Multi-objective Evolution based Dynamic Job Scheduler in Grid

Debjyoti Paul (11111015)
debpaul@cse.iitk.ac.in
Advisor: Dr. Sanjeev K. Aggarwal

Department of Computer Science & Engineering,
Indian Institute of Technology, Kanpur

June 6, 2013

Outline

Motivation

- An Essential part of a Grid system is an efficient scheduling system i.e. resource sharing problem in dynamic and multi-institutional organizations.
- Maximum utilization of grid resources and minimizing makespan is the most cogitated objective in scheduling literatures more.
- Factors like maintaining Quality of Service, cost effectiveness, energy efficient scheduling should also be entertained while scheduling.
- Fair amount of importance should be given to user satisfaction, time and cost deadline of jobs.

Motivation

- In 2007, Gartner estimated that the Information and Communication Technology industry is liable for 2% of the global CO_2 emission annually, which is equal to that from the aviation industry [Pet07].
- An average data center consumes as much energy as 25,000 households [KFK08].
- Trade off among energy consumption, performance, cost and deadline of jobs.

Definition

Grid Computing (Defn)

Grid is a type of parallel and distributed system that enables sharing, selection and aggregation of geographically distributed resources dynamically at run time depending on their availability, capability, performance, cost, user's quality-of self-service requirement [BV05] more

- On the basis of functionality grid can be classified as
  - Computational Grid more
  - Data Grid more

Grid Environment

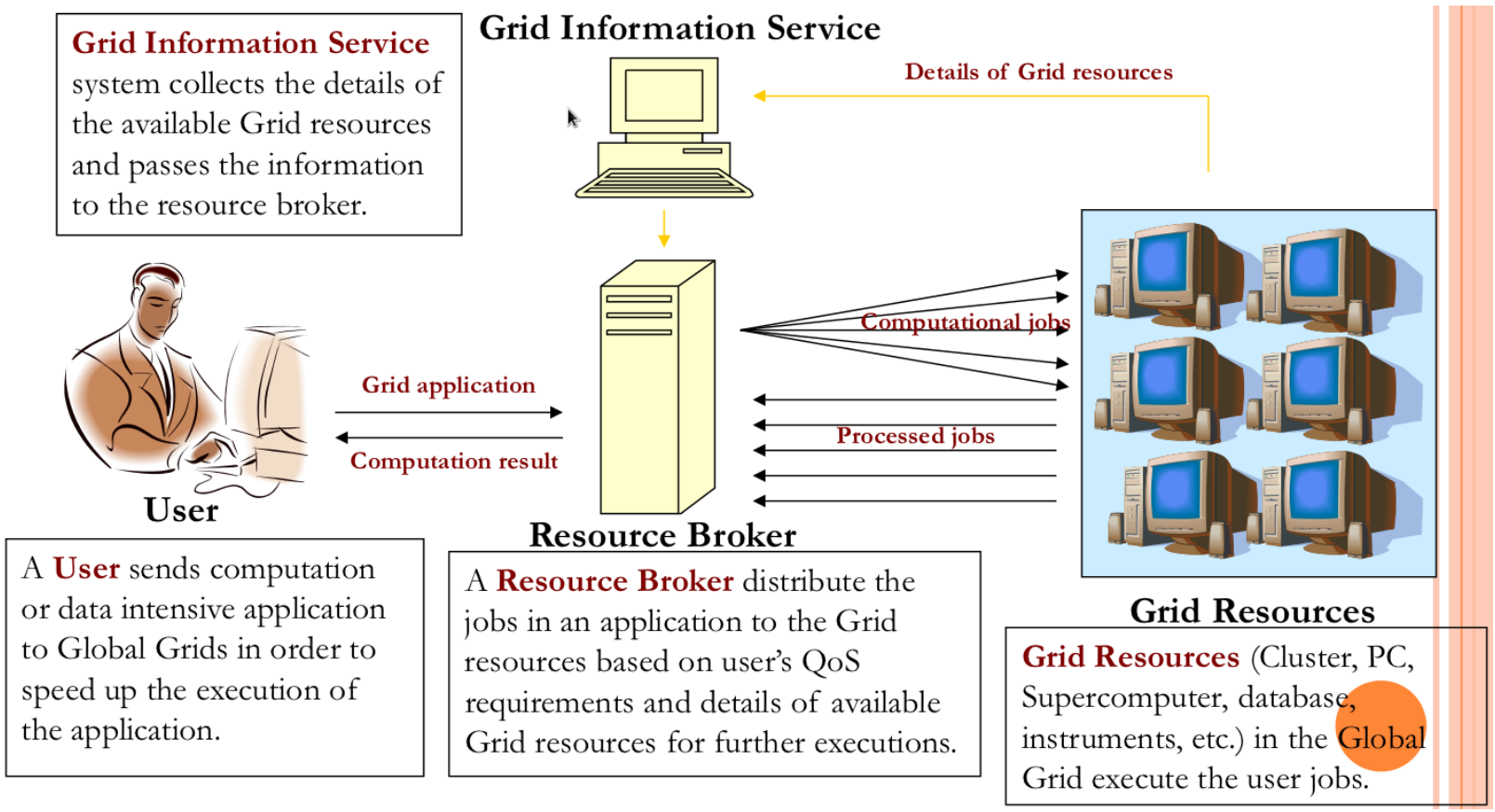


Figure: *A typical view of Grid environment*

Problem Statement

- **Multi-objective dynamic scheduler in Grid**

1. Flexible Job Shop Scheduling

The typical job-shop problem is formulated as a work order that consists of set of n jobs, each of which contains m tasks and need to be scheduled on a set of resources with an objective of minimizing makespan.

  - Predecessor constraint: Each task has predecessors to complete before they can be executed.
  - Resource constraint: Each task requires a certain type of resource to maintain their quality of service.

- Scheduling jobs on resources by considering multiple objectives in scenario is a challenging task. more

Problem Statement

2. Real time dynamic scheduler

Dynamic scheduler considers dynamic environment of grid where resources can change its configuration and availability. In dynamic scheduling re-evaluation is allowed for already taken assignment decisions during job execution [Cht05].
The grid must schedule jobs on resources as early as possible, giving scheduler few minutes to find a suitable strategy.

Problem Statement

3. Gridlet

Grid job is often referred to as Gridlet. Job characteristics e.g. job size varies widely making scheduler task difficult.
So, we need to provide a job grouping strategy for fine grained jobs for faster execution of scheduler.

Contributions

- We present a multi-objective job scheduler based on Genetic Algorithm which considers following objectives
  - Minimize makespan
  - Maximum utilization
  - Energy efficient schedule
  - Minimization of Time penalty
  - Minimization of Cost penalty
- This provide grid administrator a better grip on the trade off among cost, time limit, energy, performance. back

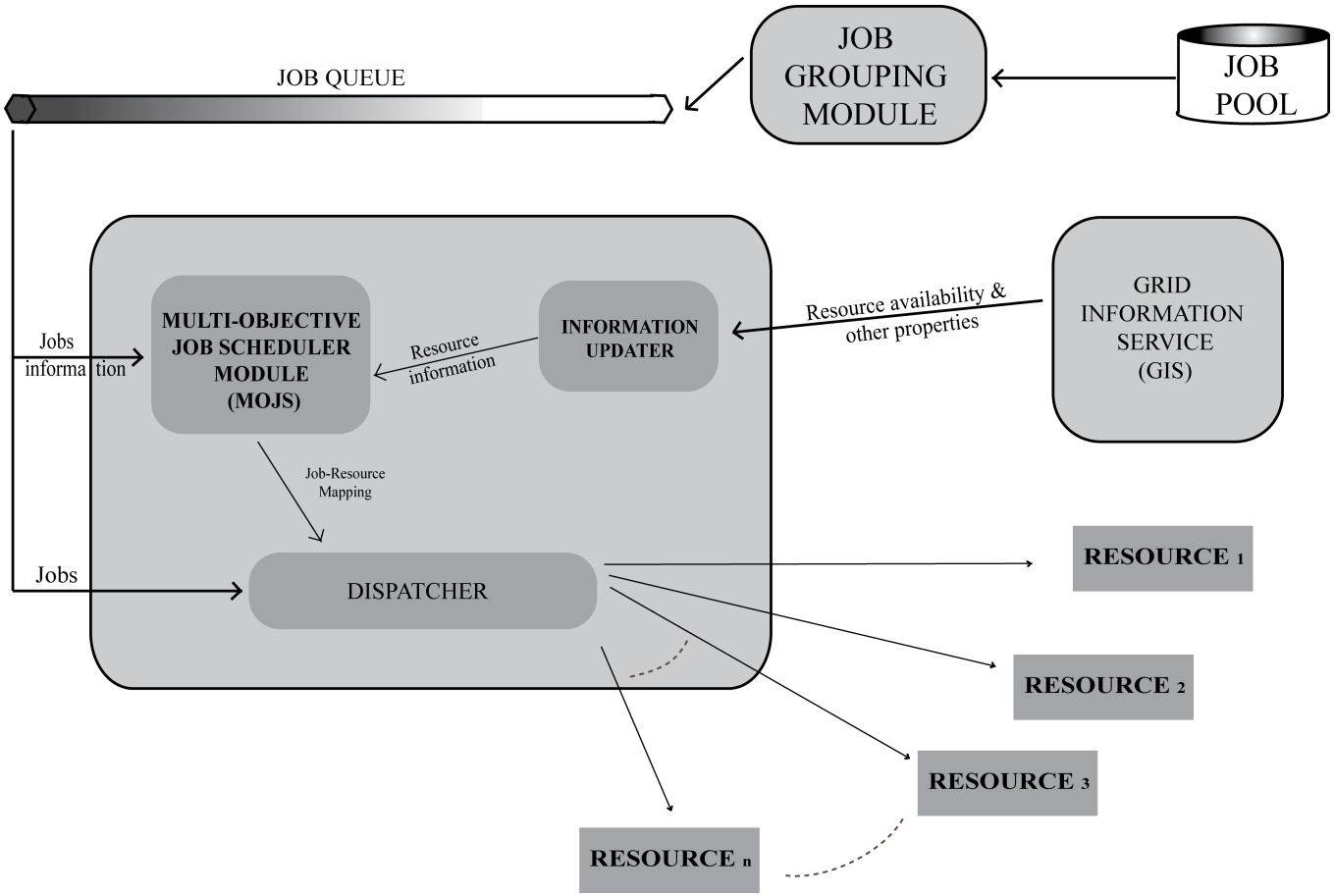
Contributions (contd.)

- Pareto front approach (unlike weighted sum) for optimizing each objective separately in each generation of GA more.
- Non dominated sorting with elitist mechanism to preserve good scheduling strategies during transition of generation more.
- Avant-garde crossover and mutation operators to explore search space minutely and still fast enough to produce near optimal solution in real time.

Contributions (contd.)

- Dynamic scheduler; resources leaving or joining grid environment triggers rescheduling of jobs.
- Job grouping strategy with predecessor and resource constraint.

Scheduling model



Formulation of problem

- Given are a set $M = \{M_1, M_2, M_3, ...M_m\}$ of resources.
- A set $J = \{J_1, J_2, J_3, ...J_j\}$ of application jobs, and a set O of grid jobs.
- n Grid jobs of application job J_i are denoted by $O_{i1}, ..., O_{in}$.
- A set $W = \{W_1, W_2, \ldots, W_m\}$ denotes normalized energy dissipation factor of resources.

Notations

Table: *Notations and their definitions*

Notation	Definition
M_i	Resource with ID i
J_i	Application job with ID i
O_{ij}	jth Grid Job or task of Application job J_i
W_i	Energy dissipation factor of Resource M_i, normalized with the max value from set W
$t(O_{ij}, R_{ij})$	Processing time of O_{ij} mapped to resource R_{ij}
$c(O_{ij}, R_{ij})$	Cost of O_{ij} mapped to resource R_{ij}
$s(O_{ij})$	Start time of job O_{ij}
$e(O_{ij})$	End time of job O_{ij}
d_{ij}	Time limit for completion of O_{ij}
c'_{ij}	cost limit for O_{ij}
$tsum(M_i)$	Running time or Uptime of M_i

Restrictions

A schedule strategy is valid, if the following two **restrictions** are met:

1. All grid jobs are planned and resources are allocated exclusively:
   - $\forall O_{ij} : \exists s(O_{ij}) \in \mathbb{R}, R_{ij} \in \mu_{ij} : \forall M_j \in R_{ij}$:
   - M_j is in $[s(O_{ij}); s(Oij) + t(O_{ij}), R_{ij}]$ exclusively allocated by O_{ij}
2. Precedence relations are adhered to:
   - $\forall i, j \neq k : p(O_{ij}, O_{ik}) \Rightarrow s(O_{ik}) \geq s(O_{ij}) + t(O_{ij}, R_{ij})$

Soft constraints

Exceeding the time limit and budget cost will affect QoS of grid jobs. A penalty factor is imposed when jobs violates following **constraints**.

1. All grid jobs O_{ij} have time limit d_{ij} which must be adhered to:
   - $\forall O_{ij} : d_{ij} \geq s(O_{ij}) + t(O_{ij}, R_{ij})$:
2. All grid jobs O_{ij} have a cost limit c'_{ij} which must be adhered to:
   - $\forall O_{ij} : c'_{ij} \geq c(O_{ij}, R_{ij})$

Objective functions or fitness functions

This work focuses on achieving near-optimal scheduling strategy on following objective functions

1. Minimizing makespan, $e(O_{ij})$ is the end time of grid job O_{ij}
   - $f_1 = makespan = max\{e(I(M_1)), e(I(M_2)), \ldots, e(I(M_m))\}$

Makespan

It is the time at which all the resources becomes free defn.

2. Maximizing utilization of resources i.e. minimizing f_2
   - $f_2 = non - utilization = \frac{1}{m}\sum_{j=1}^{m}\{e(I(M_j)) - tsum(M_j)\}$

Objective functions or fitness functions (contd.)

3. Minimizing time limit penalty (minimizing number of jobs completing after due date)

$$f_3 = \frac{1}{j * n} \sum_{\forall i,j} \varphi_1(e(O_{ij}) - d_{ij})$$

where $\varphi_1(x)$ is a non-negative continuous exponential non-decreasing function, if $x > 0$ else 0.

Time penalty

The grid jobs failed to complete within time limit contribute to the penalty function f_3. Minimizing time limit penalty is our aim.

Objective functions or fitness functions (contd.)

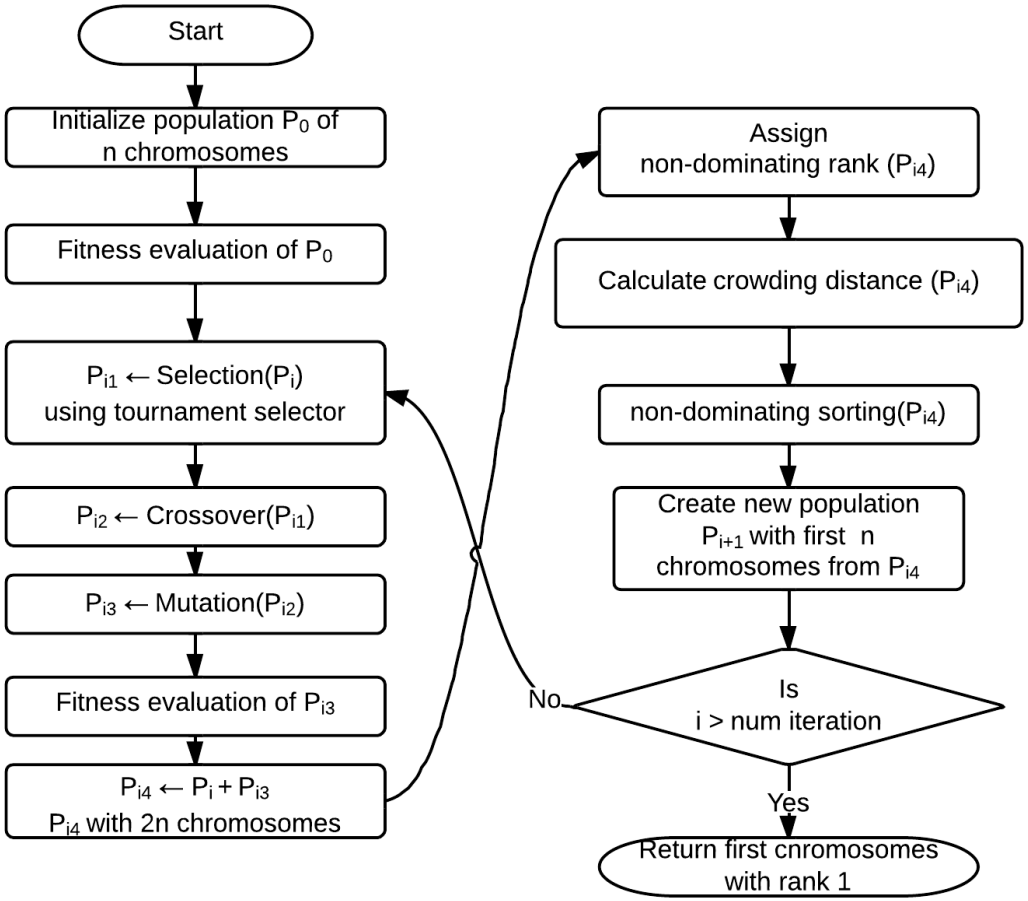
4. Minimizing cost penalty

$$f_4 = \frac{1}{j * n} \sum_{\forall i,j} \varphi_2\{c(O_{ij}, R_{ij}) - c'_{ij}\}$$

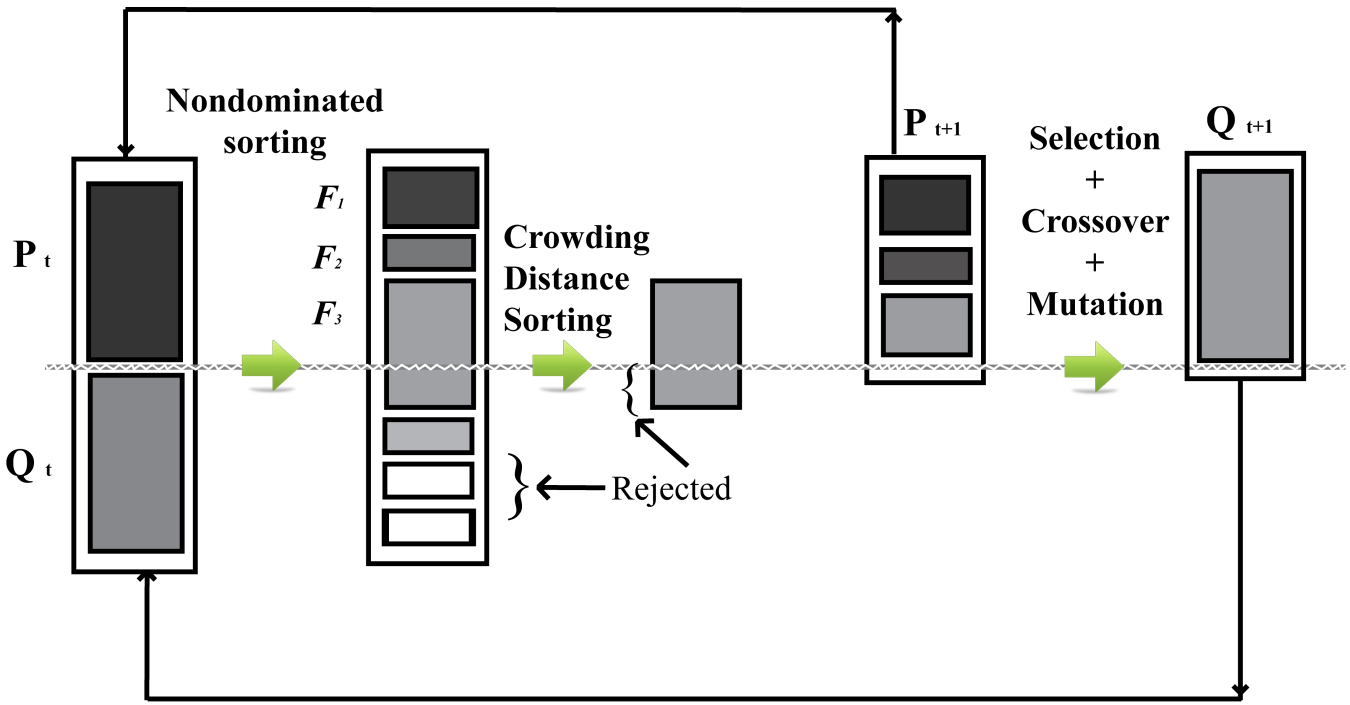
where $\varphi_2(x)$ is a non-negative continuous linear non-decreasing function, if $x > 0$ else 0.

5. Minimizing Overall Energy consumption
   - $f_5 = \sum_{i=1}^{m} tsum(M_i) * W_i$
   - f_5 is the energy consumption of all the resources.

MOJS flowchart



Schematic overview



Chromosome model

- A scheduling strategy $\leftrightarrow$ A chromosome.
  - Resource id corresponding to each job
  - Start time of every job
  - End time of every job
  - Predecessor job ID of each job
  - Five objective function values
  - Rank of chromosome more.
  - Crowding distance more.
- Start time for execution of jobs is calculated according to heuristic rules
  1. Schedule grid job as early as its precedent job is completed.
  2. Schedule grid jobs according to shortest due date.

Crossover operators: k-point crossover

	j1	j2	j3	j4	j5	j6	j7	j8	j9	j10	j11	j12	j13	j14	j15	j16	j17	j18	j19	j20
parent1	2	1	3	2	3	4	1	3	2	2	1	3	4	2	1	4	3	3	1	3
parent2	1	4	4	2	1	3	2	2	1	2	3	1	4	4	2	3	1	1	4	1

k-point crossover

	j1	j2	j3	j4	j5	j6	j7	j8	j9	j10	j11	j12	j13	j14	j15	j16	j17	j18	j19	j20
child1	2	4	4	2	1	3	1	3	1	2	3	1	4	2	1	4	3	1	4	1
child2	1	1	3	2	3	4	2	2	2	2	1	3	4	4	2	3	1	3	1	3

Figure: *k-point crossover*

Crossover operators: Mask crossover

	j1	j2	j3	j4	j5	j6	j7	j8	j9	j10	j11	j12	j13	j14	j15	j16	j17	j18	j19	j20
parent1	2	1	3	2	3	4	1	3	2	2	1	3	4	2	1	4	3	3	1	3
parent2	1	4	4	2	1	3	2	2	1	2	3	1	4	4	2	3	1	1	4	1

mask	1	0	0	0	1	1	1	1	1	0	1	0	1	0	0	1	1	0	0	1

uniform crossover

	j1	j2	j3	j4	j5	j6	j7	j8	j9	j10	j11	j12	j13	j14	j15	j16	j17	j18	j19	j20
child1	1	1	3	2	1	3	2	2	1	2	3	3	4	2	1	3	1	3	1	1
child2	2	4	4	2	3	4	1	3	2	2	1	1	4	4	2	4	3	1	4	3

Figure: *Mask crossover*

Crossover operators: Fitness based crossover

- $g_1[i], g_2[i]$ are energy efficiency parameter of $chromosome_{parent_1}[i]$ and $chromosome_{parent_2}[i]$ respectively.

$$\forall i, chromosome_{new_1}[i] = \begin{cases} chromosome_{parent_1}[i] & \text{with probability } p = \frac{g_1[i]}{g_1[i]+g_2[i]} \\ chromosome_{parent_2}[i] & \text{with probability } 1-p \end{cases} \quad (1)$$

$$\forall i, chromosome_{new_2}[i] = \begin{cases} chromosome_{parent_1}[i] & \text{with probability } p = \frac{h_1[i]}{h_1[i]+h_2[i]} \\ chromosome_{parent_2}[i] & \text{with probability } 1-p \end{cases} \quad (2)$$

- $h_1[i], h_2[i]$ are processing capability parameter of $chromosome_{parent_1}[i]$ and $chromosome_{parent_2}[i]$ respectively.

Crossover operators: Fitness based crossover

	j1	j2	j3	j4	j5	j6	j7	j8	j9	j10	j11	j12	j13	j14	j15	j16	j17	j18	j19	j20
parent1	2	1	3	2	3	4	1	3	2	2	1	3	4	2	1	4	3	3	1	3
parent2	1	4	4	2	1	3	2	2	1	2	3	1	4	4	2	3	1	1	4	1

normalized	r1	r2	r3	r4
energy parameter	0.9	0.85	1.0	0.7
processing parameter	0.85	0.6	0.7	1.0

probability of inherence of gene for child1 from

parent1	0.485	0.562	0.588	0.5	0.526	0.411	0.514	0.540	0.485	0.5	0.473	0.526	0.5	0.548	0.514	0.411	0.526	0.526	0.562	0.526

fitness over energy parameter

probability of inherence of gene for child2 from

parent2	0.413	0.459	0.411	0.5	0.451	0.588	0.586	0.538	0.413	0.5	0.548	0.451	0.5	0.375	0.586	0.588	0.451	0.451	0.459	0.451

fitness over processing capability parameter

Figure: *Fitness based crossover*

Mutation operators: Move

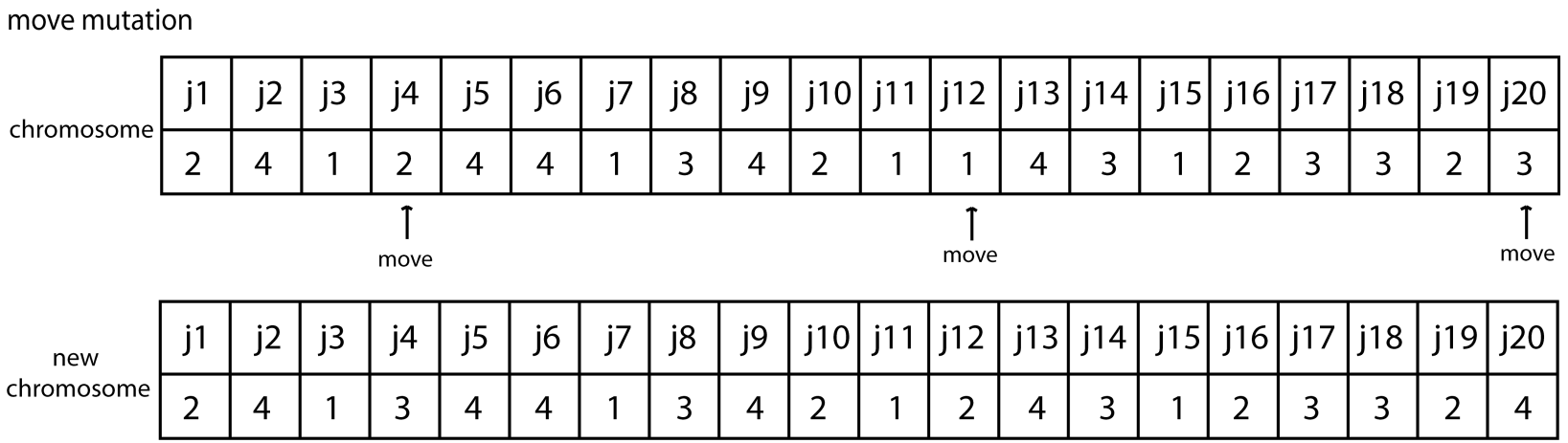


Figure: *Mutation- move*

Mutation operators: Swap

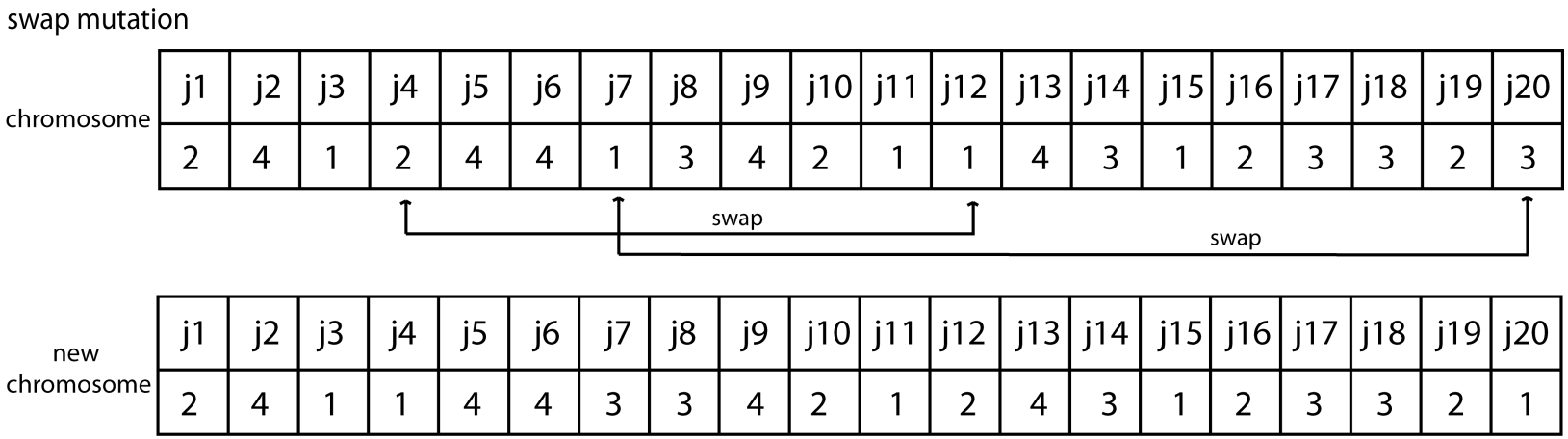


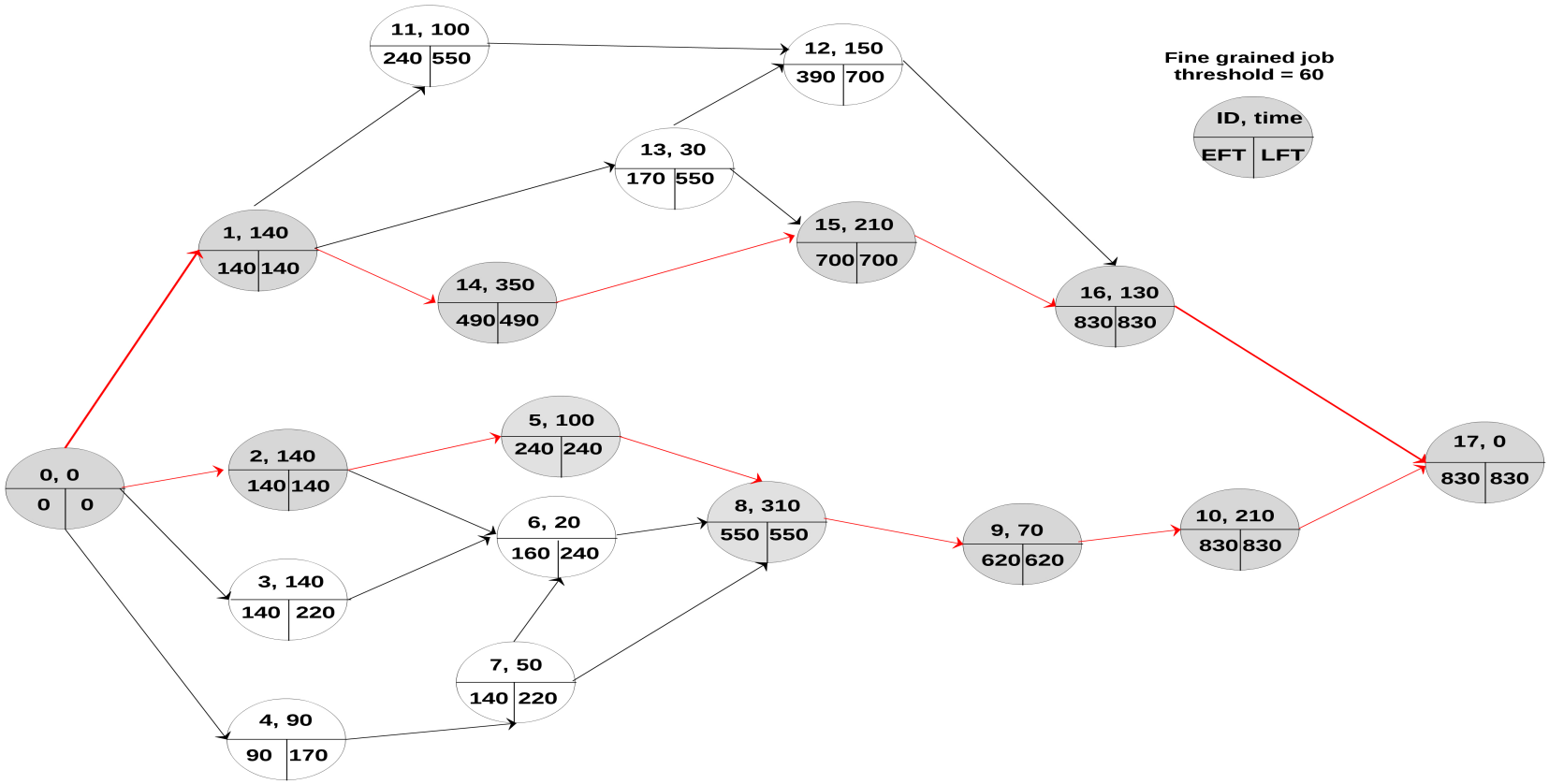
Figure: *Mutation- swap*

Mutation operators: Rebalancing

- This operator takes into account number of jobs assigned to each resource. It chooses most overloaded resource and randomly pick a job assigned to it. Then the job is moved to a resource which is less overloaded.

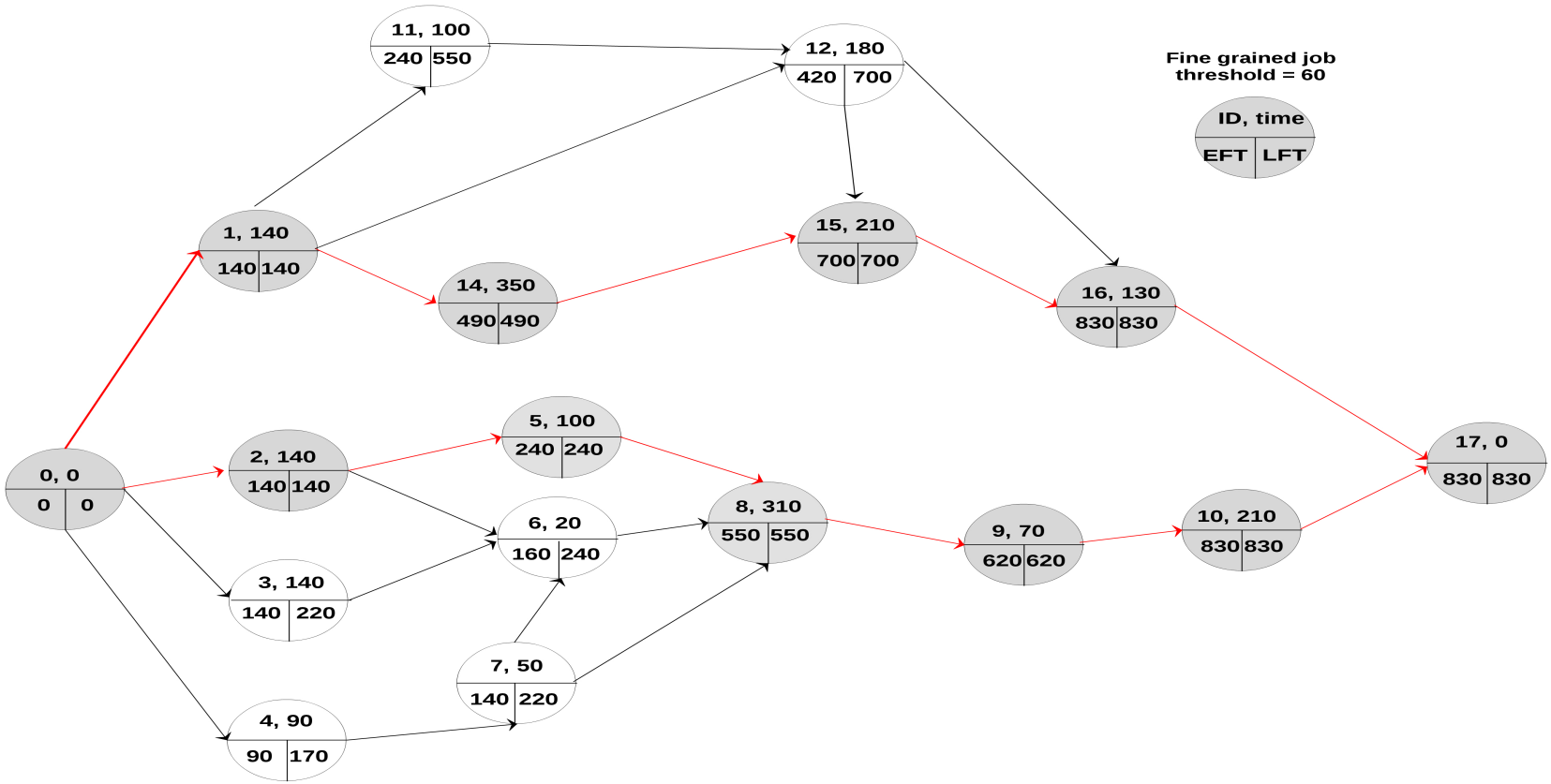
Job grouping

algorithm link



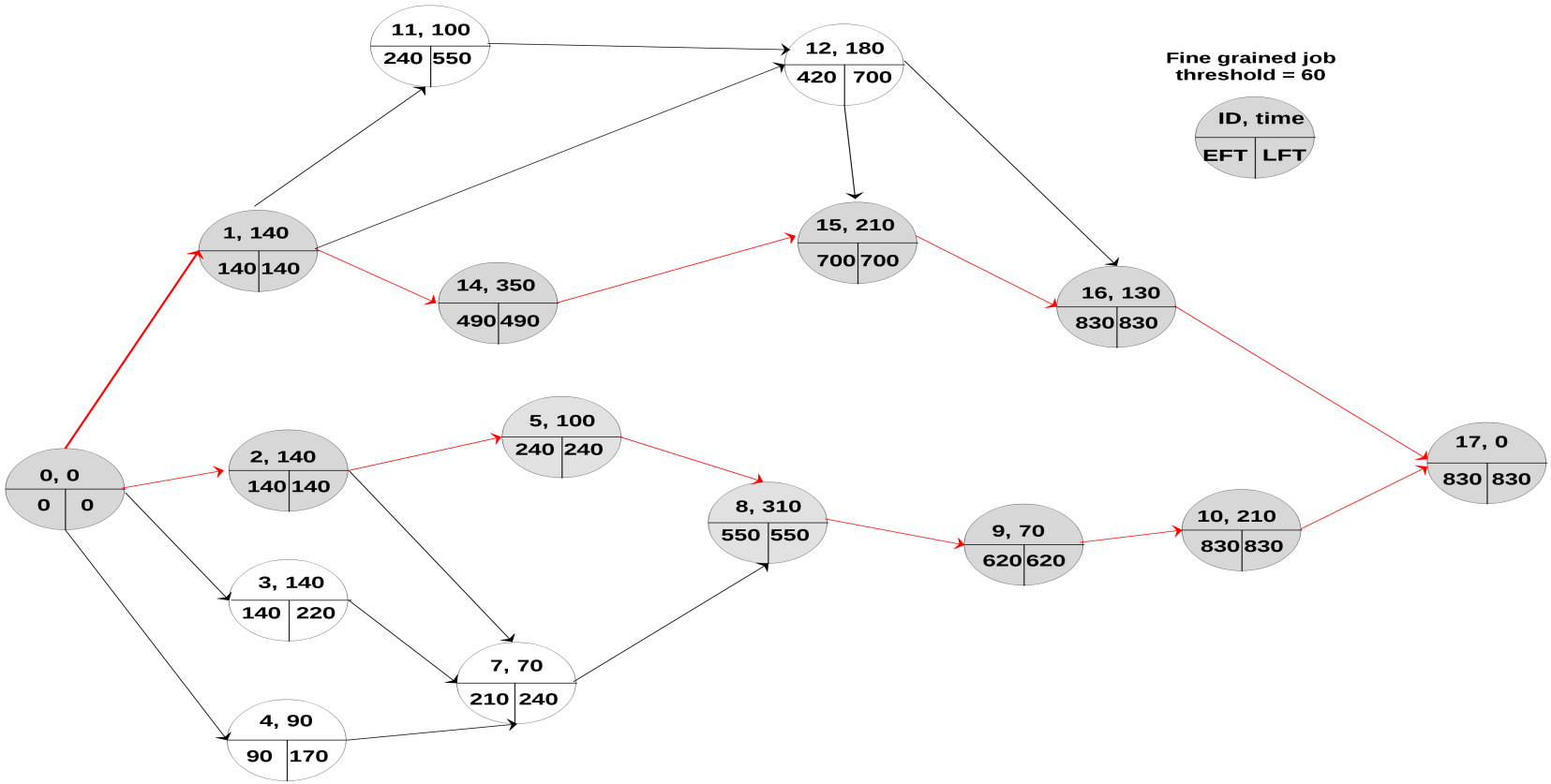
Job grouping

algorithm link



Job grouping

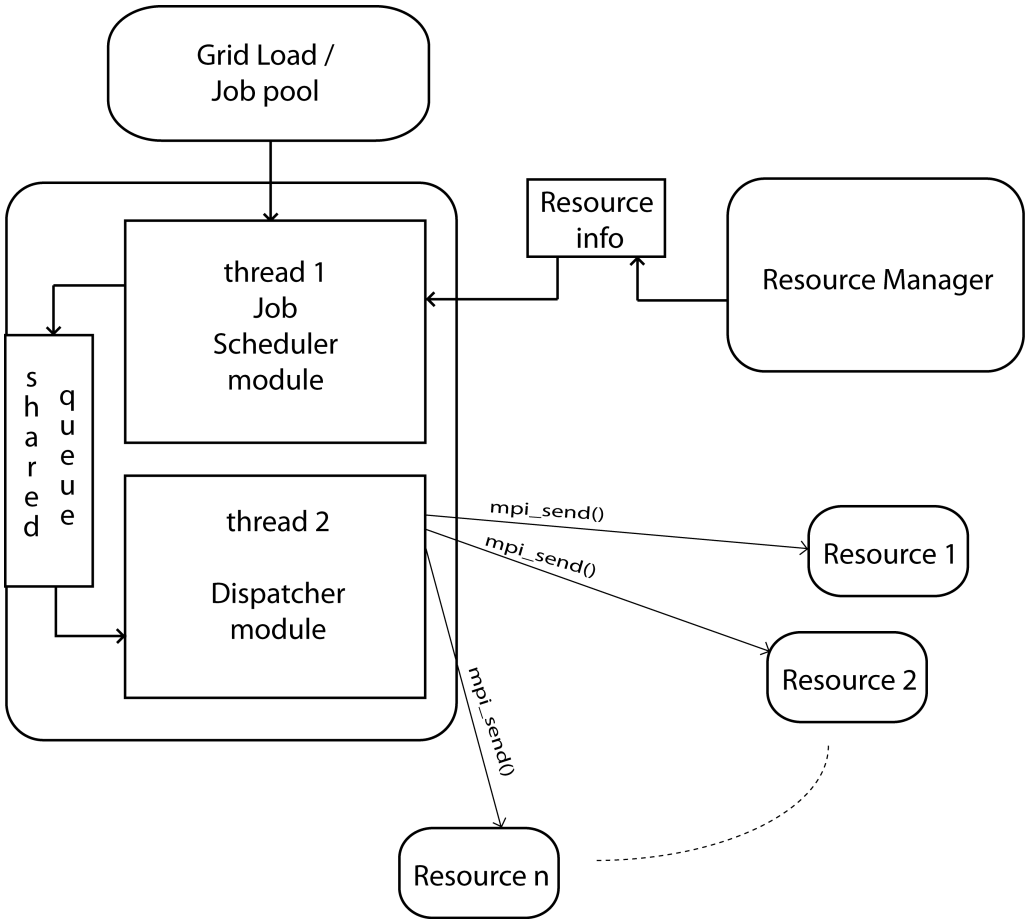
algorithm link



Dynamic scheduler

- Change in the resource pool can trigger running of MOJS module which can either reschedule the jobs whose resources have left the grid or can request processing of new jobs on addition of one or more resources or both of them.
- Jobs whose predecessors is present in the set of rescheduled jobs are also rescheduled.
- Weighted sum approach is applied on multiple objectives to finalize a chromosome as scheduling strategy from first pareto front.
- Some jobs are queued on their respective resource while others are again fed to MOJS module. The number of jobs queued depend on the fitness of the chromosome and time available for the scheduler to re-run and find a better schedule.
- Progress is ensured by setting a minimum number of jobs to be queued on a single run of module.

Implemented system model



Data sets

- Standard grid workload from Grid Workload Archive have been used in this experiments [GWA13].
  - SHARCNET & DAS-2 are the two traces that have been processed.
  - Traces shows that execution time of jobs ranges from 1 to 20000 time units in DAS-2, and 1 to 100000 time units in SHARCNET.

Workload_id	Trace	Precedence constraint	Resource constraint
1	DAS-2	✗	✗
2	SHARCNET	✗	✗
3	DAS-2	✗	✓
4	SHARCNET	✗	✓
5	DAS-2	✓	✗
6	SHARCNET	✓	✗
7	DAS-2	✓	✓
8	SHARCNET	✓	✓

1. Experiment on independent jobs

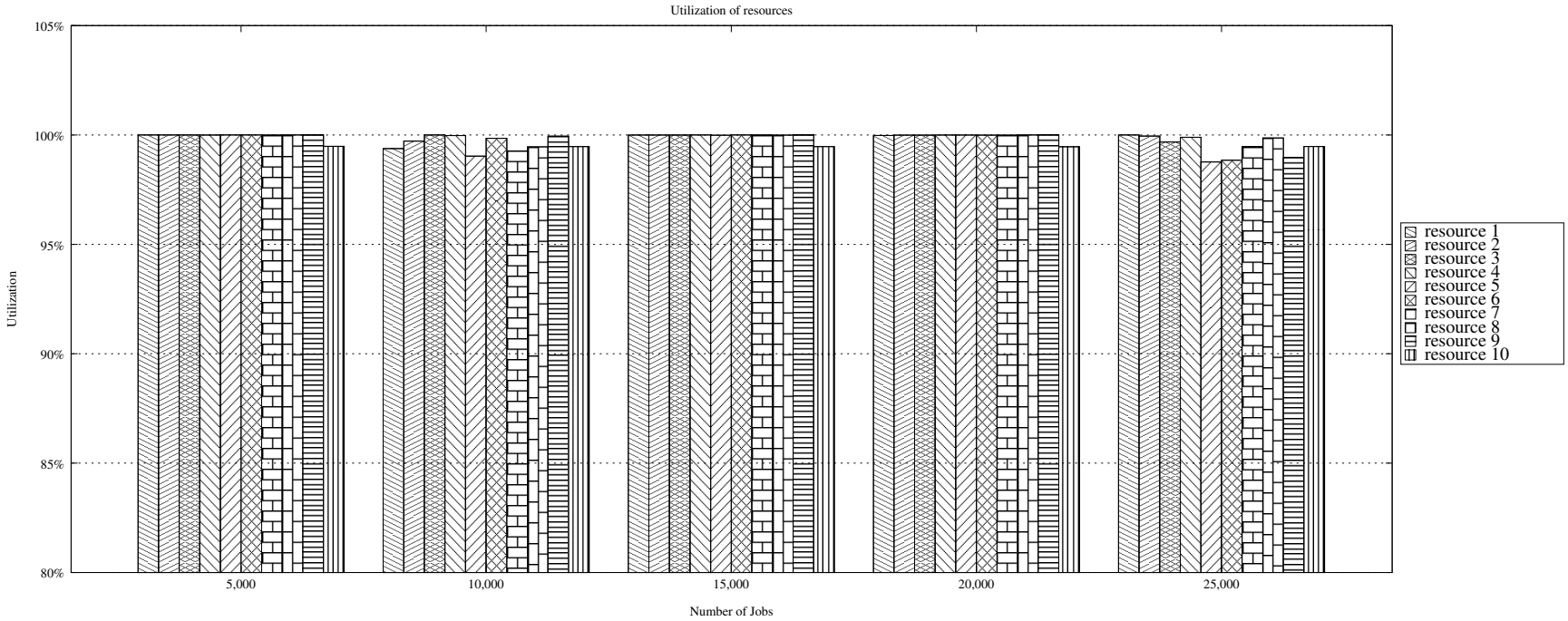


Figure: *Evaluation of makespan and utilization on 10 resources on workload 2*

1. Experiment on independent jobs

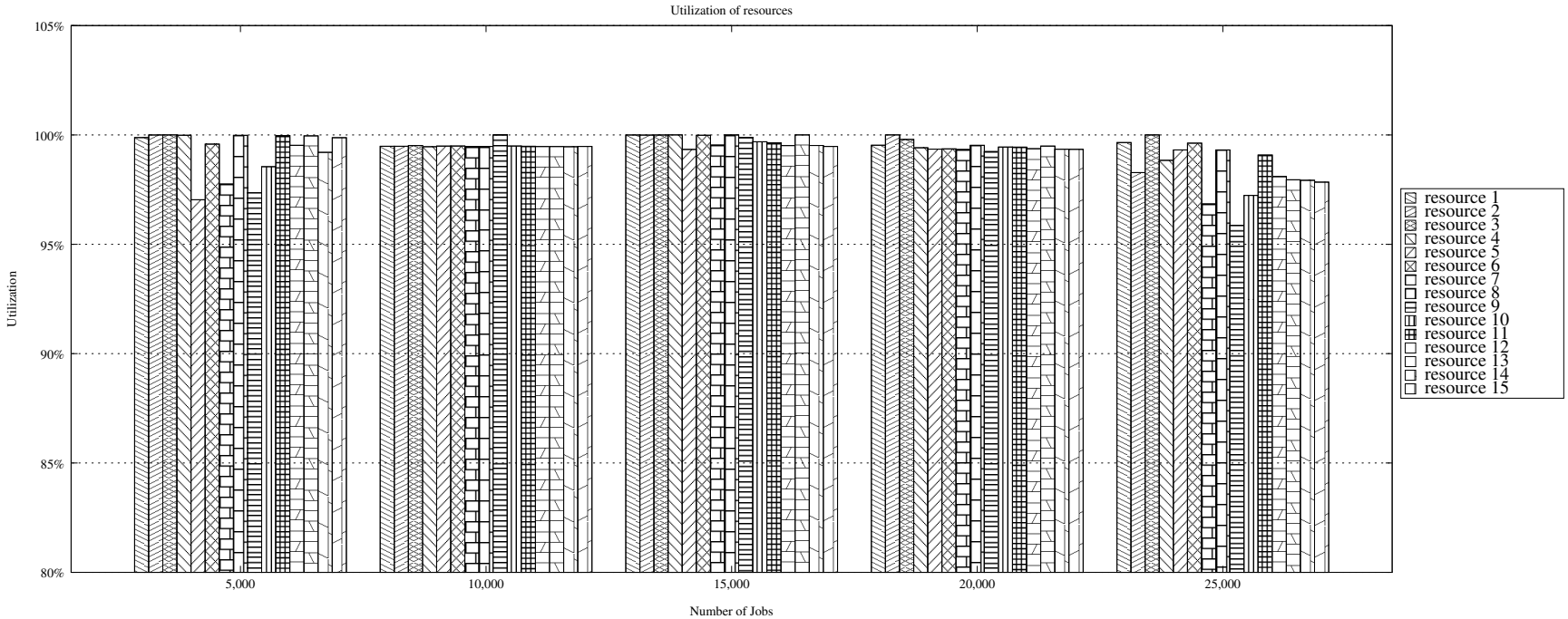


Figure: *Evaluation of makespan and utilization on 15 resources on workload 1*

1. Experiment on independent jobs

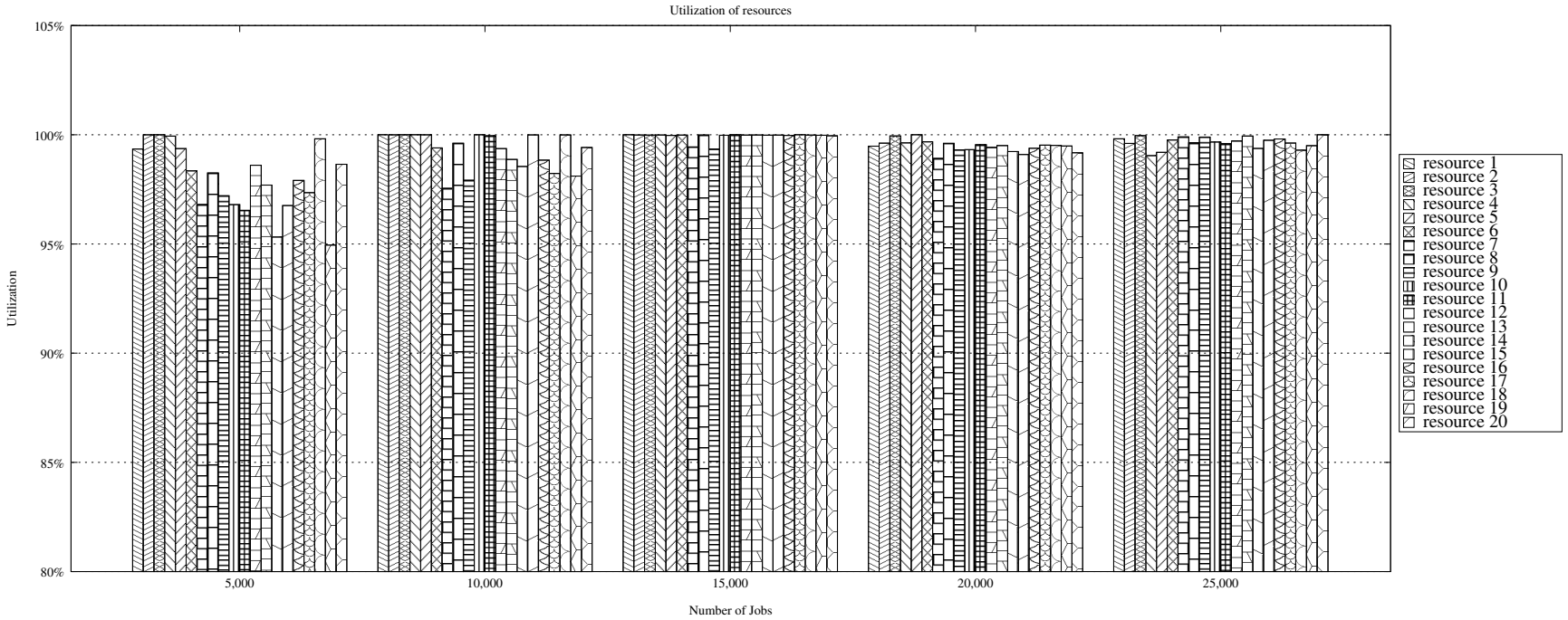


Figure: *Evaluation of makespan and utilization on 20 resources on workload 1*

1. Experiment on independent jobs

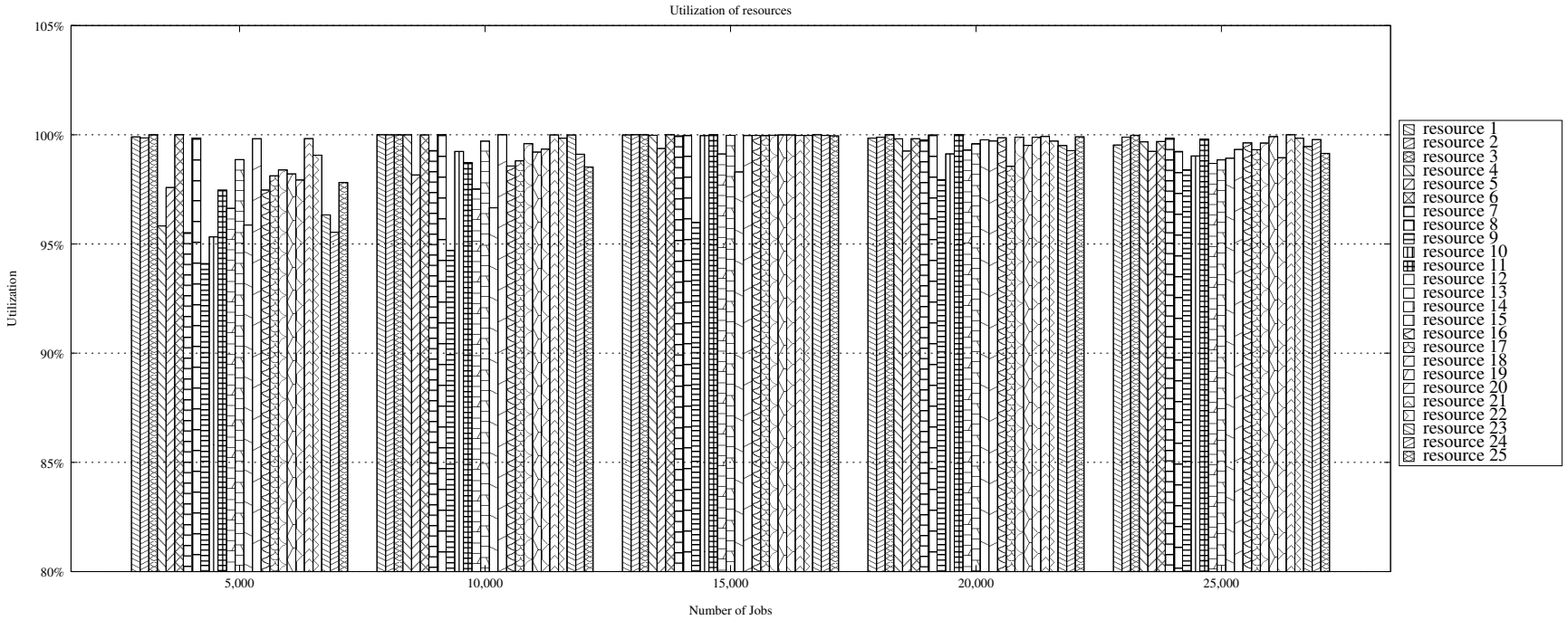


Figure: *Evaluation of makespan and utilization on 25 resources on workload 2*

2. Experiment on trade off between energy consumption and performance

Table: *Resource configuration for experiment on workload 1 and 2*

Machine	Frequency	Watts	MIPS/core	Resource_id
Pentium 4 Extreme Edition	3.2 GHz	92.1	9,726	1,2
Intel Core 2 X6800 (Dual core)	2.93 GHz	75	13,539	3,4
Intel Core 2 QX6700 (Quad core)	2.66 GHz	95	12,290	5,6
Intel Core i7 920 (Quad core)	2.667 GHz	130	20,575	7,8
Intel Core i7 3960X (Hex core)	3.3 GHz	130	29,621	9,10
Core i7-2600	3.4 GHz	95	32,075	11, 12

2. Experiment on trade off between energy consumption and performance

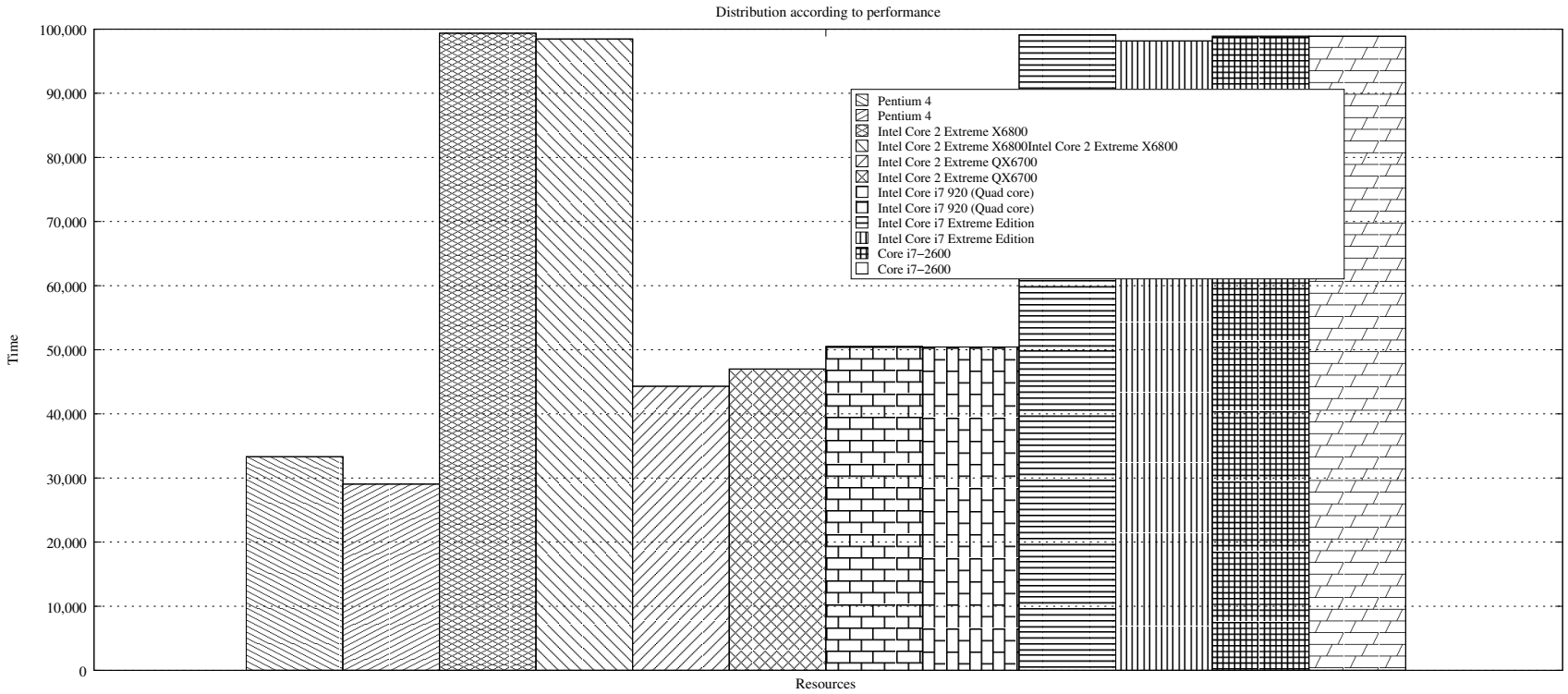


Figure: *Workload 2 (SHARCNET)*

2. Experiment on trade off between energy consumption and performance

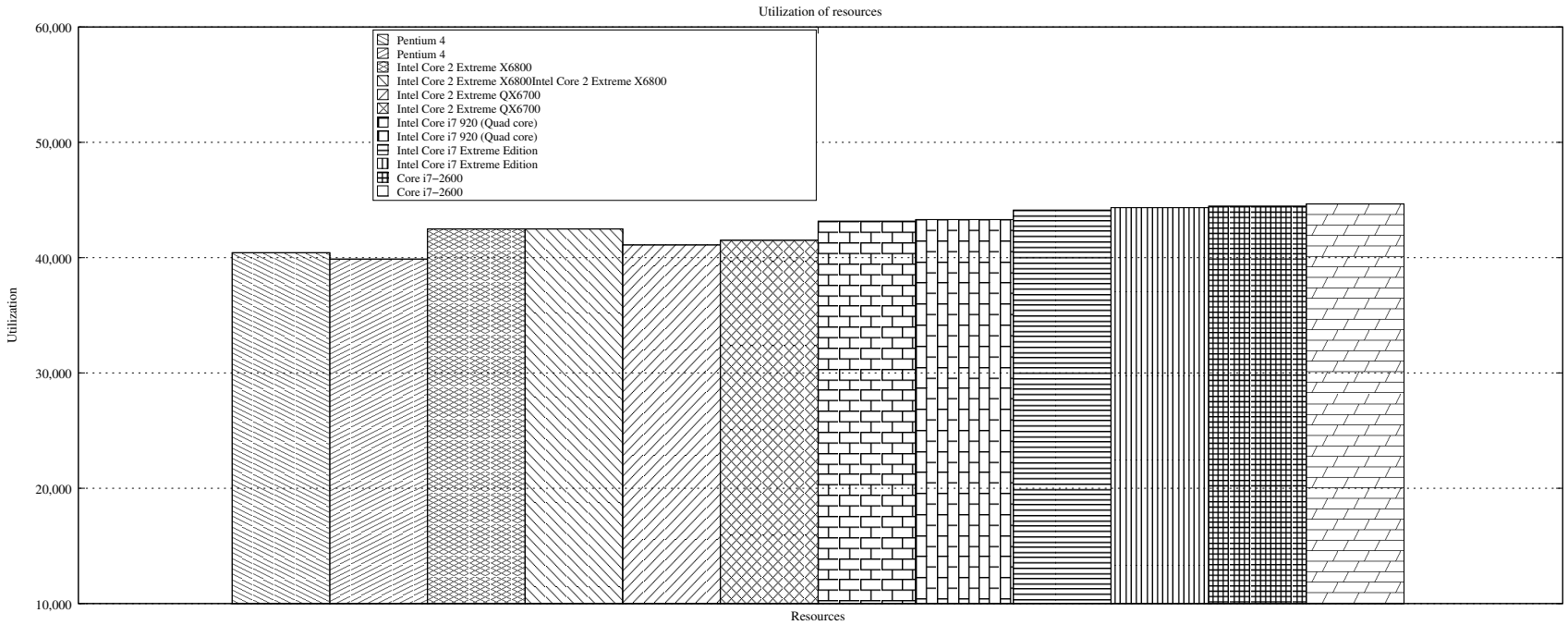


Figure: *Workload 1 (DAS-2)*

3. Experiment: Introducing job type constraint

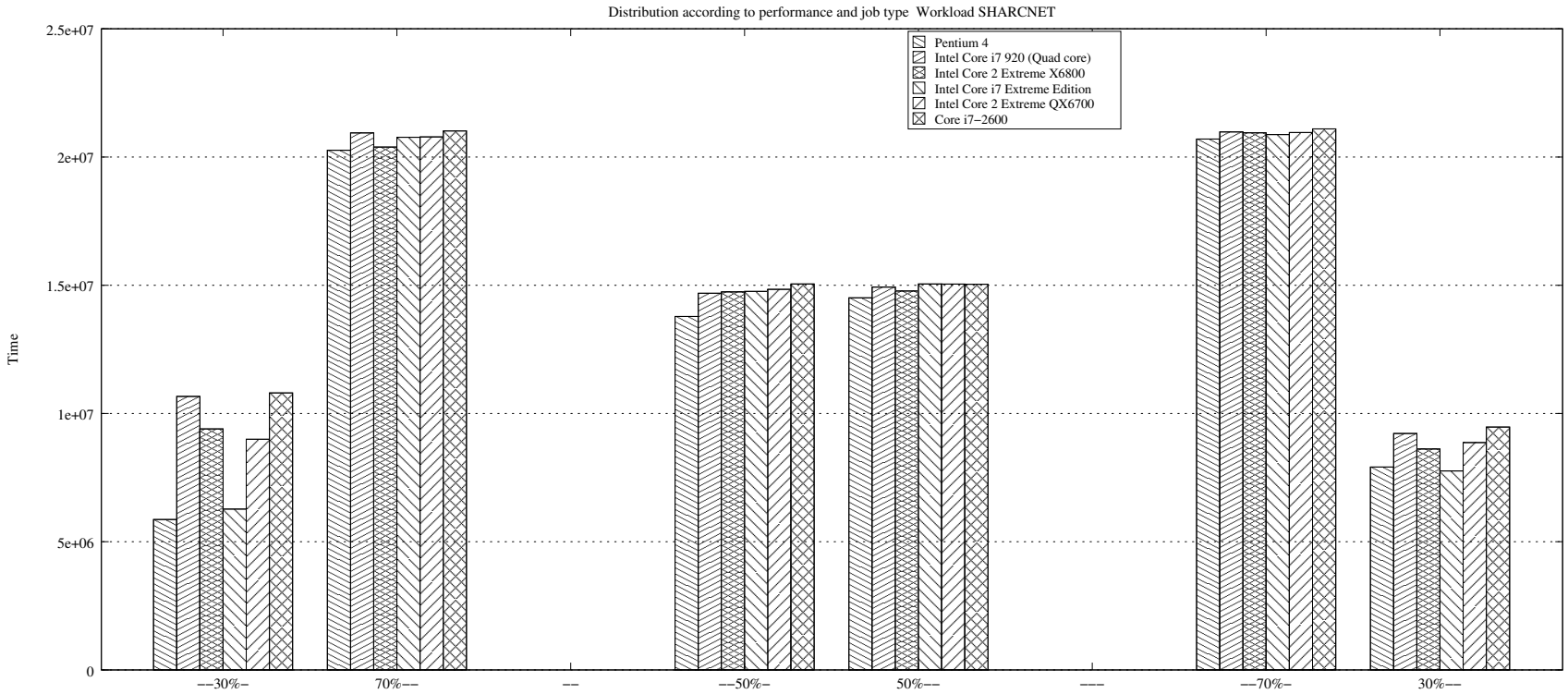


Figure: *Performance under Job type constraint, SHARCNET Workload*

3. Experiment: Introducing job type constraint

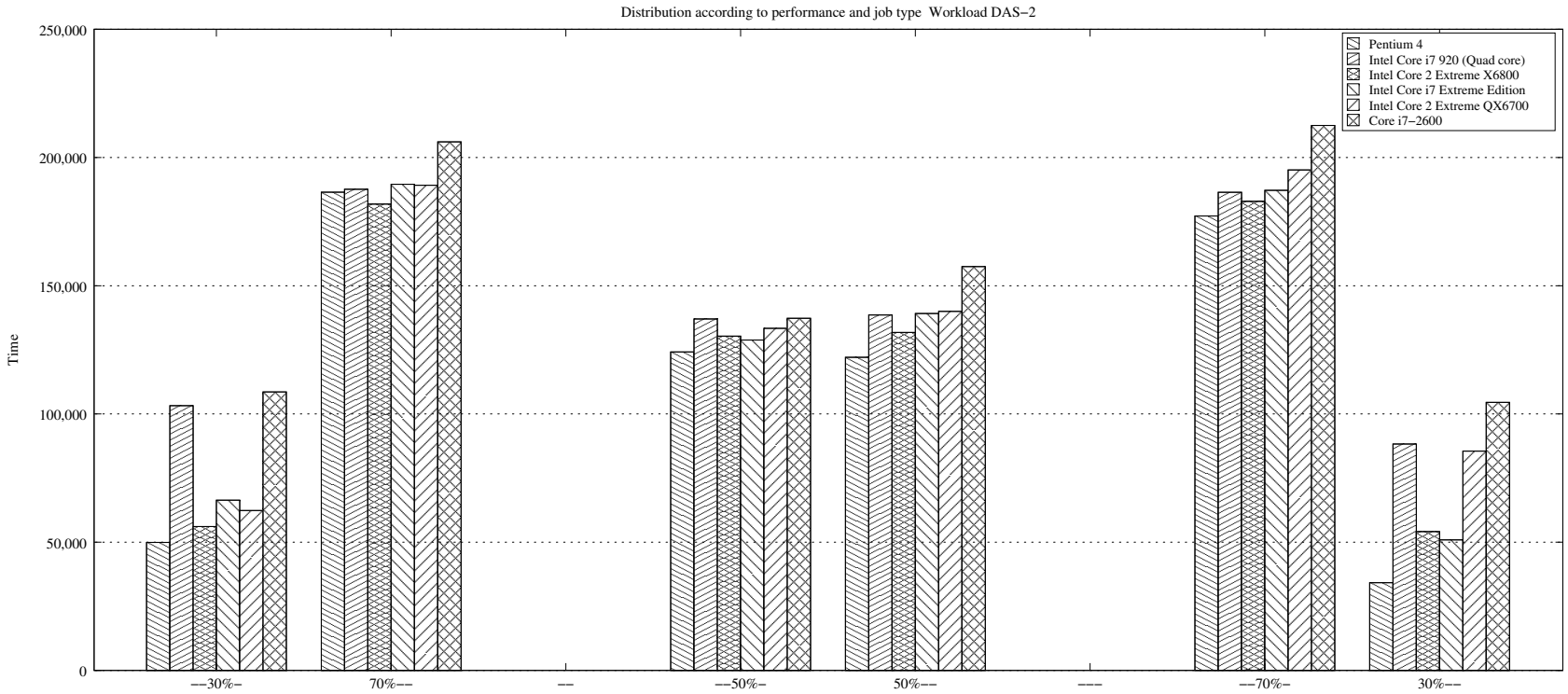


Figure: *Performance under Job type constraint, DAS-2 Workload*

4. Experiment: Introducing pricing model and precedence constraint

Table: *Resource Utilization after introduction of Job constraint and pricing model Workload 6 (SHARCNET)*

ID	Resource	Pricing model	Execution time	Makespan %age	Actual utilized time	Utilization %age
1	Pentium 4	0.05	12048592.25	98.01	11813044.19	98.04
2	Pentium 4	0.05	12163752.56	98.95	11300235.12	92.90
3	Pentium 4	0.05	12164478.67	98.95	11686679.20	96.07
4	Intel Core i7 920 (Quad core)	0.1	12046834.85	97.99	11272379.33	93.57
5	Intel Core i7 920 (Quad core)	0.1	12047842.36	98.00	11749485.96	97.52
6	Intel Core 2 Extreme X6800	0.09	12048727.00	98.01	11734951.47	97.39
7	Intel Core 2 Extreme X6800	0.09	12160673.18	98.92	11776008.86	96.83
8	Intel Core i7 Extreme Edition	0.15	12162928.27	98.94	11422999.16	93.91
9	Intel Core i7 Extreme Edition	0.15	12160661.70	98.92	11777797.37	96.85
10	Intel Core 2 Extreme QX6700	0.165	12160968.73	98.92	11864565.48	97.56
11	Intel Core 2 Extreme QX6700	0.165	12156329.05	98.89	11976902.34	98.52
12	Intel Core 2 Extreme QX6700	0.165	12160817.18	98.92	11727719.11	96.43
13	Intel Core 2 Extreme QX6700	0.165	12160965.55	98.92	11931317.99	98.11
14	Core i7-2600	0.18	12293364.46	100.00	11959913.00	97.28
15	Core i7-2600	0.18	12161210.37	98.92	12059303.00	99.16

4. Experiment: Introducing pricing model and precedence constraint

Table: *Resource Utilization after introduction of Job constraint and pricing model Workload 5 (DAS-2)*

ID	Resource	Pricing model	Execution time	Makespan %age	Actual utilized time	Utilization %age
1	Pentium 4	0.05	307118.82	99.20	280545.30	91.34
2	Pentium 4	0.05	301366.18	97.34	270360.92	89.71
3	Pentium 4	0.05	300697.12	97.12	270477.22	89.95
4	Intel Core i7 920 (Quad core)	0.1	307367.74	99.28	273357.18	88.93
5	Intel Core i7 920 (Quad core)	0.1	307456.72	99.31	283748.63	92.28
6	Intel Core 2 Extreme X6800	0.09	305703.06	98.74	269904.98	88.28
7	Intel Core 2 Extreme X6800	0.09	305227.13	98.59	276089.83	90.45
8	Intel Core i7 Extreme Edition	0.15	305428.16	98.65	288772.30	94.54
9	Intel Core i7 Extreme Edition	0.15	303222.35	97.94	278433.59	91.82
10	Intel Core 2 Extreme QX6700	0.165	309606.43	100.00	291890.84	94.27
11	Intel Core 2 Extreme QX6700	0.165	309296.74	99.90	291225.75	94.15
12	Intel Core 2 Extreme QX6700	0.165	305602.29	98.71	288181.42	94.29
13	Intel Core 2 Extreme QX6700	0.165	303673.86	98.08	297436.40	97.94
14	Core i7-2600	0.18	308464.86	99.63	303151.00	98.27
15	Core i7-2600	0.18	308339.70	99.59	303203.00	98.33

5. Experiment with all constraints

Table: *Resource Utilization under all constraints on Workload 7 (DAS-2)*

ID	Resource	Pricing model	Execution time	Makespan %age	Actual utilized time	Utilization %age
1	Pentium 4	0.05	388054.55346	95.31	337986.508894	87.10
3	Intel Core i7 920 (Quad core)	0.1	390646.860537	95.95	362241.059321	92.72
5	Intel Core 2 Extreme X6800	0.09	391917.860219	96.26	357671.610452	91.26
7	Intel Core i7 Extreme Edition	0.15	389061.135613	95.56	367173.559685	94.37
9	Intel Core 2 Extreme QX6700	0.165	389834.143125	95.75	368452.037427	94.51
11	Core i7-2600	0.18	394967.616183	97.01	365376	92.50
2	Pentium 4	0.05	407130.811479	100.00	362875.39219	89.12
4	Intel Core i7 920 (Quad core)	0.1	406698.612175	99.89	343454.864029	84.44
6	Intel Core 2 Extreme X6800	0.09	403146.303747	99.02	348012.012859	86.32
8	Intel Core i7 Extreme Edition	0.15	402622.406754	98.89	378981.878416	94.12
10	Intel Core 2 Extreme QX6700	0.165	405572.293961	99.62	385571.358202	95.06
12	Core i7-2600	0.18	394268.616183	96.84	374894	95.08

5. Experiment with all constraints

Table: *Resource Utilization under all constraints on Workload 8 (SHARCNET)*

ID	Resource	Pricing model	Execution time	Makespan %age	Actual utilized time	Utilization %age
1	Pentium 4	0.05	15856966.328786	99.65	14643863.906402	92.34
3	Intel Core i7 920 (Quad core)	0.1	15777189.495383	99.14	14224208.443954	90.15
5	Intel Core 2 Extreme X6800	0.09	15777954.142035	99.15	14347179.713218	90.93
7	Intel Core i7 Extreme Edition	0.15	15853934.301256	99.63	14962167.332284	94.37
9	Intel Core 2 Extreme QX6700	0.165	15855065.258749	99.63	14971170.083674	94.42
11	Core i7-2600	0.18	15807884.835386	99.34	14039647	88.81
2	Pentium 4	0.05	15855945.977623	99.64	15192585.514258	95.81
4	Intel Core i7 920 (Quad core)	0.1	15856889.388767	99.65	15492772.72683	97.70
6	Intel Core 2 Extreme X6800	0.09	15856577.788845	99.64	15509484.322644	97.81
8	Intel Core i7 Extreme Edition	0.15	15808788.880171	99.34	15123906.518069	95.66
10	Intel Core 2 Extreme QX6700	0.165	15913317.542563	100.00	15439774.23157	97.02
12	Core i7-2600	0.18	15884947.971385	99.82	15685884	98.74

5. Experiment with all constraints

Table: *Resource Utilization under all constraints on Workload (SHARCNET) with 24 resources*

ID	Resource	Pricing model	Execution time	Makespan %age	Actual utilized time	Utilization Percentage
1	Pentium 4	0.05	8243698.72	99.49	7556234.16	91.66
2	Pentium 4	0.05	7275262.72	87.81	6253412.08	85.95
3	Intel Core i7 920 (Quad core)	0.1	7682671.48	92.72	6326204.11	82.34
4	Intel Core i7 920 (Quad core)	0.1	8233315.82	99.37	7351339.16	89.28
5	Intel Core 2 Extreme X6800	0.09	8117606.74	97.97	7269743.03	89.55
6	Intel Core 2 Extreme X6800	0.09	8118002.88	97.98	7072373.18	87.11
7	Intel Core i7 Extreme Edition	0.15	8117468.76	97.97	7183725.78	88.49
8	Intel Core i7 Extreme Edition	0.15	8244584.64	99.50	7426916.89	90.08
9	Intel Core 2 Extreme QX6700	0.165	8117904.66	97.98	7220275.87	88.94
10	Intel Core 2 Extreme QX6700	0.165	8119335.22	97.99	7202661.19	88.70
11	Core i7-2600	0.18	8229761.81	99.33	7763631	94.33
12	Core i7-2600	0.18	8244728.91	99.51	7615311	92.36
13	Pentium 4	0.05	8119374.38	97.99	7906705.97	97.38
14	Pentium 4	0.05	8231574.39	99.35	7705017.38	93.60
15	Intel Core i7 920 (Quad core)	0.1	8229150.06	99.32	7628743.04	92.70
16	Intel Core i7 920 (Quad core)	0.1	8117134.83	97.97	7339766.91	90.42
17	Intel Core 2 Extreme X6800	0.09	8244402.12	99.50	7126312.76	86.43
18	Intel Core 2 Extreme X6800	0.09	8120187.28	98.00	7600934.79	93.60
19	Intel Core i7 Extreme Edition	0.15	8285650.01	100.00	8151742.16	98.38
20	Intel Core i7 Extreme Edition	0.15	8230477.53	99.33	7746832.68	94.12
21	Intel Core 2 Extreme QX6700	0.165	8232608.55	99.36	7776374.29	94.45
22	Intel Core 2 Extreme QX6700	0.165	8230191.41	99.33	7517359.04	91.33
23	Core i7-2600	0.18	8244461.91	99.50	7950420	96.43
24	Core i7-2600	0.18	8244329.91	99.50	7745014	93.94

Conclusions

- This work provides a flexible scheduler keeping multiple objectives into consideration.
- The scheduler module yields best scheduling strategies on various parameters in a pareto front. This is upto grid administrator and dynamic grid environment to choose a scheduling strategy of its choice.
- The scheduler is scalable with resources and can process an infinite queue of jobs.
- The scheduler responded well with the change of constraints and behaviour of grid and job model.
- All resources have adhered to the makespan, energy efficiency, QoS and utilization rate is also high inspite of constraints imposed.

References I

Rajkumar Buyya and Srikumar Venugopal.

Article: A gentle introduction to grid computing and technologies.
Journal: Database, 2:R3, 2005.

Ann Chervenak, Ian Foster, Carl Kesselman, Charles Salisbury, and Steven Tuecke.

The data grid: Towards an architecture for the distributed management and analysis of large scientific datasets.
Journal of network and computer applications, 23(3):187–200, 2000.

M. Chtepen.

Dynamic scheduling in grids system.
Sixth Firw PhD Symposium, Faculty of Engineering, Ghent University, page 110, 2005.

GWA.

The grid workloads archive, 2013.
Online; accessed 10-April-2013, http://gwa.ewi.tudelft.nl/pmwiki/.

James M Kaplan, William Forrest, and Noah Kindler.

Revolutionizing data center energy efficiency.
McKinsey & Company, Tech. Rep, 2008.

Cris Pettey.

Gartner estimates ict industry accounts for 2 percent of global co2 emissions, 2007.

THANK YOU!!

Makespan

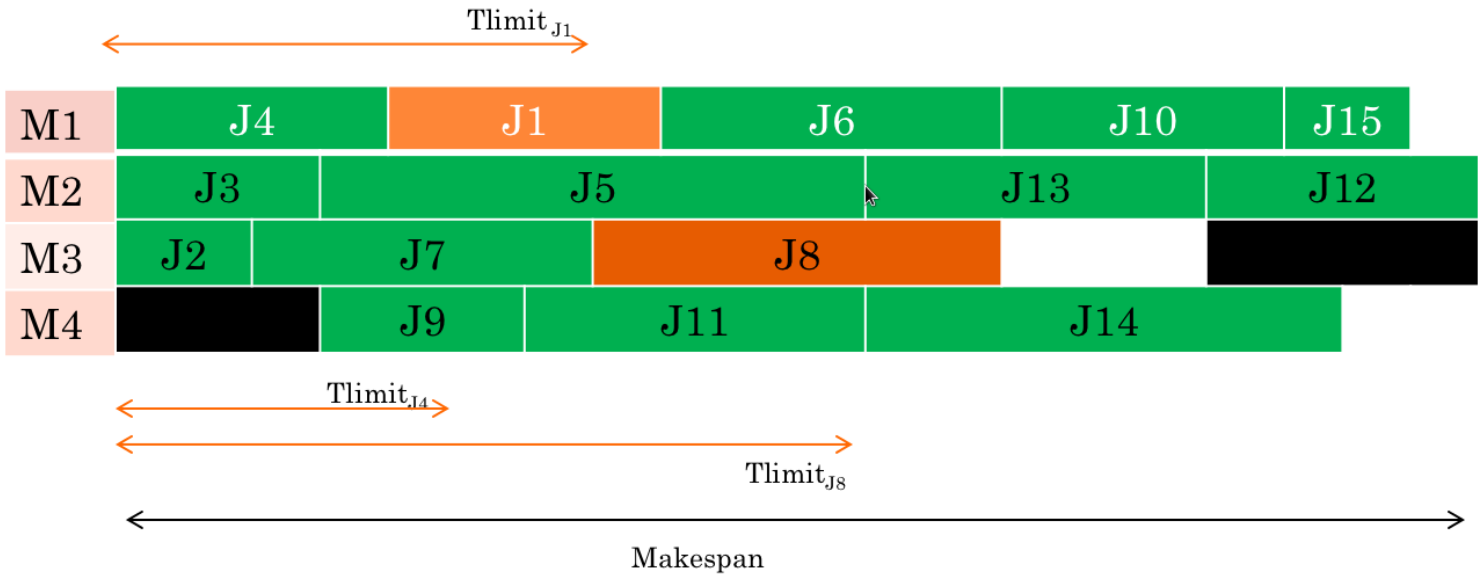


Figure: *Makespan example*

- The completion time for the entire set of Jobs on all resources

back objectives.

Grid computing (Defn)

- "A Grid is a collection of distributed computing resources available over a local or wide area network that appears to an end user or application as one large virtual computing system." - IBM
- "Conceptually, a grid is quite simple. It is a collection of computing resources that perform tasks. In its simplest form, a grid appears to users as a large system that provides a single point of access to powerful distributed resources." - Sun
- "Grid computing is computing as a utility - you do not care where data resides, or what computer processes your requests. Analogous to the way utilities work, clients request information or computation and have it delivered - as much as they want, and whenever they want." - Oracle

back

Types of Grid

- **Computational Grid:** A computational grid is a collection of distributed computing resources, within or across locations that are combined to act as a unified computing resource.
- **Data Grid:** Data grid primarily deals with providing services and infrastructure for distributed data-intensive applications that need to access, transfer and modify massive datasets stored in distributed storage resources [CFK+00]. back

Non-dominating points

- A chromosome a is said to be dominated by chromosome b iff $\forall i \in \{1, 2, \ldots, k\} : f_i(a) \leq f_i(b)$ and $\exists i \in \{1, 2, \ldots, k\} : f_i(a) < f_i(b)$.
- A chromosome a is said to be Non-dominated if there does not exist any chromosome $b \in \mathbb{V}$ search space that dominates a. A set of such non-dominated chromosome in objective space is called pareto optimal front. back

Non-dominating points

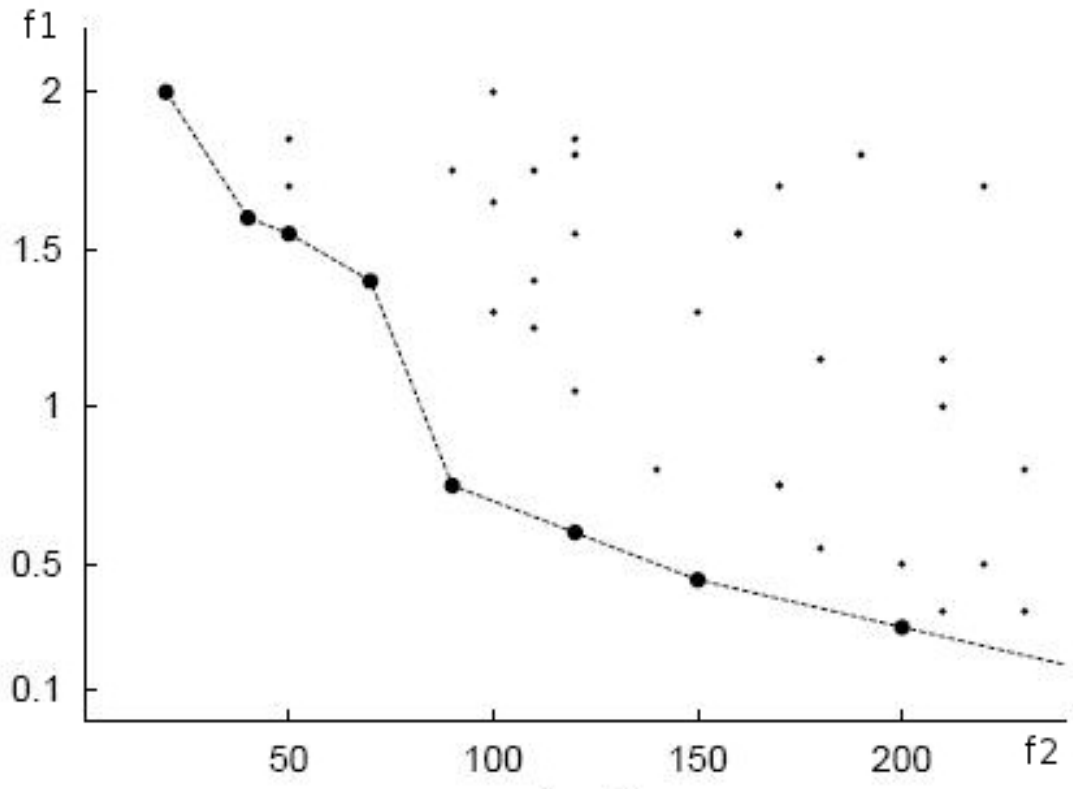


Figure: *An example of non-dominating points in 2-d* back rank

Pareto front

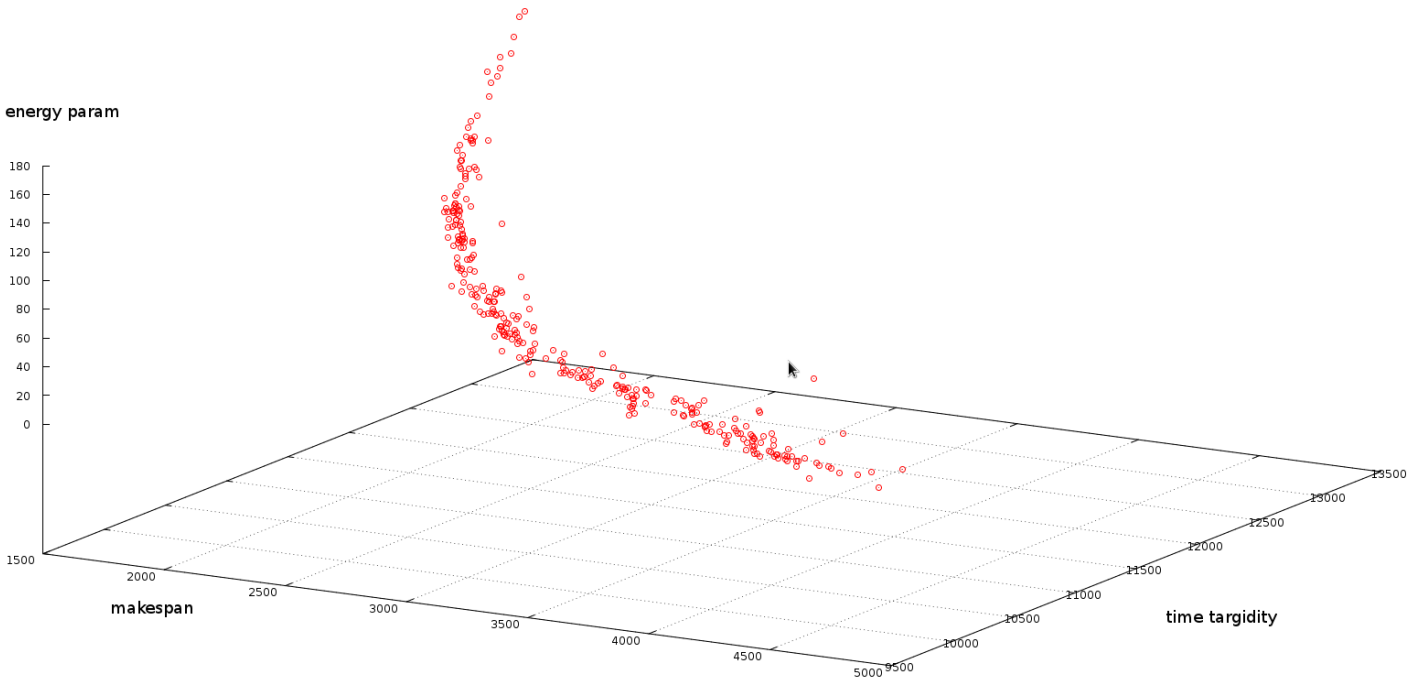


Figure: *An example of pareto front from our experiment* back

Pareto front

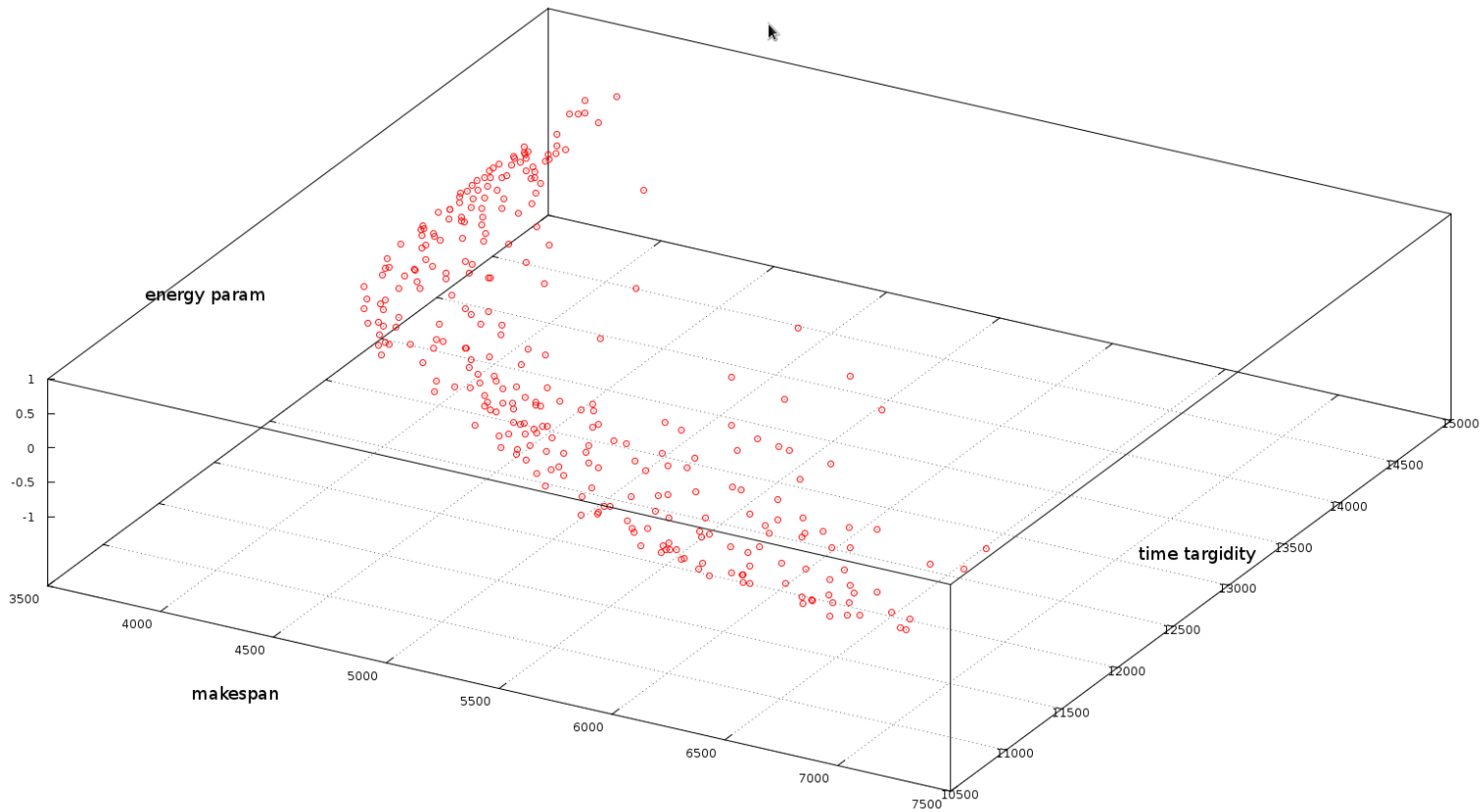


Figure: *An example of pareto front from our experiment* back

Pareto front

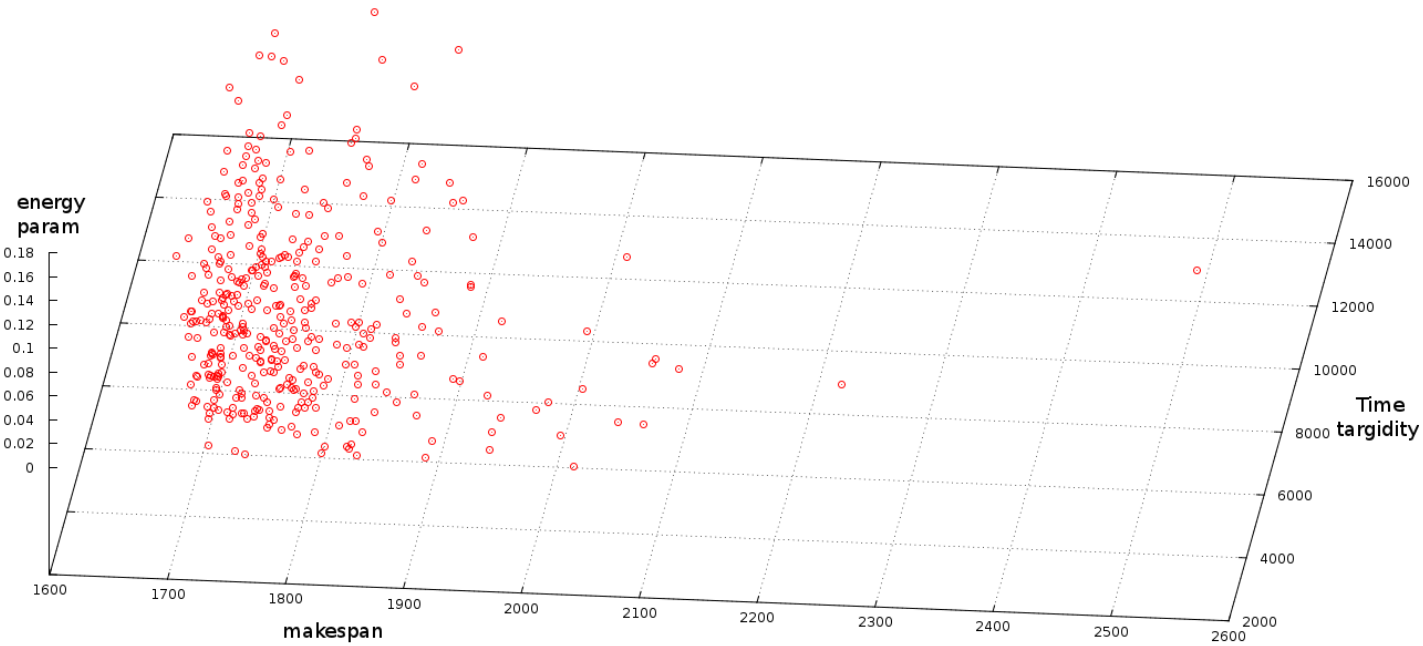


Figure: *An example of pareto front from our experiment* back

Pareto front

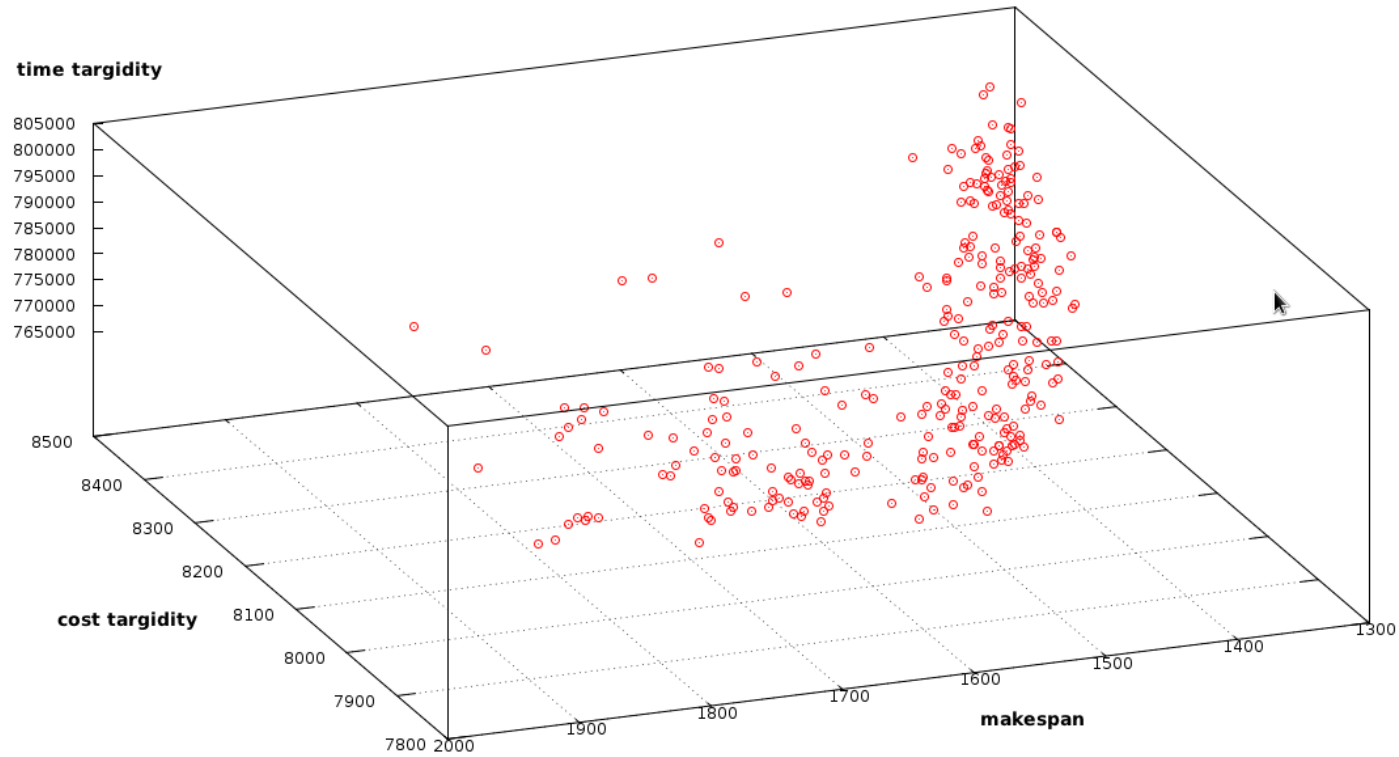


Figure: *An example of pareto front from our experiment* back

Crowding distance

Crowding distance

Crowding distance ($dist_x$) of a particular chromosome x in population measures the density of chromosomes surrounding it.

- $dist_x = \sum_{j=1}^{k} \frac{f_j(x_{left}) - f_j(x_{right})}{f_j^{max} - f_j^{min}}$
  where f_j is jth objective function, and number of objectives is k.
- A partial order $\prec$ between chromosomes are defined as:
  $a \prec b$ if $r_a \prec r_b$
  or $(r_a = r_b)$ and $(dist_a \succ dist_b)$ back

Job grouping; Critical length

Critical length

It denoted as $crit(j)$ refers to the longest distance from j_{entry} to j_{exit} passing through the job j.

The Upward Critical length of job j is the longest distance from j to the exit job j_{exit}. It is denoted as $crit_{up}(j)_{<type>}$ where $<type>$ is computational and storage. Upward critical length is computed with the equation 3 starting from $jexit$ and moving upward towards j.

$$crit_{up}(j)_{<type>} = job\_size(j)_{<type>} + max_{j' \in succ(j)}(crit_{up}(j')_{<type>}) \quad (3)$$

Similarly, the Downward Critical length of job j is the longest distance from the entry job j_{entry} to j. It is denoted as $crit_{down}(j)_{<type>}$ where $<type>$ is computational and storage. Downward critical length is computed with the equation 4 starting from $jentry$ and moving downward towards j.

$$crit_{down}(j)_{<type>} = job\_size(j)_{<type>} + max_{j' \in pred(j)}(crit_{down}(j')_{<type>}) \quad (4)$$

algorithm

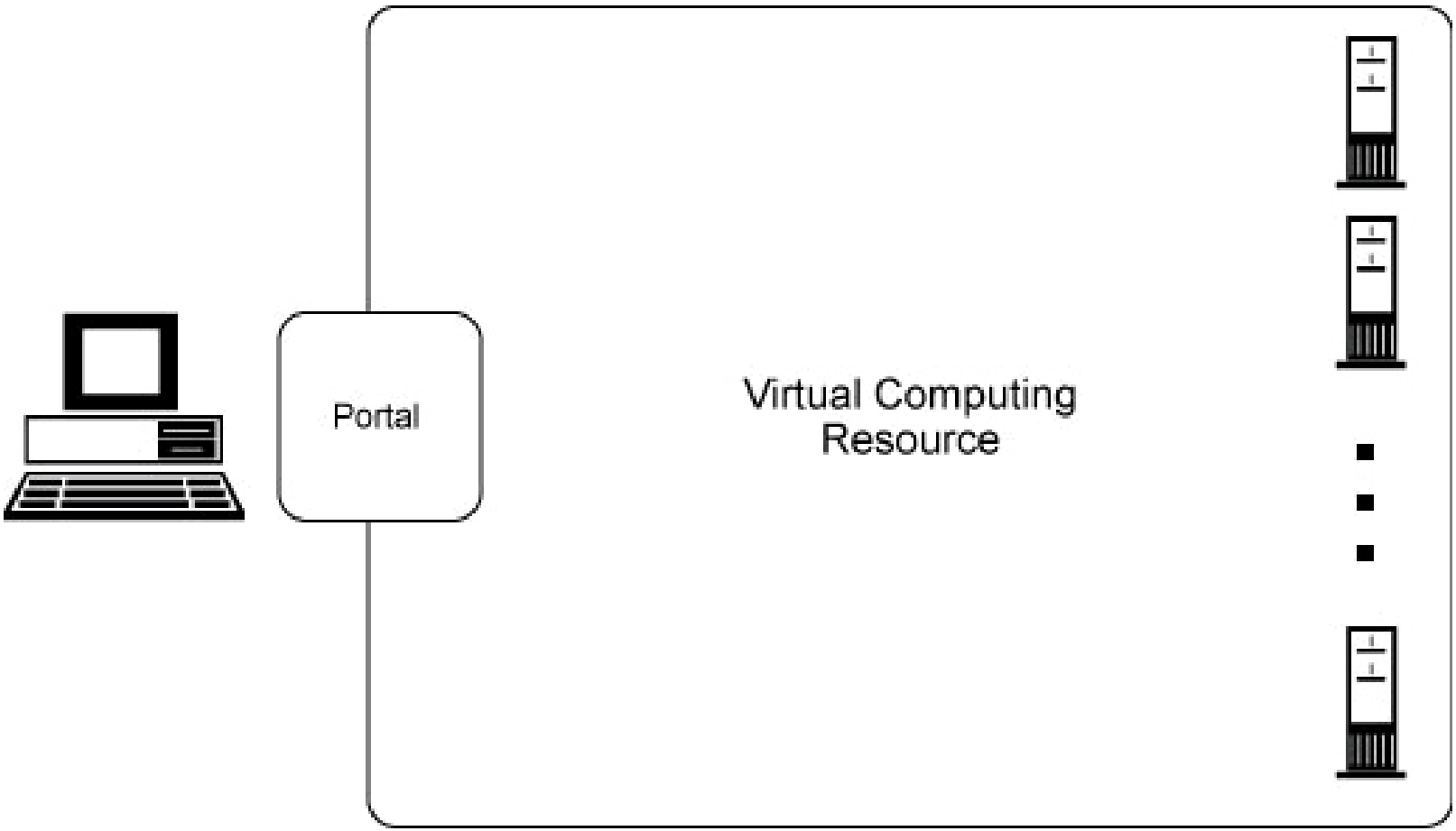
Job Grouping Algorithm

eqn. link back

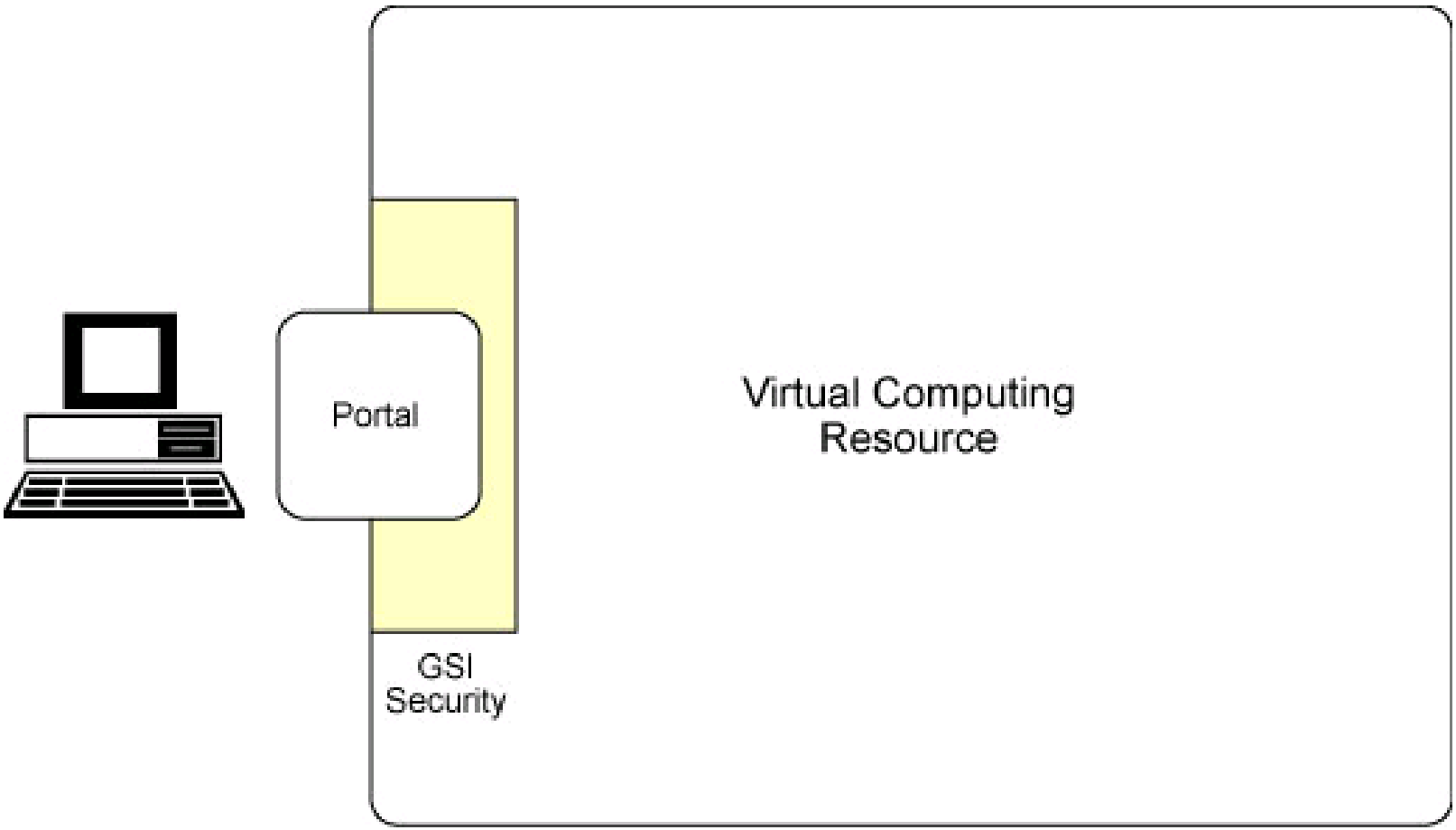
Algorithm 1 Job grouping

Require: Job pool with DAG representation
  Compute $crit_{up}(j)_{<type>}$ for each job j according to the equation 3
  Compute $crit_{down}(j)_{<type>}$ for each job j according to the equation 4
  Compute $crit_{<type>}$ for each job j
  while Job $a \in$ job pool exists, where a is unprocessed fine-grained job do
    $flag \leftarrow 0$
    while a is fine-grained job and $flag = 0$ do
      for each $b \in adjacent\_node(a)$ and same type i.e. computational or storage do
        Temporary merge adjacent node b and a to form t
        Calculate new $crit_{up}(t)_{<type>}$, $crit_{down}(t)_{<type>}$ and $crit(t)_{<type>}$
        if new $crit(t)_{<type>} \leq crit(j_{entry})_{<type>}$ and $crit(t)_{<type>}$ is minimum till now then
          $merge\_node \leftarrow b$
        end if
      end for
      if $merge\_node$ is found then
        Permanently merge $merge\_node$ with a to form a'
        Change parent and child relation accordingly
        if a' is not fine-grained job then
          $flag \leftarrow 1$
        end if
      else
        $flag \leftarrow 1$
      end if
    end while
  end while

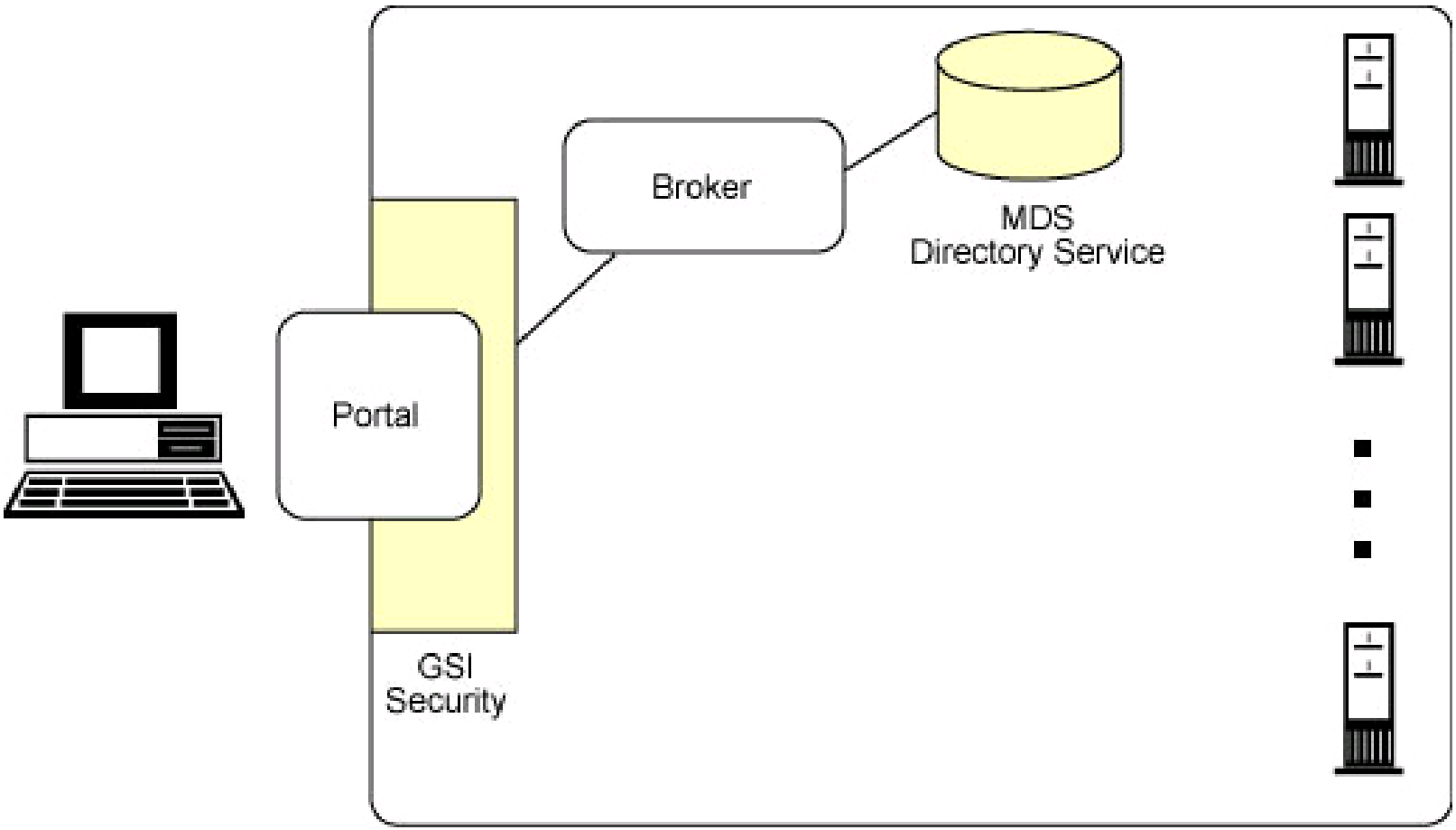
Portal or User Interface



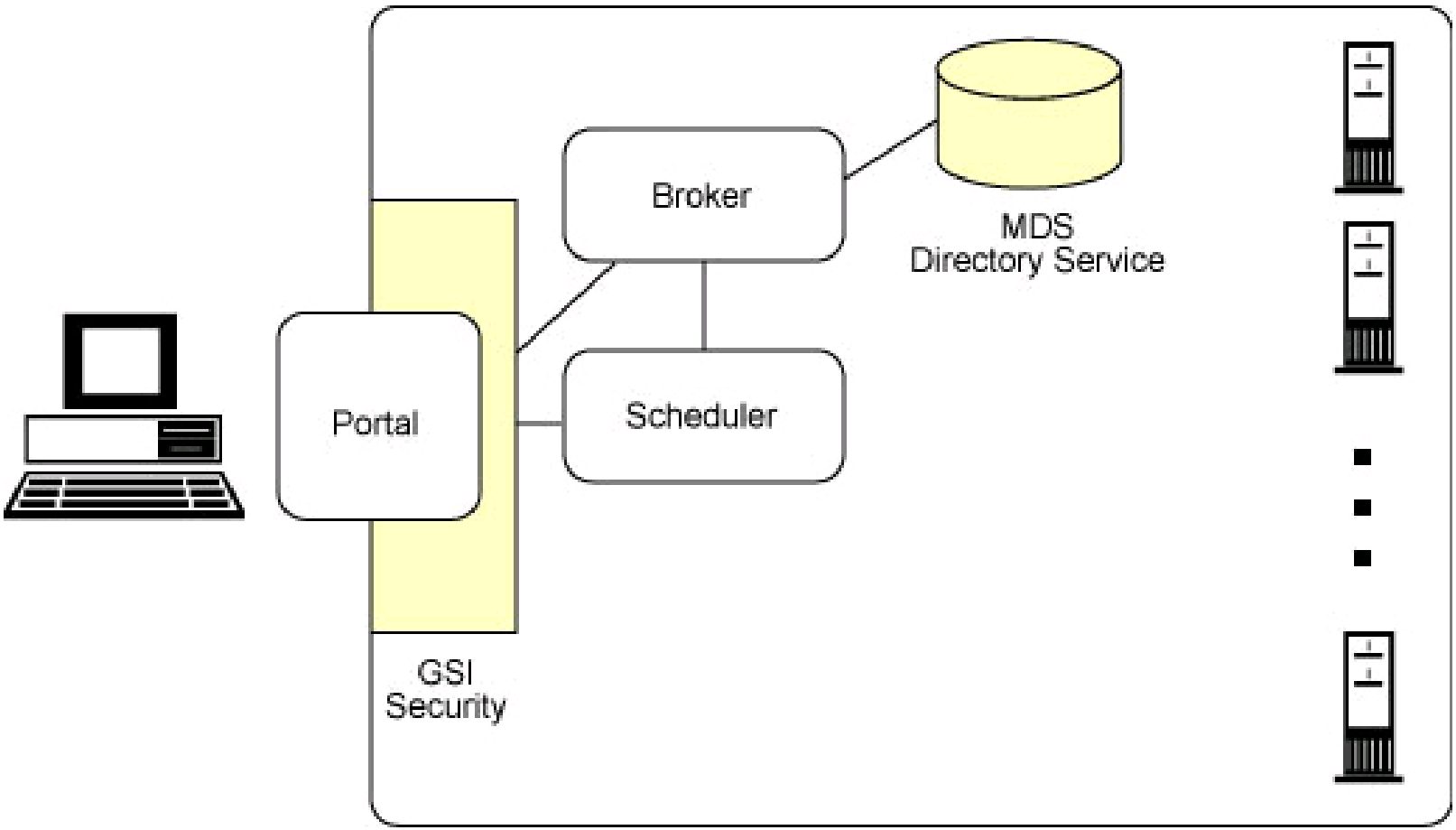
Grid Security Infrastructure (GSI)



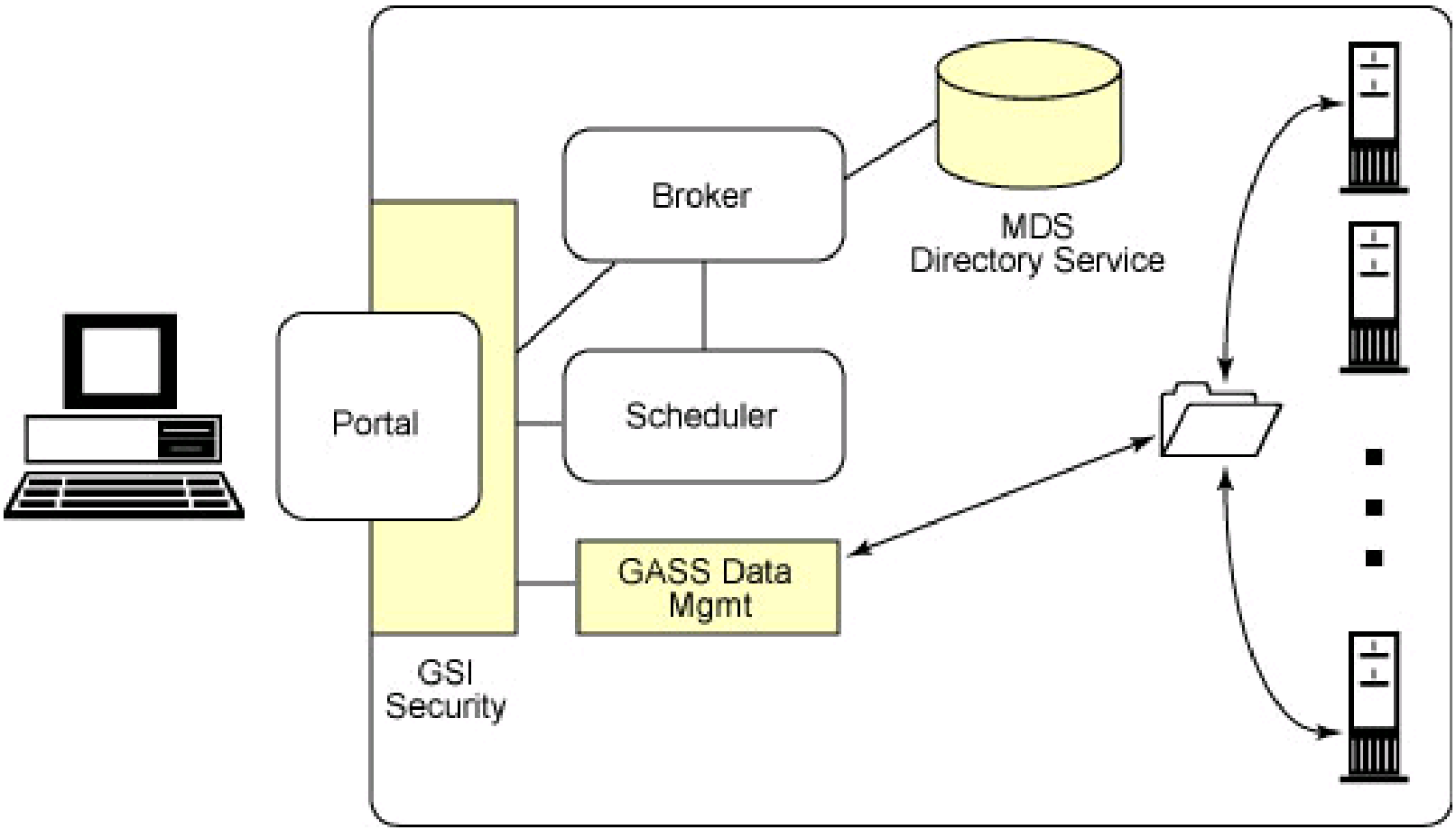
Broker, Monitoring and Discovery Service



Scheduler


Broker
MDS
Directory Service
Portal
Scheduler
GSI
Security

Data Management



Dispatcher or Grid Resource Allocation Manager