INVAR: Inversion Aware Resource Provisioning and Workload Scheduling for Edge Computing

Bin Wang, David Irwin, Prashant Shenoy and Don Towsley

Manning College of Information and Computer Sciences, UMass Amherst, USA

作者团队背景

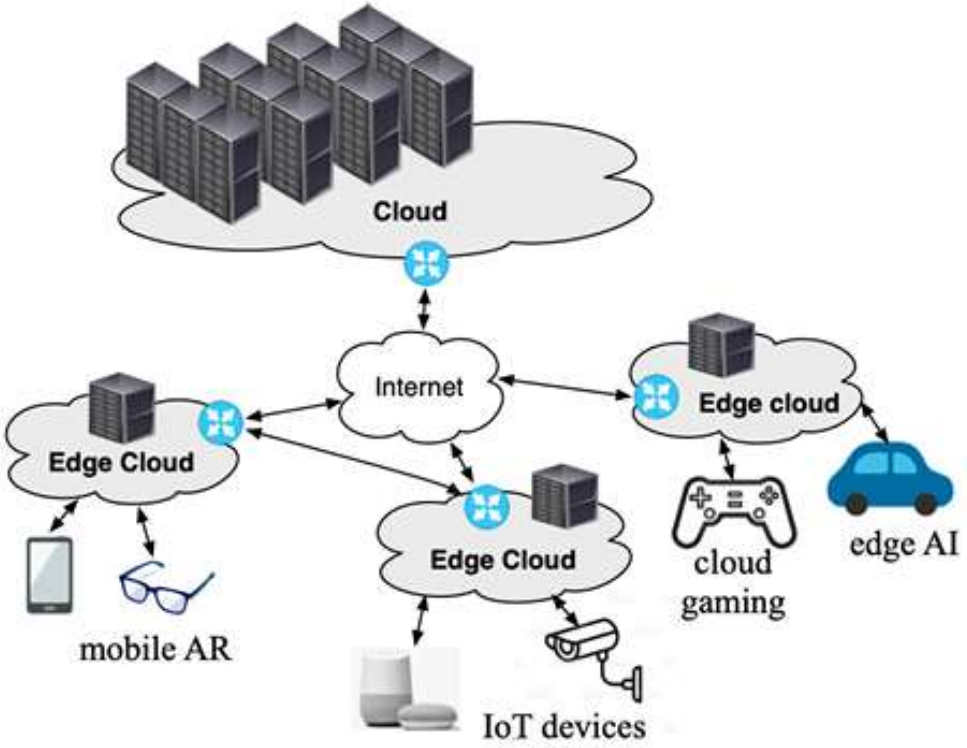
Research Interests

- Cloud Computing & Cloud Platform
- Data Center & Edge Server
- Queueing System & Response Time
- Resource Allocation
- Amazon Web Services

https://people.cs.umass.edu/~binwang/

目录

Introduction

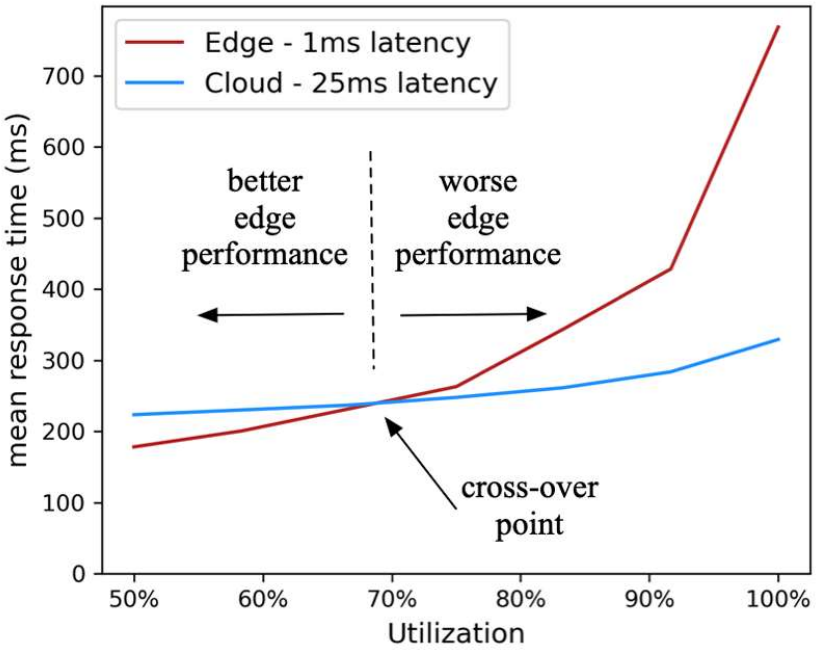


边缘计算 => 云计算：

高实时性高并发的应用场景

更低的传输延时

Introduction

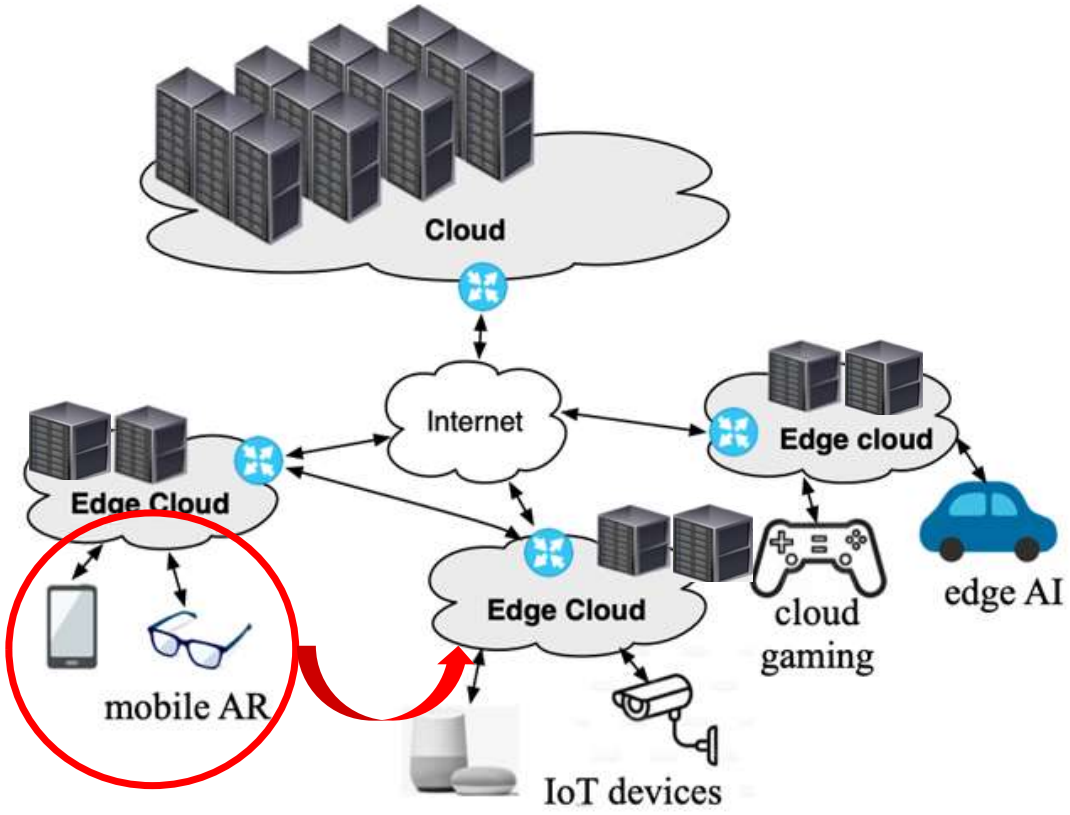


边缘性能倒置现象：

边缘优先响应时间 > 云计算响应时间

响应时间 = 传输延迟 + 排队延迟 +服务时间

Introduction

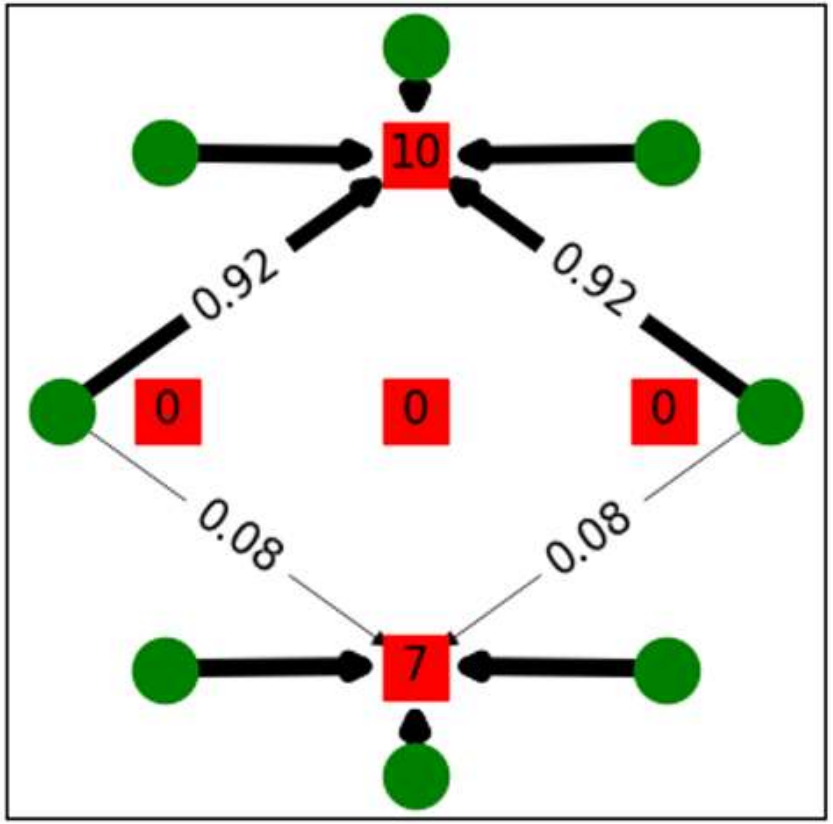


常见的解决方法：

1. 增加服务器资源

2. 重定向至其他数据中心

Introduction



概率调度方法：

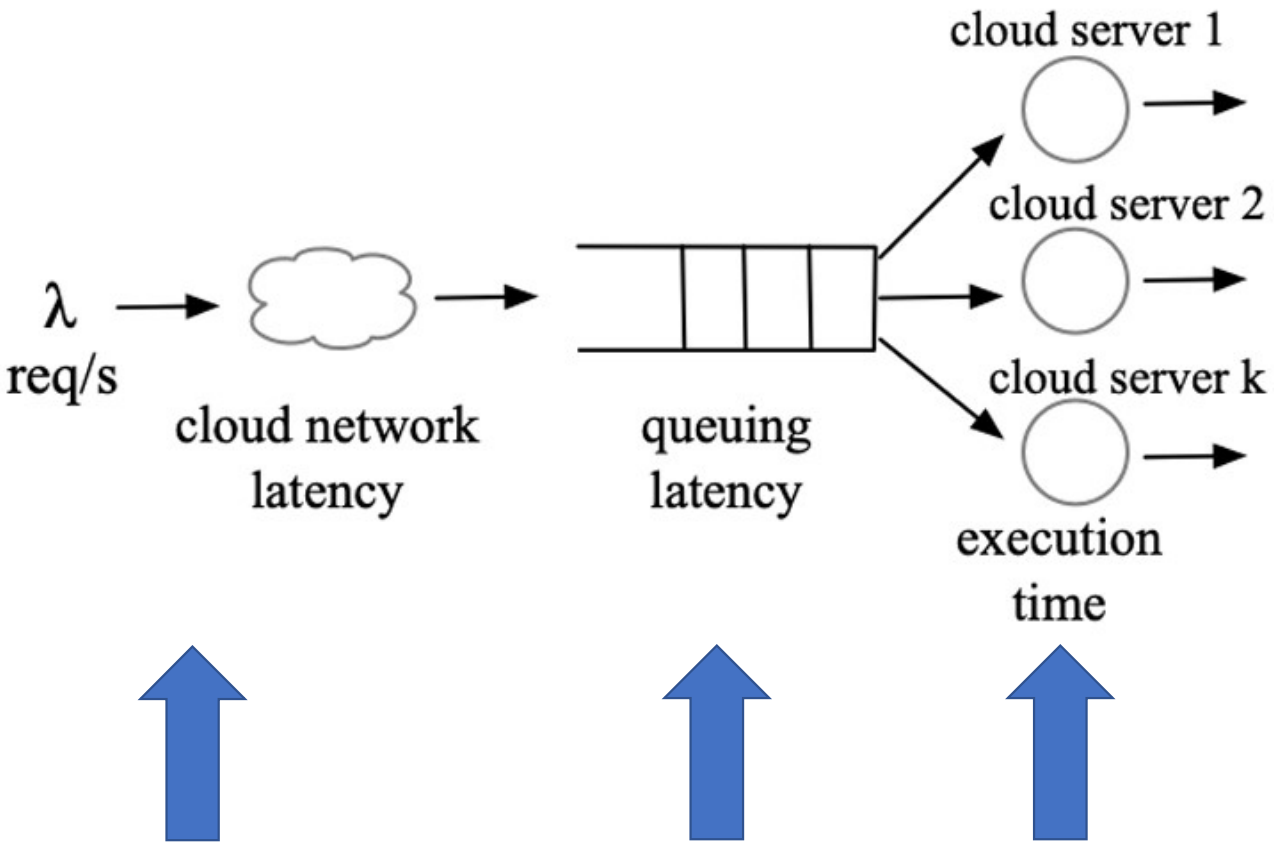
用户请求概率地指向某个数据中心

绿色：用户地址

红色：数据中心（中间为云，四周为边缘）

有向线段：请求发送给某个数据中心的概率

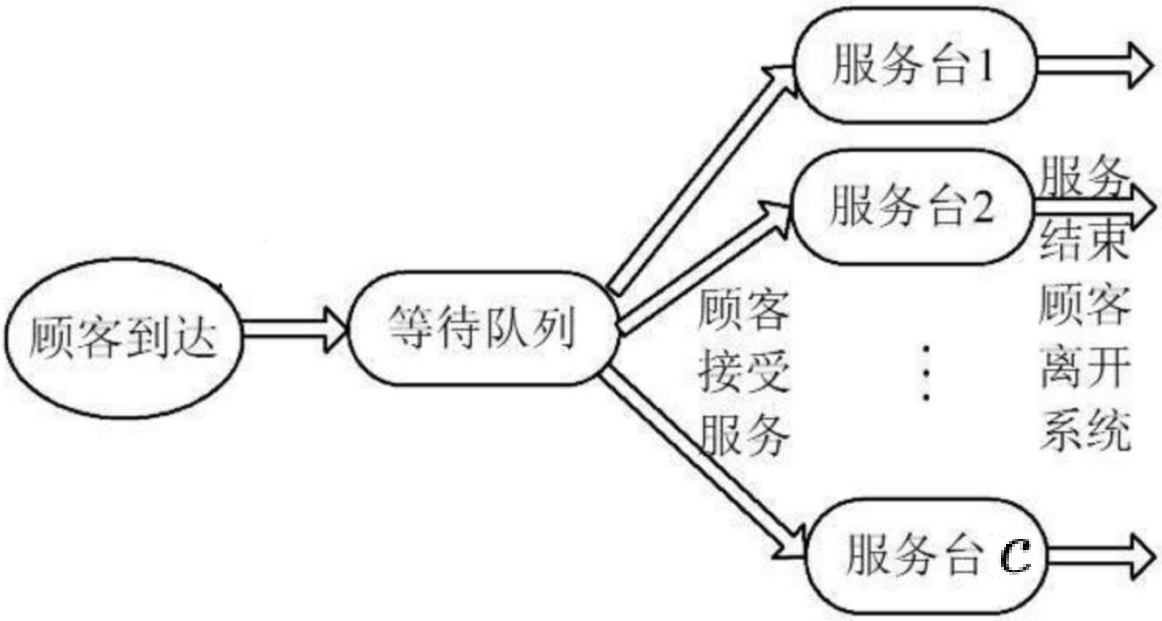
Background



云/边缘端处理请求：

1. 传输延迟
2. 排队延迟
3. 服务时间

Design



单个数据中心的M/M/c队列

L个用户位置（顾客）

K个数据中心

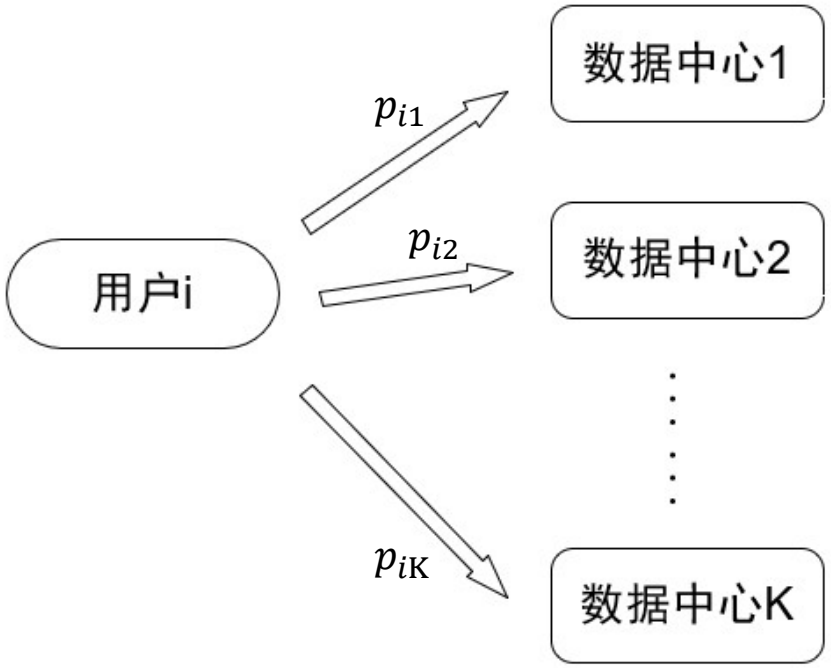
每个数据中心j有c_j个服务器（服务台）

用户i平均请求达到率为γ_{i}

数据中心j平均服务率为μ_{j}

数据中心j开销为$c_j\omega_{\mathrm{j}}$

Design



概率调度M/M/c队列

数据中心j的有效到达率：

$$\lambda_{\mathrm{j}} = \sum_{i=1}^{L} \gamma_i * p_{ij}$$

Design

在数据中心的平均花费时间 = 等待时间 + 服务时间：

$$T_j^D = \frac{C(c_j, \lambda_j/\mu_j)}{c_j\mu_j - \lambda_j} + \frac{1}{\mu_j}$$

其中：

$$C(c, \lambda/\mu) = \frac{(\frac{(c\rho)^c}{c!})(\frac{1}{1-\rho})}{\sum_{k=0}^{c-1}\frac{(c\rho)^k}{k!} + \frac{(c\rho)^c}{c!}\frac{1}{(1-\rho)}} \qquad \rho = \frac{\lambda}{c\mu}$$

所有用户的平均响应时间：

$$T(c, p) = \sum_{i=1}^{L}\frac{\gamma_i}{||\gamma||_1}\sum_{j=1}^{K}p_{ij}(T_{ij}^N + T_j^D)$$

Design

$$
\begin{aligned}
\min_{c,P} \quad & T(c,P) \\
\text{s.t.} \quad & 0 \le p_{i,j} \le 1 \quad \forall i \in 1 \dots L, \forall j \in 1 \dots K \\
& 0 \le c_j \le C_j \quad \forall j \in 1 \dots K \\
& \sum_{j=1}^{K} p_{i,j} = 1 \quad \forall i \in 1 \dots L \\
& \lambda_j < c_j \mu_j \quad \forall j \in 1 \dots K \\
& \sum_{j=1}^{K} c_j w_j \le W
\end{aligned}
$$

NP难！

Design

$$C(c,\lambda/\mu) \le [\frac{\lambda}{c\mu} + \eta(\frac{\Phi(\alpha)}{\phi(\alpha)} + \frac{2}{3}\frac{1}{\sqrt{c}})]$$ [1]

$$\phi(\alpha) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}\alpha^2} \qquad \Phi(\alpha) = \frac{1}{2}[1 + erf(\frac{\alpha}{\sqrt{2}})]$$

$$erf(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$$

$$\alpha = \sqrt{-2c(1-\rho+ln\rho)} \qquad \eta = (c-\lambda/\mu)/\sqrt{c} \qquad \rho = \frac{\lambda}{c\mu}$$

[1] Refining square-root safety staffing by expanding Erlang C

Design

$$\min_{c,P} \quad T'(c,P) + \tau \left(\sum_{j=1}^{K} c_j w_j - W \right)^2$$

$$\begin{aligned}
\text{s.t.} \quad & 0 \leq p_{i,j} \leq 1 && \forall i \in 1 \ldots L, \forall j \in 1 \ldots K \\
& 0 \leq c_j \leq C_j && \forall j \in 1 \ldots K \\
& \sum_{j=1}^{K} p_{i,j} = 1 && \forall i \in 1 \ldots L \\
& \lambda_j < c_j \mu_j && \forall j \in 1 \ldots K
\end{aligned}$$

PERT-OPT

Design

$$\begin{aligned}
\min_{P} \quad & T(c, P) \\
\text{s.t.} \quad & 0 \leq p_{i,j} \leq 1 \quad \forall i \in 1 \ldots L, \forall j \in 1 \ldots K \\
& \sum_{j=1}^{K} p_{i,j} = 1 \qquad \forall i \in 1 \ldots L \\
& \lambda_j < c_j \mu_j \qquad \forall j \in 1 \ldots K
\end{aligned}$$

FLOW-OPT

Design

Algorithm 1 Inversion Aware Deployment Plan Search Algorithm

Input: L, γ, K, C, μ, w, T^N, T^{cloud}, δ, W, ϵ
Output: c, P

$T^{target} \leftarrow T^{cloud} - \delta$
$W^l \leftarrow 0$
$W^u \leftarrow W$
while $W^u - W^l > \epsilon$ **do**
  $W^m \leftarrow \frac{W^l + W^u}{2}$
  status, $c, P \leftarrow$ PERF_OPT$(L, \gamma, K, C, \mu, w, W^m)$
  if status = solved **and** $T'(c, P) \leq T^{target}$ **then**
    $W^u \leftarrow W^m$
  else
    $W^l \leftarrow W^m$
  end if
end while
status, $c, P \leftarrow$ PERF_OPT$(L, \gamma, K, C, \mu, w, W^u)$
if $T'(c, P) \leq T^{target}$ **then**
  $c \leftarrow$ round(c)
  $P \leftarrow$ FLOW_OPT(L, γ, μ, c)
  return c, P
else
  return "NO SOLUTION FOUND"
end if

STEP1：二分查找得出最小成本

利用PERT-OPT得出分配向量

STEP2：分数解转为整数解

STEP3：通过FLOW-OPT得出分配概率

Evaluation

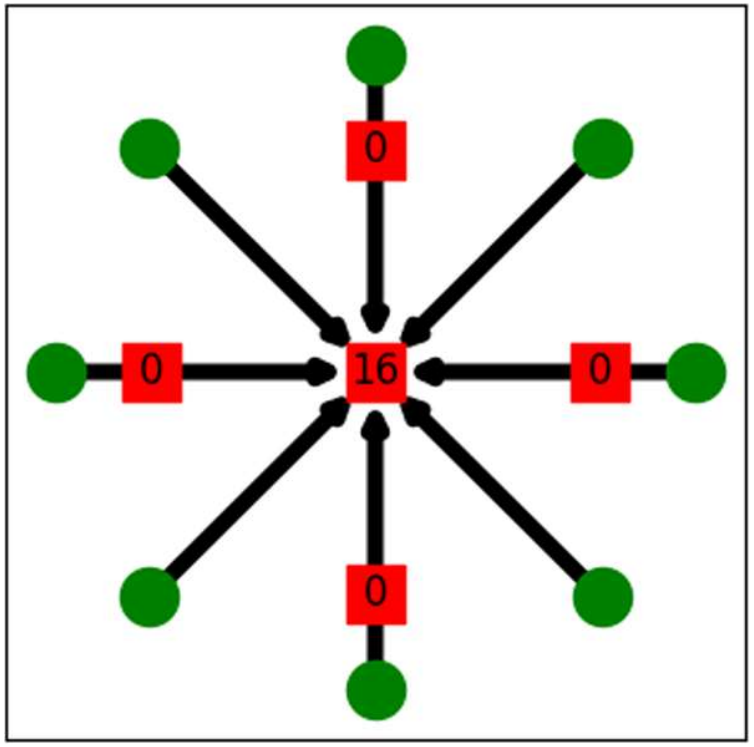
Julia + JuMP

PERT-OPT：Artelys Knitro + 多起点

FLOW-OPT：Ipopt + 内点法

数值模拟：离散事件模拟器DES

Evaluation



模拟场景

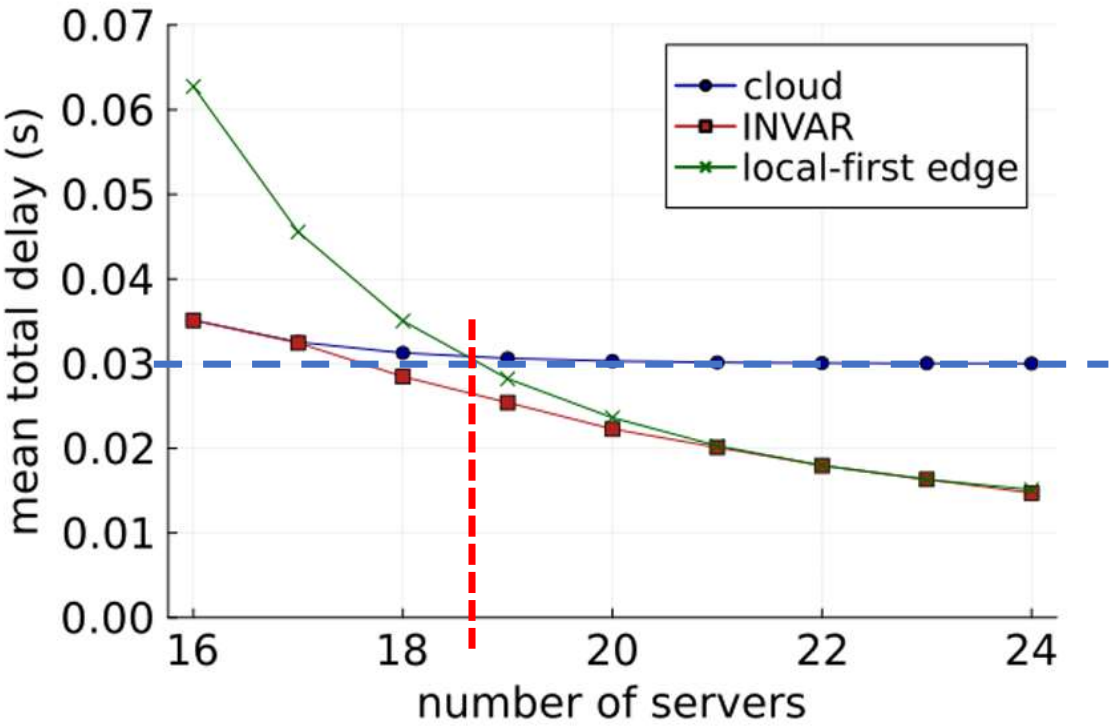
绿色：用户地址$[\gamma_i = 15]$

红色：数据中心（中间为云，四周为边缘）

$[\mu_j = 10, \omega_j = 1]$

云数据中心容量上限为50，边缘数据中心为10

Evaluation



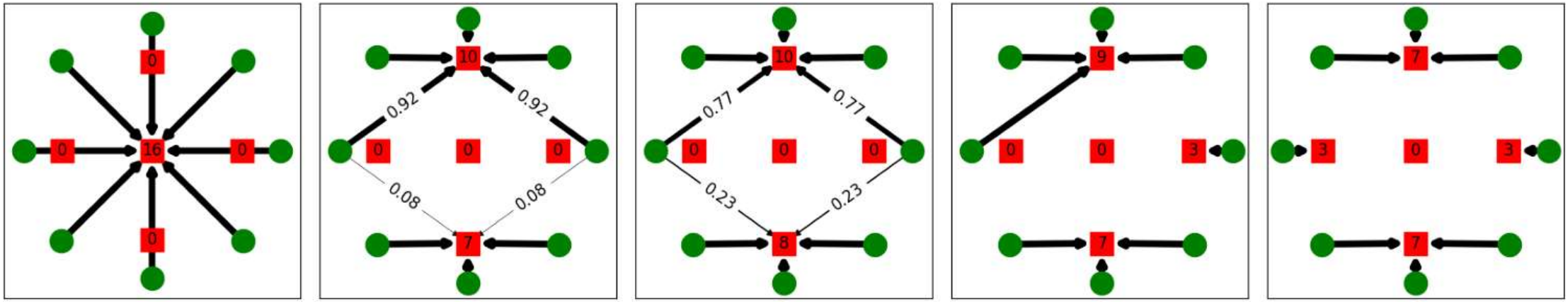
服务器总数由16 => 24

云部署逐渐趋近30ms极限

边缘性能倒置：< 19台

INVAR 一直保持最优

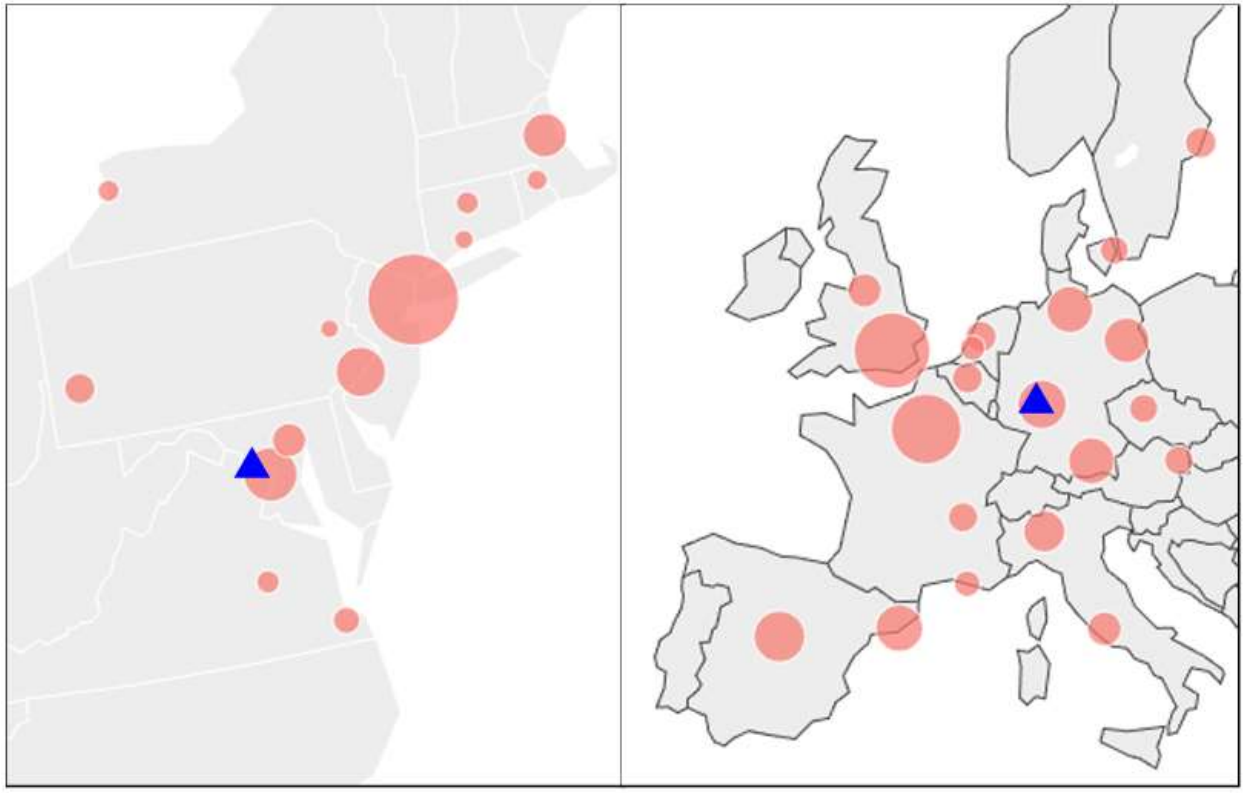
Evaluation



(a) budget=16 (b) budget=17 (c) budget=18 (d) budget=19 (e) budget=20

概率调度部署计划 (服务器数量 16 => 24)

Evaluation



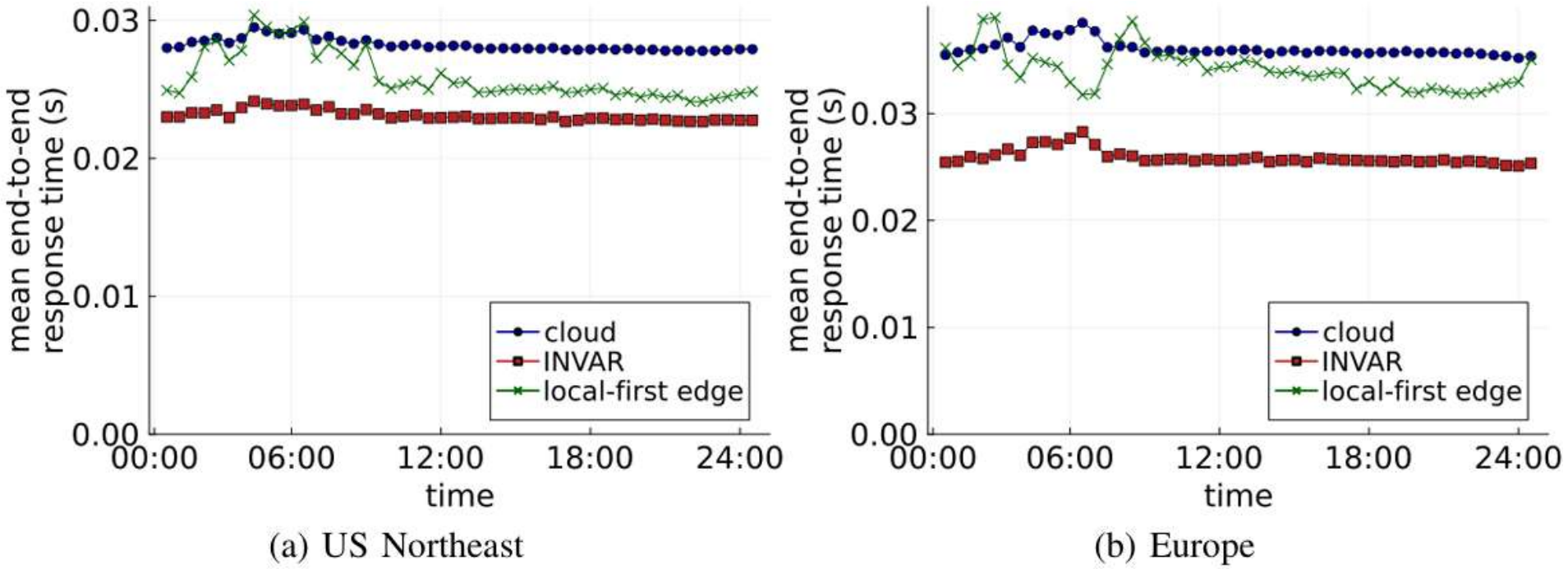
(a) US Northeast (b) Europe

2013 年 8 月 Akamai

美国东北部 (请求主要分布四大城)

欧洲地区 (请求分布更为均匀)

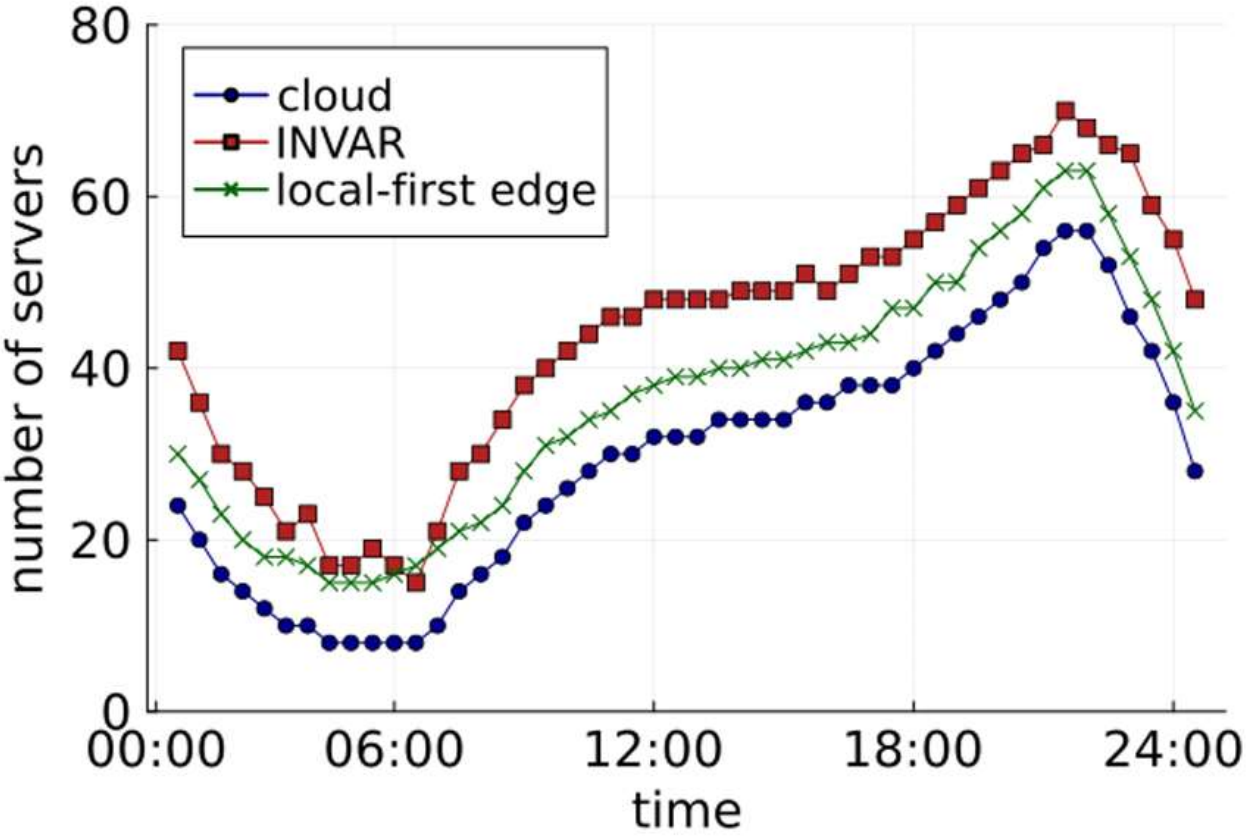
Evaluation



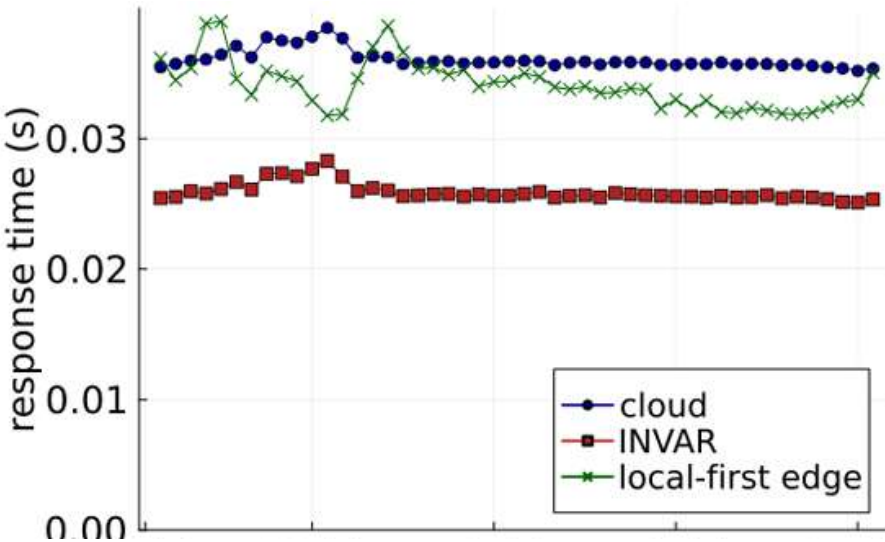
(a) US Northeast (b) Europe

出现了边缘性能倒置

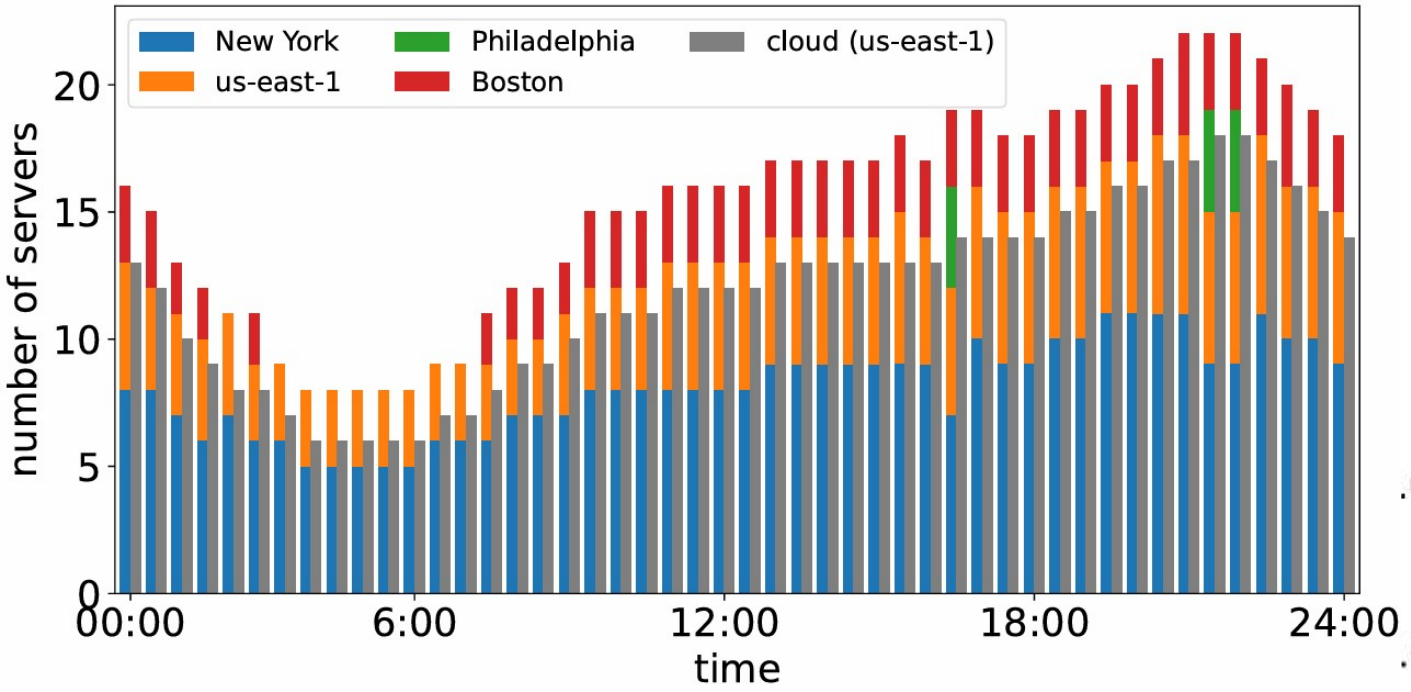
Evaluation



欧洲地区服务器部署数量

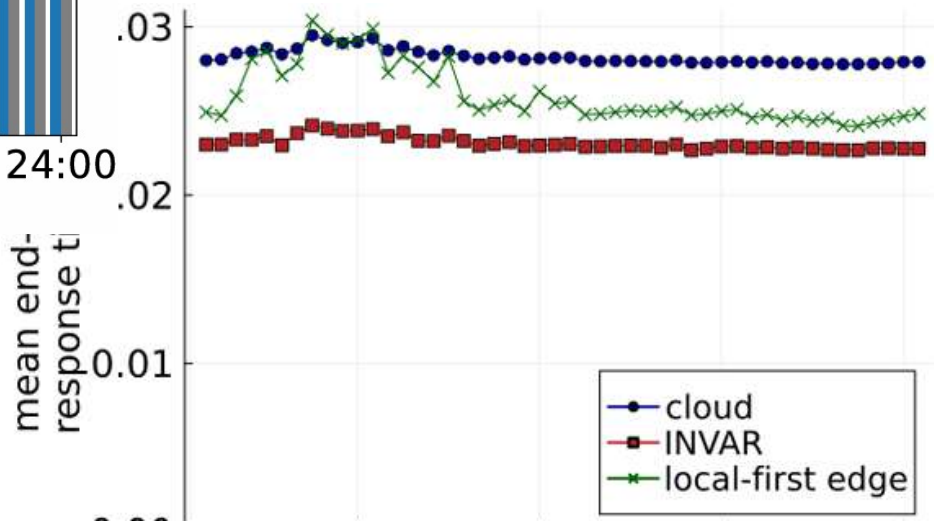


Evaluation



美国东北部服务器部署数量

不严谨！



Evaluation

Algorithm 1 Inversion Aware Deployment Plan Search Algorithm

Input: L, γ, K, C, μ, w, T^N, T^{cloud}, δ, W, ϵ
Output: c, P

$T^{target} \leftarrow T^{cloud} - \delta$
$W^l \leftarrow 0$
$W^u \leftarrow W$
while $W^u - W^l > \epsilon$ **do**
  $W^m \leftarrow \frac{W^l + W^u}{2}$
  status, $c, P \leftarrow$ PERF_OPT$(L, \gamma, K, C, \mu, w, W^m)$
  if status = solved **and** $T'(c, P) \leq T^{target}$ **then**
    $W^u \leftarrow W^m$
  else
    $W^l \leftarrow W^m$
  end if
end while
status, $c, P \leftarrow$ PERF_OPT$(L, \gamma, K, C, \mu, w, W^u)$
if $T'(c, P) \leq T^{Target}$ **then**
  $c \leftarrow$ round(c)
  $P \leftarrow$ FLOW_OPT(L, γ, μ, c)
  return c, P
else
  return "NO SOLUTION FOUND"
end if

必然的延时降低

Evaluation

Method — Distribution	mean	95% CI
Numerical Analysis — $E(\gamma)$	0.0229955	–
Simulation — $E(\gamma)$	0.0230106	(0.0229797, 0.0230415)
Simulation — $U(0, 2/\gamma)$	0.0227394	(0.0227214, 0.0227574)
Simulation — $\Gamma(3, 3\gamma)$	0.0226909	(0.022672, 0.0227097)
Simulation — $\Gamma(5, 5\gamma)$	0.0226576	(0.0226449, 0.0226702)

到达时间

Method — Distribution	mean	95% CI
Numerical Analysis — $E(\mu)$	0.0229955	–
Simulation — $E(\mu)$	0.0230106	(0.0229797, 0.0230415)
Simulation — $U(0, 2/\mu)$	0.0228921	(0.0228825, 0.0229017)
Simulation — $\Gamma(3, 3\mu)$	0.0228781	(0.022865, 0.0228911)
Simulation — $\Gamma(5, 5\mu)$	0.0228574	(0.0228469, 0.0228679)

服务时间

Conclusion

边缘性能倒置问题：

引入概率调度方法降低高请求边缘数据中心排队延迟

存在逻辑问题，可以改进！

IDEAS

1.在同一个成本下进行INVAR相关实验

2. 云数据中心 + 边缘数据中心协同调度

3. 一个请求可能需要n个服务器共同处理

4. 分配概率动态调整

Thanks for listening
请老师同学们批评指正

汇报人：黄　　凯
2024 年 11 月 10 日