DONAR
Decentralized Server Selection for Cloud Services

Patrick Wendell, Princeton University

Joint work with Joe Wenjie Jiang,
Michael J. Freedman, and Jennifer Rexford

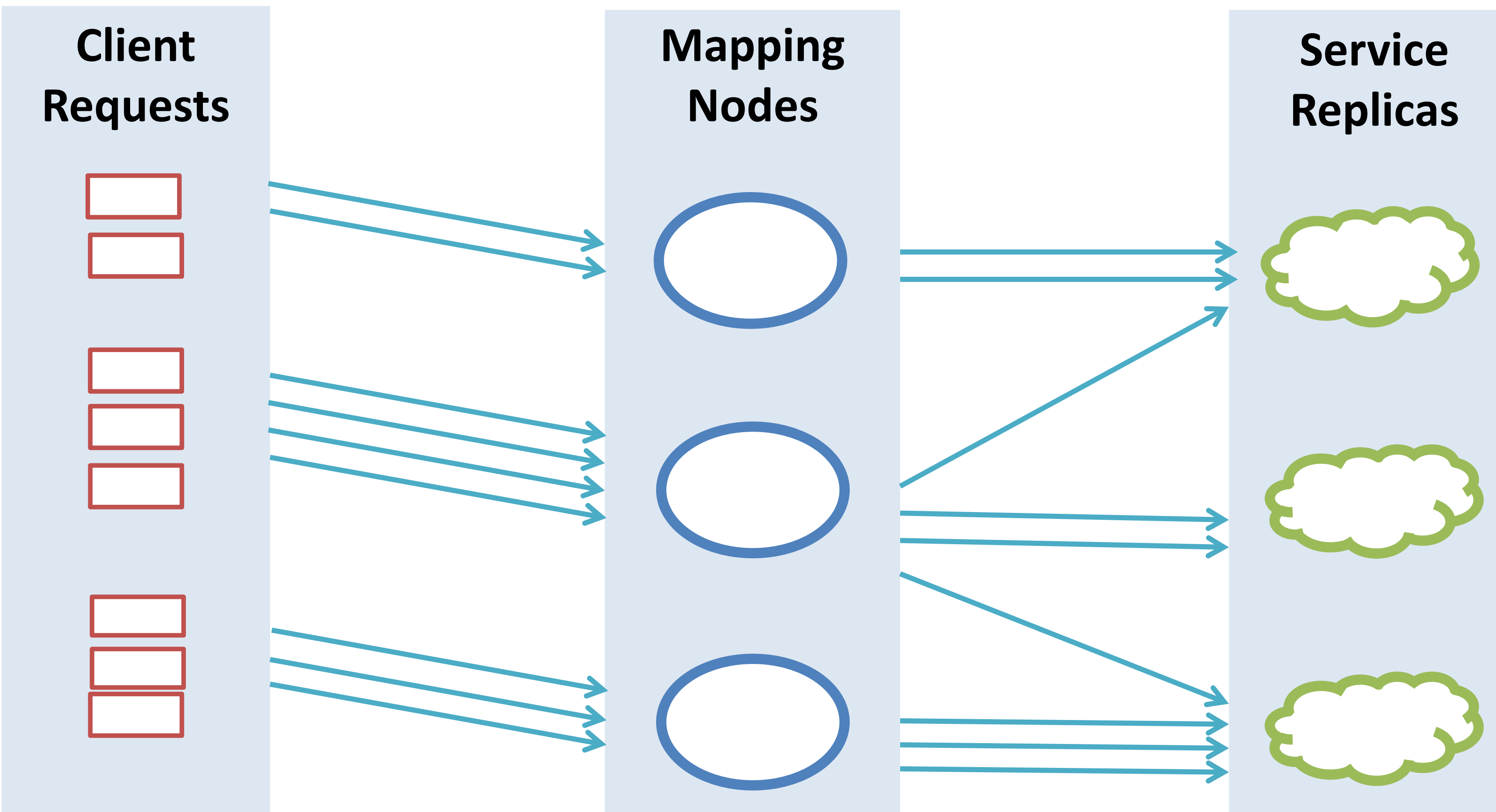
Outline

- Server selection background
- Constraint-based policy interface
- Scalable optimization algorithm
- Production deployment

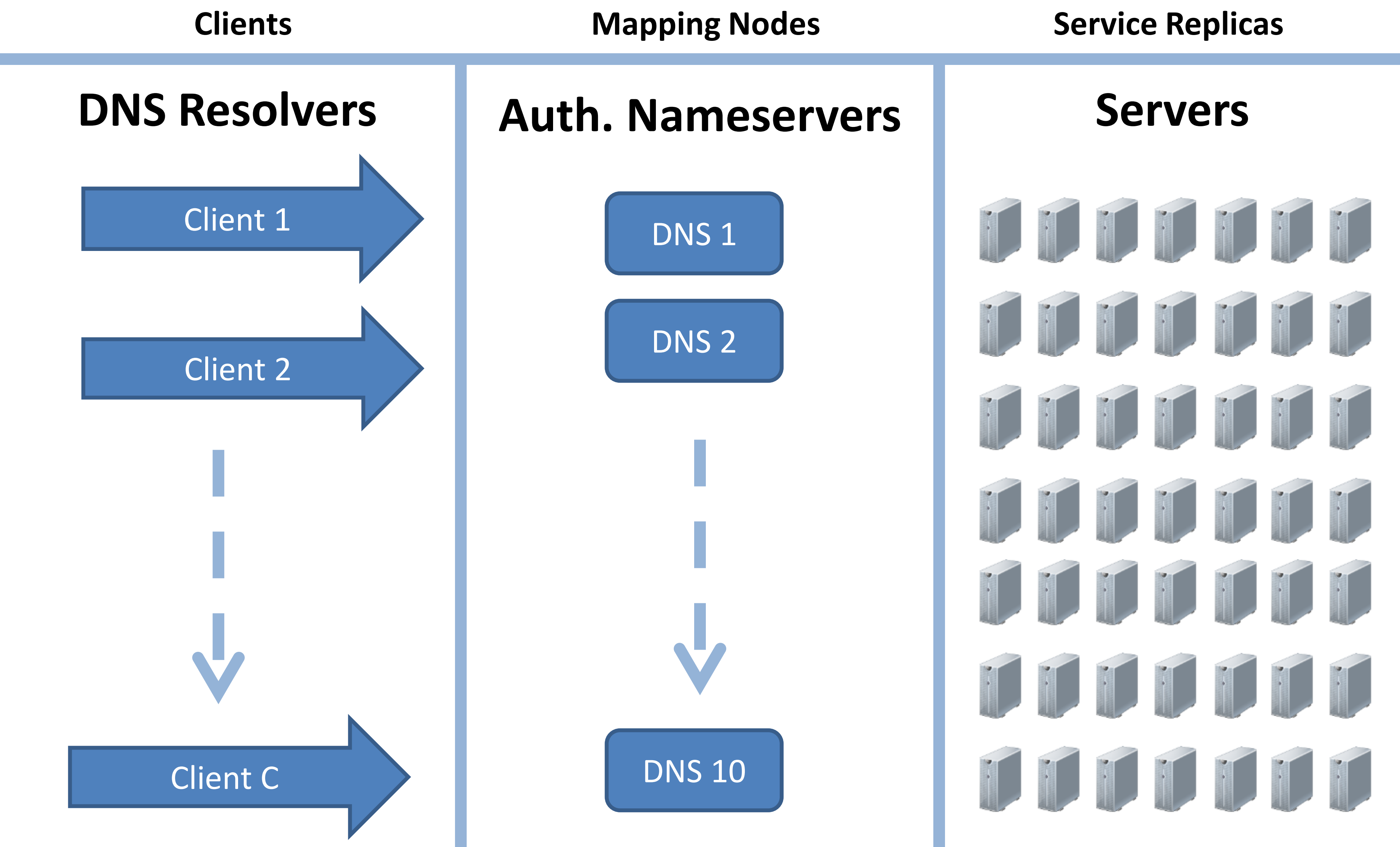
User Facing Services are Geo-Replicated

LIMELIGHT NETWORKS™

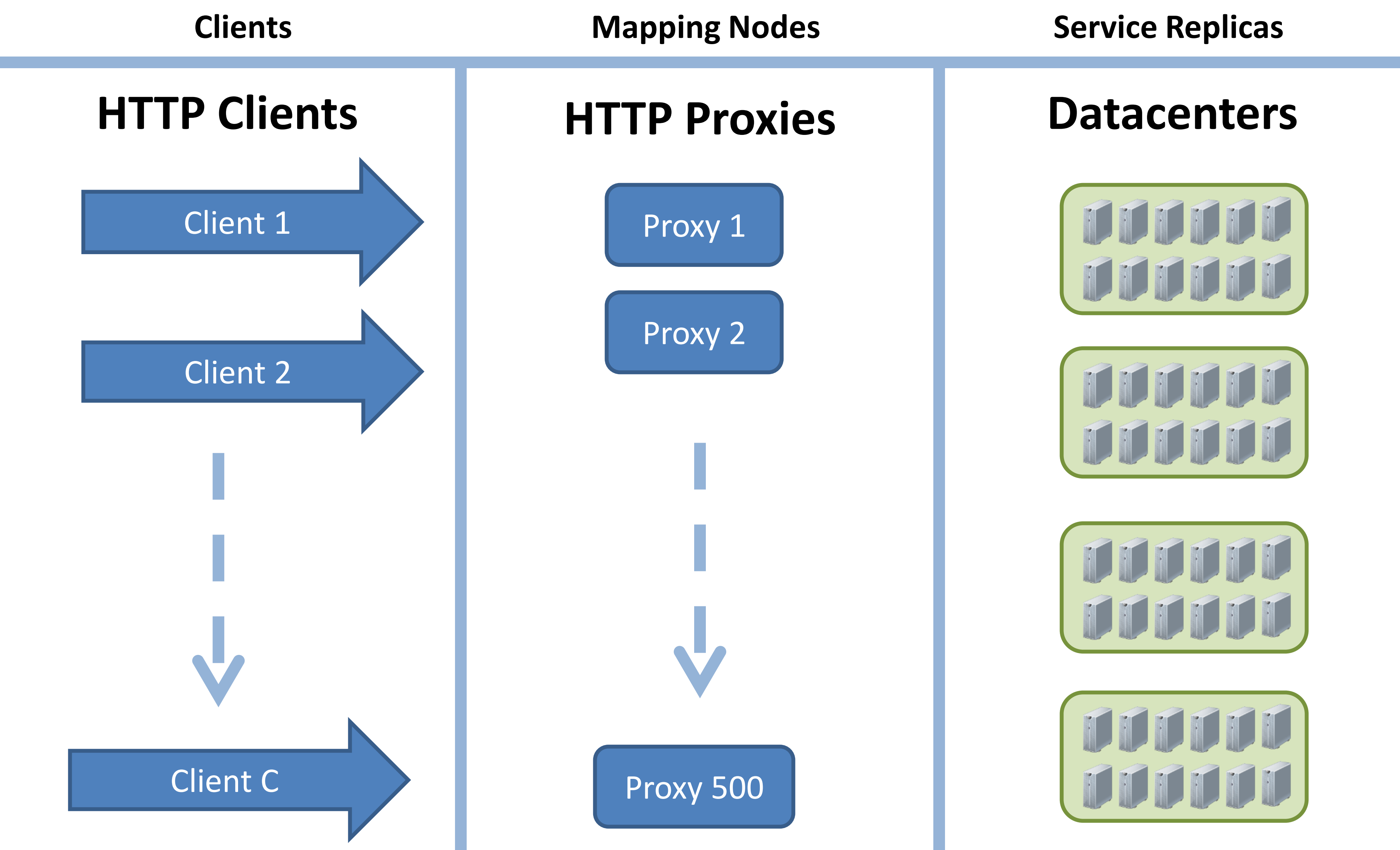
Reasoning About Server Selection



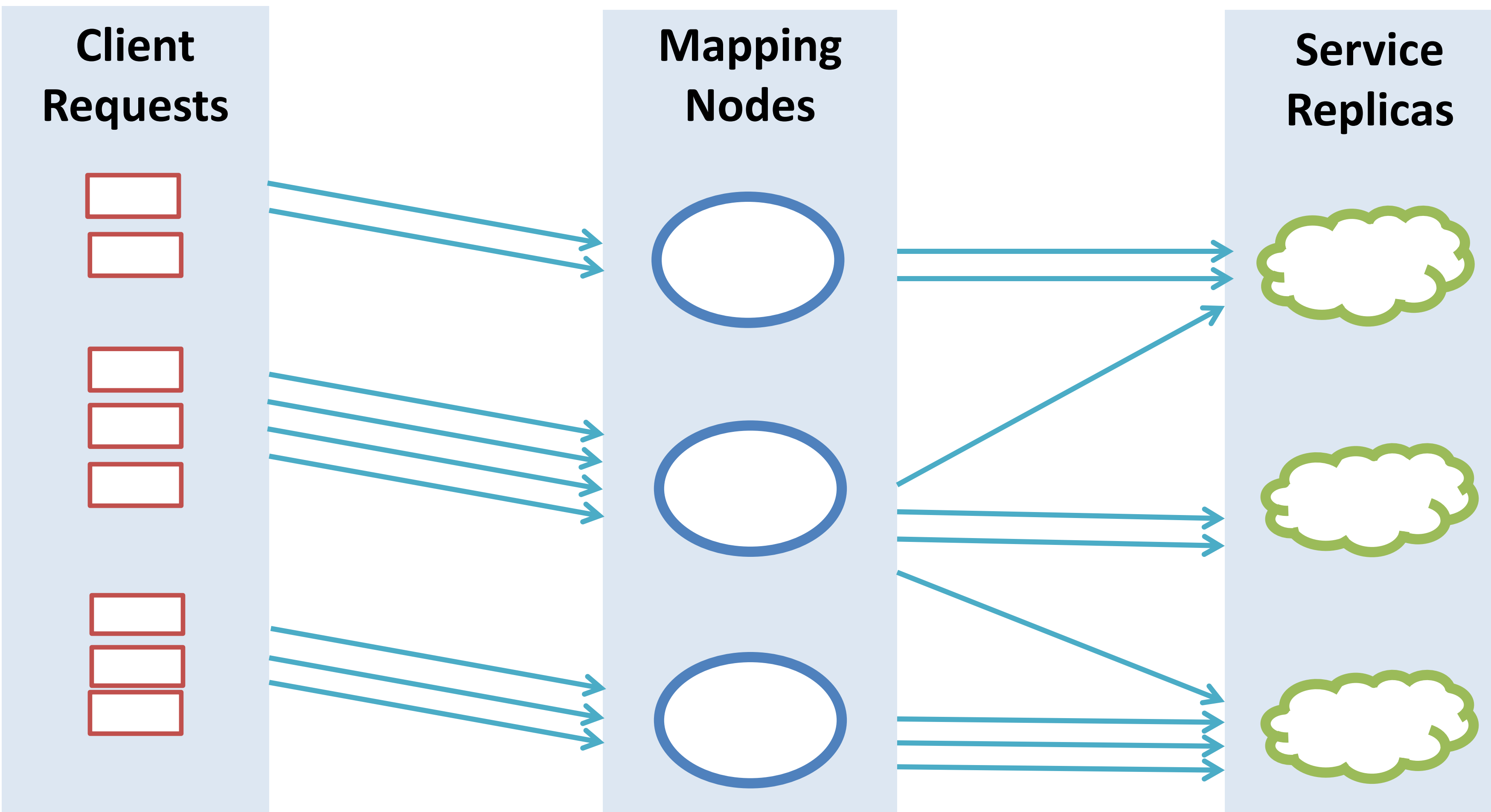
Example: Distributed DNS



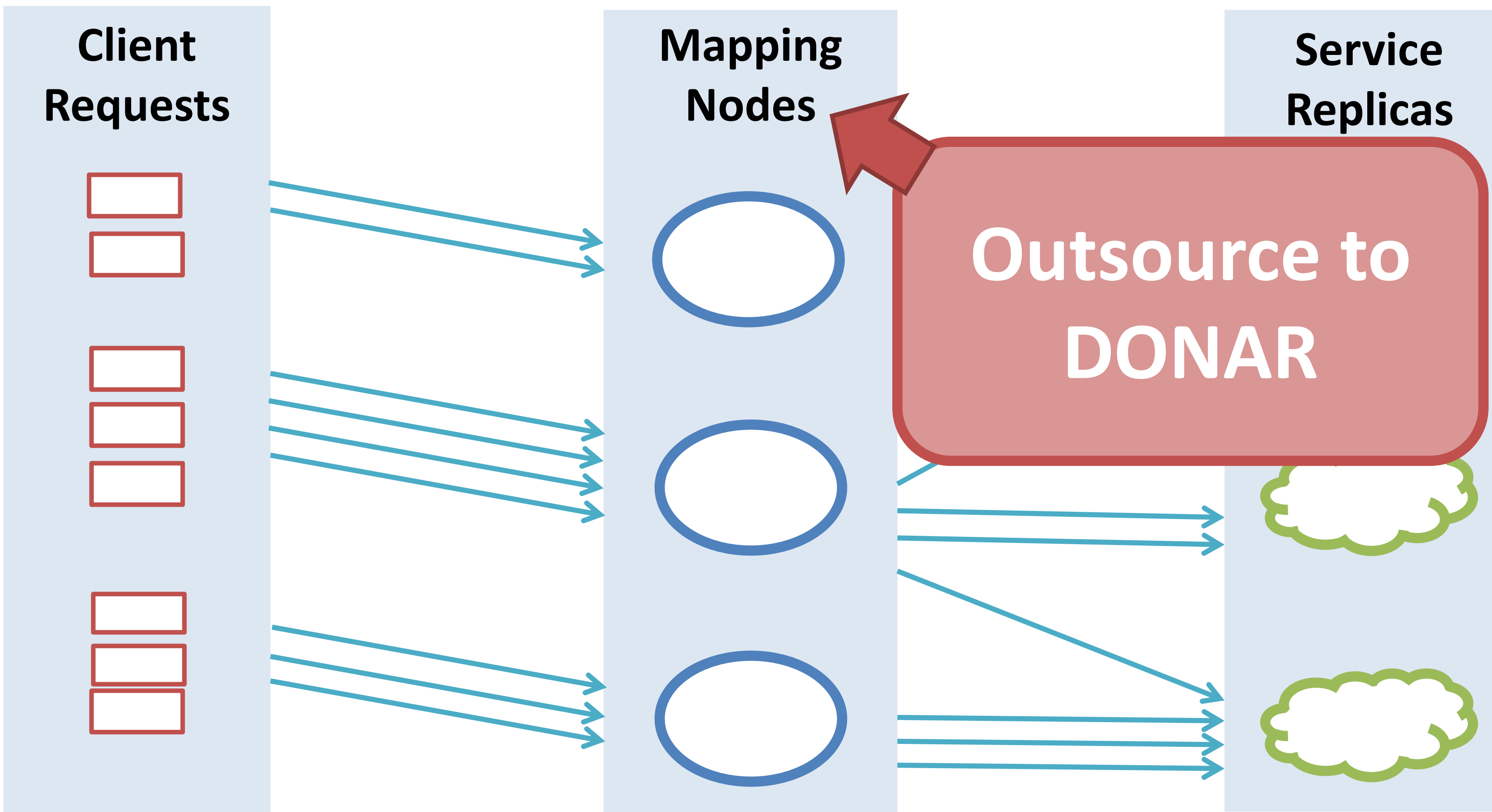
Example: HTTP Redir/Proxying



Reasoning About Server Selection



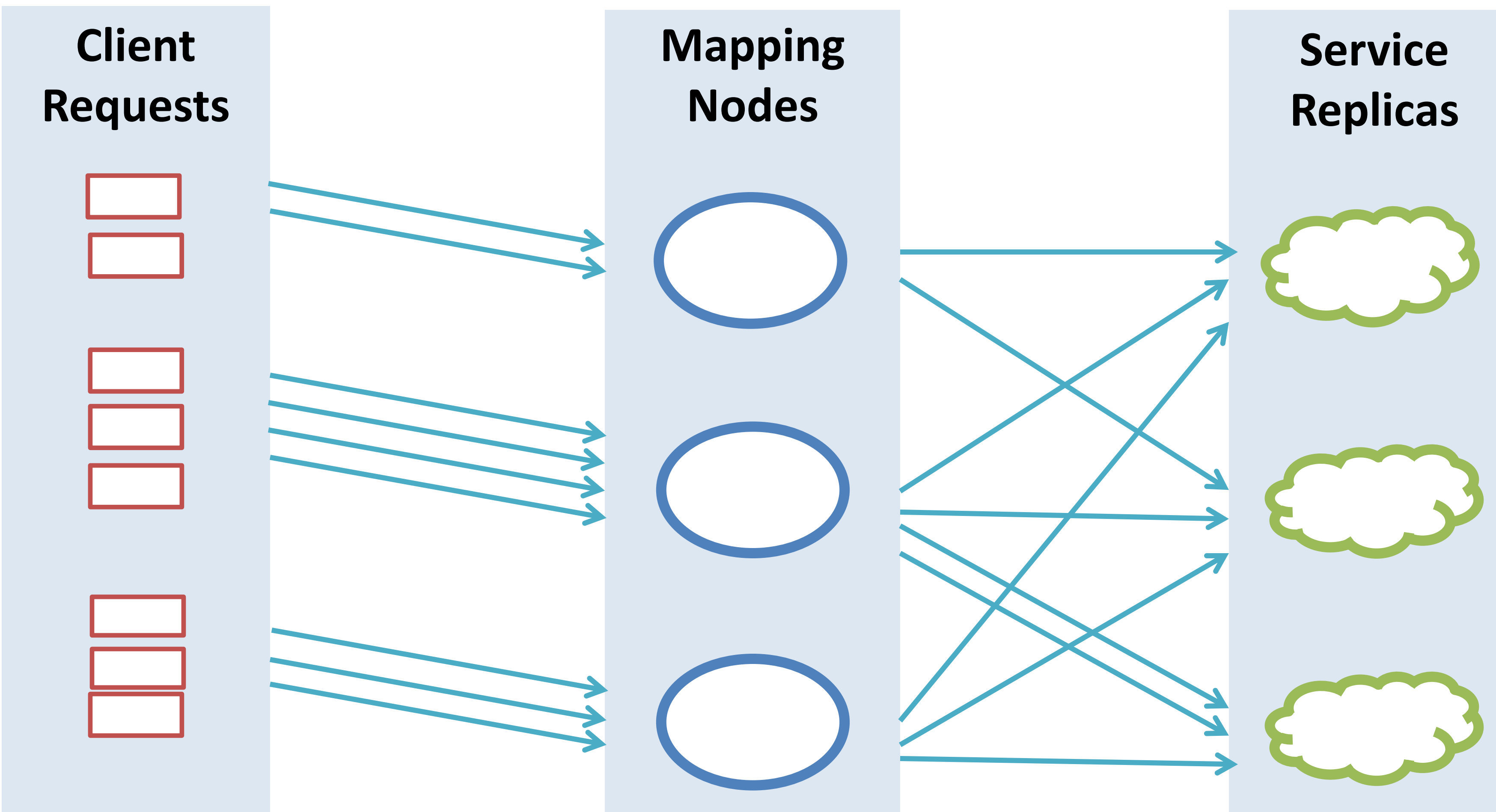
Reasoning About Server Selection



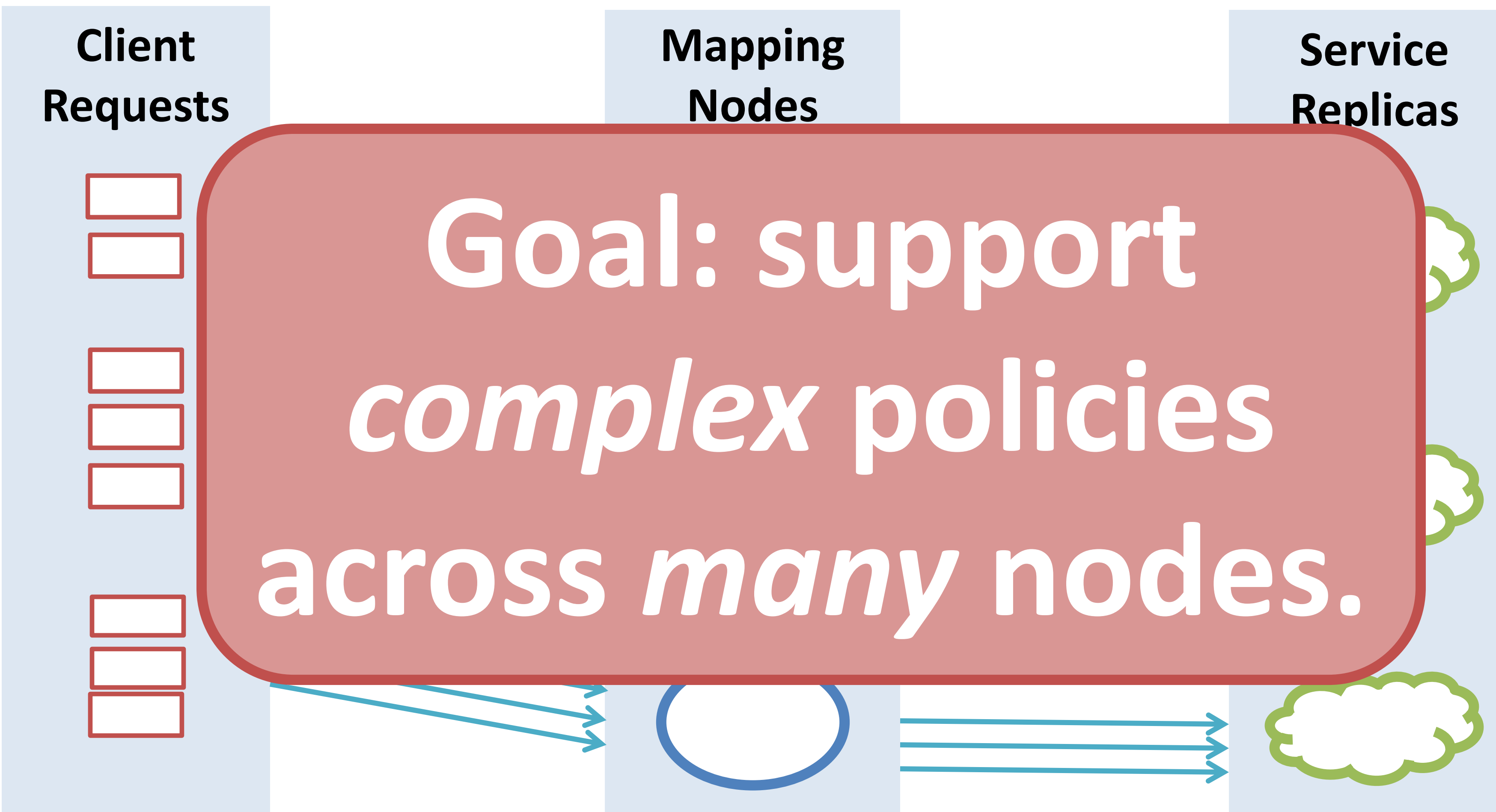
Outline

- Server selection background
- Constraint-based policy interface
- Scalable optimization algorithm
- Production deployment

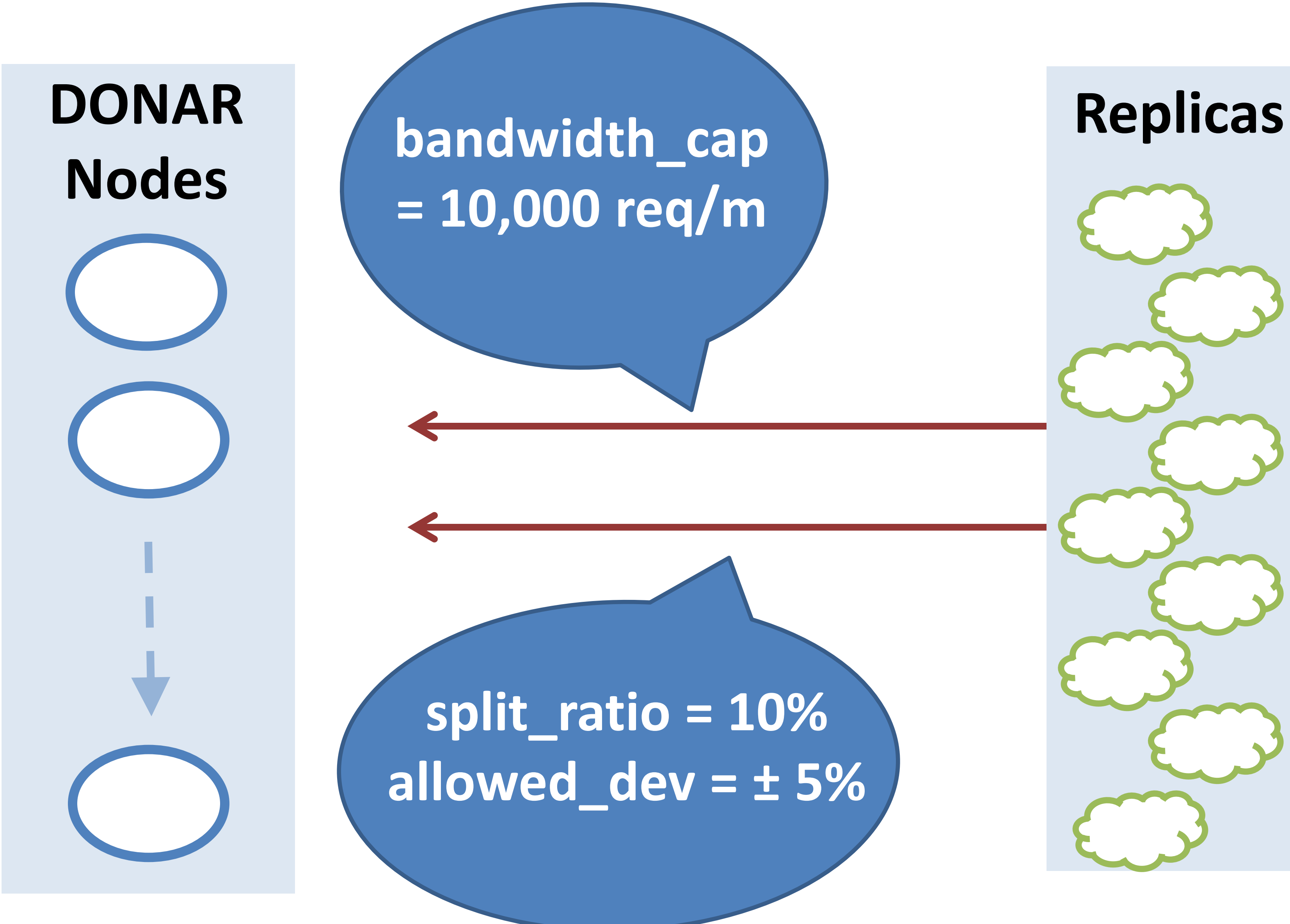
Naïve Policy Choices
Load-Aware: "Round Robin"



Naïve Policy Choices
Location-Aware: “Closest Node”



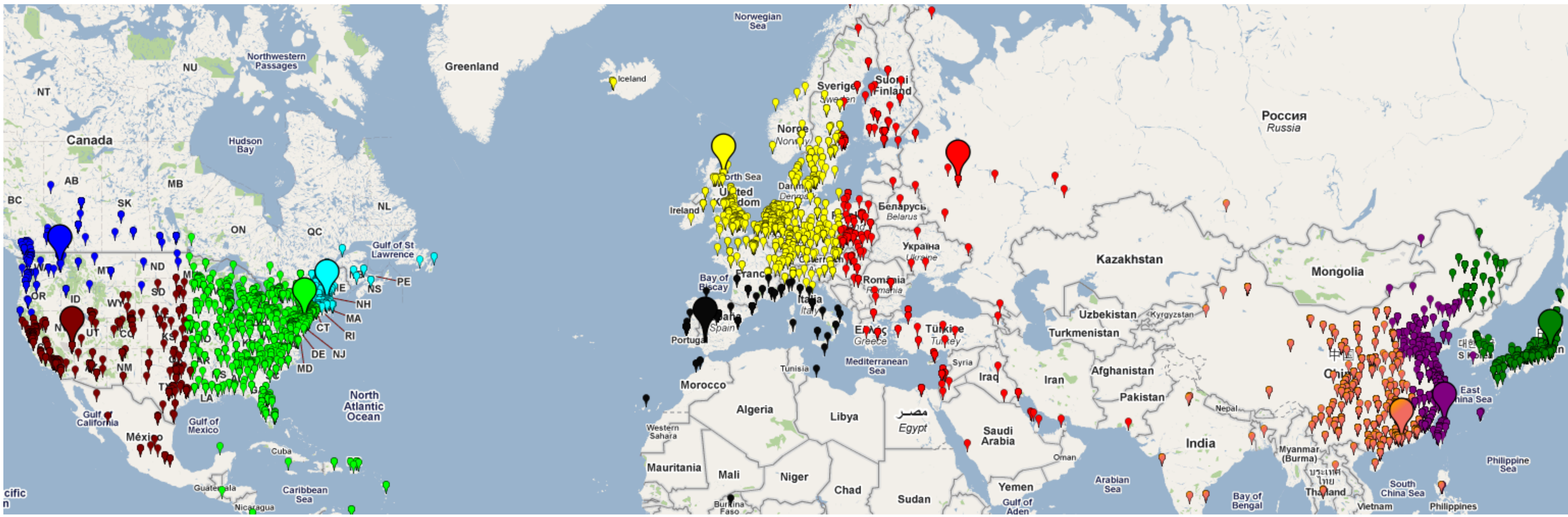
Policies as Constraints

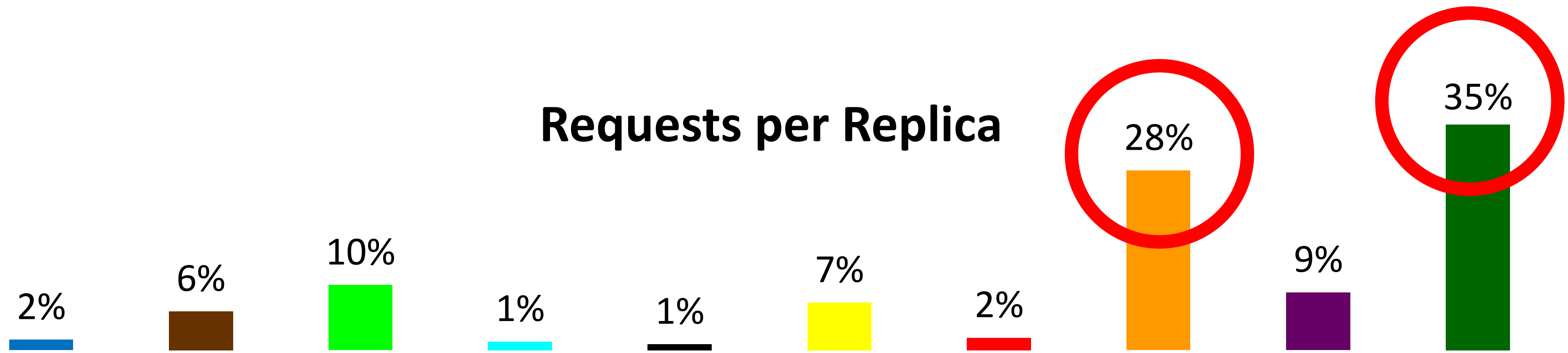


Eg. 10-Server Deployment

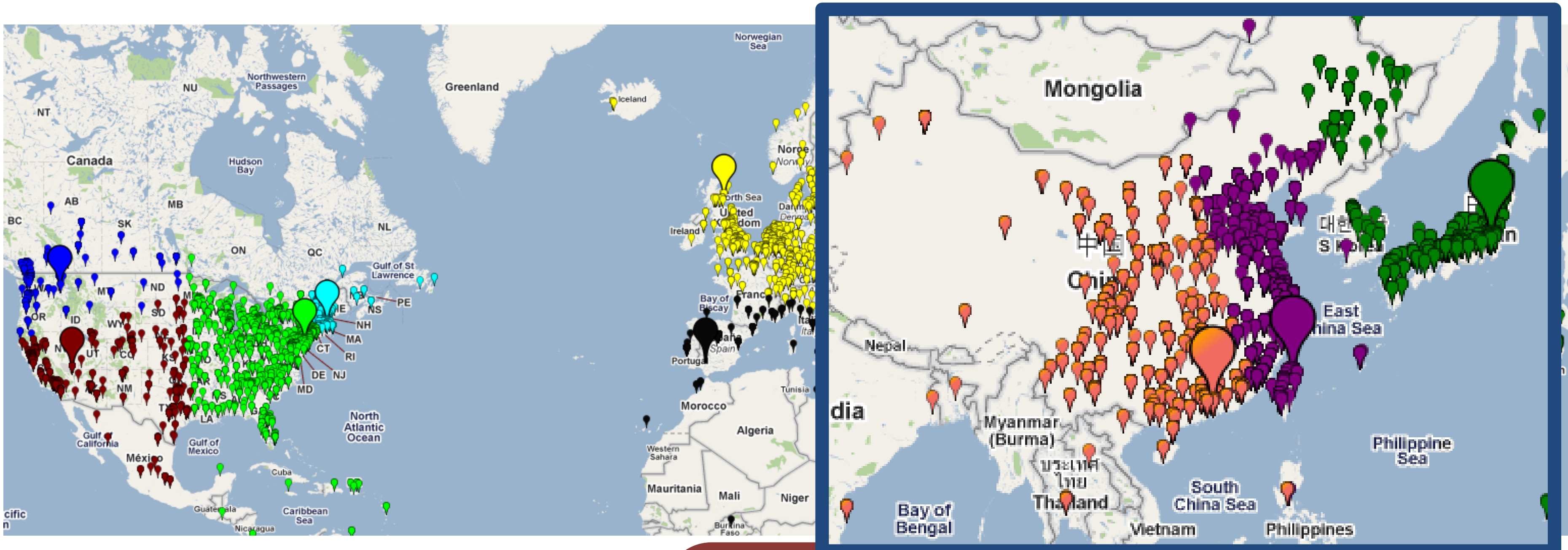
How to describe policy
with constraints?

No Constraints
Equivalent to "Closest Node"

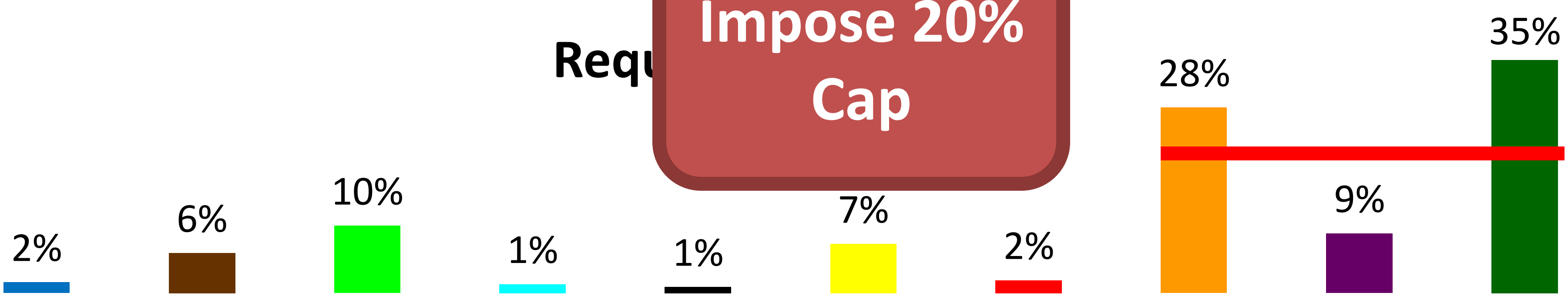




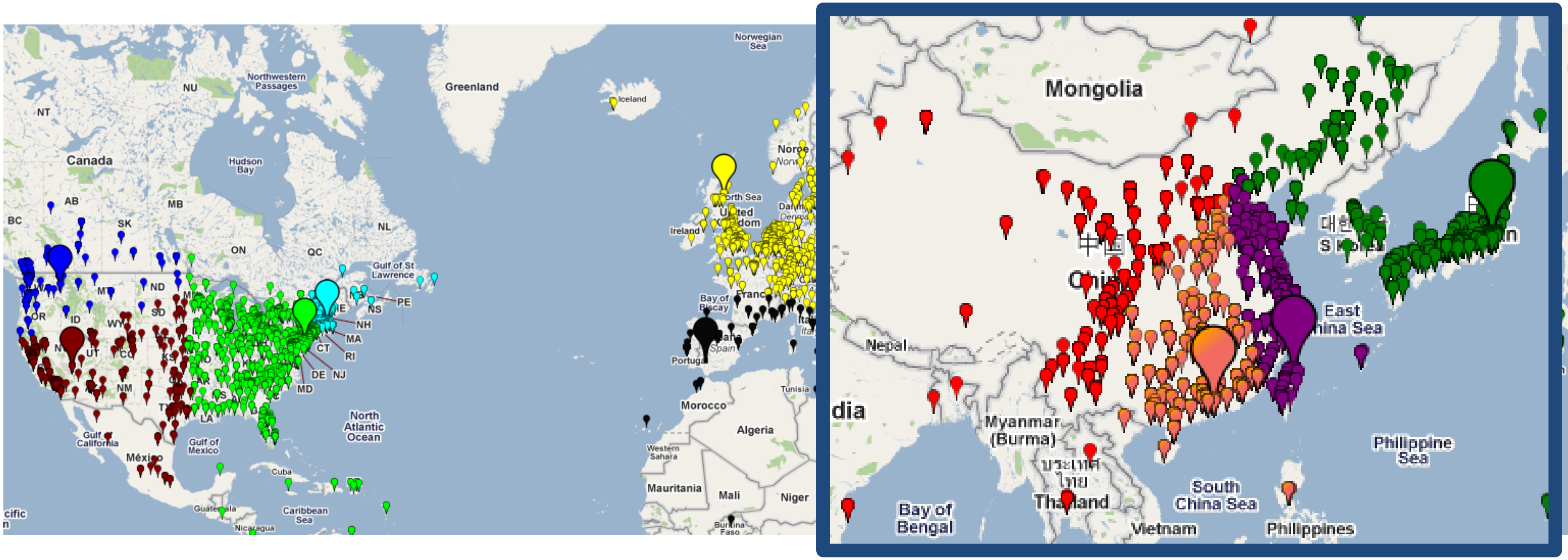
No Constraints
Equivalent to “Closest Node”



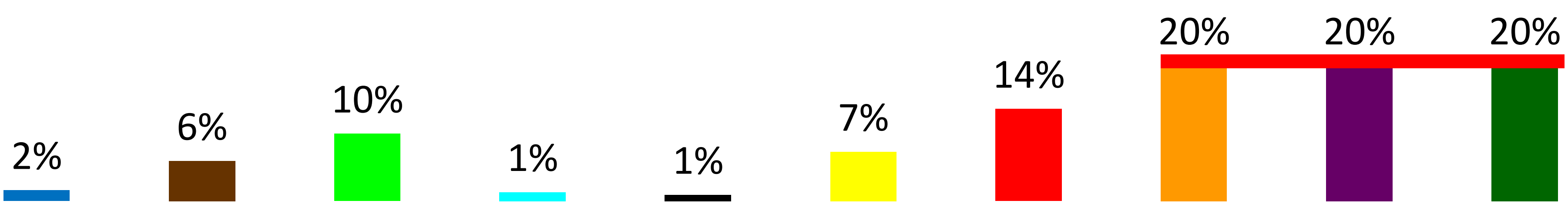
Impose 20% Cap



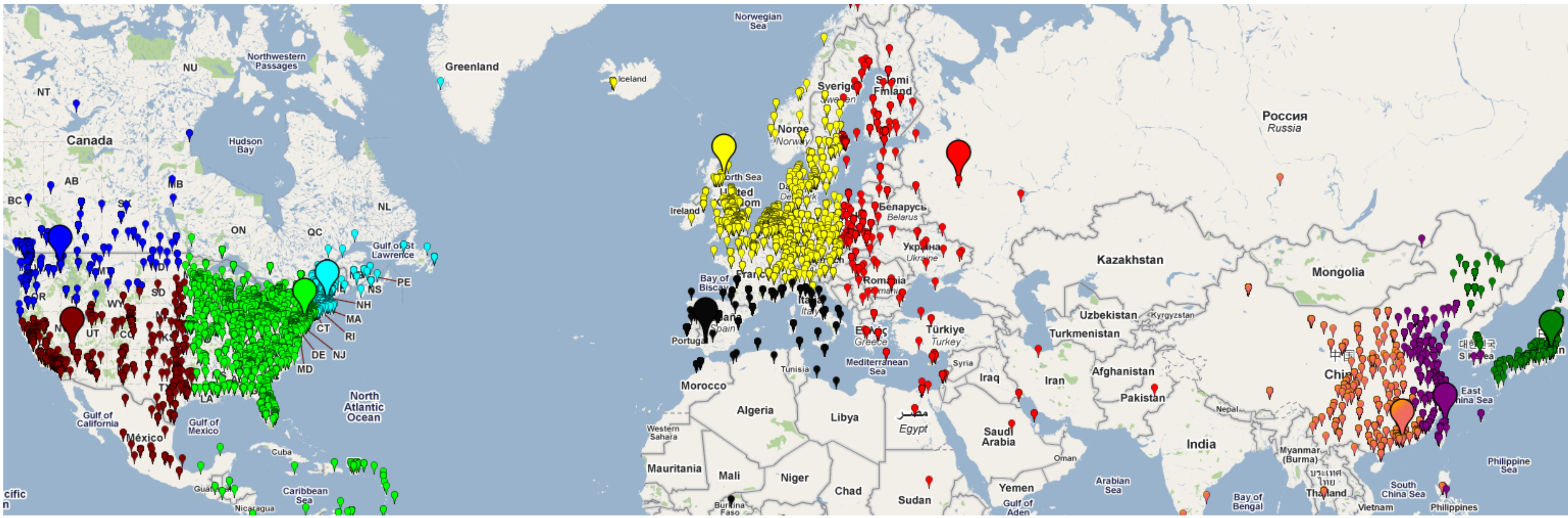
Cap as Overload Protection



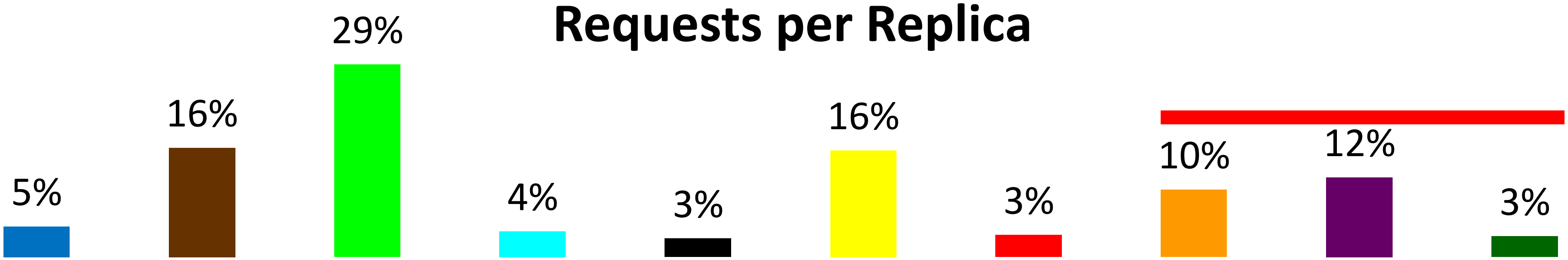
Requests per Replica



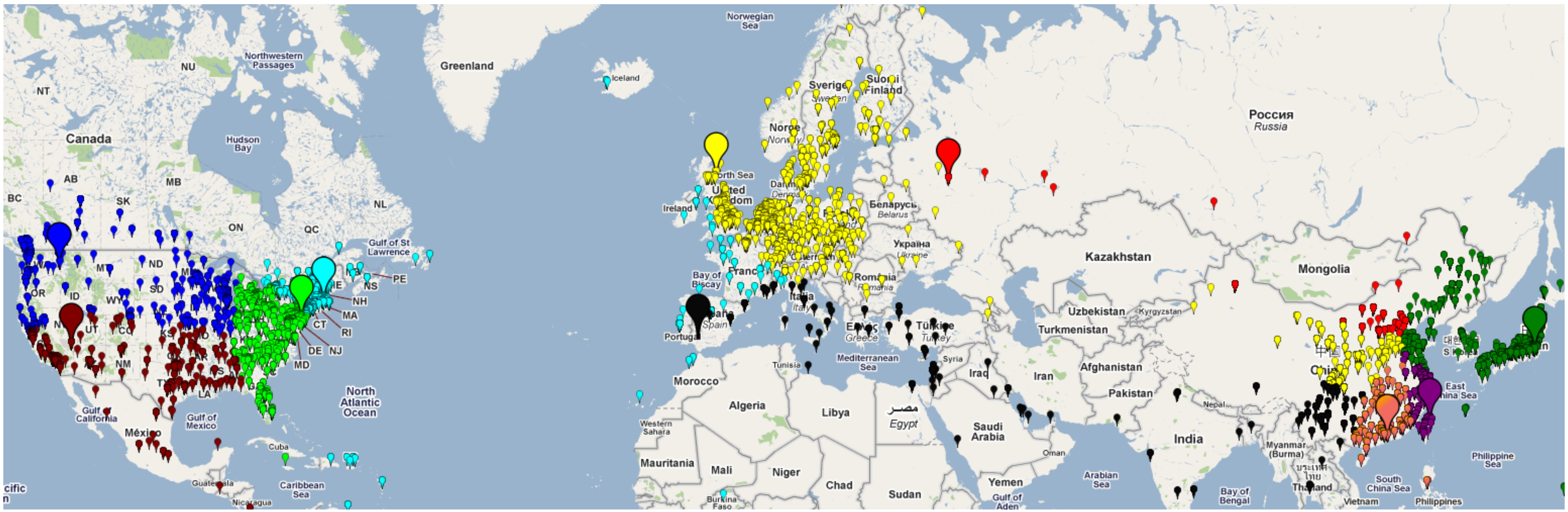
12 Hours Later…



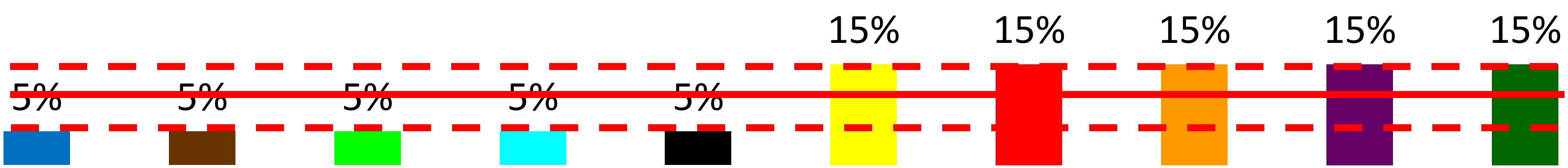
Requests per Replica



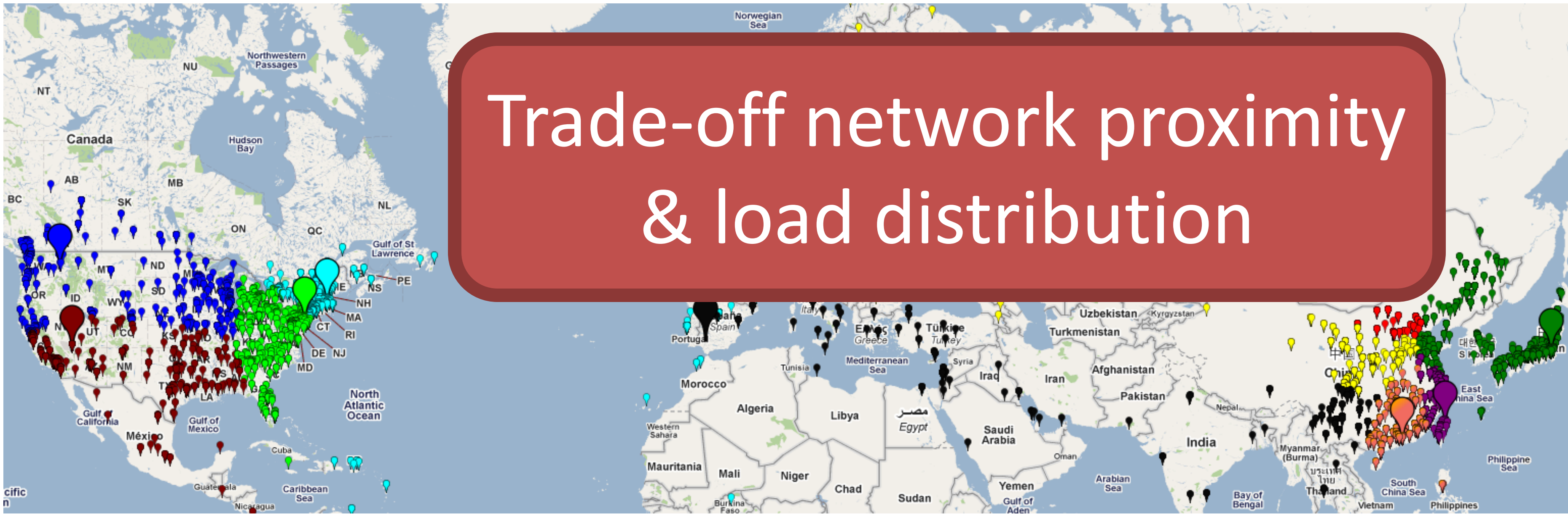
“Load Balance”
(split = 10%, tolerance = 5%)



Requests per Replica

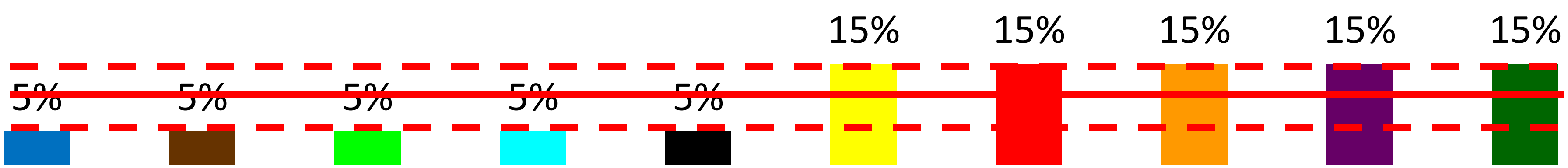


“Load Balance”
(split = 10%, tolerance = 5%)



Trade-off network proximity & load distribution

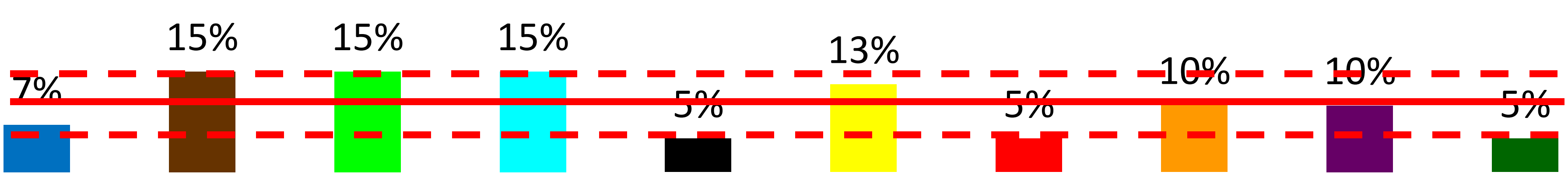
Requests per Replica



12 Hours Later…

Large range of policies by varying cap/weight

Requests per Replica



Outline

- Server selection background
- Constraint-based policy interface
- Scalable optimization algorithm
- Production deployment

Optimization: Policy Realization

Clients: $c \in C$	Nodes: $n \in N$	Replica Instances: $i \in I$

- Global LP describing "optimal" pairing

Minimize network cost

$$\min \sum_{c \in C} \sum_{i \in I} \alpha_{\boldsymbol{c}} \cdot R_{ci} \cdot cost(c, i)$$

s.t.

Server loads within tolerance

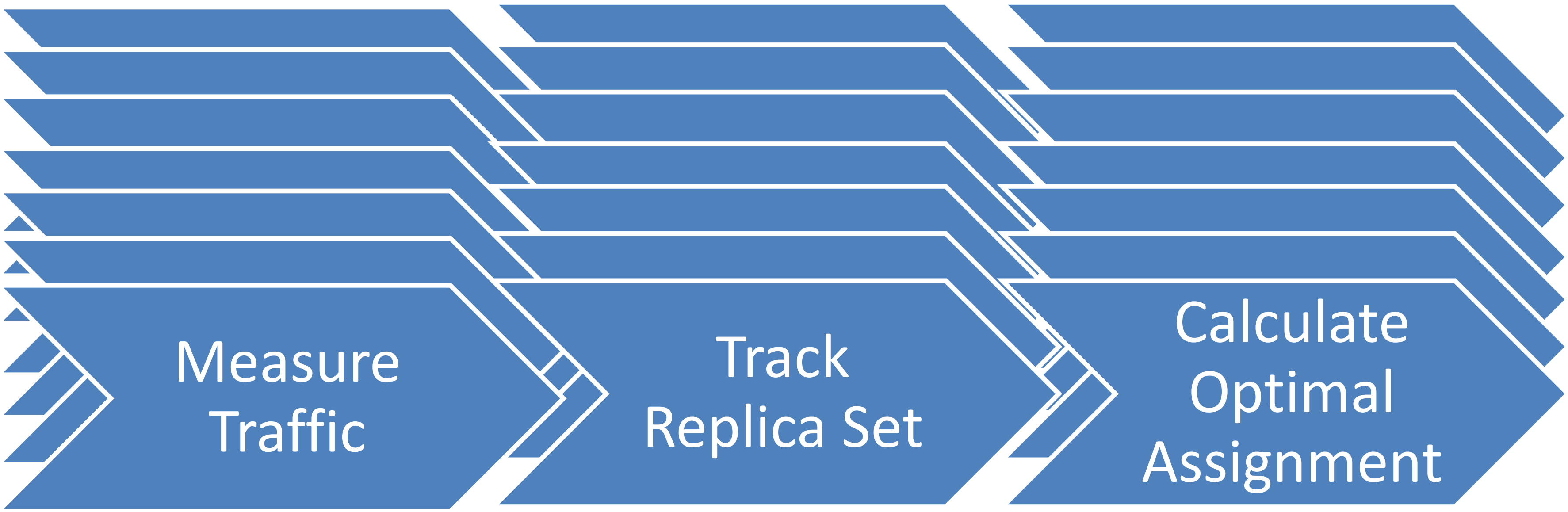
$$|P_i - \omega_i| \leq \varepsilon_i$$

Bandwidth caps met

$$B_i < B \cdot P_i$$

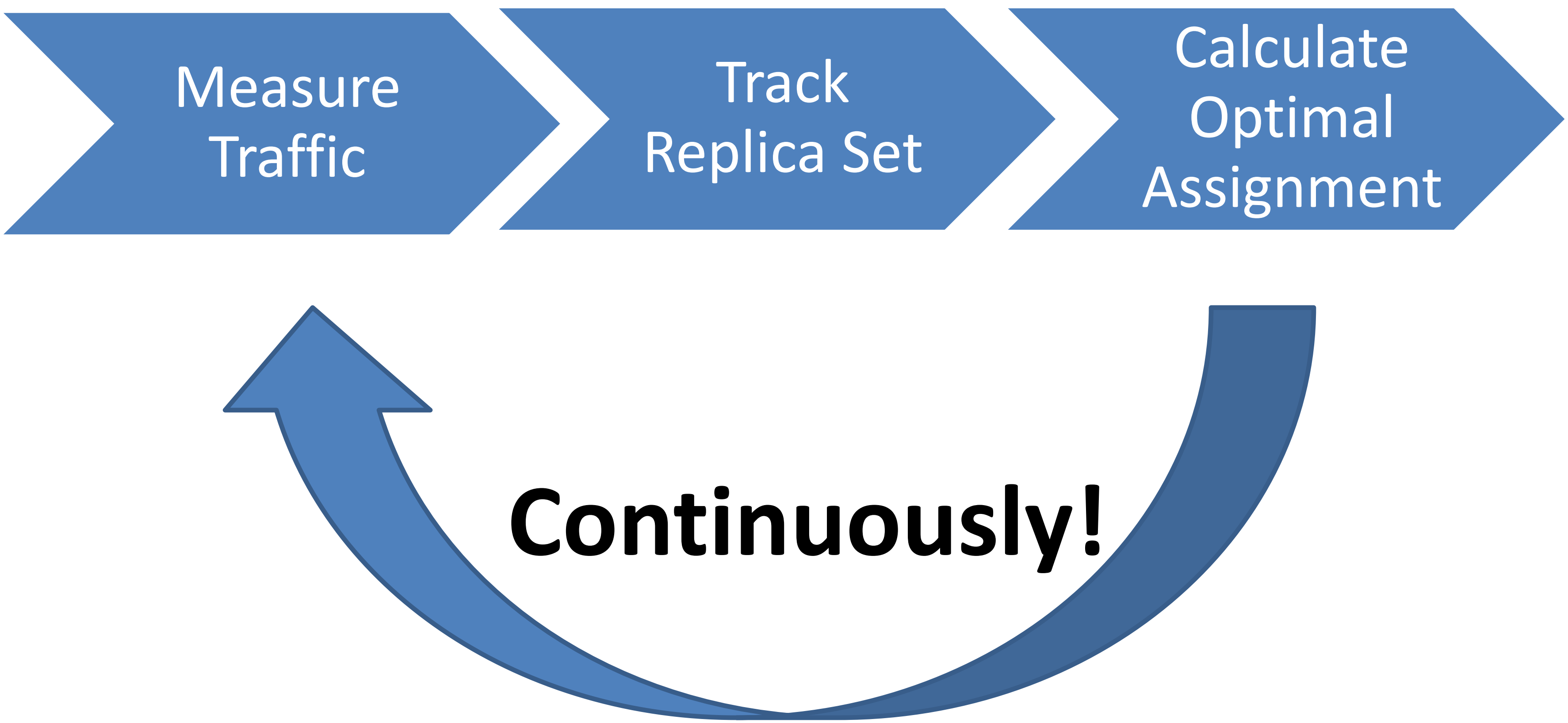
Optimization Workflow

Optimization Workflow



Per-customer!

Optimization Workflow



(respond to underlying traffic)

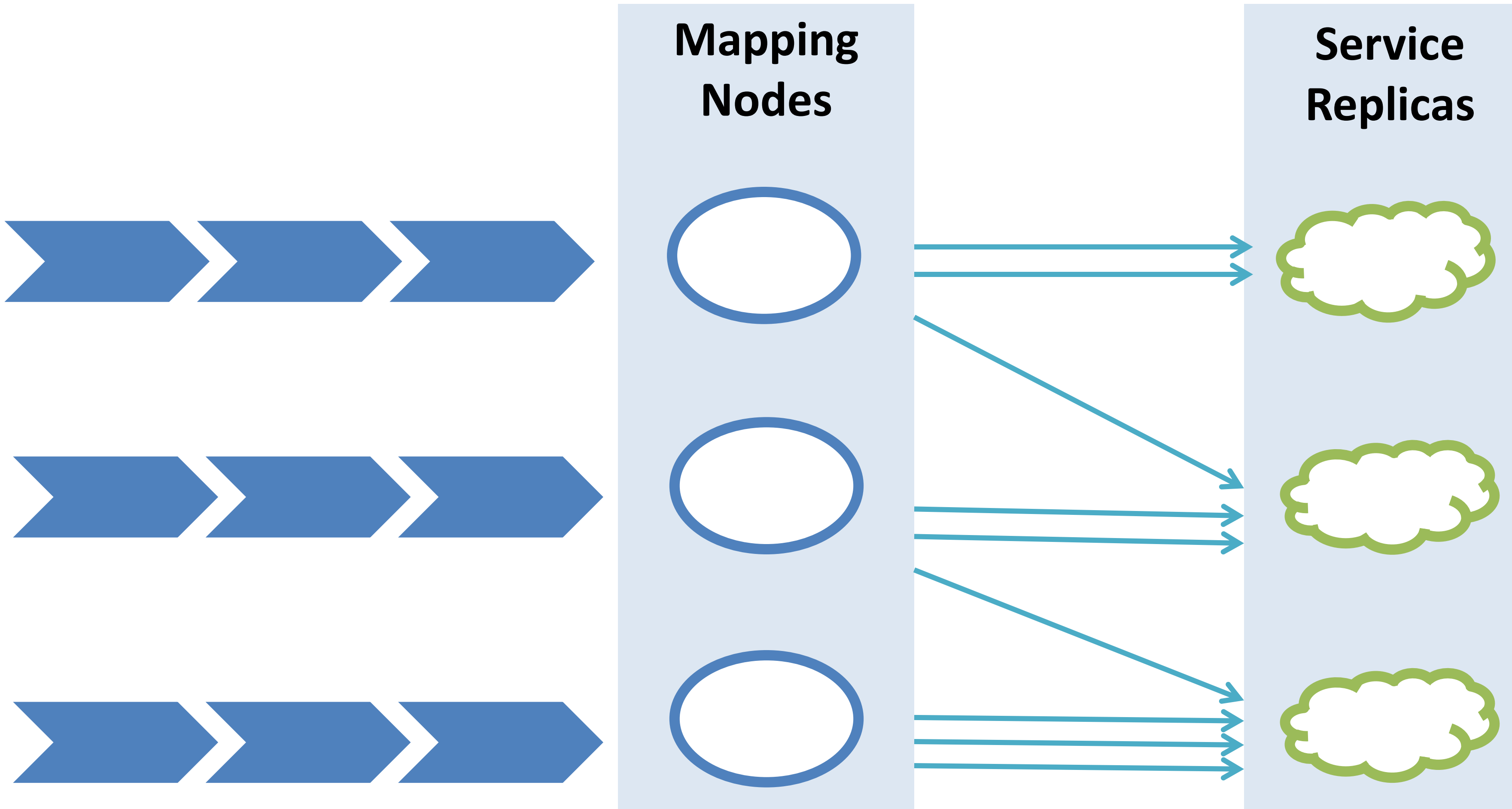
By The Numbers

	10^1	10^2	10^3	10^4
DONAR Nodes		●		
Customers		●		
replicas/customer		●		
client groups/ customer				●

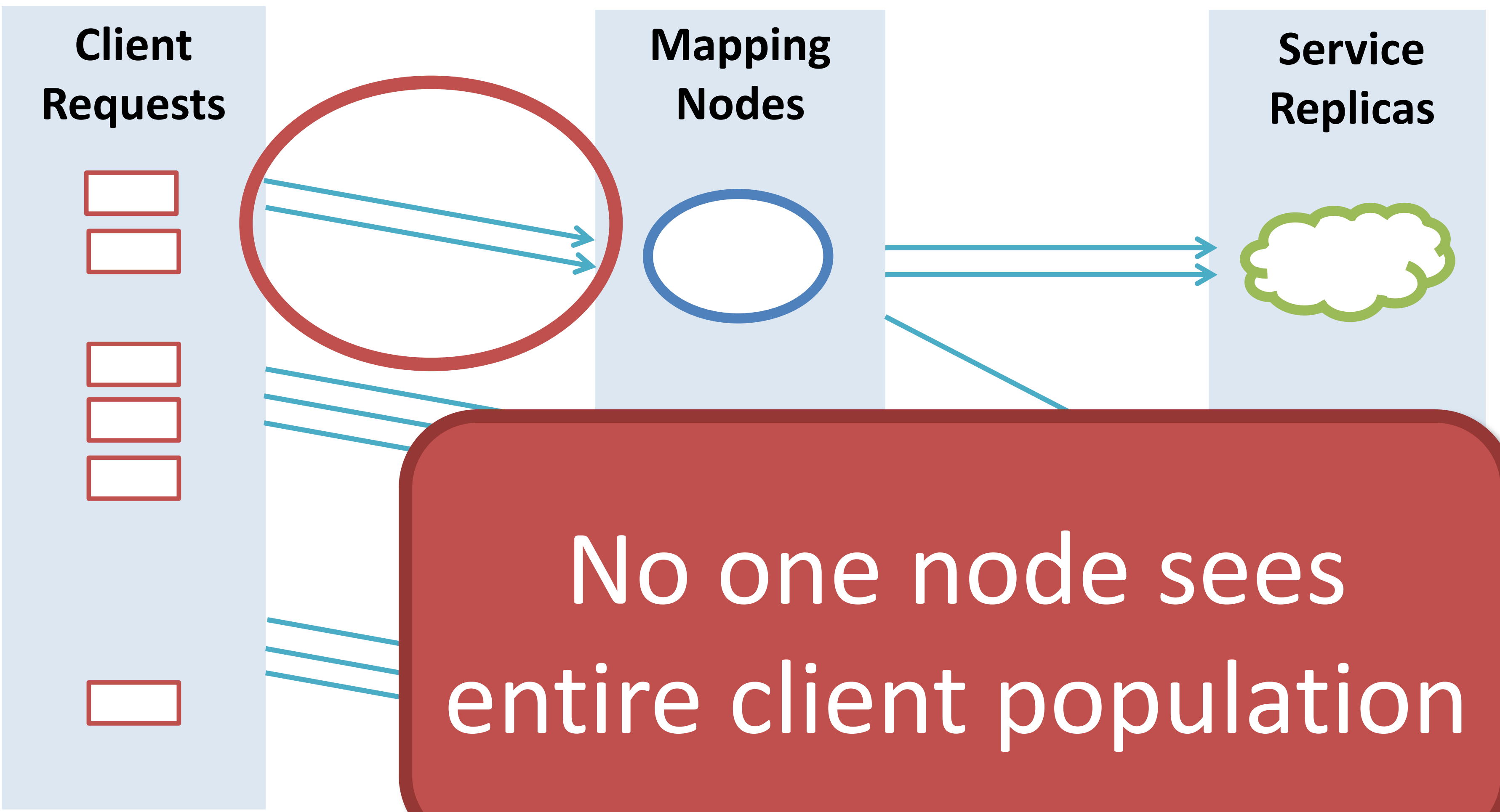
Problem for each customer:

$$\mathbf{10^2 * 10^4 = 10^6}$$

Measure Traffic & Optimize Locally?

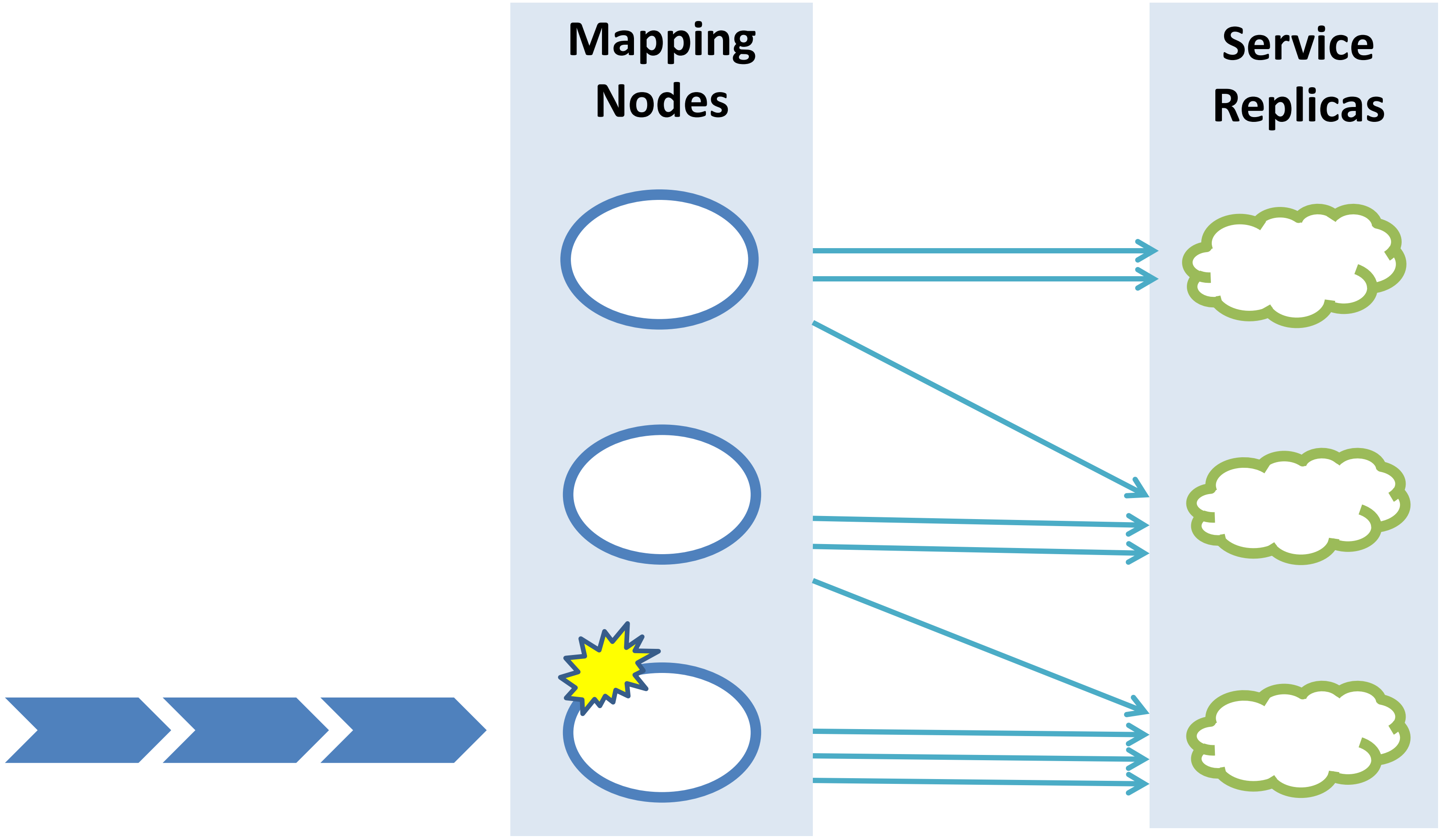


Not Accurate!

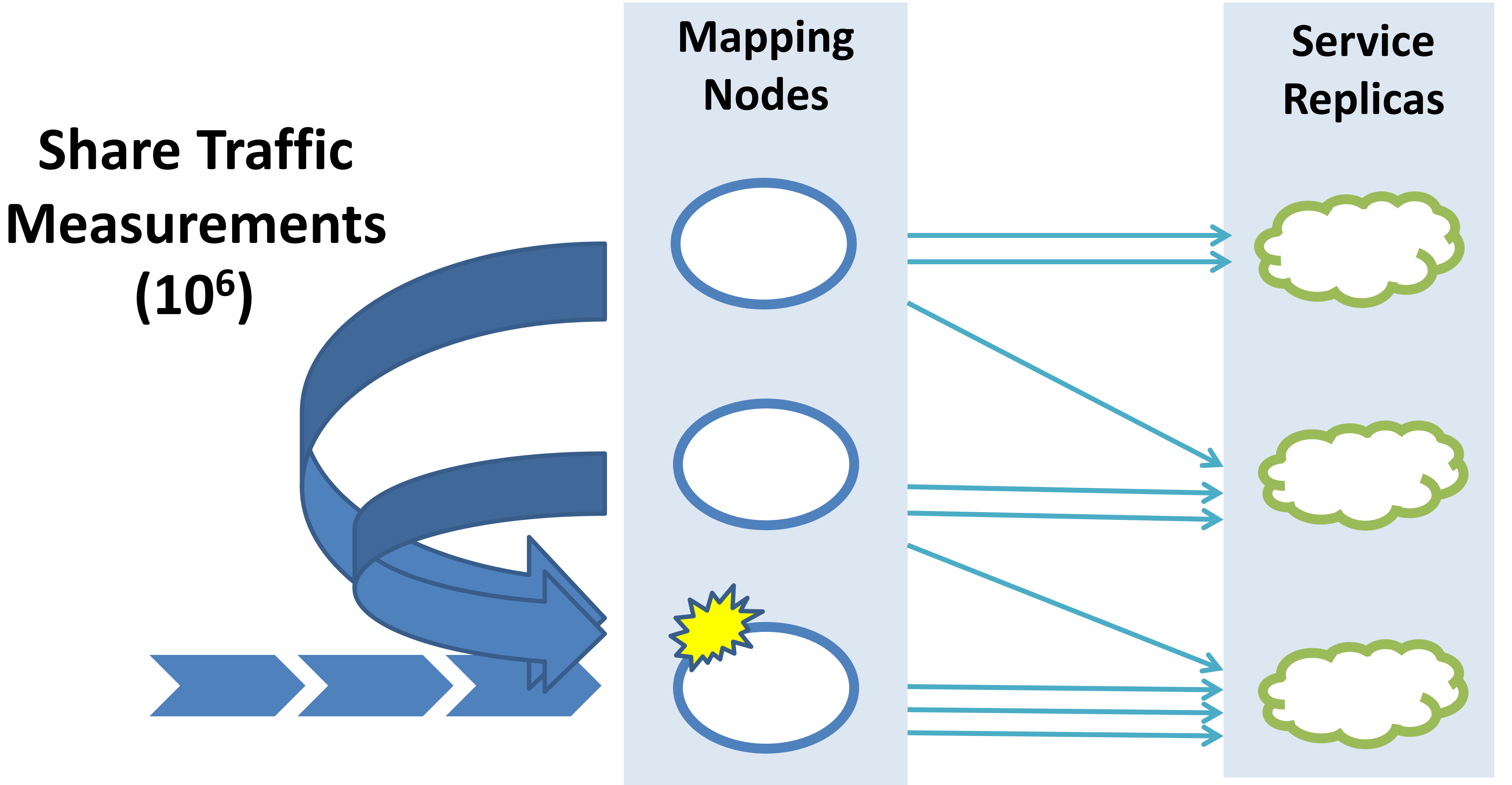


No one node sees entire client population

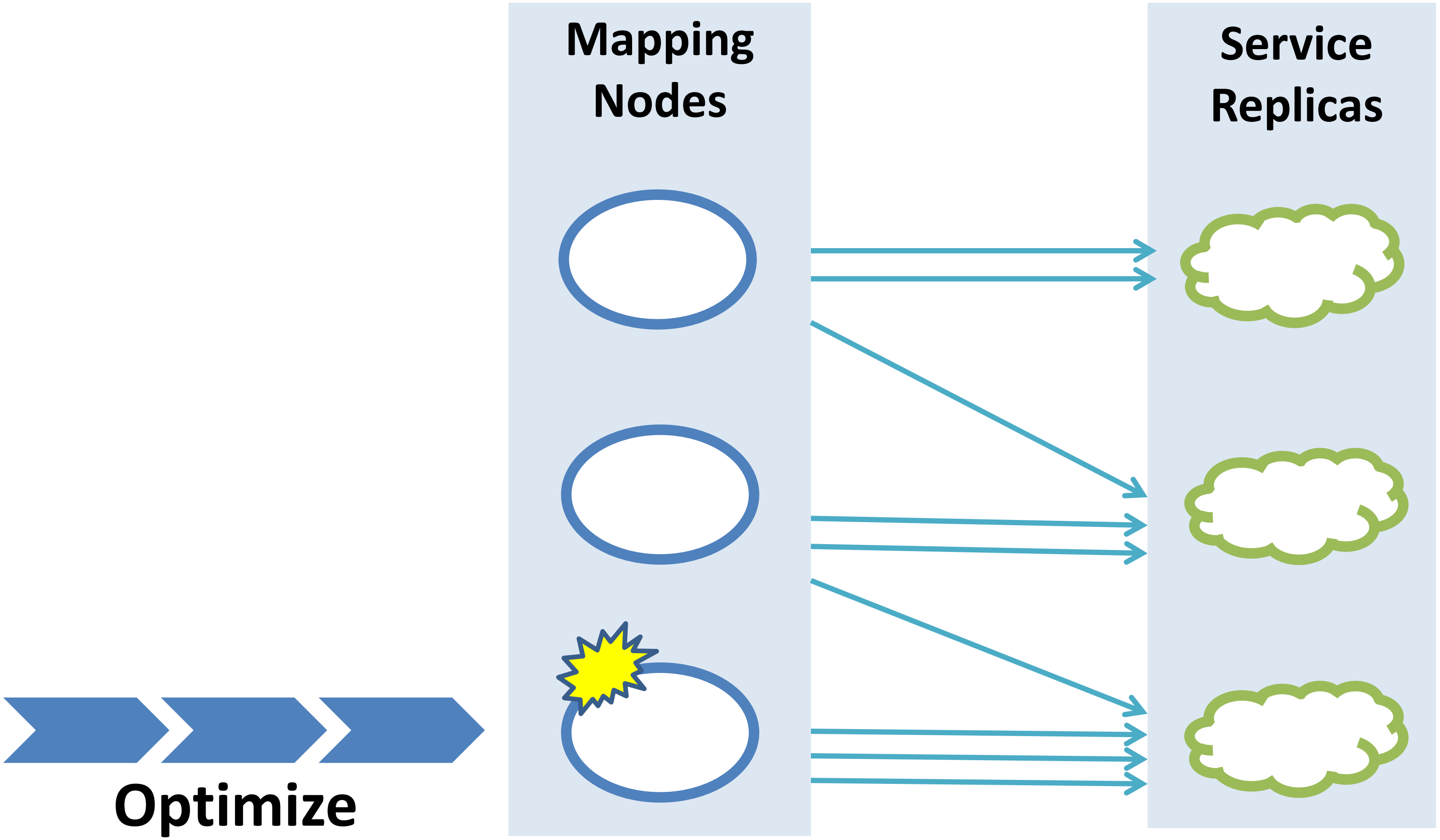
Aggregate at Central Coordinator?



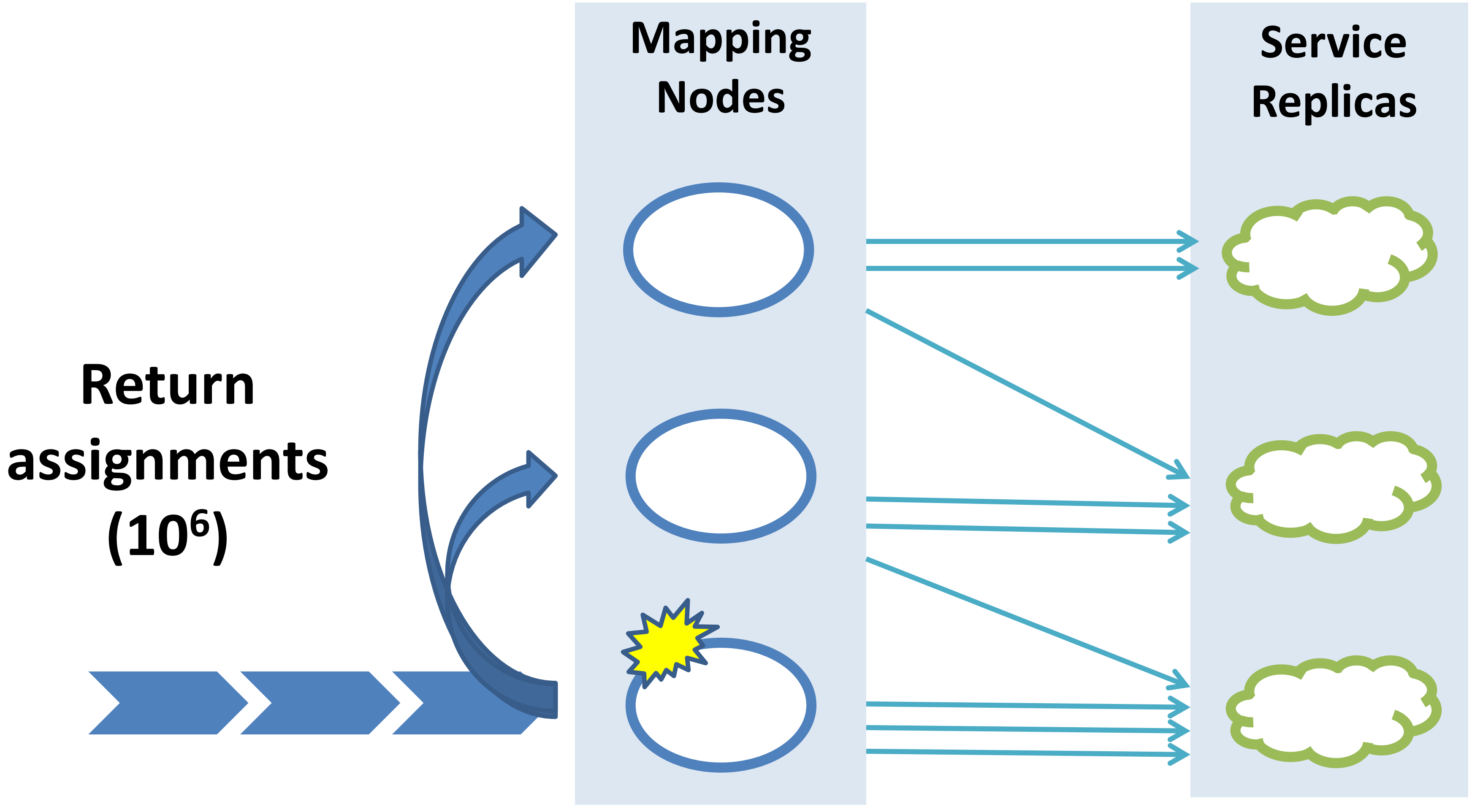
Aggregate at Central Coordinator?



Aggregate at Central Coordinator?



Aggregate at Central Coordinator?



So Far

	Accurate	Efficient	Reliable
Local only	No	Yes	Yes
Central Coordinator	Yes	No	No

Decomposing Objective Function

Traffic from c cost of mapping c to i

min

∀ cli

We also decompose constraints (more complicated)

$$\sum_{n \in N} s_n \sum_{c \in C} \sum_{i \in I} \alpha_{cn} \cdot R_{nci} \cdot cost(c,i)$$

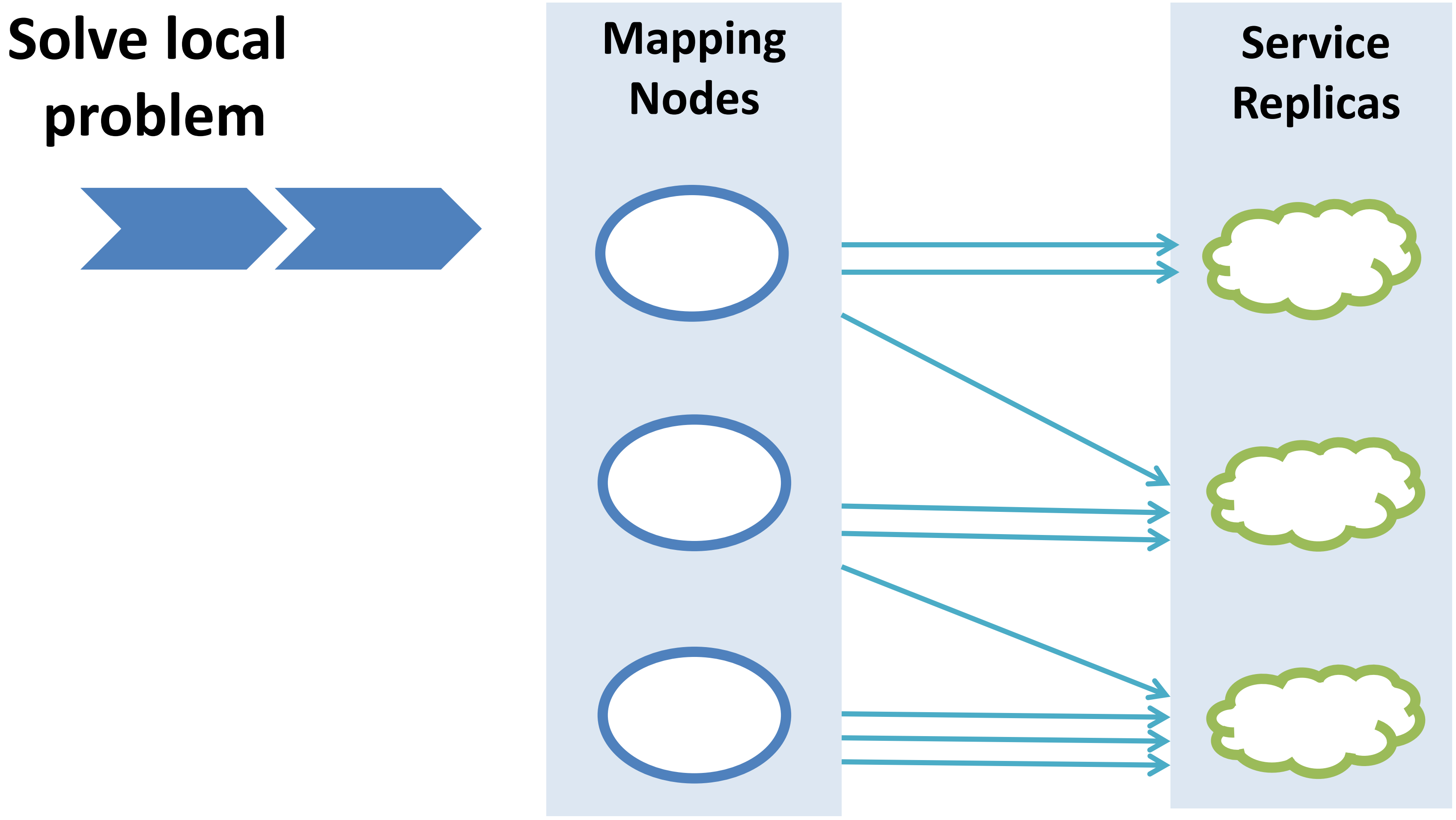
∀ nodes Traffic to this node

Decomposed Local Problem For Some Node (n*)

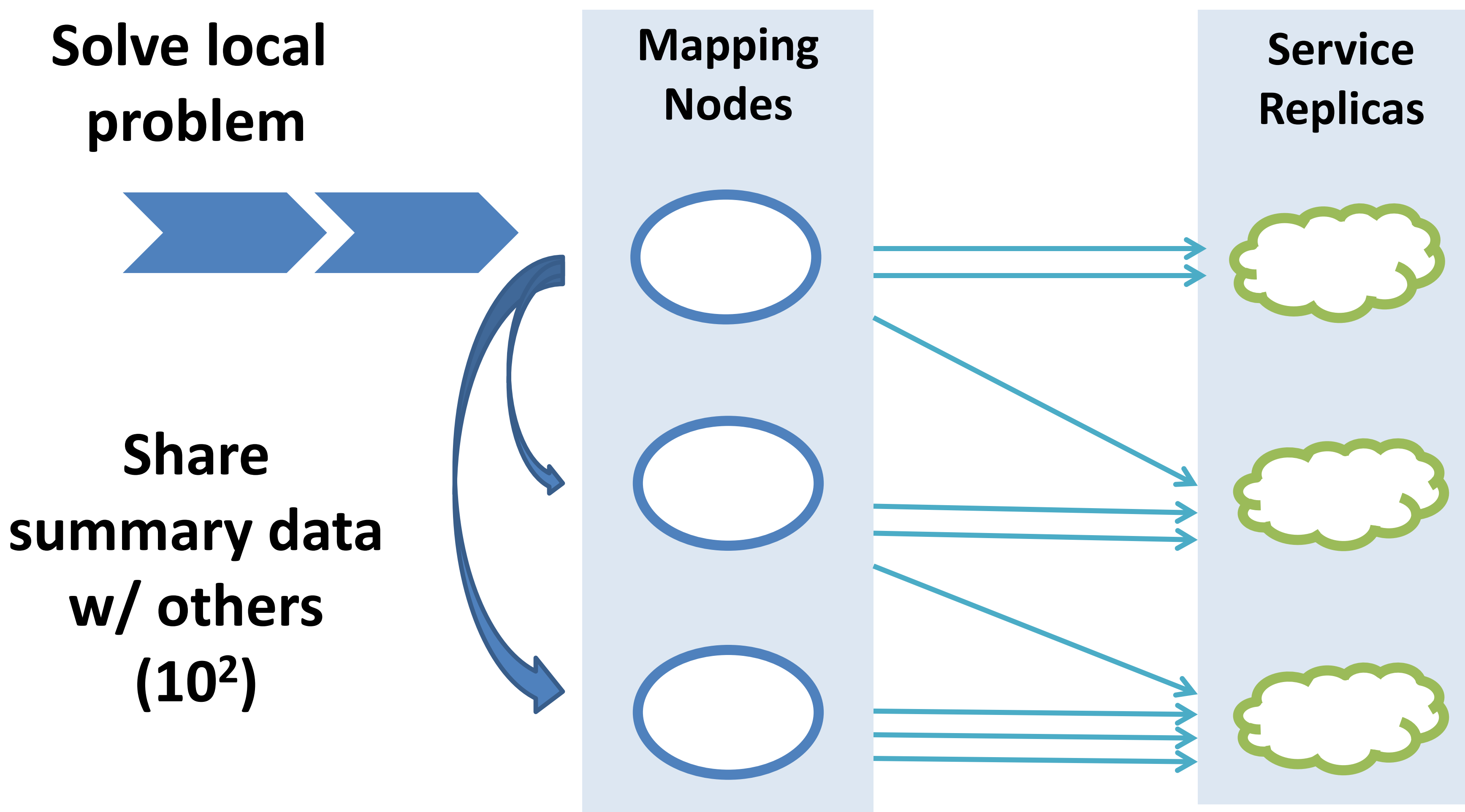
$load_i$ = f(prevailing load on each server + load I will impose on each server)

$$\min \quad \underbrace{\forall_i load_i}_{\text{Global load information}} + s_{n^*} \sum_{c \in C} \sum_{i \in I} \underbrace{\alpha_{cn^*} \cdot R_{n^*ci} \cdot cost(c,i)}_{\text{Local distance minimization}}$$

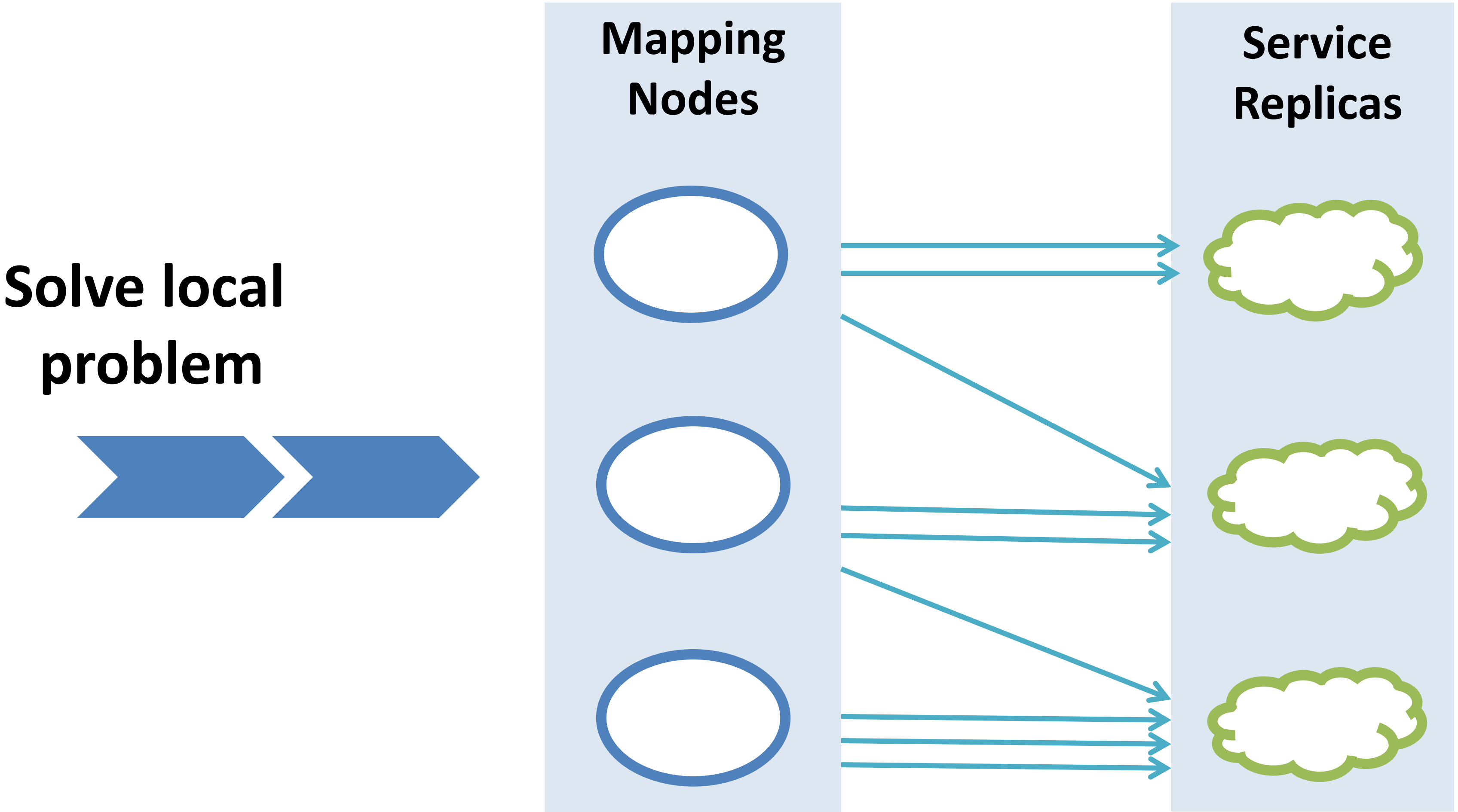
DONAR Algorithm



DONAR Algorithm

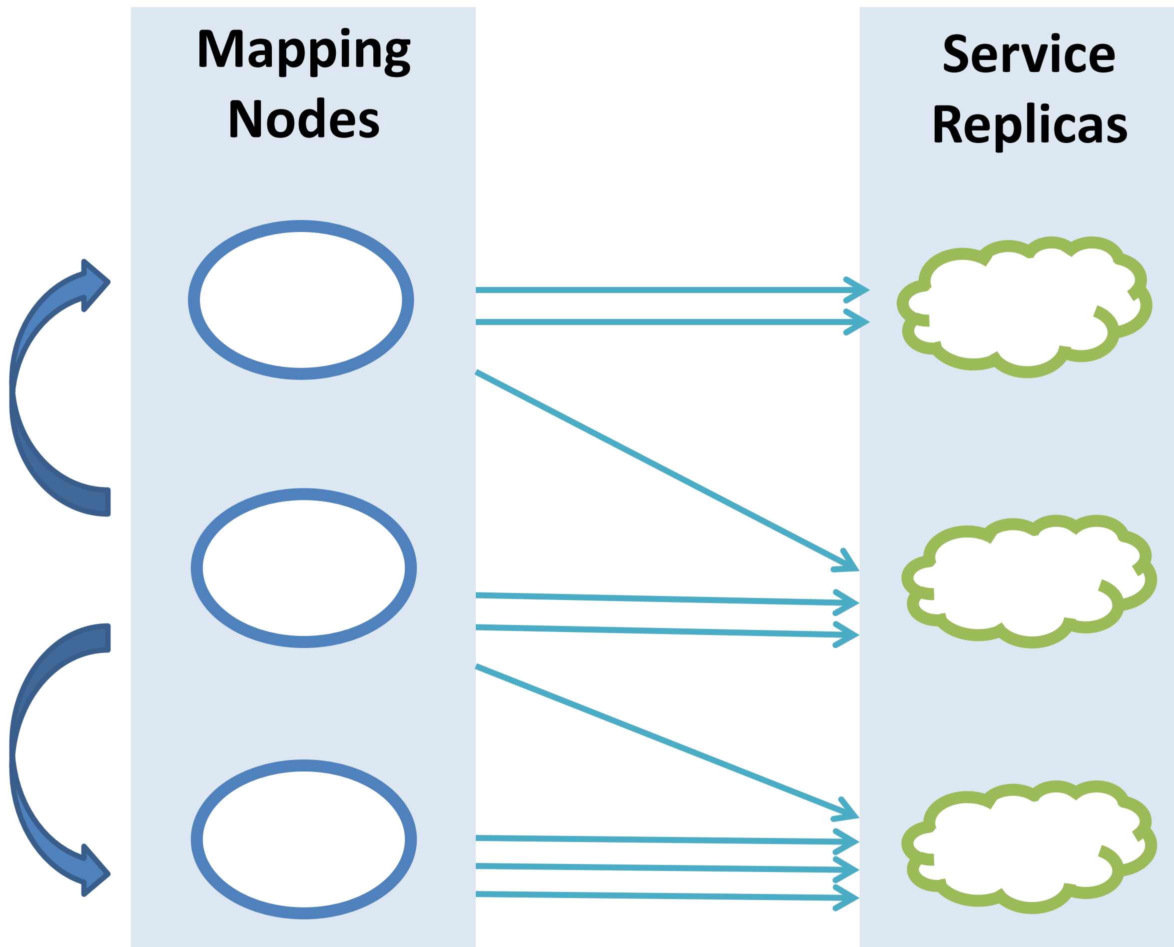


DONAR Algorithm



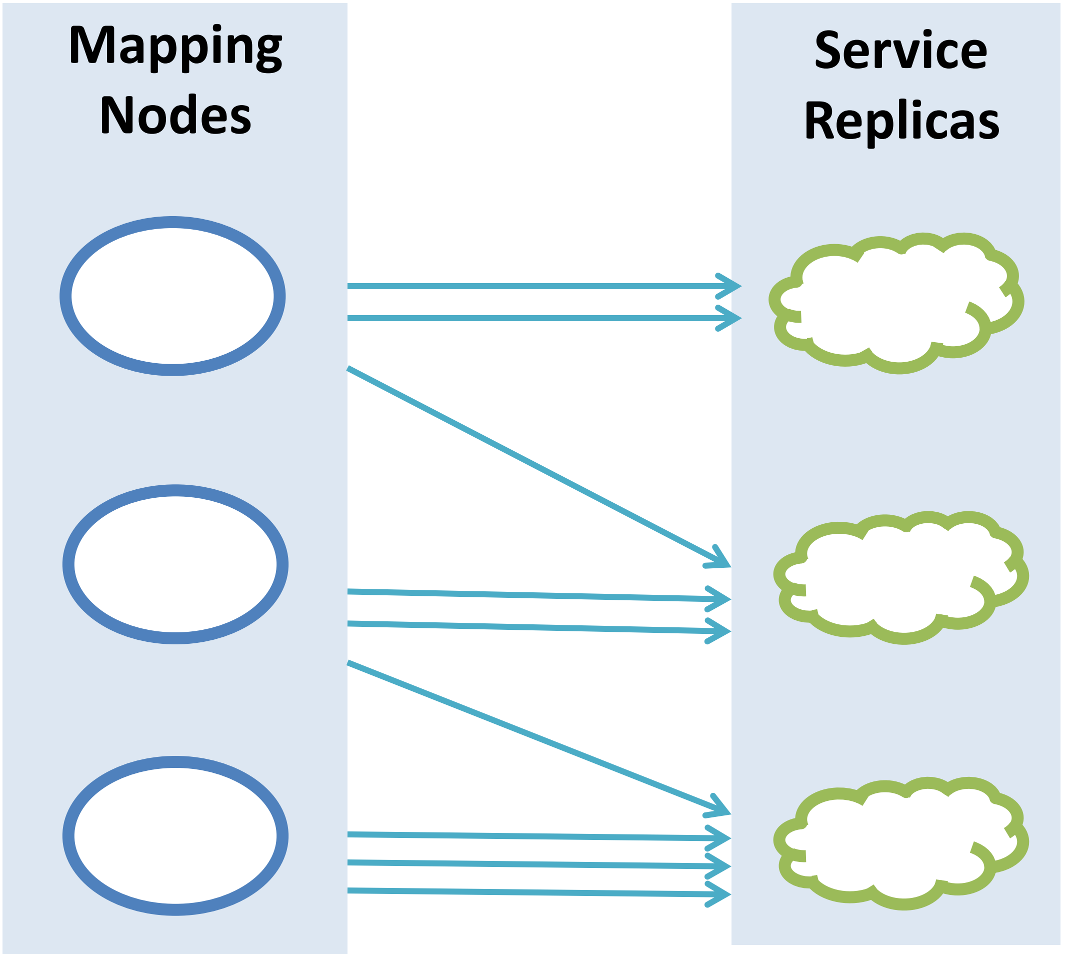
DONAR Algorithm

Share summary data w/ others (10^2)



DONAR Algorithm

- **Provably converges to global optimum**
- **Requires no coordination**
- **Reduces message passing by 10^4**



Better!

	Accurate	Efficient	Reliable
Local only	No	Yes	Yes
Central Coordinator	Yes	No	No
DONAR	Yes	Yes	Yes

Outline

- Server selection background
- Constraint-based policy interface
- Scalable optimization algorithm
- Production deployment

Production and Deployment

- Publicly deployed 24/7 since November 2009
- IP2Geo data from Quova Inc.
- Production use:
  - All MeasurementLab Services (incl. FCC Broadband Testing)
  - CoralCDN
- Services around 1M DNS requests per day

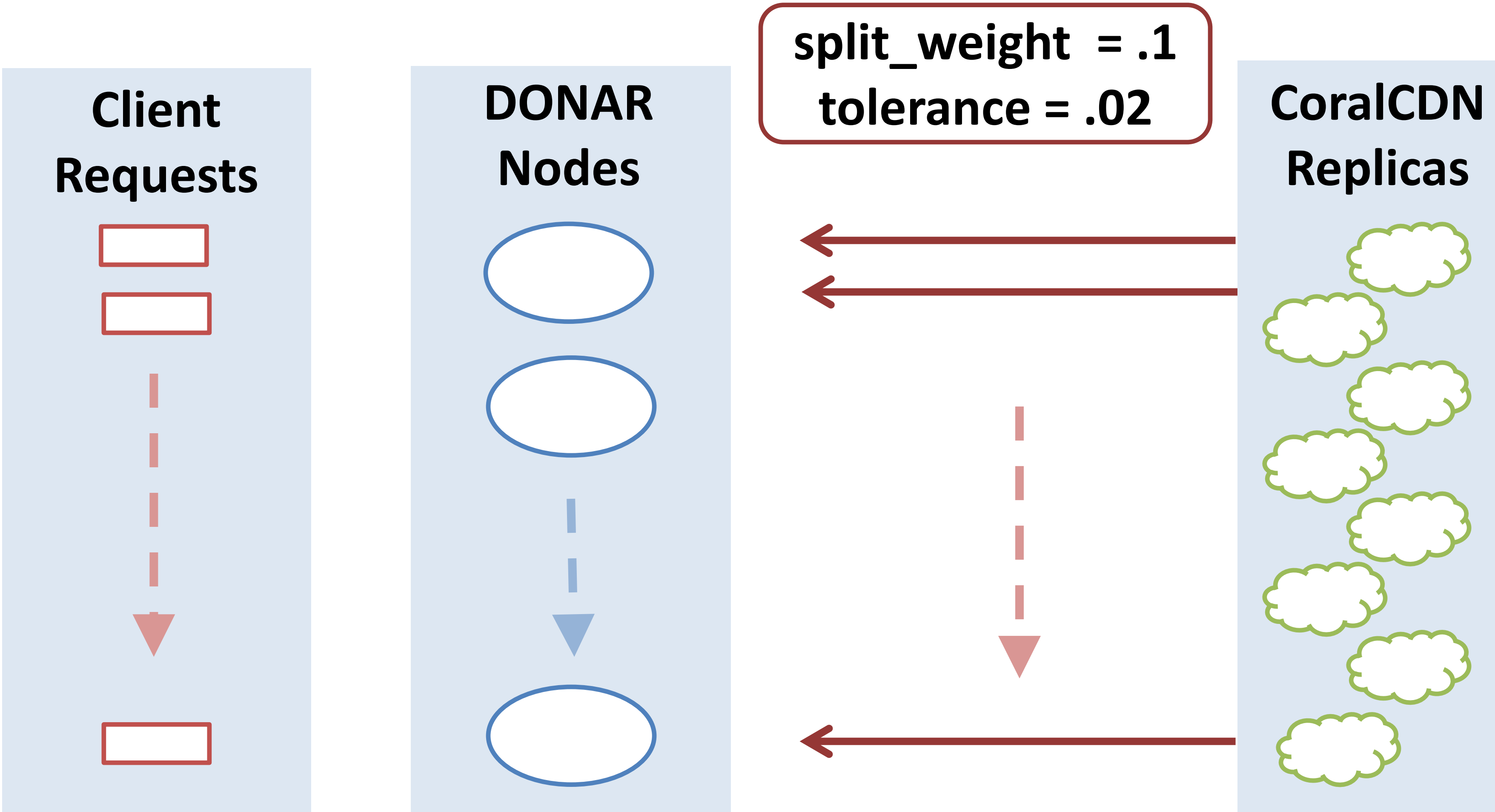
Systems Challenges (See Paper!)

- **Network availability**
  Anycast with BGP

- **Reliable data storage**
  Chain-Replication with Apportioned Queries

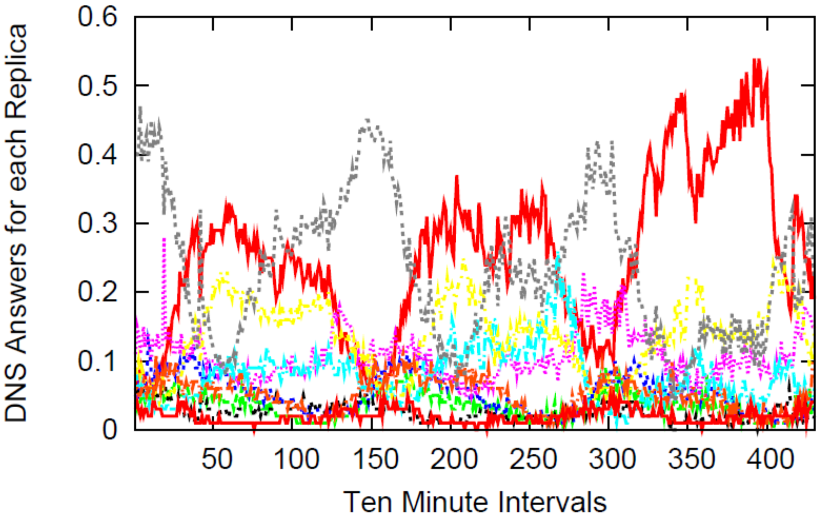
- **Secure, reliable updates**
  Self-Certifying Update Protocol

CoralCDN Experimental Setup

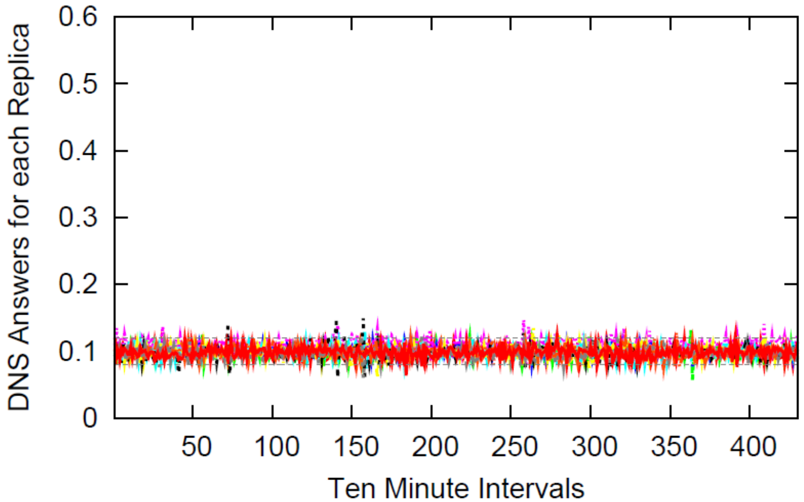


Results: DONAR Curbs Volatility

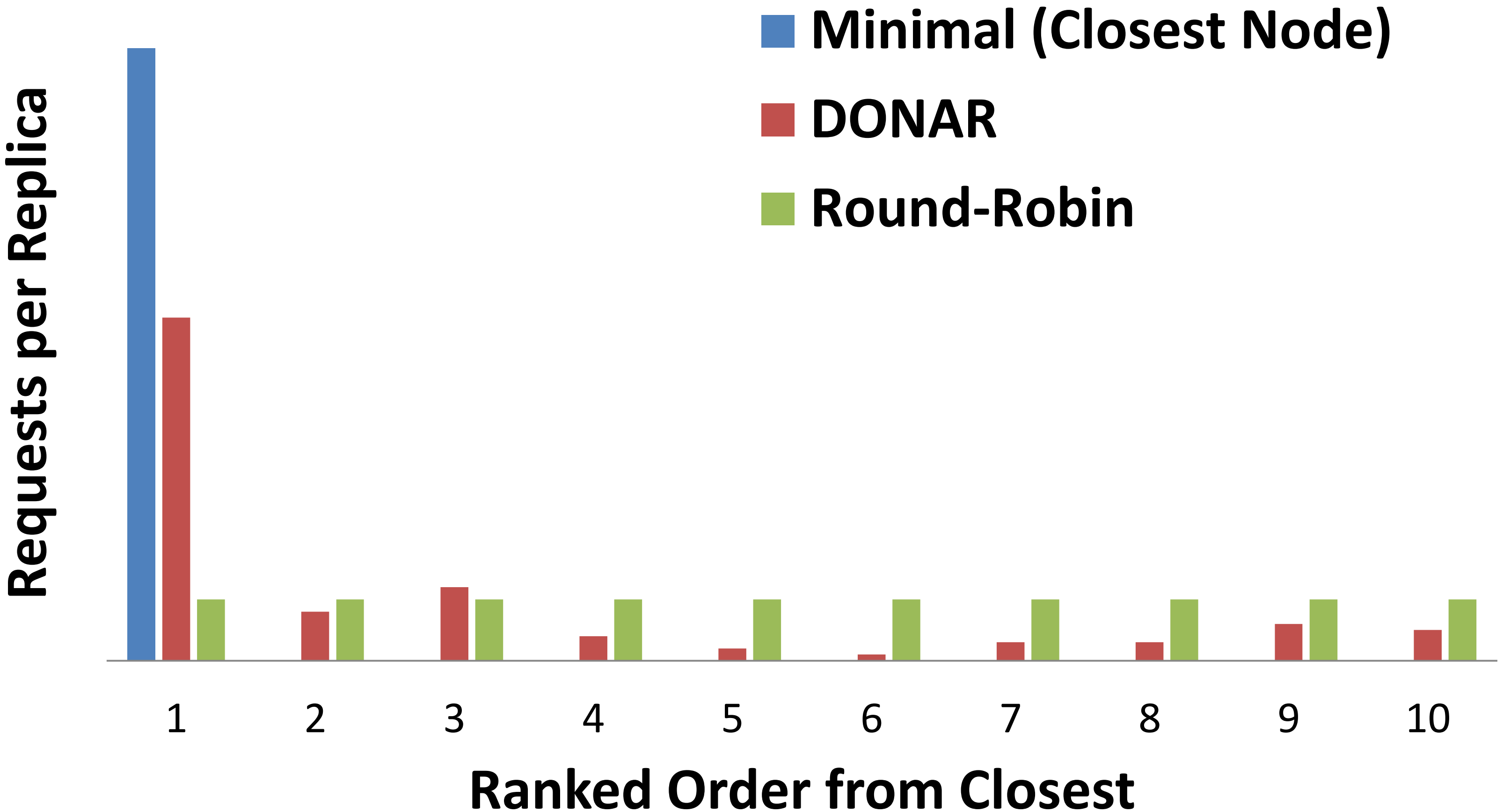
“Closest Node” policy



DONAR “Equal Split” Policy



Results: DONAR Minimizes Distance



Conclusions

- Dynamic server selection is difficult
  - Global constraints
  - Distributed decision-making
- Services reap benefit of outsourcing to DONAR.
  - Flexible policies
  - General: Supports DNS & HTTP Proxying
  - Efficient distributed constraint optimization
- Interested in using? Contact me or visit http://www.donardns.org.

Questions?

Related Work (Academic and Industry)

- Academic
  - Improving network measurement
    - iPlane: An informationplane for distributed services
      H. V. Madhyastha, T. Isdal, M. Piatek, C. Dixon, T. Anderson,
      A. Krishnamurthy, and A. Venkataramani, “,” in OSDI, Nov. 2006
  - “Application Layer Anycast”
    - OASIS: Anycast for Any Service
      Michael J. Freedman, Karthik Lakshminarayanan, and David Mazières
      Proc. 3rd USENIX/ACM Symposium on Networked Systems Design and Implementation
      (NSDI '06) San Jose, CA, May 2006.

- Proprietary
  - Amazon Elastic Load Balancing
  - UltraDNS
  - Akamai Global Traffic Management

Doesn’t [Akamai/UltraDNS/etc] Already Do This?

- Existing approaches use alternative, centralized formulations.

- Often restrict the set of nodes per-service.

- Lose benefit of large number of nodes (proxies/DNS servers/etc).