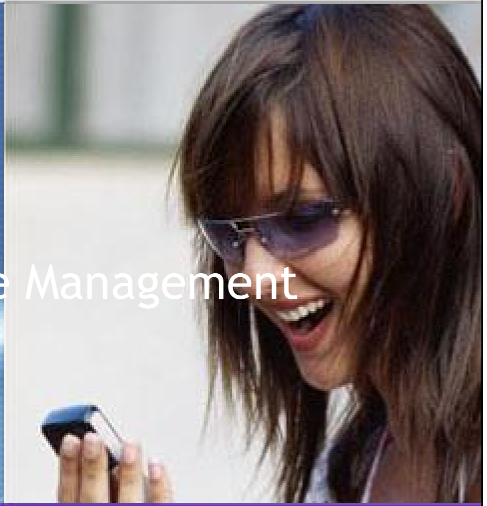
The Personal Cloud

Design, Architecture and Matchmaking Algorithms for Resource Management



Adiseshu Hari, Ramesh Viswanathan, T.V. Lakshman

Bell Labs, NJ

Y. J. Chang

ITRI, Taiwan

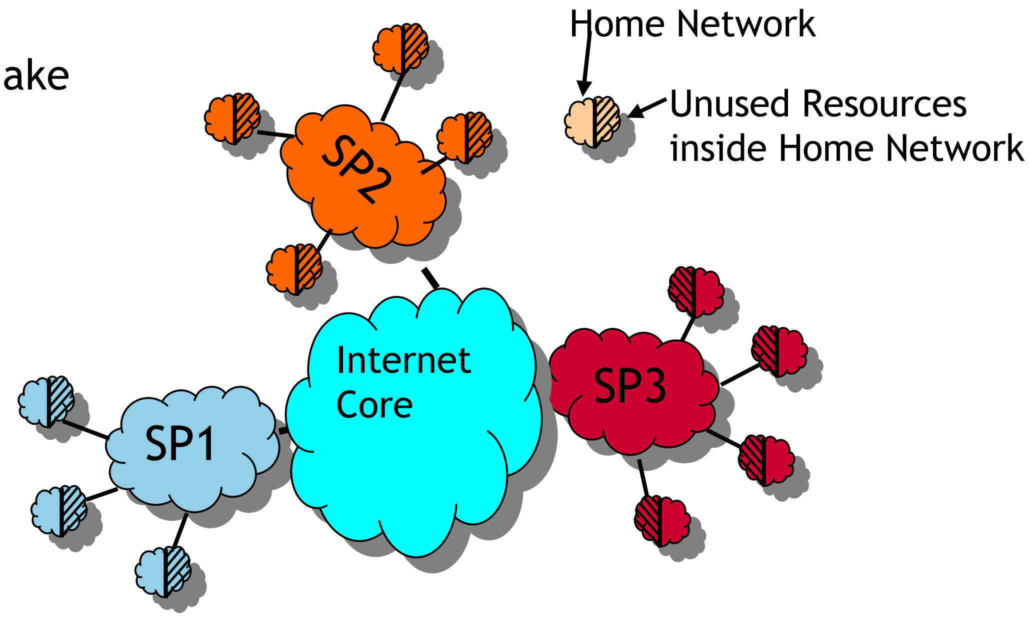
Usenix Hot-ICE Workshop 2012 San Jose, CA

Cloud@Edge

- Phase 1
  - How to share resources at the edge?
- Phase 2
  - How to create, manage and use a Personal Cloud?
- Phase 3
  - Distributed Virtual Edge Cloud

Phase 1

- Observations
  - Everyone has unused computers at home
  - What is the most effective way to make use of them?
  - How to share BW, compute, storage resources?
- Sample Applications
  - Web Server, Data Backup
- Dedicated P2P model
  - No DHT
  - Direct Assignment of resources



Direct Assignment of Resources - Matchmaking Problem

Each node has an **offer** and a **request** for resources

R(i): Number of resource units requested by i

O(i): Number of resource units offered to be hosted by i

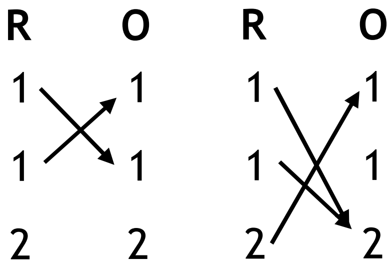
Note – A node cannot use its own resources

Feasible Assignment: Satisfy some of the requests

Request Optimal Assignment: Satisfy as much of the requests as possible

Request Satisfying Assignment: Satisfy all the requests

Greedy does not provide Request Satisfying assignment



Offer Matrix **O** Request Matrix **R**

$$\begin{bmatrix} o_1 \\ o_2 \\ \\ o_n \end{bmatrix} \qquad \begin{bmatrix} r_1 \\ r_2 \\ \\ r_n \end{bmatrix}$$

$$\begin{bmatrix} r_1 \\ r_2 \\ \\ r_n \end{bmatrix} = \begin{bmatrix} m_{11} \; m_{12} \ldots m_{1n} \\ m_{21} \; m_{22} \ldots m_{2n} \\ \\ m_{n1} \; m_{n2} \ldots m_{nn} \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ \\ 1 \end{bmatrix}$$

$\mathbf{R} = \mathbf{MI}$

Request Assignment Matrix **M**

Identity Matrix *I*

Phase 1 – How to share edge resources

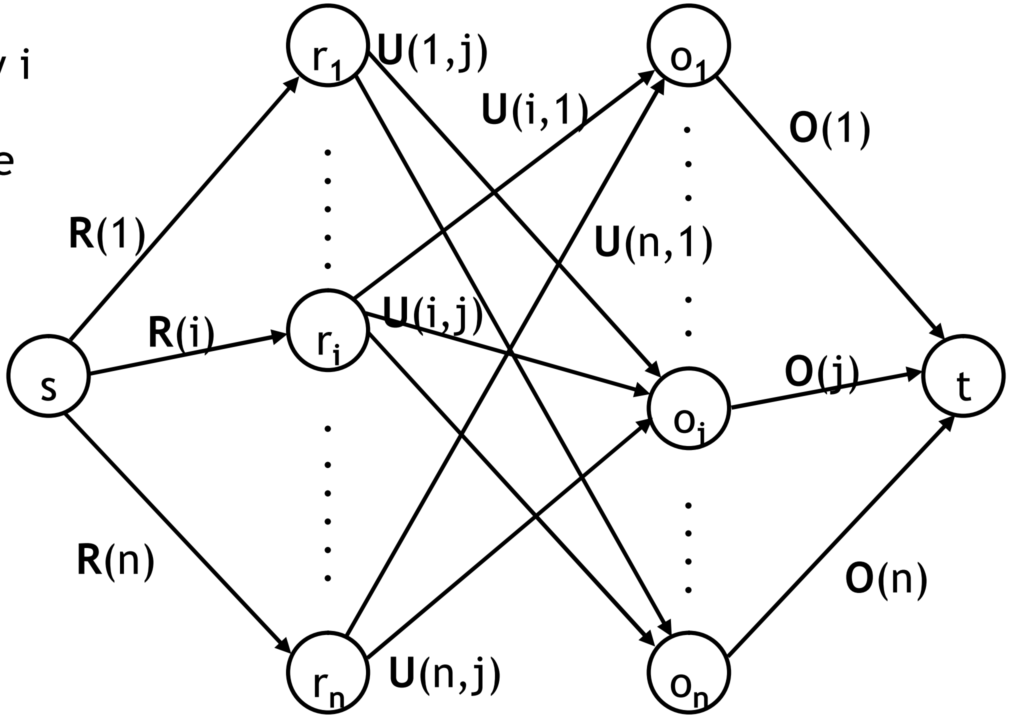
- Matchmaking Issues:
  - N^2 variables
  - Not certain of integral solutions
- App Building Issues
  - Web Server
    - Which Web Server (IIS, Apache,…)
    - Which framework to support (J2EE, PHP…)
    - Backend support?
    - Sandboxing?
  - Data Backup
    - File System or Disk Abstraction?
    - AoE, NBD, iSCSI, NFS, CIFS…?
    - Sandboxing?
  - Observation
    - Build resource sharing infrastructure
    - Layer applications on top
    - Match requests and offers using a Matchmaking algorithm

Phase 2 - Personal Cloud – Matchmaking via Max Flow Reduction

R(i): Number of resource units requested by i

O(i): Number of resource units offered to be hosted by i

U(i,j): Maximum number of i's request that can be hosted at j



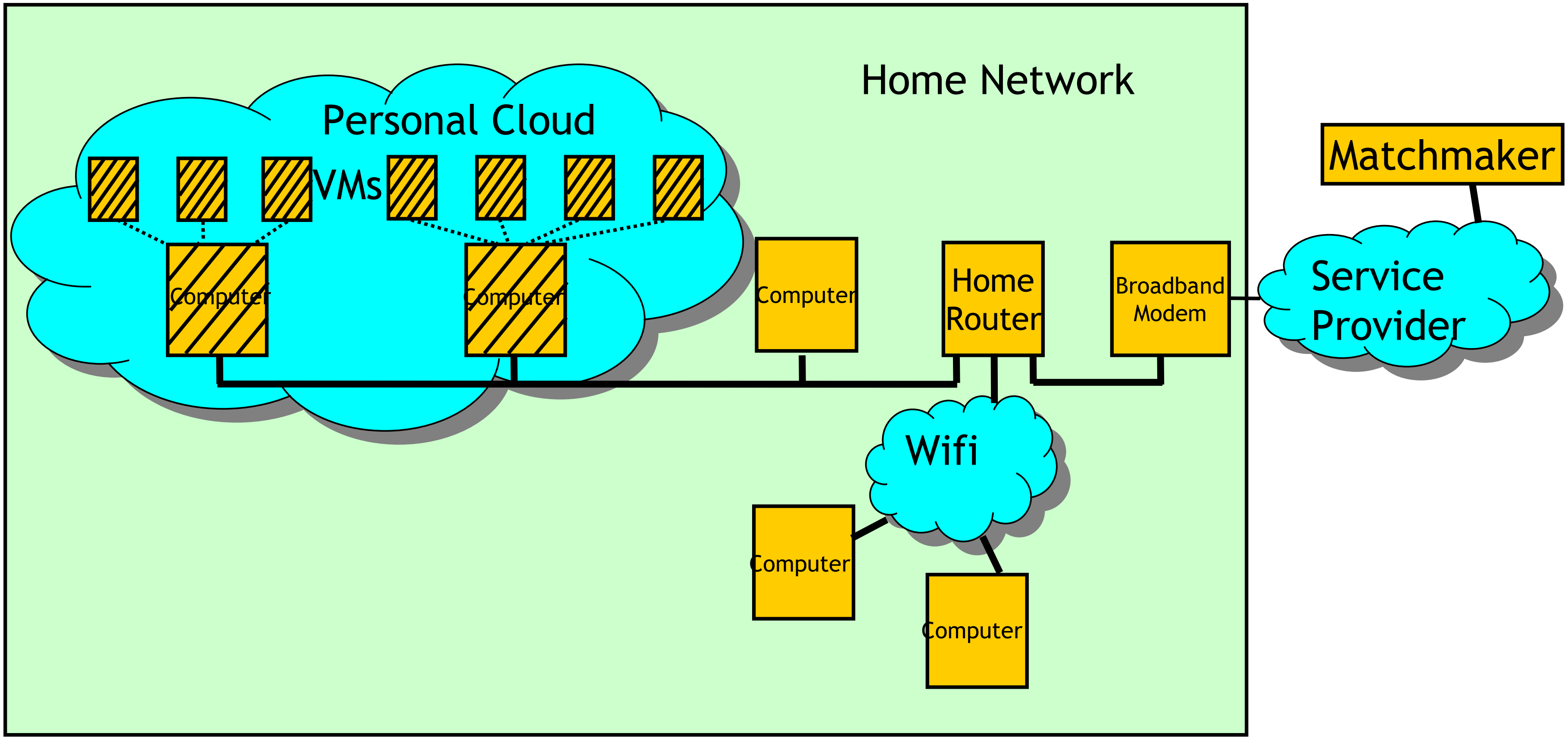
Max Flow Reduction

Provides integral results

Supports cost based matchmaking

Supports incremental matchmaking

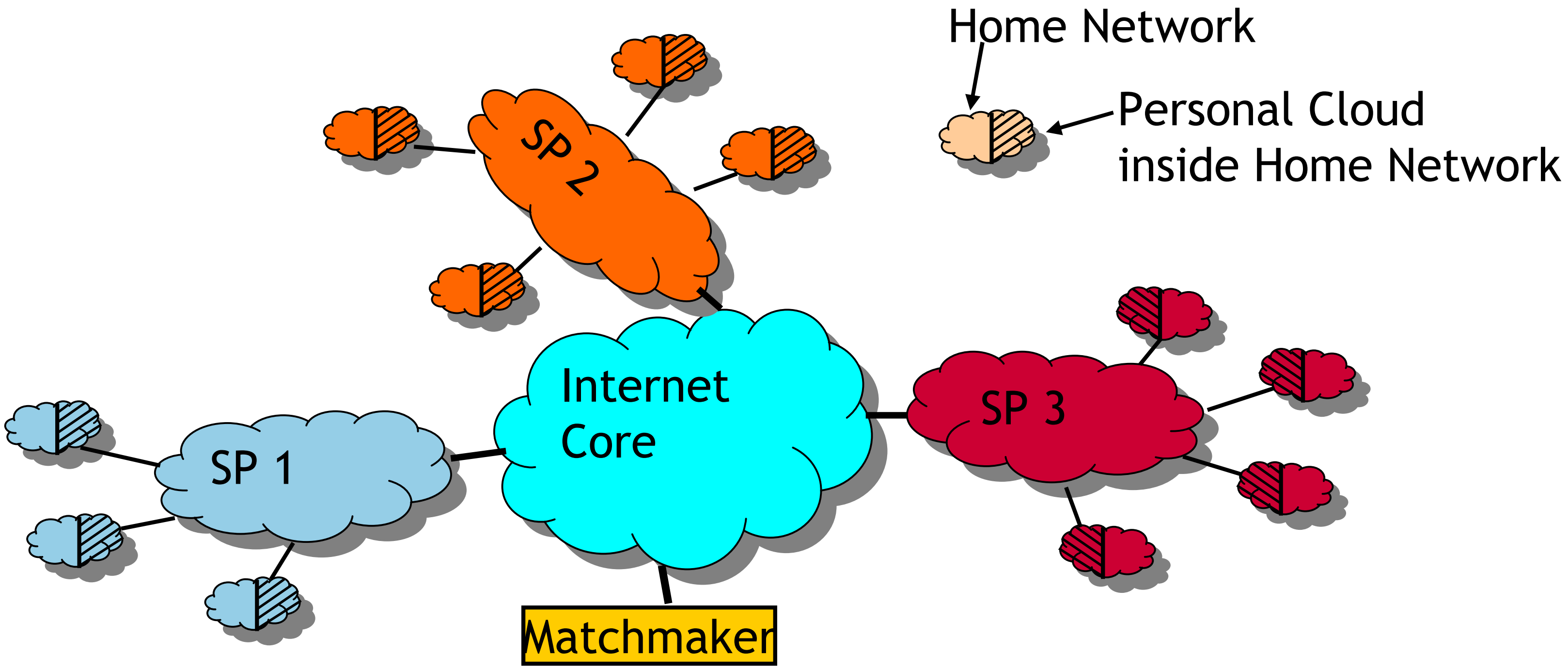
Phase 2 – The Personal Cloud – VM based approach to edge resource sharing



Run VMs on each unused computer and isolate cloud traffic from home traffic

Rely on Management Agents in each computer for isolation and bandwidth management

Personal Cloud – Usage and Life Cycle



1. Endpoints install Personal Cloud SW

2. Run matchmaking algorithm

3. Allocate VMs based on Matchmaking results

4. As nodes/computers are added/deleted, run matchmaking algorithm

Personal Cloud – System design Issues

Security

- Securing VMs from hosting entity
- Securing hosting entity from VMs

Resiliency

- Dealing with VM/physical machine/network outages

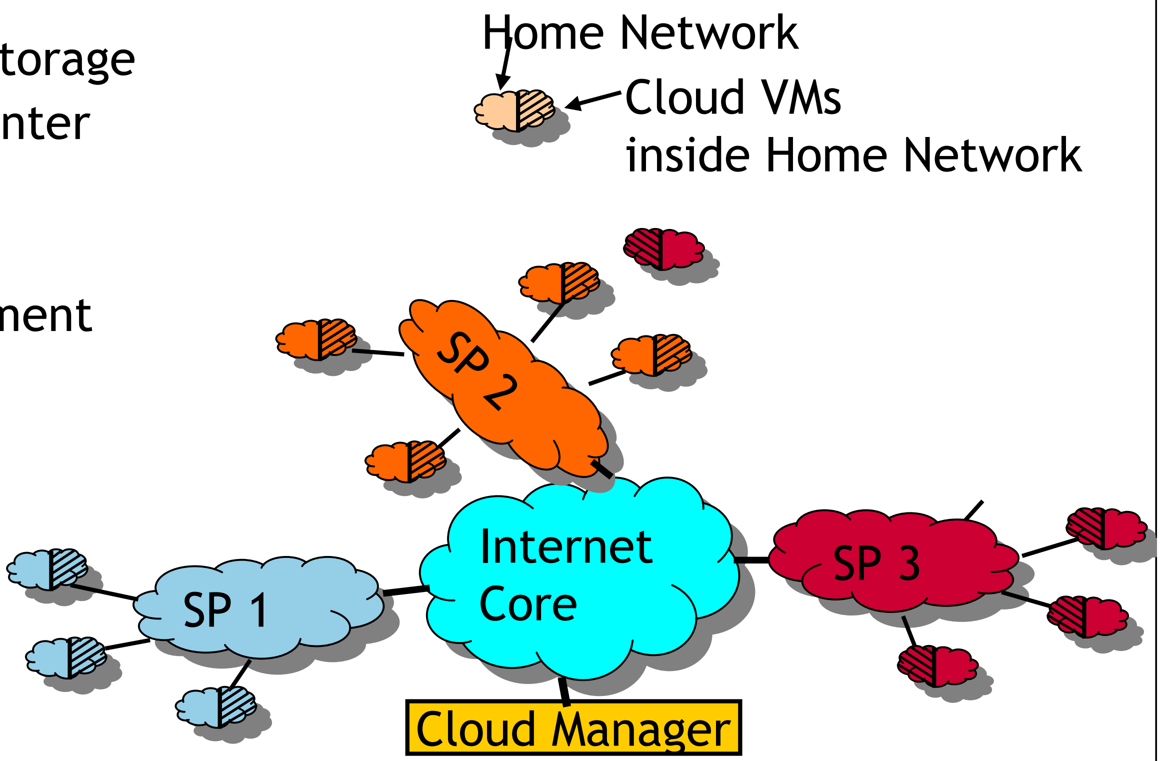
NAT traversal and BW Management

Phase 3 – Virtual Distributed Edge Data Center

Aggregate Edge Compute and Storage resources into a virtual data center

Goals:

- Standard Cloud IaaS Management
- Standard PaaS (Hadoop)
- Backup application



A. Hari, S. Mukherjee, H. Chang, T.V. Lakshman

Matchmaking Extensions

How to handle heterogeneous resources?

How to handle affinity?

Existing Edge Resource sharing frameworks

BOINC, SEATTLE

- Special application programs on both used and unused computers
- No VPC. Only one central, global entity

PlanetLab

- No ability to run VMs or arbitrary network traffic

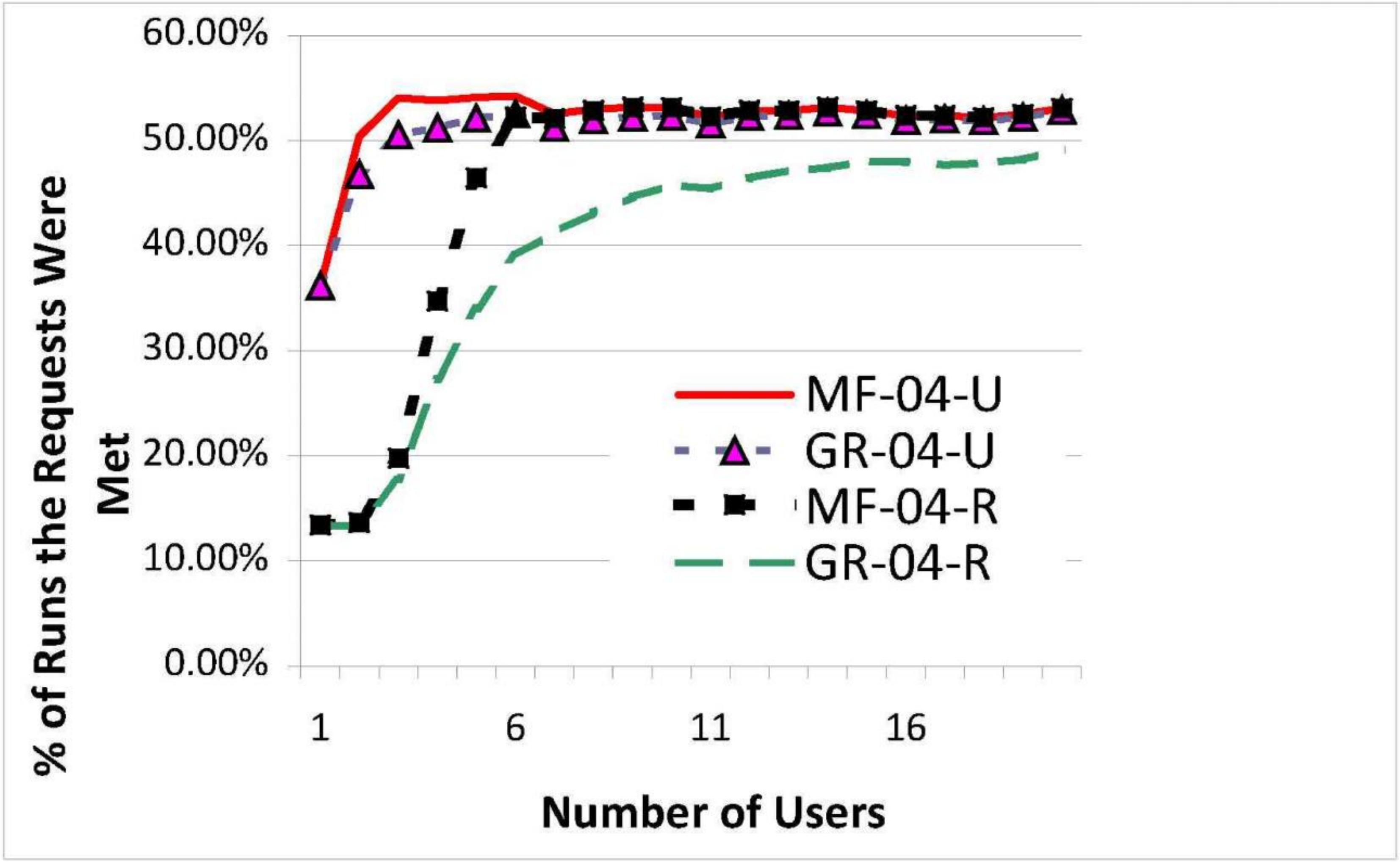
Nano Data Center

- Run VMs on edge appliances (set top boxes, routers etc), not edge PCs

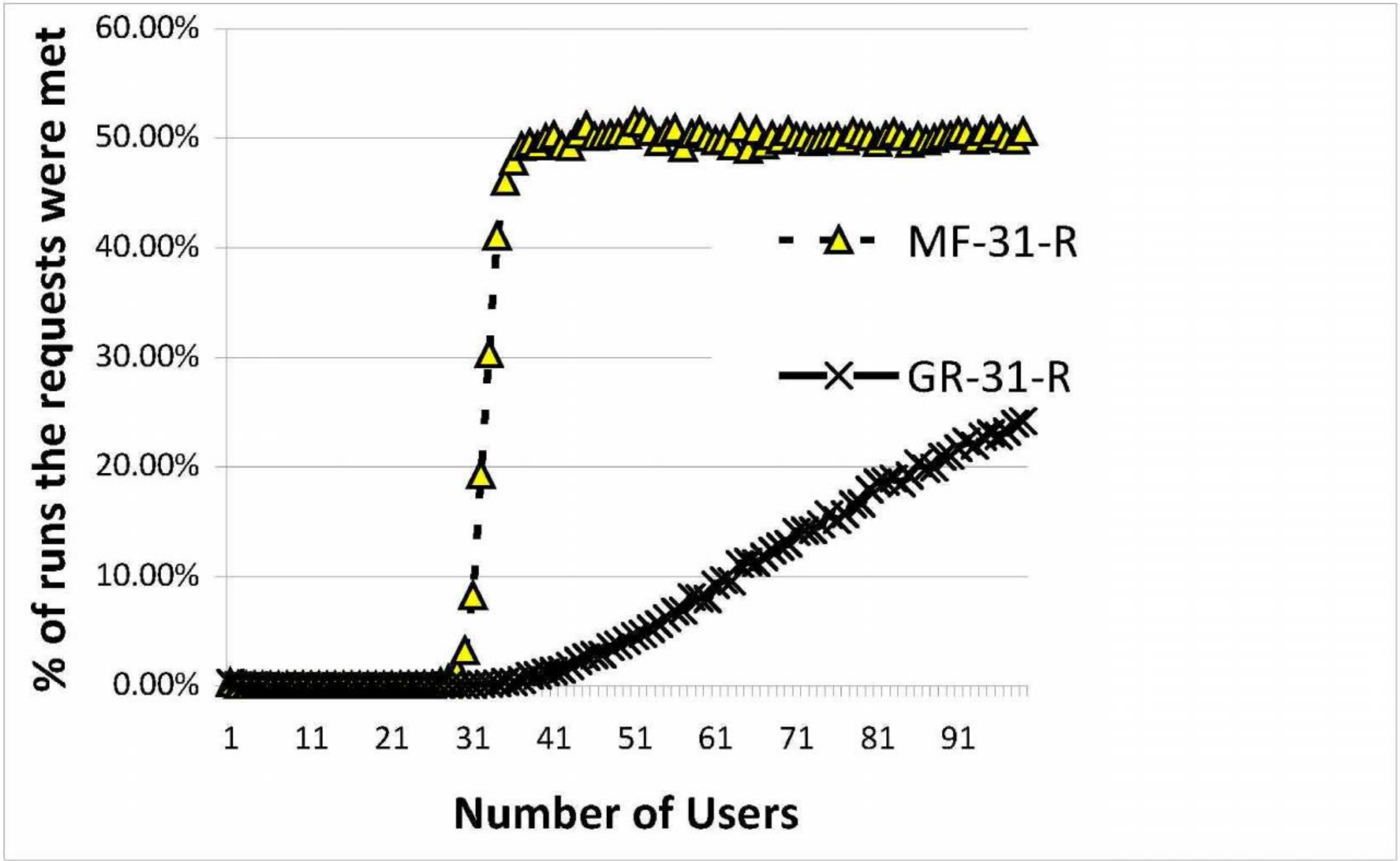
ONRC SDN Home Networks

- SDN capable home router

Matchmaking Performance – Upto 4 requests per user



Matchmaking Performance Results – Upto 31 requests per user


% of runs the requests were met
Number of Users
MF-31-R
GR-31-R