JOINT ROUTING AND PLACEMENT OF VIRTUAL NETWORK FUNCTIONS

Jorge Crichigno[1,2], D. Oliveira[3], M. Pourvali[3], N. Ghani[3] , D. Torres[2]

[1]University of South Carolina, SC, USA

[2]Northern New Mexico College, NM, USA

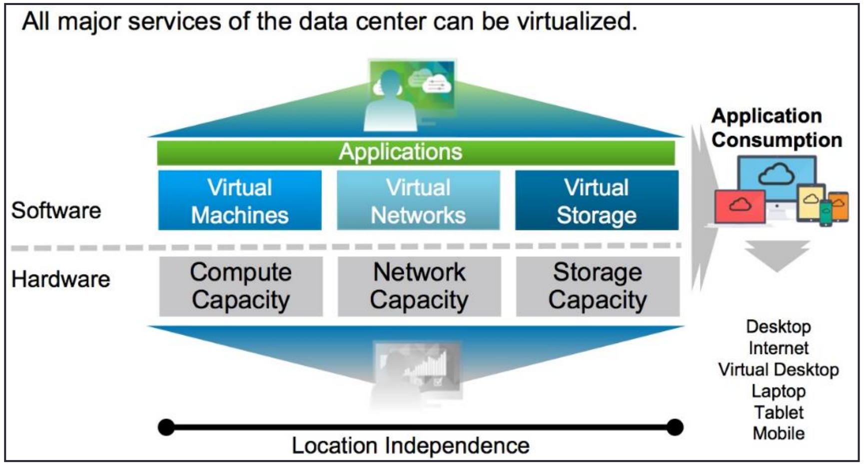
[3]University of South Florida, FL, USA

Agenda

- Introduction
- Optimization model
- Numerical examples
- Concluding remarks

Introduction

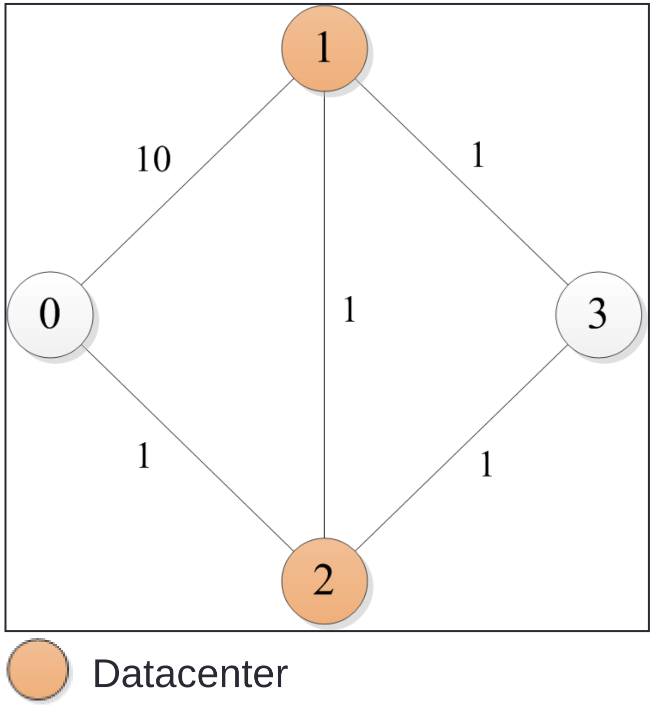
- Network Function Virtualization (NFV) is a technology that permits the implementation of Network Functions (NFs) on datacenters' commodity servers
- Network functions include
  - Firewall, access control lists
  - Routers, switches, NAT, DHCP



http://www.vmware.com/

Introduction

- Consider the weighted network below
- A set of datacenter that implement particular functions
- There is a set of function $F = \{0, 1\}$
- A client request is interested in both functions to apply them to a flow from ingress switch 0 to egress switch 3
- A datacenter *d* implements $F_d \subseteq F$
- The cost and resources to implement a function are datacenter-dependent
- What should the path of the flow be, in order to minimize the routing and deployment costs?

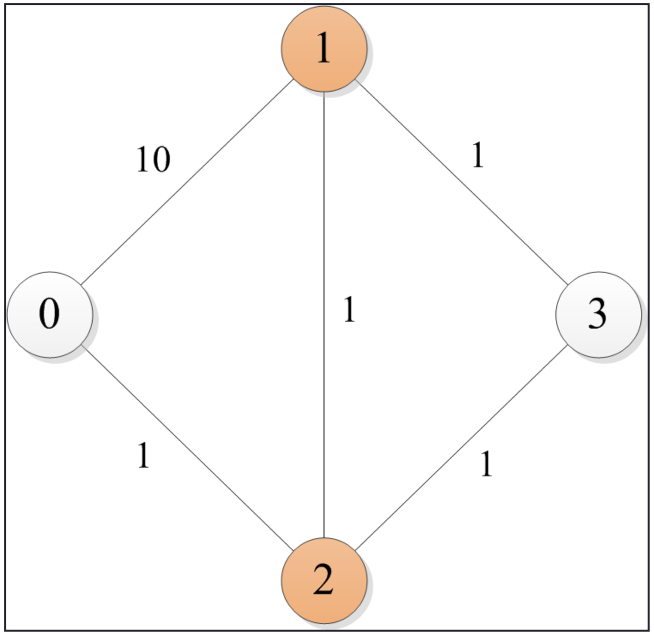


Introduction

Example:

- Datacenter 1 implements functions 0 and 1 at costs 1 and 10
- Datacenter 2 implements functions 0 and 1 at costs 10 and 1

Function 0 at cost 1
Function 1 at cost 10



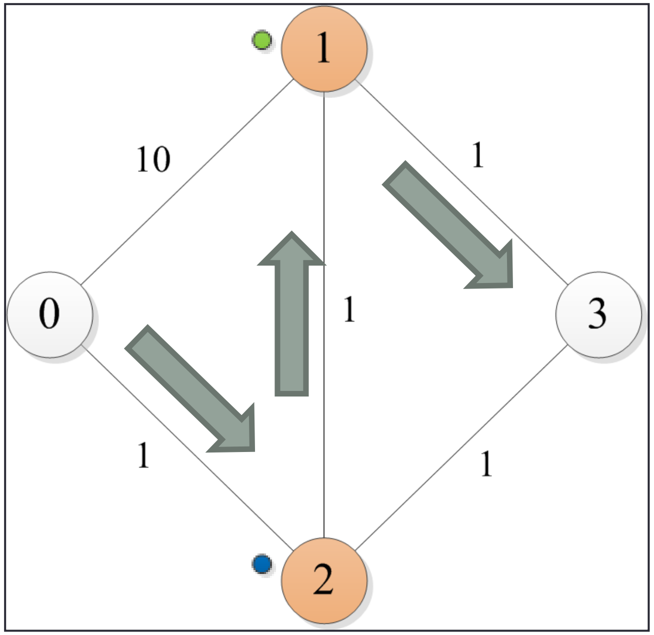
Function 0 at cost 10
Function 1 at cost 1

Introduction

Example:

- Datacenter 1 implements functions 0 and 1 at costs 1 and 10
- Datacenter 2 implements functions 0 and 1 at costs 10 and 1
- The optimal solution places functions 0 and 1 at datacenters 1 and 2 respectively, and route the traffic through (0, 2), (2, 1), (1,3)

Function 0 at cost 1
Function 1 at cost 10



Function 0 at cost 10
Function 1 at cost 1

Function 0 instance
Function 1 instance

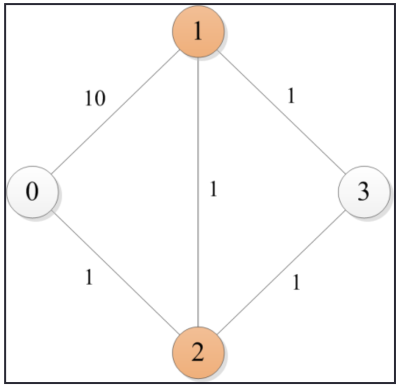
Optimization Model

- The network is represented as a graph $G = (V, E)$
- Each link $(i, j) \in E$ has an associated cost c^{ij}
- The subset $D \subseteq V$ represents the set of datacenters
- A datacenter $d \in D$ implements a subset of functions $F_d \subseteq F$
- Each request $r \in R$ is characterized by a 3-tuple (src_r, dst_r, F_r)
- A datacenter has a set of resources $W = \{w_{d,1}, w_{d,2}, ..., w_{d,m}\}$
- To implement function $i \in F_d$, the datacenter uses $w^i_{d,1}, w^i_{d,2}, ..., w^i_{d,m}$
- The setup cost of an instance $i \in F_d$ is c^i_d
- Each instance $i \in F_d$ can serve up to λ^i_d requests
- Variable $x^i_{r,d}$ indicates whether datacenter d serves function $i \in F_r$ requested by $r \in R$
- Variable y^i_d indicates the number of instances of function i at d
- Variable l^{ij}_r indicates whether link $(i, j) \in E$ is used by flow $r \in R$

Optimization Model

- The objective is the maximization of the number of satisfied network functions (NFs)

$$\text{Max } F_1 = \sum_{r \in R} \sum_{i \in F_d} \sum_{d \in D} x^i_{r,d} = x^0_{0,1} + x^1_{0,1} + x^0_{0,2} + x^1_{0,2}$$



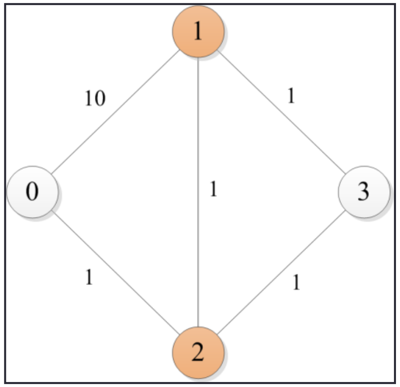
Optimization Model

- The objective is the maximization of the number of satisfied network functions (NFs)

$$\text{Max } F_1 = \sum_{r\in R}\sum_{i\in F_d}\sum_{d\in D} x^i_{r,d} = x^0_{0,1} + x^1_{0,1} + x^0_{0,2} + x^1_{0,2}$$

- Minimization of the NF deployment cost

$$\text{Max } -F_2 = \sum_{d\in D}\sum_{i\in F_d} c^i_d y^i_d = y^0_1 + 10y^1_1 + 10y^0_2 + 1y^1_2$$

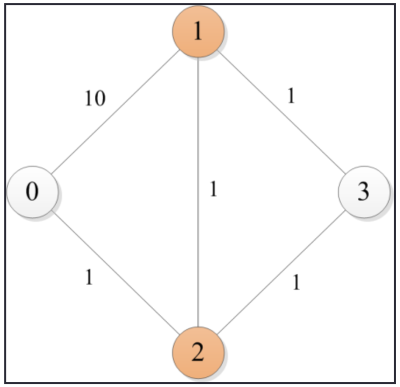


Optimization Model

- Minimization of the routing cost

$$\text{Max - F3} = \sum_{r \in R} \sum_{(i,j) \in E} c^{ij} l_r^{(i,j)} =$$

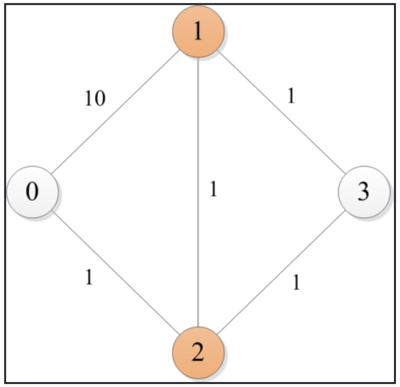
$$10l_0^{(0,1)} + l_0^{(0,2)} + 10l_0^{(1,0)} + l_0^{(1,2)} + l_0^{(1,3)} + l_0^{(2,0)} + l_0^{(2,1)} + l_0^{(2,3)} + l_0^{(3,1)} + l_0^{(3,2)}$$



Optimization Model

- Requested functions 0 and 1 are only implemented in one datacenter

$$\sum_{d \in D} x^i_{r,d} \leq 1 \quad \Rightarrow \quad \begin{aligned} x^0_{0,1} + x^0_{0,2} &\leq 1 \quad \text{Function 0} \\ x^1_{0,1} + x^1_{0,2} &\leq 1 \quad \text{Function 1} \end{aligned}$$

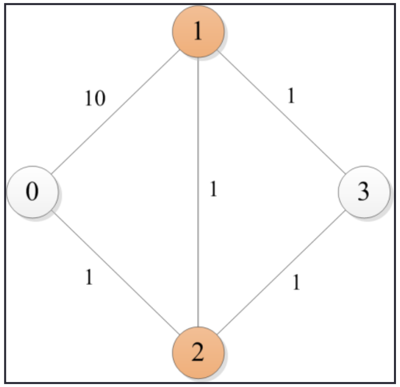


Optimization Model

- The total amount of resources (memory, CPU, storage) is limited at each datacenter
- E.g., 15 and 20 storage units used by an instance of function 0 and 1 respectively at datacenter 1. Datacenter has 100 storage units

$$\sum_{i \in F_d} w^i_{d,j} y^i_d \leq w_{d,j} \quad \Rightarrow \quad 15y^0_1 + 20y^1_1 \leq 100$$

Datacenter 1, storage resource



Optimization Model

- There is a path from the ingress switch 0 to egress switch 3

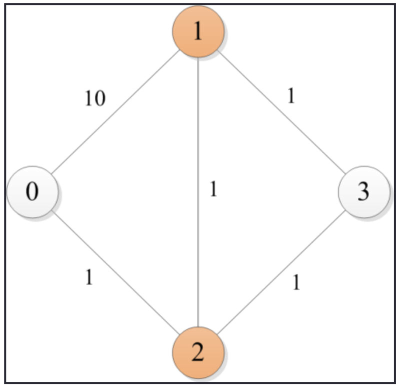
Node 0: $\left(l_0^{(0,1)} + l_0^{(0,2)}\right) - \left(l_0^{(1,0)} + l_0^{(2,0)}\right) = 1$

Node 1: $\left(l_0^{(1,0)} + l_0^{(1,2)} + l_0^{(1,3)}\right) - \left(l_0^{(0,1)} + l_0^{(2,1)} + l_0^{(3,1)}\right) = 0$

Node 2: $\left(l_0^{(2,0)} + l_0^{(2,1)} + l_0^{(2,3)}\right) - \left(l_0^{(0,2)} + l_0^{(1,2)} + l_0^{(3,2)}\right) = 0$

Node 3: $\left(l_0^{(3,1)} + l_0^{(3,2)}\right) - \left(l_0^{(1,3)} + l_0^{(2,3)}\right) = -1$

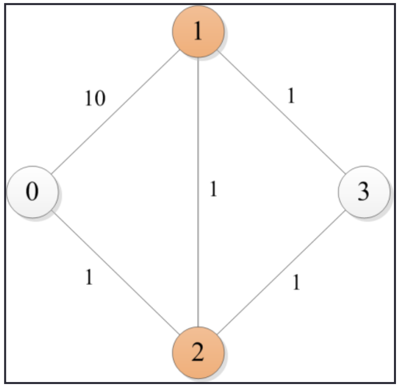
$$\sum_{j:(i,j)\in E} l_{ij}^r - \sum_{j:(j,i)\in E} l_{ji}^r = \begin{cases} -1; i = dst_r,\ src_r \neq dst_r \\ 1; i = src_r,\ src_r \neq dst_r \\ 0; \text{otherwise.} \end{cases}$$



Optimization Model

- If a function 0 is placed at datacenter 1, then the path from the ingress switch 0 to egress switch 3 must include datacenter 1

$$l_0^{(1,0)} + l_0^{(1,2)} + l_0^{(1,3)} \geq x_{0,1}^0$$



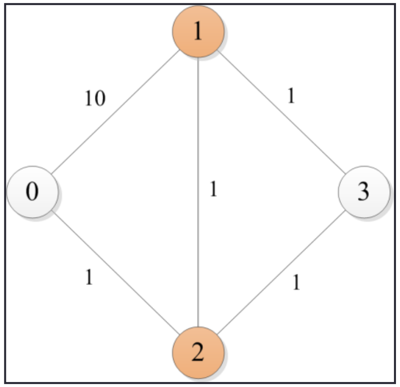
Optimization Model

- If a function 0 is placed at datacenter 1, then the path from the ingress switch 0 to egress switch 3 must include datacenter 1

$$l_0^{(1,0)} + l_0^{(1,2)} + l_0^{(1,3)} \geq x_{0,1}^0$$

- If a function 1 is placed at datacenter 1, then the path from the ingress switch 0 to egress switch 3 must include datacenter 1

$$l_0^{(1,0)} + l_0^{(1,2)} + l_0^{(1,3)} \geq x_{0,1}^1$$



Optimization Model

- If a function 0 is placed at datacenter 1, then the path from the ingress switch 0 to egress switch 3 must include datacenter 1

$$l_0^{(1,0)} + l_0^{(1,2)} + l_0^{(1,3)} \geq x_{0,1}^0$$

- If a function 1 is placed at datacenter 1, then the path from the ingress switch 0 to egress switch 3 must include datacenter 1

$$l_0^{(1,0)} + l_0^{(1,2)} + l_0^{(1,3)} \geq x_{0,1}^1$$

$$\sum_{(d,j)\in E} l_r^{dj} \geq x_{r,d}^i$$

Optimization Model

- Variables $x^i_{r,d}$, y^i_d , l^r_{ij} are binary, integer, and real – NP hard
- For large instances of the problem, finding the optimal solution is not practical

$$\max F = w_1 \sum_{r\in R}\sum_{i\in F_r}\sum_{d\in D|i\in F_d} x^i_{r,d} - w_2 \sum_{d\in D}\sum_{i\in F_d} c^i_d y^i_d - w_3 \sum_{r\in R}\sum_{(i,j)\in E} c^{ij} l^{ij}_r \quad (1)$$

$$\sum_{d\in D} x^i_{r,d} \leq 1 \qquad r\in R, i\in F_r \quad (2)$$

$$x^i_{r,d} \leq y^i_d \qquad r\in R, i\in F_r, d\in D|i\in F_d \quad (3)$$

$$\sum_{i\in F_d} w^i_{d,j} y^i_d \leq w_{d,j} \qquad d\in D, r\in R, j\in\{1,2,...,|W_d|\} \quad (4)$$

$$\sum_{r\in R} x^i_{r,d} \leq \lambda^i_d y^i_d \qquad d\in D, i\in F_d \quad (5)$$

$$\sum_{j:(i,j)\in E} l^r_{ij} - \sum_{j:(j,i)\in E} l^r_{ji} = \begin{cases} -1; i = dst_r,\ src_r \neq dst_r \\ 1; i = src_r,\ src_r \neq dst_r \\ 0; \text{otherwise.} \quad i\in V, r\in R \end{cases} \quad (6)$$

$$\sum_{(d,j)\in E} l^{dj}_r \geq x^i_{r,d} \qquad r\in R, i\in F_r, d\in D|i\in F_d \quad (7)$$

$$x^i_{r,d} \in \{0,1\} \qquad r\in R, i\in F_r, d\in D|i\in F_d \quad (8)$$

$$y^i_d \in Z^+ \qquad d\in D, i\in F_d \quad (9)$$

$$l^{ij}_r \in \{0,1\} \qquad r\in R, (i,j)\in E \quad (10)$$

Greedy Approach

- Greedy approach based on Dijkstra algorithm

Algorithm 1 Greedy Routing and Placement of NFs

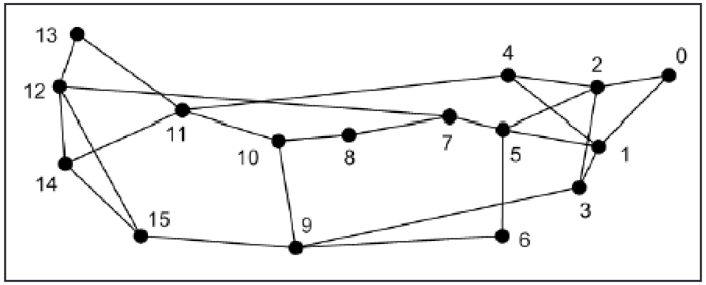
INPUT: $G(V, E), c^{ij} \forall (i, j) \in E, R, F, D$
OUTPUT: $x^i_{r,d}, y^i_d, l^{ij}_r$ values
set $x^i_{r,d} = 0$, $y^i_d = 0$, $l^{ij}_r = 0$ for all $r \in R$, $i \in F_r$, $d \in D$, $(i, j) \in E$
for all $r \in R$ **do**
$D(r) = \{\}$
$k = 1$
for all $i \in F_r$ **do**
$d_k =$ datacenter that implements i at minimum cost and has enough resources to serve an additional request
update resources of d_k
update $y^i_{d_k}$
set $x^i_{r,d_k} = 1$
$D(r) = D(r) \cup d_k$
$k = k + 1$
end for
end for
for all $r \in R$ **do**
$src = src_r$
$C(r) = \{src\}$
for $k = 1$ to $|D(r)|$ **do**
$dst = d_k$
if $d_k \ni C(r)$ **then**
$SP = Dijkstra(src, dst)$
set $l^{ij}_r = 1$ for all link $(i, j) \in SP$
$C(r) = C(r) \cup d_k$
$C(r) = C(r) \cup j$, for all datacenter $j \in SP$, $j \in D(r)$
end if
$src = dst$
end for
$dst = dst_r$
$SP = Dijkstra(src, dst)$
set $l^{ij}_r = 1$ for all $(i, j) \in SP$
end for
return $x^i_{r,d}, y^i_d, l^{ij}_r$

Placement of network functions, one request at a time

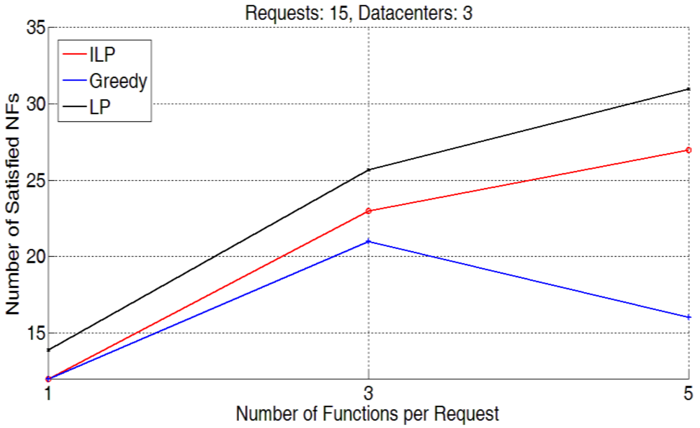
Routing of flows through datacenters implementing the functions, one request at a time

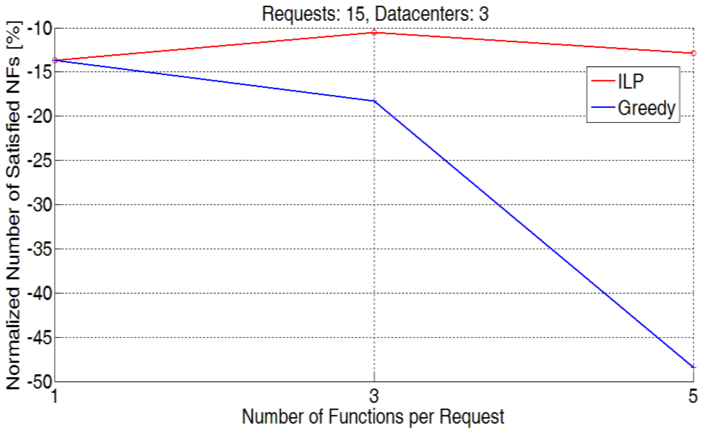
Numerical Examples

- The number of types of resources at a datacenter was set to three (e.g., RAM, storage, CPU)
- The amount of resources of a type at a datacenter is uniform in [.33, 300]
- There are five network functions; each datacenter implements three functions
- The amount of resources of a type needed for an instance of a function is uniform in [0,100]
- The cost of instantiate a function is uniform in [0, 100]
- Datacenters were randomly located in the topology below



Numerical Example 1



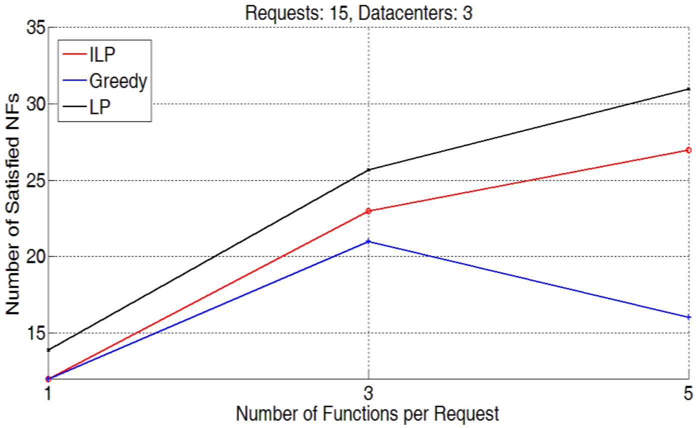


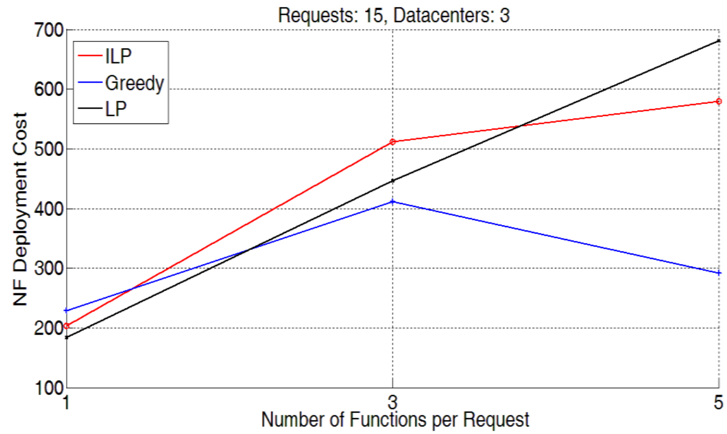
$$\text{Gap} = \frac{ov_{LP} - ov_{alg}}{ov_{LP}}$$

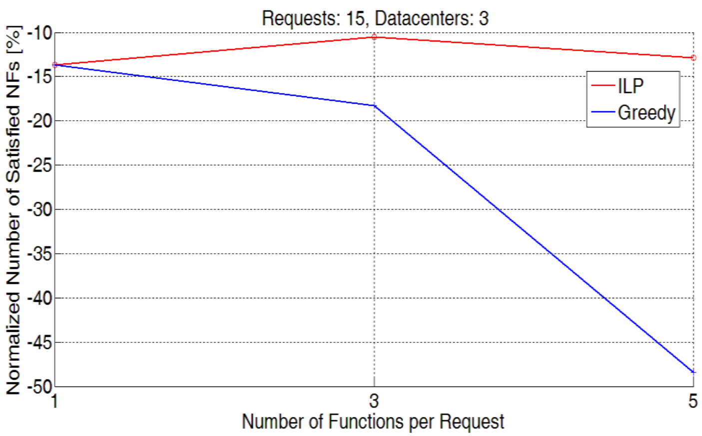
where ov_{LP} is the optimal value obtained with the LP scheme, and ov_{alg} is the optimal value obtained with the ILP or greedy heuristic.

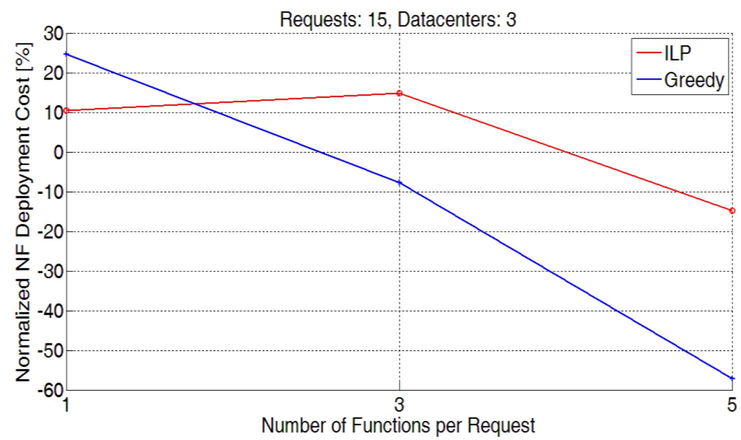
- When there is a small number of datacenters (3) and multiple requests (15), ILP has a comparable performance to that of LP; deployment cost increases with the number of function per request
- The gap of the heuristic increases with the number of function per requests; finding the optimal solution requires the evaluation of a large number of combinations

Numerical Example 1



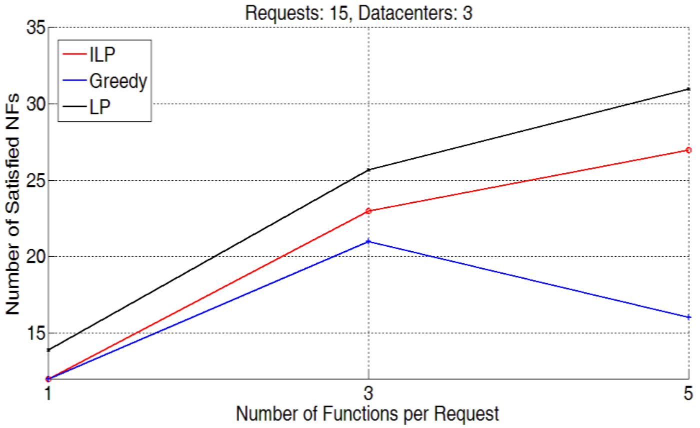


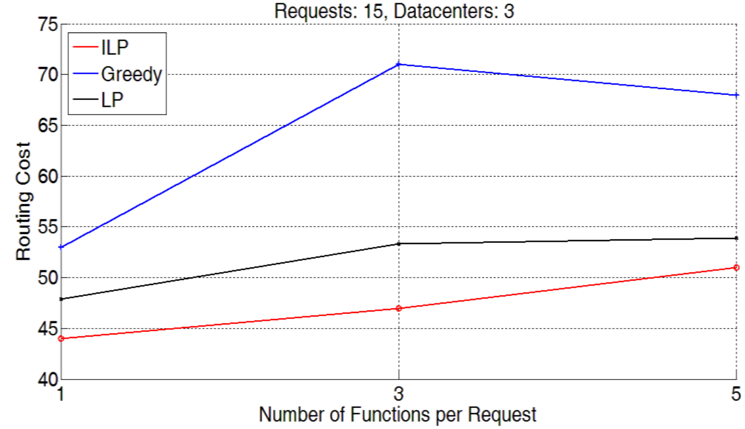


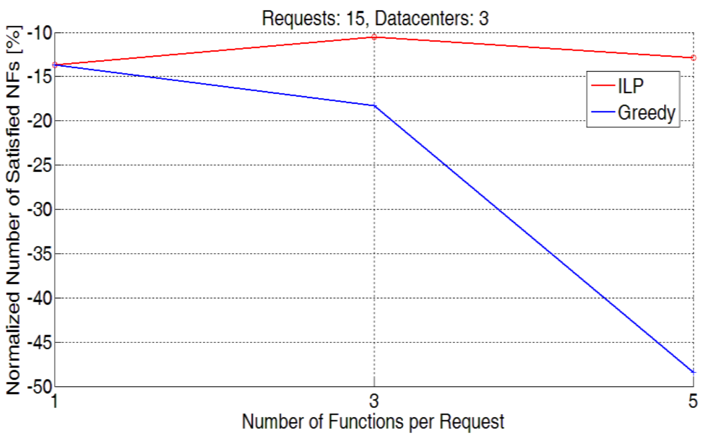


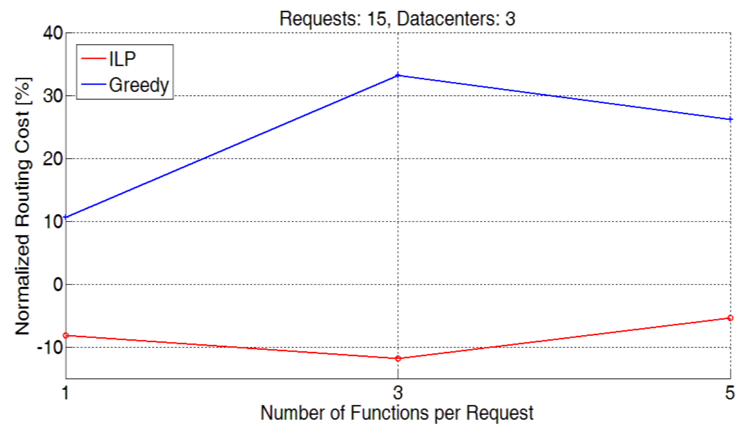
- Deployment cost increases with the number of functions per request

Numerical Example 1



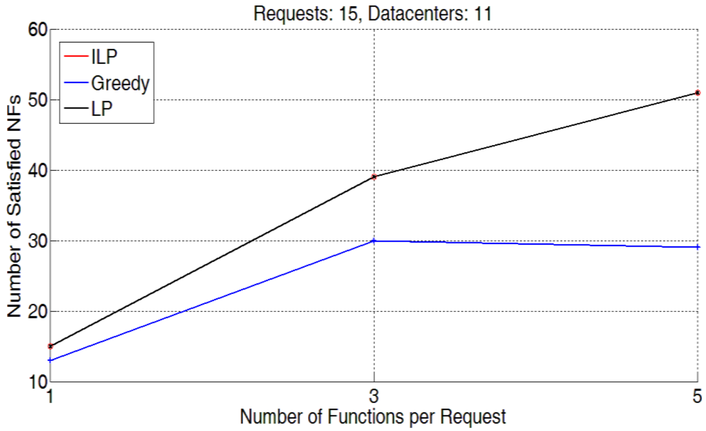


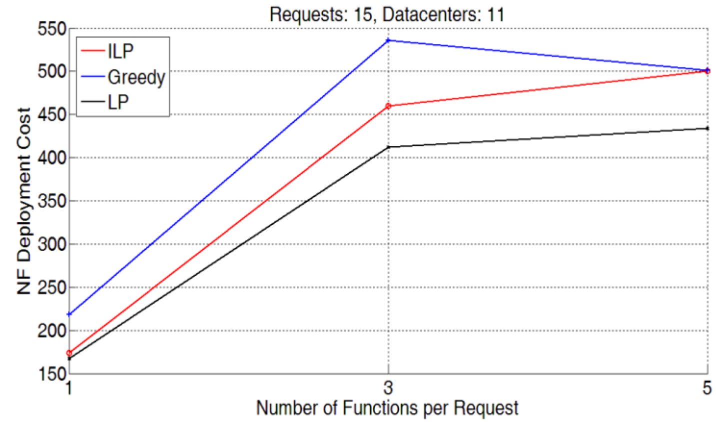


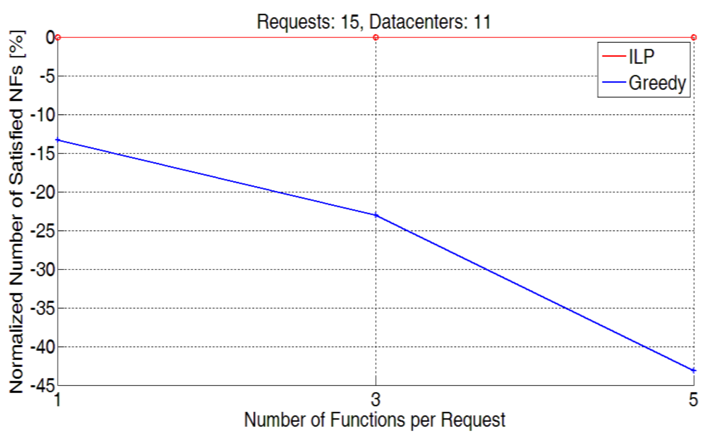


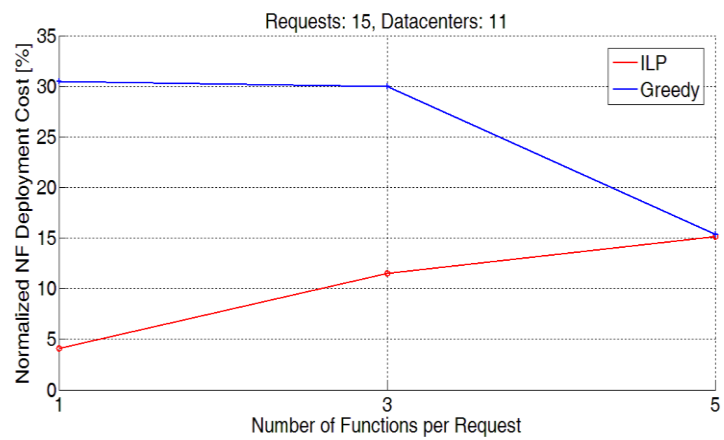
- For LP and ILP, the increase in routing cost is mostly flat; i.e., when the number of datacenters is small, routing is 'less important', because the implementation of functions are concentrated in few datacenters

Numerical Example 2



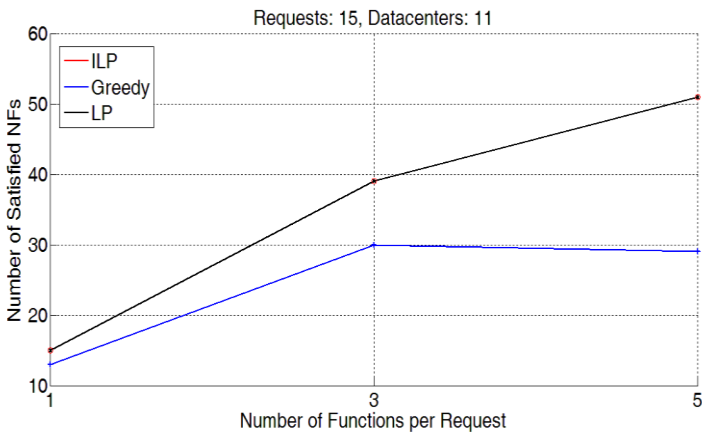


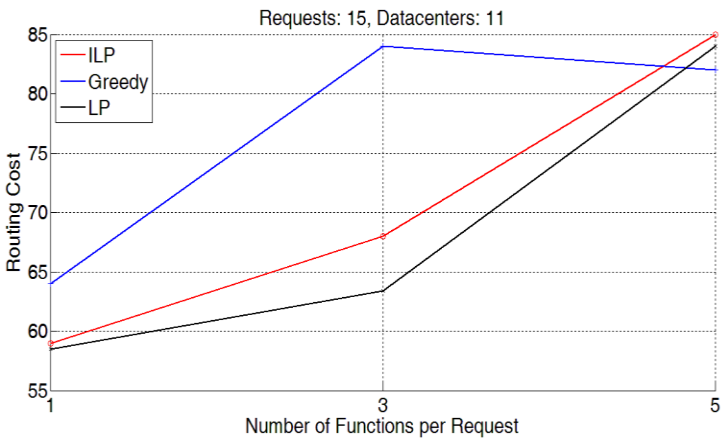


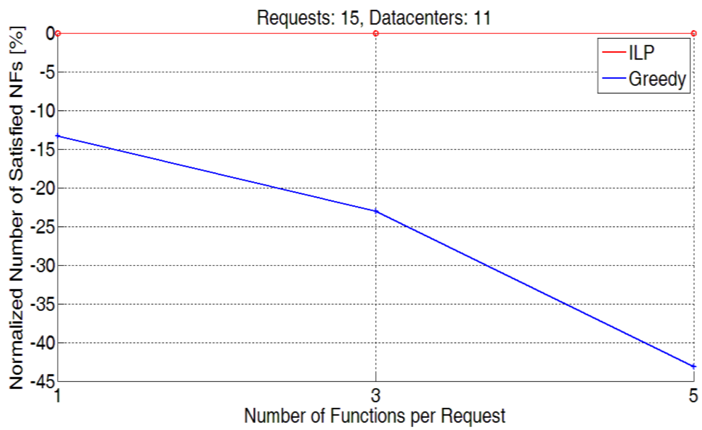


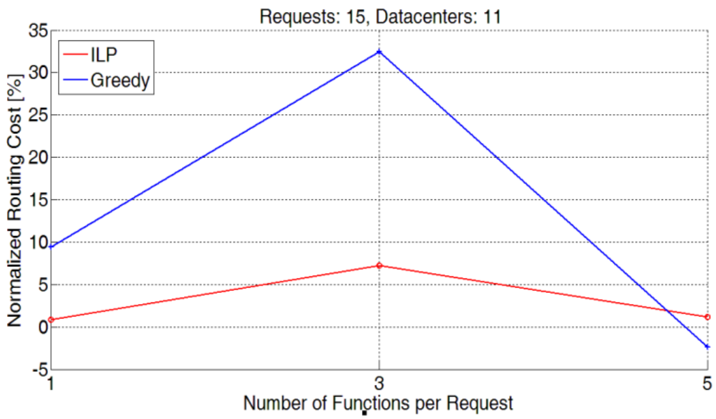
- When there is a large number of datacenters (11) and multiple requests (15), ILP continues to have a comparable performance to LP
- Deployment cost increases substantially when the number of functions per request increases from 1 to 3. However, the increase in cost is minimal when the number of functions per request increases from 3 to 5; i.e., a single instance serves multiple requests without an increase of deployment of functions

Numerical Example 2









- For LP and ILP, the routing cost increases with the number of function per requests; i.e., when the number of datacenters is large, routing is ‘more important’, because the implementation of functions are dispersed in many datacenters

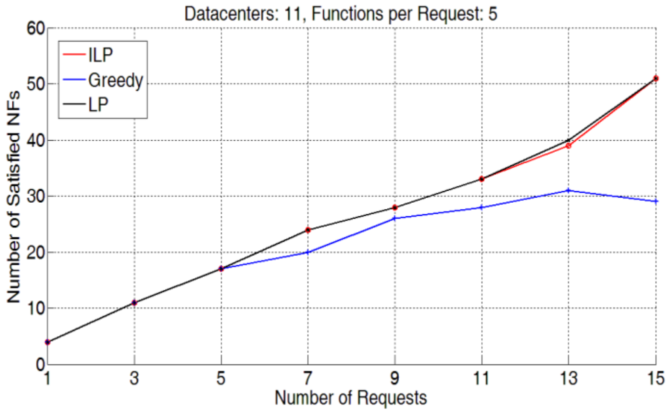
Concluding Remarks

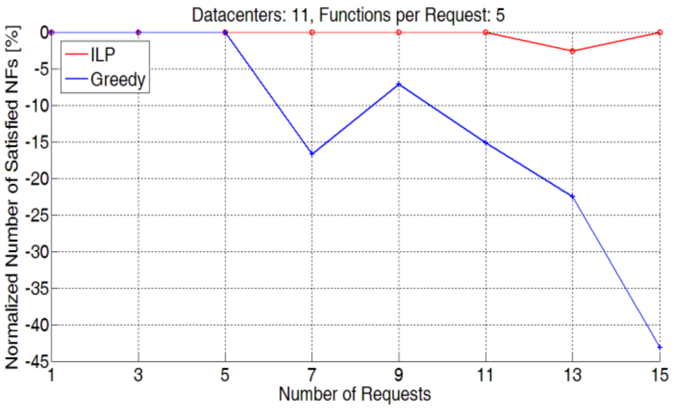
- We are currently working on an optimization scheme for the joint routing and placement of virtual network functions (NFs) problem
- The proposed ILP maximizes the number of satisfied NFs while at the same time minimizes the deployment and routing costs
- A heuristics and ILP are currently being tested
- The implementation of the proposed schemes in small testbeds using ONOS SDN is being implemented

Open Network Operating System (ONOS) SDN, [Online]. Available: http://onosproject.org/

THANK YOU

Numerical Example 3



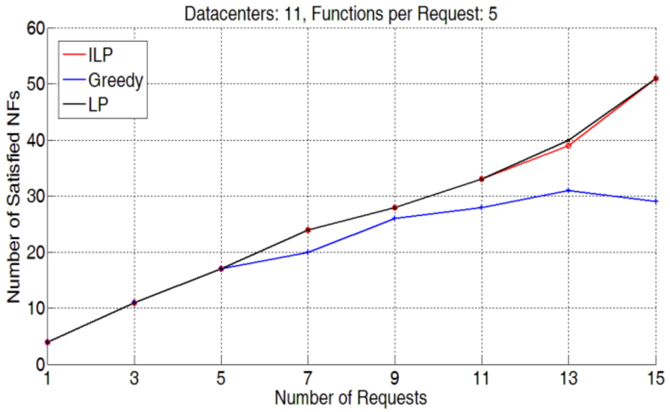


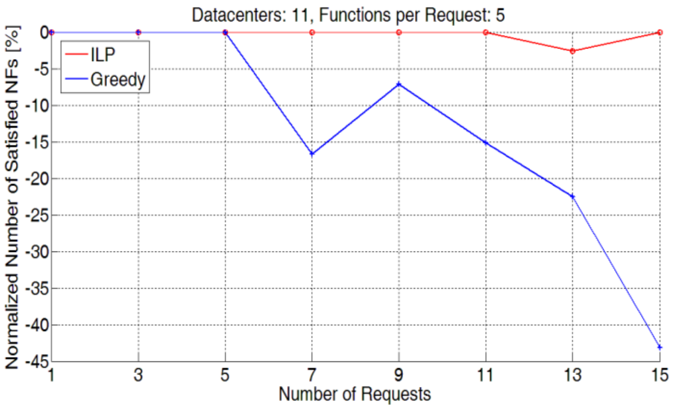
$$\text{Gap} = \frac{ov_{LP} - ov_{alg}}{ov_{LP}}$$

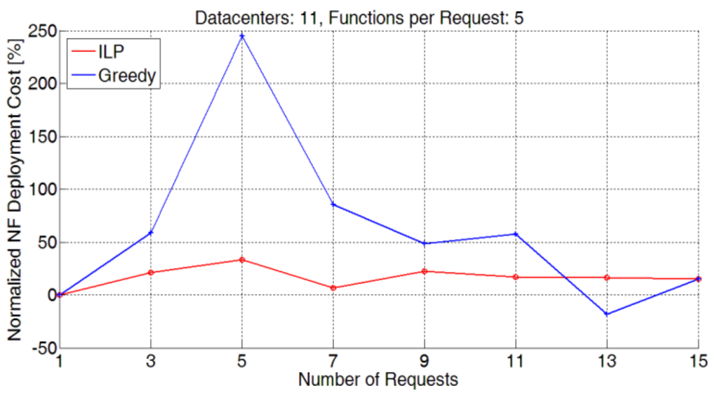
where ov_{LP} is the optimal value obtained with the LP scheme, and ov_{alg} is the optimal value obtained with the ILP or greedy heuristic.

- ILP and LP performances similar; ~2% gap
- As the number of request increases, the heuristic gap substantially increases; finding the optimal solution requires the evaluation of a large number of combinations

Numerical Example 3

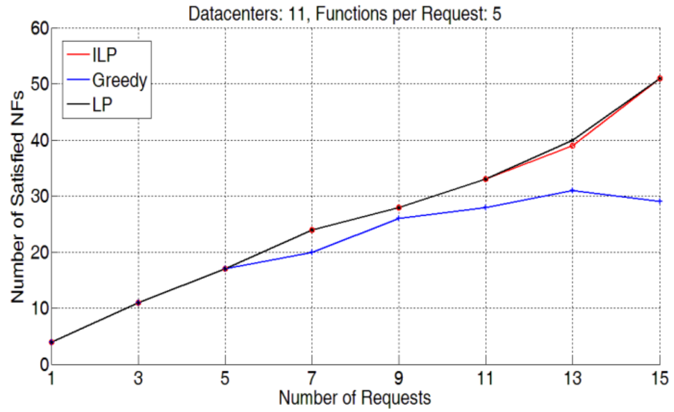


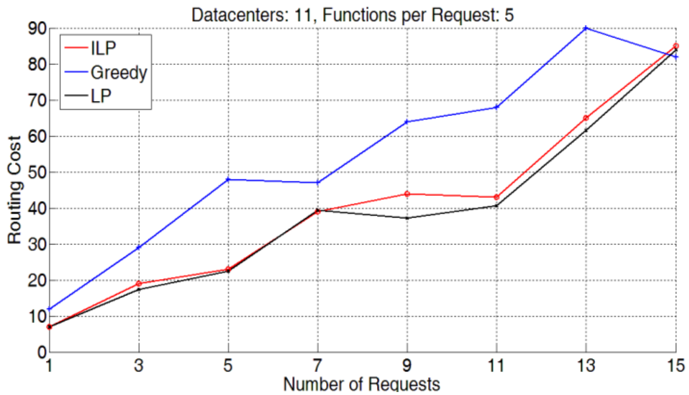


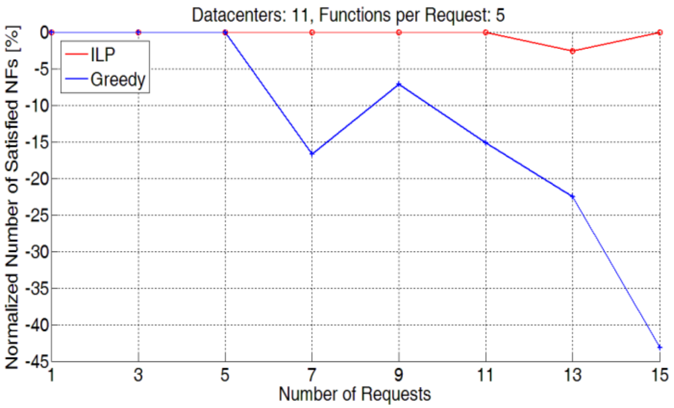


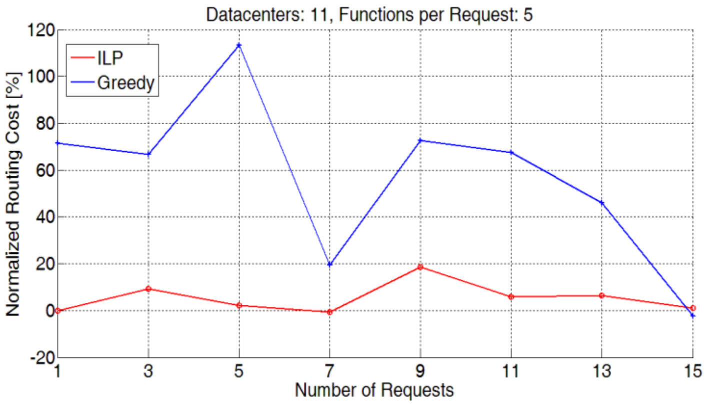
- Deployment cost increases with the number of requests; ILP performance is comparable to that of LP – ’small’ performance gap

Numerical Example 3









- Routing cost increases with the number of requests; ILP performance is comparable to that of LP –'small' performance gap
- While the gap of the routing cost of the greedy approach decreases with the number of requests, the number of satisfied requests is mostly flat

Introduction

About Linear Programming

- Many of the problems for which we want algorithms are *optimization* tasks
- Optimization tasks seek a solution that (1) satisfies certain constraints and (2) is the best, with respect to a criterion
- *Linear programming* describes a broad class of optimization tasks in which both the constraints and the optimization criterion are *linear functions*

Introduction

About Reductions

- Sometimes a computational task is sufficiently general that any subroutine for it can also be used to solve a variety of other tasks, which at glance might seem unrelated
- Once we have an algorithm for a problem, we can use it to solve other problems