

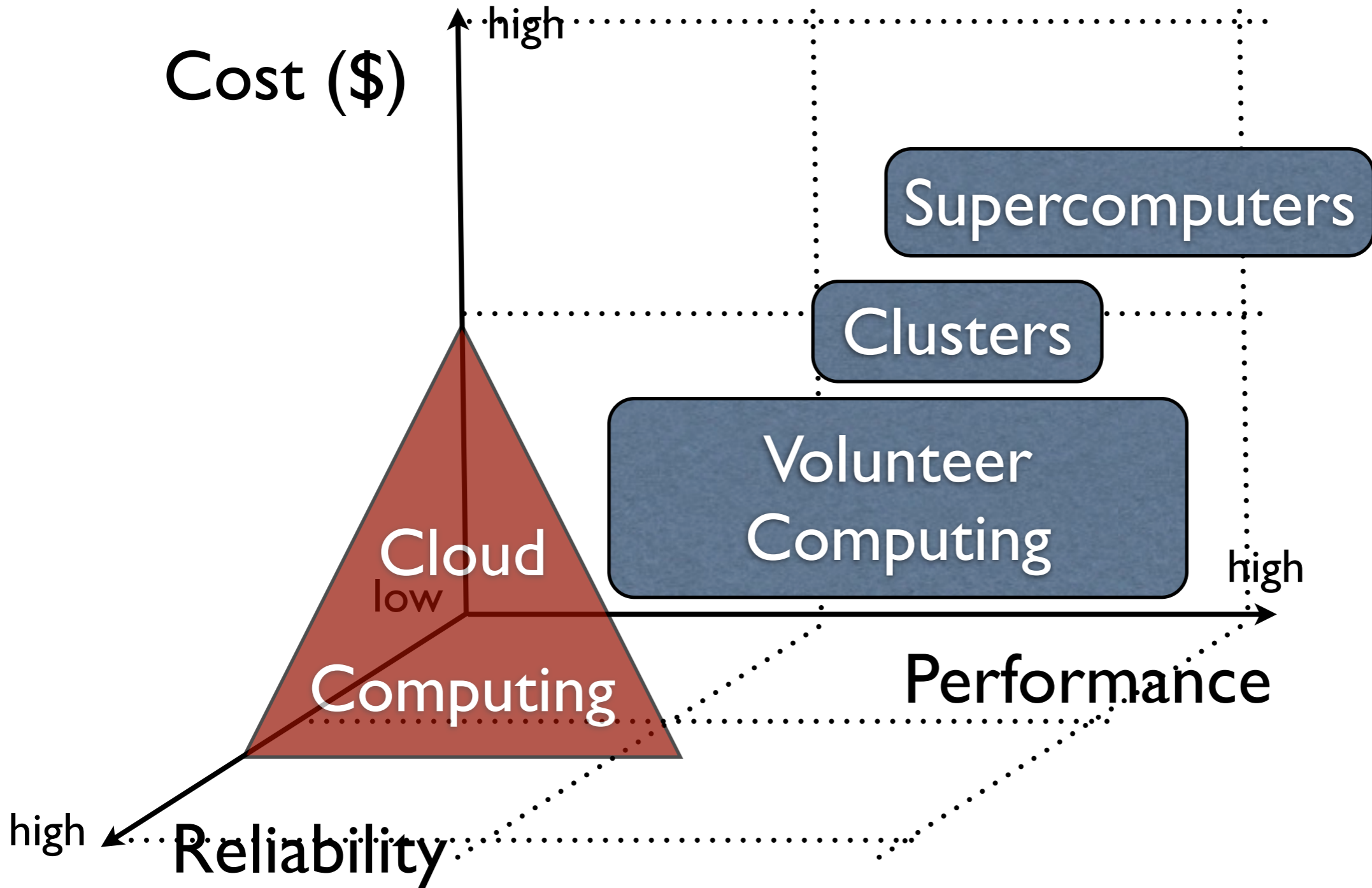
Volunteer Computing in the Clouds

Artur Andrzejak¹, Derrick Kondo², Sangho Yi²

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

²INRIA Grenoble, France

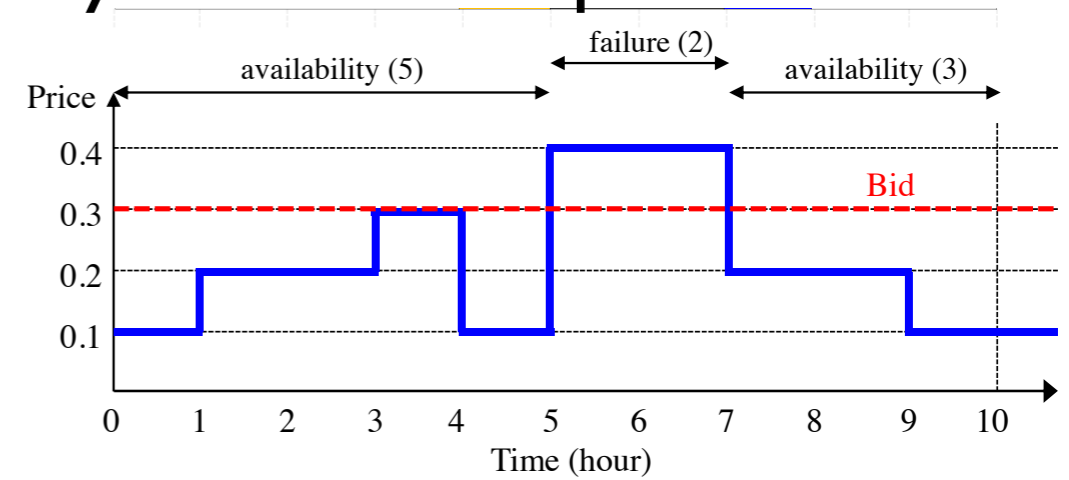
Trade-offs



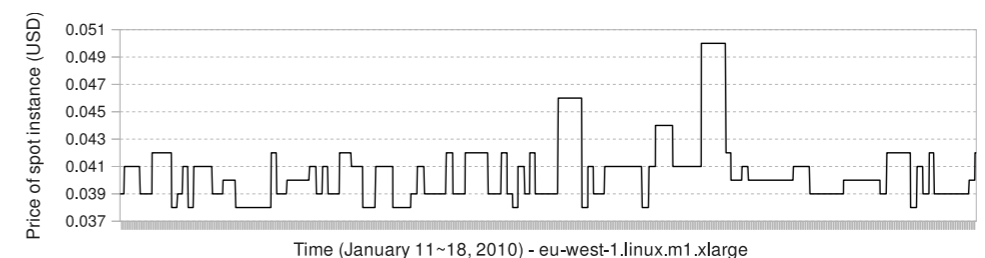
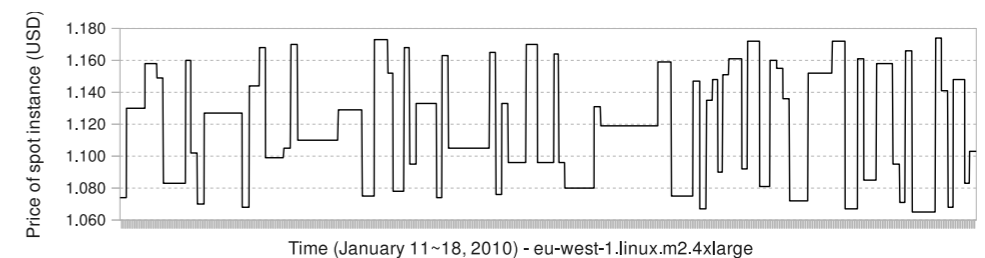
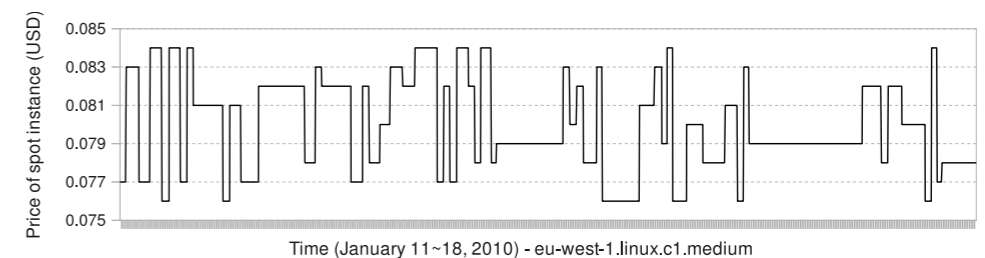
Market-based Resource Allocation Systems

- 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 \leq current price
- Amazon charges by the last price at each hour

Synthetic Example:



Real Amazon Price Trace:



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
- Relation with BOINC
- 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

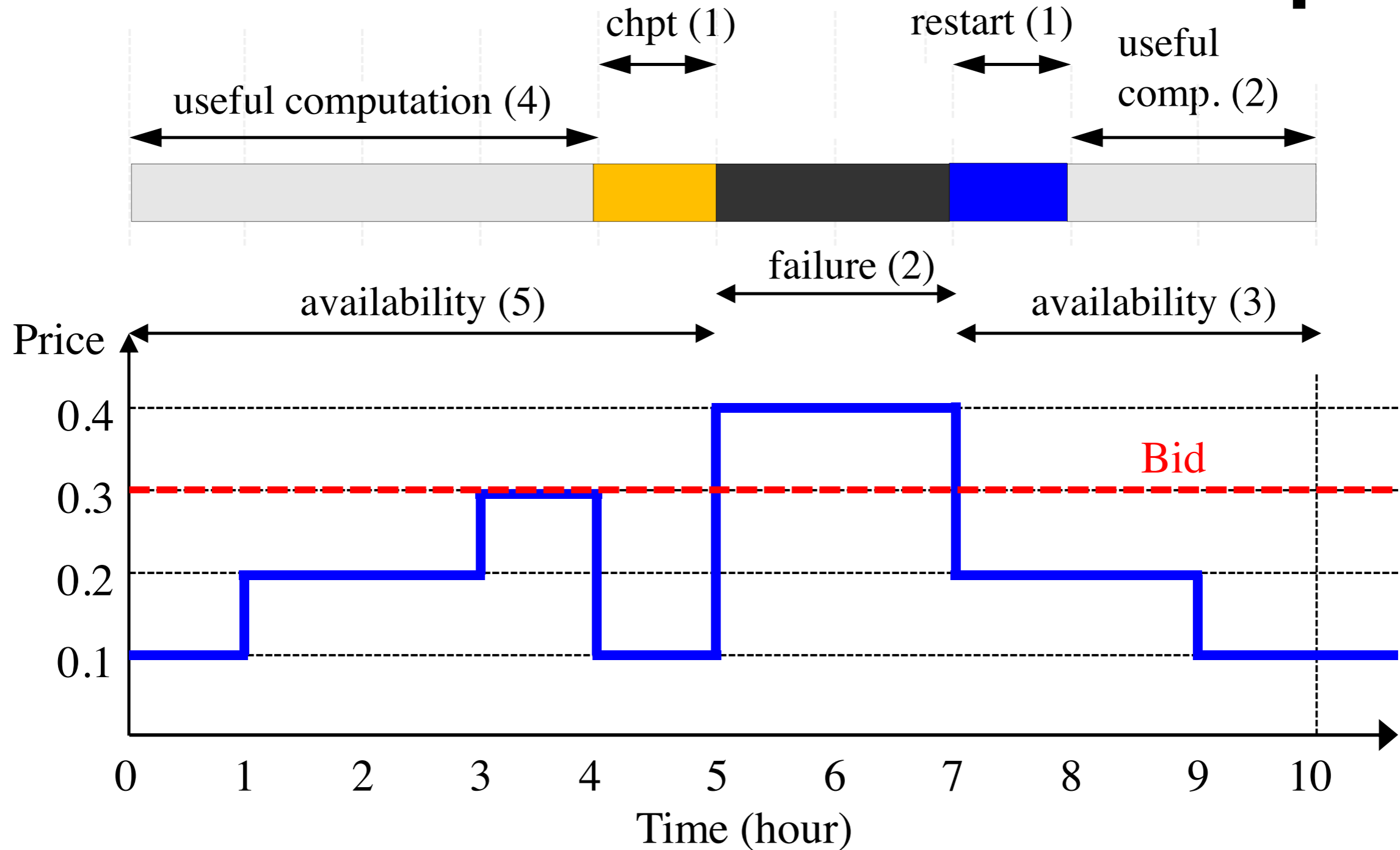
-  Job parameters
-  Job constraints
-  User decision variables

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



$$T = 6\text{h}$$

$$ET = 10\text{h}$$

$$AT = 5 + 3 = 8\text{h}$$

$$EP = 1.4 / 8 = 0.175 \text{ USD/h}$$

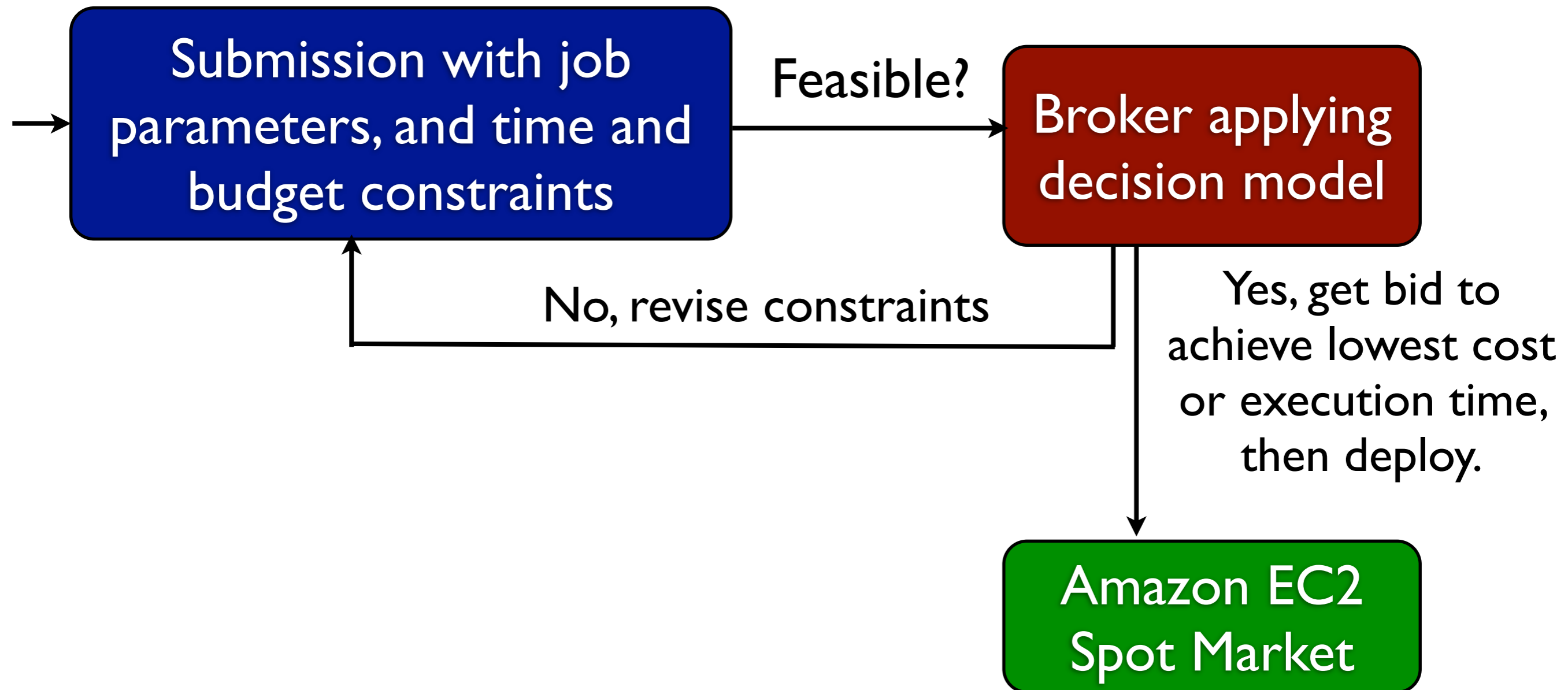
$$M = 3 * 0.1 + 4 * 0.2 + 1 * 0.3$$

$$= 1.4 \text{ USD}$$

$$AR = 8 / 10 = 0.8$$

$$UR = 6 / 10 = 0.6$$

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
 $c_{\text{dead}} \Leftrightarrow t_{\text{dead}} \geq ET(c_{\text{dead}})$
 - 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 $ET(c_{\text{dead}})$

Outline

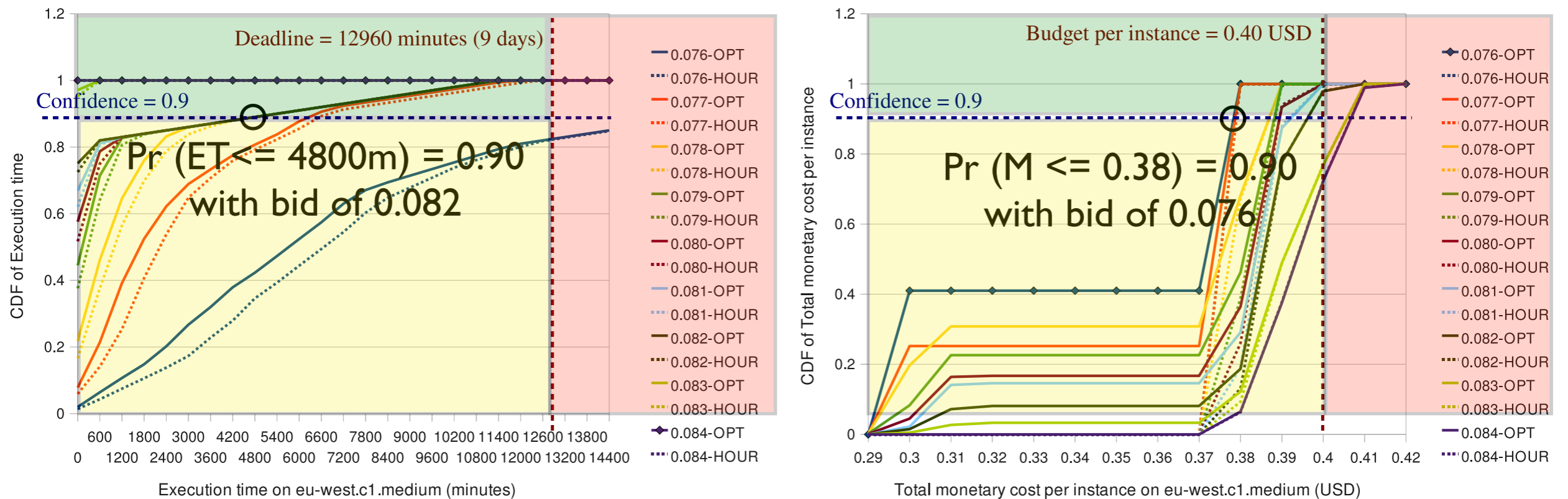
- System model
- Decision model
- Simulations method and results
- Relation with BOINC
- Conclusion & Future work

Simulation Method

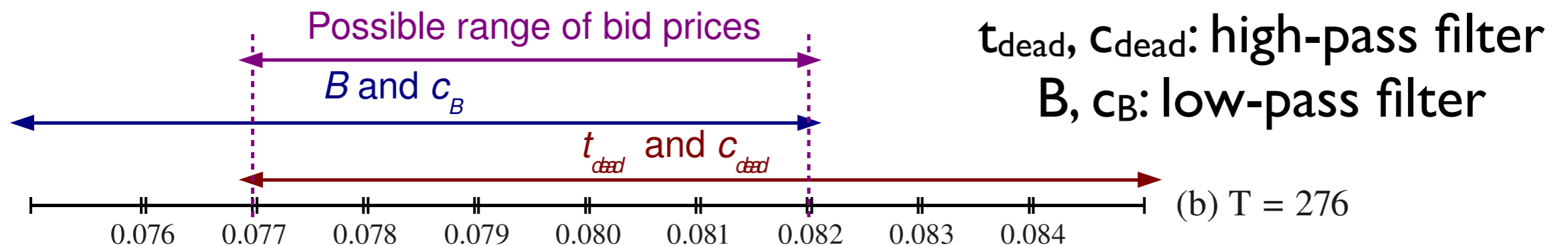
- Determine distributions of model variables via price trace-driven simulation
- Prices: trace of Spot instance prices obtained from Amazon
- Workload model
 - W1: “Big”, based on Volunteer Computing, parameters derived from BOINC catalog
 - W2: “Small”, based on Grids, parameters derived from the Grid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

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



(b) when task length $T = 276$ minutes



(b) $T = 276$

Relation to BOINC?

- Amazon does not provide any middleware for Spot instances
- BOINC is ideal as it handles nondeterministic failures, and ongoing work with VM integration would allow transparent checkpointing
- Use BOINC with decision model to be cost-aware
- Cloud-enabled BOINC client or server?
- Integrate with volunteers on the Internet, Grids etc?

Why not just use Internet volunteers?

- Reliability of Spot Instances is tunable (at a cost)
- Greater inter-node connectivity + higher bandwidth
 - ~1 Gbit among EC2 instances*.
 - ~100Mbit down/55Mbit up between EC2 and S3*
- Scientific data can be hosted on Amazon for free

* <http://blog.rightscale.com/2007/10/28/network-performance-within-amazon-ec2-and-to-amazon-s3/>

Hybrid Use Case

- Scientist submit 10,000 jobs
- Last 7%* are stragglers and delay job completion
- Run last 700 jobs on Amazon Spot Instances in parallel all at once
- Spot instance cost: $\sim \$210 \pm \20
- Could be cheaper if use reliable host mechanism
- Tune reliability according to budget and time constraints of user

* Personal communication with Kevin Reed

Implementation Approach*

- Distinguish BOINC cloud nodes
 - Create accounts with special id
- Schedule on cloud nodes
 - Use matchmaking function `is_wu_feasible_custom?`
- Prioritize work units later in batch
 - Use feeder to prioritize by `result_id` or priority

*Thanks to David Anderson

Discussion Questions

- Would application scientists use hybrid volunteer computing / cloud platforms?
 - Accounting model?
- Would volunteers use cloud platforms?
- Would hybrid system allow for new types of applications in terms of data intensity or message passing?

Plug

- EU project
 - European Desktop grid Initiative (EDGI)
 - Open 2-year post-doc in Lyon

Thank you