Optimal Bounds for Online Page Migration with Generalized Migration Costs

Johannes Schneider (IBM Zurich) & **Stefan Schmid (T-Labs Berlin)**

April, 2013

CloudNets = VNets Connecting Cloud Resources

Success of Node Virtualization

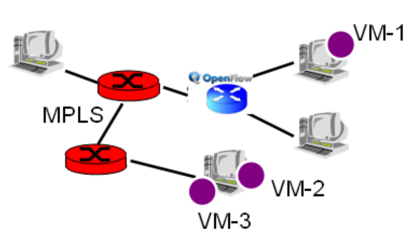
- a.k.a. end-host virtualization
- VMWare revamped server business
- OpenStack
- VM = flexible allocation, migration..
- «**Elastic computing**»

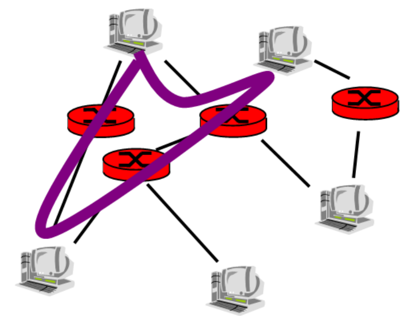
Trend of Link Virtualization

- MPLS, VPN networks, VLANs
- Software Defined Networks (SDN), OpenFlow, ...
- «The VMWare of the net»
- «**Elastic networking**»

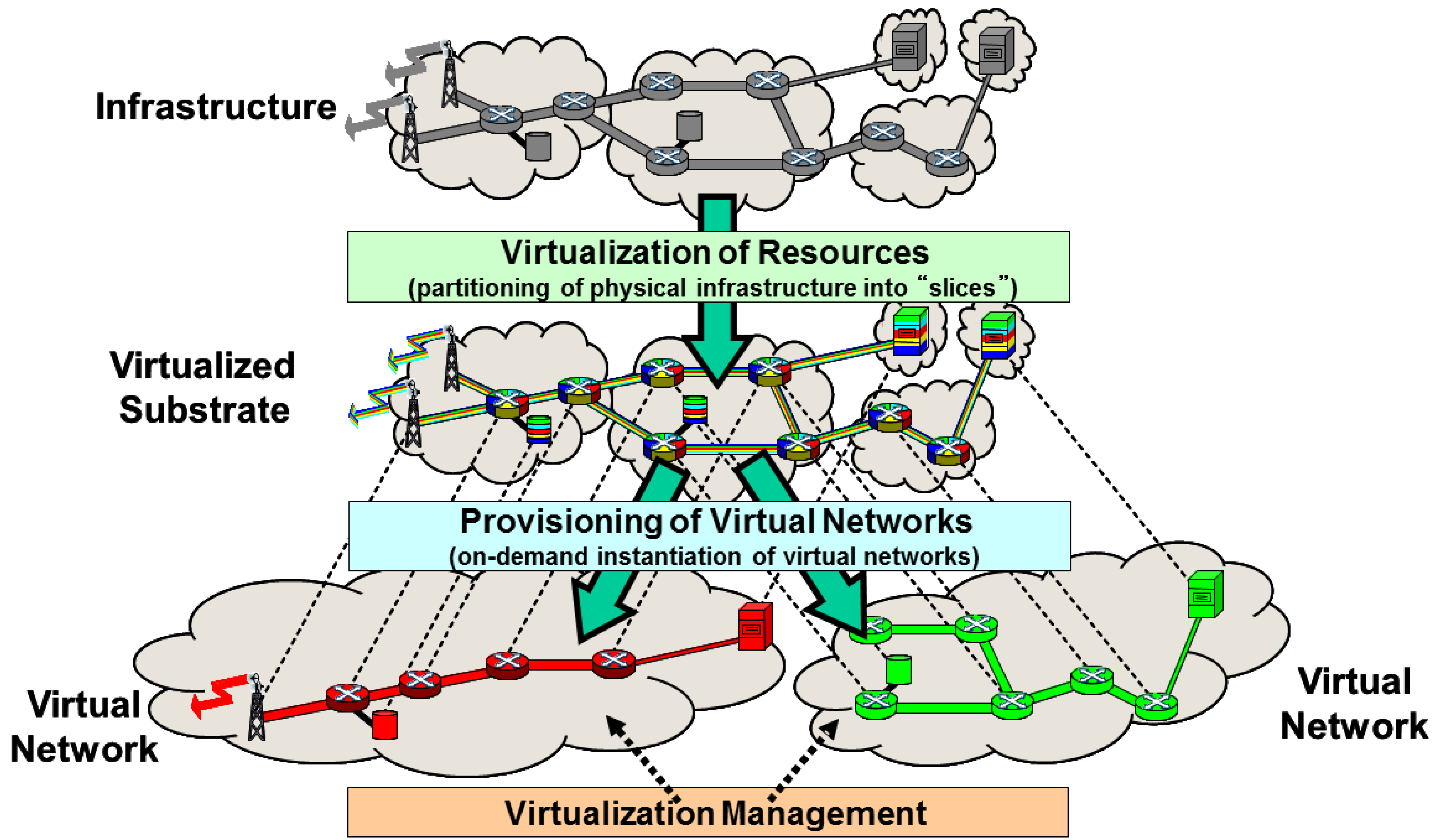
Unified, fully virtualized networks: **CloudNets**

„Combine networking with heterougeneous cloud resources (e.g., storage, CPU, ...)!“

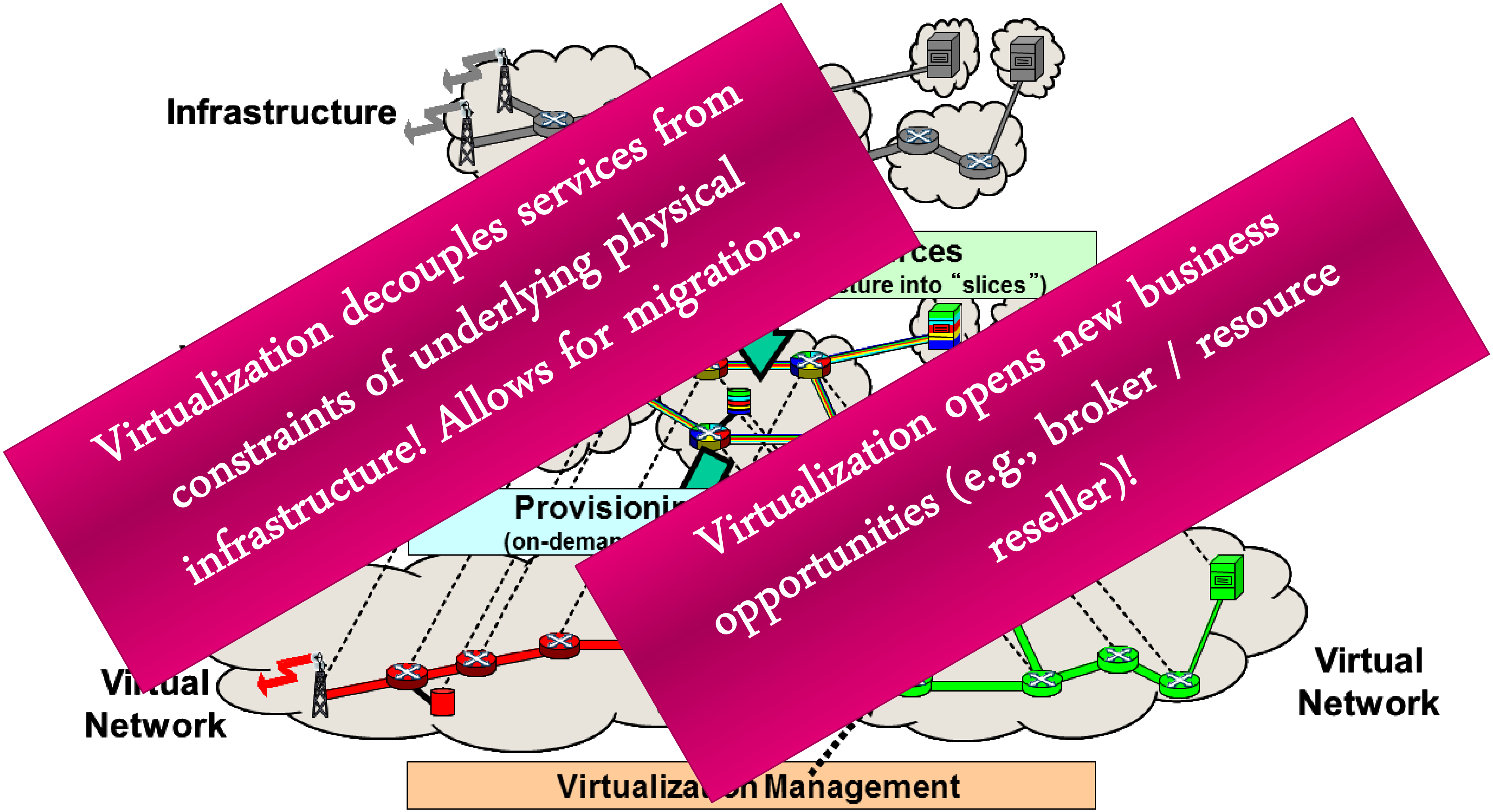




The Vision: Sharing Resources.



The Vision: Sharing Resources.



New Economic Models.

As in Internet today:

Netflix, Google, World of Warcraft…

As in Internet today: Telekom, AT&T, …

+ resource control interface (bootstrapping etc.)

Roles in CloudNet Arch.

Service Provider (SP)

(offers services over the top)

Virtual Network Operator (VNO)

(operates CloudNet, Layer 3+, triggers migration)

Virtual Network Provider (VNP)

(resource broker, compiles resources)

Physical Infrastructure Provider (PIP)

(resource and bit pipe provider)

New Economic Models.

Broker may give / get **discounts** for more resources!

Roles in CloudNet Arch.

Service Provider (SP)
(offers services over the top)

Virtual Network Operator (VNO)
(operates CloudNet, Layer 3+, triggers migration)

Virtual Network Provider (VNP)
(resource broker, compiles resources)

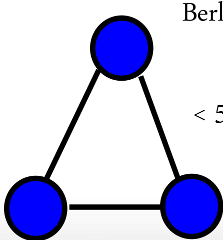
Physical Infrastructure Provider (PIP)
(resource and bit pipe provider)

Scenarios for Service Migration.

Spillover/Out-Sourcing

Elastic computing

Berlin



< 50ms

Berlin

< 50ms

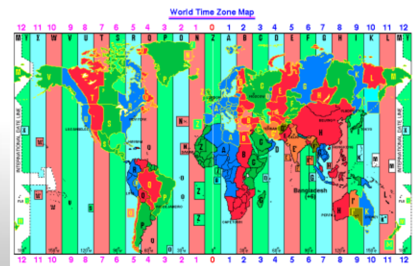
(corporate access network)

„50 TB storage, 10 Tflops computation!“

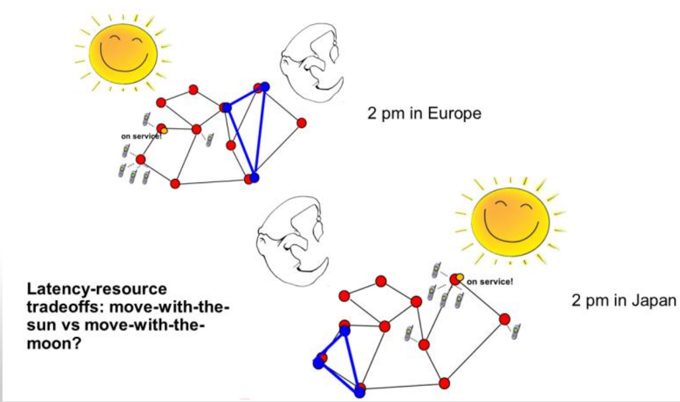
„any European cloud provider (e.g. due to legal issues?)“

Service Migration / Deployment

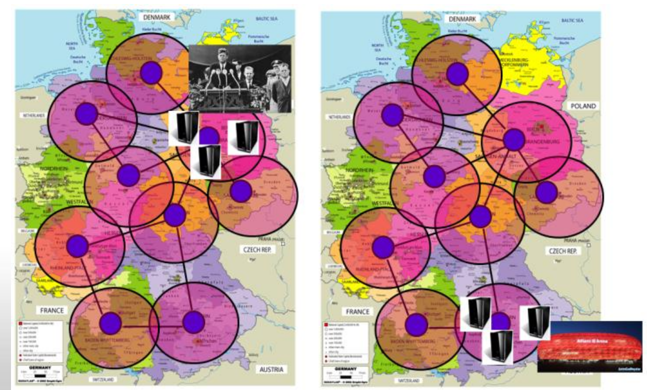
Goal: *Move with the sun, with the commuters, (QoS) allow for **maintenance**, avoid roaming costs…: e.g., **SAP/game/translator server, small CDN server…***



Move with Moon/Sun



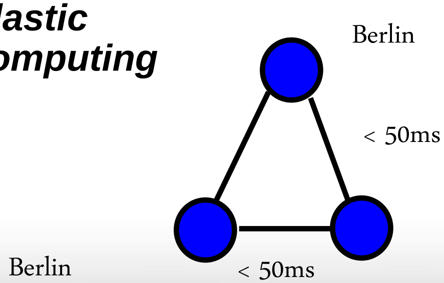
Dynamic Resources



Scenarios for Service Migration.

Spillover/Out-Sourcing

Elastic computing



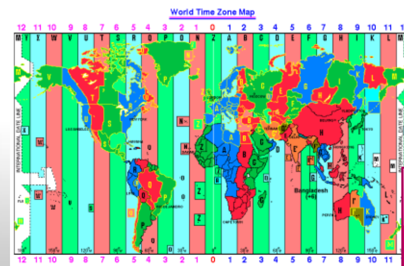
(corporate access network)

„50 TB storage, 10 Tflops computation!"

„any European cloud provider (e.g. due to legal issues?)"

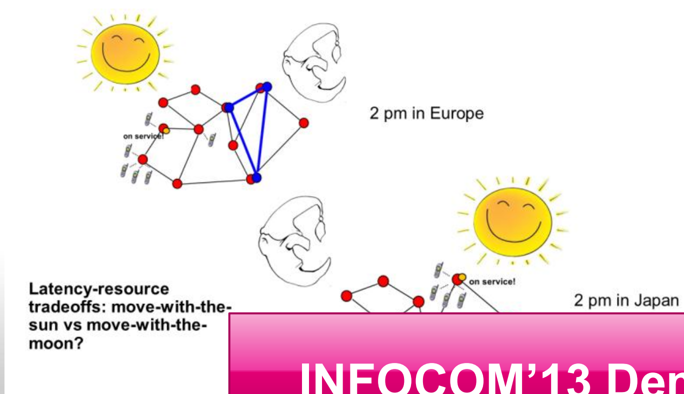
Service Migration / Deployment

<u>Goal</u>: *Move with the sun, with the commuters, (QoS) allow for* ***maintenance****, avoid roaming costs…: e.g.,* ***SAP/game/translator server, small CDN server…***



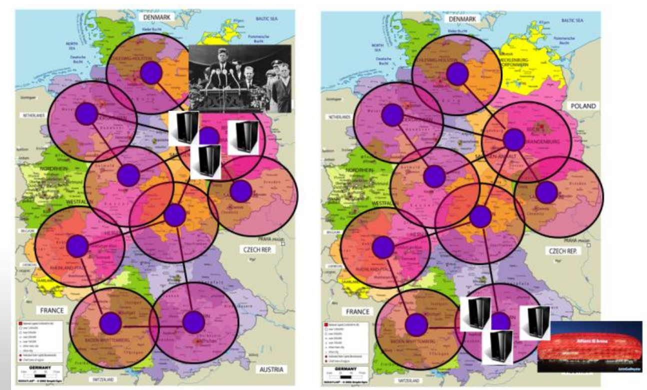
Focus of this talk

Move with Moon/Sun

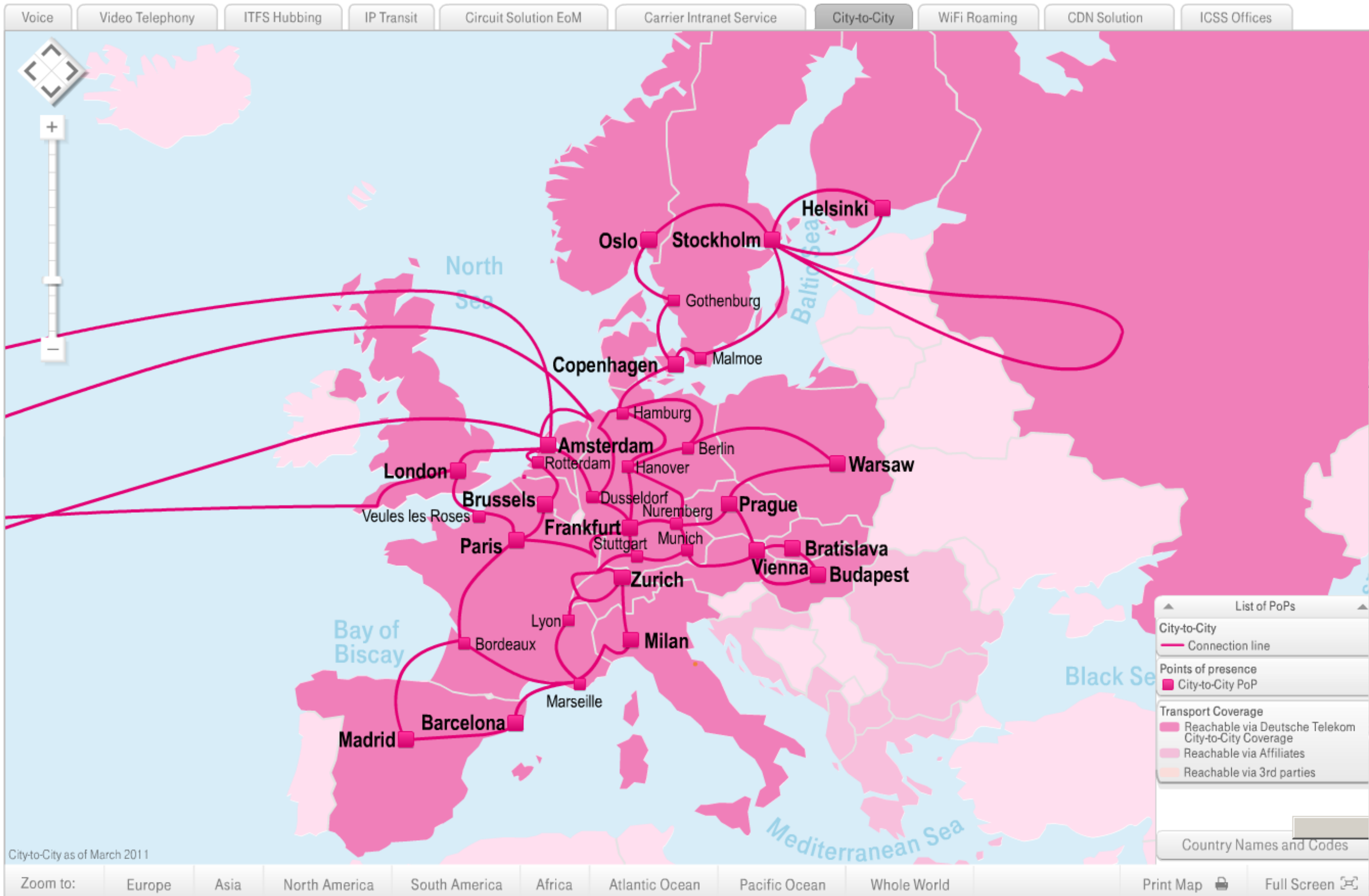


INFOCOM'13 Demo

Dynamic Resources

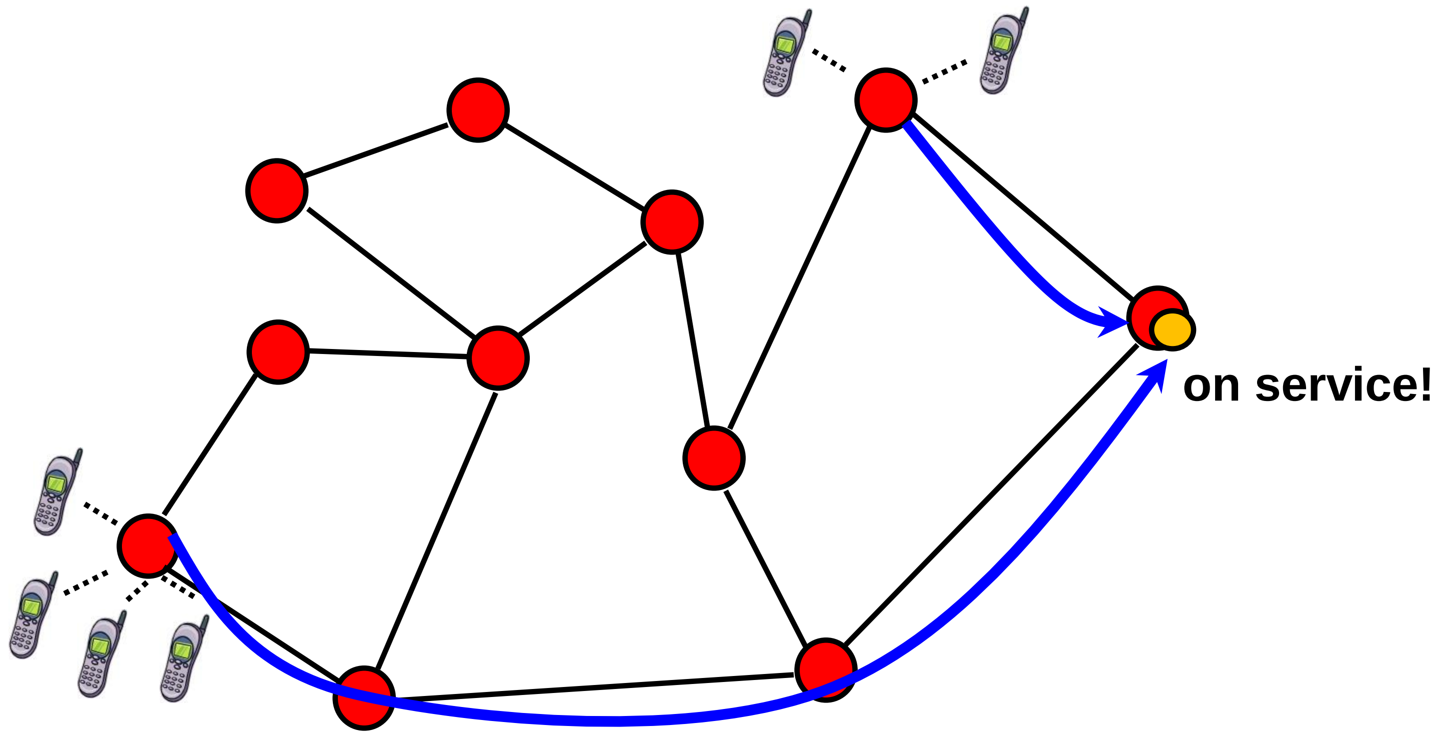


This Paper: Competitive Service Migration.



Model: E.g., using resources at Central Offices.

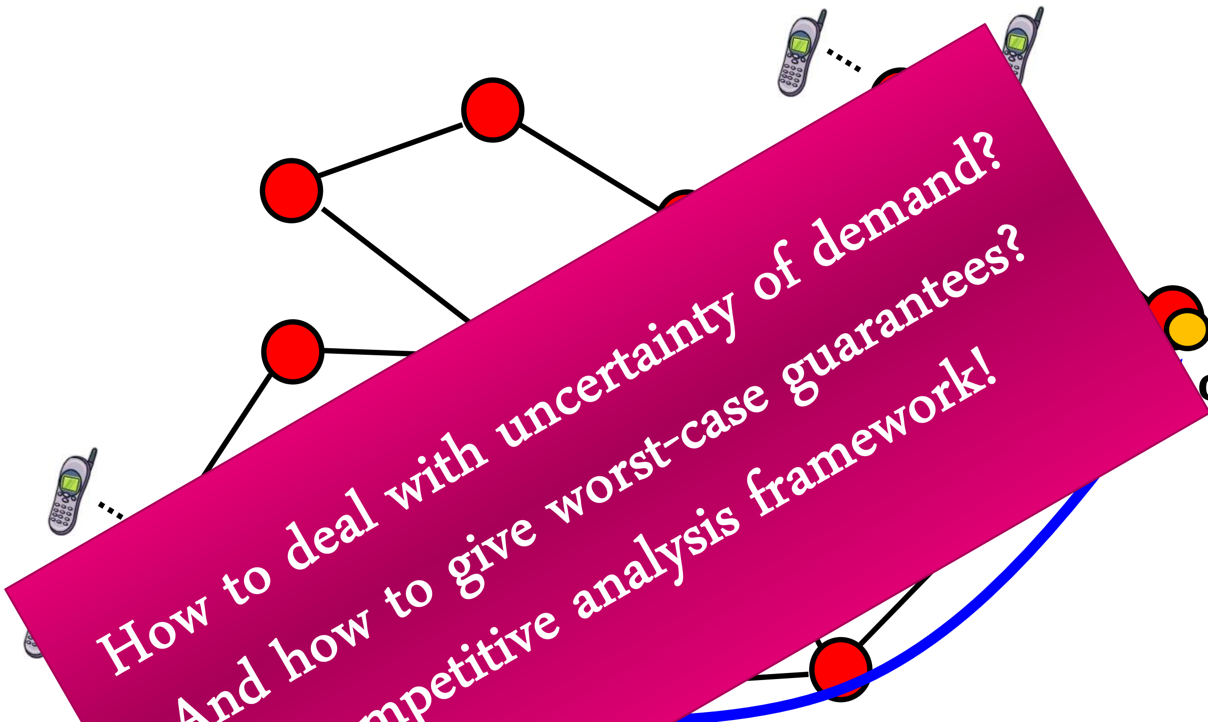
This Paper: Competitive Service Migration.



Model: Service should *be close to users* (total access latency) while minimizing *migration costs* (service size divided by min bw)!

(Special instance of metrical task system…)

This Paper: Competitive Service Migration.



How to deal with uncertainty of demand?
And how to give worst-case guarantees?
Competitive analysis framework!

…on service!

Model... should *be close to users* (total access latency) while minimiz... *migration costs* (service size divided by min bw)!

(Special instance of metrical task system…)

Competitive Approach.

How to deal with dynamic changes (e.g., mobility of users, arrival of CloudNets, etc.)?

Online Algorithm

Online algorithms make decisions at time t without any knowledge of inputs / requests at times t'>t.

Competitive Analysis

An *r-competitive online algorithm* ALG gives a worst-case performance guarantee: the performance is at most a factor r worse than an optimal offline algorithm OPT!

Competitive Ratio

Competitive ratio r,

r = Cost(ALG) / cost(OPT)

Is the price of not knowing the future!

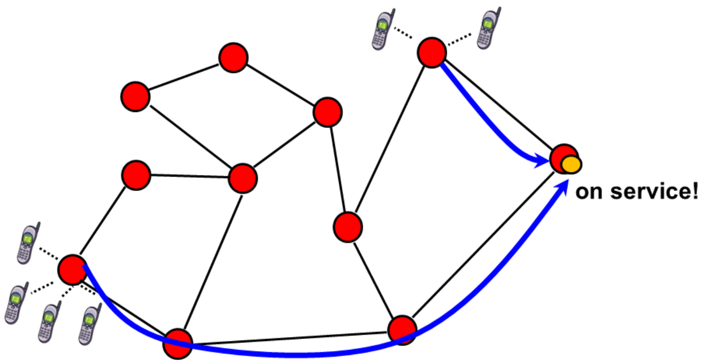
In virtual networks, many decisions need to be made online: online algorithms and network virtualization are a perfect match! ☺

No need for complex predictions but still good! ☺

Modeling Access and Migration Costs.

Access Costs

Latency along shortest path in graph.
(Graph distances, and in particular: metric!)



Migration Costs

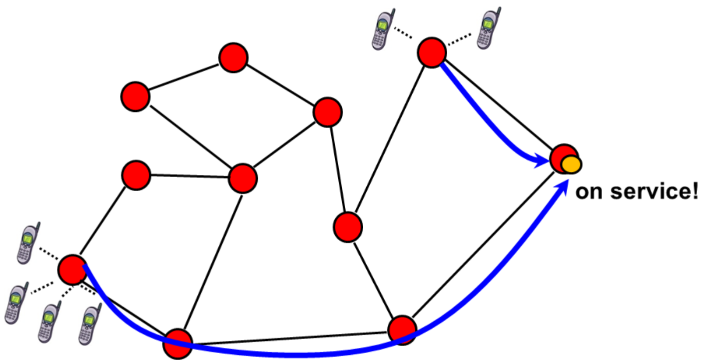
Generalized models:

- E.g., depends on bandwidth along path (duration of service interruption)
- E.g., depends on distance travelled (latency)
- Discount: e.g., VNP (number of migrations, distance travelled, ...)

Modeling Access and Migration Costs.

Access Costs

Latency along shortest path in graph.
(Graph distances, and in particular: metric!)



Migration Co...

Generalized mo...

- E.g. ...
(d...

...of

...elled, ...)

General cost function g(x|y): cost of migrating distance x given already travelled y

Or g(1|y): cost of migration given we already migrated y times

The Online Algorithm FOLLOWER.

Algorithm *Follower*

```
i := 0; k_0 := 0 ∀j: F_j = {} {The server starts at an arbitrary node f_0}
Upon a new request r do:
Serve request r with server at f_i
F_i := F_i ∪ r
f' := arbitrary u ∈ CG(F_i)
x' := d(f_i, f') {for co.di., and x' := 1 for co.nb.m.}
if C(f_i, F_i) ≥ g(x'|k_i) then
    f_{i+1} := f'; x_i := x'
    y(w) := d(f_i, w) + d(w, f_{i+1}) {for co.di., and for co.nb.m. y(w) := 2 for w ≠ f_{i+1} and y(w) := 1 otherwise }
    slack(w ∈ V) := g(y(w)|k_i) − C(f_i, F_i)
    w_i := Node w with minimum slack(w) such that slack(w) ≥ 0
    Move server to w_i and if w_i ≠ f_{i+1} onto f_{i+1}
    k_{i+1} := k_i + y(w_i)
    i := i + 1
end if
```

Concepts:

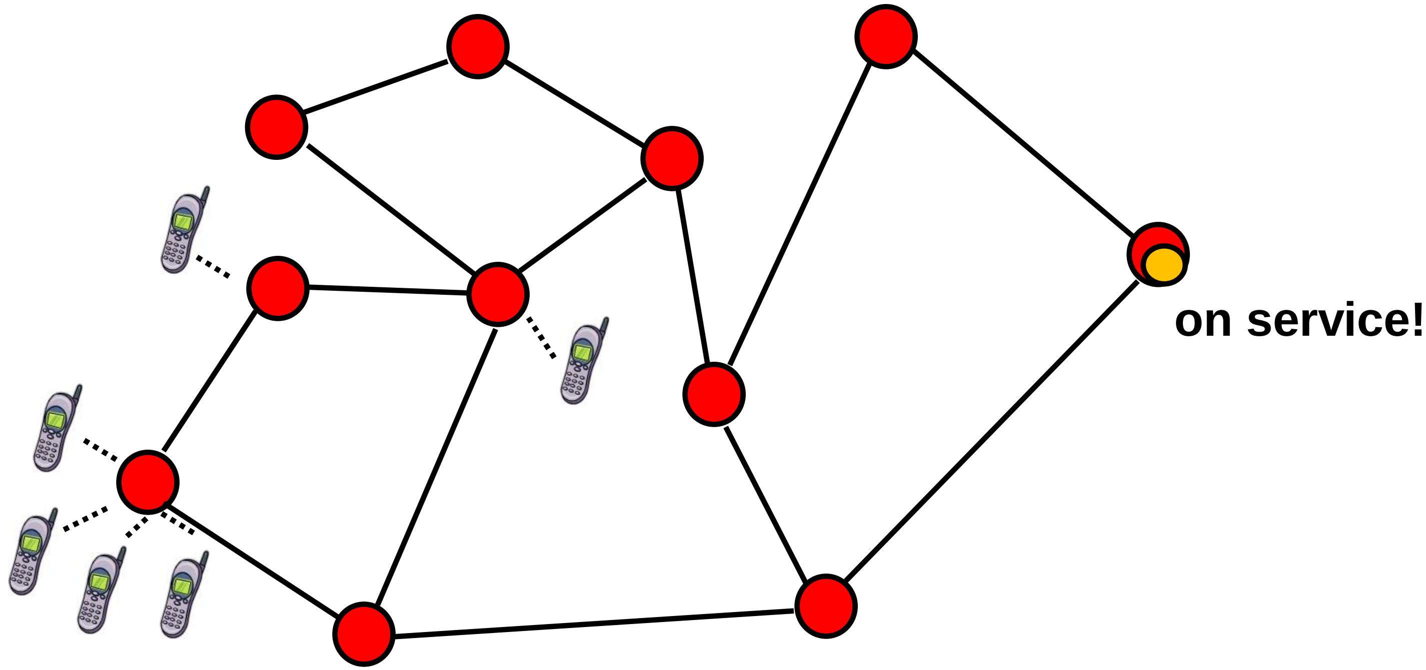
- **Learn from the past:** migrate to center of gravity of best location *in the past*
- **Amortize:** migrate only when access cost at current node is as high as migration cost!

Simplified Follower

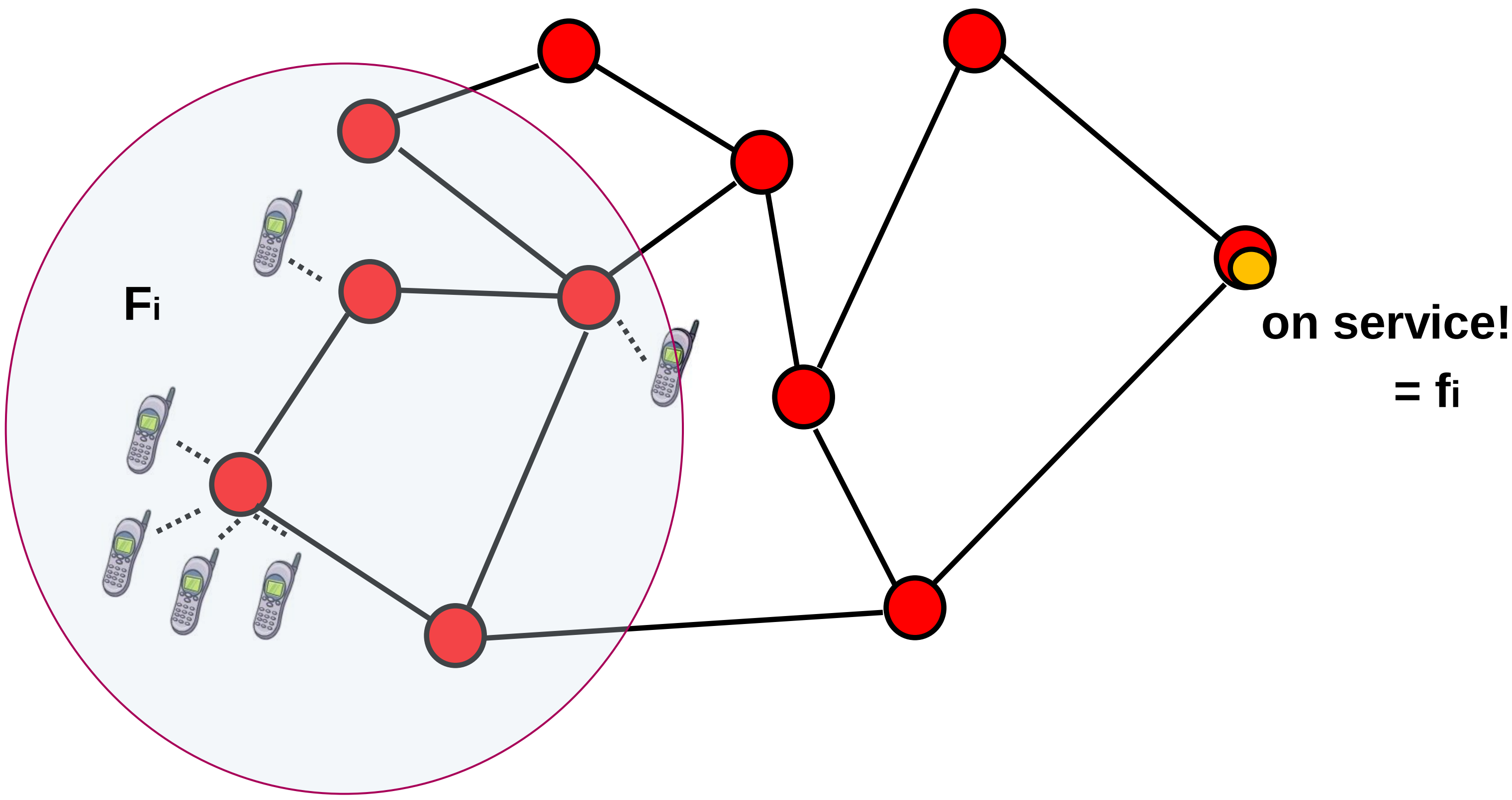
1. F_i are requests handled while service at f_i
2. to compute f_{i+1} (new pos), Follower only takes into account requests during f_i: F_i
3. migrate to center of gravity of F_i, as soon as migration costs there are amortized!

Also works for migrations with discount!
(Reseller/broker gives discount!)

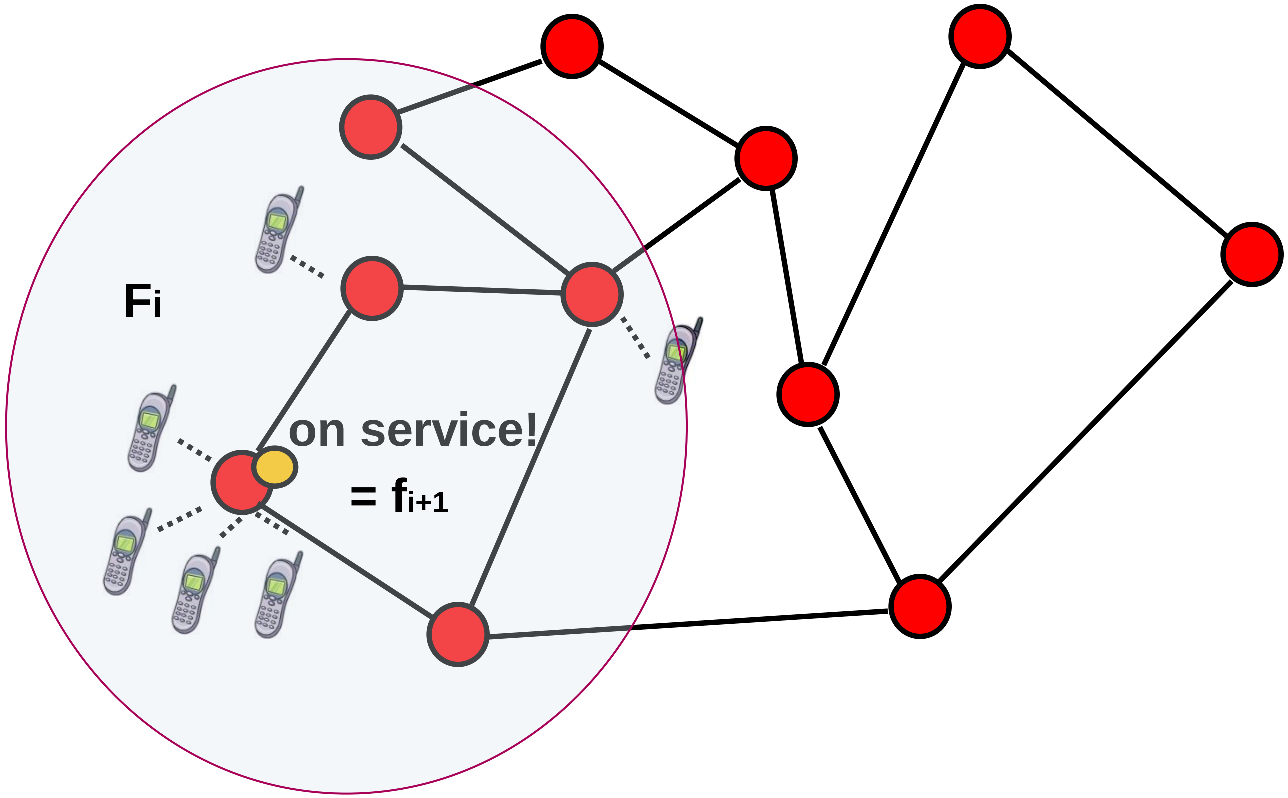
Intuition.



Intuition.



Intuition.



Competitive Ratio of FOLLOWER.

Competitive analysis? **FOLLOWER / OPT?**

Theorem

If no discounts are given,
Follower is log(n)/loglog(n) competitive!

Simple model with *migration costs = bandwidth*, and *homogeneous*

Page migration model with *migration costs = distance*, but discounts

Theorem

If migration costs depend on travelled distance (page migration), competitive ratio is O(1), even with discounts.

Competitive Ratio of FOLLOWER.

Competitive analysis? **FOLLOWER / OPT?**

Theorem

If no discounts are given,
Follower is log(n)/loglog(n) competitive!

Simple model with *migration costs = bandwidth*, and *homogeneous*

This is asymptotically optimal! (Open question from Online Service Migration model at VISA'10, IPTComm'11, J. ToN'13)

…orem

…tion costs depend on travelled …e (page migration), competitive …O(1), even with discounts.

Competitive Ratio of FOLLOWER.

Competitive analysis? **FOLLOWER / OPT?**

Theorem

If no discounts are given,
Follower is log(n)/loglog(n) com

log/loglog(n) lower bound follows from reduction to online function tracking

This is asymptotically optimal! (Open question from Online Service Migration model at VISA'10, IPTComm'11, J. ToN'13)

rem

tion costs depend on travelled
e (page migration), competitive
O(1), even with discounts.

Related Work.

- **Metrical Task Systems:**
  - Classic online problem: task system with a set of states and costs to change states. Task system gets sequence of requests and each request assigns processing costs to states. Goal: minimize processing («latency») and minimize configuration change («migration»).
  - MTS cost function is more general (we have graph access costs) and less general (we allow for migration discounts: infinite state space)
  - E.g., uniform space metrical task system: migration costs constant, but access costs more general than graph distances! Lower bound of log(n) vs log(n)/loglog(n) upper bound in our case.

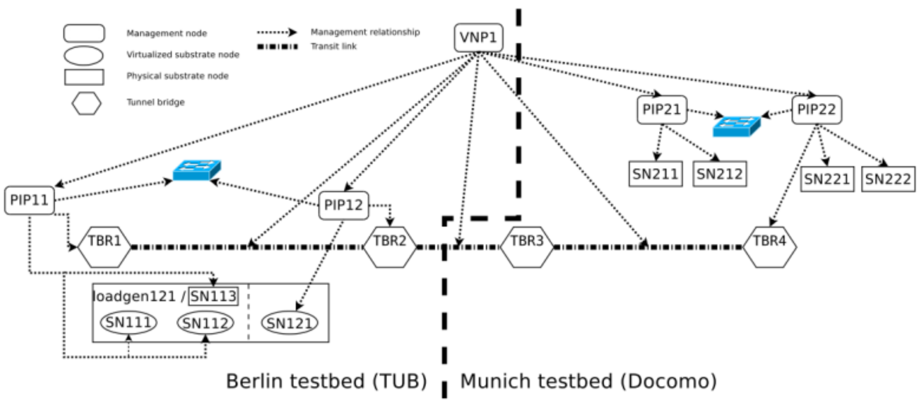
- **Online Page Migration:**
  - Classical online problem from the 80ies; we generalize cost function to distance discounts, while keeping O(1)-competitive

Our work lies between!

CloudNet Prototype at T-Labs: Decoupling Services.

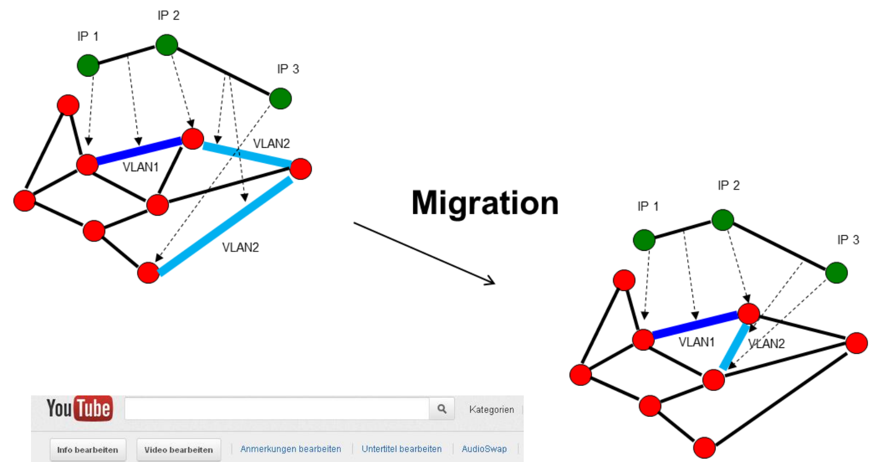
Testbed

Two connected sites: NTT DoCoMo and T-Labs



Migration

E.g., move VM by reconfiguring VLANs



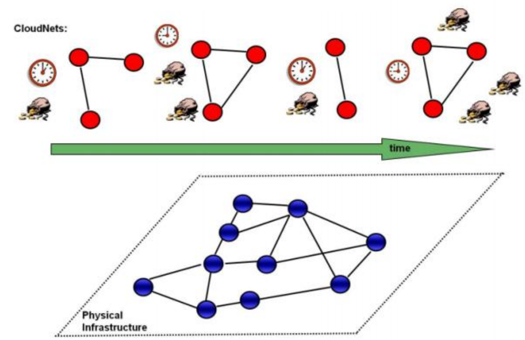
Conclusion.

- CloudNets:
  - Elastic computing and networking
  - Virtualization decouples services from infrastructure: flexibility and migration

- Competitive analysis: prove performance of online algorithms

- Good when:
  - No reliable prediction models exist, no data available, worst case guarantees matter

- Examples: online embedding (s. ICDCN'12 paper):



Competitive and Deterministic Embeddings of Virtual Networks
Guy Even, Moti Medina, Gregor Schaffrath, and Stefan Schmid.
Journal Theoretical Computer Science (**TCS**), Elsevier, to appear.
Documents: paper pdf link

- Fully incorporated in prototype

Collaborators and Publications.

- **People**
  - **T-Labs / TU Berlin:** Anja Feldmann, Carlo Fürst, Johannes Grassler, Arne Ludwig, Matthias Rost, Gregor Schaffrath, Stefan Schmid
  - **Uni Wroclaw:** Marcin Bienkowski
  - **Uni Tel Aviv:** Guy Even, Moti Medina
  - **NTT DoCoMo Eurolabs:** Group around Wolfgang Kellerer
  - **LAAS:** Gilles Tredan
  - **ABB:** Yvonne Anne Pignolet
  - **IBM Research:** Johannes Schneider
  - **Arizona State Uni:** Xinhui Hu, Andrea Richa

- **Publications**
  - **Prototype:** J. Information Technology 2013, ICCCN 2012, ERCIM News 2012, SIGCOMM VISA 2009
  - **Migration:** ToN 2013, INFOCOM 2013, ICDCN 2013 + Elsevier TCS Journal, Hot-ICE 2011, IPTComm 2011, SIGCOMM VISA 2010
  - **Embedding:** INFOCOM 2013 (Mini-Conference), CLOUDNETS 2012, 2 x ACM UCC 2012, DISC 2012, ICDCN 2012 (*Best Paper Distributed Computing Track*)

Contact.

Dr. Stefan Schmid

Telekom Innovation Laboratories
Ernst-Reuter-Platz 7, D-10587 Berlin
E-mail: stefan@net.t-labs.tu-berlin.de
Project website:
http://www.net.t-labs.tu-berlin.de/~stefan/virtu.shtml