Service Adaptation in Open Grid Platforms

Krishnaveni Budati, Jinoh Kim, Abhishek Chandra and Jon Weissman

Department of Computer Science
Digital Technology Center

University of Minnesota
ridge.cs.umn.edu

NFS NGS Support

Background

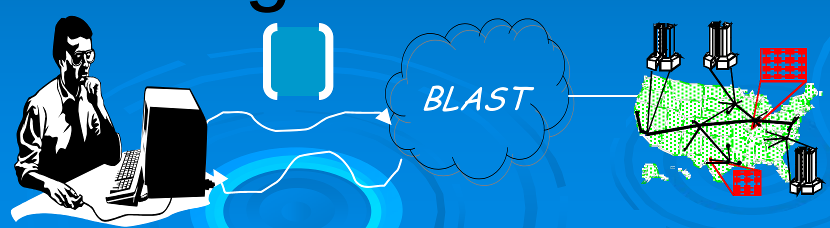
- High-end Network Services
  - Data-rich && Compute-rich && End-client
    - image matching: “is this brain-scan cancer?”
    - genomics: “run BLAST on my target sequence DB”
    - parallel/distributed services (per request)
- Public donation-based “Open” Grid infrastructures
  - positives: cheap, scalable, fault tolerant
  - negatives: uncertainty => best effort service

The Challenge

- Uncertainty at many levels
  - service demand
  - resource availability (CPU, network, storage)
  - resource capacity
  - resource integrity
- Service must adapt to these uncertainties

The Challenge (cont'd)

- Provide and maintain service quality despite uncertainty
- Dimensions of quality
  - **performance** (latency, throughput, response)
  - availability/**reliability**
  - accuracy
  - …
- Today: minimizing service makespan



Problem

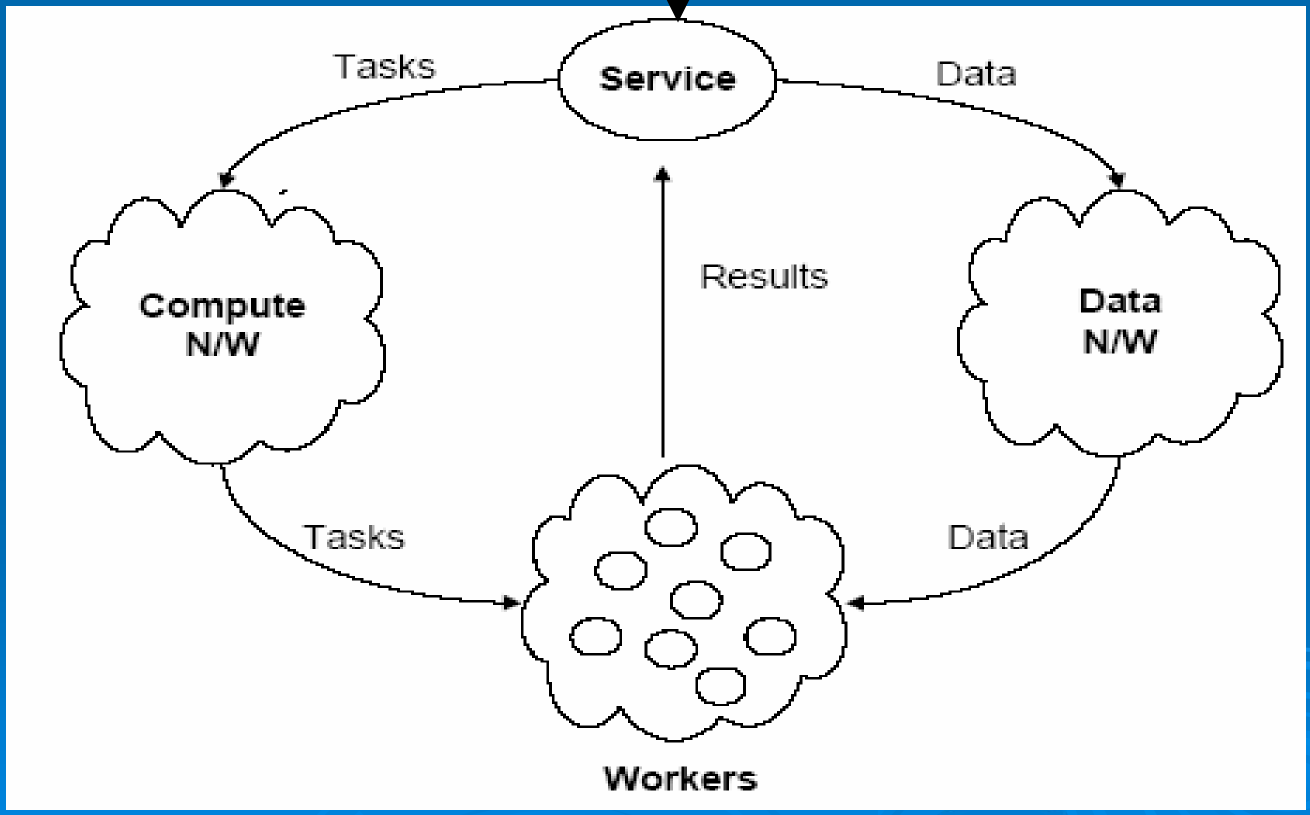
- Makespan challenges
  - Communication
    - efficient data download despite network and data server behavior – variable latency, b/w, capacity
  - Reliable Computation
    - efficient execution despite “imperfect” node behavior – slow, hacked, cheating

Context: System Model

end-clients

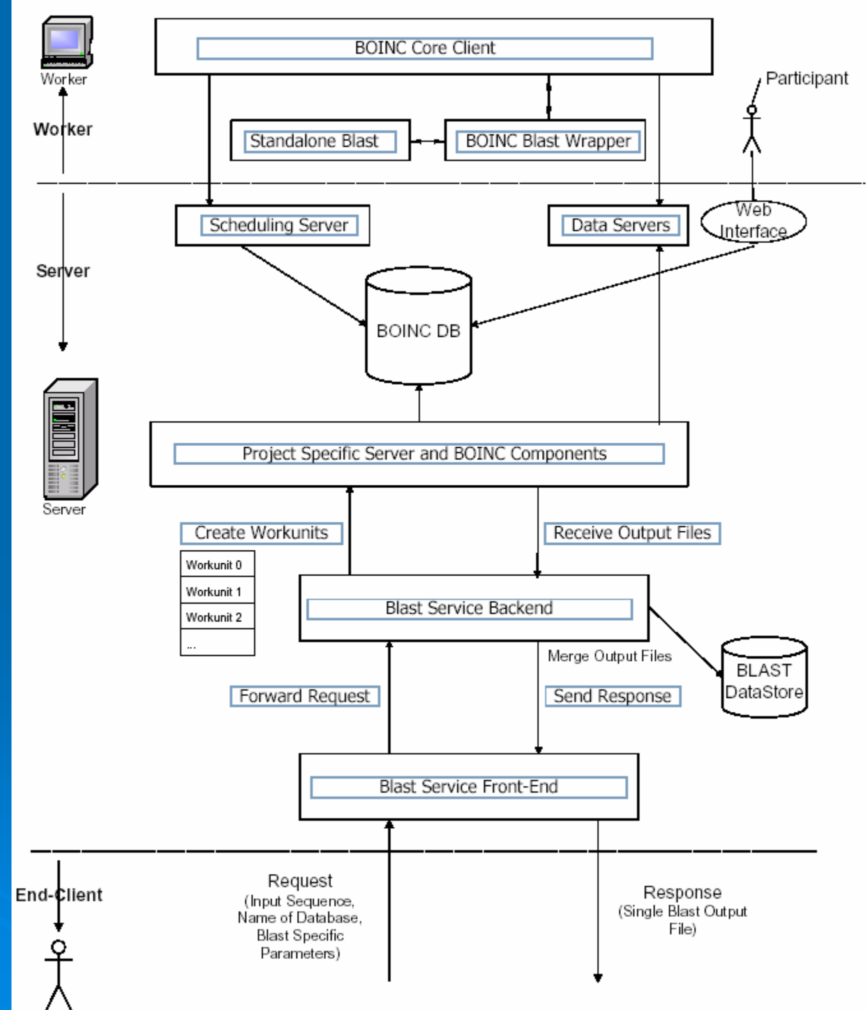
tasks are decomposed
and replicated per request

data is decomposed
and replicated



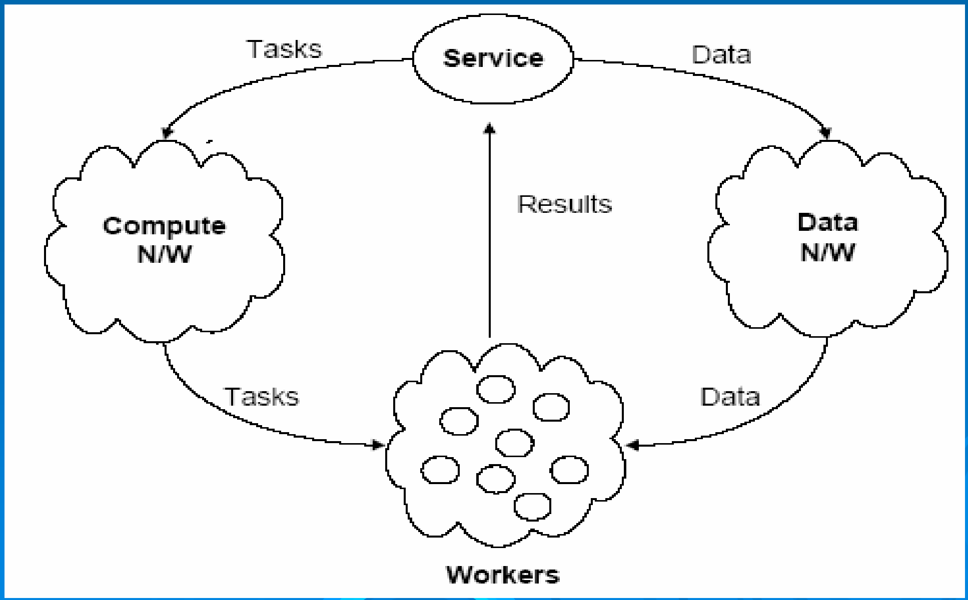
RIDGE

- Host services on BOINC
- Deployed on PlanetLab
  - 30-120 nodes
- Open Grid Emulation



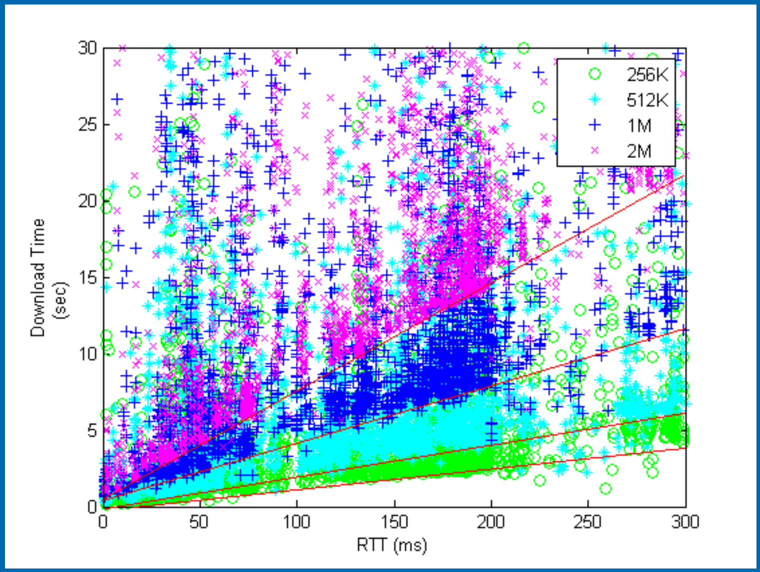
Communication Makespan

- Worker nodes download data from replicated servers
  - Worker nodes choose “servers” independently
  - **M**inimize the maximum download time for all worker nodes (*communication makespan*)

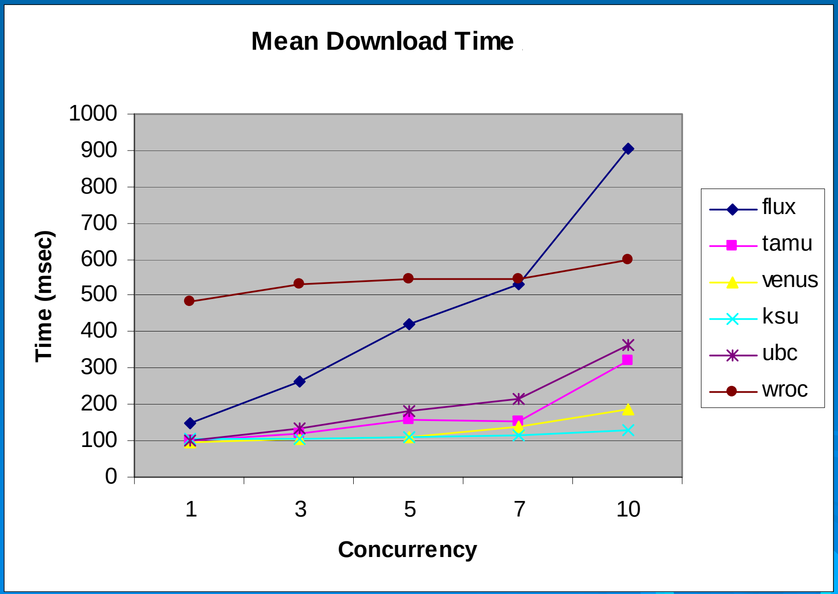


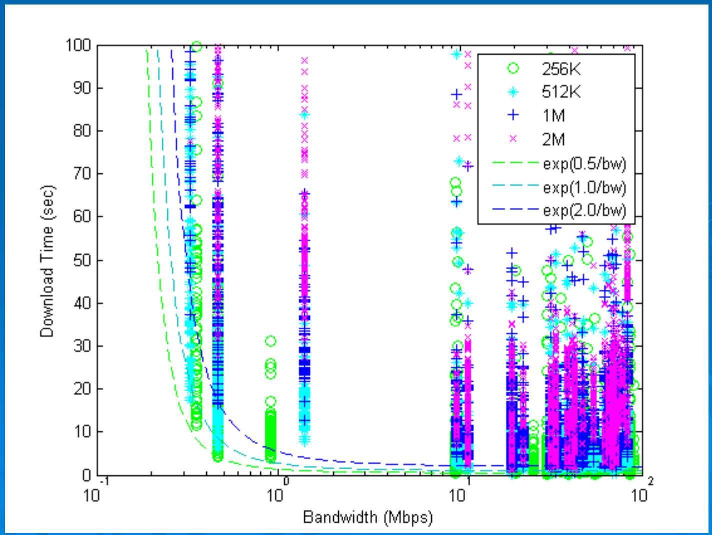
Server Selection

- Several possible factors
  - Proximity (RTT)
  - Network bandwidth
  - Server capacity
  - Server reliability



[Download Time vs. RTT - linear]





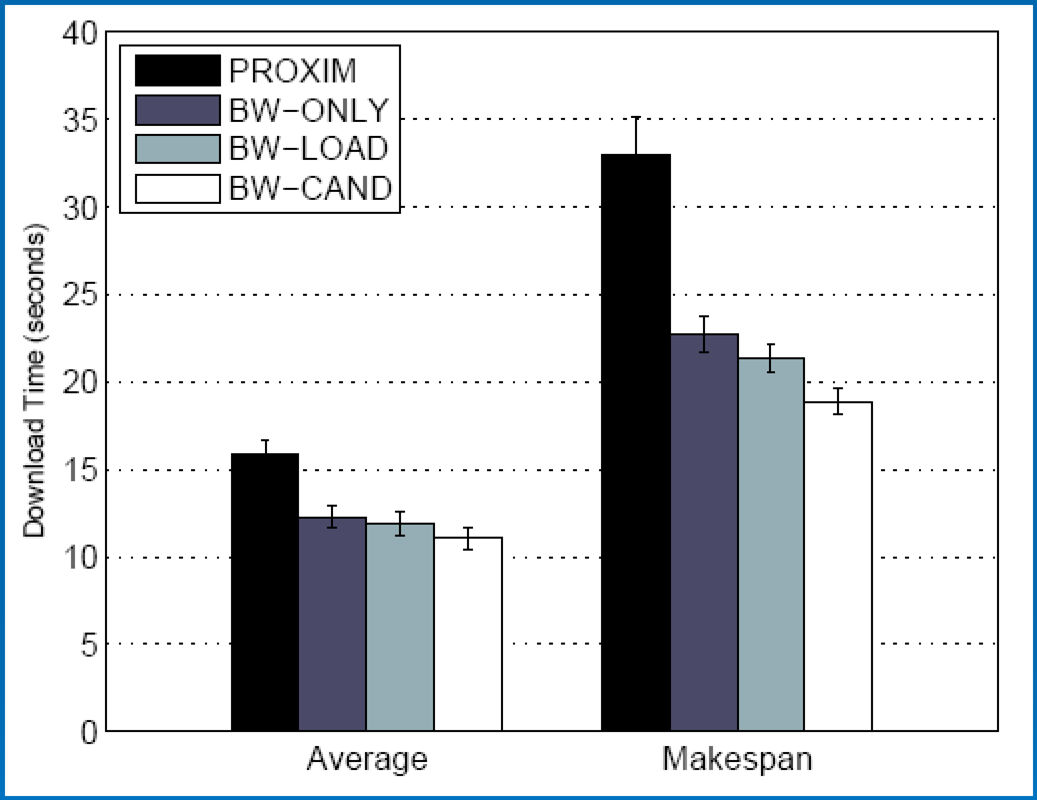
[Download Time vs. Bandwidth - exp]

Heuristic Ranking Function

- Worker i, server j
  - Cost function $f_{i,j} = \alpha_j * rtt_{i,j}$
  - Weight $\alpha_j = exp(k_j / bw_j)$
- k_j accounts for load/capacity
- Least cost server selected independently

- Three server selection heuristics that use k_j
  - BW-ONLY: $k_j = 1$
  - BW-LOAD: k_j = *n-minute average server load (past)*
  - BW-CAND: k_j = *# of candidate server responses in last m seconds (~ future load)*

Performance Comparison



Parameters:

Data: 2MB

Replication: 10

Candidates: 5

Benefit of BW-CAND grows as query rate increases

Take away

- Bandwidth, proximity, load, and capacity all matter

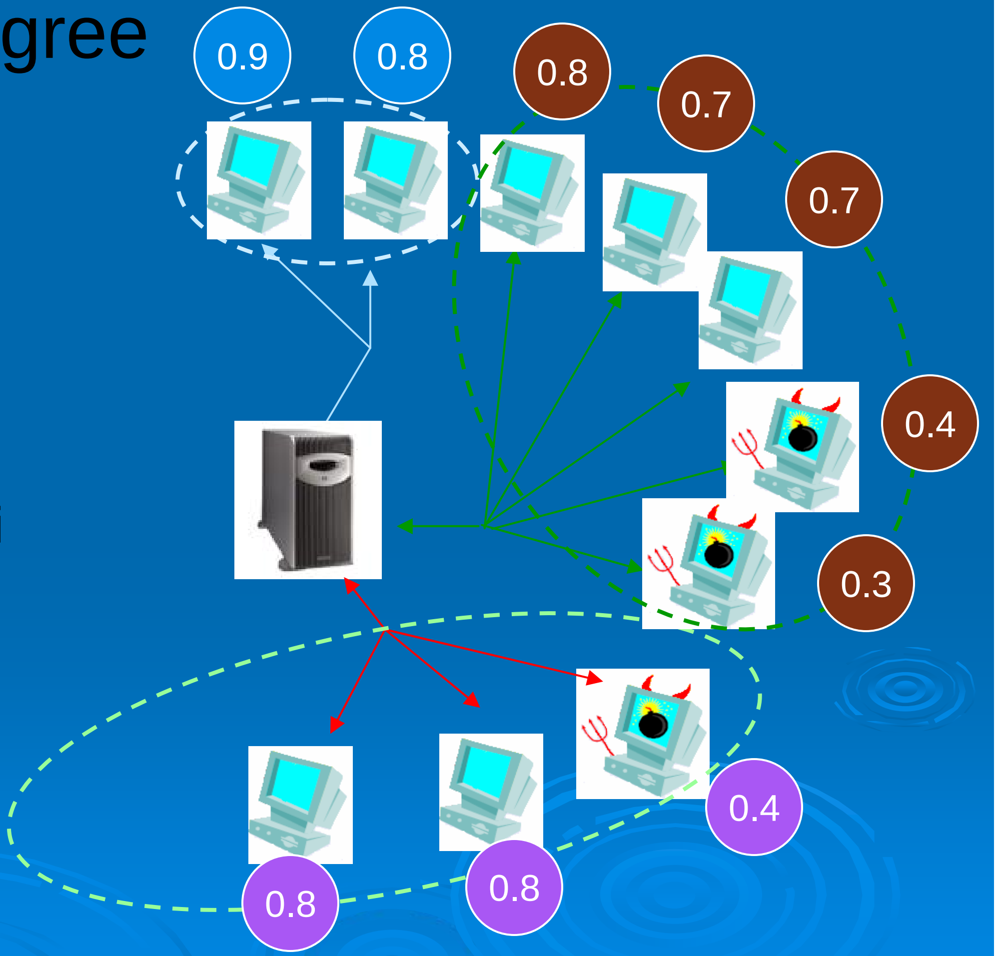
- Both heterogeneity and load balancing must be taken into account

Reliability

- Voluntary systems are not reliable
  - node churn, cheating, untimely
- Solution:
  - replicate, compare answers, and vote
- Ad-hoc redundancy wastes resources threatens correctness
- Idea: Use smarter redundancy to maximize efficiency

Reputation-based Scheduling

- Reputation rating R_i – degree of client reliability wrt correctness/timeliness
- Smarter redundancy – dynamically size the redundancy based on R_i
- Adapt to changes in R_i

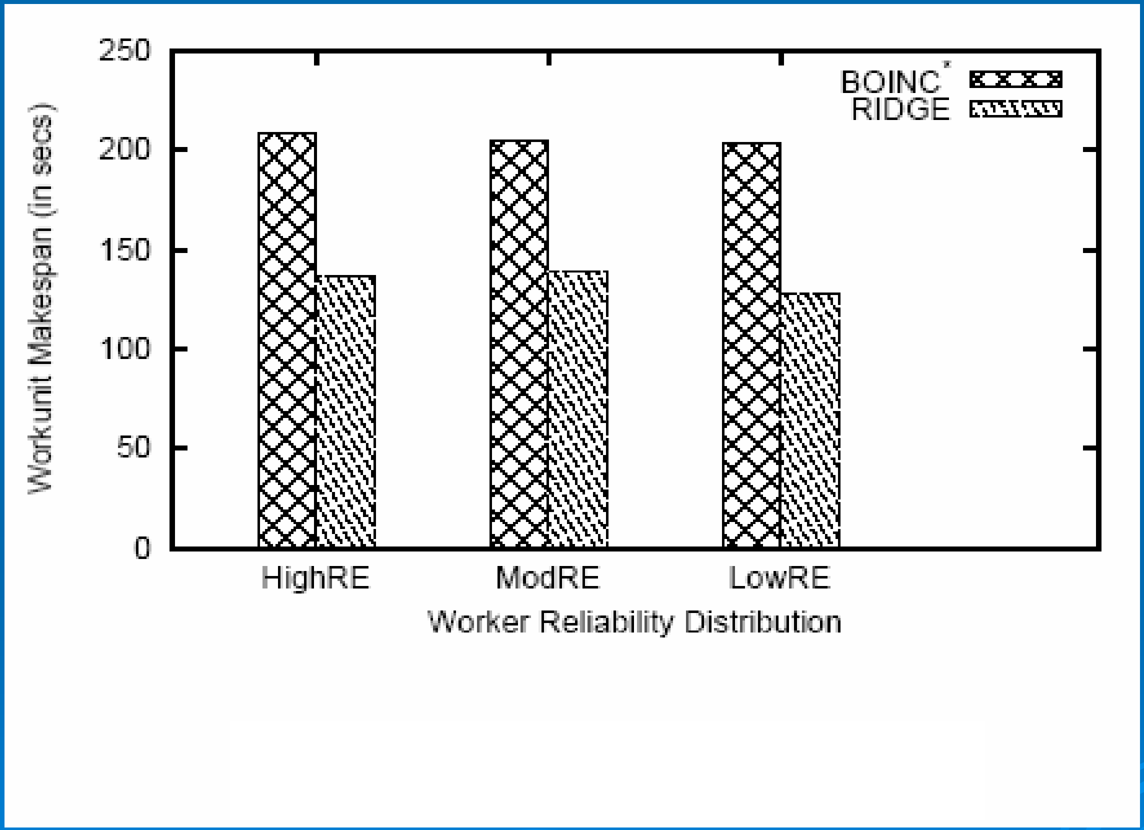


Scheduling Algorithms

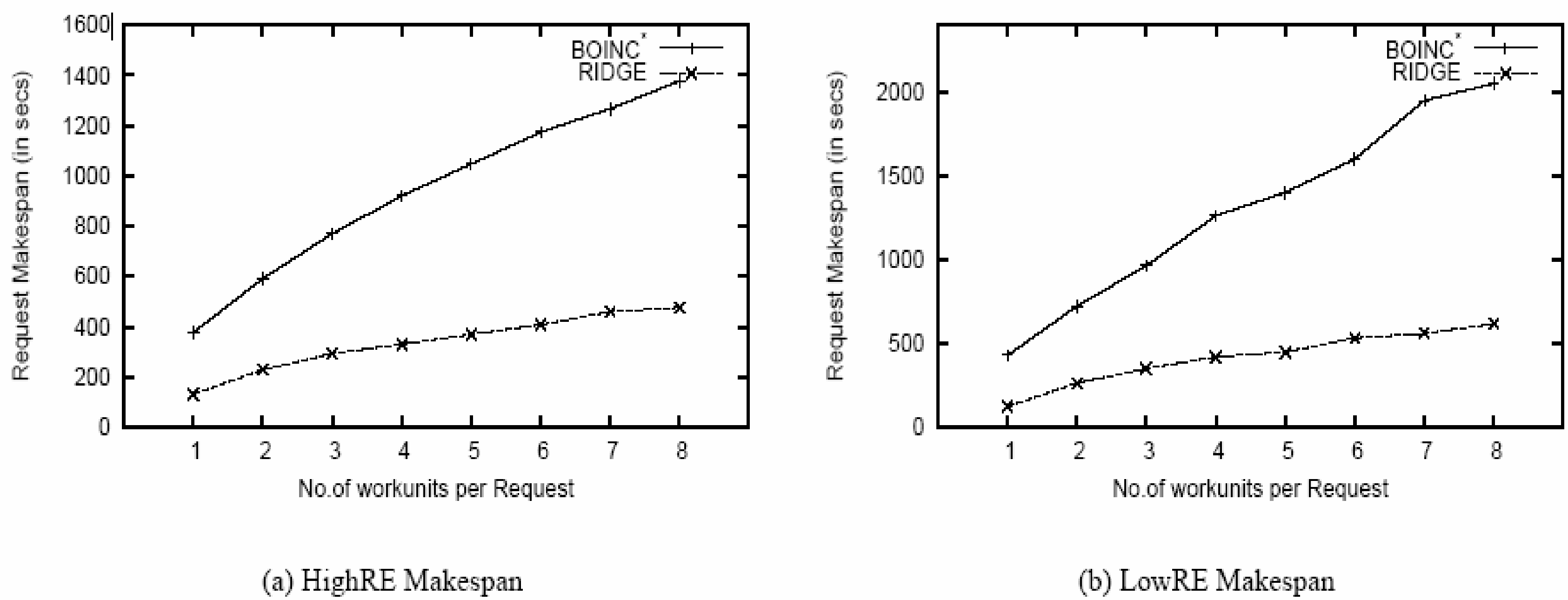
Algorithm 3 Best-Fit (w worker-list, τ task-list, λ_{target} target LOC, R_{min} min-group-size)

Sort the list w of all available workers on the basis of the reliability ratings r_i
while $|w| \geq R_{min}$ **do**
  Select task τ_i from τ
  Search for a set s of n workers w_n from w such that λ_s exceeds λ_{target} minimally
  if such a set s is found **then**
    Assign the w_n workers to G_i
  else
    Select the set of n workers s for which $\lambda_{target} - \lambda_s$ is minimized
    Assign the w_n workers to G_i
  end if
  $w \leftarrow w$ - w_n
end while

Makespan: RIDGE vs. BOINC (task level)



Makespan Comparison (request level)



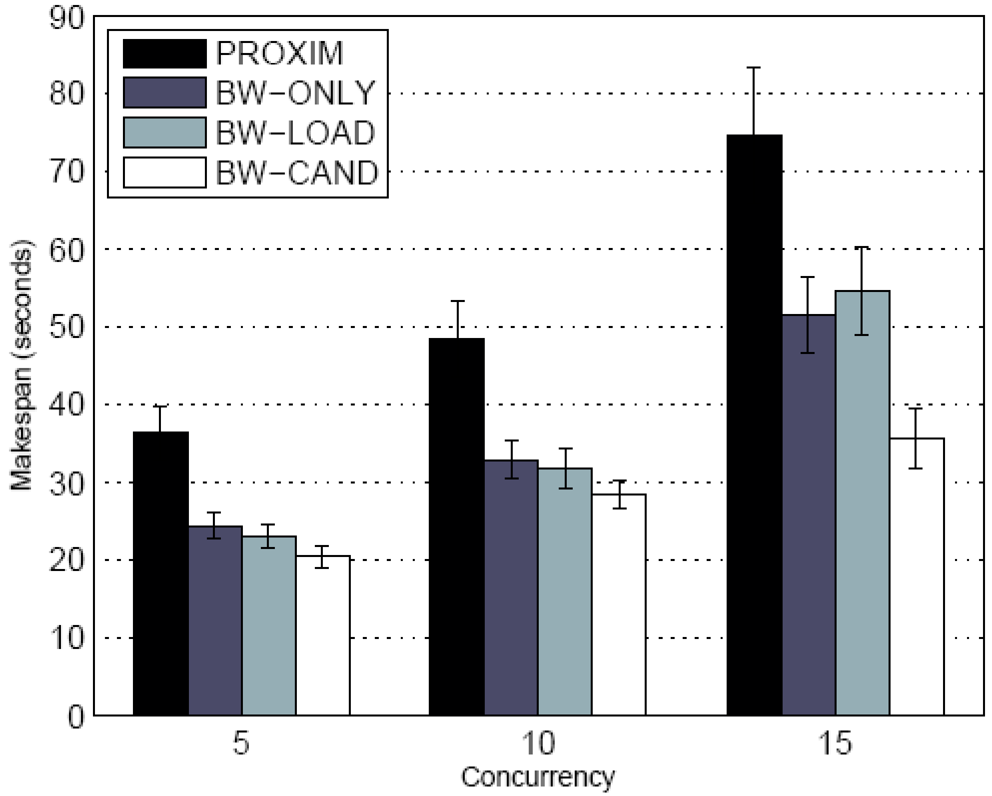
Comparison of Request Makespan for different reliability environments

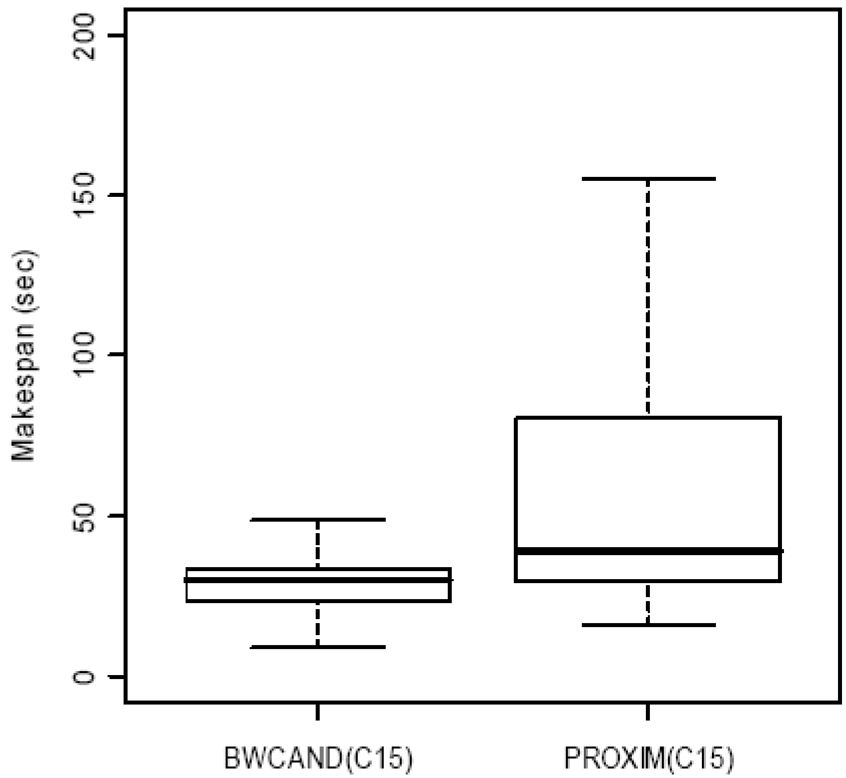
Summary

- Service hosting + Open Grids
- Request performance dependent on communication (to get data), computation (to process data)
- Uncertainties impact makespan
- RIDGE is developing techniques to smooth out these uncertainties
- Papers: CCGrid07, ICDCS06, HPDC07, TPDS07
- Website: ridge.cs.umn.edu

Questions?

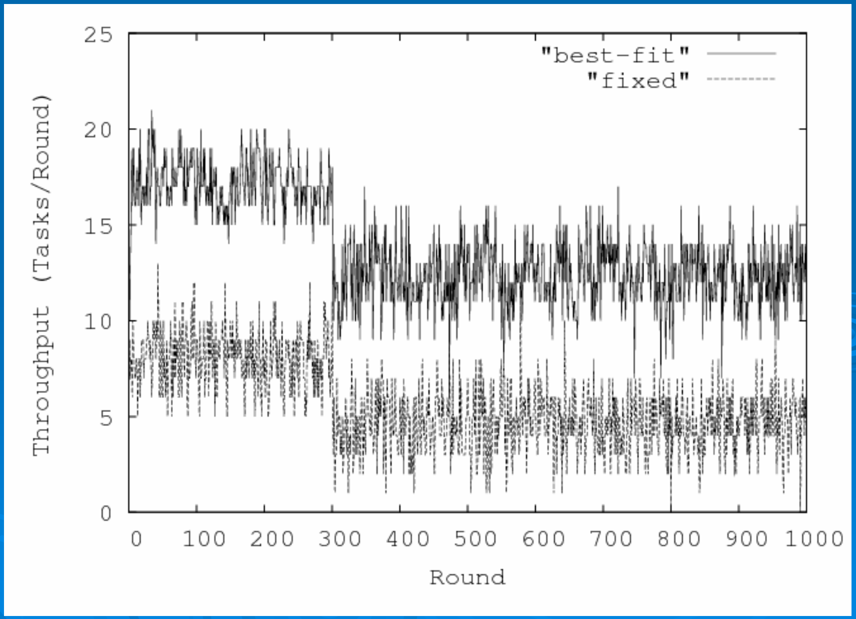
Impact of Load



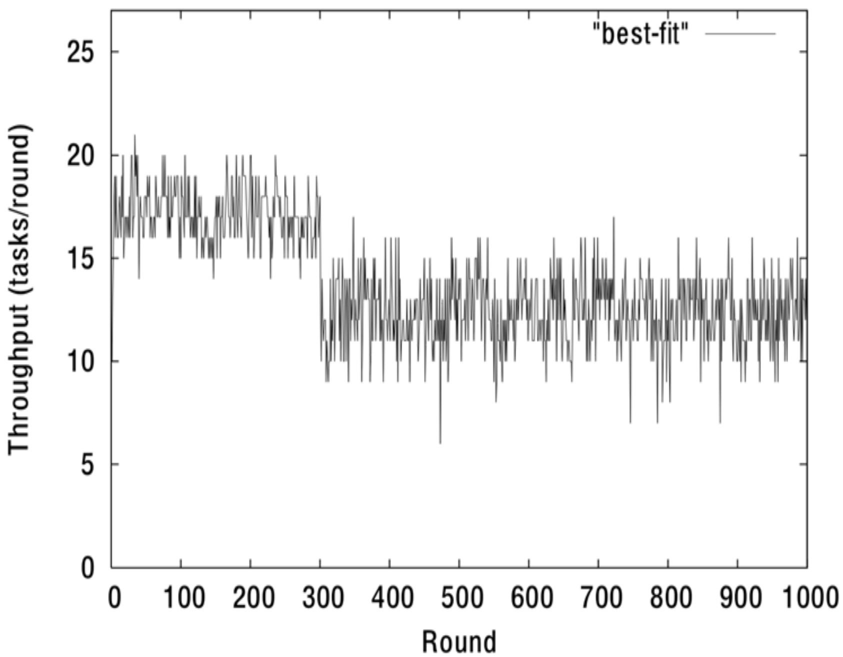


Non-stationarity

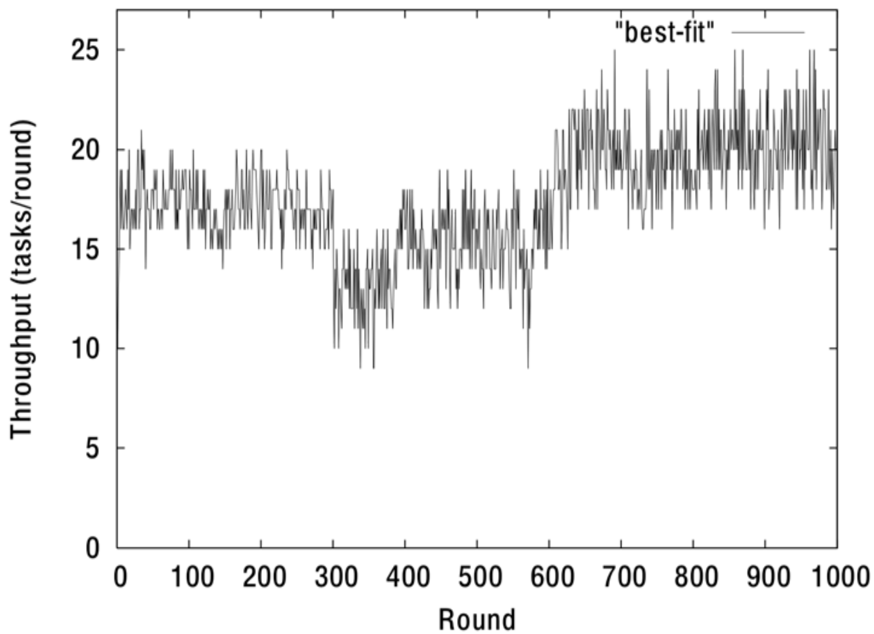
- Nodes may suddenly shift gears
  - deliberately malicious, virus, detach/rejoin
  - underlying reliability distribution changes
- Solution
  - window-based rating
  - adapt/learn λ_{target}
- Experiment: blackout at round 300 (30% effected)



Adapting …



"best-fit"
Throughput (tasks/round)
Round
0
5
10
15
20
25
0
100
200
300
400
500
600
700
800
900
1000



"best-fit"
Throughput (tasks/round)
Round
0
5
10
15
20
25
0
100
200
300
400
500
600
700
800
900
1000

Algorithm 1 UpdateTargetLOC (λ_{target} target LOC, α throughput-weight)

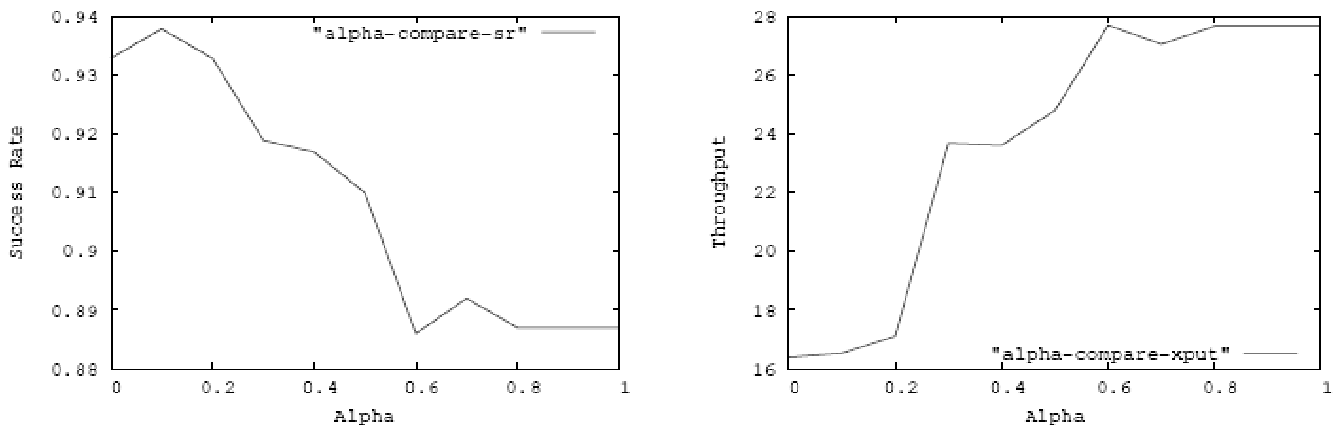
```
Local variables: $s$ state, $d$ direction

if (round % $p$) = 1 ... $p$-1 then
   Update measures of mean normalized throughput $xp$ and success rate $sr$
else
   if $s$ = CONVERGING then
      if round = $p$ then
         Set initial direction $d$ based on mean client reliability
         $j \leftarrow 4$
      end if
      $G_{last} \leftarrow G$
      Gain $G \leftarrow \alpha$ * $xp$ + (1-$\alpha$) * $sr$
      if $G$ / $G_{last} \geq delta_{mod}$ then
         $j \leftarrow j-1$
         Switch direction $d$
         if $j = 0$ then
            $s \leftarrow$ STEADY-STATE
         end if
      else if $G > G_{avg}$ OR $G$ / $G_{last} \leq delta_{in}$ then
         if $d$ = left then
            $\lambda_{target} \leftarrow \lambda_{target}$ - 0.01$*j$
         else
            $\lambda_{target} \leftarrow \lambda_{target}$ + 0.01$*j$
         end if
      else
         $\lambda_{target}$ unchanged
         if $\lambda_{target}$ unchanged for $maxrounds$ rounds then
            $s \leftarrow$ STEADY-STATE
         end if
      end if
   else
      Gain $G \leftarrow \alpha$ * $xp$ + (1-$\alpha$) * $sr$
      if $G$ / $G_{last} \geq delta_{sig}$ then
         $s \leftarrow$ CONVERGING, $j \leftarrow 4$
      end if
      $G_{avg} \leftarrow weight_{curr}$ * $G$ + $weight_{hist}$ * $G_{avg}$
   end if
end if
```

$$G(\rho, s) = \alpha \cdot \overset{\text{throughput}}{\rho} + (1-\alpha) \cdot \overset{\text{success rate}}{s},$$

Reputation-based Scheduling

- Reputation rating
  - Techniques for estimating client reliability based on past task executions
- Reputation-based scheduling algorithms
  - Using reliabilities for allocating work
- Adaptive scheduling algorithms
  - Dynamically tune system parameters in the presence of changing reliability conditions

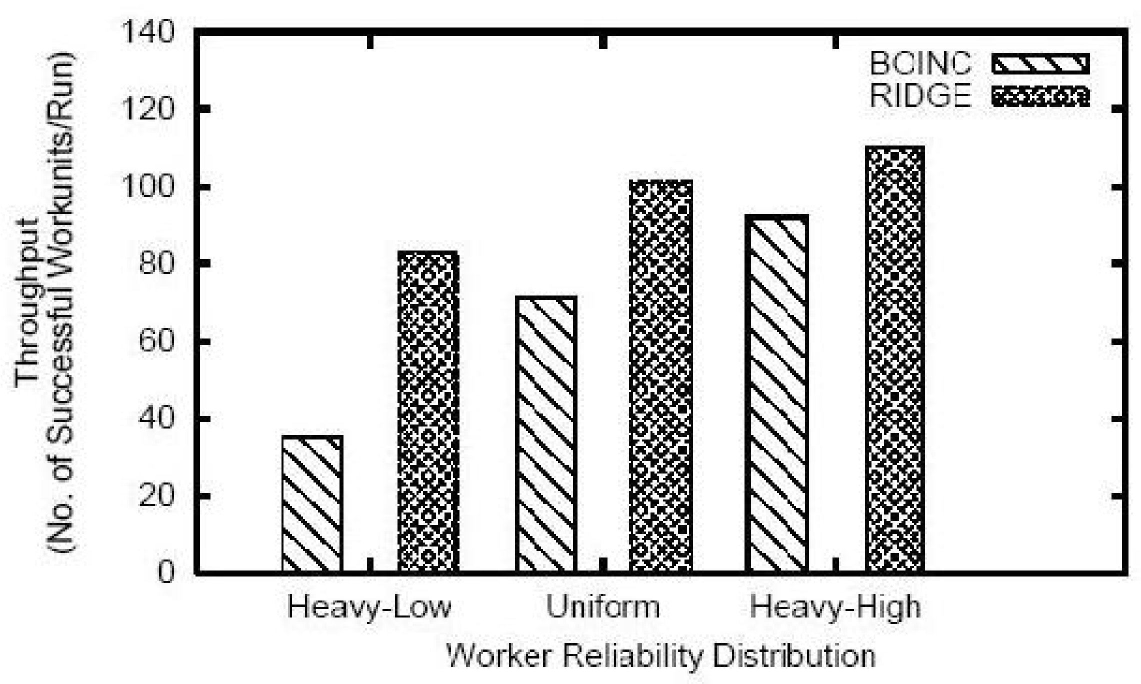
Adaptive Algorithm



(a) Success Rate (b) Throughput

Fig. 9. Comparison of throughput/success rate achieved using adaptive algorithm with varying α

Experimental Result I – Throughput

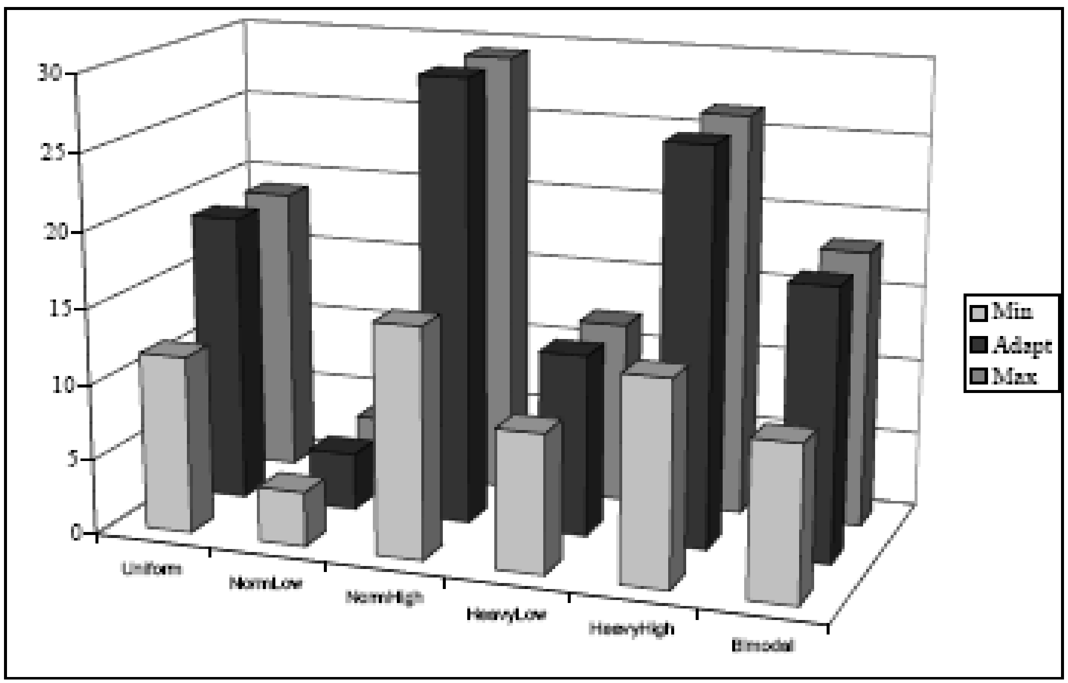


Grid Size = 32
Threshold Size = 15
BOINC Replication= 5
RIDGE Replication= 3-5
Scheduling Algorithm – Best-Fit

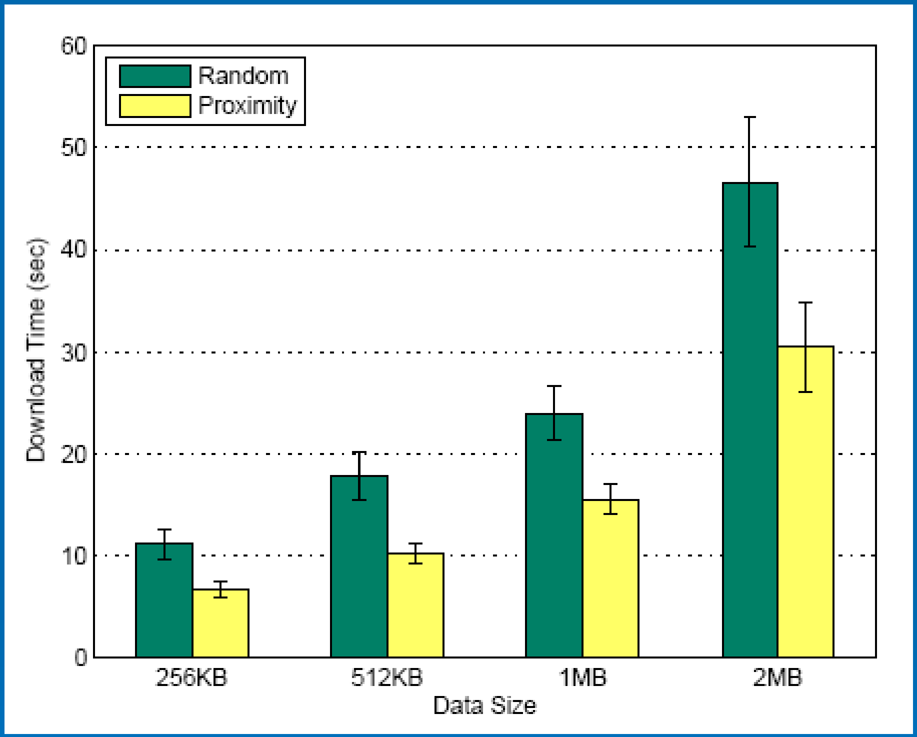
Adaptive Algorithm

throughput success rate

$$G(\rho, s) = \alpha \cdot \rho + (1 - \alpha) \cdot s,$$



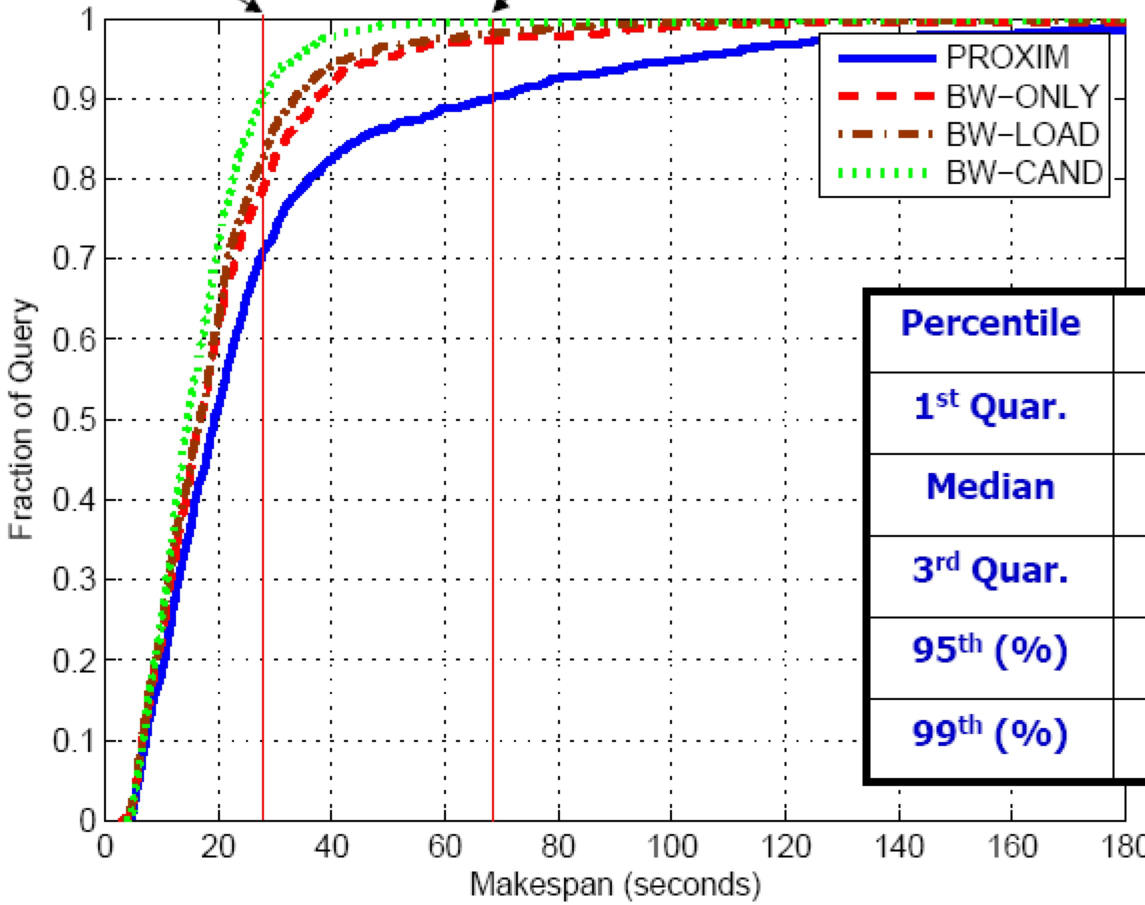
Proximity – keep?



Pretty good, but can do better

BW-CAND 90% Completion (~ 30 sec)

PROXIM 90% Completion (~ 70 sec)



(Unit: second)

Percentile	PROXIM	BW-CAND
1st Quar.	13.3	10.9
Median	21.9	16.3
3rd Quar.	33. 0	23.2
95th (%)	95.4	36.4
99th (%)	192.6	59.1

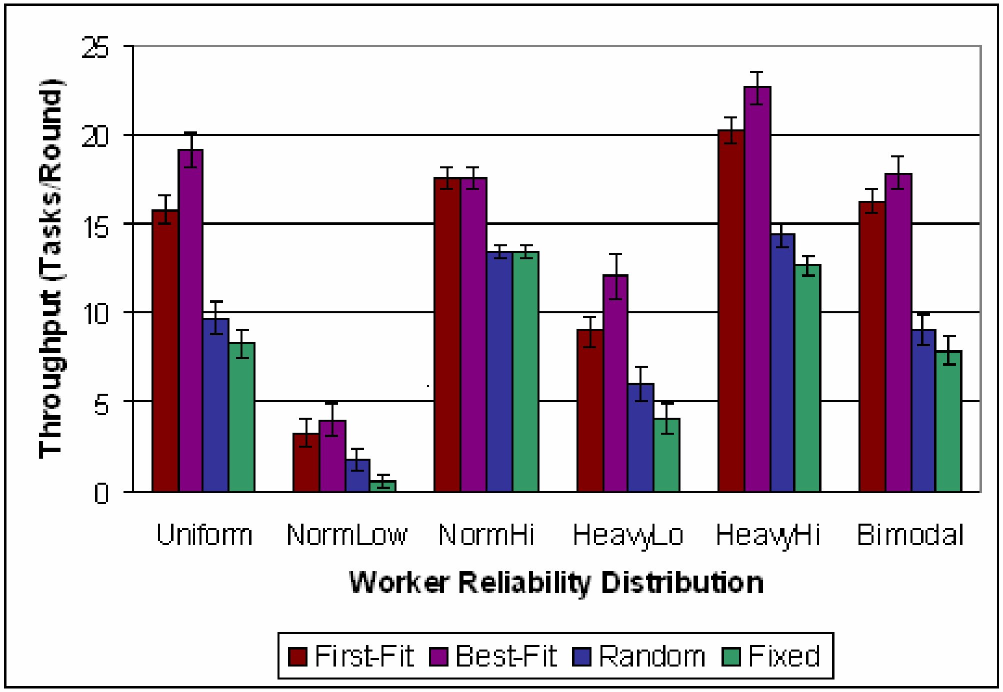
Future Work

- Resource Collectives

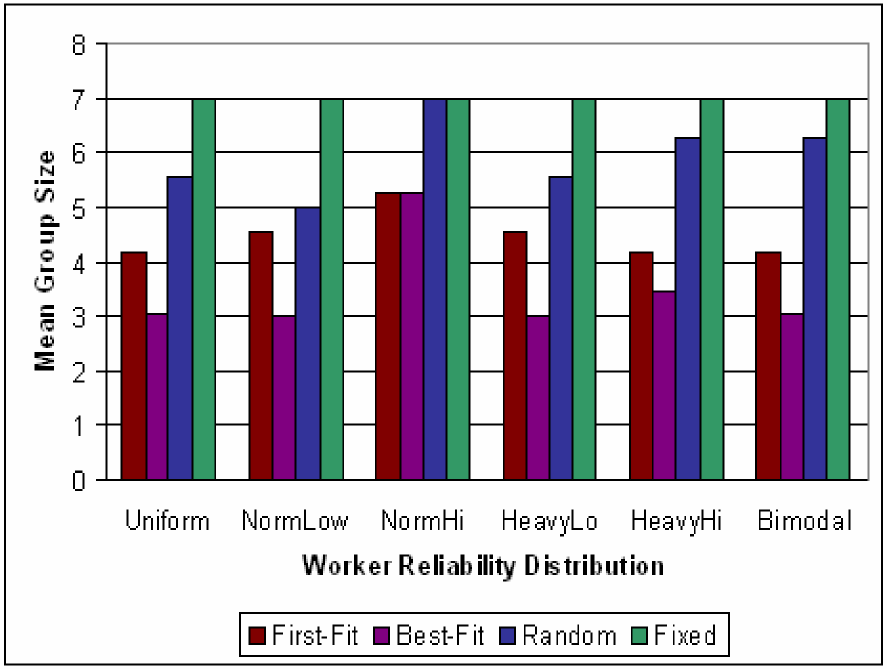
Evaluation

- Baselines
  - **Fixed** algorithm: statically sized equal groups uses no reliability information
  - Random algorithm: forms groups by randomly assigning nodes until λ_{target} is reached
- Simulated a wide-variety of node reliability distributions

Comparison

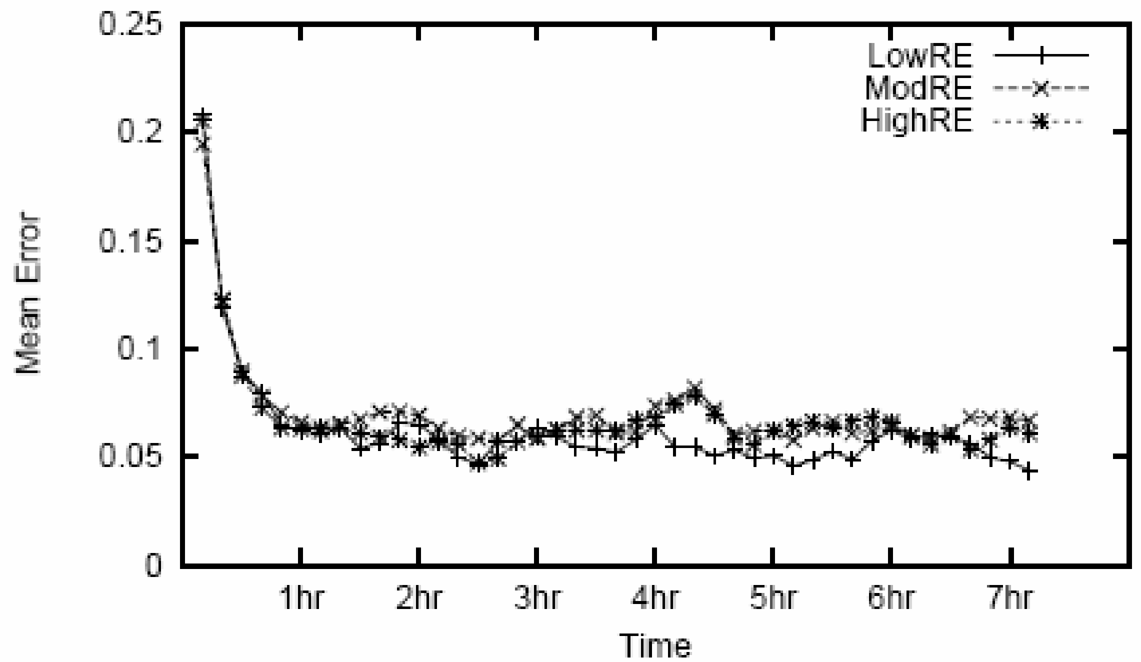


Throughput (Tasks/Round)
25
20
15
10
5
0
Uniform
NormLow
NormHi
HeavyLo
HeavyHi
Bimodal
Worker Reliability Distribution
First-Fit
Best-Fit
Random
Fixed



Mean Group Size
8
7
6
5
4
3
2
1
0
Uniform
NormLow
NormHi
HeavyLo
HeavyHi
Bimodal
Worker Reliability Distribution
First-Fit
Best-Fit
Random
Fixed

Learning Rate



Learning Behavior of RIDGE

- Why open systems?
  - Or why not provisioned properly with sharing built in?
  - Lower TCO, not incompatible – could have a cluster and a mix of voluntary resources (support diff. classes of users)
- Bullet