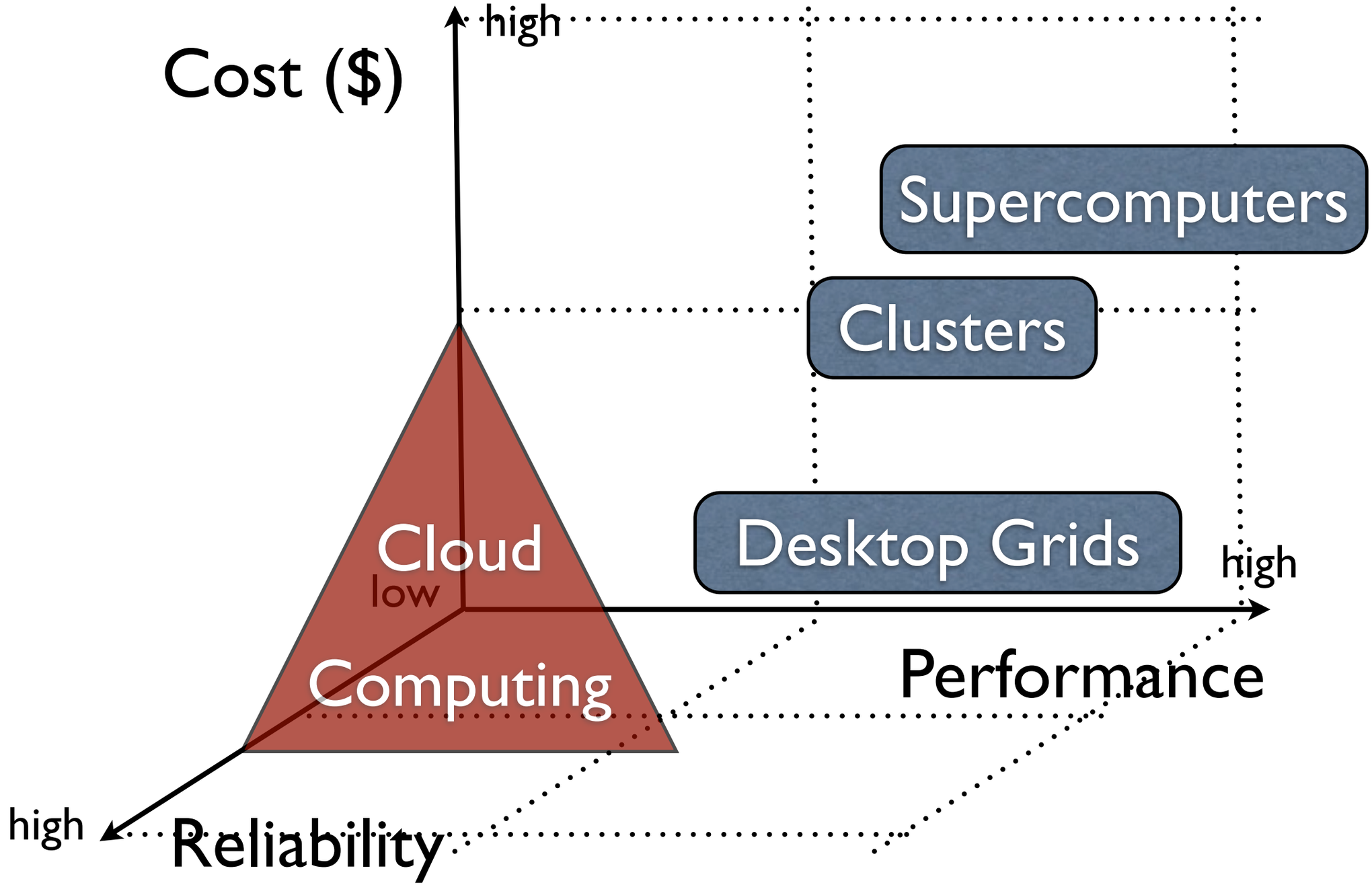
Decision Model for Cloud Computing under SLA Constraints

Artur Andrzejak[1], Derrick Kondo[2], Sangho Yi[2]

[1]Zuse Institute Berlin, but now at Institute for Infocomm Research (I2R), Singapore

[2]INRIA Grenoble, France

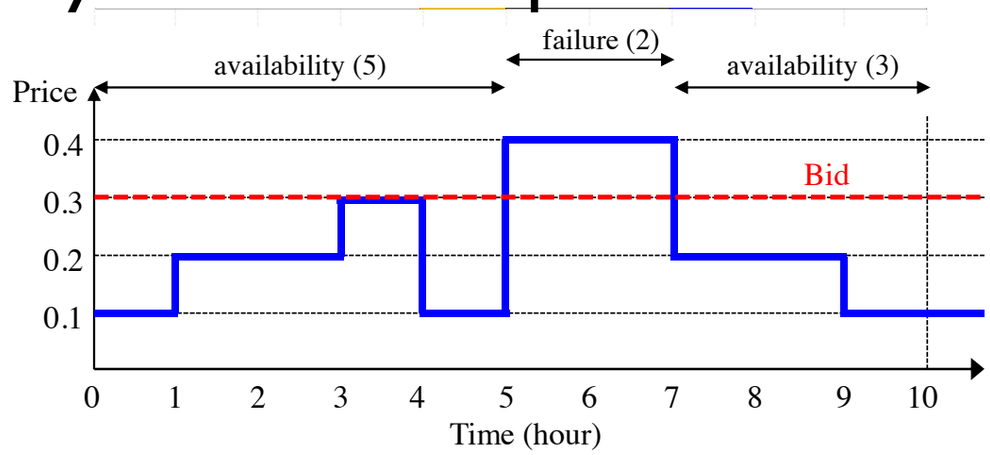
Trade-offs


Cost ($)
high
Supercomputers
Clusters
Desktop Grids
Cloud
low
Computing
high
Performance
high
Reliability

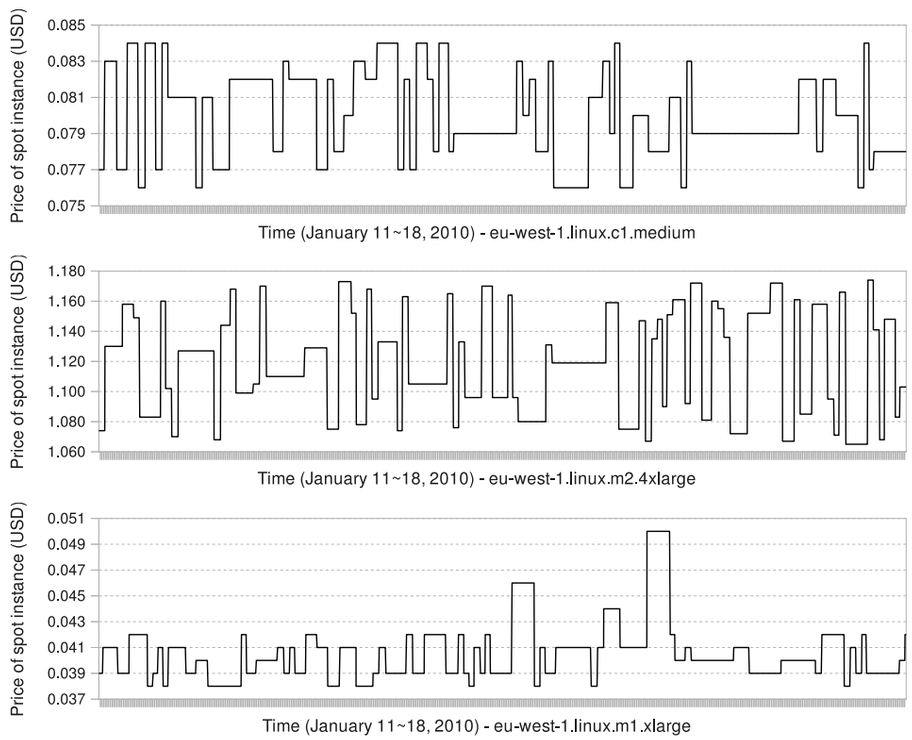
Market-based Resource Allocation Systems

- Amazon Spot Instances
- "Spot" instance price varies dynamically
- Spot instance provided when user's bid is greater than current price
- Spot instance terminated when user's bid ≤ current price
- Amazon charges by the last price at each hour

Synthetic Example:



Real Amazon Price Trace:



cloudexchange.org [tim lossen]

Optimization Problem

- Given job with batch of parallel, independent, divisible tasks
  - Deadline and budget constraints
- Objectives
  - Can the job be executed under budget and deadline constraints?
  - What is the bid price and instance type that minimizes the total monetary costs?
  - What is the distribution of monetary costs and execution times for a specific instance type and bid price?

Goal and Approach

- Formulate and show how to apply user decision model
- Characterize relationship between job execution time, monetary cost, reliability, bid price
- Compare costs of different instance types

Outline

- System model
- Decision model
- Simulations method and results
- Related work
- Conclusion & Future work

User Parameters and Constraints

Notation	Description
n_{inst}	number of instances that process the work in parallel
n_{max}	upper bound on n_{inst}
W	total amount of work in the user's job
W_{inst}	workload per instance (W/n_{inst})
T	task length, time to process W_{inst} on a specific instance
B	budget per instance
c_B	user's desired confidence in meeting budget B
t_{dead}	deadline on the user's job
c_{dead}	desired confidence in meeting job's deadline
u_b	user's bid on a Spot Instance type
I_{type}	EC2 instance type

Job parameters

Job constraints

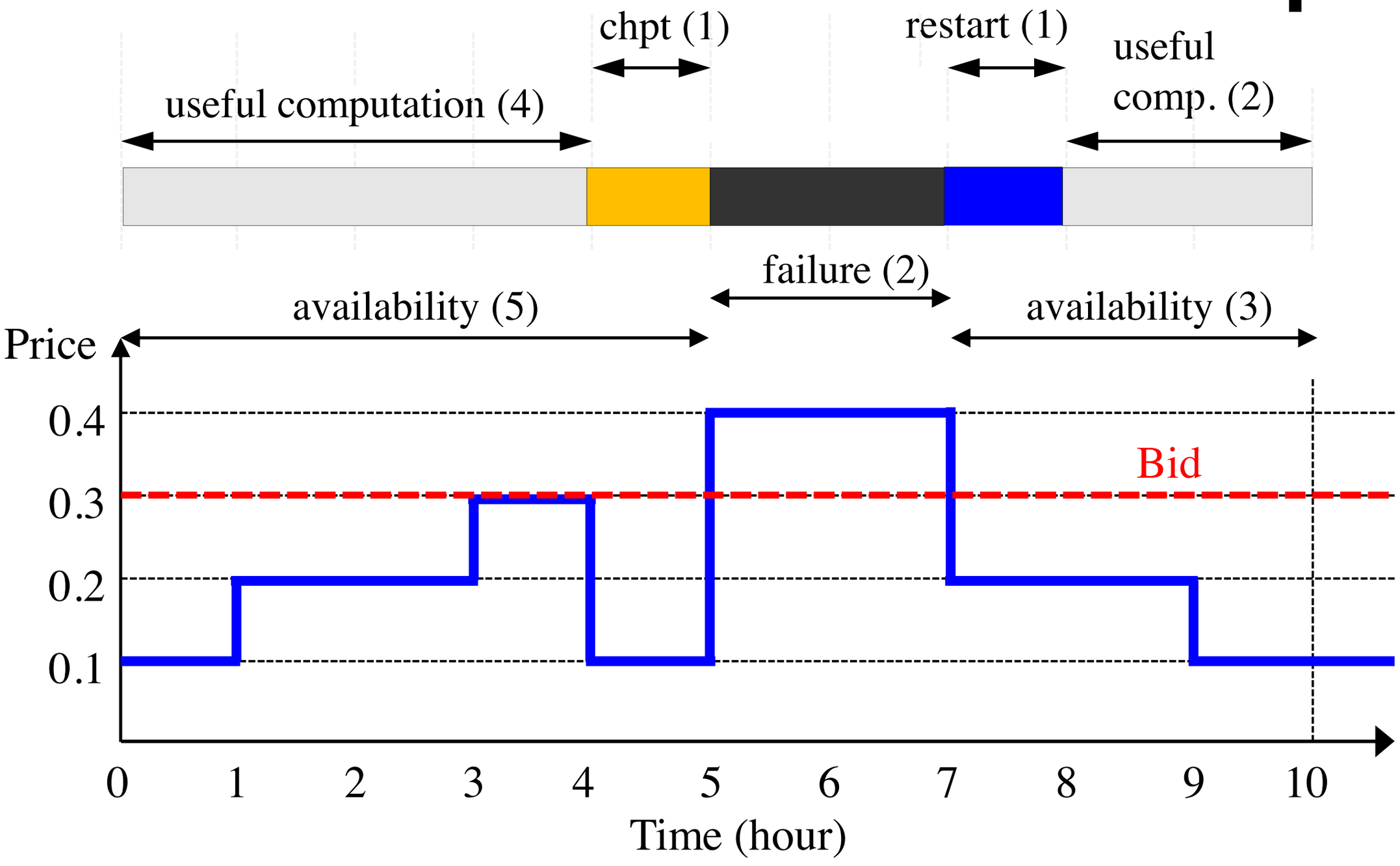
User decision variables

Random Variables of Model

Notation	Description
ET	execution time of the job (clock time)
AT	availability time (total time in-bid)
EP	expected price, i.e. (cost per instance)/AT
M	monetary cost $AT \cdot EP$ per instance
AR	availability ratio AT/ET
UR	utilization ratio T/ET

- performance
- reliability
- monetary cost

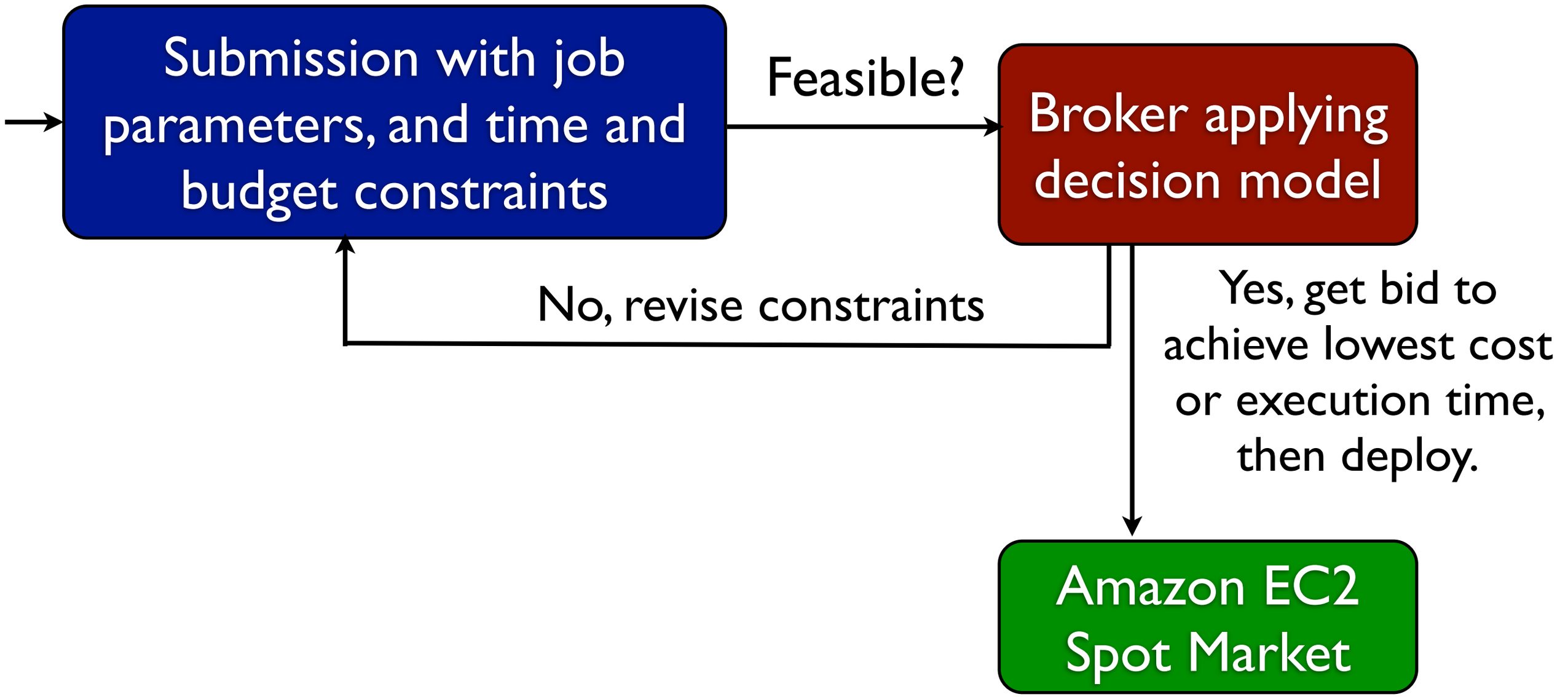
Execution Model Example



T = 6h
ET = 10h
AT = 5+3 = 8h
EP = 1.4/8 = 0.175 USD/h

M = 3*0.1+4*0.2+1*0.3
= 1.4 USD
AR = 8/10 = 0.8
UR = 6/10 = 0.6

Decision Workflow



Decision Model

- For a random variable, X, we write X(y) for x s.t. Pr (X < x) = y.
    - E.g. ET(0.50) is the median execution time
- Feasibility decisions
    - Deadline constraint achievable with confidence $c_{dead} \Leftrightarrow t_{dead} \geq ET(c_{dead})$
    - Budget constraint achievable with confidence $c_B \Leftrightarrow B \geq M(c_B)$
- Among the feasible cases, we choose the one with the smallest $M(c_B)$ or lowest execution time $ET(c_{dead})$

Outline

- System model
- Decision model
- Simulations method and results
- Related work
- Conclusion & Future work

Simulation Method

- Determine distributions of model variables via price trace-driven simulation
- Prices: trace of Spot instance prices obtained from Amazon
- Workload model
  - W1: "Big", based on Desktop Grids, parameters derived from BOINC catalog
  - W2: "Small", based on Grids, parameters derived from the Grid Workload Archive

Workload	I_{type}	n_{max}	W_{inst}	T	t_{dead}	c_{dead}
W1	2.5GHz	$20,000$	11.5	4.6h	9d	0.9
W2	2.5GHz	50	6.83	2.7h	17.9h	0.8

Experiments

- 10,000 simulation experiments per set of unique input parameters (I_{type}, u_b, T)
- Experiment corresponds to single task execution
- Starting point randomly selected during period [Jan 11, Mar 18] of Amazon's Spot Instance price traces

Experiments

- Run simulations for all 7 types of instances
- If not stated otherwise, show results for instance type A, with task length T of 276 minutes (W1) and hourly checkpointing

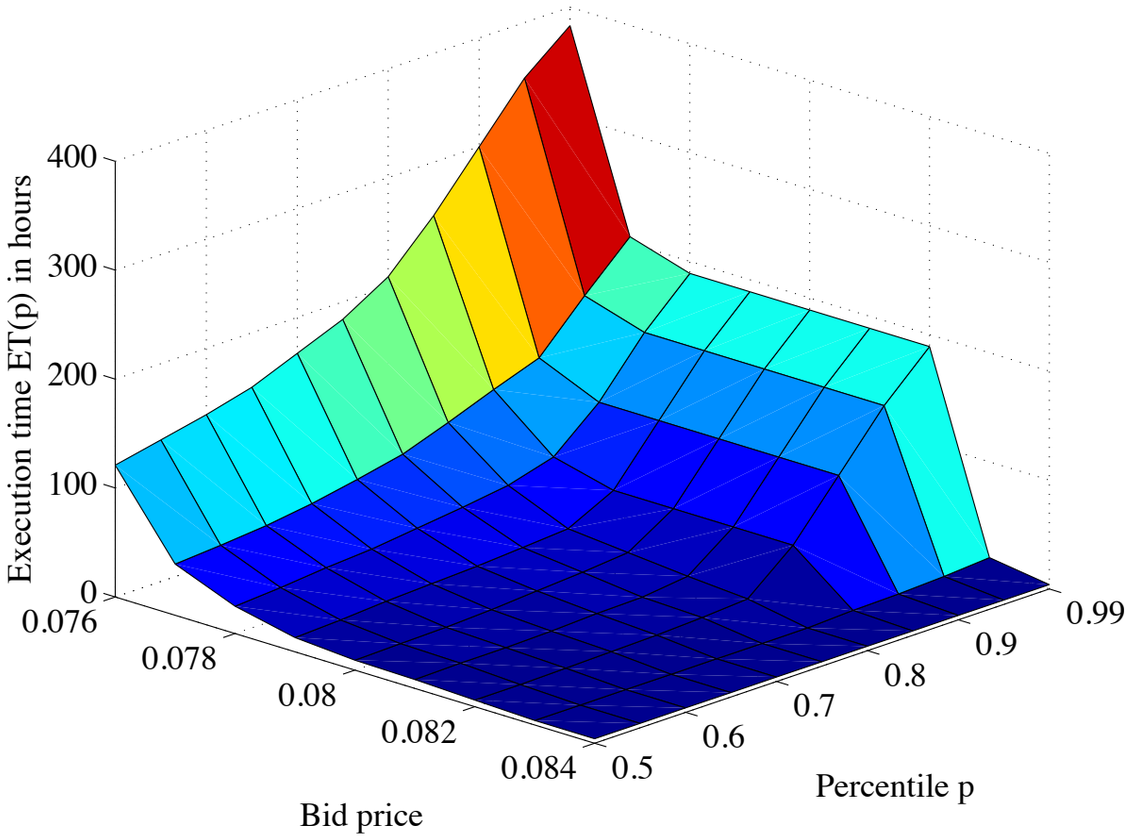
Table IV
USED INSTANCE TYPES (ZONE: EU-WEST-1; OS: LINUX; CLASS: HI-CPU = HIGH-CPU, STD = STANDARD, HI-MEM = HIGH-MEMORY)

Symbol	Class	API Name	Mem. (GB)	Total Units	Num. Cores	Units / Core
A	hi-cpu	c1.medium	1.7	5	2	2.5
B	hi-cpu	c1.xlarge	7	20	8	2.5
C	std	m1.small	1.7	1	1	1
D	std	m1.large	7.5	4	2	2
E	std	m1.xlarge	15	8	4	2
F	hi-mem	m2.2xlarge	34.2	13	4	3.25
G	hi-mem	m2.4xlarge	68.4	26	8	3.25

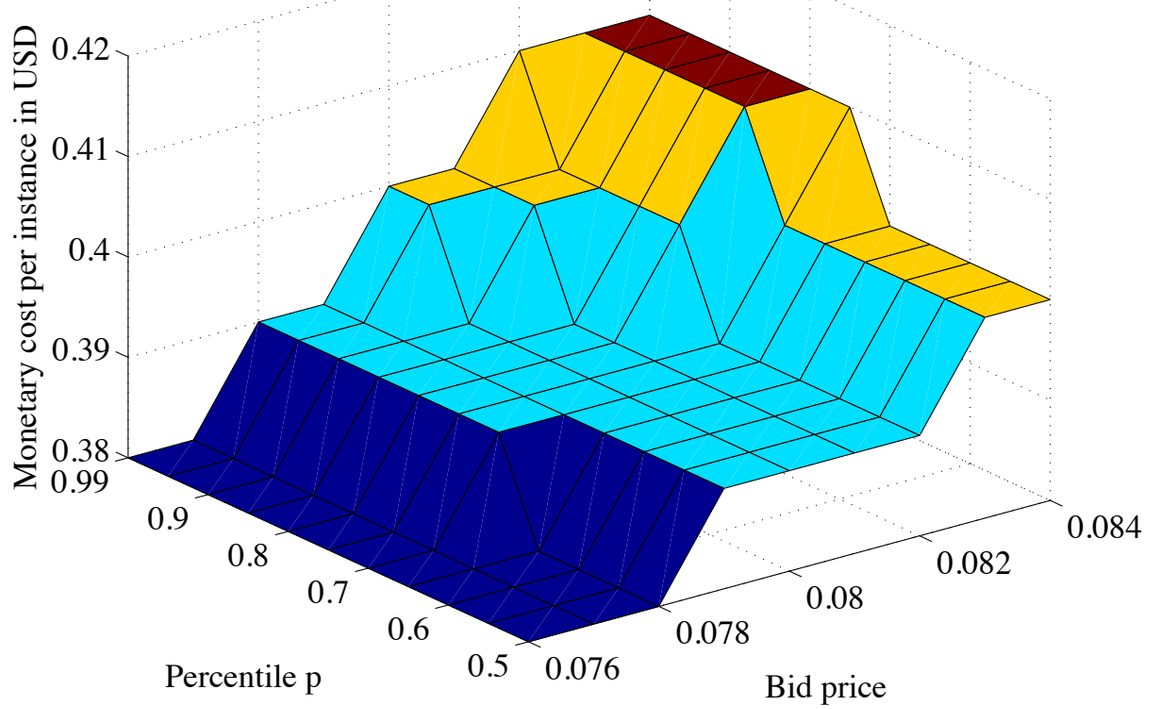
Assumptions

- Checkpointing
  - OPT: optimal checkpointing taken just before failure
  - HOUR: hourly checkpointing taken at each paid hour
- Use only 1 instance type per job
  - Note that deployment time, costs, failures are identical for all instances deployed in parallel.

Execution Time & Monetary Cost

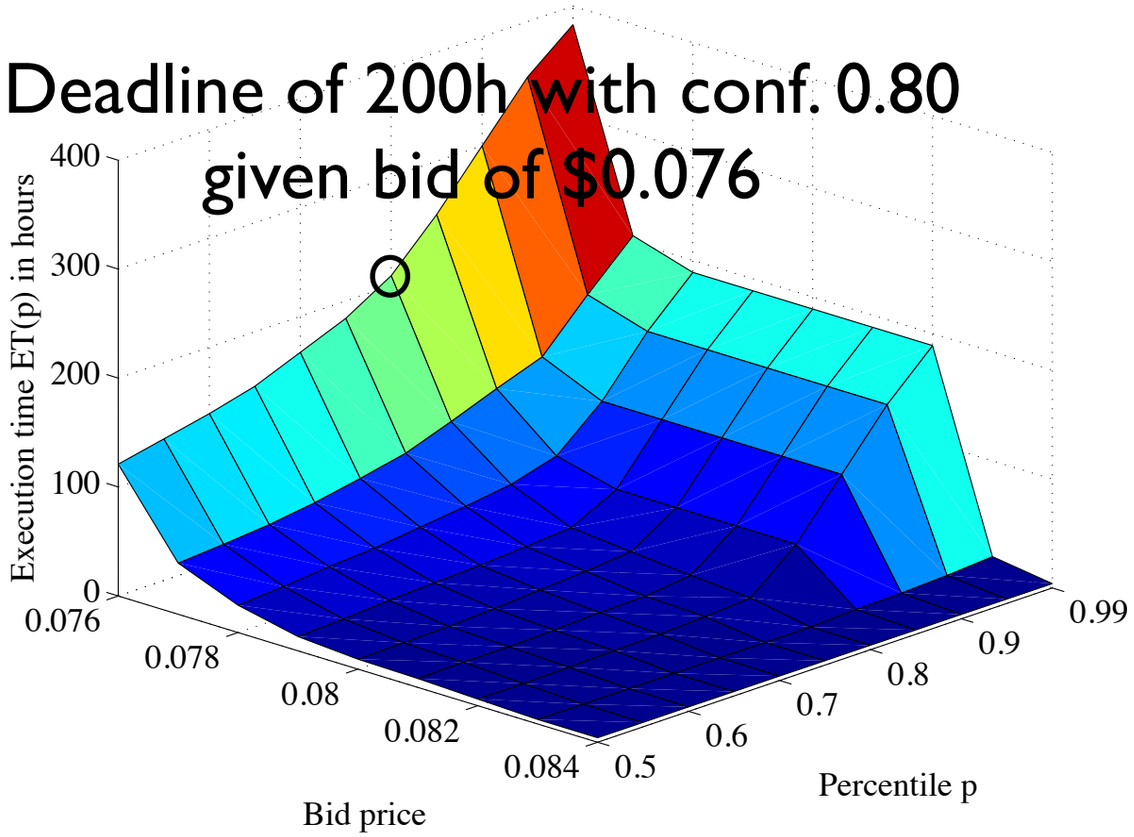


Execution time

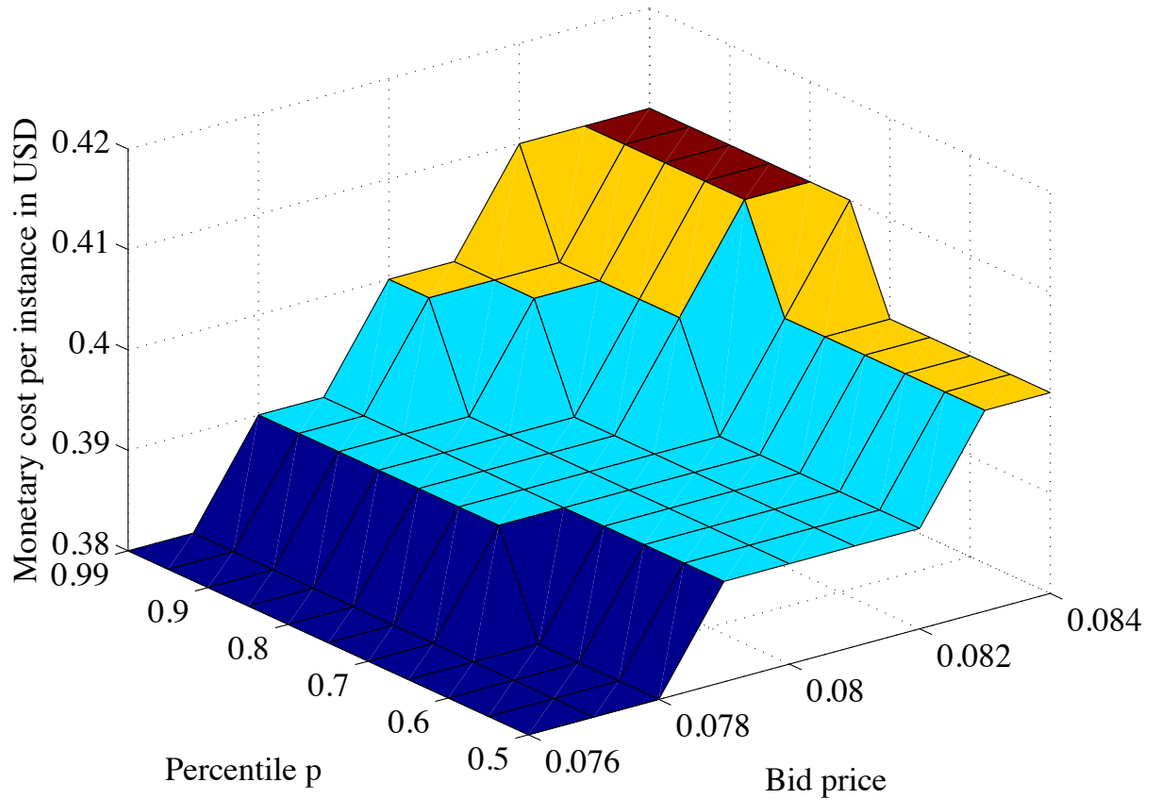


Monetary cost

Execution Time & Monetary Cost



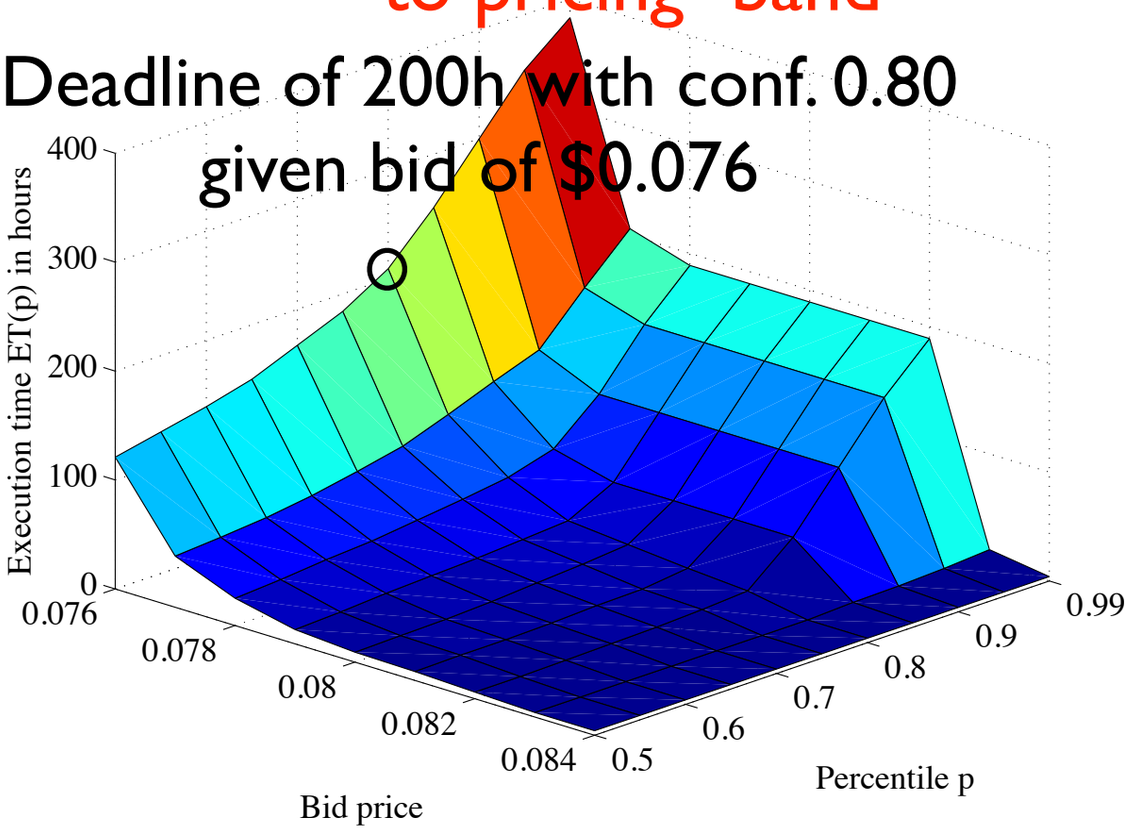
Execution time



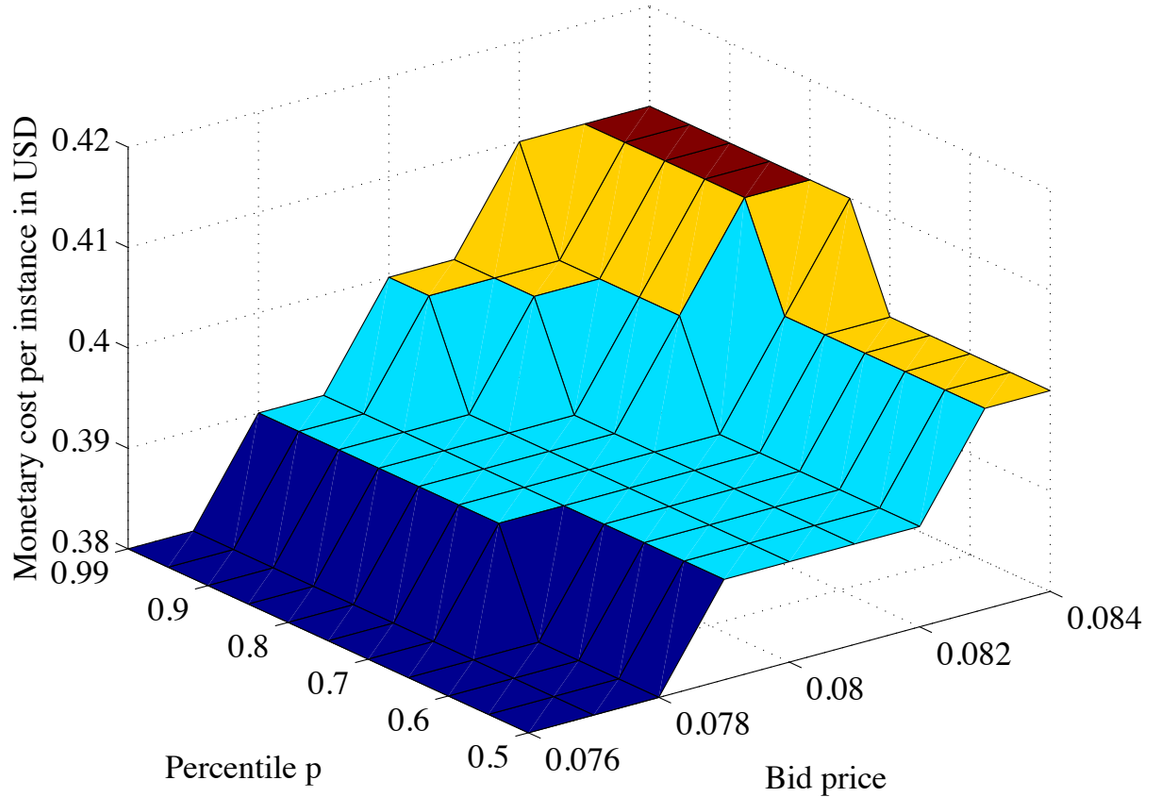
Monetary cost

Execution Time & Monetary Cost

Low bid prices can cause spike in ET due to pricing "band"



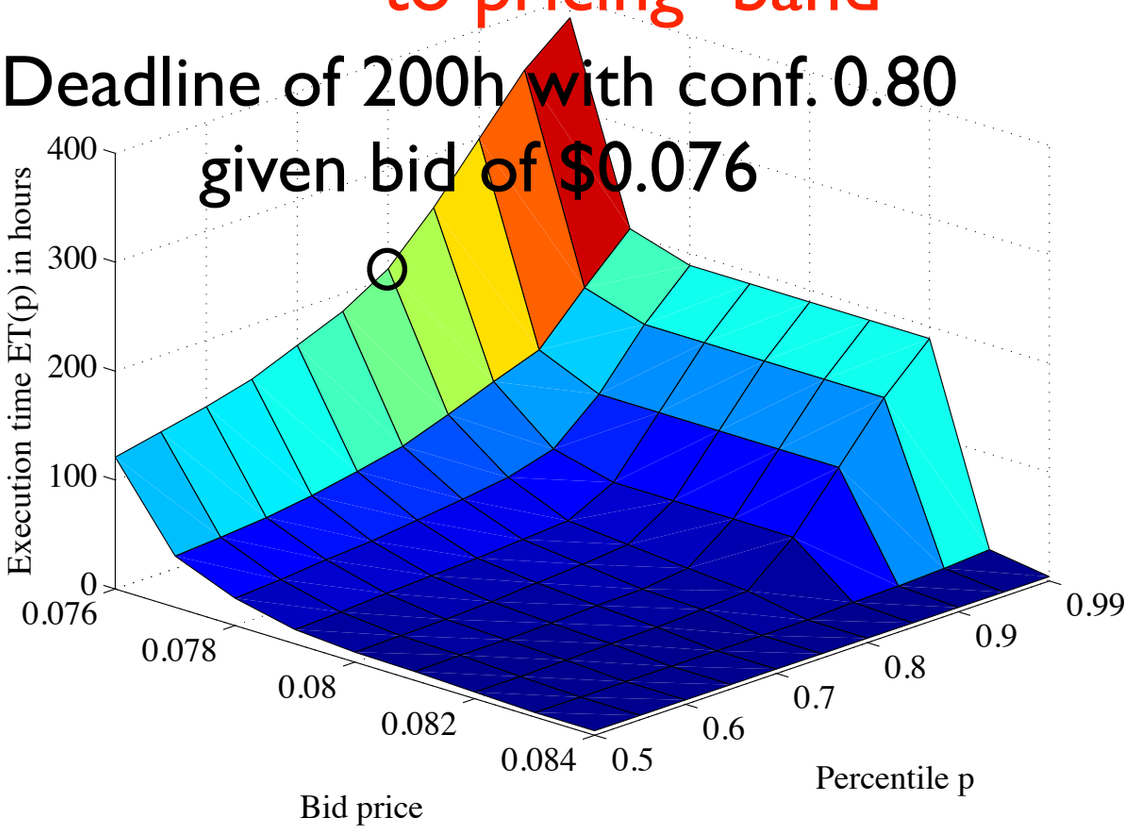
Execution time



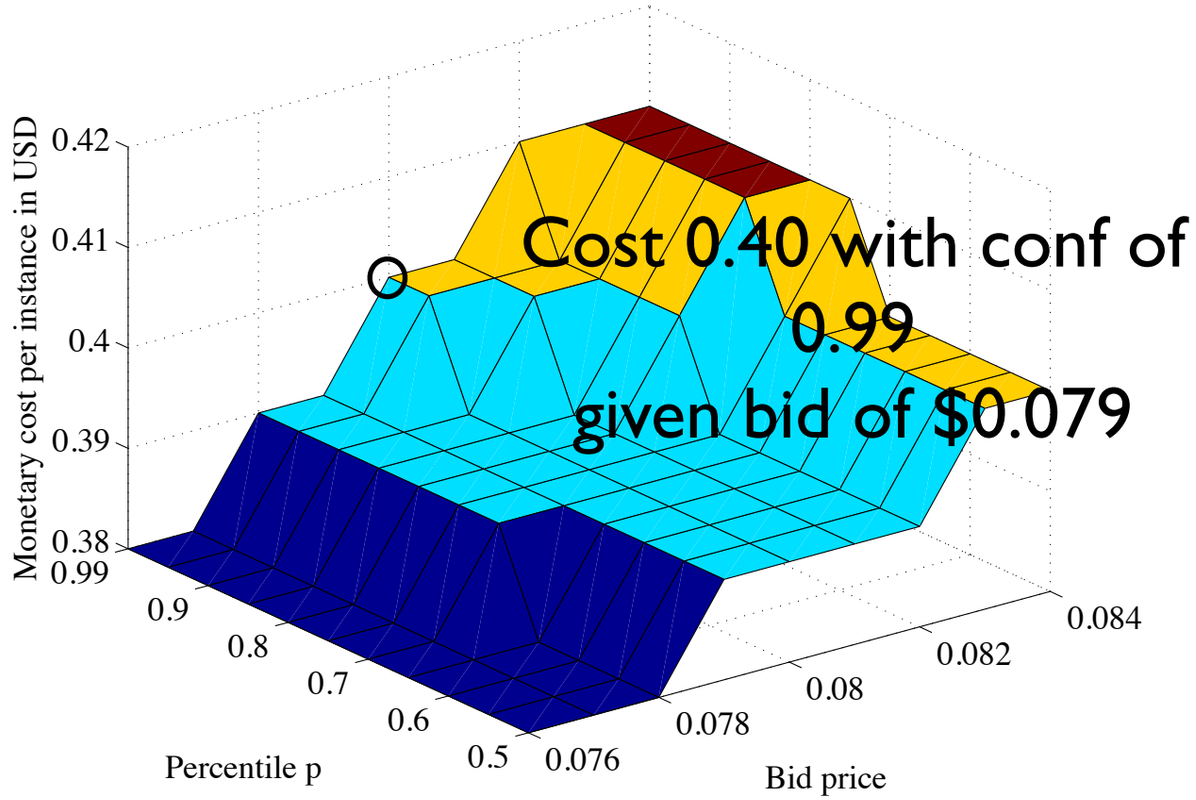
Monetary cost

Execution Time & Monetary Cost

Low bid prices can cause spike in ET due to pricing "band"



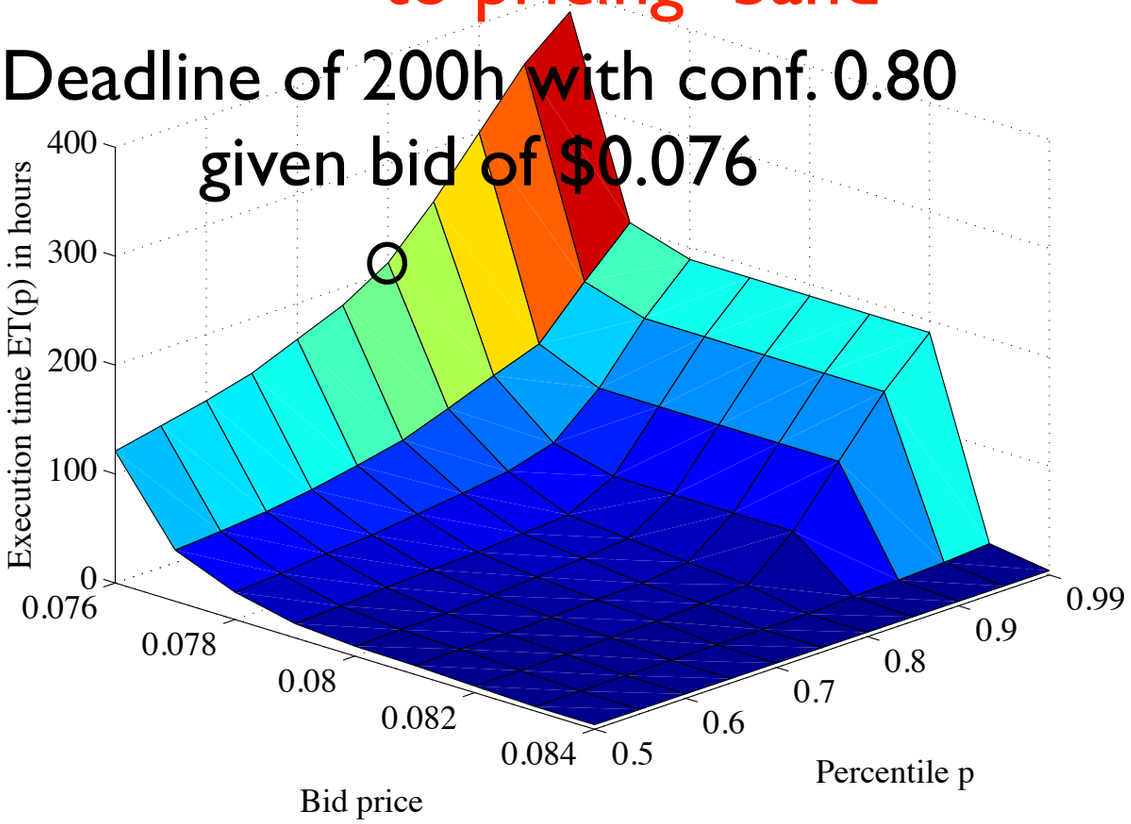
Execution time



Monetary cost

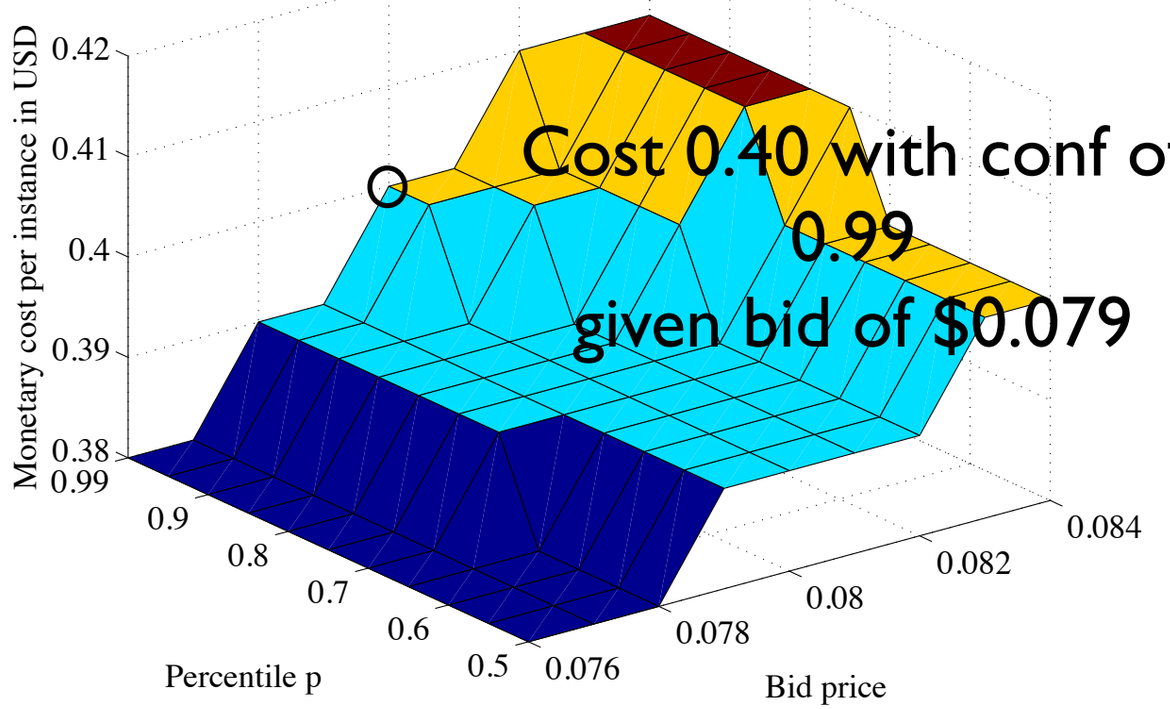
Execution Time & Monetary Cost

Low bid prices can cause spike in ET due to pricing "band"



Execution time

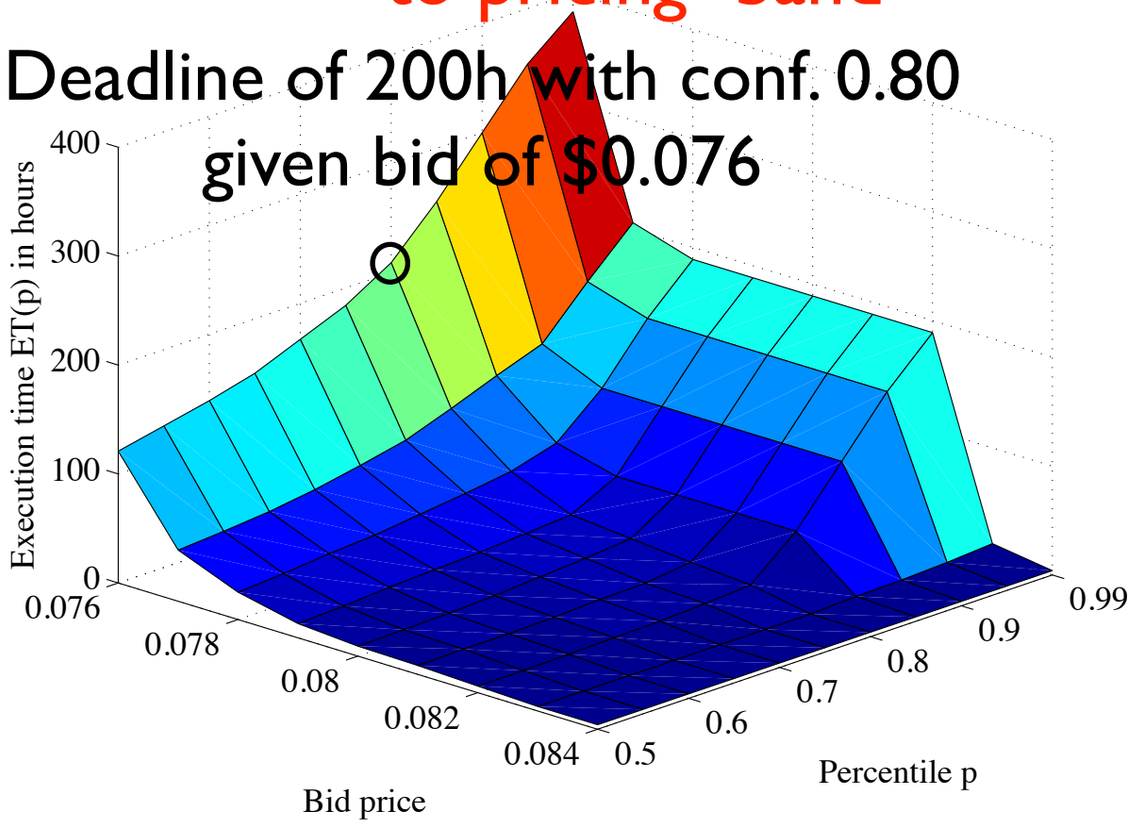
Cost decreases slightly with bid price



Monetary cost

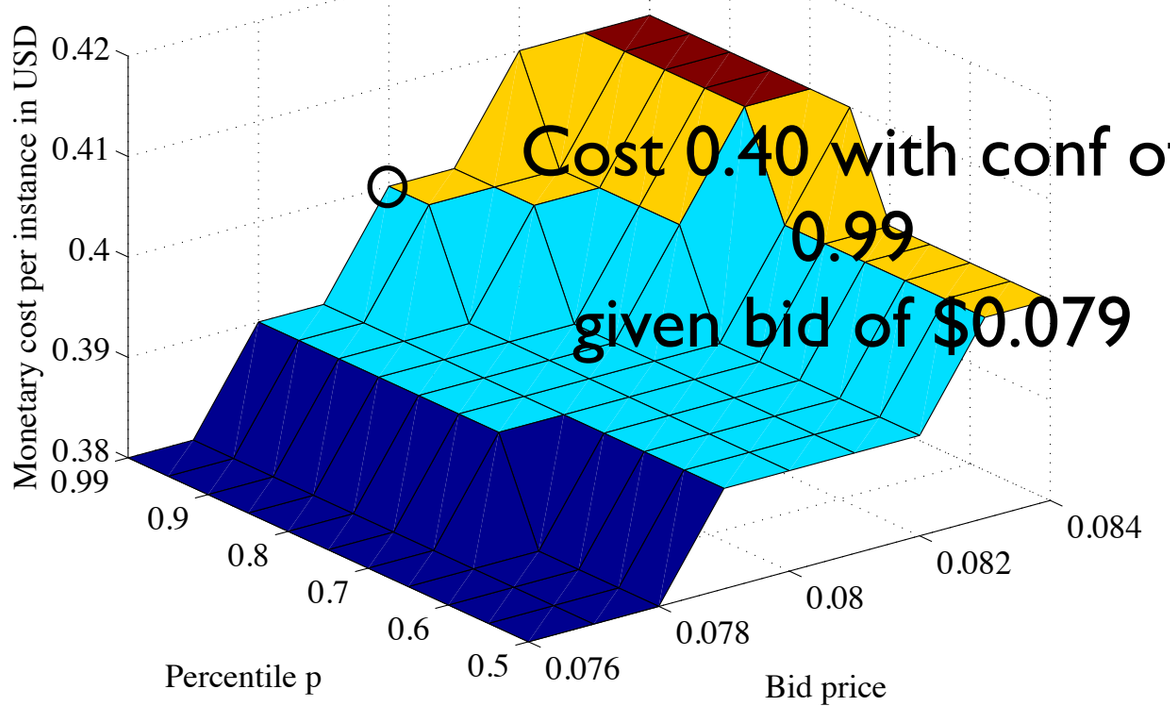
Execution Time & Monetary Cost

Low bid prices can cause spike in ET due to pricing "band"



Execution time

Cost decreases slightly with bid price



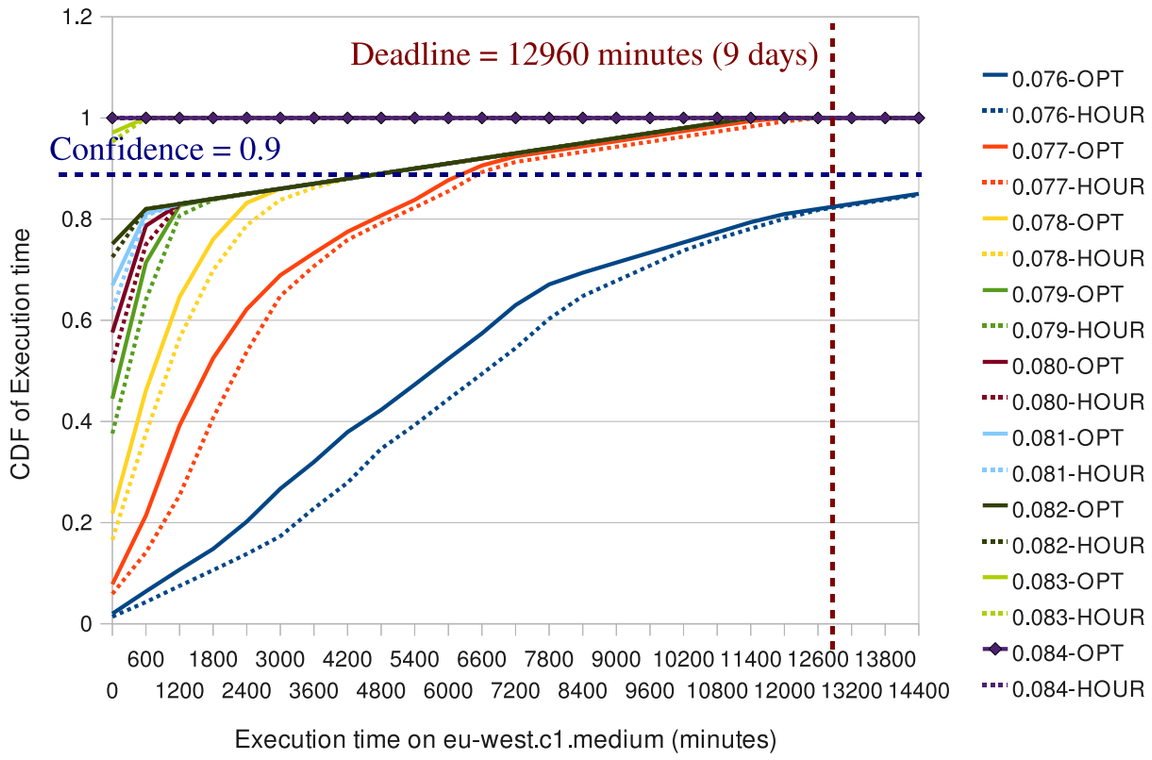
Monetary cost

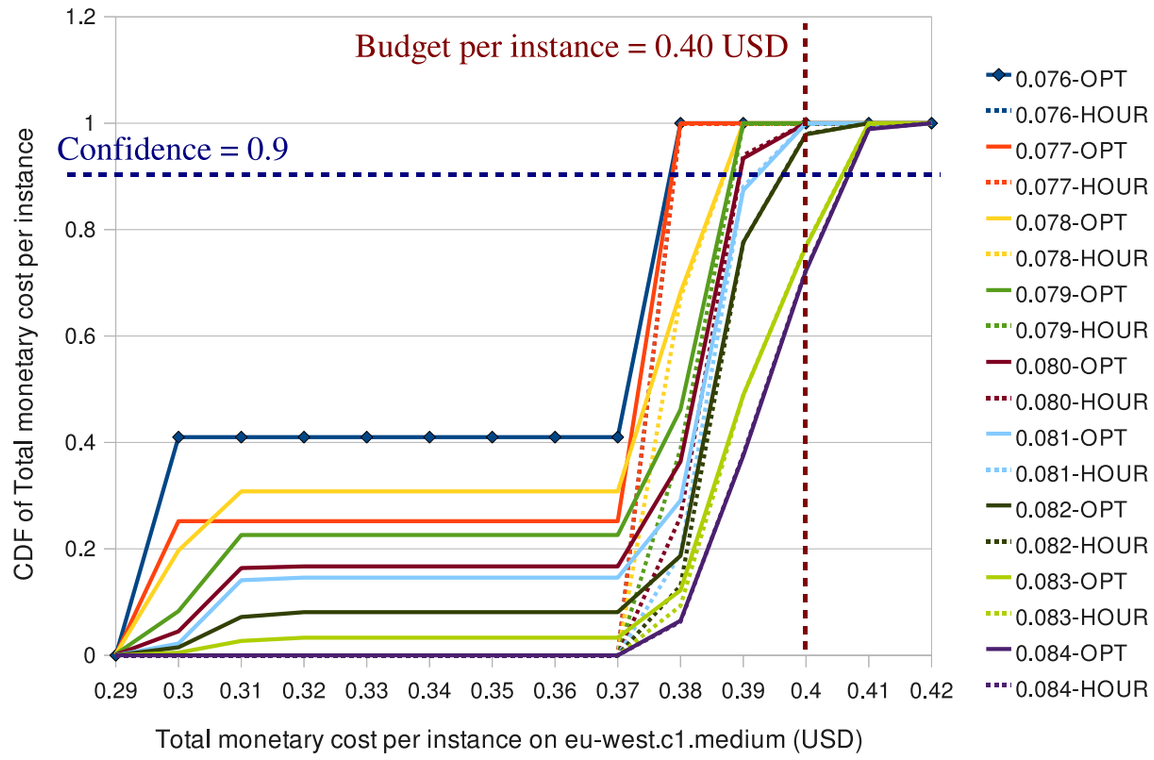
User can save about 10% in costs when bidding low but ET can be much higher

Other Relationships

- Checkpointing overhead versus bid price
  - HOUR checkpointing can amount to over 40% of task length T for low bids and availability confidence p
- AR versus task length T
  - AR is function of task length and bid price, and must consider both factors in decision model
- Coefficient of variation of ET versus task length T
  - Decreases sub-linearly with T. Can be used to determine "slack" between deadline and T.

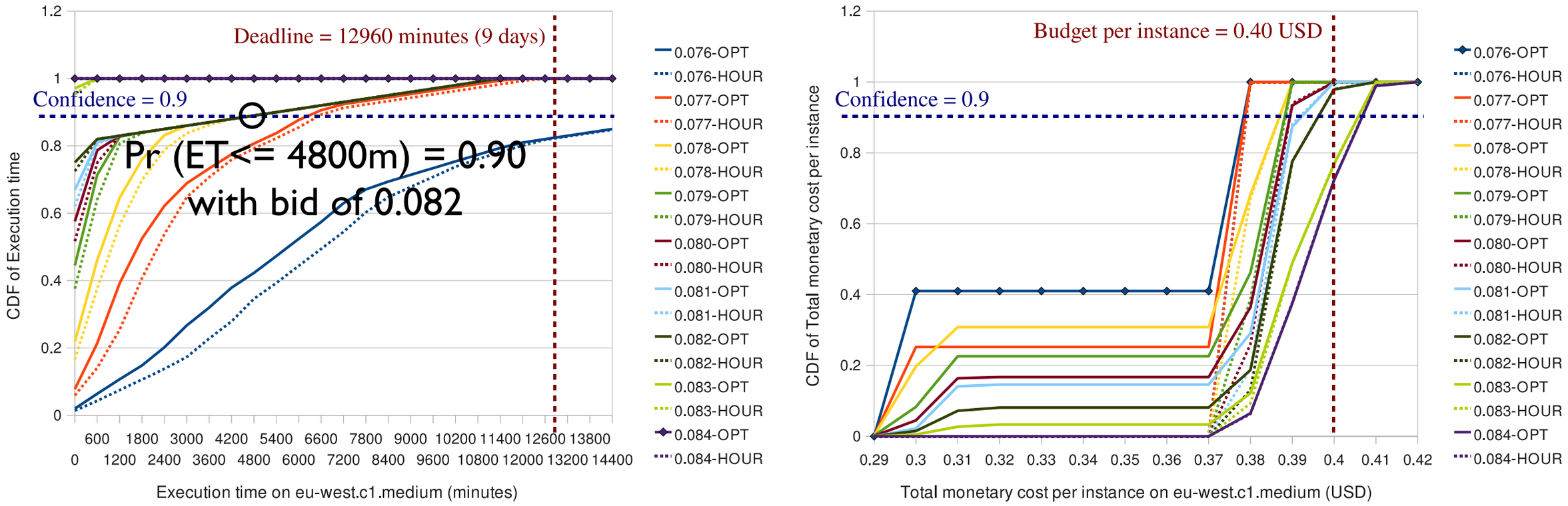
Distribution of Execution Time and Costs (Instance Type A and Workload W1)





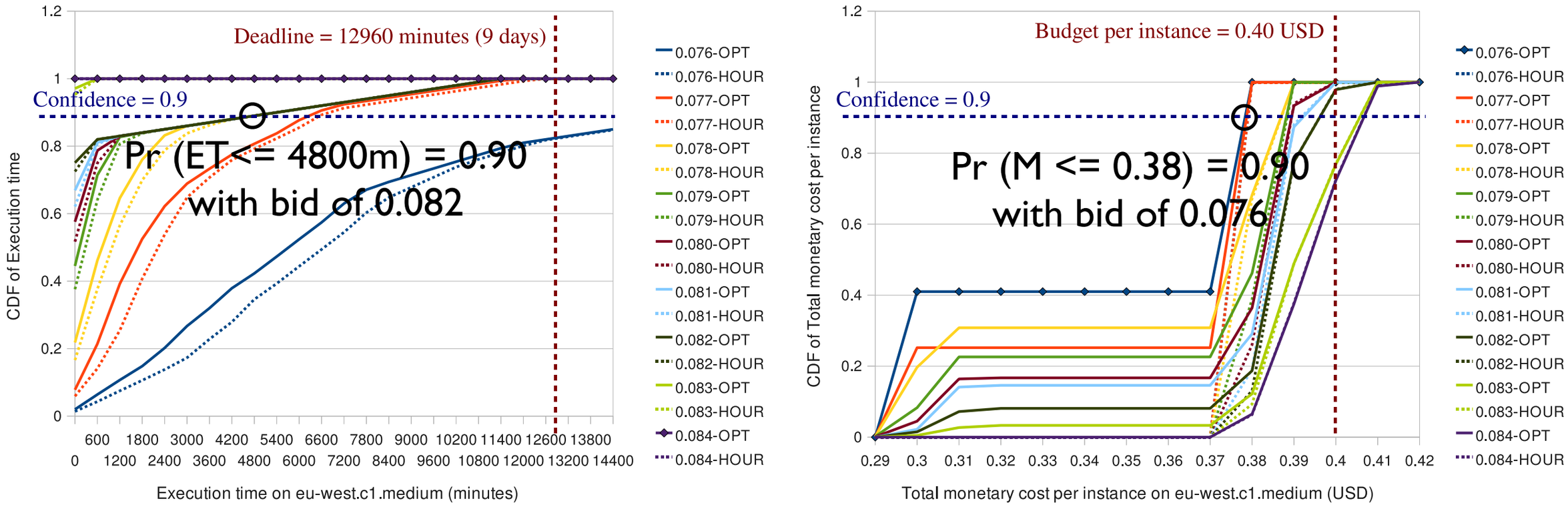
(b) when task length T = 276 minutes

Distribution of Execution Time and Costs (Instance Type A and Workload W1)



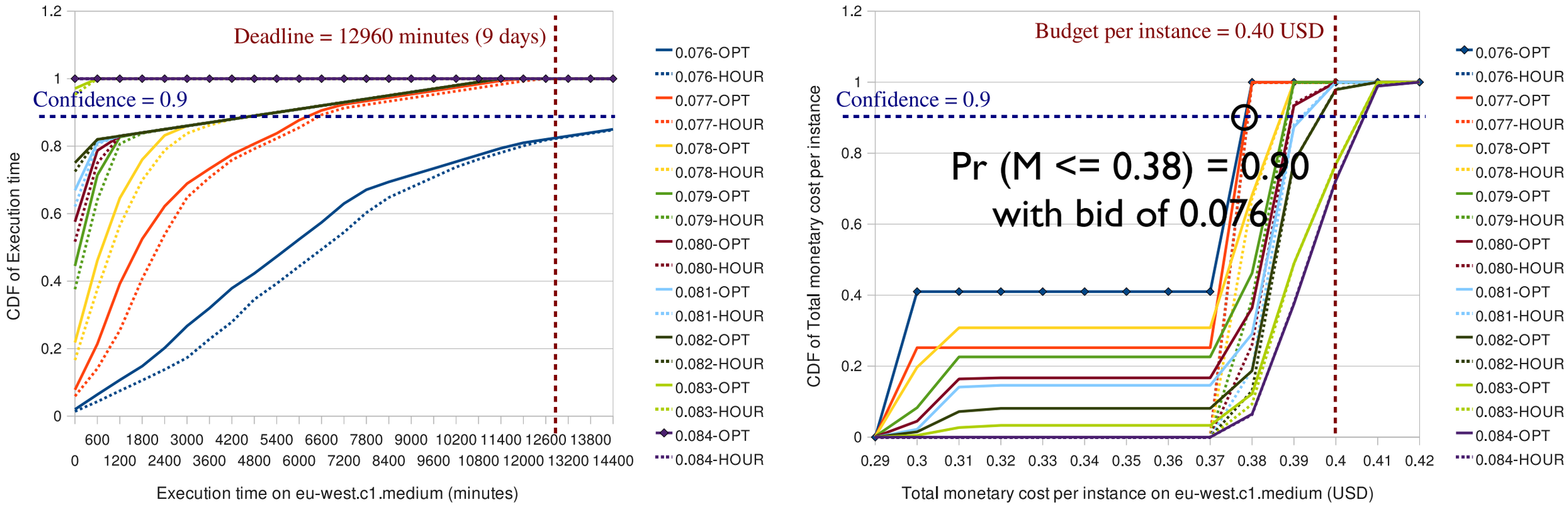
(b) when task length T = 276 minutes

Distribution of Execution Time and Costs (Instance Type A and Workload W1)



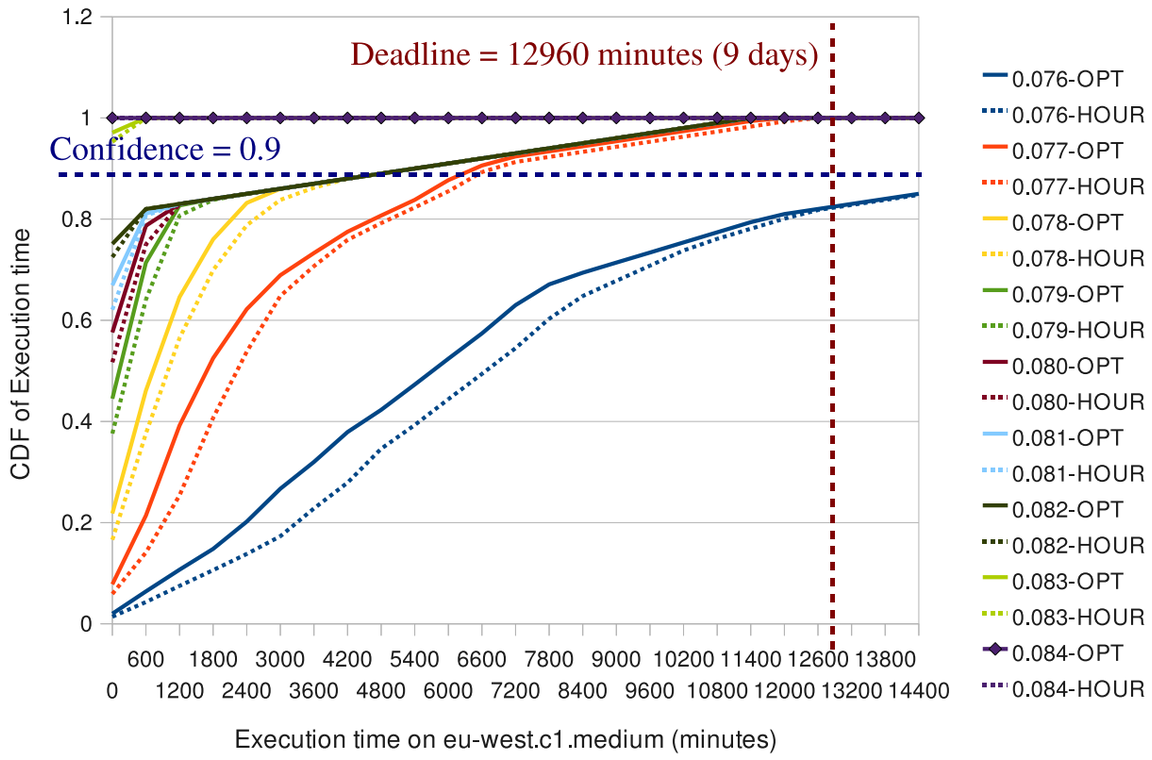
(b) when task length T = 276 minutes

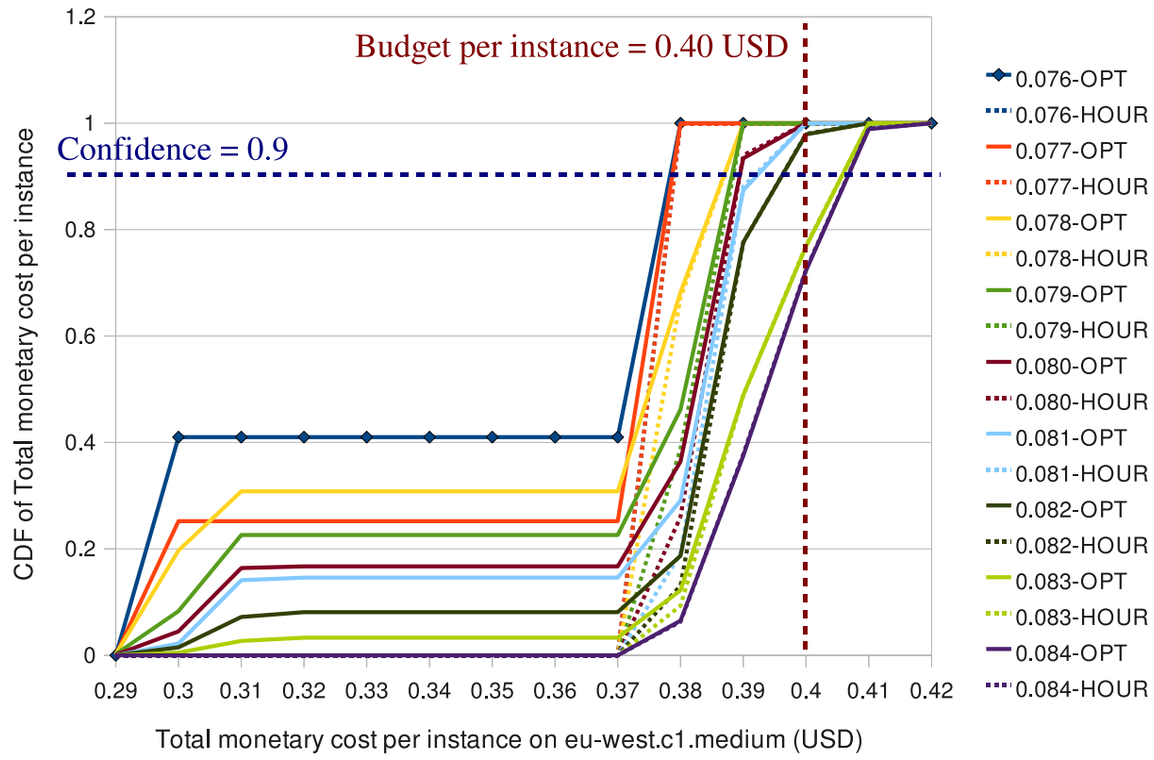
Distribution of Execution Time and Costs (Instance Type A and Workload W1)



(b) when task length T = 276 minutes

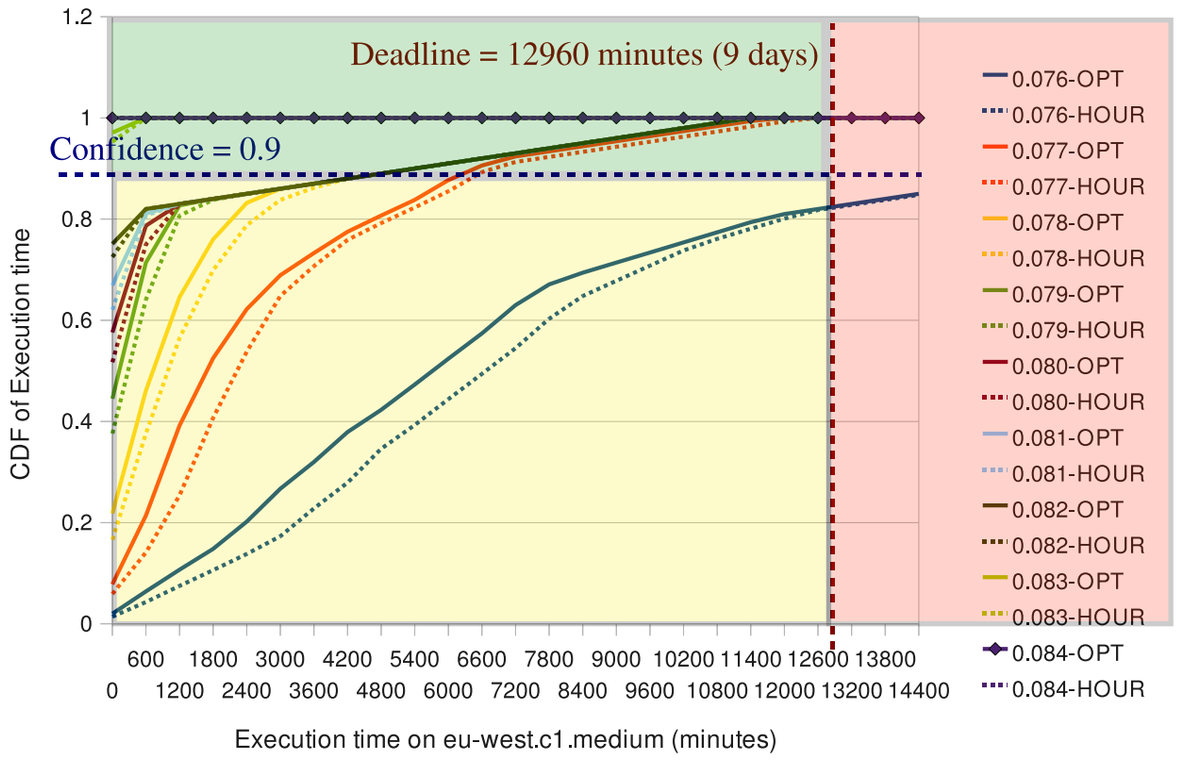
Distribution of Execution Time and Costs (Instance Type A and Workload W1)

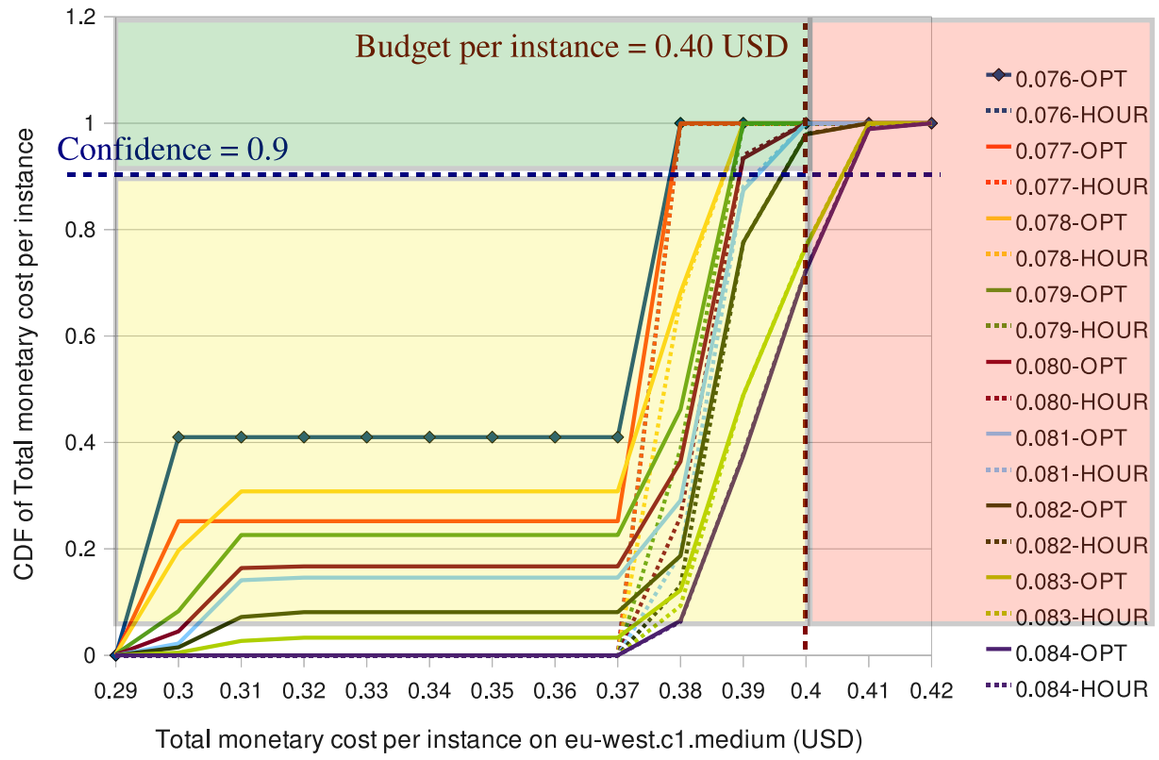




(b) when task length T = 276 minutes

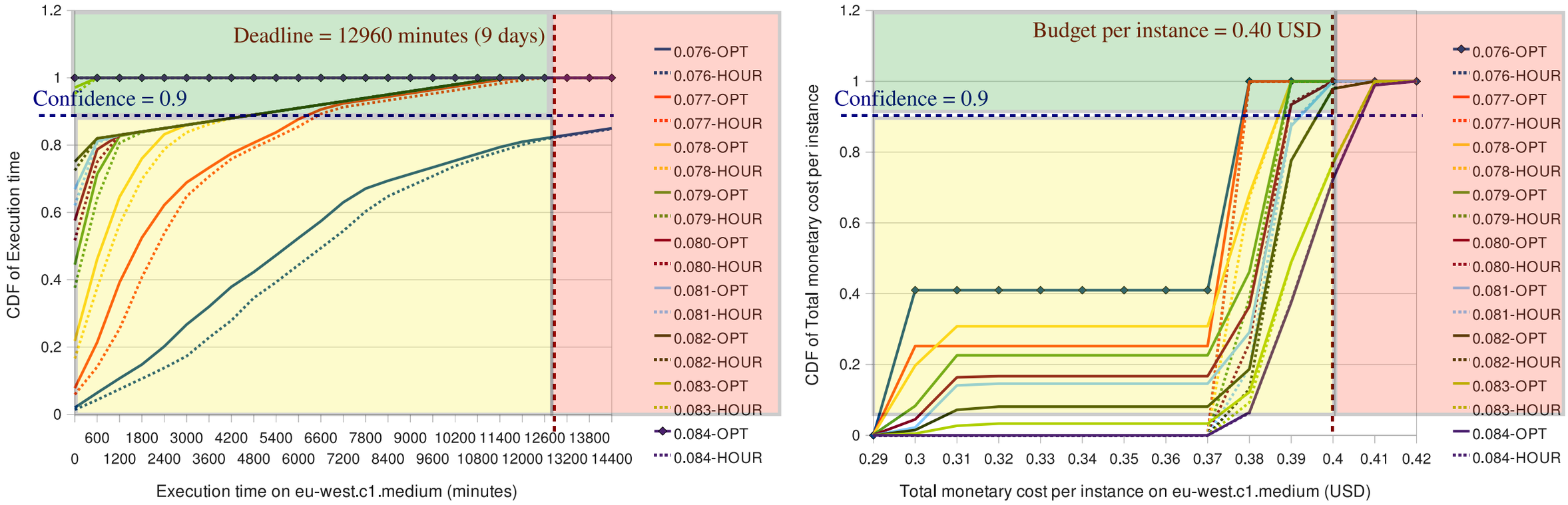
Distribution of Execution Time and Costs (Instance Type A and Workload W1)



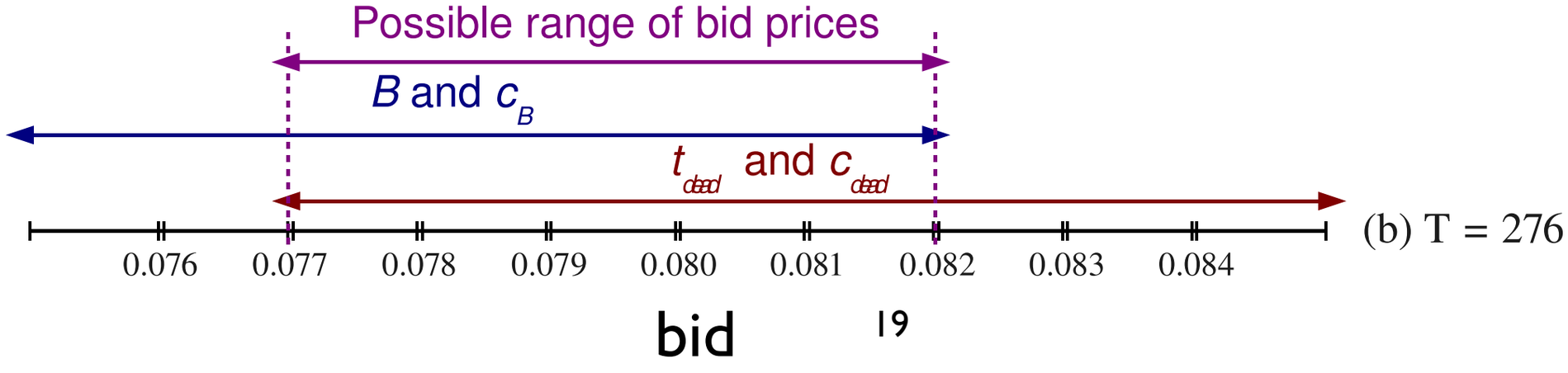


(b) when task length T = 276 minutes

Distribution of Execution Time and Costs (Instance Type A and Workload W1)

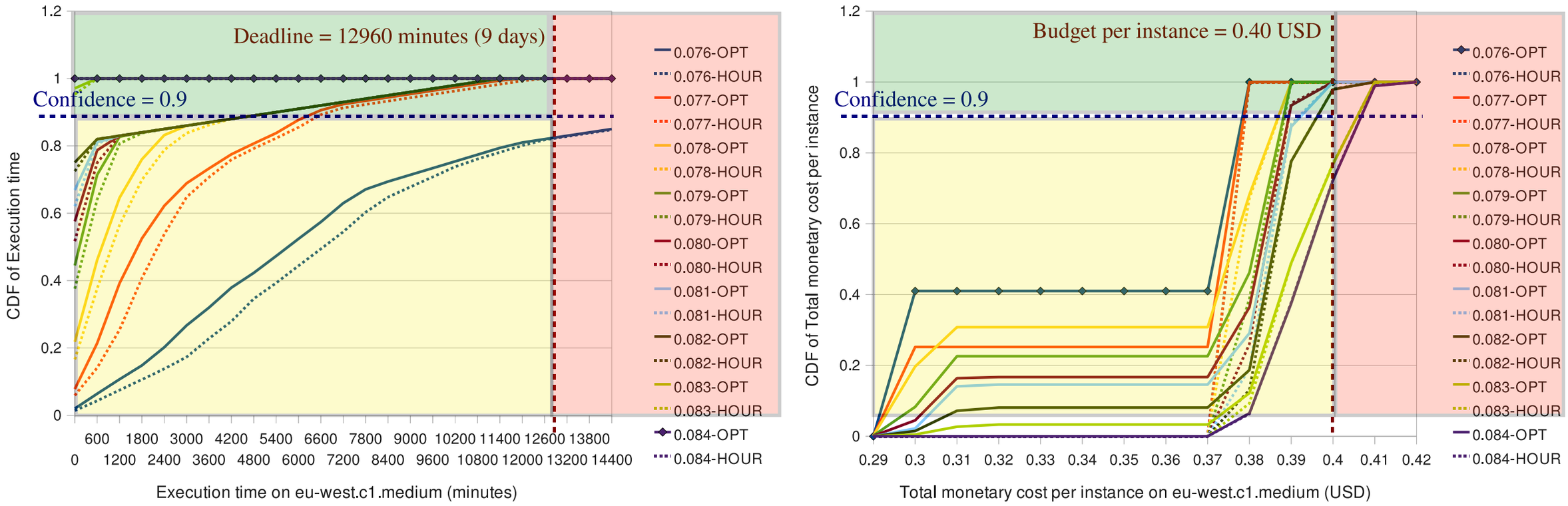


(b) when task length T = 276 minutes

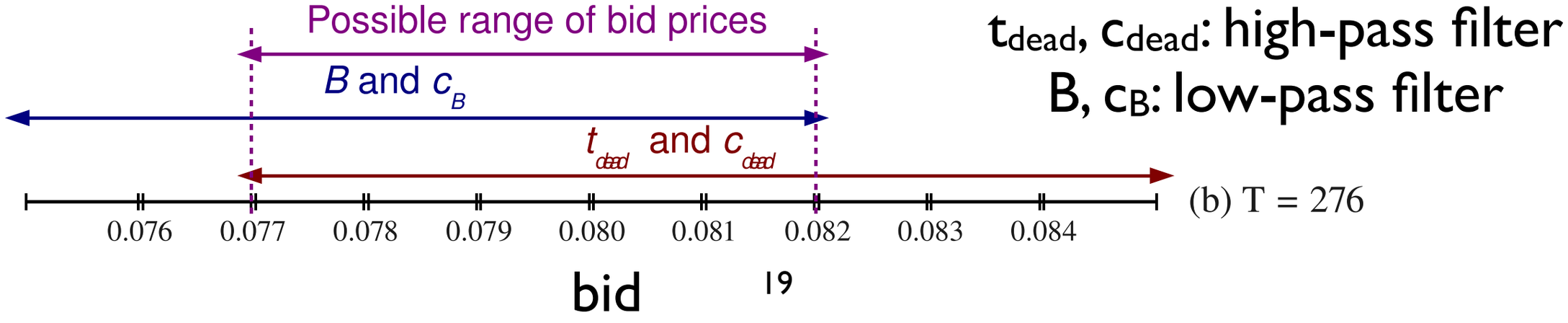


(b) T = 276

Distribution of Execution Time and Costs (Instance Type A and Workload W1)



(b) when task length T = 276 minutes



t_{dead}, c_{dead}: high-pass filter

B, c_B: low-pass filter

(b) T = 276

Gains in Adjusting Budget or Deadline Constraints

- Slightly changing confidence value for deadline can significantly reduce costs
  - For $c_{dead} = 0.90$ and $c_{dead} = 0.82$, we observe that using slightly lower confidence can reduce more than 21% of the monetary costs.
- Slightly changing budget or its confidence can have significant effect on execution time

Comparing Instance Types

BIDING PRICE COMPARISON ACROSS INSTANCE TYPES (IN US-CENTS)

Symbol	Class	Total Units	Low Bid	High Bid	Low / Unit	High / Unit	Ratio in %
A	hi-cpu	5	7.6	8.4	1.52	1.68	10.5
B	hi-cpu	20	30.4	33.6	1.52	1.68	10.5
C	std	1	3.77	4.2	3.77	4.2	11.4
D	std	4	15	16.8	3.75	4.2	12.0
E	std	8	30.4	33.6	3.8	4.2	10.5
F	hi-mem	13	53.3	58.8	4.1	4.52	10.3
G	hi-mem	26	106	118	4.08	4.54	11.3

Ratio in % = (H-L)/L * 100

Price variation is about 10-12% (likely due to regulation)

High CPU instances have lowest cost per unit hour and saves ~60% wrt standard instance

Related Work

- Batch schedulers
  - Don't have decision models that consider costs, performance, and reliability for scheduling
- Cloud services and monitoring
  - Don't make bidding recommendations
- Cloud economics
  - Don't take into account dynamic and tunable pricing or availability

Summary

- Largest cost savings achieved by using high-CPU instance types instead of standard or high-memory instance types
- Bidding low prices yields cost savings of about 10%, but can lengthen execution time significantly
- Our model allows a user to turn several knobs (parameters) to achieve desired balance between monetary costs and service levels (job deadlines or reliability).
- Data, simulator, results available at:
  - http://spotmodel.sourceforge.net

Future Work

- Include mixes of instances (wrt size, availability zones, dynamically adjusting number of instances, rebidding and restarting)
- Price prediction or reverse engineering of pricing scheme
- Implement web service and middleware that applies it

Thank you