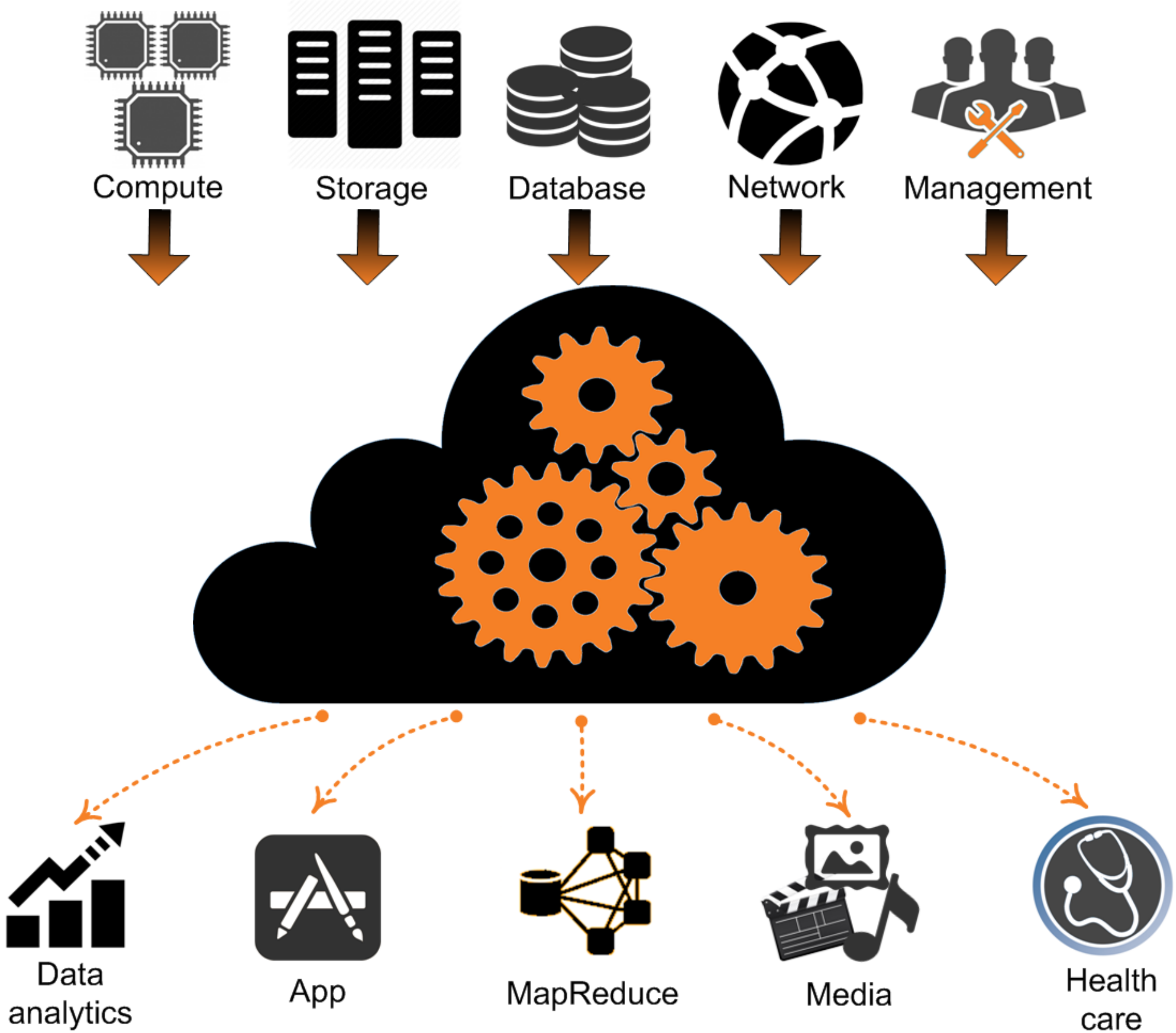
SCALING UP COMPUTATION: CLOUD ECONOMICS & AUTOMATION

Chee Wei Tan

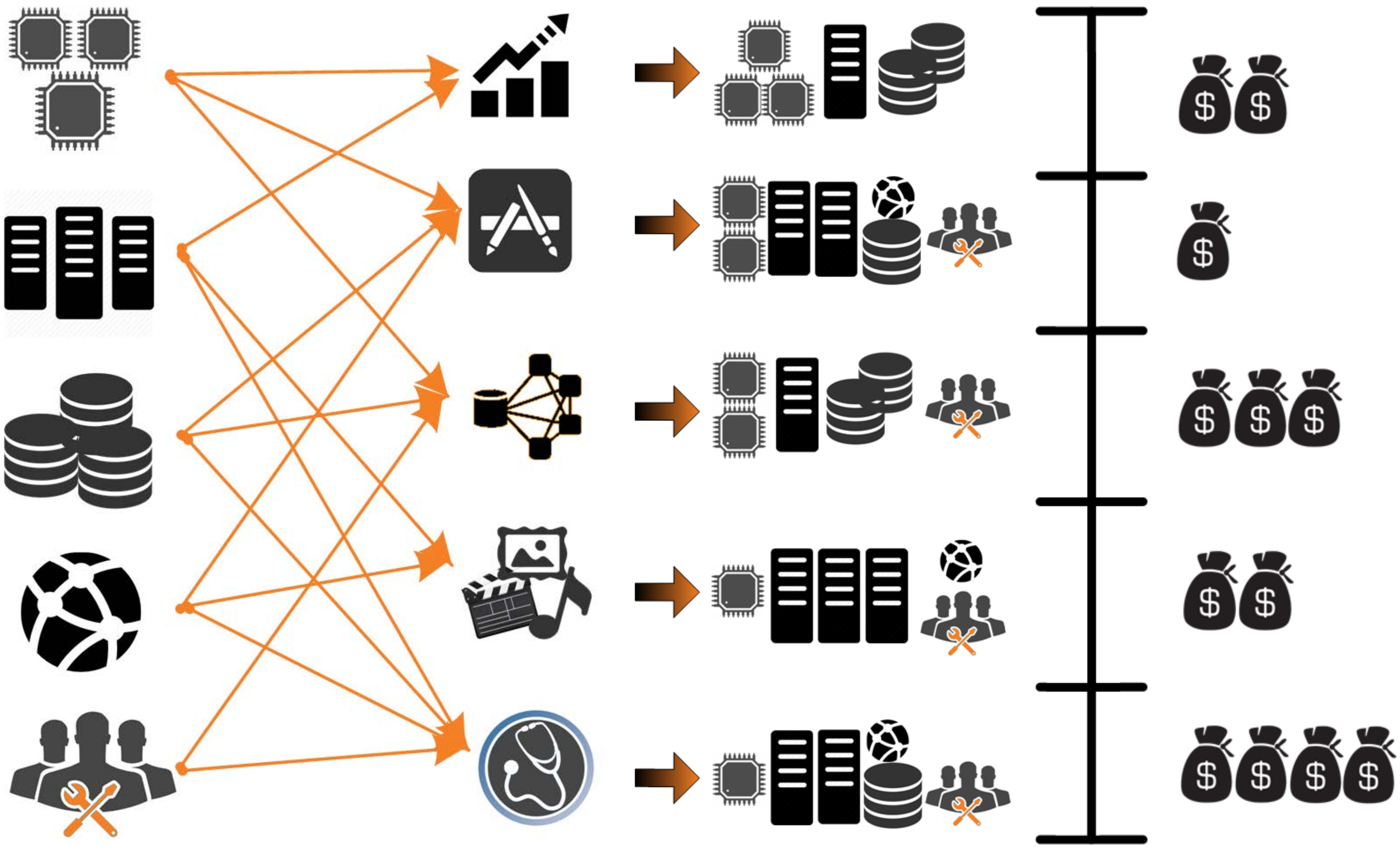
香港城市大學
City University of Hong Kong

Shenzhen University | Shenzhen | November 2015

Cloud Computing



Cloud Resource Allocation and Pricing



Cloud Pricing

Usage-based cloud pricing

Auction-based cloud pricing

Amazon's Elastic Compute Cloud (EC2) spot instances

Spot Instance

Step 1: Choose the instance type

Amazon Linux Free tier eligible	☐	Memory optimized	r3.large	2	15	1 x 32 (SSD)	-	Moderate
	☐	Memory optimized	r3.xlarge	4	30.5	1 x 80 (SSD)	Yes	Moderate
Red Hat Free tier eligible	☐	Memory optimized	r3.2xlarge	8	61	1 x 160 (SSD)	Yes	High
	☐	Memory optimized	r3.4xlarge	16	122	1 x 320 (SSD)	Yes	High
SUSE Linux Free tier eligible	☐	Memory optimized	r3.8xlarge	32	244	2 x 320 (SSD)	-	10 Gigabit

Step 2: Configure the instance details

– Number of instances & bid price

Number of instances (i) 1

Purchasing option (i) ☑ Request Spot Instances

Current price (i)

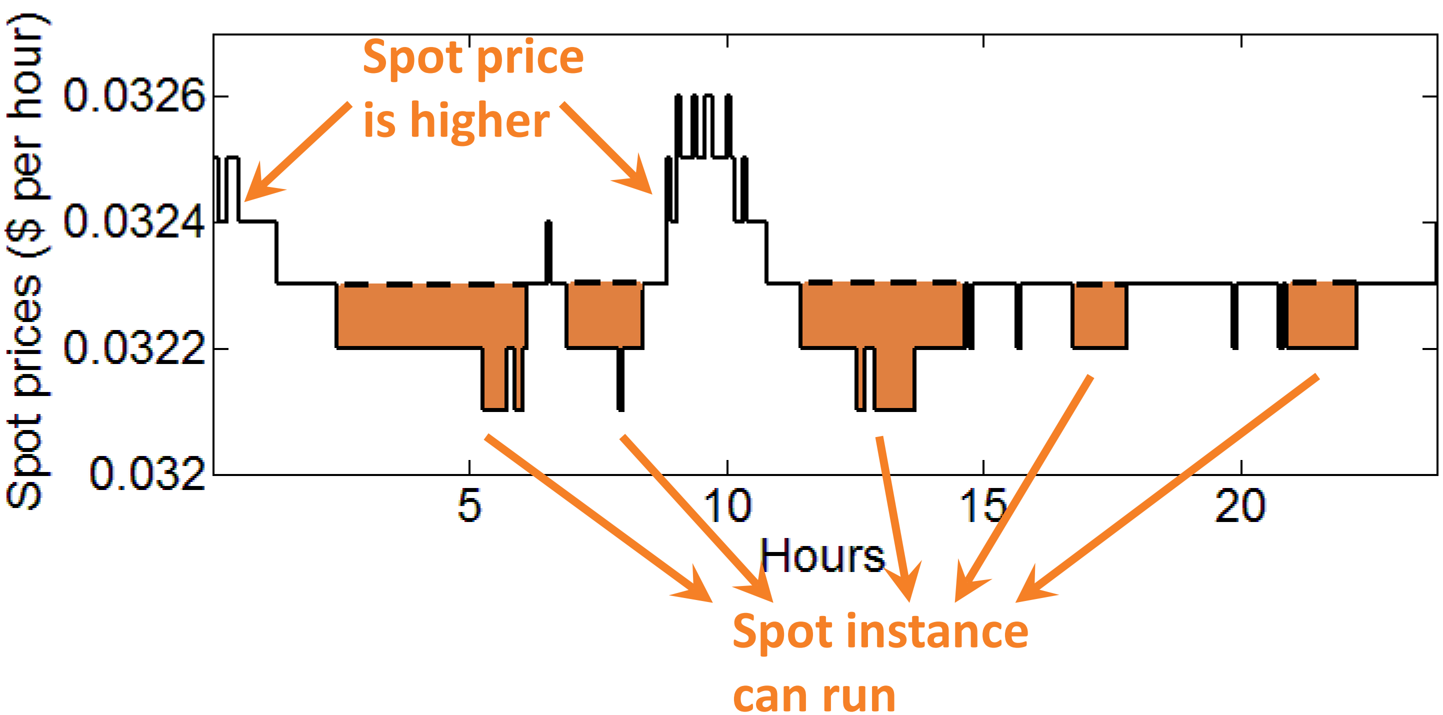
us-east-1a	0.1582
us-east-1b	0.1587
us-east-1d	0.1821
us-east-1e	0.1856

Maximum price (i) $ (e.g. 0.045 = 4.5 cents/hour)

Name the price YOURSELF!

Spot Pricing

Spot price history for an r3.xlarge instance in the US Eastern region on September 09, 2014



Our Questions

- Question #1

  How might the cloud provider set the price?

- Question #2

  What prices should users bid?

Our Questions

- Question #1

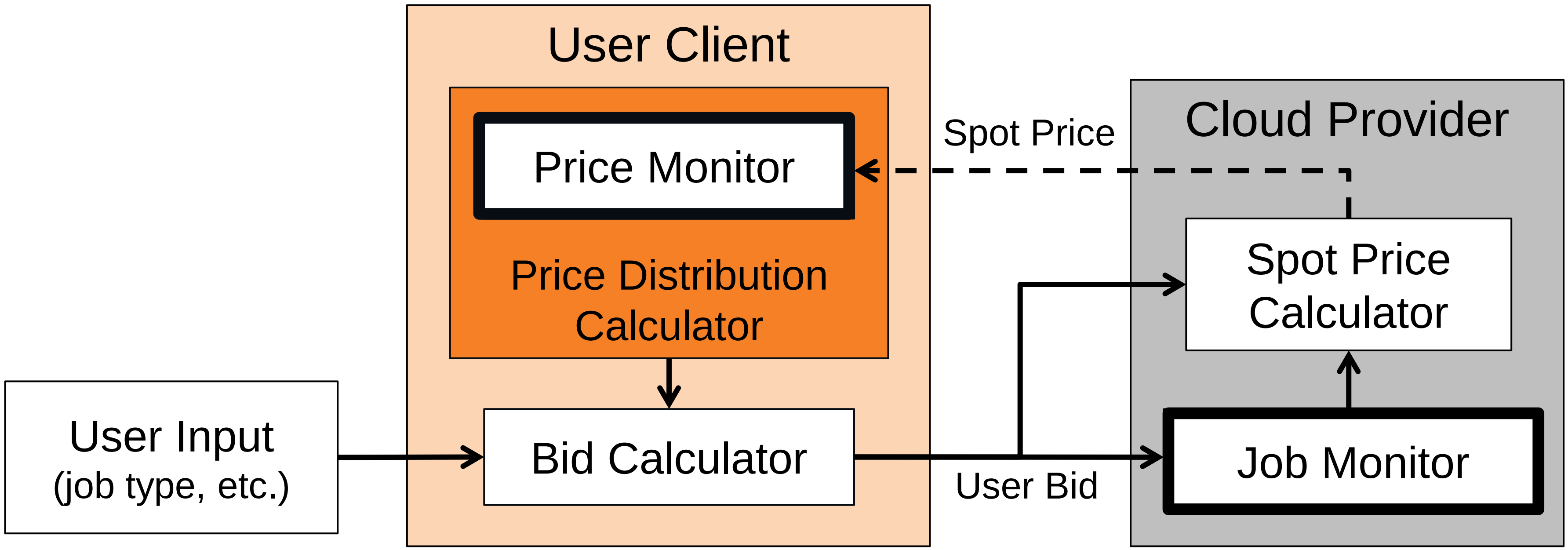
  How might the cloud provider set the price?

- Question #2

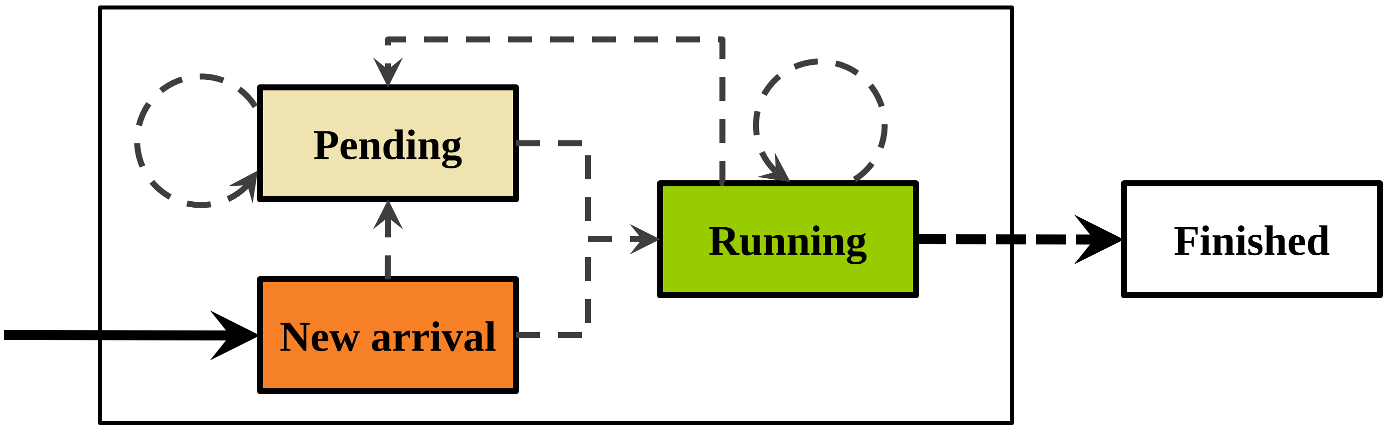
  What prices should users bid?

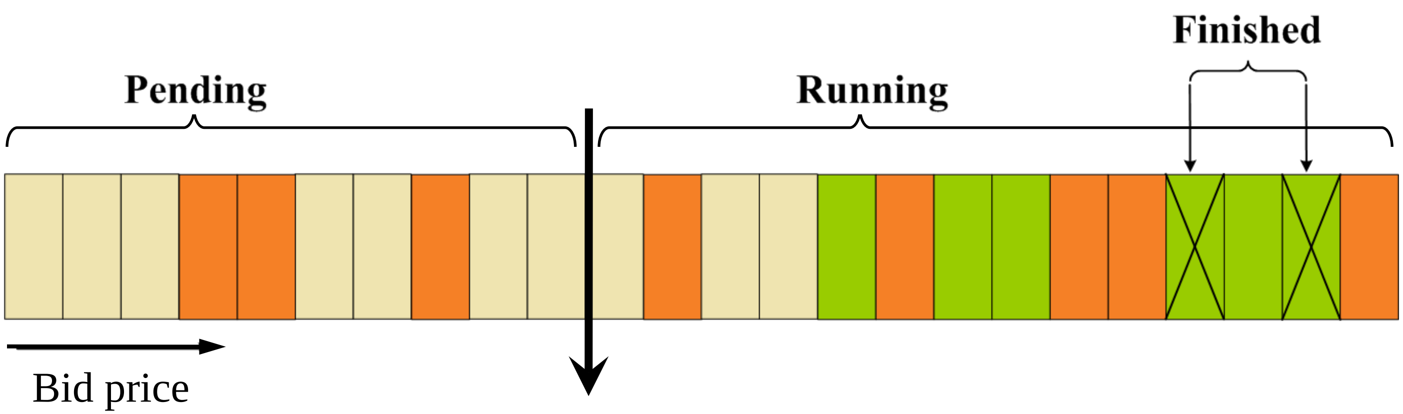
L. Zheng, C. Joe-Wong, C. W. Tan, M. Chiang & X. Wang, How to bid the cloud? **ACM SIGCOMM**, 2015

Our Solution



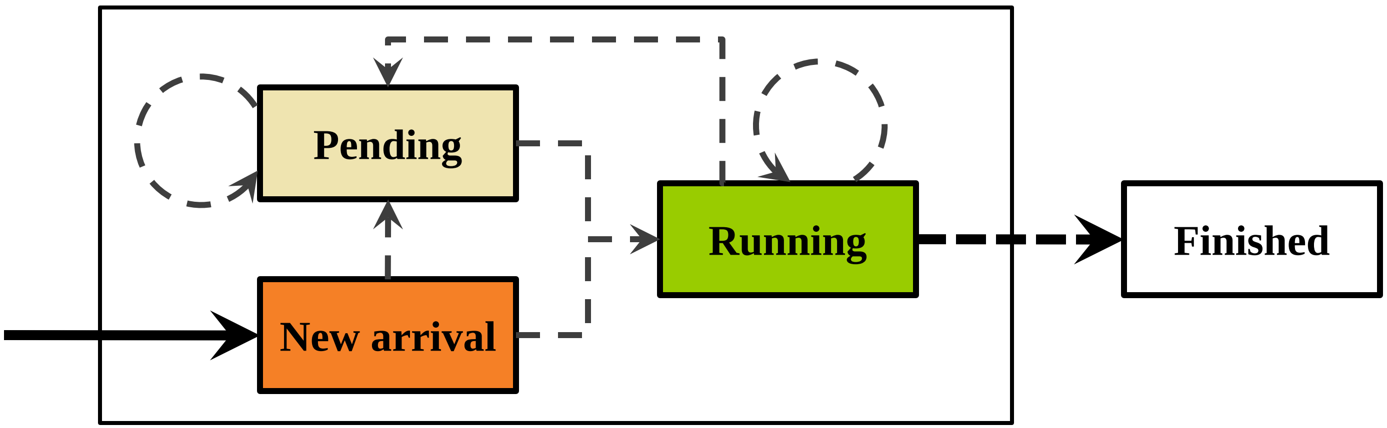
Cloud Provider Model

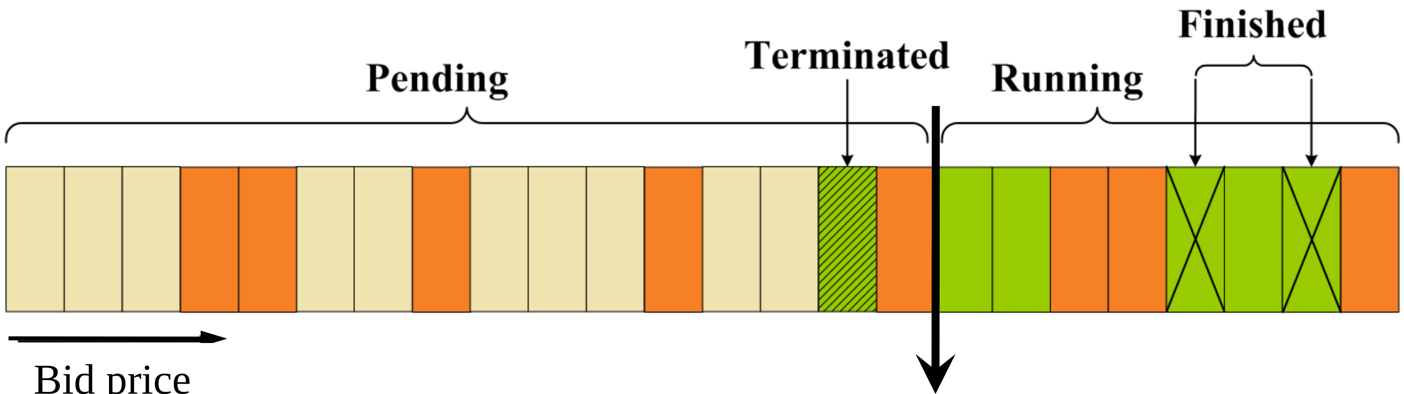




$$\pi^{\star}(t) = h\big(\Lambda(t)\big) = \frac{1}{2}\left(\bar{\pi} - \frac{\beta}{1 + \frac{1}{\theta}\Lambda(t)}\right)$$

Cloud Provider Model



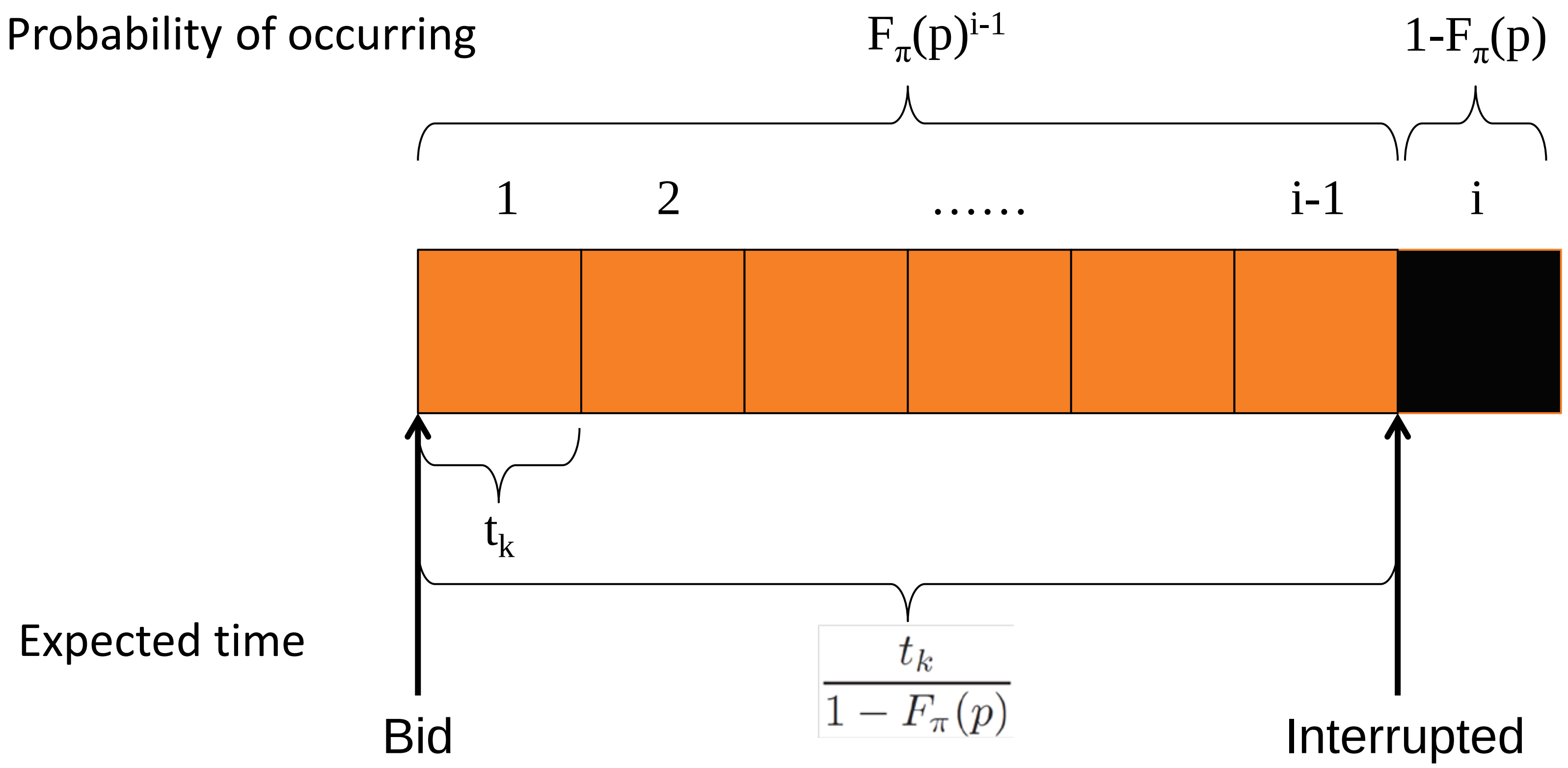


$$\pi^\star(t) = h\big(\Lambda(t)\big) = \frac{1}{2}\left(\bar{\pi} - \frac{\beta}{1 + \frac{1}{\theta}\Lambda(t)}\right)$$

Bid Types

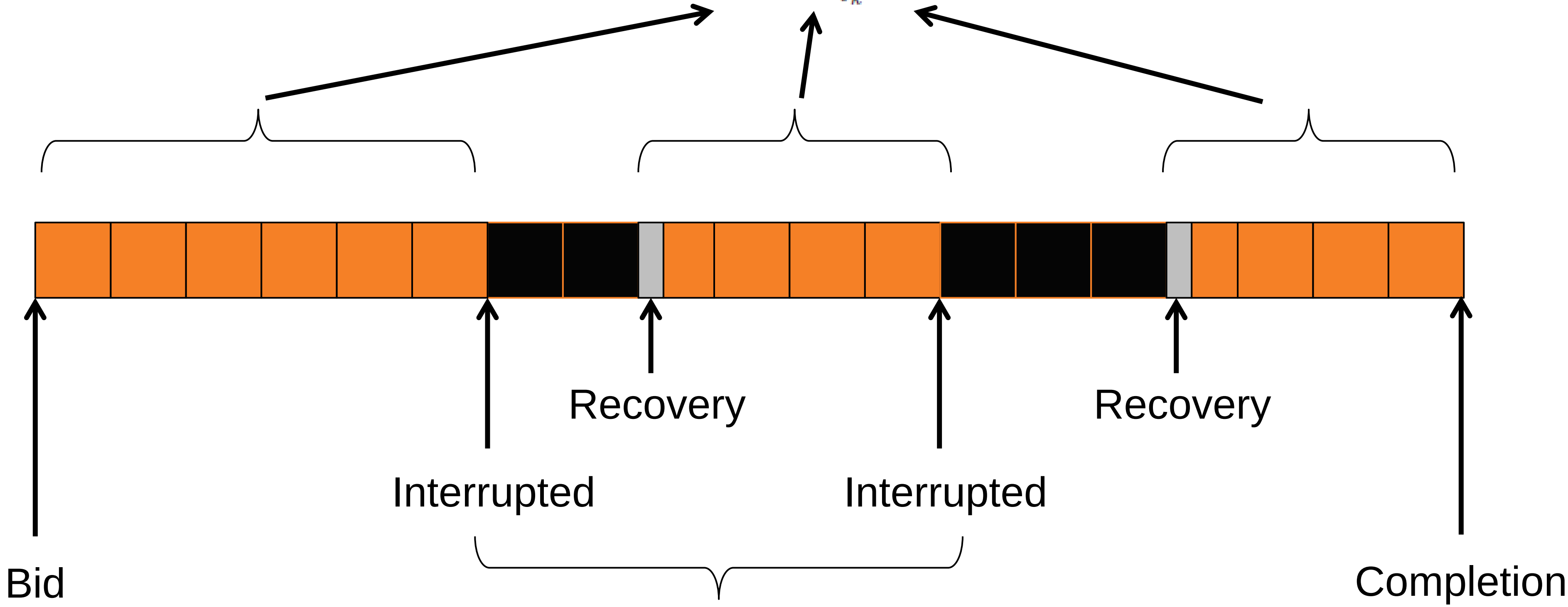
- One-time user bid
  - Submitted once and then exit the system once they fall below the current spot price.
  - Job interrupted without completing
- Persistent user bid
  - Resubmitted in each time period until the job finishes or is manually terminated by the user.
  - Longer waiting and completion time.

Placing One-time Bids



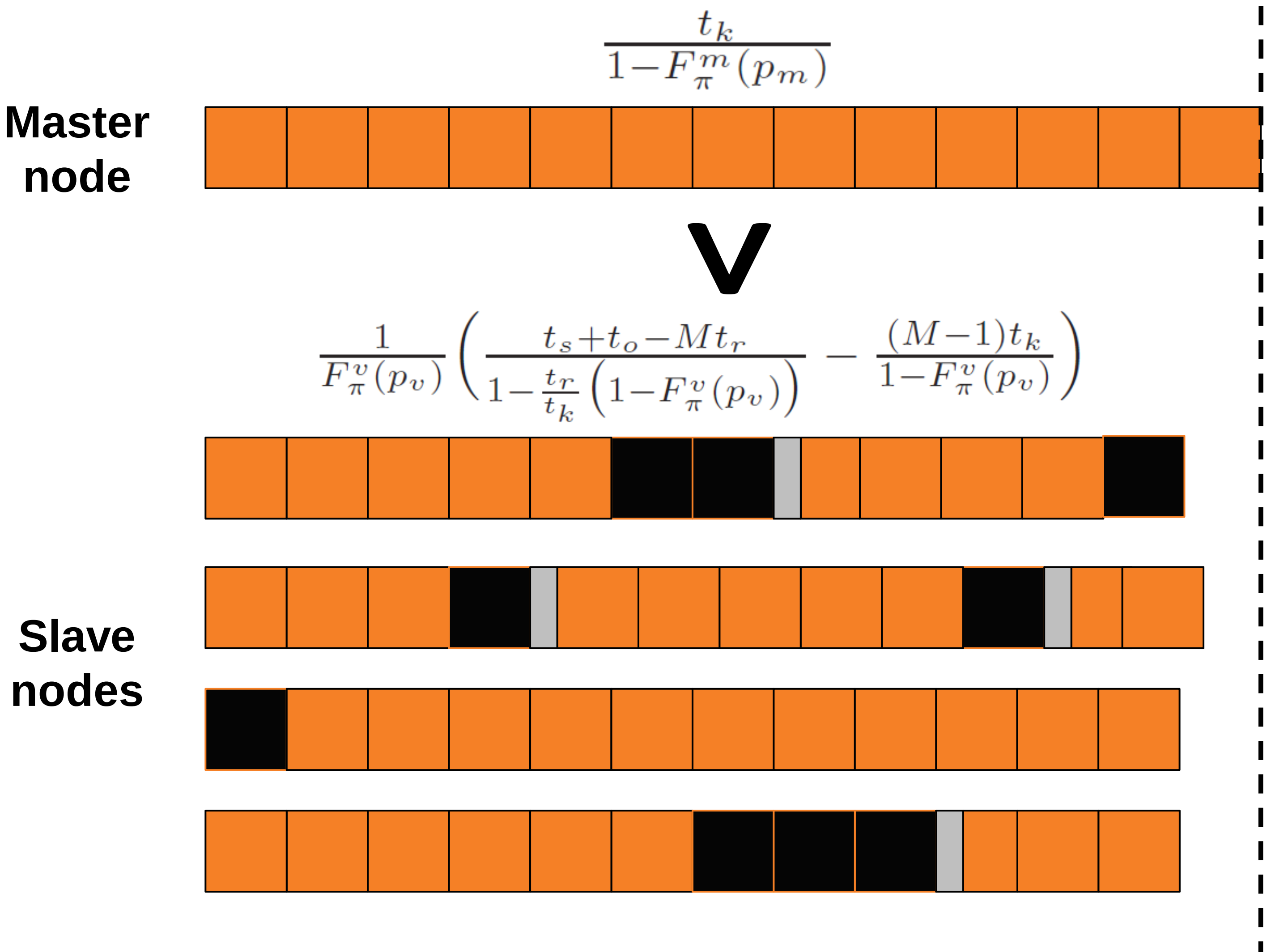
Placing Persistent Bids

$$TF_\pi(p) = \frac{t_s - t_r}{1 - \frac{t_r}{t_k}\left(1 - F_\pi(p)\right)}$$

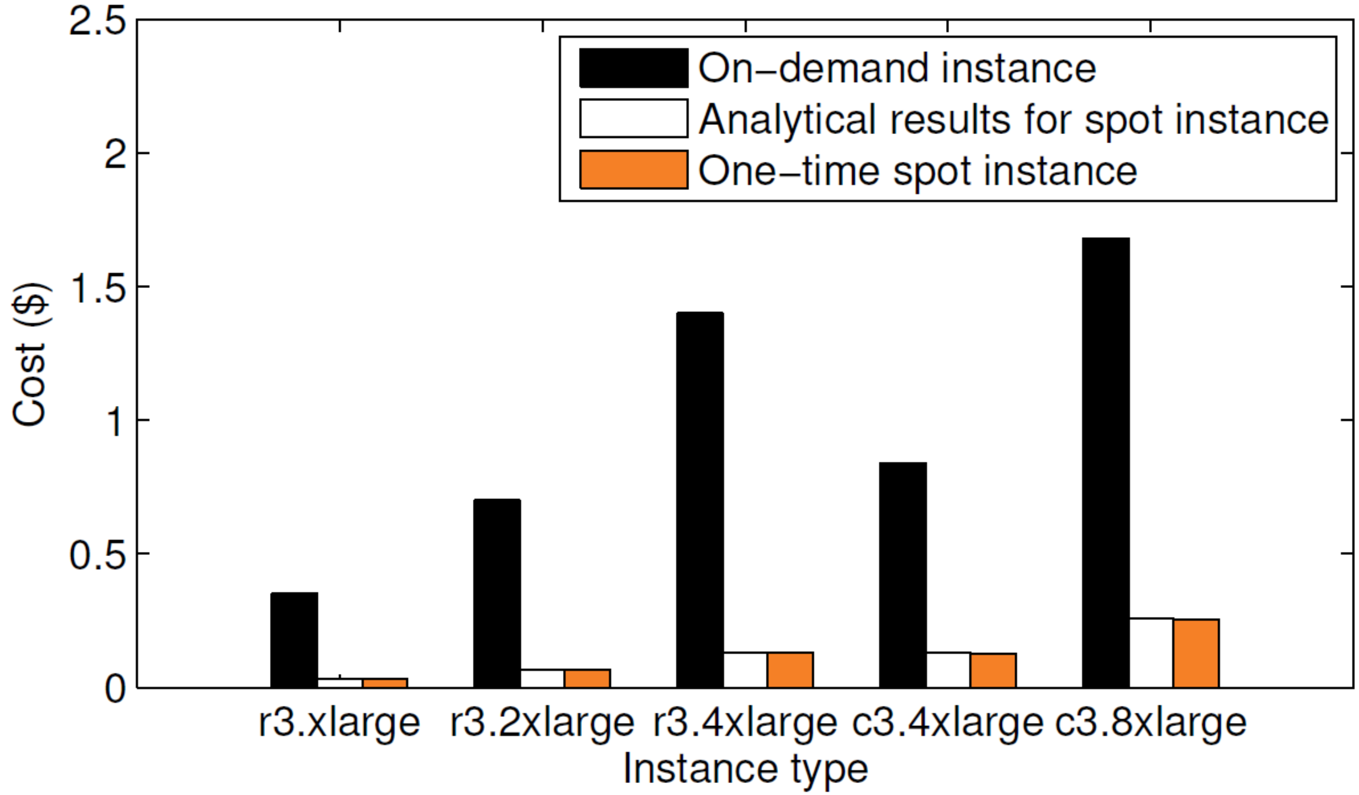


$$2 = \frac{T}{t_k} F_\pi(p)\left(1 - F_\pi(p)\right)$$

Bidding MapReduce Jobs



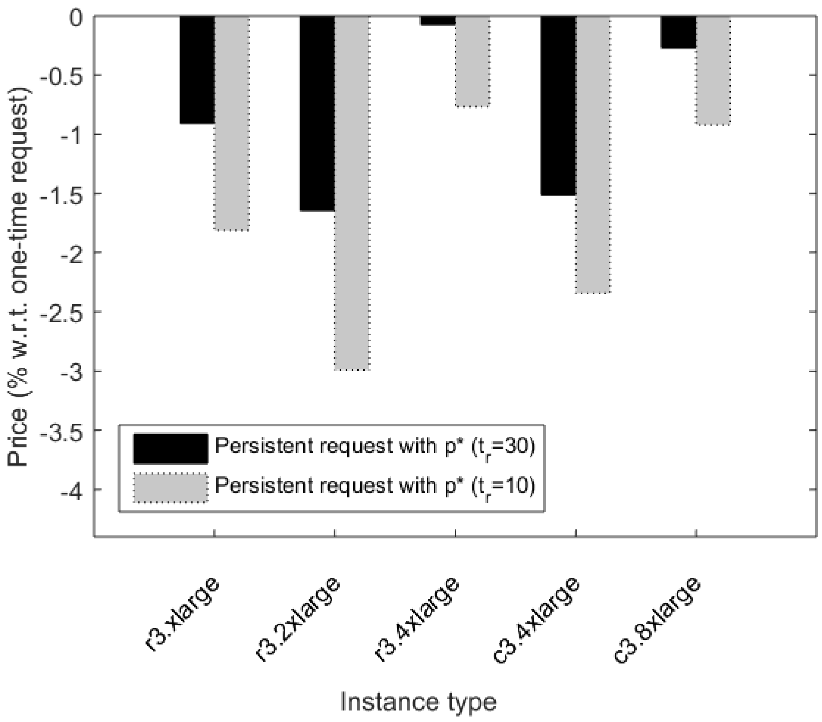
One-time Bids

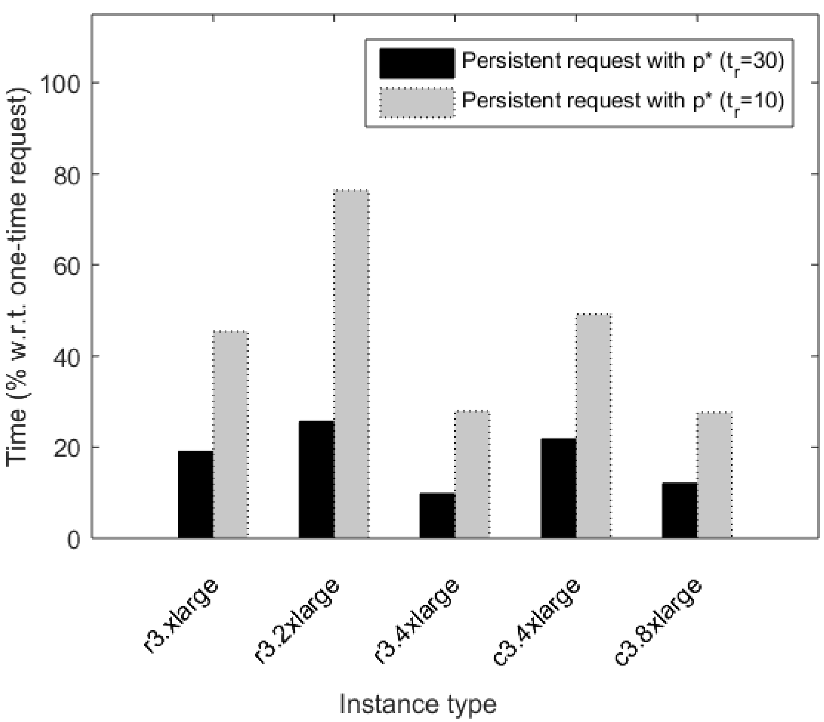


User costs are reduced by up to 91%, without any interruptions.

Persistent Bids

$$\frac{\text{bid price (time) of persistent bids} - \text{bid price (time) of one-time bids}}{\text{bid price (time) of one-time bids}}$$

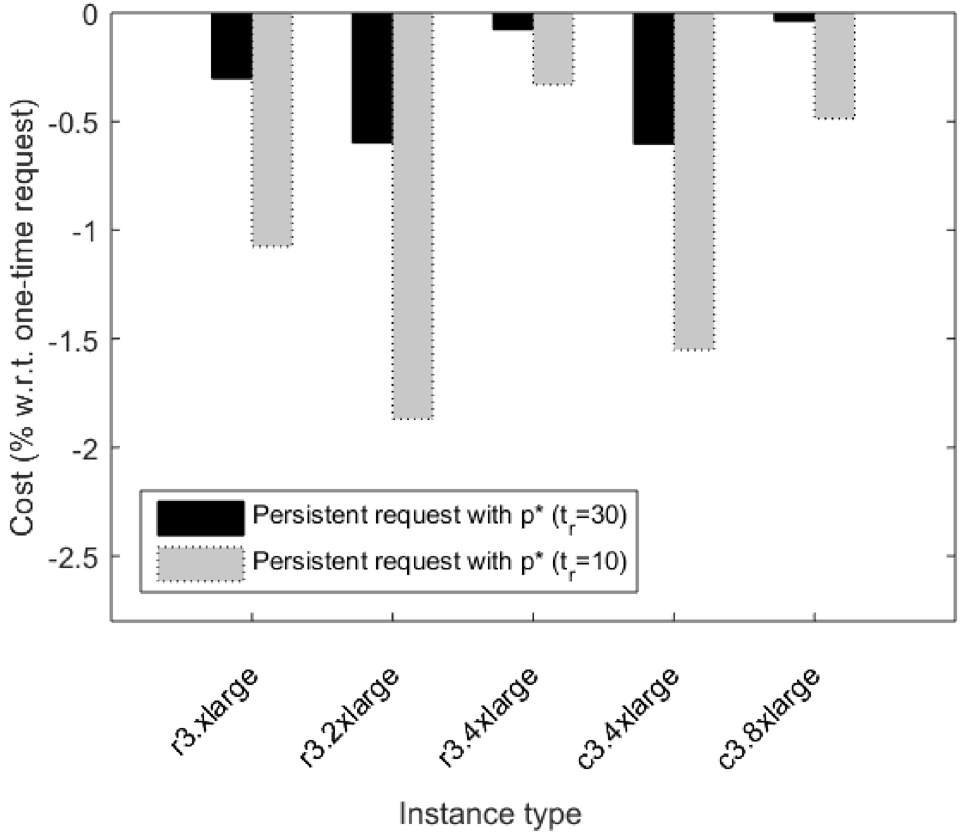




A lower optimal bid price.

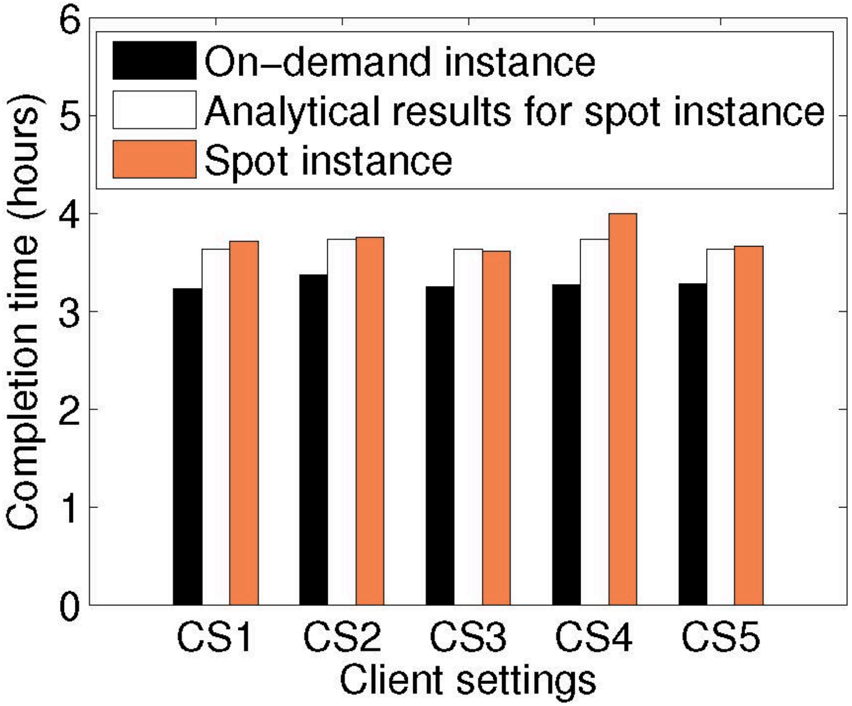
A longer completion time.

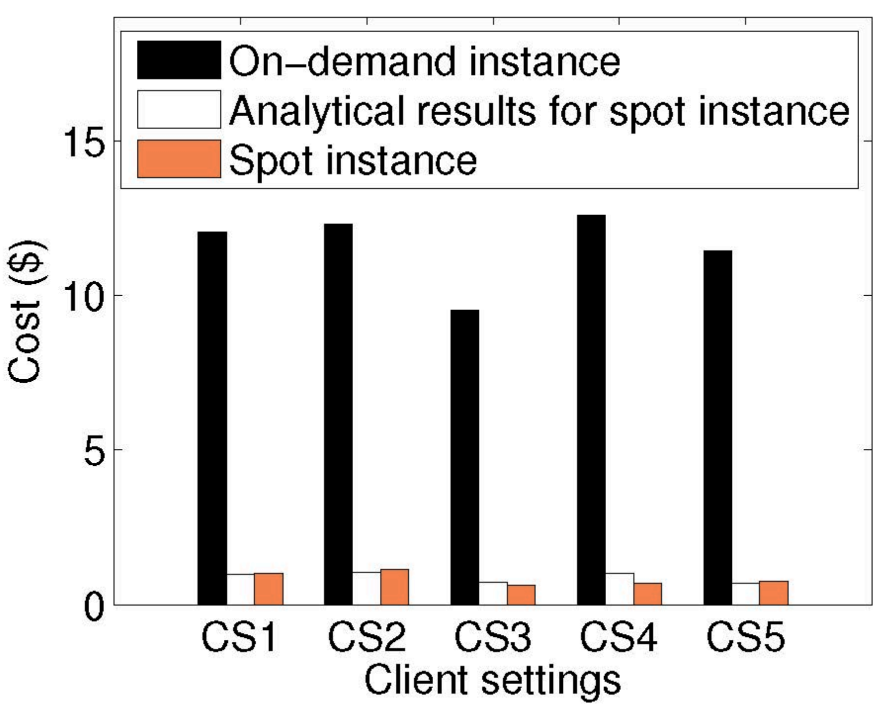
Persistent Bids



The overall costs are further reduced.

MapReduce Jobs

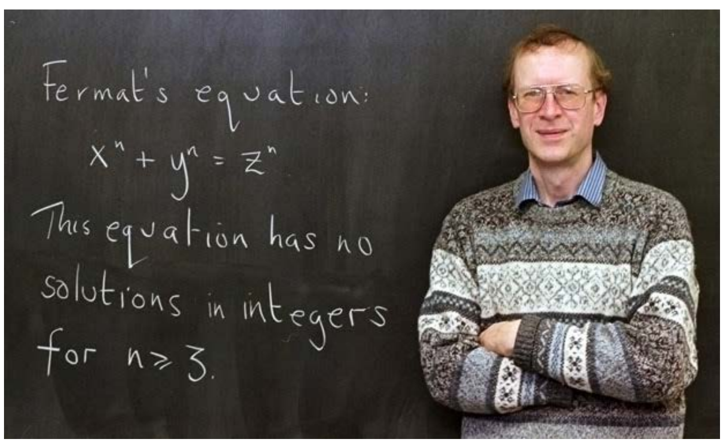


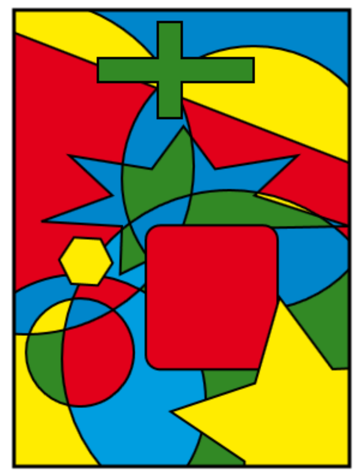


The cost is reduced by up to 92.6% with just a 14.9% completion time increase.

Scientific-Computing-As-A-Service

- Scientific computing is the holy grail of cloud computing
- Major recent breakthroughs in scientific discovery
  - Four Color Theorem
  - Fermat's Last Theorem





How to make bucks selling Theorems?!

Introduction

- Proving an information inequality is a crucial step in establishing the converse results in coding theorems.
- An information inequality involving many random variables is difficult to be proved manually.

- [Yeung 1997] developed a framework that uses linear programming for verifying linear information inequalities.
- Under this framework, our paper considers a few other problems that can be solved by using Lagrange duality and convex approximation.
- S. Ho, C. W. Tan and R. W. Yeung, Proving and disproving information inequalities, IEEE ISIT 2014

To Prove:

- ITIP("H(U) <= H(R)", "I(U;X) = 0"; "H(U|RX) = 0")
- True. The inequality follows from
- H(U) + H(R) = (-H(U,X) + H(U,R,X)) + (H(R) + H(X) - H(R;X)) +

  {-H(U) - H(X) + H(U;X)} + {H(R;X) - H(U;R;X)}

  >= 0;
- where (·) is non-negative as it is either conditional entropy or conditional mutual information. All {·} are equal to 0 due to the given constraints. Equality holds iff all (·) are equal to 0.

To Disprove:

- ITIP("I(A;B|CD) + I(B;D|AC) <= I(A;B|D) + I(B;D|A) + H(A) + I(B;D|C)")
- Not provable by ITIP.
- It can be disproved by a probability distribution satisfying all the following Shannon's information measures equal to zero:

H(A|B,C,D); H(C|A,B,D), H(D|A,B,C), I(A;B|C), I(A;B|D), I(A;C|D), I(A;D), I(B;C|A), I(B;D|A), I(B;D|C), I(C;D|A).

From the above output from ITIP, we can deduce the following counterexample. Let X; Y and Z be three independent binary random variables with entropy equal to 1.

Let $(A, B, C, D) = (X \oplus Y, X, Y \oplus Z, Z)$.

$I(A;B|D) + I(B;D|A) + H(A) + I(B;D|C) - I(A;B|CD) - I(B;D|CA)$

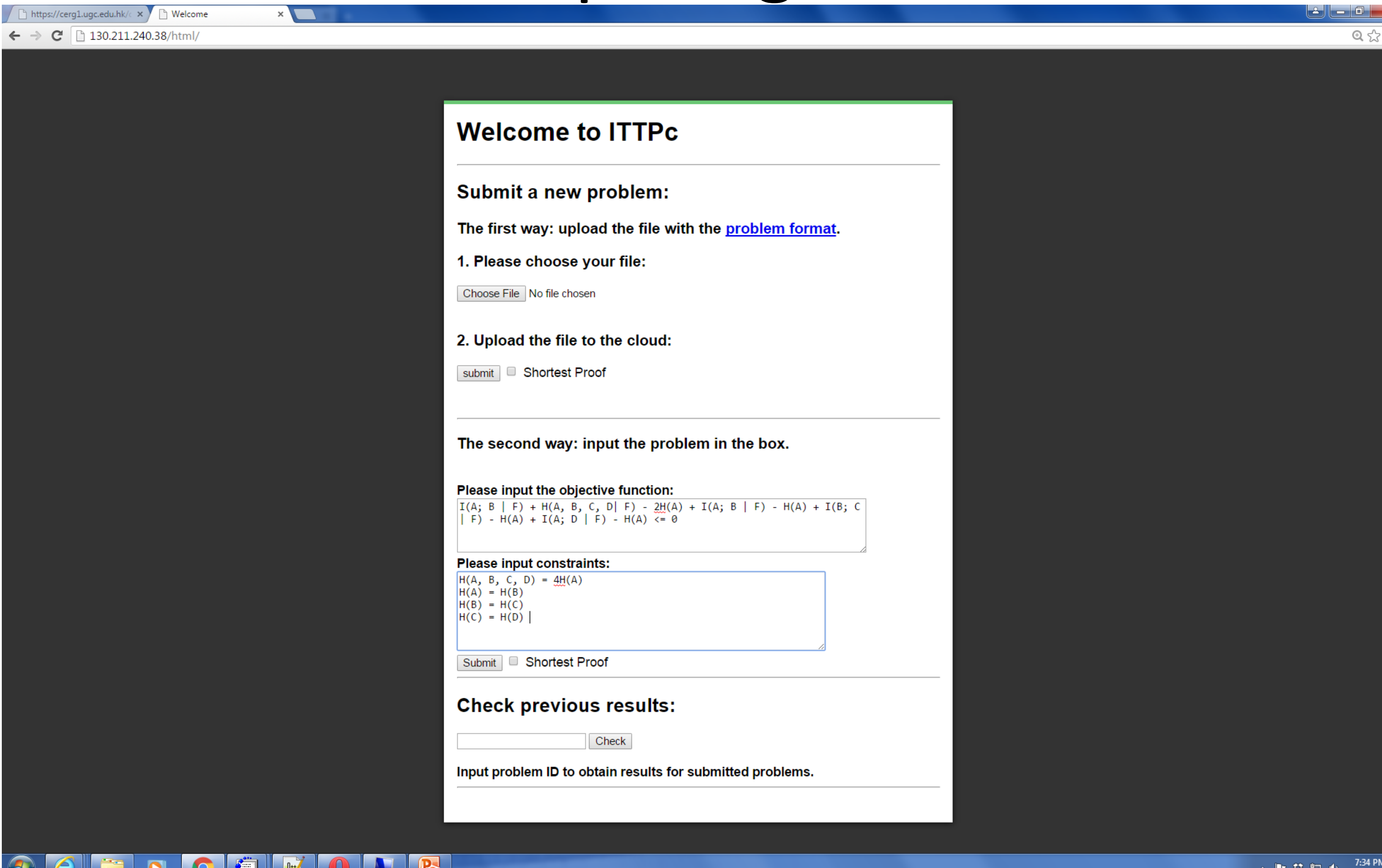
$= -1 < 0.$

Scale Up Computation

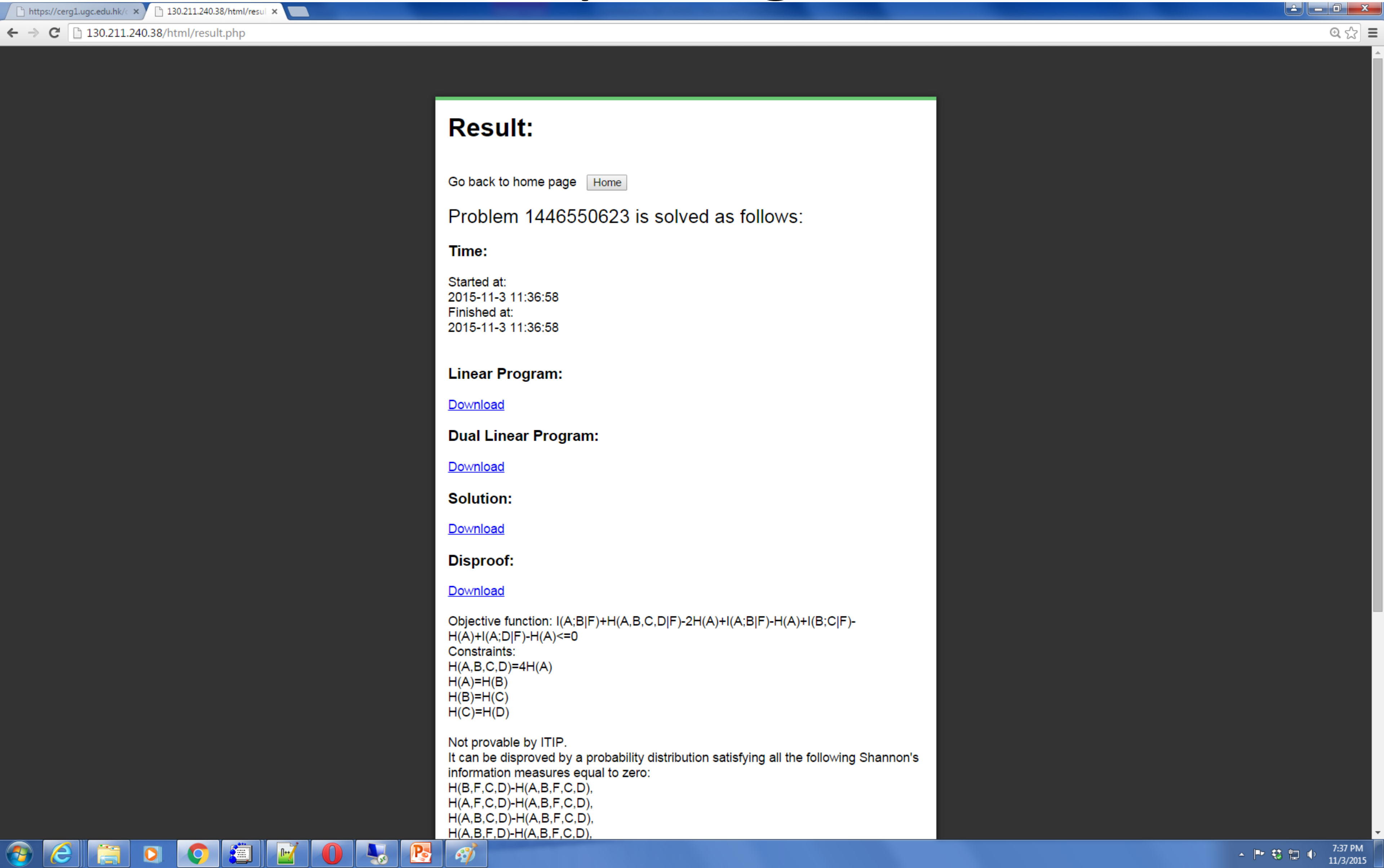
- Automation of Information Theory Prover
  - Computational algorithms (optimization based)
  - Cloud-computing (scale-up no. of random variables)

- Distributed data storage and security applications
  - Secured Storage Code
    - Privacy of data by information leakage to eavesdroppers during repair
    - Hybrid repair (gap with functional repair), when
  - Storage Network Topology Optimization
    - All prior work assumes a complete connectivity topology for storage network, however practical networks have different communication capacities and diverse (sparse) network topology
  - Information-theoretic Security and Network Security
    - Application-specific equality constraints in LP most interesting (Markov chain, security)

Scientific-Computing-As-A-Service



Scientific-Computing-As-A-Service



Scientific-Computing-As-A-Service

IEEE TRANSACTIONS ON

INFORMATION THEORY

A Journal Devoted to the Theoretical and Experimental Aspects of Information Transmission, Processing, and Utilization

JULY 1999 VOLUME 45 NUMBER 5 IETTAW (ISSN 0018–9448)

PAPERS

Conclusion

- Model for cloud provider's setting of the spot prices.
- Bidding strategies: Tradeoff between prices and times
  - One-time bids: bidding higher prices to avoid interruptions .
  - Persistent bids: bidding lower prices to save money.
- Application to the MapReduce jobs.
- Temporal correlations, risk-awareness, task dependence, collective user behavior, etc.
- Scaling up Scientific Computing as a Service

Thank you!

- L. Zheng, C. Joe-Wong, C. W. Tan, M. Chiang & X. Wang, How to bid the cloud? **ACM SIGCOMM**, 2015
- S. Ho, C. W. Tan and R. W. Yeung, Proving and disproving information inequalities, IEEE ISIT 2014

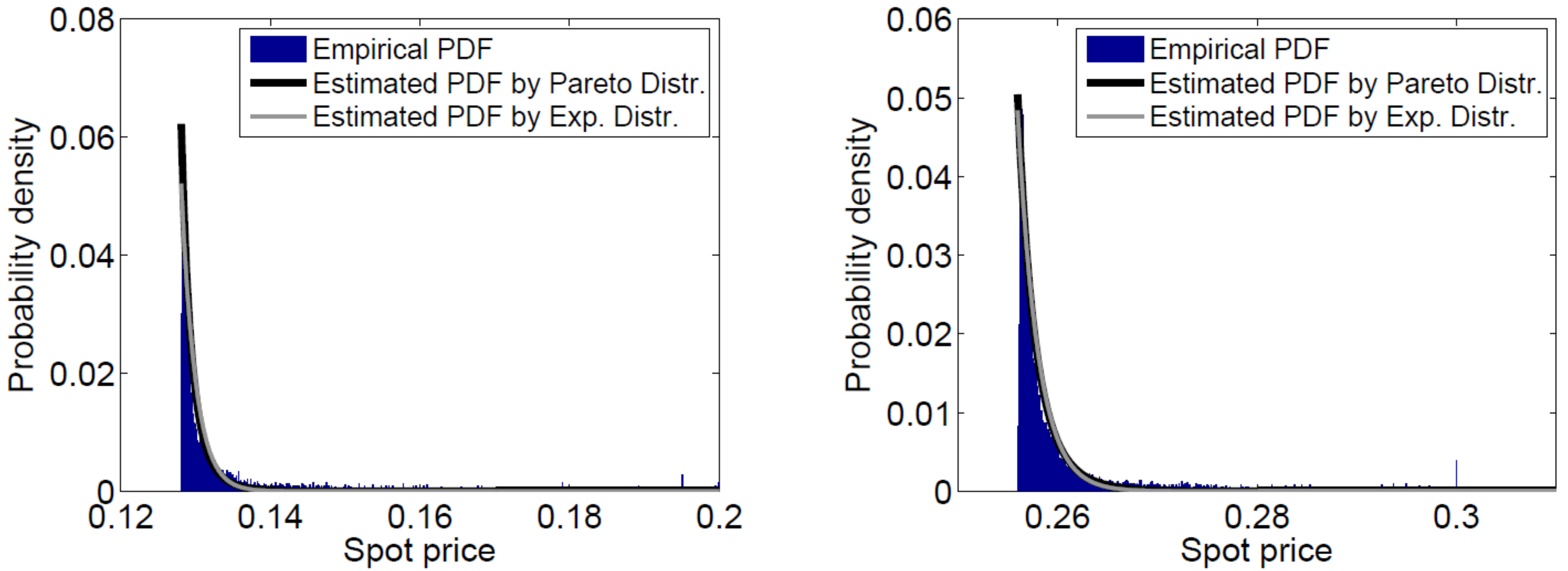
Backup Slides

Cloud provider revenue maximization.

$$
\begin{aligned}
\underset{\pi(t)}{\text{maximize}} \quad & \beta \log\left(1 + L(t)\frac{\bar{\pi} - \pi(t)}{\bar{\pi} - \underline{\pi}}\right) \\
& +\pi(t)L(t)\frac{\bar{\pi} - \pi(t)}{\bar{\pi} - \underline{\pi}} \\
\text{subject to} \quad & \underline{\pi} \le \pi(t) \le \bar{\pi}.
\end{aligned}
$$

$$
\pi^\star(t) = h(\Lambda(t)) = \frac{1}{2}\left(\bar{\pi} - \frac{\beta}{1 + \frac{1}{\theta}\Lambda(t)}\right)
$$

Validation from Historical Spot Prices



Real-life Spot Instance Example

Mozilla Amazon EC2 usage got cheaper by using spot instance with fixed-price bidding in 2013-2014.

September: cost-effective m3.xlarge on-demand instances.
October: m3.xlarge spot instances with unexpected interruptions.
December: upgrading to use cheaper c3.xlarge on-demand instances.
February: a mix of c3.xlarge on-demand and m3.xlarge spot instances.
March: the majority of workload switched to spot instances.
April: Amazon further drops its spot prices.

Source: http://taras.glek.net/blog/2014/05/09/how-amazon-ec2-got-15x-cheaper-in-6-months/

EC2 Instance Types

	Balanced			Memory-optimized			Compute-optimized		
	m3			r3			c3		
	vCPU	Memory	Storage	vCPU	Memory	Storage	vCPU	Memory	Storage
.xlarge	4	15	1x32	4	30.5	1x80	4	7.5	2x40
.2xlarge	8	30	2x80	8	61	1x160	8	15	2x80
.4xlarge	----	----	----	16	122	1x320	16	30	2x160
.8xlarge	----	----	----	----	----	----	32	60	2x320

Single-Instance One-time Bids

Optimal bid prices for one-time bids that run for one hour.

Instance type	On-demand price	One-time bid	
		Optimal price	Actual price
r3.xlarge	$0.35	$0.0374	$0.033
r3.2xlarge	$0.70	$0.0795	$0.066
r3.4xlarge	$1.40	$0.1430	$0.130
c3.4xlarge	$0.84	$0.1669	$0.128
c3.8xlarge	$1.68	$0.2903	$0.256

Single-Instance One-time Bids

Optimal bid prices for one-time bids that run for one hour.

Instance type	On-demand price	One-time bid		
		Optimal price	Offline retrospective price	Actual price
r3.xlarge	$0.35	$0.0374	$0.0324	$0.033
r3.2xlarge	$0.70	$0.0795	$0.0644	$0.066
r3.4xlarge	$1.40	$0.1430	$0.1304	$0.130
c3.4xlarge	$0.84	$0.1669	$0.1324	$0.128
c3.8xlarge	$1.68	$0.2903	$0.2640	$0.256

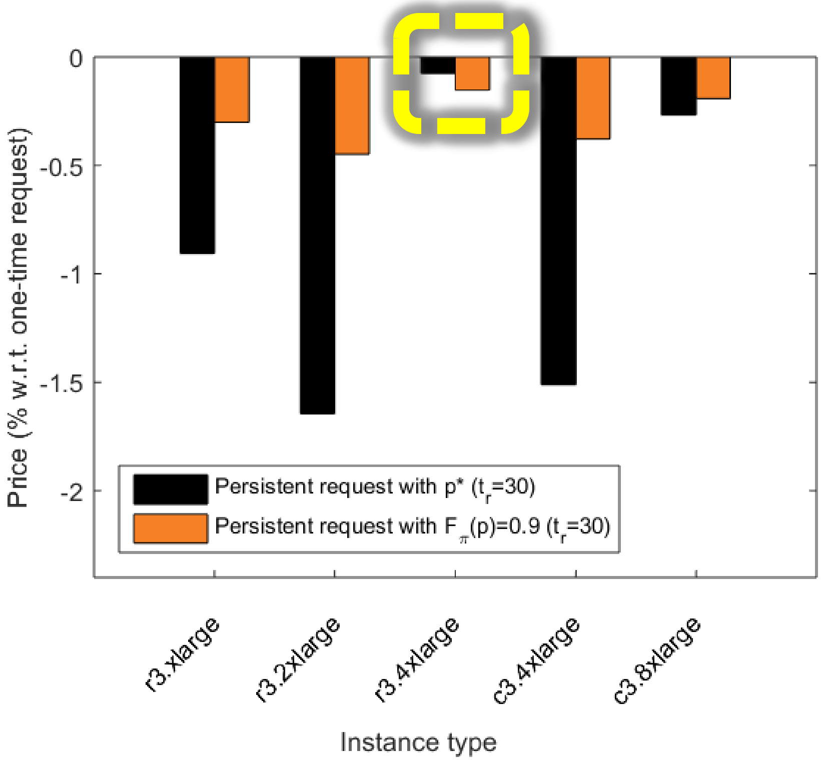
Single-Instance Persistent Bids

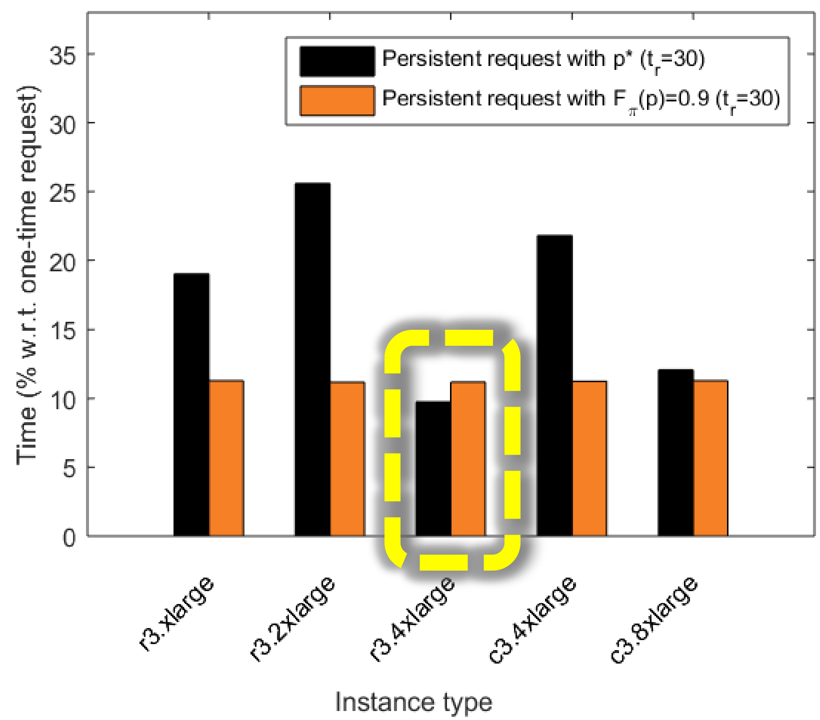
Optimal bid prices with different recovery times.

Instance type	On-demand price	Persistent bid	
		Optimal price (t_r=10s)	Optimal price (t_r=30s)
r3.xlarge	\$0.35	\$0.0332	\$0.0355
r3.2xlarge	\$0.70	\$0.0661	\$0.0711
r3.4xlarge	\$1.40	\$0.1327	\$0.1422
c3.4xlarge	\$0.84	\$0.1322	\$0.1413
c3.8xlarge	\$1.68	\$0.2648	\$0.2831

Longer recovery times yield higher bid prices.

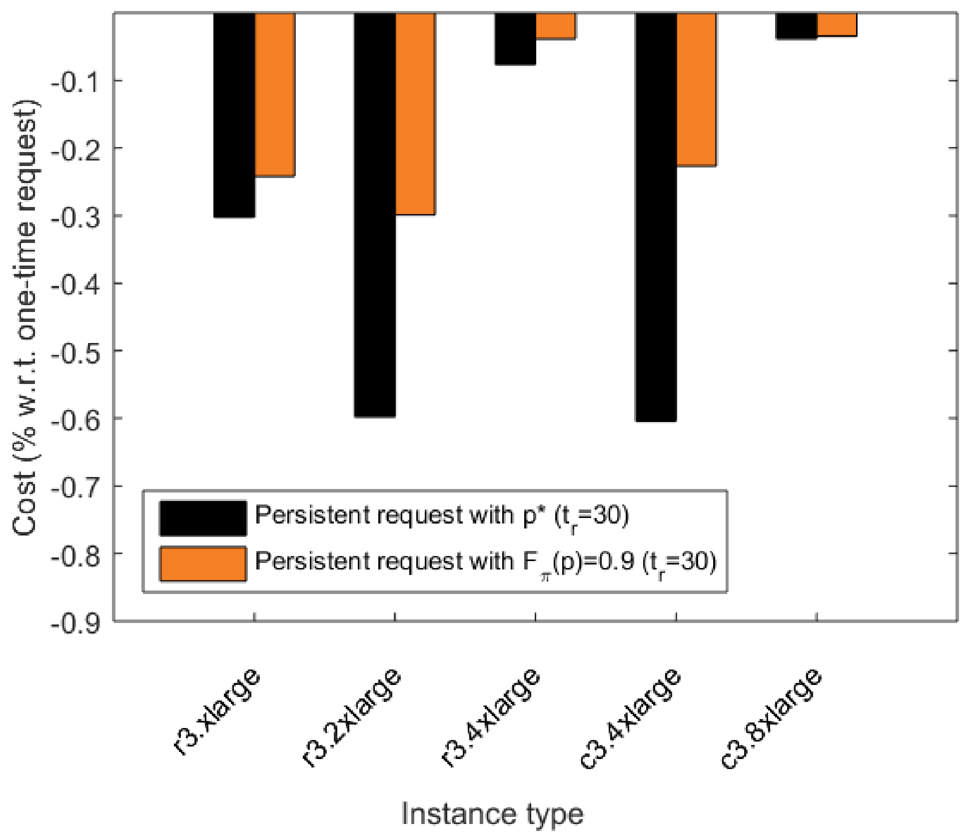
Single-Instance Persistent Bids





Bidding at the 90th percentile price yields either higher bid prices and lower completion times or lower bid prices and longer completion times.

Single-Instance Persistent Bids



Our bid prices are optimal for minimizing users' costs.

MapReduce Jobs

Optimal bid prices and actual costs for a MapReduce job.

Client Setting	Master node			Slave nodes			
	Instance Type	Bid prices	Actual cost	Instance type	Bid prices	Node numbers	Actual cost
CS1	c3.xlarge	$0.133	$0.10	m3.2xlarge	$0.070	5	$0.90
CS2	m3.xlarge	$0.101	$0.13	m3.2xlarge	$0.070	5	$1.03
CS3	m3.xlarge	$0.102	$0.13	r3.2xlarge	$0.071	3	$0.51
CS4	r3.xlarge	$0.042	$0.13	m3.2xlarge	$0.070	5	$0.58
CS4	r3.xlarge	$0.042	$0.13	r3.2xlarge	$0.071	3	$0.64