

Andrew Page
apage@cs.nuim.ie

March 30, 2005

Computer Science Department, National University of Ireland,
Maynooth,
County Kildare, Ireland.

- 1 Task Scheduling
 - Why is it hard?
- 2 Scheduling Algorithm
- 3 Experiments
 - Rebalancing
 - Other Schedulers
 - Random distribution of task sizes
- 4 Future Work

Why is it hard?

Challenges

- Task allocation problem NP-hard in general case.
- Dynamic arrival of tasks.
- Varying availability of heterogeneous resources:
 - Network - congestion/failures.
 - Processors - added/removed/failed/non-dedicated.
- Task priorities.
- Trivial properties of tasks - Halting problem.
- Representative set of heterogeneous computing task benchmarks do not exist [1].

Why is it hard?

Practical applications



Folding@home distributed computing



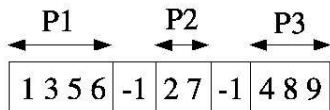
Condor
High Throughput Computing

Genetic Algorithms

- Initialisation: Most-Into-Least list scheduling heuristic
- Fitness function: Euclidean distance from lower bound to current solution
- Roulette wheel selection
- Cycle crossover
- Elitism
- Mutations: random swaps and re-balancing heuristic



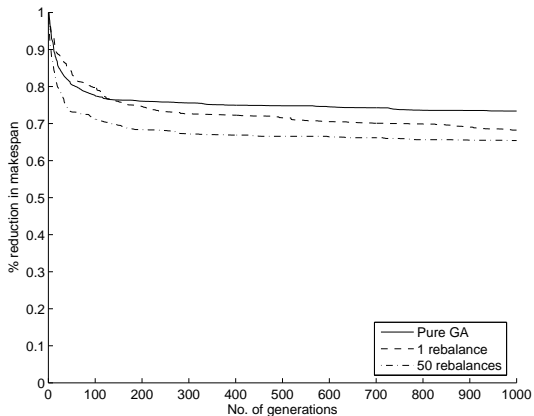
String encoding



Each processor queue is encoded within a single string. A delimiter (-1 in this case) separates each queue. Each other element of the string contains a unique identifier of a task to be processed.

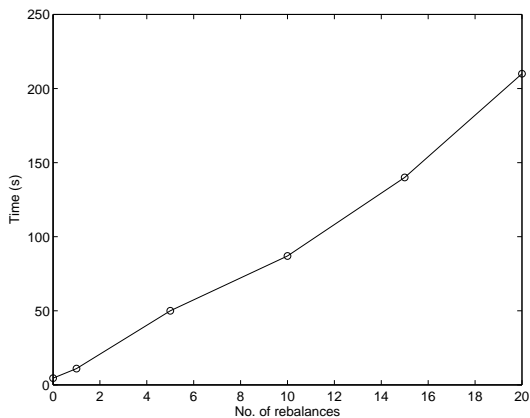
Rebalancing

Rebalancing heuristic



Rebalancing

Time to perform re-balancing

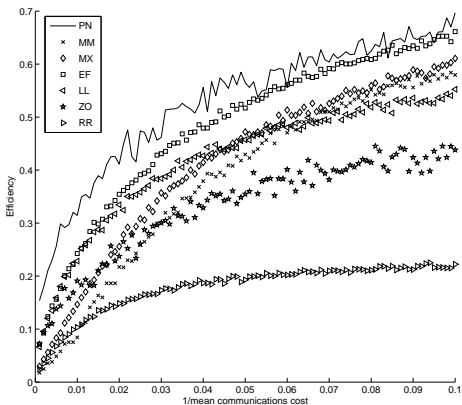


Other Schedulers

- Immediate Mode schedulers
 - Earliest First (EF)
 - Lightest Loaded (LL)
 - Round Robin (RR)
- Batch Schedulers
 - Min-Min (MM)
 - Min-Max (MX)
 - Zomaya *et al.* [2] (ZO)
 - My GA scheduler (PN)

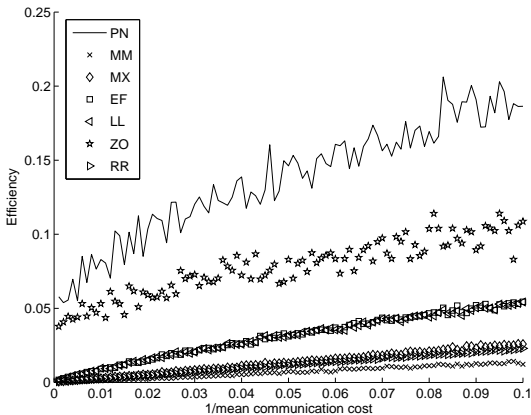
Random distribution of task sizes

Normal distribution of task sizes



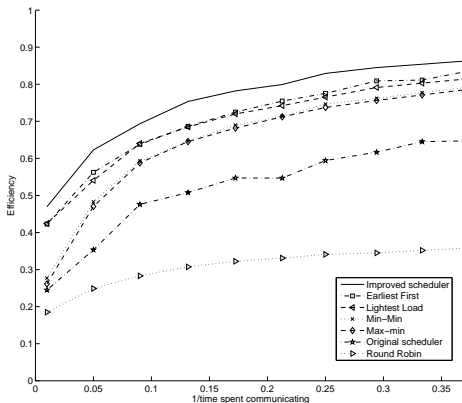
Random distribution of task sizes

Uniform distribution of task sizes



Random distribution of task sizes

Poisson distribution of task sizes



Future Work

- Vary number of tasks and arrival rates.
- Vary processing resources and number of processors.
- Find lower bounds and worst cost measures.
- Implement and test on real-world heterogeneous system.

Java Heterogeneous Distributed System

- Utilizes spare clock cycles of desktop PCs.
- 200+ PCs including 64 processor cluster.
- Multiple operating systems:
 - Windows 98/NT/2000/XP
 - Linux Debian/Fedora/Mandrake
 - FreeBSD
- Spread over 3 different locations.
- Currently processes problems from Bioinformatics, Biomedical Engineering and Cryptography.

Applications

- DPRml: Distributed Phylogeny Reconstruction by maximum likelihood
- DSEARCH: Sensitive database searching
- Distributed Monte Carlo simulation of light propagation through brain tissue.
- ElGamal - Distributed Pollard Rho for testing key strengths.
- MD5 - bruteforce MD5 collision tester.

Conclusion

- Dynamic task scheduler for heterogeneous distributed computing.
- Statistical estimation of system properties.
- Better performance than existing techniques.
- Still a long way to go!

Acknowledgements

- Support is acknowledged from the Irish Research Council for Science, Engineering, and Technology, funded by the National Development Plan.



NUI MAYNOOTH

Ollscoil na hÉireann Má Nuad





Mitchell D. Theys, Tracy D. Braun, Howard Jay Siegal, Anthony A. Maciejewski, and Yu-Kwong Kwok.
Mapping Tasks onto Distributed Heterogeneous Computing Systems Using a Genetic Algorithm Approach, chapter 6, pages 135–178.
John Wiley and Sons, New York, USA, 2001.



A. Y. Zomaya and Yee-Hwei Teh.
Observations on using genetic algorithms for dynamic load-balancing.
IEEE Transactions on Parallel and Distributed Systems, 12(9):899–911, September 2001.