Baiji: Domain Planning for CDNs under the 95th Percentile Billing Model

Juan Vanerio [1,2]   Huiran Liu [3]   Qi Zhang [3]   Stefan Schmid [3,4]

[1]Faculty of Computer Science, University of Vienna, Austria

[2]AIT Austrian Institute of Technology, Austria

[3]INET, Faculty IV (EECS), Technical University Berlin, Germany

[4]Fraunhofer SIT, Germany

June 3-6, 2024

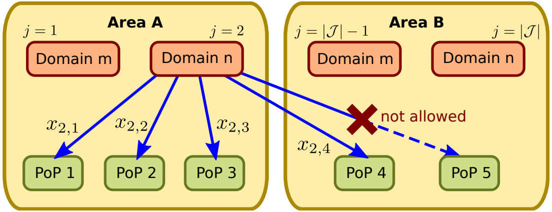
IFIP/IEEE Networking 2024
Thessaloniki, Greece

Content Delivery Networks (CDN)

- Network of geographically distributed cache servers.
- Crucial component in delivering domain content quickly to users worldwide.
  - Video: 65% of Internet traffic, mostly served from CDNs.
- **Reduced Latency**: Smaller RTTs result in faster service.
- **Content Distribution**:
  - Servers located in Points-of-Presence (PoPs)
  - Close to end-users → reduced latency → faster service.

Domain Planning

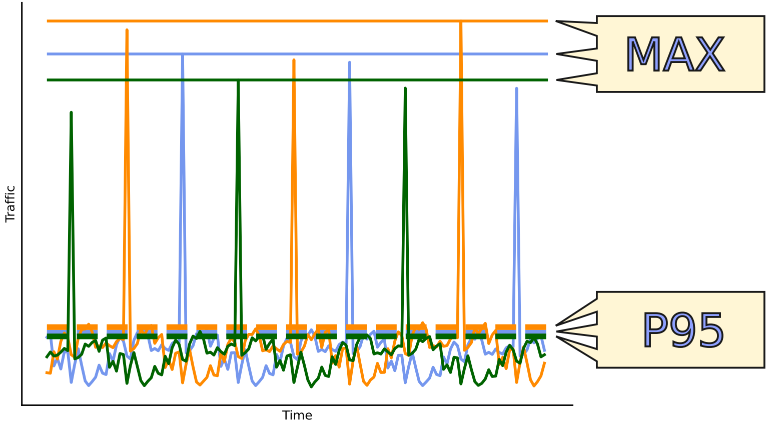
- **Key Challenge: How to plan the placement of domains to PoPs?**
  - Service quality.
  - Operational costs to pay ISPs.
  - Cannot change often (static).



- **Ideally: Minimize costs without compromising service quality.**

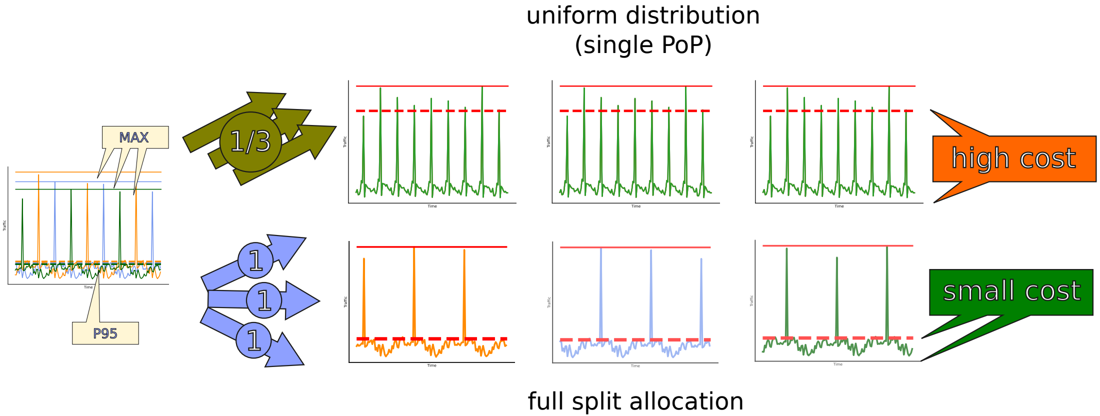
95th-percentile Billing Model

- Bandwidth usage is sampled every 5 minutes and the **top 5% is for free!**
- Widely used for transmission traffic (backbone, DC,...).



Oportunity

Two possible domain content planning alternatives

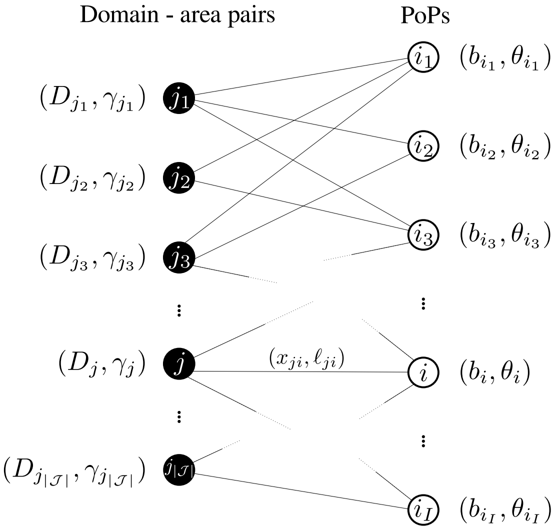


Planning Requirements

Real systems have practical requirements!

- Demand conservation (all requests must be satisfied!).
- PoPs have committed access rates.
- Content cannot be served from distant PoPs $\rightarrow$ locality quotas.
- High domain cache-hit-ratio per PoP preferred$\rightarrow$ placement sparsity preferred!
    - also, replication uses more space.
- Stability concerns: few changes.

Model Overview



- j: demand-area pair.
- $X = \{x_{ji}\}$: placement/allocation matrix.
- Demand time series D_j are known in advance.
- θ_i: minimum agreed bandwidth cost to be paid for PoP i.
- γ_j: minimum demand fraction served from local PoPs.
- **Forbidden allocations** Matrix $C = (c_{ji})$, with $c_{ji} \in \{0, 1\}$.
- **Hit ratio**: Penalty function $f(X) \geq 0$ favors sparse allocation matrices X.

Mathematical Model

Find allocation matrix X that maximizes

$$\max_X r = \frac{Q_{95}(\sum_i (X^T D)_i)}{\sum_i \max\left(Q_{95}(X^T D)_i, \theta_i\right)} - f(X) \qquad (1)$$

subject to:

- **Bandwidth constraint**: $\sum_{j \in \mathcal{N}_G(i)} \mathbf{D}_j x_{ji} \leq b_i \quad \forall i \in [1, I], \quad \forall t$
- **Demand conservation**: $\sum_{i \in \mathcal{N}_G(j)} x_{ji} = 1 \quad \forall j \in \mathcal{J}$
- **Forbidden allocations**: Binary matrix $C = \{c_{ji}\}, c_{ji} = 0 \Rightarrow x_{ji} = 0$
- **Local ratio**: $\sum_{i \in \mathcal{N}_G(j)} x_{ji} \cdot \ell_{ji} \geq \gamma_j \quad \forall j \in \mathcal{J}$

where:

- θ_i is the minimum agreed bandwidth cost to be paid for PoP i.
- **Hit ratio**: Non-negative penalty $f(X) = \lambda \sum_{i,j} \mathbb{1}_{x_{i,j} > 0}$, favours sparse allocation matrices.
- Demand time series D_j are known in advance.

Bounds and Baselines

universität wien

Upper bound on performance

$$r = \frac{Q_{95}(\sum_i (X^T D)_i)}{\sum_i \max\left(Q_{95}(X^T D)_i, \theta_i\right)} - f(X) \leq \frac{Q_{95}\left(\sum_j d_{jt}\right)}{\sum_i \theta_i}$$

- Not tight.
- Performance also depends on forbidden constraints and time series characteristics.

Single-PoP model

- Objective function compares transmission costs between the distributed solution and a *single-PoP* system.
- Simplified planning: decision free!
- Traditionally competitive solution, yet opportunity cost?

Baiji: Algorithms

- Static problem (offline)
- non-convex objective function with linear constraints.
- Solve directly: **too slow** (SciPy: years).
- **Approach**: Use multiple algorithms and enhance their placements through a genetic algorithm.
- **Tradeoffs:** between computational complexity (speed) and performance.
- Different solution candidates!

- **Greedy Algorithms**
  - Greedy Deficit-Affinity-based Algorithm (GDAA)
  - Max-based GDAA (GDMAA)
  - Greedy Fast with Post Rebalance (GFPR)
- **Uniform Balancing** (UB)
- **Randomized algorithms**
  - Monte Carlo (MC)
  - Randomized GDAA (RGDAA)
- **Genetic Algorithm** (GA)

Greedy Algorithms - GDAA

Define *affinity* between time series $\mathbf{y}$ and $\mathbf{z}$ as

$$\text{affinity}(\mathbf{y}, \mathbf{z}) = Q_{95}(\mathbf{y}) + Q_{95}(\mathbf{z}) - Q_{95}(\mathbf{y} + \mathbf{z})$$

- **Iteratively** allocates (D-A pair) **demands** sorted desc. by $\gamma-$**deficit** or by **p95 traffic**.
- Filter among **allowed** PoPs that can **partially** support the demand **at all times.**
  - local and θ-deficitary PoPs have priority.
  - $\theta-$deficit$= \max(0, \theta_i - Q_{95}(\mathbf{P}_i))$
- Allocate into PoP the **largest affinity** between current PoP traffic $(\mathbf{P}_i)$ and maximum allocable pair demand $(\alpha_{ji}\mathbf{D}_j)$ where $\alpha_{ji} \in [0, 1]$ used to maximize the affinity:

$$\begin{aligned} \arg\max_{i, \alpha_{ji}} \quad & \text{affinity}(\mathbf{P}_i, \alpha_{ji}\mathbf{D}_j) \\ \text{s.t.:} \quad & p_{it} + \alpha_{ji} d_{jt} \leq b_i \quad \forall t \in [1, T] \end{aligned} \tag{2}$$

- Update remaining (D-A pair) demand and repeat.

Greedy Algorithms

- **Randomized GDAA**: As GDAA but processing D-A pairs in **random** order.
- **Max-based GDAA**: As GDAA but allocating to the PoP with the largest remaining bandwidth, regardless of affinity.
- **GFPR**: Allocate D-A pair with largest unallocated peak demand into the PoP with the largest remaining peak bandwidth.

Uniform Balancing - UB

- **Straightforward strategy** to compute X with as few different cells as possible.
- Start by allocating **all pairs** into **all allowable PoPs with equal weight**.
- Iterate through violated constraints and **rebalance X's rows and columns** (weights) until satisfied.
- **Anchor effect**: If there are no constraints, matches centralized performance ($r = 1$).

Genetic Algorithm

- A population of candidate solutions is iteratively evolved toward better solutions.
  - Including the algorithms' solutions and both feasible and unfeasible candidates.
- Each candidate matrix X is one individual in the population, matrix cells are *genes*.
- On each step (generation) the fitness (ϕ) of each individual is evaluated.

$$\phi(X) = \begin{cases} r(X), & \text{if } X \text{ is feasible,} \\ \sum_{k=1}^{K} \min(0, g_k(X)) & \text{otherwise} \end{cases}$$

Where model constraints are expressed as $g_k(X) \geq 0 \, \forall k \in [1, K]$.

Genetic Algorithm

- **Selection**: 20% fittest + 5% top penalized survive.
- **Cross-over** of survivors' genes create new individuals (offspring).
  - Gene exchange based on random interpolation/swapping.
  - Parent sampling with uniform/fitness-proportional probability.
  - Elitst strategy.
- **Mutation** Decreasing additive uniform noise on the offspring's genes.
  - Depends on population diversity to avoid getting stuck.
- Stop after a fixed number of generations or if fitness is not increasing.

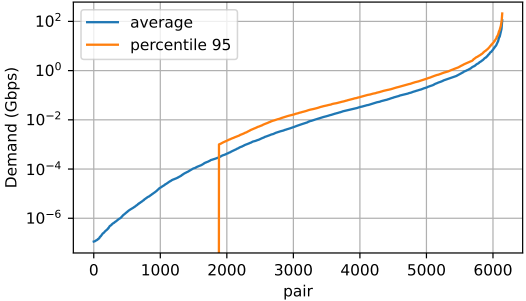
Evaluation

- Performance evaluation on a real-world and two synthetics datasets:
- Individual evaluation of each algorithm.
- Comparison against bounds and single-PoP baseline when feasible.

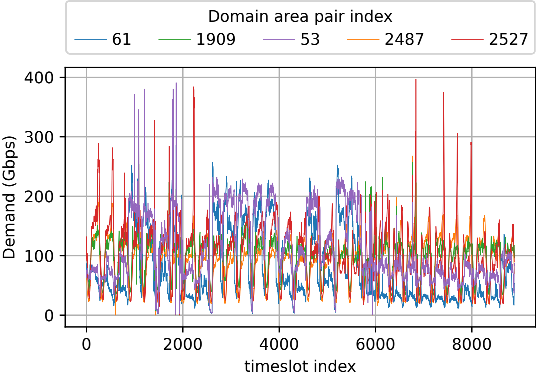
	Real-world	***SYN1***	***SYN2***
Domains	267	450	240
Areas	30	30	30
PoPs	140	120	100
Pairs	6,140	11,475	6,120
Timeslots	8,879	8,000	8,000
Allowed options	145,457	228,761	101,843

Table: **Characteristics of the datasets used for evaluation**.

Real-world Dataset

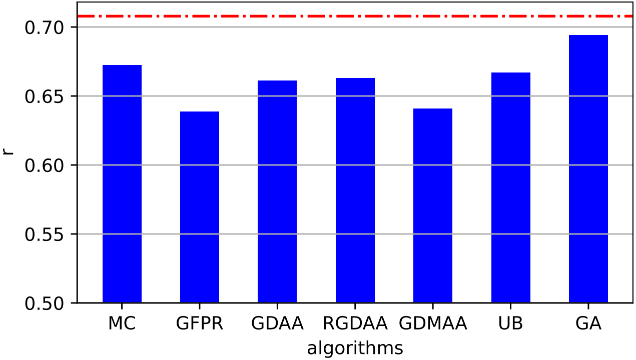


(a) Cactus plot of demand patterns.



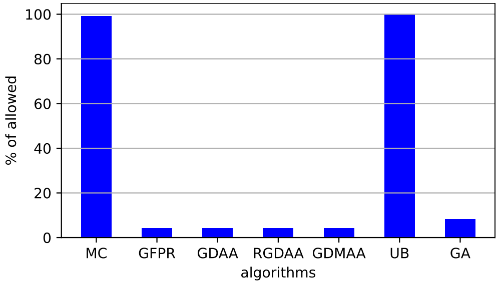
(b) Top demand time series.

Real-World Results - Unpenalized Performance



- Dashed red line: Upper bound.
- Algorithms yield different solutions.
- Best: GA (half the gap).
- Best greedy: GDAA.

Real-World Results - Sparsity



- MC and UB: too dense (**Low CHR**).
- Greedy algorithms: extremely sparse (**High CHR**).
- GA is a compromise between sparsity and performance.

Real-World Results - Elapsed times

Algorithm	Elapsed Time	Comments
UB	0.56 s	
GFPR	0.71 s	
GDMAA	17 s	
GDAA	23 s	
RGDAA	2,315 s	200 sortings
MC	5,467 s	10,000 candidates
GA	14,753 s	2,000 generations, 500 candidates

- **GA** linear on number of generations, super-linear on population size.
- Randomized algorithms are slower as they need many tries for feasible solutions.
- Greedy algorithms are orders of magnitude faster.

Characteristics of synthetic datasets

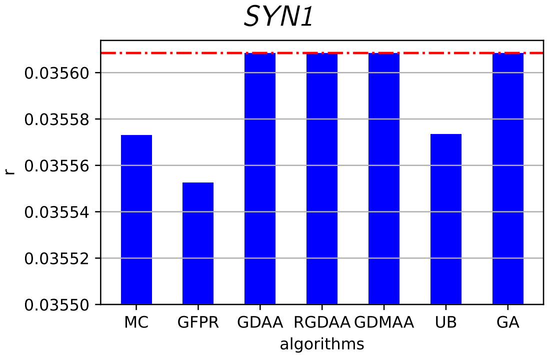
Dataset *SYN1*

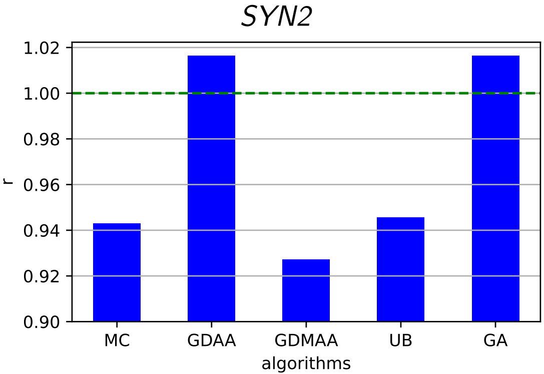
- Demands are combinations of sine patterns.
- Demand values 10x lower, 87% more pairs, 57% more allowed allocations.
- Dominant committed rates (upper bound ≈ 0.036).

Dataset *SYN2*

- Pulse-shaped demand patterns (null demand for 96% of time), random offsets.
- Demand values similar to real-world dataset
- single-PoP cost twice as in real-world dataset.

Performance Results on Synthetic Datasets





- Different solutions match the upper bound.
- All algorithms have the **potential to solve** problem instances.

- GDAA and GA **outperform the single-PoP solution** (dashed green line).

Conclusions

- **Initiated** constrained CDN domain planning study with flexible domain-to-PoP placements under 95th percentile billing model.
- **Formalized** the problem mathematically and derived a performance **upper bound**.
- **Introduced** affinity concept for 95th percentile billing-based optimization.
- Proposed **BAIJI**, an **integrated system** of algorithms for CDN planning.
  - Multiple heuristic methods yield different candidate solutions
    - tradeoffs between performance, speed and sparsity.
  - Especially designed Genetic Algorithm improves results further (larger computational cost/time).
- **Empirical evaluation** exhibits high-quality solutions on real-world and synthetic datasets.