# Balancing Risk and Reward in a Market-based Task Service

**David Irwin,** Laura Grit,

Jeff Chase

Department of Computer Science
Duke University

## Resource Management in the Large

Grids enable resource sharing

- Each user has ability to use more resources
- Requires global coordination of resource sharing

Current technology: *private* grids (Virtual Organizations)

Next generation: *public* Grid

- Larger scale of resources and participants
- Dynamic collection suppliers and consumers
- Varying supply and demand

Market-based approaches are attractive

- Decentralized resource management
- Independent actors acting on self-interest produce desired global outcomes

  e.g. Spawn, Mariposa, G-commerce framework, Nimrod-G

- Increasingly important as we move to larger grids

## Example: Market-Based Task Service

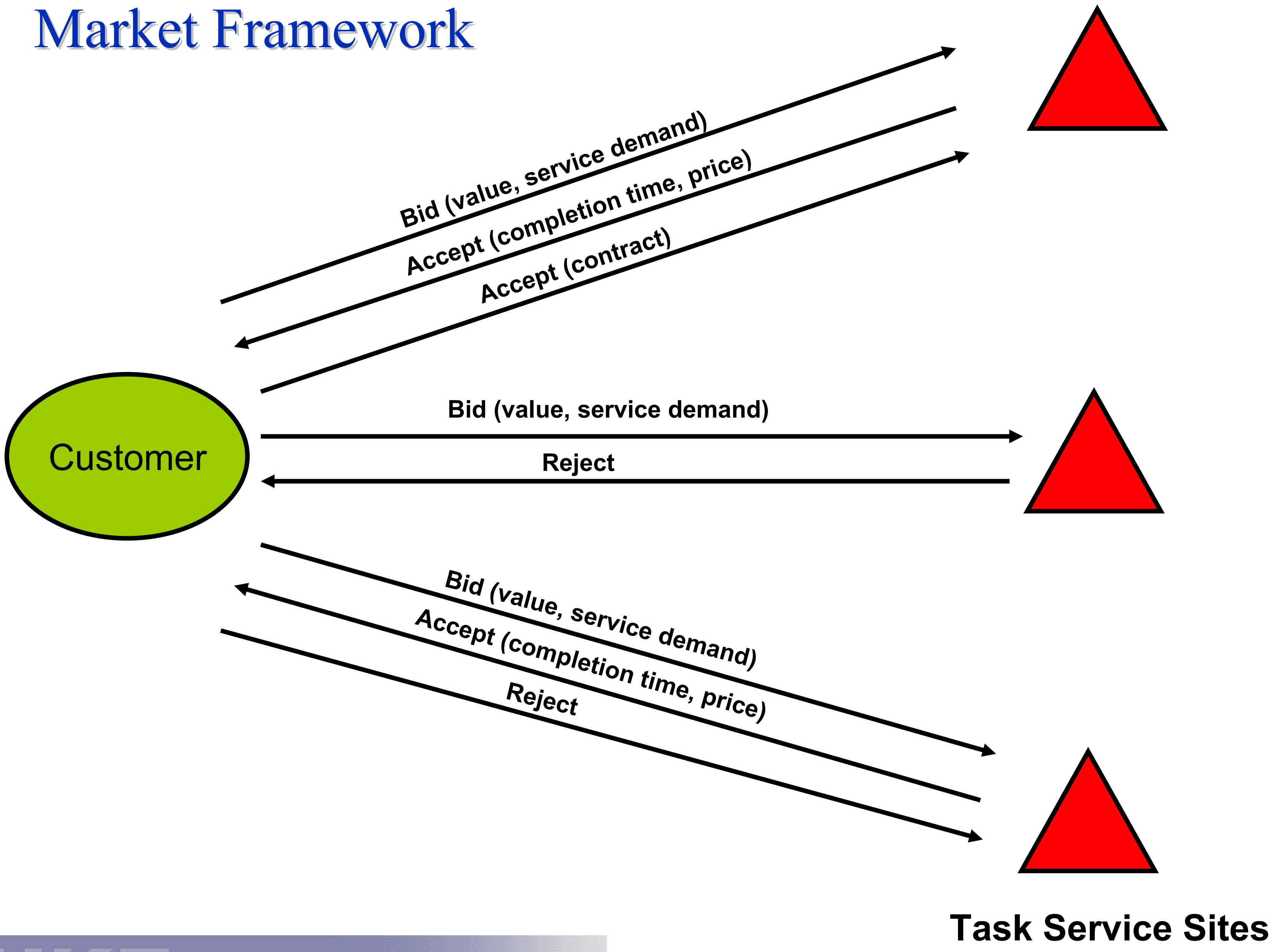
Tasks are batch computation jobs

- Self-contained units of work
- Execute anywhere
- Consume known resources

Characteristics of a market-based task service

- Tasks deliver value when they complete
- Negotiation between customers and task service sites
  - Value (price) and quality of service (completion time)
- Form contracts for task execution
  - Breach of contract implies a penalty
- Consumers look for the best deal; sites maximize their profits

## Market Framework


Bid (value, service demand)
Accept (completion time, price)
Accept (contract)
Customer
Bid (value, service demand)
Reject
Bid (value, service demand)
Accept (completion time, price)
Reject
Task Service Sites

## Goals and Non-goals

Goals

- Define profit-maximizing heuristics for acceptance (admission control) and scheduling for task service sites

  Which tasks should a site accept? When? For how much?

- Financial metaphor: balance risk and reward subject to user bids that trade off price and quality of service

Non-goals

- Other pieces for a fully functioning economy

  How is currency supplied and replenished?

  How to make payments and enforce contracts?

  How to propagate price signals to buyers?

  What incentive mechanisms will induce truthful user bids?

## Outline

## Specifying User Bids and Contracts

Negotiation establishes agreement on price and service quality

Use *value functions* giving an explicit mapping of service quality to value

- Need a representation that is **simple**, **rich**, and **tractable**
- Millennium: **linearly decaying** value functions [Chun02]

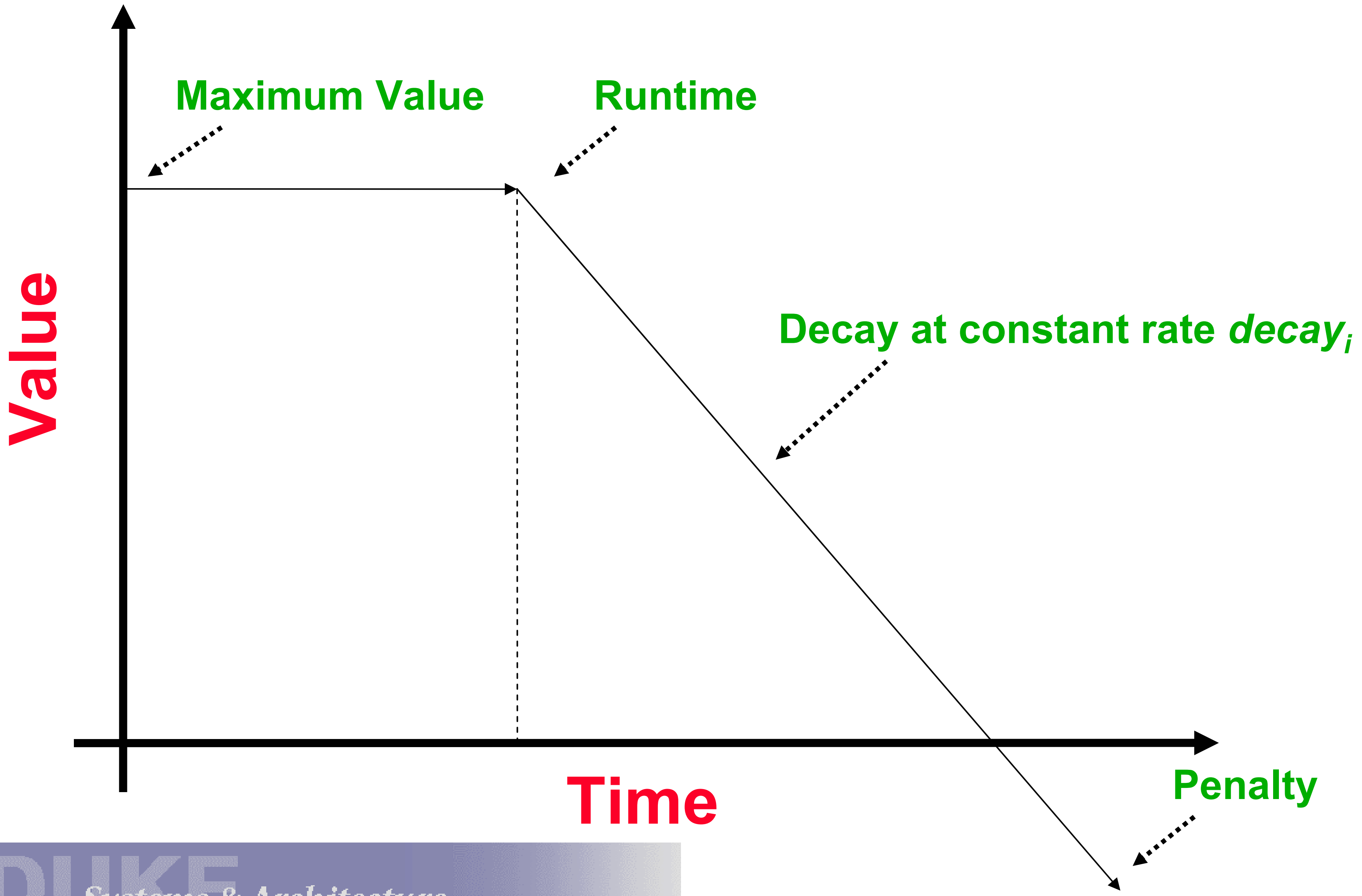
  Delayed tasks decay at constant rate of $decay_i$ (urgency)

Extend functions to include penalties

- May specify an optional *bound* on a penalty

  Penalties expire at time $expire_i$

# Example Value Function



# Problem Statement

Based on user bids we must decide…

- *Admission control*: which tasks to commit to?
- *Scheduling*: when to run a task?

  Schedule accepted tasks to maximize value
- How much to charge for tasks?

  Price to service quality tradeoff specified in value functions

Problem extends classical value-based scheduling problems

- Total Weighted Tardiness and Total Weighted Completion Time

  NP-hard for off-line instances; problem is difficult
- Examine on-line instances of the problem: need heuristics
- Site can negotiate for higher value or reject some tasks

## Server Scheduling Heuristics

Discounting future gains

- Bias schedule for shorter tasks
- Realizing gains quickly may be more important than value

Accounting for opportunity cost

- Bias towards high urgency tasks
- Account for losses in other tasks from a scheduling decision

Admission control

- High valued tasks earn value for the system
- High urgency tasks constrain the future task mix

## Experimental Methodology

Develop heuristics to maximize value and opportunity cost

- Heuristics have multiple components

Evaluate in on-line open market setting

- Schedule varying task mixes over emulated batch task engine
- Evaluate components in isolation and combination
- Compare with Millennium *FirstPrice* policy

Generated workloads to drive task sites

- Explore different areas of the parameter space

Focus on relative value and sensitivity analysis

## Workload Considerations

Workload characteristics

- Arrival and cost distributions representative of real batch workloads as characterized by previous studies
  - Exponential inter-arrival times and durations [Downey99]
- Previous studies give little guidance on how users value their jobs
  - Distribution of value and urgency similar to Millennium study [Chun02]
  - Adapt bimodal distributions for value and urgency
  - Characterize by *skew ratios*: ratio of high/low means

Magnitude of results dependent on workload characteristics

- Results are conservative: look at stable markets

## Discounting Future Gains

Account for risk of deferring future gains

- Example: Given two tasks with same unit gain and urgency it is preferable to run shorter task first
  - Shorter tasks carry lower risk of preempting newly arriving tasks
- Risk-averse scheduler may choose to run lower-yield task if it can realize gains quickly

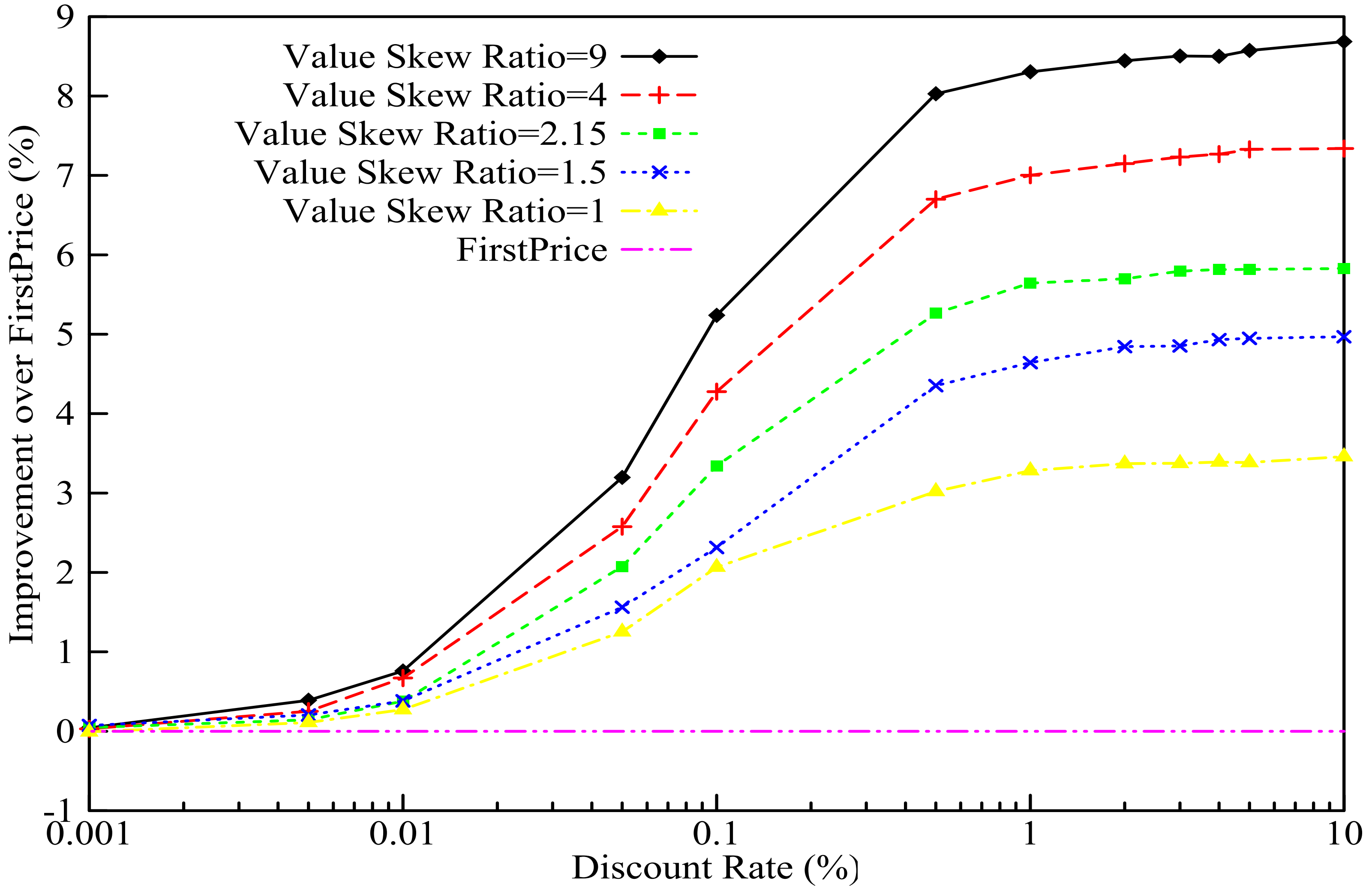
Approach based on notion of *present value* common in finance

- $PV_i = yield_i / (1 + (discount\_rate * RPT_i))$
- $PV_i$ represents investment value
  - Earning simple interest at *discount_rate* for $RPT_i$
- Higher *discount_rate* results in more risk-averse system

Present value heuristic (*PV)* selects jobs in order of discounted unit gain

- $PV_i/RPT_i$

# Improvement vs. Discount Rate



## Opportunity Cost

Extend heuristic to consider *opportunity cost*

- Losses occurring from choosing task *i* instead of task *j,* causing task *j* to decay in value
- Opportunity cost depends only on the urgency of competing tasks

Opportunity Cost: $cost_j = \sum_{j=0; j \neq i} decay_j * MIN(RPT_i, expire_j)$

- Bounded penalties requires *O($n^2$)* to compute least cost task
- We can simplify with unbounded penalties

  *Takes O(log n) to compute least cost task*

- Equivalent to Shortest Weighted Processing Time First (SWPT)

## Balancing Gains and Opportunity Cost

Risky to defer gains on basis of opportunity cost alone

- *FirstReward* metric combines task gains with opportunity cost

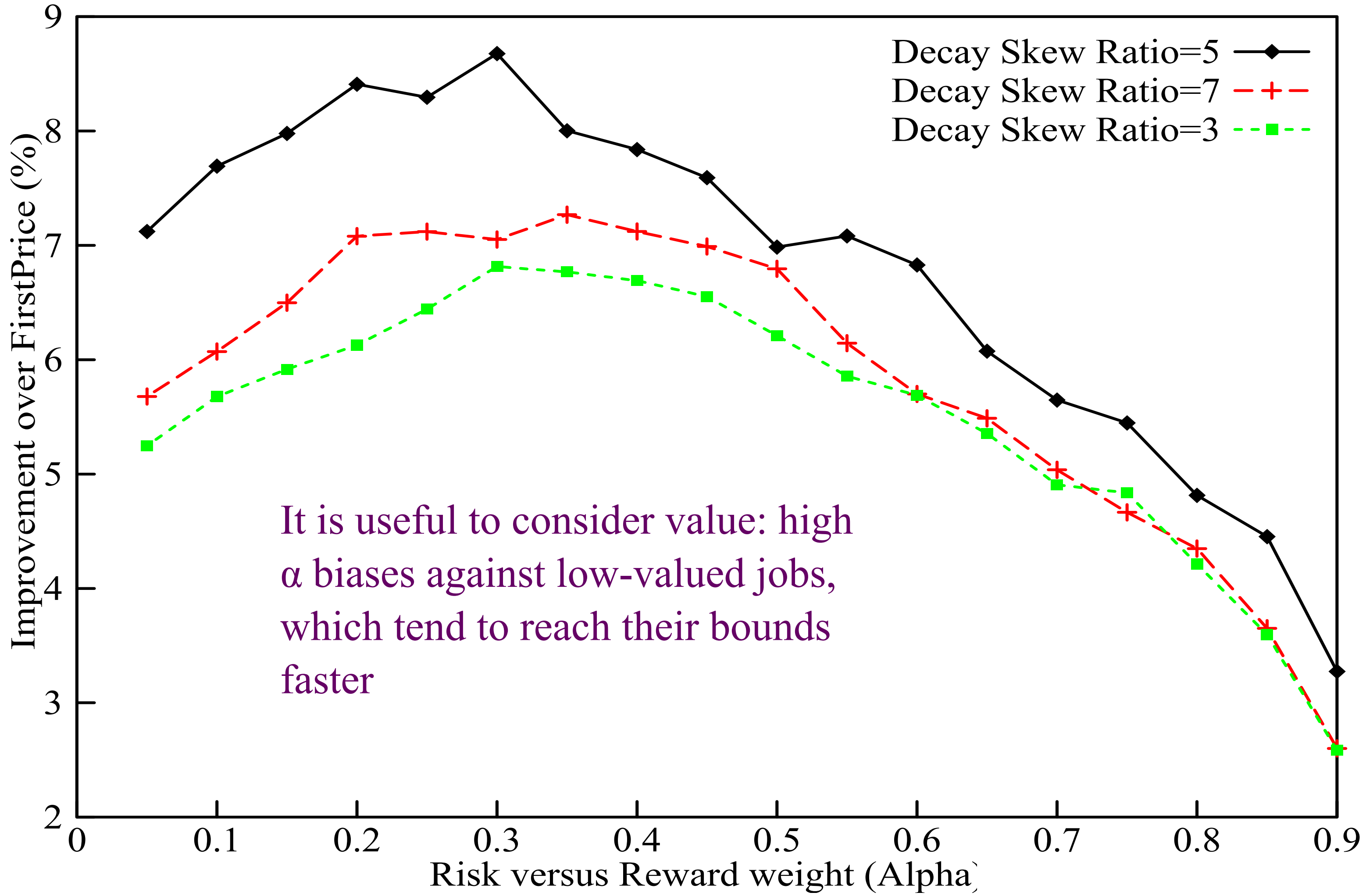
  $reward_i = ((\alpha)*PV_i - (1-\alpha)*cost_i)/RPT_i$

- α controls degree to which system considers expected gains

  With α=1 and *discount_rate = 0* $reward_i$ reduces to *FirstPrice*

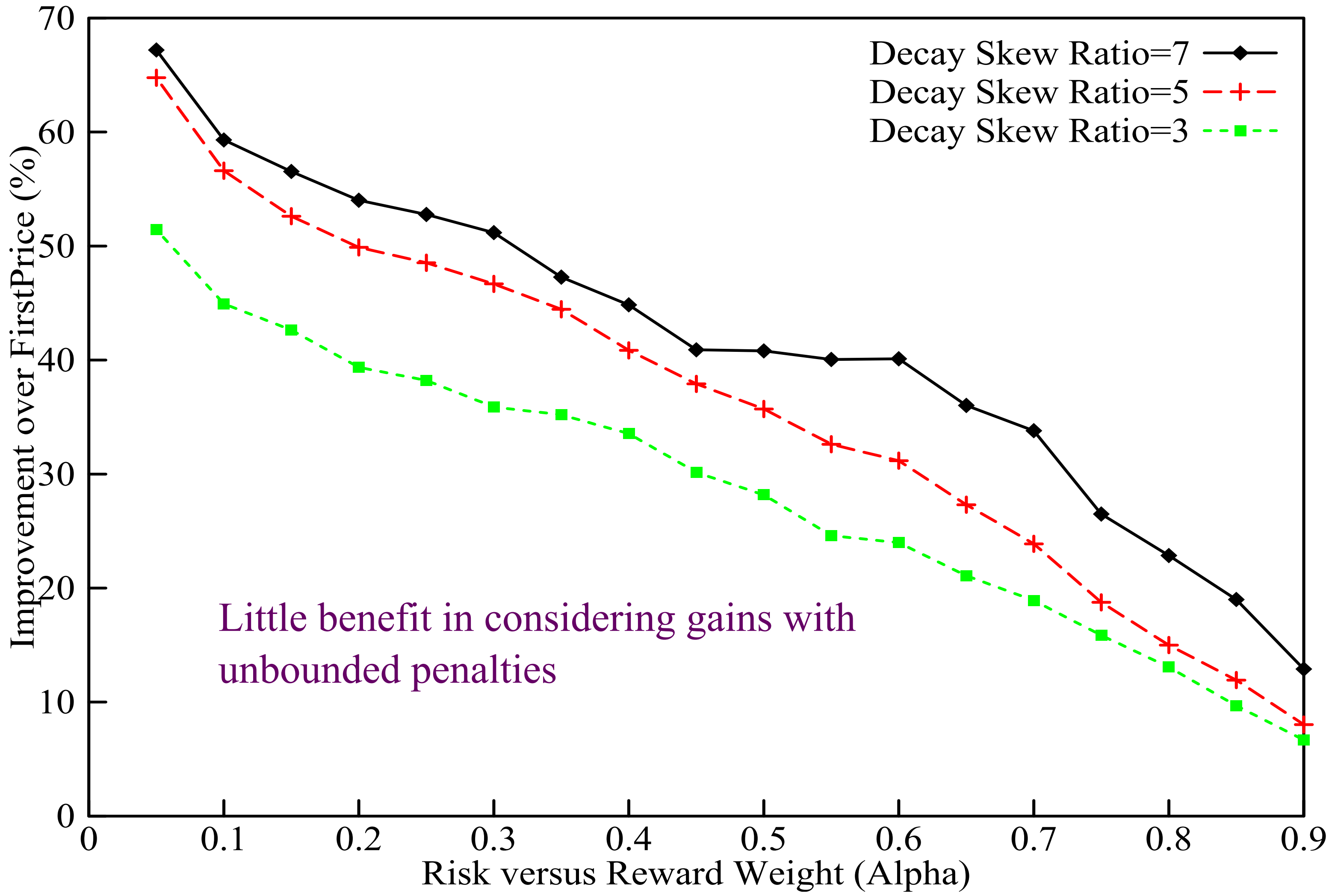
  With α=0 $reward_i$ reduces to a variant of SWPT

# Bounded: Improvement vs. Risk/Reward Weight



It is useful to consider value: high α biases against low-valued jobs, which tend to reach their bounds faster

# Unbounded: Improvement vs. Risk/Reward Weight



Little benefit in considering gains with unbounded penalties

## Negotiation and Admission Control

Each site may accept or reject a task

- Accepted tasks negotiate to establish a price and expected completion time

Admission control procedure

- Integrate task into current schedule according to heuristic
- Determine expected yield for task if completed
- Apply *acceptance heuristic* to determine acceptance
- If task is profitable then accept the bid and issue a server bid to client
- If client accepts the contract then execute the task

Later arrivals could delay task beyond its expected completion time

## Admission Control Heuristic

Acceptance Heuristic

- Consider potential reward and constraining future task mix
- Urgent tasks incur more risk

Heuristic based on task's *slack*

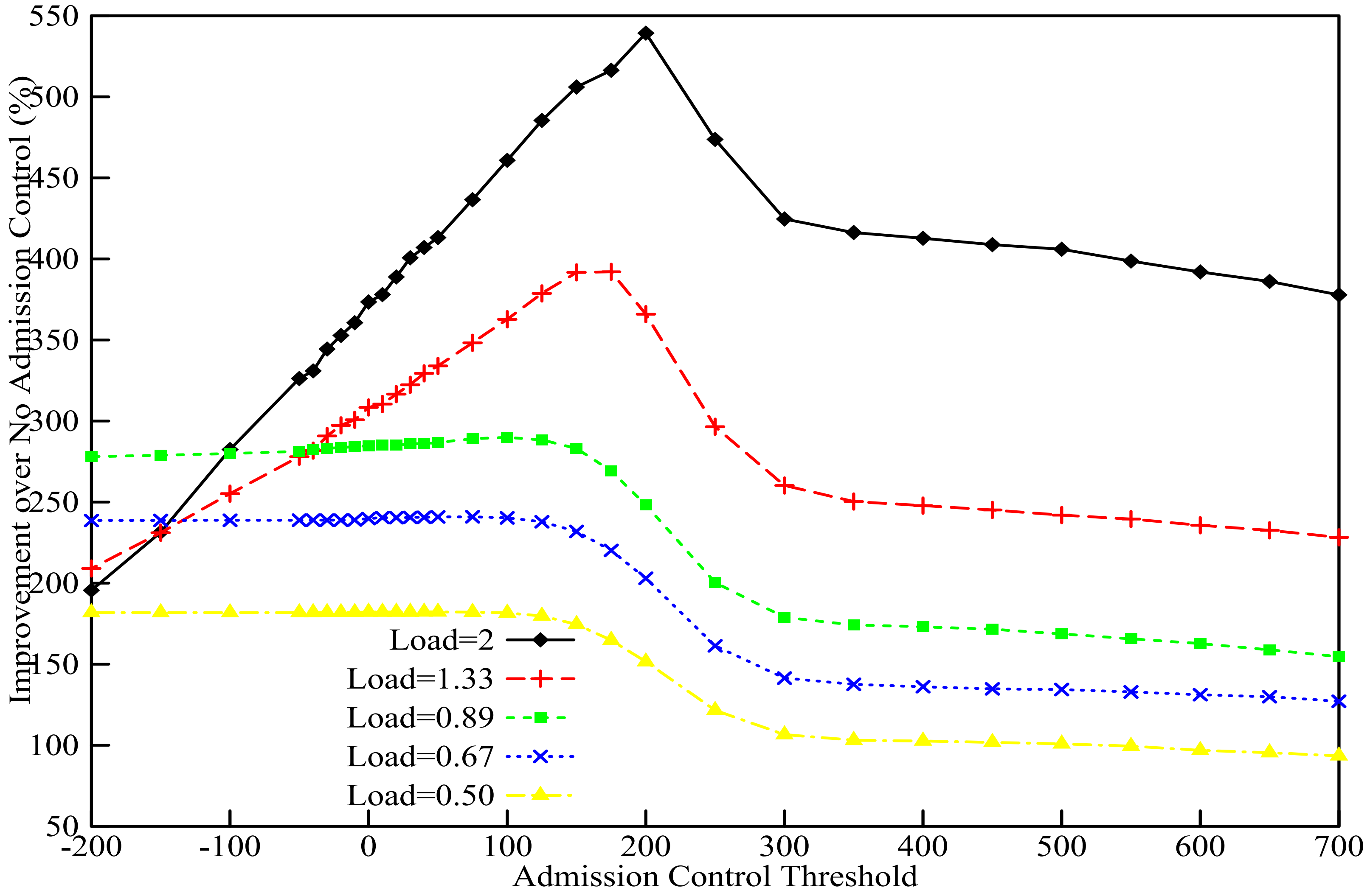
- *Slack* is the amount of additional delay that the task can incur before its reward falls below some yield threshold

  $Slack_i = (PV_i - cost_i)/decay_i$

- Policy rejects tasks whose slack falls below some *slack threshold*

Slack captures the risk of accepting tasks as determined by its decay rate and position in the schedule

# Improvement vs. Admission Threshold



## Conclusions

Develop heuristics for market based task scheduling and admission control

Bids capture both user value and urgency

- Approach based on a financial metaphor
- Cost and risk often more important than gains
  - Heuristics that consider gains are effective in some cases

Contributions

- Detail different areas of scheduling risk
- Explore parameter space for a general scheduling heuristic
- Show how value-based schedulers can drive server bidding and admission control in a computational economy

## Questions