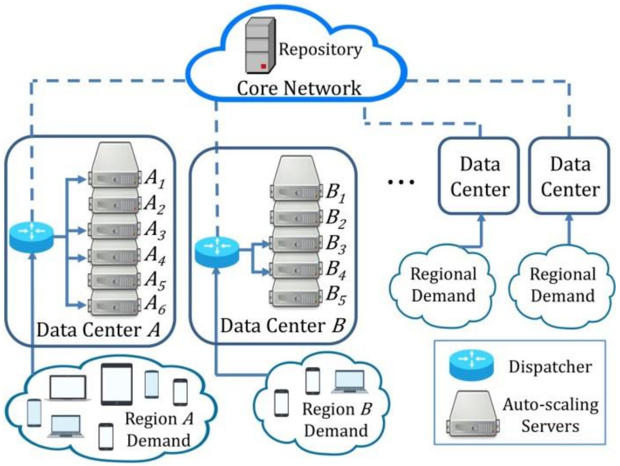
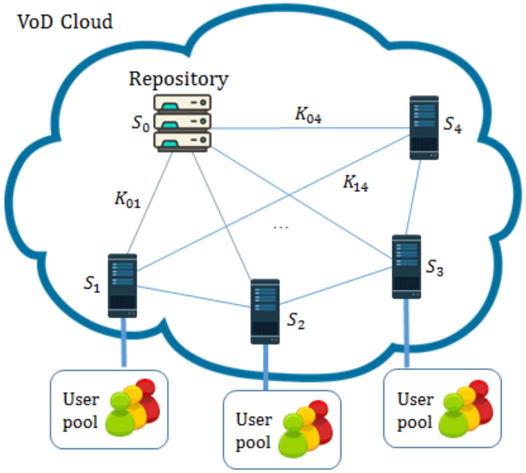
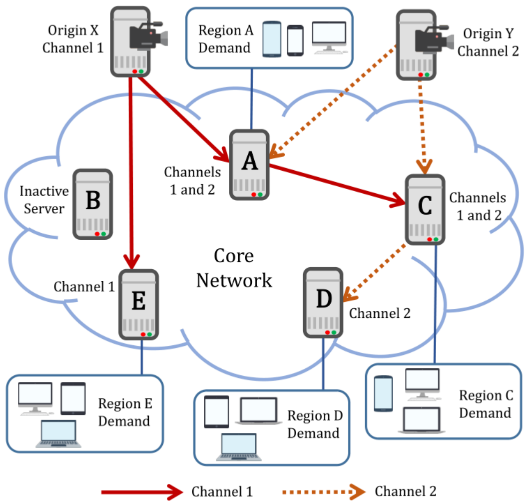






Approximation Algorithms for Auto-Scaling Video Cloud

Ph.D. Thesis Examination

Chang, Zhangyu

Supervised by Prof. Gary Chan

21 December 2023

香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

THE DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
計算機科學及工程學系

Contents

Background: Video-on-Demand and Live Streaming

<table>
<tr><th></th><th>Video-on-Demand (VoD)</th><th>Live Streaming</th></tr>
<tr><td>Features</td><td>• Pre-recorded
• Users can access the video content anytime/anywhere.</td><td>• Created in real-time
• Geo-dispersed users access the video as it is being created.</td></tr>
<tr><td>Examples</td><td>TV shows, movies on Netflix(13.7%), Disney+(4.2%), Amazon Prime, Hulu, iQiyi, Tencent Video, etc.</td><td>Live broadcasting of sports games, concerts, news, online education (lectures), etc.</td></tr>
<tr><td rowspan="2">Resource Required</td><td>• Storage (VoD only)</td><td></td></tr>
<tr><td colspan="2">• Server Processing
• Network Link
No less than the video streaming rate for any request</td></tr>
</table>

Background:
Video Traffic’s Huge Volume and Dynamic Daily Pattern

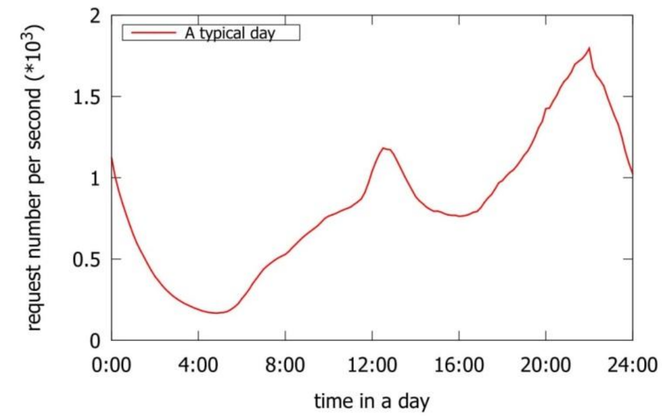
Video Traffic: Huge Volume	
Global Internet Report (Sandvine '23 [1])	
2022 video traffic: As the percentage of total Internet traffic (excluding video calls/conferencing)	66%
2022 video traffic growth rate (compared to Year 2021)	24%

Daily Pattern: Volatile Traffic & Stable Popularity

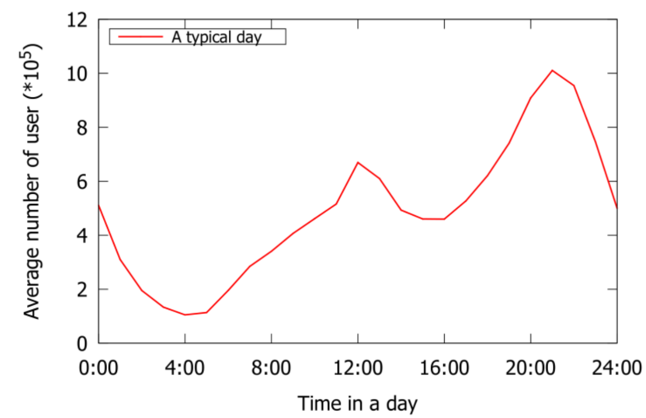
Surveys indicate:

- Traffic varies by more than an order of magnitude over merely hours (Liu et al. TOMM '14 [2])
- Video popularities are stable and predictable over several days (Cha et al. Trans. Netw. '09 [3])

Data source of the plots: Tencent Video



VoD: User requests over a day

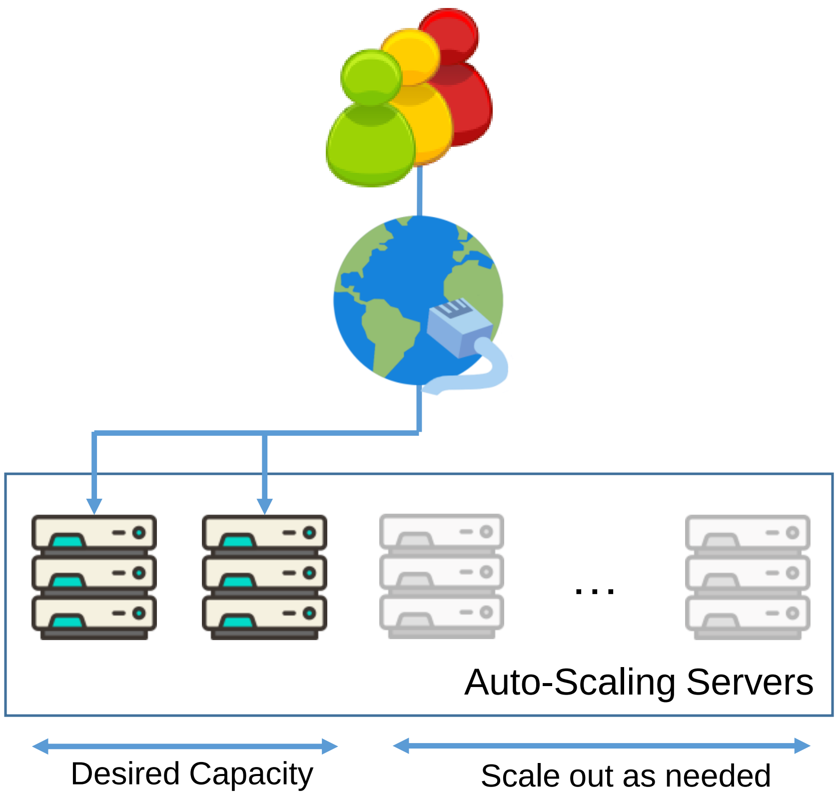


Live streaming: Average user number over a day

Benefit of Auto-Scaling: Allocate Geo-Dispersed Resources on the Fly

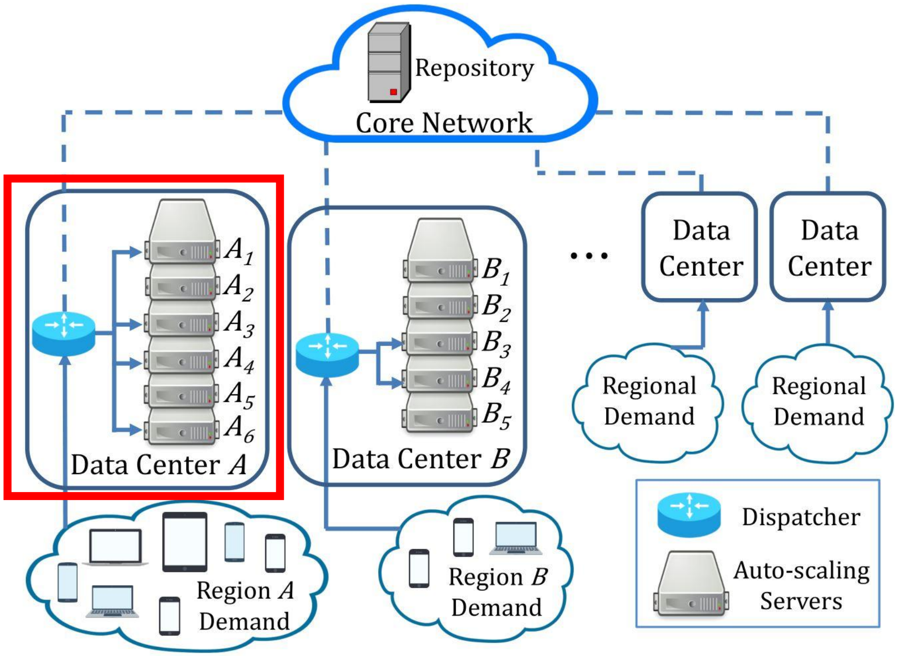
Auto-Scaling Data Center
Rescale system resources elastically: • Deploy geo-dispersed servers *on the fly* to support local audience • Activated or deactivate servers in a timely manner

Traditional Static Provisioning (For Comparison)
• Fixed number of servers • Not cost-effective due to daily traffic pattern • Inevitable overprovisioning to ensure quality of service



An auto-scaling data center

Problem 1: Optimizing an Auto-Scaling VoD Data Center



A video cloud consisting of auto-scaling VoD data centers.

System Settings

- Servers can be activated or deactivated in a short time
- A traffic dispatcher distributes request to an active server with the video

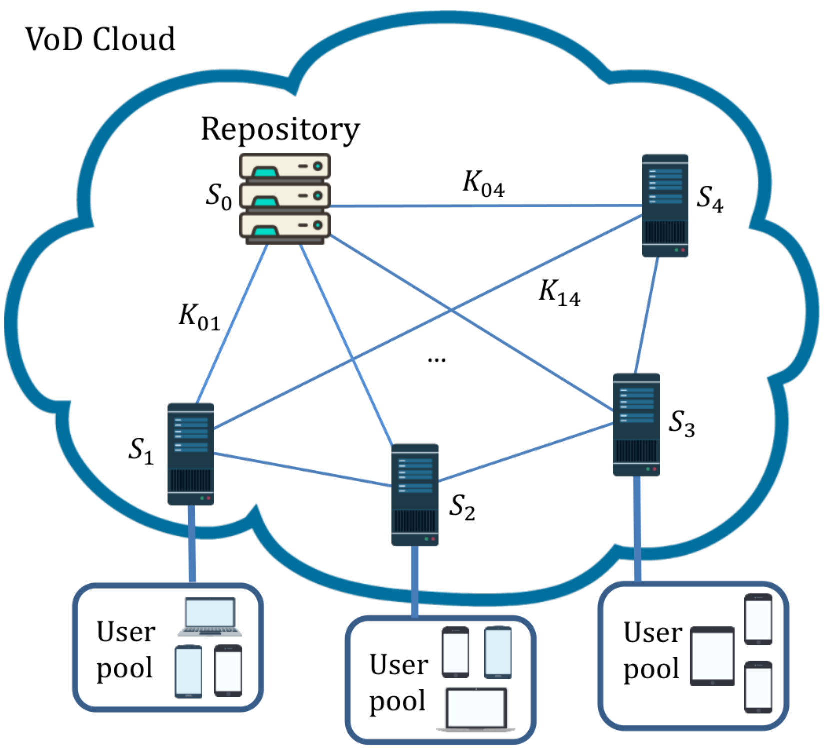
Major Objectives

- Cost-effectiveness:
  Minimize the number of active servers
- Quality of service:
  Allocate enough resource to serve users

Contributions

- Simple but efficient approximation algorithm
- **AVARDO: *A*uto-Scaling *V*ideo *A*llocation and *R*equest *D*istribution *O*ptimization**

Problem 2: Optimizing a Geo-Distributed VoD Cloud



A distributed and cooperative cloud architecture for VoD service

System Settings

- Repository: complete video replication
- Local server: partial replication to save storage
- Collaboratively serve the users to reduce cost

Major Objectives

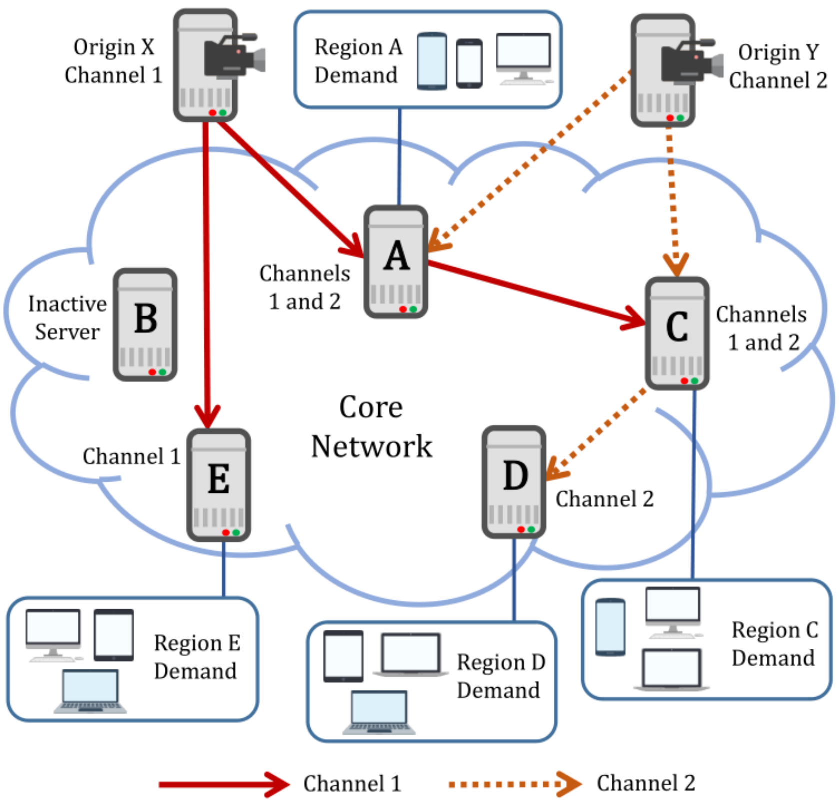
Decide where to store and access the videos

- Satisfy quality-of-service (QoS) constraints
- Minimize total deployment cost

Contributions

- Approximation algorithm based on linear programming
- **RAVO: *R*esource *A*llocation and *V*ideo Management *O*ptimization**

Problem 3: Optimizing an Auto-Scaling Live Cloud



A multi-origin multi-channel live streaming cloud.

System Settings

- Geo-dispersed auto-scaling servers
- Push each channel stream as a tree
- Cover servers demanding the channel

Major Objectives

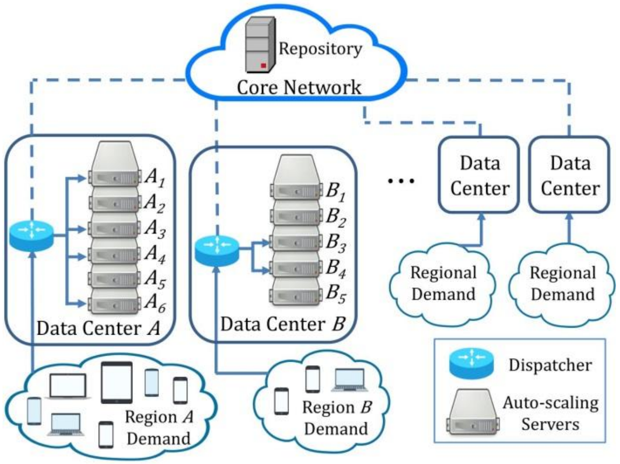
- Construct overlay trees to deliver channels
- Bi-Criteria objective:
  - Minimize deployment cost
  - Minimize origin-to-end delays

Contributions

- Bi-Criteria approximation algorithm based on linear programming
- **COCOS: Cost-Optimized Multi-Origin Multi-Channel Overlay Streaming**

Basic Concepts: Approximation Algorithms

- NP-hard problem: currently no efficient solution (Spoiler: all the 3 problems are NP-hard)
- Approximation algorithms: tradeoff between complexity of algorithms and optimality of solutions
- $\text{Super optimum} \leq \text{Exact optimum} \leq \text{Experimental result} \leq \text{Approximation solution}$
- $\text{Approximation ratio} = \frac{\text{Approximation solution}}{\text{Super optimum}} > 1$
- $\text{Optimality gap} = \text{Approximation ratio} - 1 > 0$

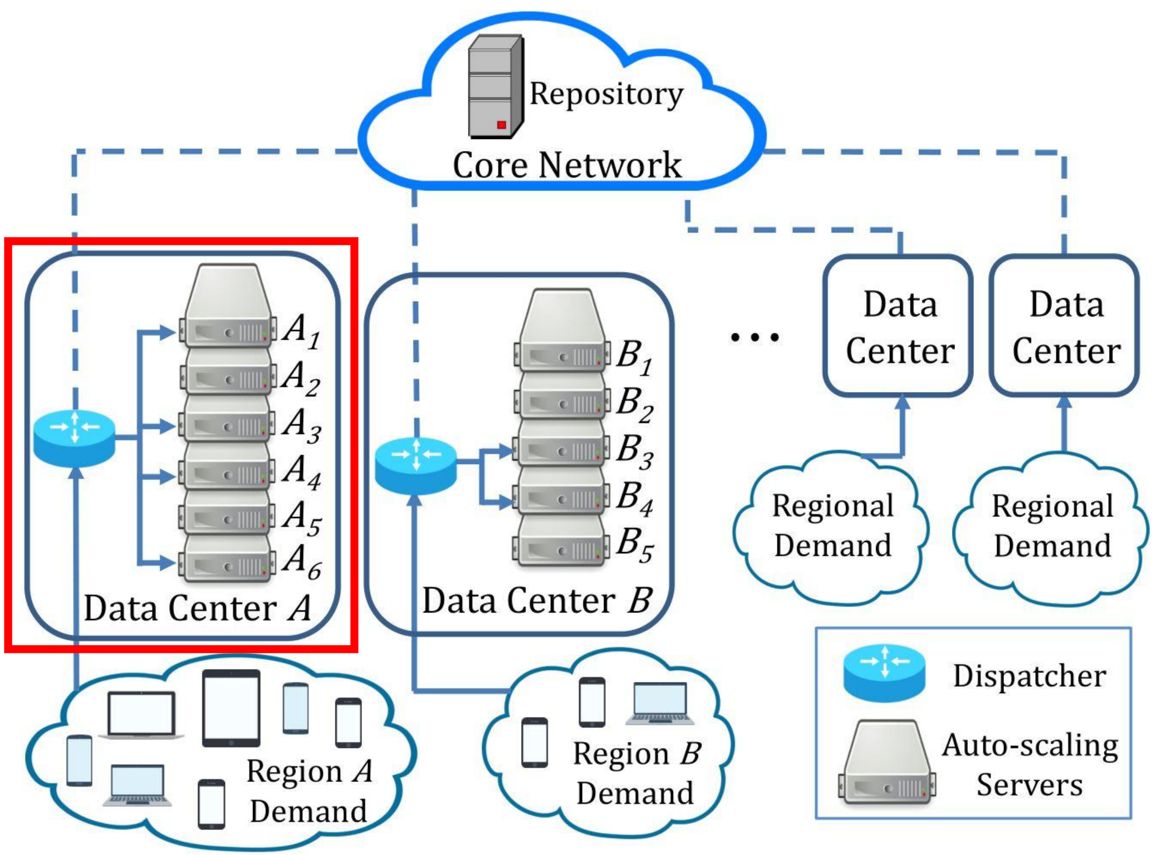


Contents

Publication:

Z. Chang and S.-H. Chan, "An Approximation Algorithm to Maximize User Capacity for an Auto-scaling VoD System," *IEEE Transactions on Multimedia*, Vol. 23, pp. 3714-3725, October 2021.

Background:
A Typical Auto-scaling VoD Cloud



A video cloud consisting of
auto-scaling VoD data centers.

Auto-scaling Server

Auto-scaling
- Server can be activated or deactivated in a short time

Homogeneous
- Server has same storage and streaming capacity

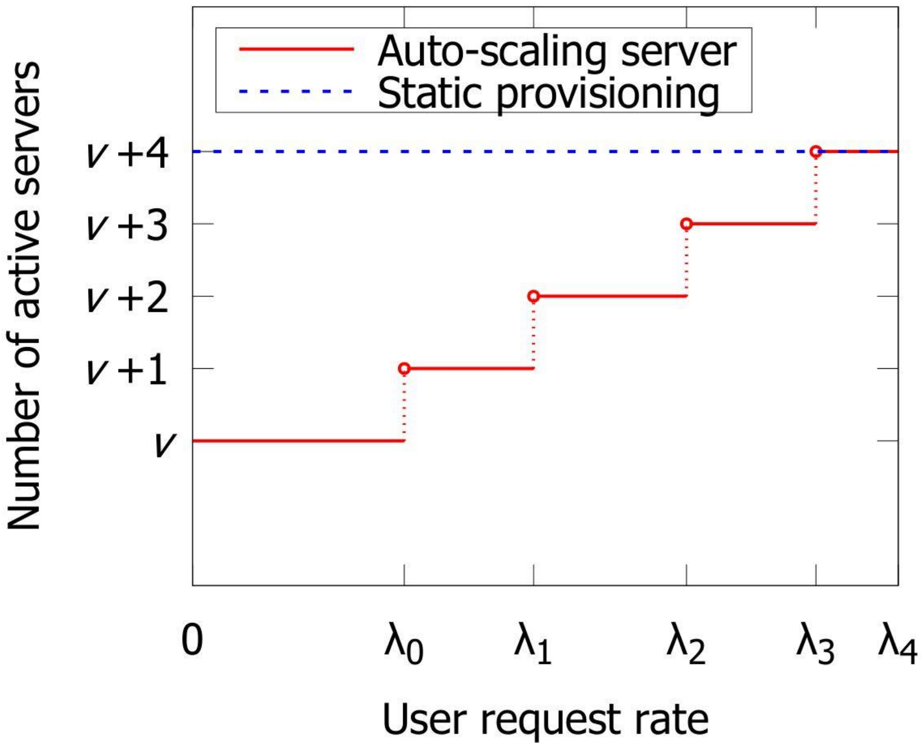
Traffic Dispatcher

- Distribute request to an active server with the video
- Otherwise to core network

Video Block

Only for management purpose (cf. DASH segments)
- Blocks have the same size
- Partition large video into blocks

Objective: Maximizing the User Request Rate Threshold λ_i



- **Auto-scaling level** i (i=0, 1, 2, ...) based on user request rate
- At level 0, we activate ν servers (minimum number) with full replicas.
- At level i, we activate $\nu + i$ servers to support at most λ_i user request rate.
- We want to maximize λ_i for every level i.

Optimization Parameters

	Block Allocation (BA)	Server Selection (SS)	Request Dispatching (RD)
Question	Which blocks should be allocated in each server?	Which servers should be activated for current traffic?	Which server for a request?
Constraints	• Number of blocks a server can store	In active servers: • At least one replica for each block • Popular blocks in enough servers	• Avoid overload of any active servers
Timescale	In day or week	In hour	In second
Dependency	Pre-allocated Videos for SS and RD	Based on BA	Based on BA and SS

- Major challenge: optimize **one** BA to fit **multiple** SS and RD

Related Work

	Related Work	AVARDO
Cloud-based VoD resource provisioning	• Data center as a black box • Yet to consider features inside the data center (Yang et al. Infosys '14 [4])	• Investigating from a more detailed point of view
Content replication in traditional and cloud-based VoD	• Static provisioning (Yang et al. IEEE TMM' 18 [5]) • Assume no change of storage and only optimize processing capacity (Bourtsoulatze et al. IEEE TMM '18 [6], Sheganaku et al. Future Gener. Comput. Syst. '23 [7])	• Optimize for every auto-scaling levels
Cloud resources auto-scaling mechanism	• Predict the user demand (Zhao et al. Multimedia Tools Appl '19 [8], Luo et al. SoCC '22 [9]) • A video is served by only one server (Du et al. IEEE TMM '16 [10])	• Considers BA, SS, and RD

Problem Formulation: Major Symbol Used in AVARDO

u	The streaming capacity of a server (bits/s)	p^m	Access probability of video block m
c	The storage capacity of a server (bits)	L^m	Average holding time of video block m (in seconds)
f	The file size of block (bits)	b^m	Video streaming rate of video block m (bits/s)
V	The set of all standby servers in data center	$R^m(\lambda)$	Traffic of block m (bits/s) at request rate λ
$\boldsymbol{V_i}$	The set of active servers at auto-scaling level i **(SS)**	$\boldsymbol{I_v^m}$	Binary variable indicating server v stores block m **(BA)**
M	The set of all blocks	$\boldsymbol{r_v^m(i)}$	Probability of streaming a request of block m from server v at auto-scaling level i **(RD)**
M_v	The set of video blocks stored in server v		
λ	Total block request rate (requests per second)	μ	Server utilization limit to ensure quality-of-service

Comprehensive Model as an NP-Hard Problem: Auto-scaling Video Allocation and Request Dispatching

Objective $\max(\lambda_0, \lambda_1, \ldots \lambda_n)$ → Maximize request rate threshold

Subject to

$$R^m(\lambda) = \lambda p^m L^m b^m, \forall m \in M$$ → Traffic of video block m (bits/s) at request rate λ

Storage

$$\sum_{m \in M(v)} I_v^m f \leq c, \forall v \in V$$ → Storage limit of each server (BA)

Streaming

$$r_v^m(i) \leq I_v^m, \forall v \in V_i, m \in M$$ → Server only serve the video it has (SS)

$$\sum_{v \in V_i} r_v^m(i) \geq 1, \forall m \in M$$ → User request shall be served (RD)

QoS

$$\sum_{m \in M} r_v^m(i) R^m(\lambda_i) \leq \mu u, \forall v \in V_i$$ → Utilization limit of each server (RD)

Multi-Objective Mixed Integer Programming

The NP-complete Partition Problem is reducible to our problem. It is **NP-hard**!

AVARDO: Approximation Algorithm for an Auto-scaling Video-on-Demand System

Simplification: Block Replication and Clustering

- Replicate videos according to their popularities
- Group video blocks into v^2 clusters (mega videos)
- Each cluster has the same *file size* and similar *user traffic*

Solution: From Cluster to Video Blocks

The system satisfies the following constraints:

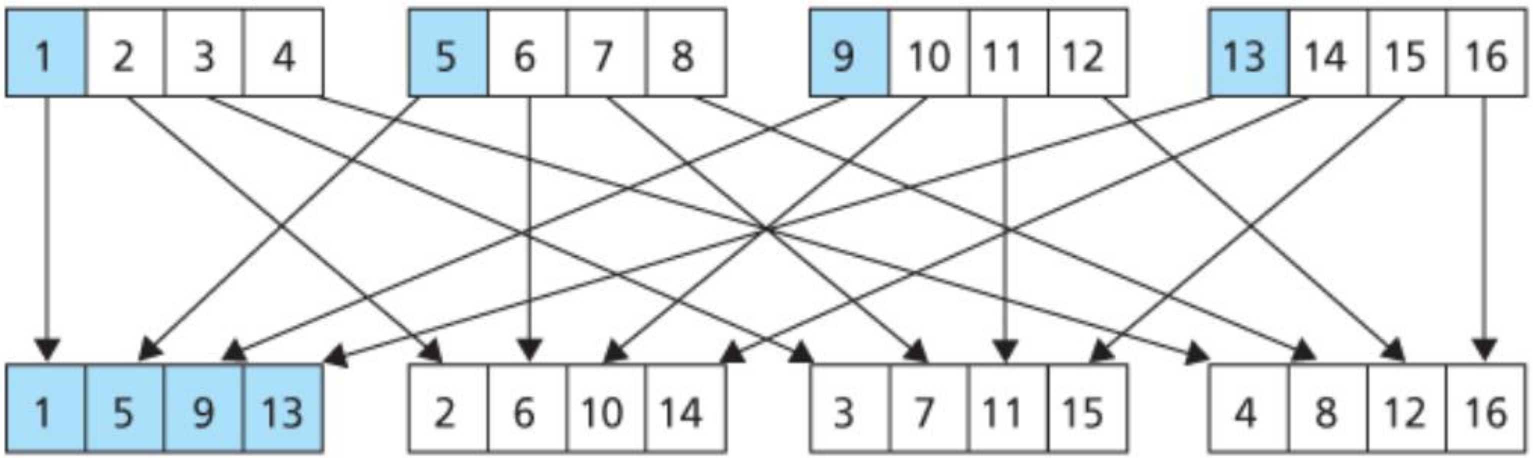
1. At auto-scaling level 0, the active servers has all clusters.
2. When activate a new server, we can evenly offload traffic from the existing active servers (interleaving).

RD can be formulated as a linear system and has closed-form solution.

AVARDO has a stack-based server selection scheme

- Push (activate) or pop (deactivate) only one server when auto-scaling level goes up or down
- Approximation ratio: $1 + v^2/|M|$
- Optimality gap under practical setting: less than 1%
- Complexity: $O(|M| \log |M|)$

Example of Interleaving



Experimental Setup

Parameter	Baseline
Number of blocks $\|M\|$	ca. 3×10^6
Video block size	100MB
Maximum block request rate λ (requests/s)	2,000
Number of blocks in a server	6×10^5
Server streaming capacity u (Gbps)	25
Server utilization limit μ	0.9

- Real-world data trace: from a leading video website (Tencent Video) in China over 2 weeks with 1.5 million videos in total.
- Synthetic date with Zipf's distribution: $p^m \propto 1/m^z$ for mth popular video

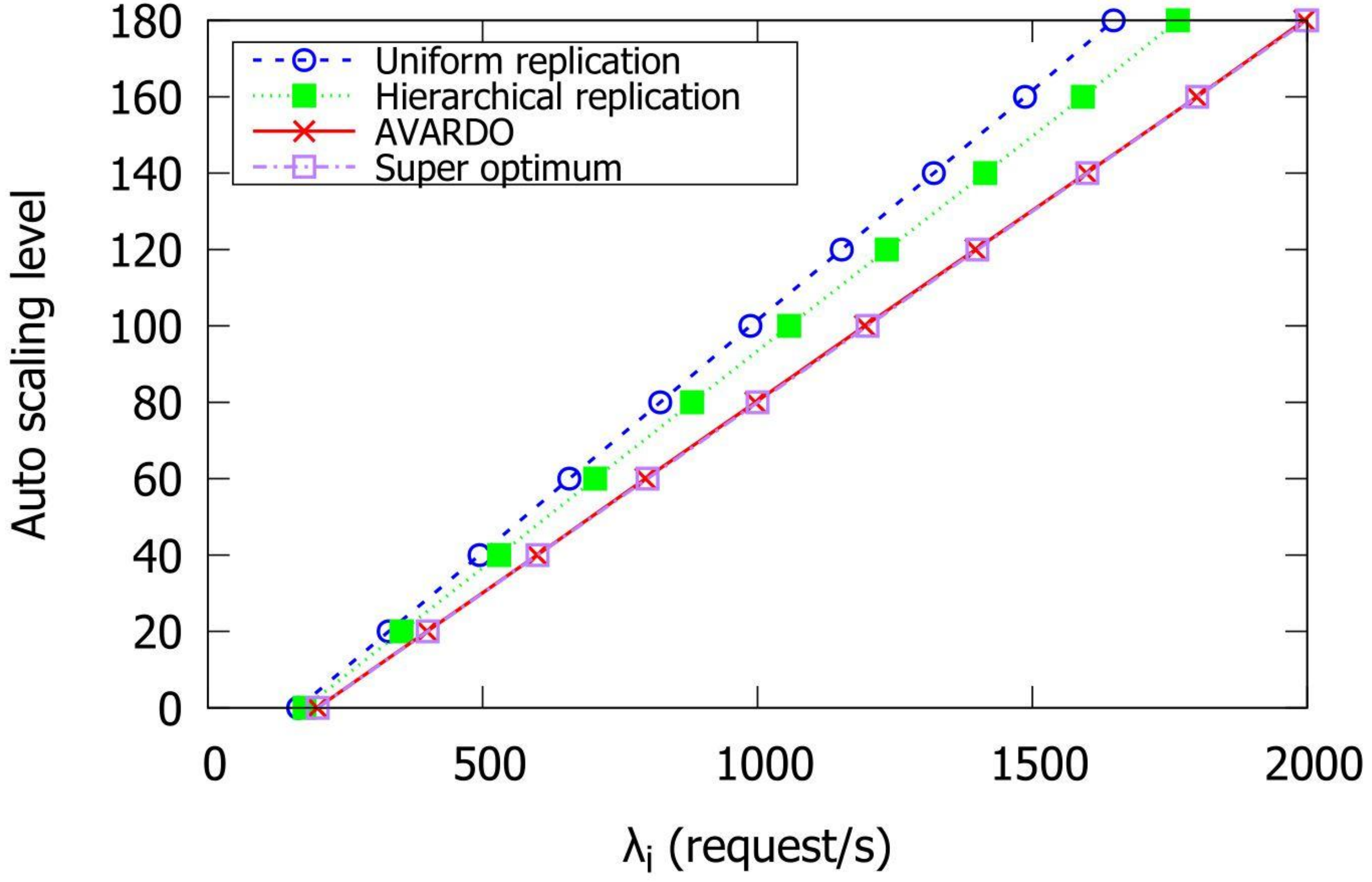
Performance Metrics

- Request rate threshold λ_n
- Optimality gap
- Number of active servers
- Fairness of active server utilization

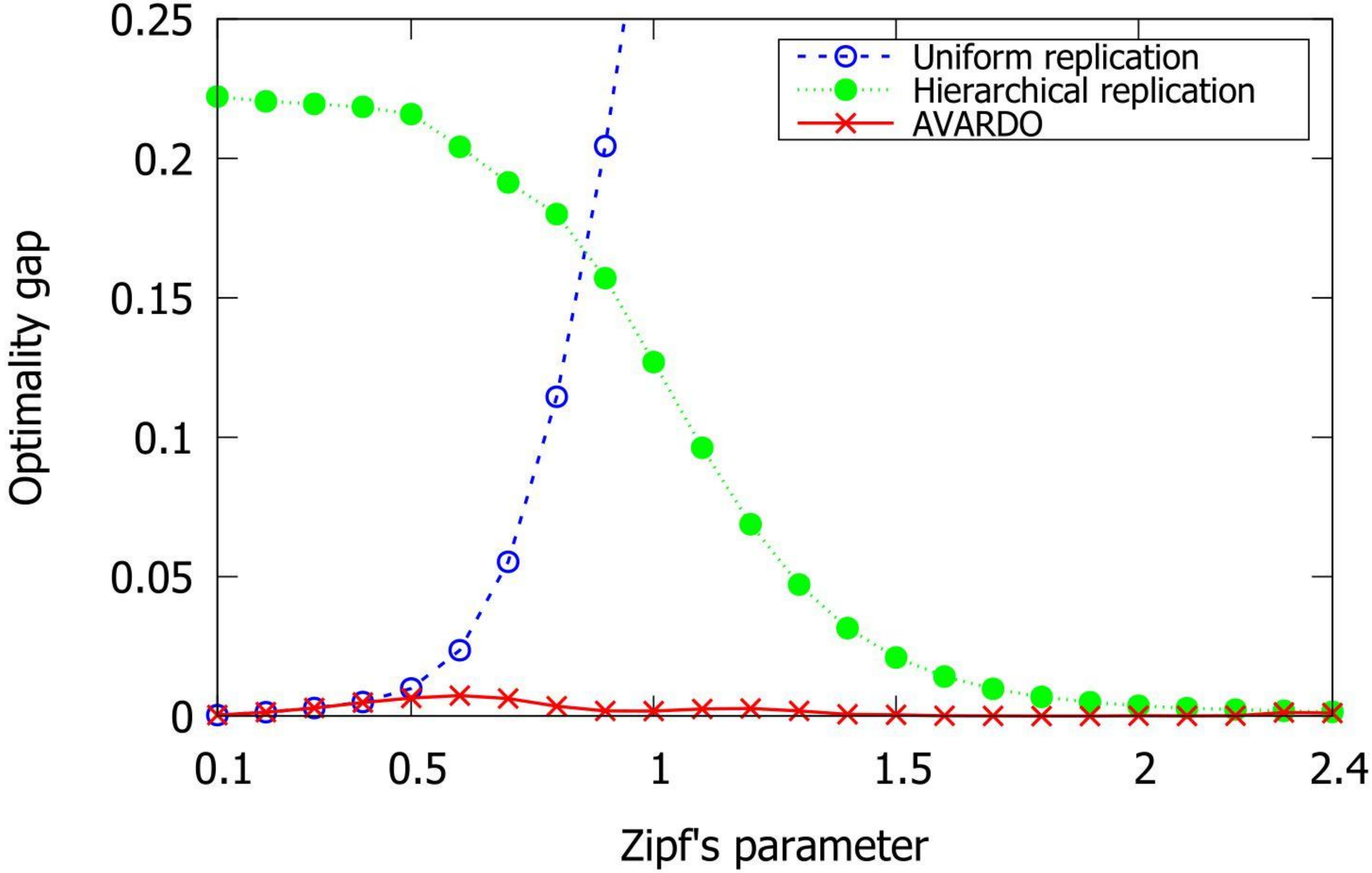
Comparison Schemes

- Uniform replication [10]
- Hierarchical popularity replication [11]

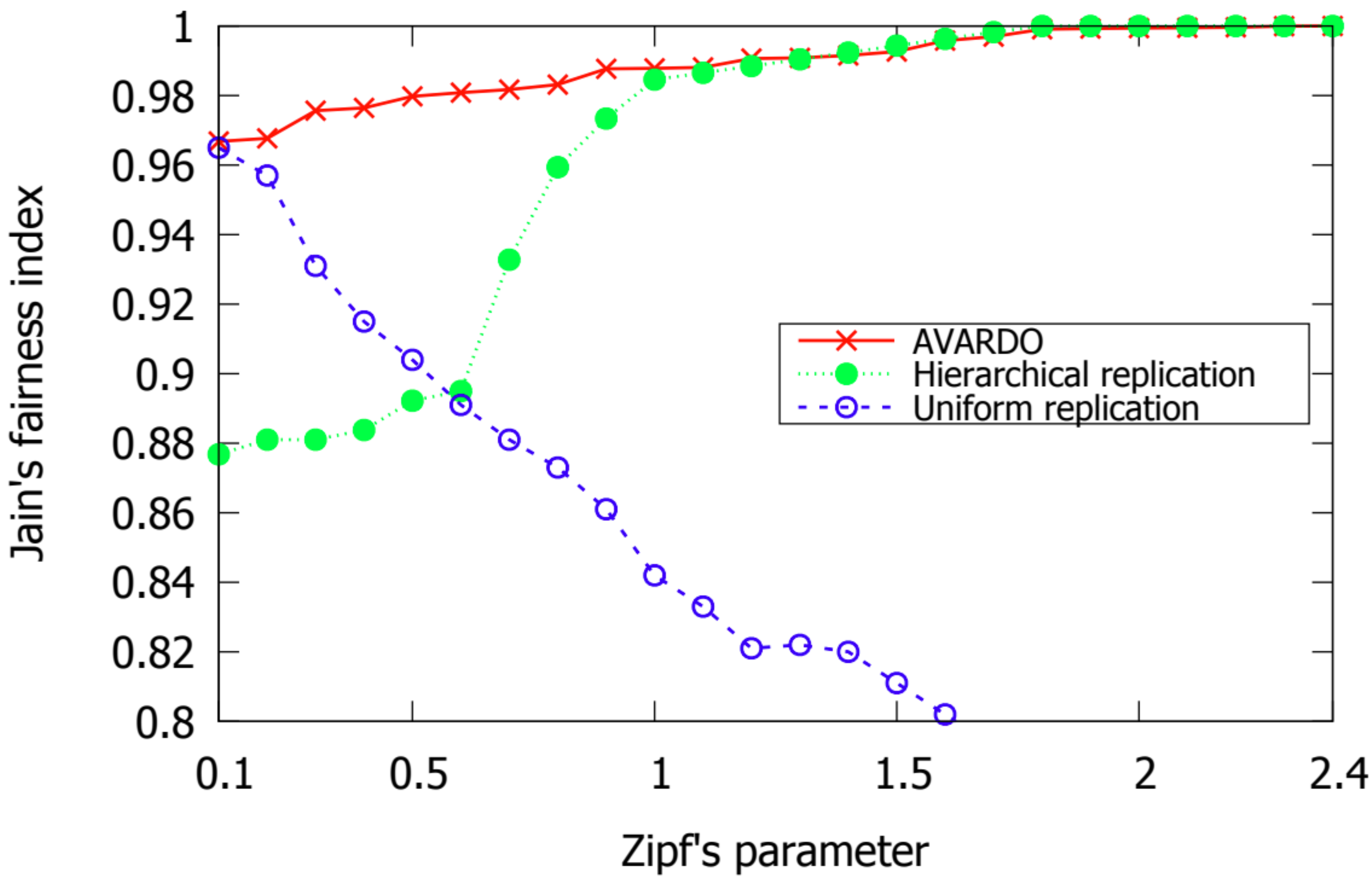
Near Optimal Performance (Real-World Data)

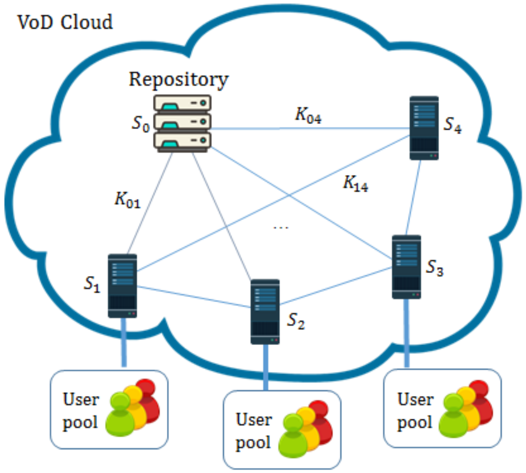


Outperform State-of-the-art Schemes (Synthetic data)



Better Load Balancing (Synthetic data)



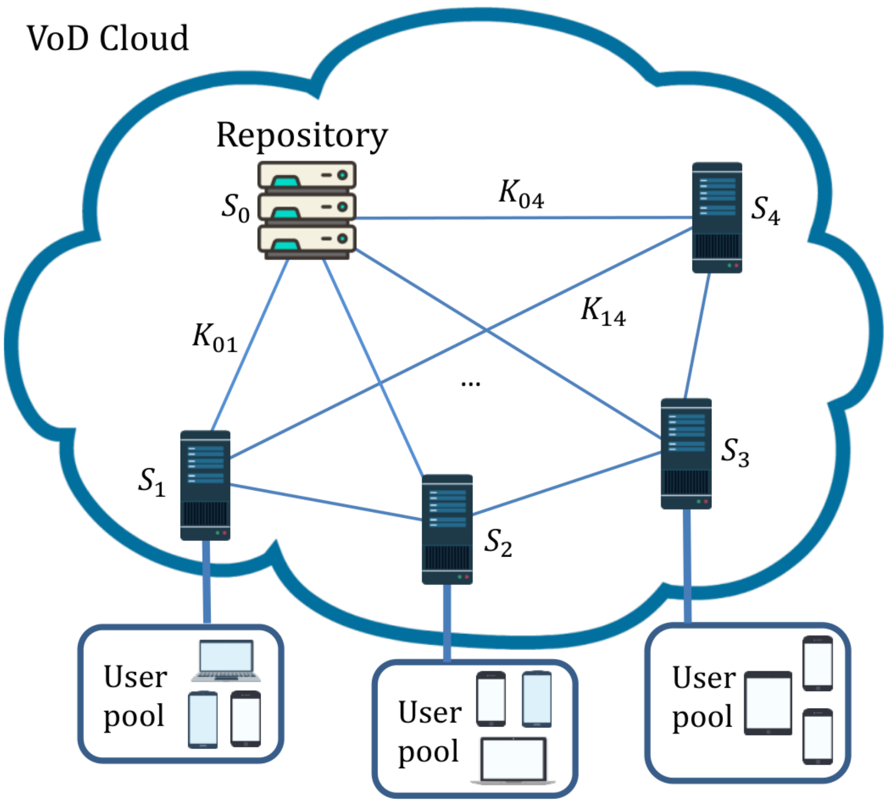


Contents

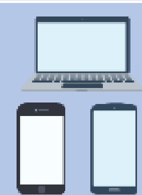
Publication:

Z. Chang and S.-H. Chan, "Video Management and Resource Allocation for a Large-scale VoD Cloud," *ACM Transactions on Multimedia Computing, Communication and Applications (TOMM) Special Issue on Multimedia Big Data: Networking*, Vol. 12, No. 5s, pp. 72:1-72:21, Sept 2016.

Background:
A Geo-Distributed Auto-Scaling VoD Cloud



A distributed and cooperative cloud architecture for VoD service

	**Repository**: Complete video replication
	**Local cloud server**: Auto-scaling data centers to serve their user pool
	**User pool:** Heterogeneous video popularities

Geographic Heterogeneity of Video Popularities

- Local servers have partial video replication to save storage
- Reduce network load through collaboration among local servers

Objectives and Tradeoff

Store Video Locally

+

- Less delay
- Less link & processing cost
- Better collaboration

-

- More storage cost

Access Video Remotely

+

- Less storage cost

-

- More delay
- More link & processing cost
- Less collaboration

Objectives:

- Satisfy the quality-of-service constraints
- Minimize total deployment cost

Optimization Parameters

Video Management (VM)

Storage (content replication)

- What video to store at each server?
- Planned on a longer time scale (days)

Retrieval (server selection)

- Which servers to stream the missing video from?
- Decided when a request comes

Resource Allocation (RA)

Server Resource

- Total **storage** and **processing capacity** at a server

Link Resource

- **Link capacity** reserved between pairs of servers

Related Work

Fundamental difference:
RAVO is an **approximation** algorithm

	Related Work	RAVO
Traditional resource allocation	• Based on heuristic approach • The optimality gap is not clear (Adhikari et al. Infocom '12 [12])	• Discretized from LP solution • Proven approximation ratio
Content Storage and Retrieval for VoD	• Need resource allocation result • Static resource provisioning (Applegate et al. Co-NEXT '10 [13], Minowa et al. NBiS '22 [14])	• One-step algorithm for both resource allocation and content management • Auto-scaling feature
Current resource allocation for cloud service	• Assume full replication • Only consider link capacity allocation (Lin et al. Infocom '11 [15], Liu et al. Mathematics '23 [16])	• Flexible replication to reduce the storage cost • Servers help each other to fully utilize the resource

Problem Formulation: Major Symbols Used in RAVO

S	The set of servers (central and proxy servers)	Γ_{mn}	Average transmission rate from server m to n (bits/s)
V	The set of videos	U_m	Total upload rate of server m (bits/s)
$L^{(v)}$	Length of video v (seconds)	$\boldsymbol{K_{mn}}$	Link capacity from server m to n (bits/s) **(RA)**
$P_m^{(v)}$	Access probability of video v at server m	$\boldsymbol{\Lambda_m}$	Processing capacity of server m for streaming (bits/s) **(RA)**
$\boldsymbol{I_m^{(v)}}$	Boolean variable indicating whether server m stores video v **(VM)**	C_{mn}^{N}	Link cost due to directed traffic from server m to n
$\boldsymbol{H_m}$	Storage capacity of server m (bits) **(RA)**	C_m^{S}	Cost of server m
$\boldsymbol{R_{mn}^{(v)}}$	Probability of streaming video v from server m to n **(VM)**	D_{mn}^{N}	Delay due to directed traffic from server m to n
μ_m	Request rate at server m (requests/second)	D_m^{S}	Delay due to upload streaming of server m

Comprehensive Model as an NP-Hard Problem:
Joint Optimization on Video Management and Resource Allocation

Server cost Link cost

minimize $\sum_{m \epsilon S} \mathbb{C}_m^{\mathrm{S}}(H_m, \Lambda_m, U_m) + \sum_{m,n \epsilon S} \mathbb{C}_{mn}^{\mathrm{N}}(\Gamma_{mn}, K_{mn})$ → System deployment cost

Storage Processing Capacity

Access bandwidth (consumed)

Subject to

Storage $I_m^{(v)} \in \{0,1\}, \forall m \in S, v \in V$ → Store a video as a whole

Retrieval $0 \le R_{mn}^{(v)} \le I_m^{(v)}, \forall m, n \in S, v \in V$ → Only retrieve video from a server with it

$\sum_{v \in V} I_m^{(v)} L^{(v)} \gamma^{(v)} \le H_m, \quad \forall m \in S$ → Server storage constraint

$\sum_{m \in S} R_{mn}^{(v)} = 1, \quad \forall n \in S, v \in V$ → User request must be served

$\Gamma_{mn} = \sum_{v \in V} p_n^{(v)} \varepsilon_n^{(v)} \mu_n R_{mn}^{(v)} L^{(v)} \gamma^{(v)}, \forall m, n \in S$ → Remote traffic

QoS $\mathbb{D}_{mn}^{\mathrm{N}}(\Gamma_{mn}, K_{mn}) + \mathbb{D}_m^{\mathrm{S}}(U_m, \Lambda_m) \le \bar{D}, \forall m, n \in S$ → Delay

Mixed Integer Programming

The NP-complete Dominating Set Problem is reducible to our problem. It is **NP-Hard**!

RAVO: Relaxing the Joint Formulation as a Linear Program and Quantization of the Solution

Step 1: Linear Program

Formulation Relaxation

- Continuous storage decision $\hat{I}_m^{(v)}$ $(0 \leq \hat{I}_m^{(v)} \leq 1)$
- Piecewise linear function to approximate delay and cost
- *Efficient algorithm* for solving linear programming

Solve Relaxed LP for Super-optimum

- Video storage: $\hat{I}_m^{(v)}$
- Video retrieval: $\hat{R}_{mn}^{(v)}$

Step 2: Quantization

Video Management

- *Randomized round* $\hat{I}_m^{(v)}$ to get $I_m^{(v)}$
- Request from the *repository* if no other local server can help
- Otherwise we obtain $R_{mn}^{(v)}$ proportional to $\hat{R}_{mn}^{(v)}$ for server having the video

Resource Allocation

- Calculate the parameters according to the formulation
- Expected approximation ratio (1+1/e) ≈ 1.37

Example of Quantization

Server	1	2	3	4
I from LP	0.2	0.8	0.9	1
I after rounding	0	1	1	1
R from LP	0.2	0.1	0.3	0.4
R *after* Quantization	0	0.125	0.375	0.5

Reducing the Algorithmic Time Complexity: Spectral Clustering for Video Group

- Time complexity: $O(|S|^6|V|^3)$, but $|V|$ could be large
- **Cluster** the videos with similar popularities into a mega video
- Use **spectral clustering method** to solve multi-dimensional *K-means*
- After solving the linear program,
    1. Evenly place the video from the same group and
    2. Let $\hat{R}_{mn}^{(v)} = \hat{R}_{mn}^{(g_i)}, \forall\, v \epsilon g_i$
    3. Then go for parameter quantization
- Reduce the complexity by $O(|V|^2)$

Experimental Setup

Video Popularity

Real data

- From a leading IPTV provider (China Telecom) over 2 weeks

Synthetic data

- *Zipf's* distribution: $P_m^{(v)} \propto 1/v^z$
- Geographic heterogeneity

Cost Functions

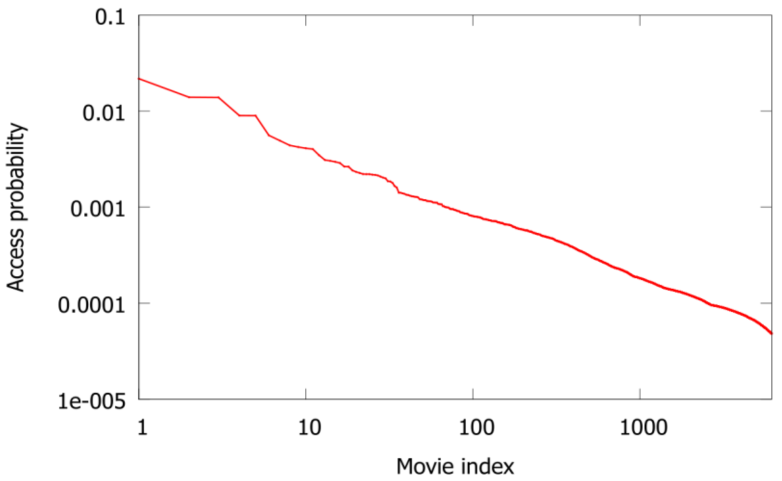
- *Proportional* to resource
- Server cost: $C_m^S = \sigma_m H_m + c_m \Lambda_m$
- Link cost: $C_{mn}^N = c_{mn} K_{mn}$

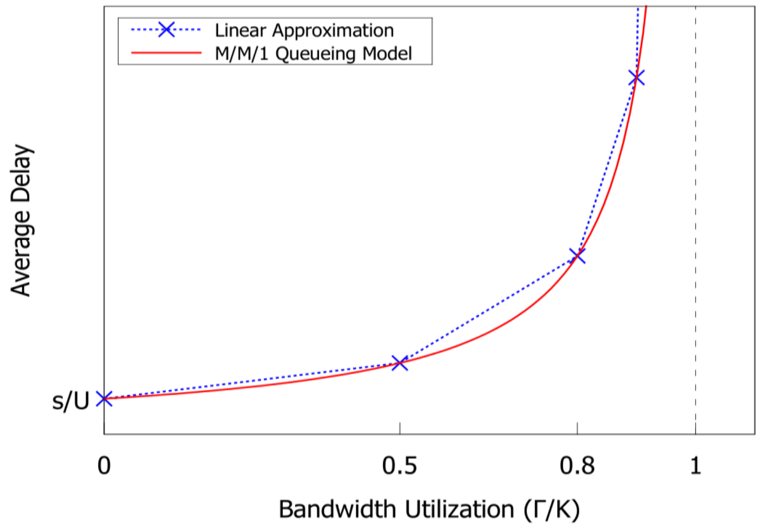
Delay Function

- *M/M/1* queueing model
- *Piece-wise linear* approximation

Comparison Schemes

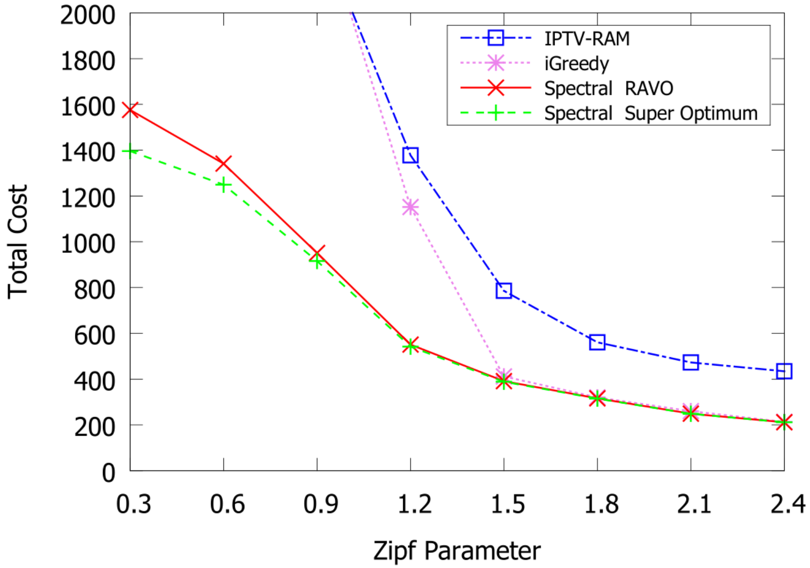
- iGreedy [17]
- IPTV-RAM [18]



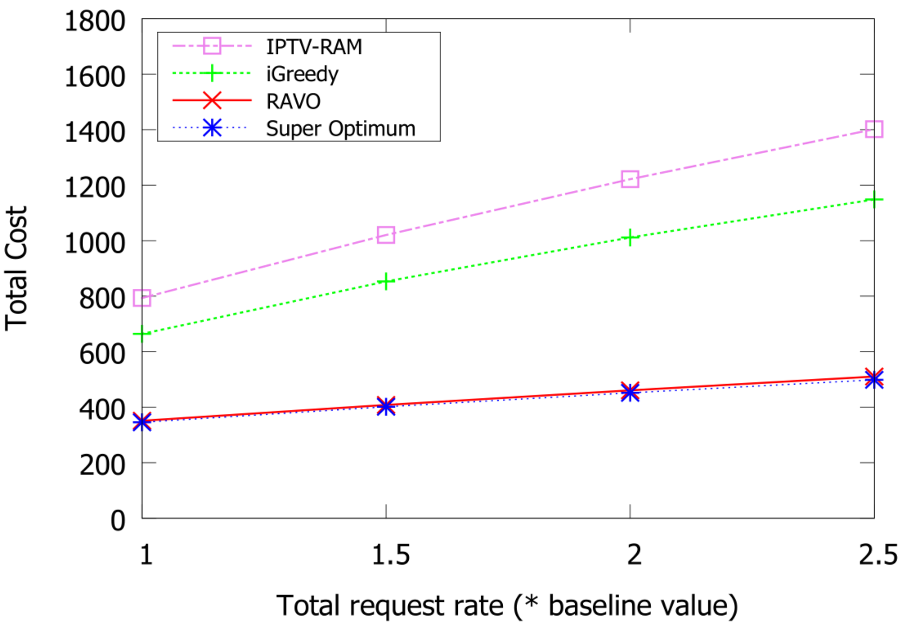


Near-Optimal Performance

Synthetic Data



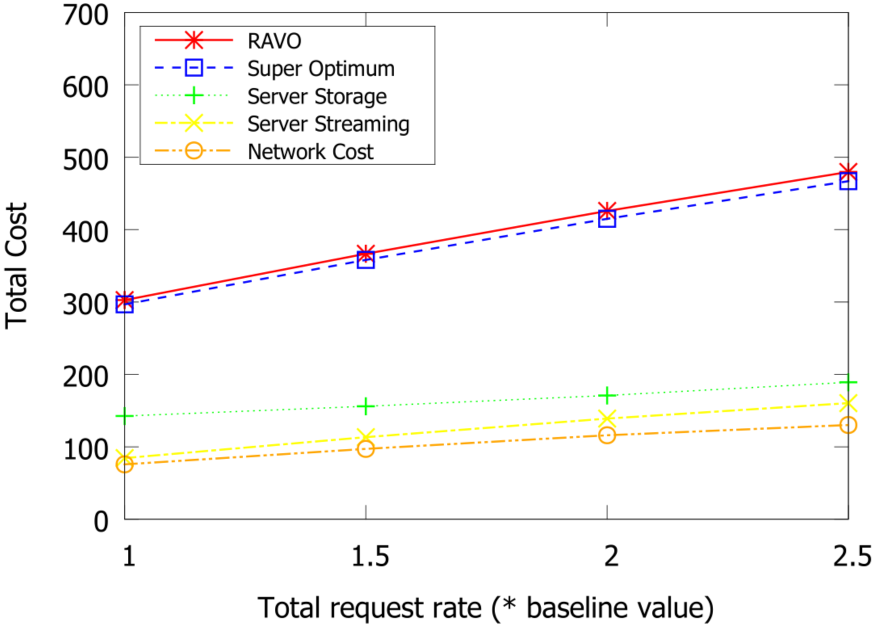
Real Data



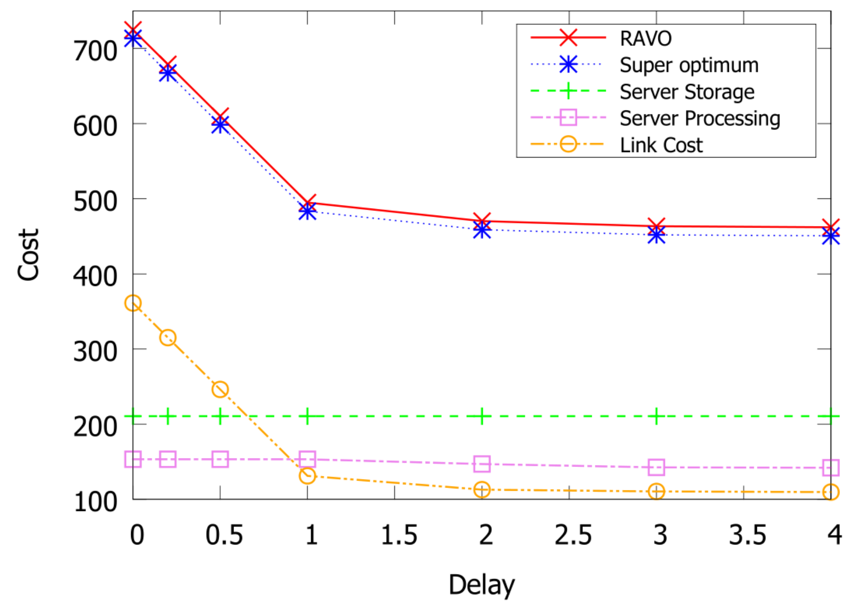
- RAVO outperforms the comparison schemes with large margin
- Close to the super optimum

Effective Use of Resources

Deployment cost given different request rate

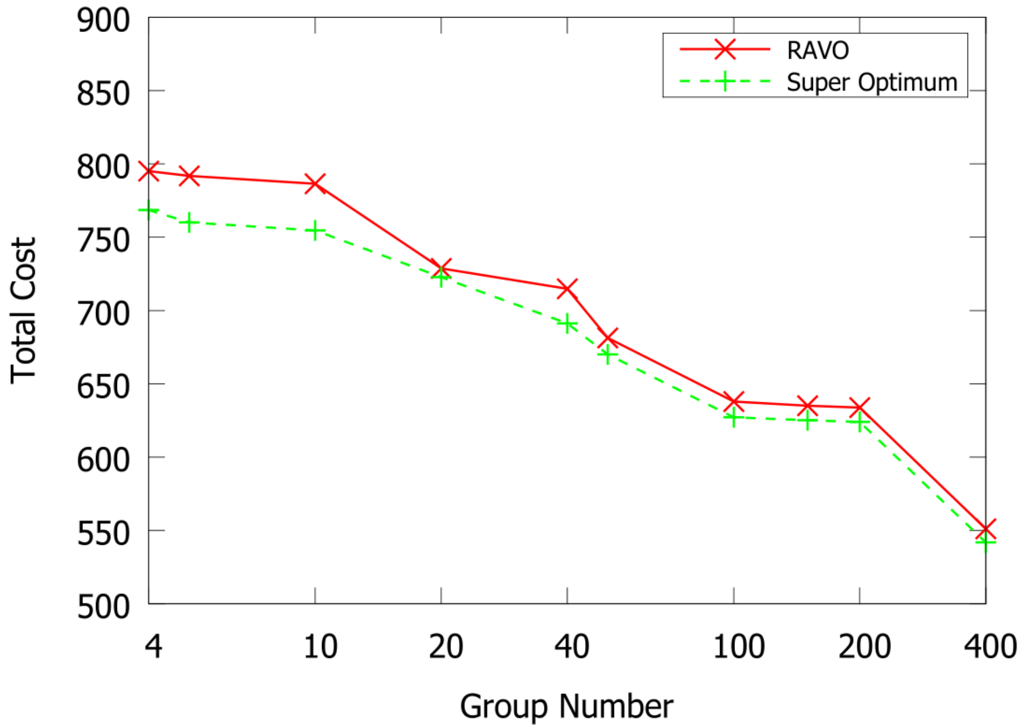


Cost versus Delay Requirement

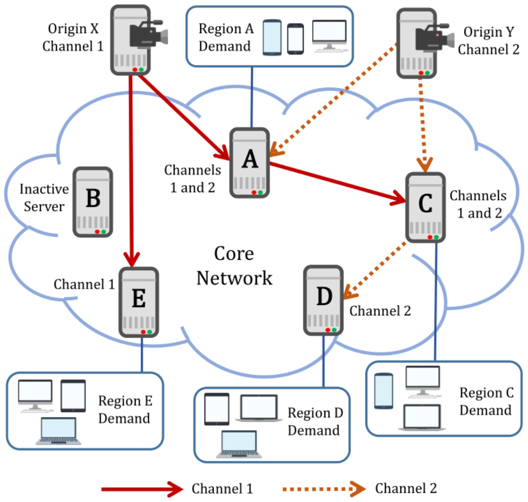


- The change of cost is distributed into all the components

Effective Clustering Method



- The cost decreases with more groups (and more computation time)

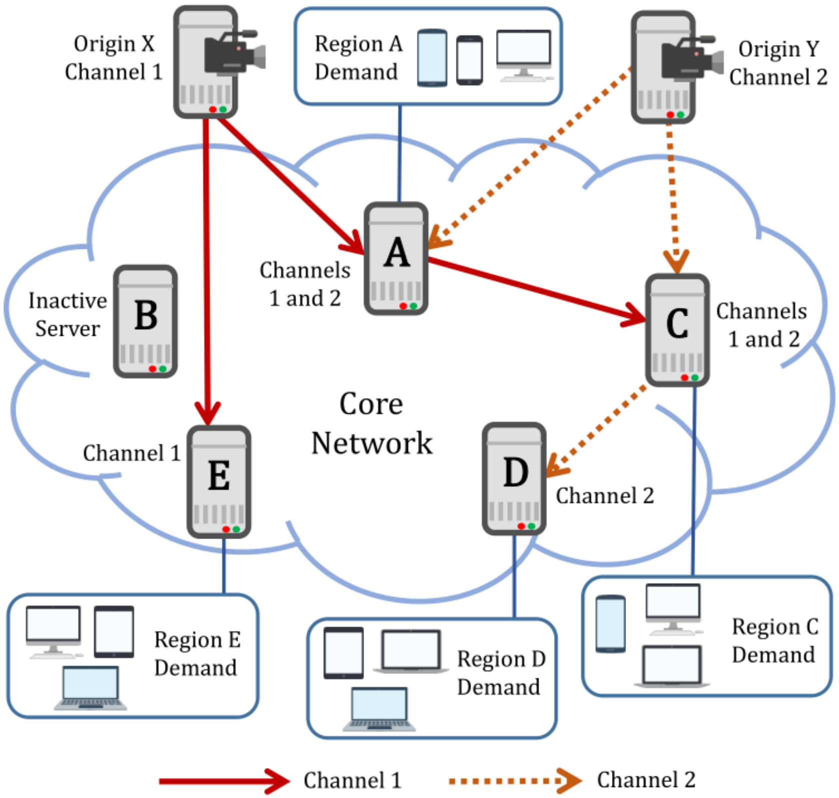


Contents

Publication:
Z. Chang and S.-H. Chan, "Bi-Criteria Approximation for a Multi-Origin Multi-Channel Auto-Scaling Live Streaming Cloud," *IEEE Transactions on Multimedia*, Vol. 25, pp. 2839-2850, July 2023.

Background:
Multi-source Multi-channel Live Streaming Cloud

Major Components in a Live Streaming Cloud	
Origin Server	Source of video channels
End Server	Stream the live content to its local users. (e.g., a CDN node, server farm, data center, etc.)
Channel	Multiple streaming rates



- **An end server** *requests* channels based on local demand and *serves* its local users.
- **End servers** (with user demands) help each other in streaming contents.
- **Channels** are *pushed* from the sources to the servers.

Bi-criteria Objectives: Deployment Cost and Source-to-end Delay

Minimize Origin-to-End Delay

Server-to-Server (S2S) Delay:
- Time to travel through a link

Origin-to-End (O2E) Delay:
- Delay from the origin server to an end server demanding the channel
- Sum of the S2S delays of links forming the path

Minimize Deployment Cost

Deployment cost consists of **Server Cost** and **Link Cost**

Server Cost: Due to the servers allocating its processing capacity to serve the other servers

Link Cost: Due to the pairwise link capacity allocated between servers

Optimizing the Bi-criteria Problem

Equivalently to

- **Minimizing** *deployment cost*
- Subject to *source-to-end delay* constraint

Overlay construction (OC):

- How to build the delivery tree of each channel?
- *To what servers* and *what channel* should a server forward?

Resource Allocation (RA)

Regularly re-optimize the overlay when parameter changes

Related Work

Fundamental difference:
COCOS is an **approximation** algorithm

	Related Work	COCOS
Traditional P2P live streaming	• Reduce server load (Tan et al. IEEE TON '13 [19], Yun et al. P2P Netw. Appl. '14 [20])	• Minimize both cost and delay
Crowdsourced live streaming	• Deliver contents from an end server to local audience (Yarnagula et al. IEEE TMM '19 [21], Irondi et al. IEEE TMM '19 [22], Li et al. SIGCOMM '22 [23])	• Orthogonal to our problem
Live streaming works focus on other objectives	• Predict QoS (Zhang et al. MM '20 [24]) • Maximize delivered channels or minimizing inter-ISP traffic (Budhkar et al. P2Pr Netw. Appl '19 [25], Sun et al. Internet Things J. '22 [26])	• Different objective

Problem Formulation: Major Symbols Used in COCOS

S	The set of sources	$\boldsymbol{b_{ij}}$	Link capacity of edge $\langle i, j \rangle$ (bits/s) **(RA)**
R	The set of auto-scaling servers	$T(m)$	The delivery tree of channel m
V	The set of all sources and servers ($V = S \cup R$)	$\boldsymbol{x_{ij}(m)}$	Binary variable indicating whether link$\langle i, j \rangle$ is in $T(m)$ **(OC)**
$\langle i, j \rangle$	The directed edge from node i to j	L_{ij}	Server-to-Server (S2S) delay of link $\langle i, j \rangle$
E	The set of all edges	$D_i(m)$	Origin-to-End (O2E) delay of channel m at server i (in second)
M	The set of all channels	$\mathbb{D}_i(m)$	O2E delay upper bound of channel m at server i (in second)
$R(m)$	The set of servers that demand channel m	Θ_i	Server cost of server i (per second)
M_i	The set of channels that server i demands	θ_i	Unit price of uploading streaming at server i (per bit)
$\tau(m)$	Streaming rate of channel m (bits/s)	Φ_{ij}	Link cost due to traffic through link $\langle i, j \rangle$ (per second)
$s(m)$	The live source of channel m	ϕ_{ij}	Unit price of data transmission through link $\langle i, j \rangle$ (per bit)
$\boldsymbol{u_i}$	Uploading capacity of node i (bits/s) **(RA)**	C	Total deployment cost (per second)

Comprehensive Model as an NP-Hard Problem: Minimum Cost Streaming with Delay Constraints

Minimize the total cost:

$$C = \sum_{\langle i,j\rangle \in E} \Phi_{ij} + \sum_{i \in V} \Theta_i$$

Cost function

- Server Cost

$$\Theta_i = \theta_i u_i, \forall i \in V.$$

- Network Cost

$$\Phi_{ij} = \phi_{ij} b_{ij}, \langle i,j\rangle \in E.$$

NP-complete Restricted Shortest Path Problem (RSP) is reducible to our problem. It is **NP-Hard**!

Delay Constraints

- Source-to-end Delay of Node j:

$$D_j\ (m) = D_i\ (m) + P_{ij} + Q_i$$

- Source-to-end Delay Constraint:

$$D_j\ (m) \leq \mathbb{D}_i\ (m), \forall i \in R(m), m \in M.$$

Fulfill Demand of Channels

$$x_{ij}(\psi) = \begin{cases} 1, & \text{if } \langle i,j\rangle \in T(\psi); \\ 0, & \text{otherwise.} \end{cases}$$

$$\sum_{\langle i,j\rangle \in E} x_{ij}(\psi) \geq 1,$$

$$\forall j \in R(m), \psi \in \Psi(m), m \in M_i.$$

Overlay Construction and Resource Allocation

Relaxed to a LP problem

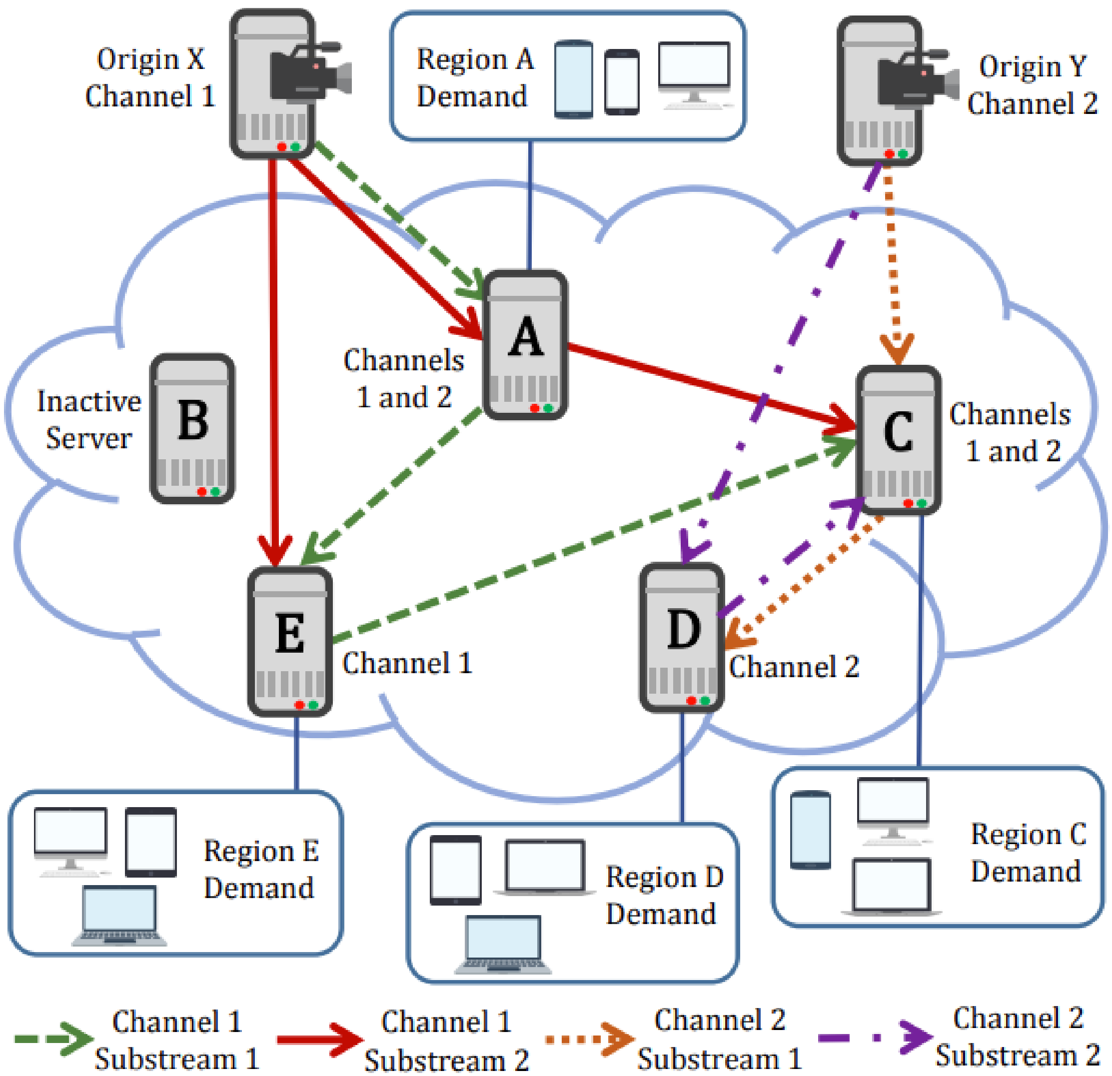
We transform the original problem through the following relaxation:

- LP solution can have arbitrary number of fractional substream paths ($0 \leq x_{ij}(\psi) \leq 1$)
- Constraints on source-to-end delay: Use average substream delay

Topology construction

- Assign a link with edges proportional to its traffic (αk edges for full stream)
- Construct trees then pick up the tree that has minimum cost and satisfies the delay constraints
- Cost approximation ratio: $\alpha + \delta$
- δ goes to 0 as k goes to infinity
- Delay approximation ratio: β
- $1/\alpha + 1/\beta \leq 1$
- Complexity $O(|V|^9|M|k)$

Example of substream solution for $k = 2$



Experimental Setup

Performance Metrics

Deployment cost

- Server cost
- Link cost

Delay

Comparison Schemes

Nearest Peer [27]

- Consider local popularity
- No cooperative replication

Prim [28]

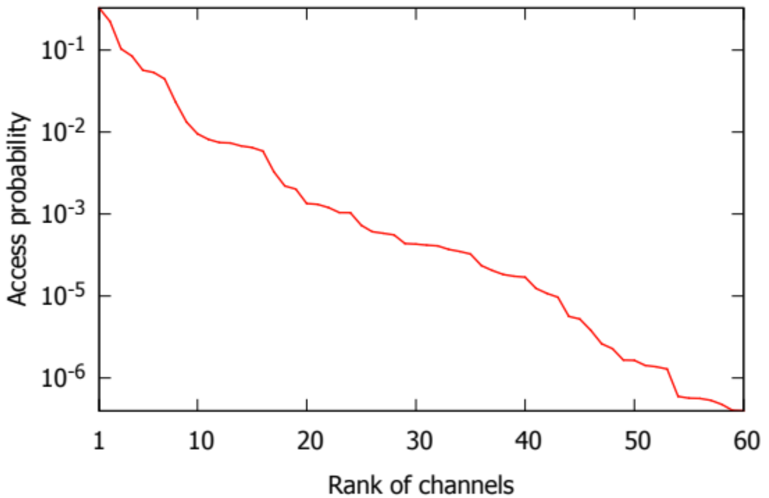
- Minimum cost tree
- Modified to meet delay constraints

Super-optimal

- Relaxed LP solution

Table 5.4. Baseline parameters used in experiments of COCOS.

Parameter	Value
Server number (origin and end) $\|V\|$	100
Number of channels $\|M\|$	60
Delay upper bound $\mathbb{D}$	800 ms
Streaming rate mean μ_τ	1.2 Mbps
Streaming rate standard deviation σ_τ	0.2 Mbps
Server price mean μ_θ	0.1 per Mbit
Server price standard deviation σ_θ	0.05 per Mbit
Link price mean μ_ϕ	0.1 per Mbit
Link price standard deviation σ_ϕ	0.05 per Mbit
Zipf's parameter z	0.5
Tradeoff parameter ε	5
Number of substreams k	10



Good Tradeoff for humble k and β

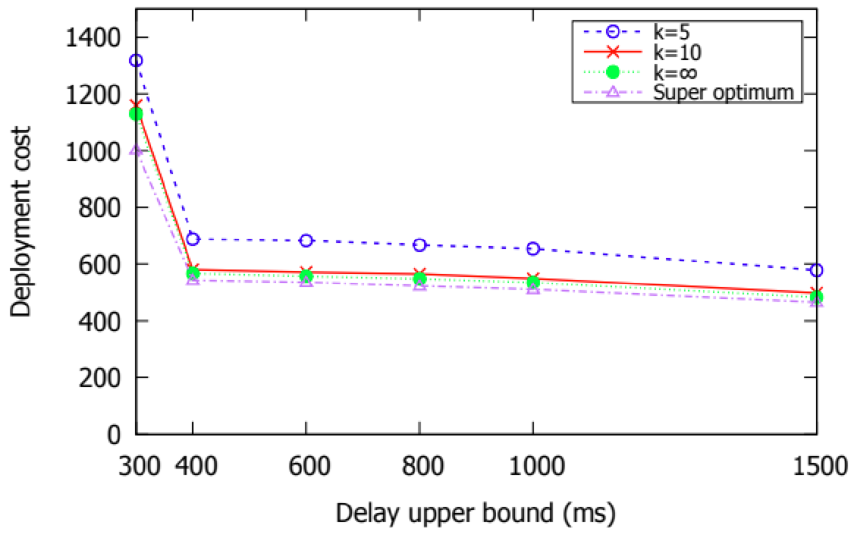


Figure 5.6. Deployment cost versus delay upper bound given different number of substreams.

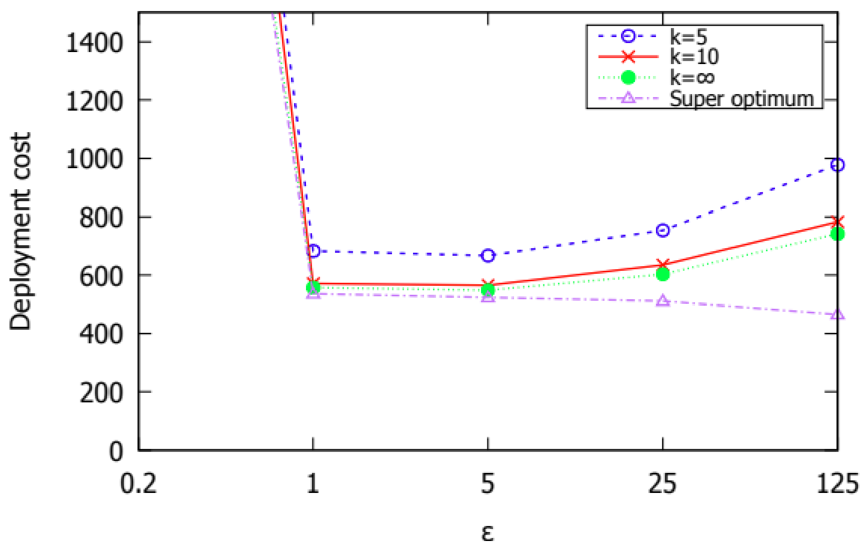
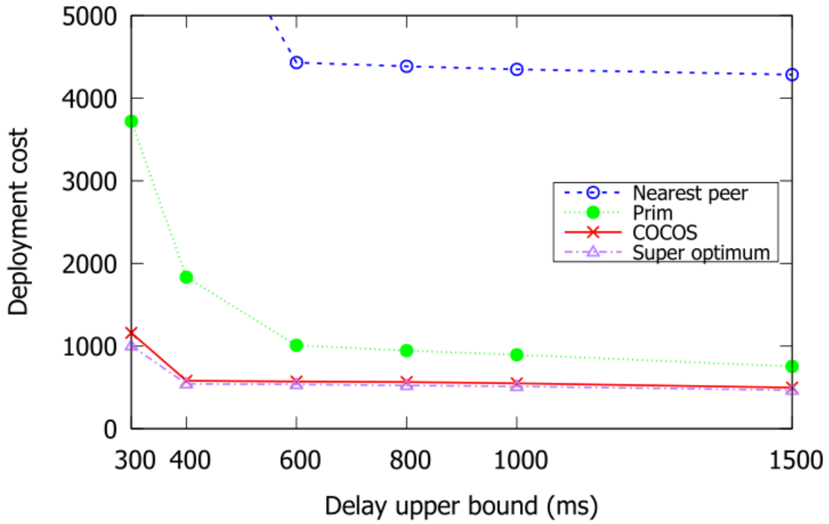


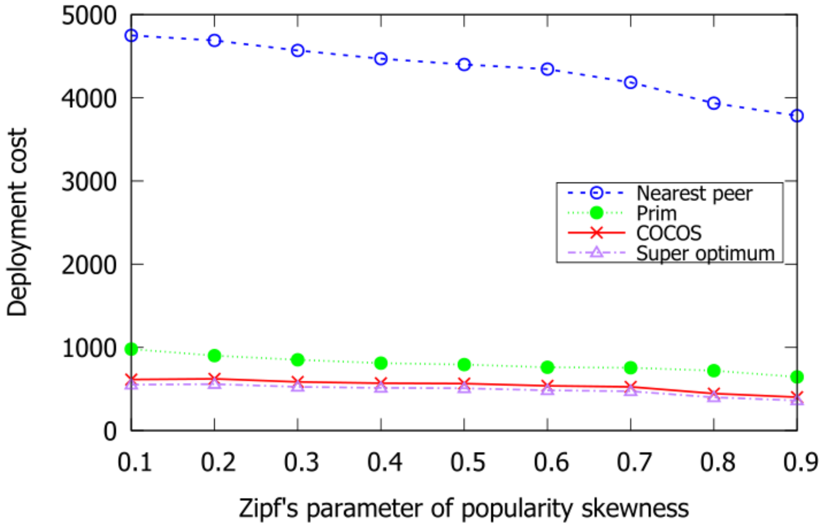
Figure 5.5. Deployment cost versus approximation ratio tradeoff parameter.

$$\alpha = 1 + \varepsilon$$

$$\beta = 1 + 1/\varepsilon$$

Near-Optimal Performance





Real-world data trace:

- From a leading video service website in China (Tencent) over 2 weeks

Near-optimal performance:

- Outperform state-of-the-art schemes

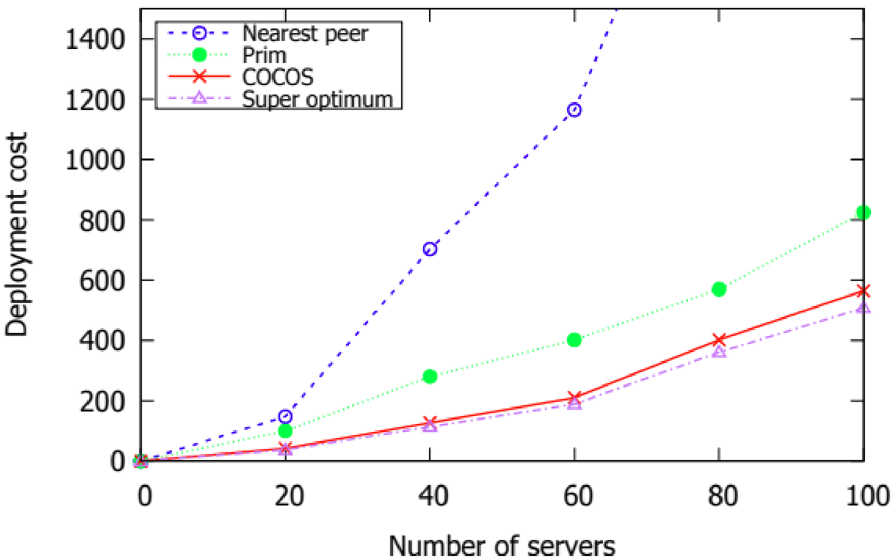
Synthetic data:

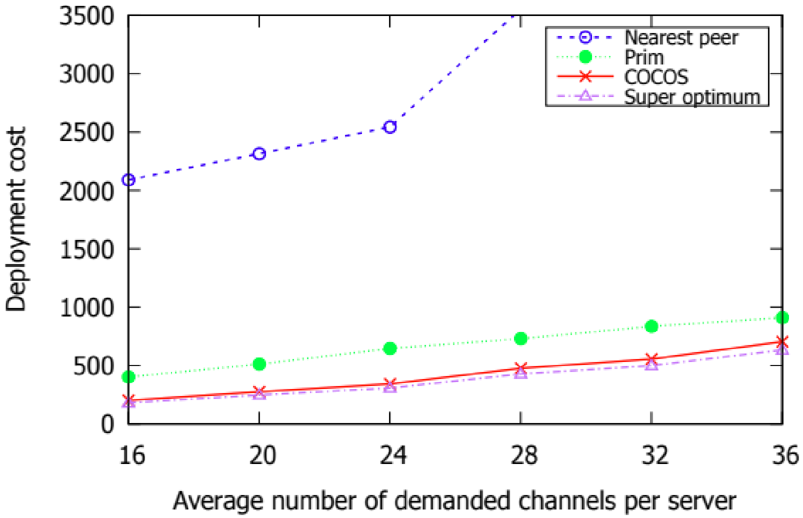
- Verify the performance given difference video popularity

Stable performance:

- Given different video popularity

Scalable Scheme





Better performance for a larger system

Contents

Conclusion

Motivations	• Video traffic: Huge volume and dynamic daily pattern • Auto-scaling: Rescale the resource elastically
Objective	• Minimize the deployment cost • Ensure the user experience
Contribution	**VoD data center** • AVARDO: Stack-based approximation algorithm **Geo-distributed VoD cloud** • RAVO: LP-based approximation • Video clustering to reduce the complexity **Multi-origin multi-channel live cloud** • COCOS: Bi-criteria approximation algorithm
Future Work	• Auto-scaling clouds for video data analytics • Auto-scaling clouds for emerging applications • Sustainable video services

References

[1] Sandvine Corporation. 2023 Global internet phenomena report, 2023.

[2] Ning Liu, Huajie Cui, S.-H. Gary Chan, Zhipeng Chen, and Yirong Zhuang. Dissecting user behaviors for a simultaneous live and VoD IPTV system. *ACM Transactions on Multimedia Computing, Communications and Applications,* 10(3):23:1 – 23:16, April 2014.

[3] M. Cha, H. Kwak, P. Rodriguez, Y. Y. Ahn, and S. Moon. Analyzing the Video Popularity Characteristics of Large-Scale User Generated Content Systems. *IEEEACM Trans. Netw.,* 17(5):1357–1370, October 2009.

[4] Jingqi Yang, Chuanchang Liu, Yanlei Shang, Bo Cheng, Zexiang Mao, Chunhong Liu, Lisha Niu, and Junliang Chen. A cost-aware auto-scaling approach using the workload prediction in service clouds. *Information Systems Frontiers*, 16(1):7–18, Mar 2014.

[5] J. Yang, Z. Yao, B. Yang, X. Tan, Z. Wang, and Q. Zheng. Software-defined multimedia streaming system aided by variable-length interval in-network caching. *IEEE Transactions on Multimedia*, 21(2):494–509, Feb 2019.

[6] E. Bourtsoulatze, N. Thomos, J. Saltarin, and T. Braun. Content-aware delivery of scalable video in network coding enabled named data networks. *IEEE Transactions on Multimedia*, 20(6):1561–1575, June 2018.

[7] Gerta Sheganaku, Stefan Schulte, Philipp Waibel, and Ingo Weber. Cost-efficient auto-scaling of container-based elastic processes. *Future Generation Computer Systems*, 138:296–312, 2023.

[8] Hui Zhao, Jing Wang, Quan Wang, and Feng Liu. Queue-based and learningbased dynamic resources allocation for virtual streaming media server cluster of multi-version VoD system. *Multimedia Tools and Applications*, 78:21827–21852, Apr 2019.

[9] Shutian Luo, Huanle Xu, Kejiang Ye, Guoyao Xu, Liping Zhang, Guodong Yang, and Chengzhong Xu. The power of prediction: Microservice auto scaling via workload learning. In Proceedings of the 13th Symposium on Cloud Computing, SoCC ’22, page 355–369, New York, NY, USA, 2022. ACM.

[10] J. Du, C. Jiang, Y. Qian, Z. Han, and Y. Ren. Resource allocation with video traffic prediction in cloud-based space systems. IEEE Transactions on Multimedia, 18(5):820–830, May 2016.

[11] Fei Chen, Haitao Li, Jiangchuan Liu, Bo Li, Ke Xu, and Yuemin Hu. Migrating big video data to cloud: A peer-assisted approach for VoD. Peer-to-Peer Netw. Appl., 11:1060–1074, July 2018.

[12] Vijay Kumar Adhikari, Yang Guo, Fang Hao, Matteo Varvello, Volker Hilt, Moritz Steiner, and Zhi-Li Zhang. Unreeling netflix: Understanding and improving multi-cdn movie delivery. In *INFOCOM, 2012 Proceedings IEEE*, pages 1620–1628, Orlando, FL, USA, 2012. IEEE, IEEE.

[13] David Applegate, Aaron Archer, Vijay Gopalakrishnan, Seungjoon Lee, and Kadangode K Ramakrishnan. Optimal content placement for a large-scale vod system. In Proc. *The 6th International Conference (Co-NEXT ’10),* page 4, Philadelphia, USA, 2010. ACM, ACM.

References (Cont’d)

[14] Kaku Minowa and Tomoki Yoshihisa. Pre-cache methods for accommodating more clients in edge-assisted video-on-demand systems. In Leonard Barolli, Hiroyoshi Miwa, and Tomoya Enokido, editors, Advances in Network-Based Information Systems, pages 289–297, Cham, 2022. Springer International Publishing.

[15] Minghong Lin, Adam Wierman, Lachlan LH Andrew, and Eno Thereska. Dynamic right-sizing for power-proportional data centers. In *INFOCOM, 2011 Proceedings IEEE*, pages 1098–1106, Shanghai, China, 2011. IEEE, IEEE.

[16] Haitao Liu, Qingkui Chen, and Puchen Liu. An optimization method of large-scale video stream concurrent transmission for edge computing. *Mathematics*, 11(12), 2023.

[17] Nikolaos Laoutaris, Vassilios Zissimopoulos, and Ioannis Stavrakakis. On the optimization of storage capacity allocation for content distribution. *Computer Networks*, 47(3):409–428, 2005.

[18] Mingfu Li and Chun-Huei Wu. A cost-effective resource allocation and management scheme for content networks supporting IPTV services. *Computer Communications*, 33(1):83–91, 2010.

[19] Bo Tan and Laurent Massoulie. Optimal content placement for peer-to-peer video-on-demand systems. IEEE/ACM Transactions on Networking, 21(2):566–579, April 2013.

[20] Sunghyun Yun, Heuiseok Lim, and Kyungyong Chung. The biometric signature delegation scheme to balance the load of digital signing in hybrid P2P networks. Peer-to-Peer Networking and Applications, pages 1–10, 2014.

[21] Hema Kumar Yarnagula, Parikshit Juluri, Sheyda Kiani Mehr, Venkatesh Tamarapalli, and Deep Medhi. QoE for mobile clients with segment-aware rate adaptation algorithm (SARA) for DASH video streaming. *ACM Transactions on Multimedia Computing Communications and Applications*, 15(2):1–23, June 2019.

[22] Iheanyi Irondi, Qi Wang, Christos Grecos, Jose M. Alcaraz Calero, and Pablo Casaseca-De-La-Higuera. Efficient QoE-Aware scheme for video quality switching operations in dynamic adaptive streaming. *ACM Transactions on Multimedia Computing Communications and Applicati*ons, 15(1):1–23, February 2019.

[23] Jinyang Li, Zhenyu Li, Ri Lu, Kai Xiao, Songlin Li, Jufeng Chen, Jingyu Yang, Chunli Zong, Aiyun Chen, Qinghua Wu, Chen Sun, Gareth Tyson, and Hongqiang Harry Liu. Livenet: A low-latency video transport network for large-scale live streaming. In Proceedings of the ACM SIGCOMM 2022 Conference, SIGCOMM ’22, pages 812–825, New York, NY, USA, 2022. ACM.

[24] Rui-Xiao Zhang, Ming Ma, Tianchi Huang, Hanyu Li, Jiangchuan Liu, and Lifeng Sun. Leveraging QoE heterogenity for large-scale livecaset scheduling. In Proceedings of *the 28th ACM International Conference on Multimedia*, MM ’20, pages 3678–3686, New York, NY, USA, 2020. Association for Computing Machinery.

References (Cont’d 2)

[25] Shilpa Budhkar and Venkatesh Tamarapalli. An overlay management strategy to improve QoS in CDN-P2P live streaming systems. *Peer-to-Peer Networking and Applications*, pages 1–17, 2019.

[26] Hui Sun, Qiyuan Li, Kewei Sha, and Ying Yu. Elastic-edge: An intelligent elastic edge framework for live video analytics. IEEE Internet of Things Journal, 9(22):23031–23046, 2022.

[27] Scalable and reliable live streaming service through coordinating CDN and P2P. In *Parallel and Distributed Systems (ICPADS), 2011 IEEE 17th International Conference on*, pages 581–588. IEEE, 2011.

[28] Rosario Giuseppe Garroppo, Stefano Giordano, Stella Spagna, Saverio Niccolini, and Jan Seedorf. Design and evaluation of an optimized overlay topology for a single operator video streaming service. *In Proceedings of the 2010 ACM Workshop on Advanced Video Streaming Techniques for Peer-to-Peer Networks and Social Networking, AVSTP2P ’10*, pages 49–54, New York, NY, USA, 2010. Association for Computing Machinery.

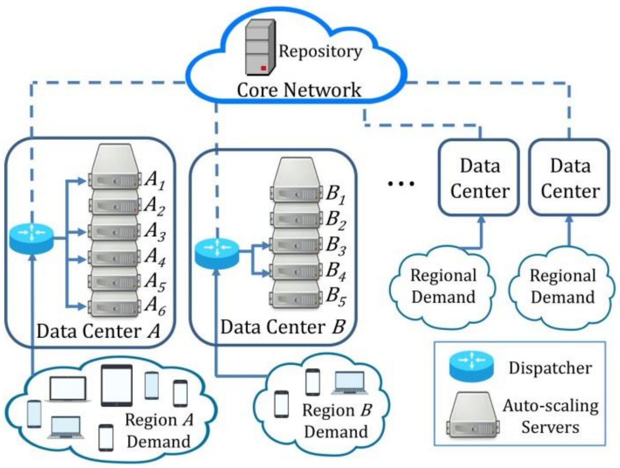
List of Related Publications

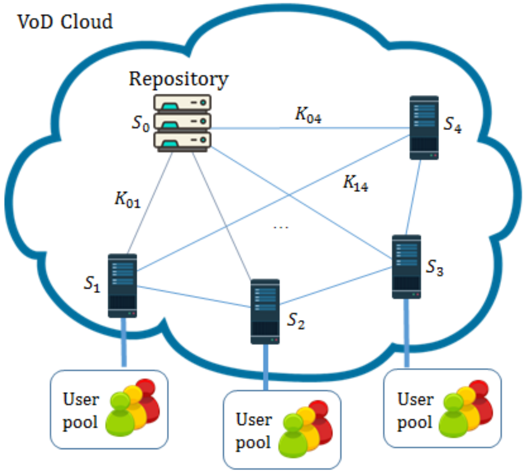
Journal publications

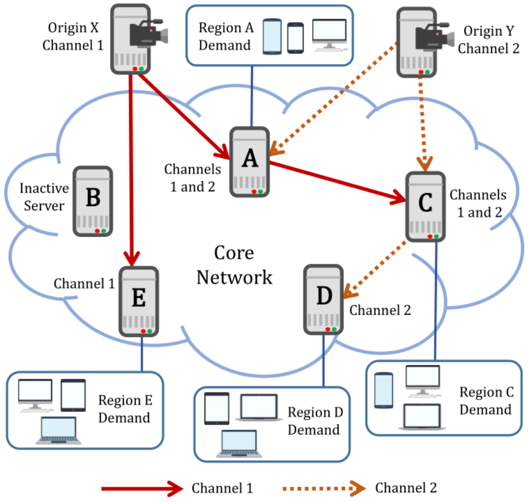
1. **Z. Chang** and S.-H. Chan, "Bi-Criteria Approximation for a Multi-Origin Multi-Channel Auto-Scaling Live Streaming Cloud," *IEEE Transactions on Multimedia*, Vol. 25, pp. 2839-2850, July 2023.
2. **Z. Chang** and S.-H. Chan, "An Approximation Algorithm to Maximize User Capacity for an Auto-scaling VoD System," *IEEE Transactions on Multimedia*, Vol. 23, pp. 3714-3725, October 2021.
3. **Z. Chang** and S.-H. Chan, "Video Management and Resource Allocation for a Large-scale VoD Cloud," *ACM Transactions on Multimedia Computing, Communication and Applications (TOMM) Special Issue on Multimedia Big Data: Networking*, Vol. 12, No. 5s, pp. 72:1-72:21, Sept 2016.
4. **Z. Chang** and S.-H. Chan, "Bucket-Filling: An Asymptotically Optimal VoD Network with Source Coding," *IEEE Transactions on Multimedia*, Vol. 17, No. 5, pp. 723-735, May 2015.

Conference publications

1. J. Dai, **Z. Chang** and S.-H. Chan, "Delay Optimization for Multi-source Multi-channel Overlay Live Streaming," in *Proceedings of IEEE ICC 2015 - Communications Software, Services and Multimedia Applications Symposium (ICC'15)*, (London, United Kingdom), pp. 8587-92, 8-12 June 2015.







Thank You!

Any Questions?