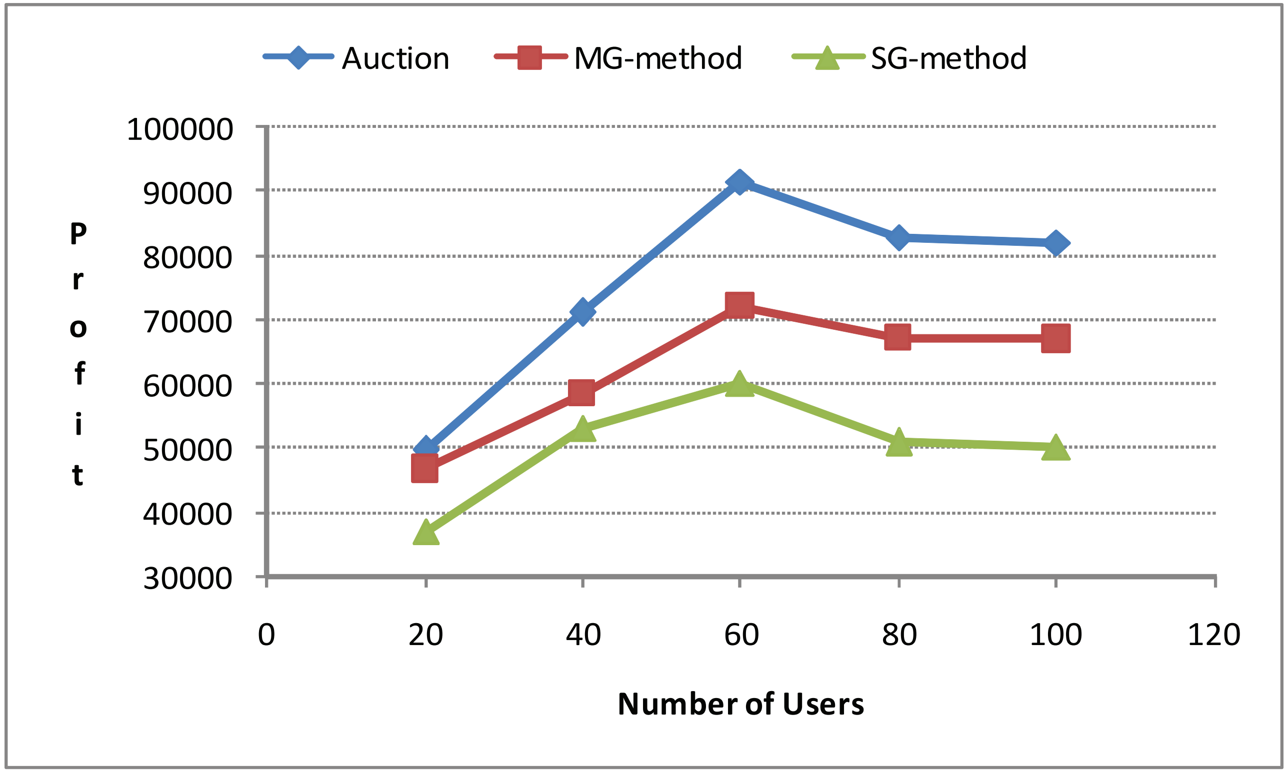

Figure 7: Auction vs. GA-II vs. GA-I - Utility

Simulation Results - E-II

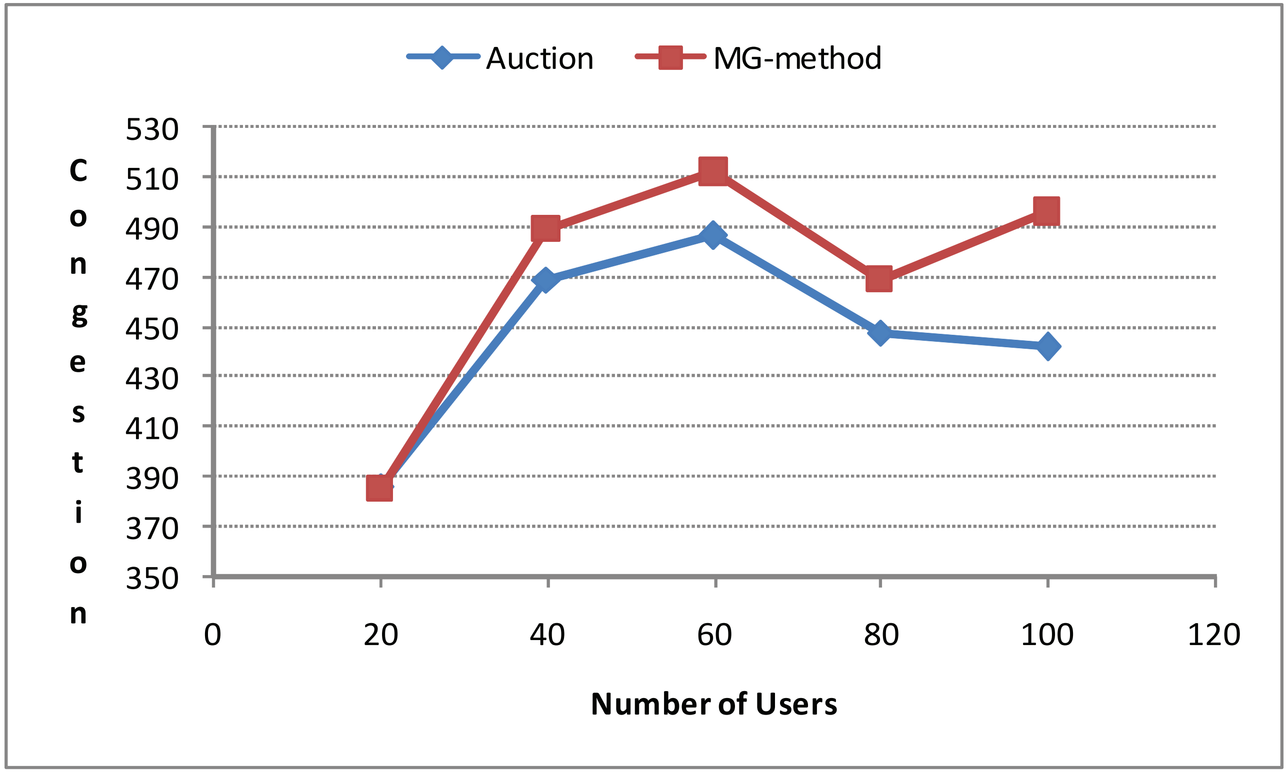

Figure 8: Auction vs. GA-II - Congestion

Simulation Results - E-II

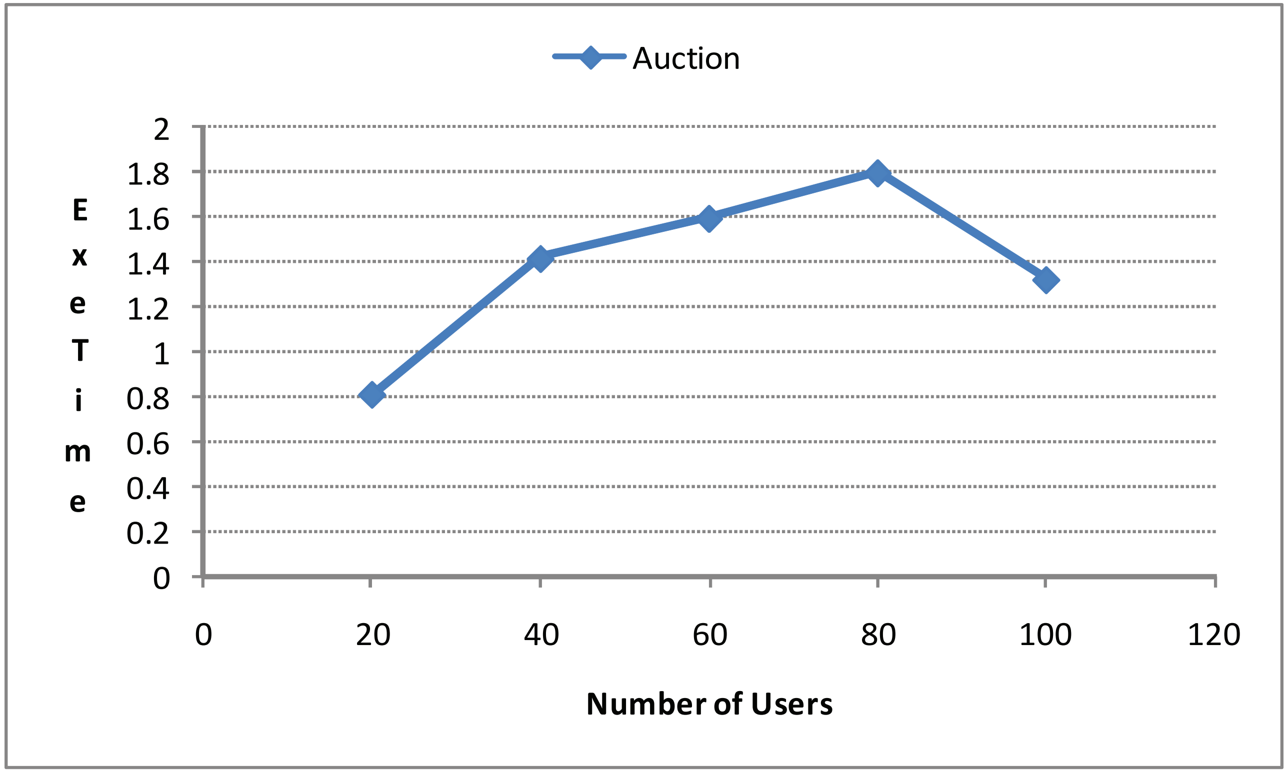

Figure 9: Execution Time of Auction in Seconds

Conclusion

1. Greedy approach is easy to implement and fast, but as expected the method does not match up to the optimal.
2. Since the auction approach is quite efficient, the usage of the method for optimally routing data center traffic is also practical.
3. Further improvements in the efficiency of the algorithm are possible.
4. Future work would extend the algorithm to non-linear objective functions.