Oregon State University

COLLEGE OF ENGINEERING | School of Electrical Engineering and Computer Science

Implementing Resource Allocation in a Simulated Cloud Datacenter

Rand Black, Sean Bullis, Kira Kopcho, Morel Kopcho

Oregon State University
College of Engineering

Outline

- Review
- CloudSim
- Genetic Algorithm
- Particle Swarm Algorithm
- Particle Swarm Implementation
- Genetic Algorithm Implementation

Oregon State University
College of Engineering

Review

- Cloud data centers need to connect clients to resources in real time
- Growing number of clients and resources creates exponentially complex problem to solve
- Possible Optimizations
  - Makespan (time to complete)
  - Cost
  - Energy use
  - Reliability
- Each client may value different optimizations

Oregon State University
College of Engineering

Review (cont.)

- Challenges
  - Real time allocation with changing resources
  - Different service expectations for different clients
  - Managing energy use for unused resources
  - Predicting loads and estimating available resources
- Solutions?
  - algorithms that find a good enough answer quickly
  - heuristic, meta-heuristic, and hybrid implementations

Oregon State University
College of Engineering

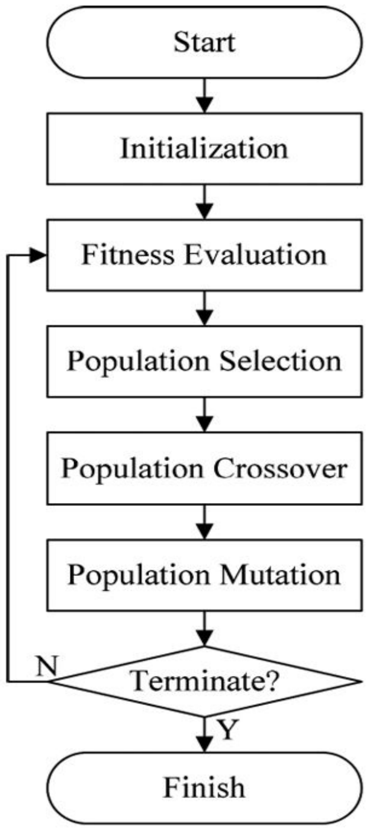
Cloudsim Plus

- Java based modelling and simulation of cloud computing architectures and application services.
- Compared to NS3 and the main version we were able to find more documentation applicable to our project, the software was better geared toward cloud computing, and more user friendly than the other options.

Oregon State University
College of Engineering

Genetic Algorithm [1]

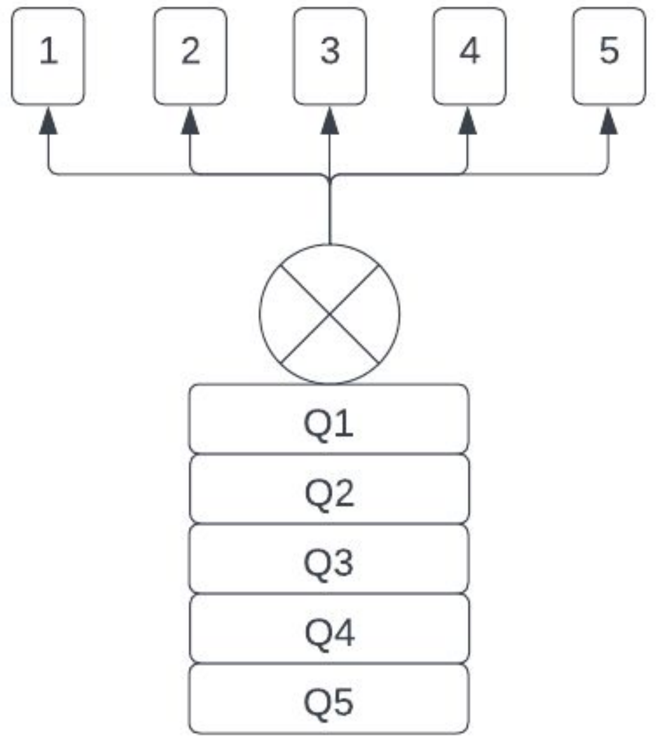
- meta-heuristic
- treats each scheduling task as a gene
- group of scheduling tasks is a chromosome
- randomized set of chromosomes are initialized

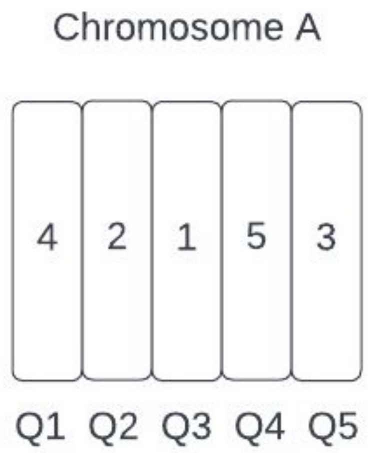


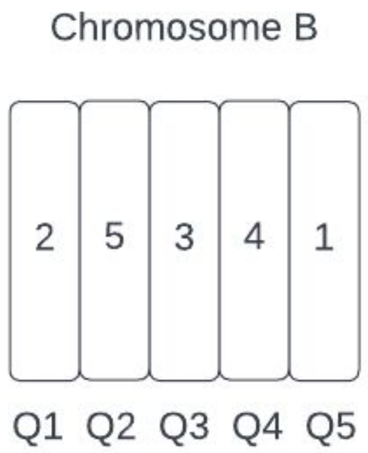
(a) Framework of GA

[1] Z.-H. Zhan *et al.*

Genetic Algorithm







Genetic Algorithm

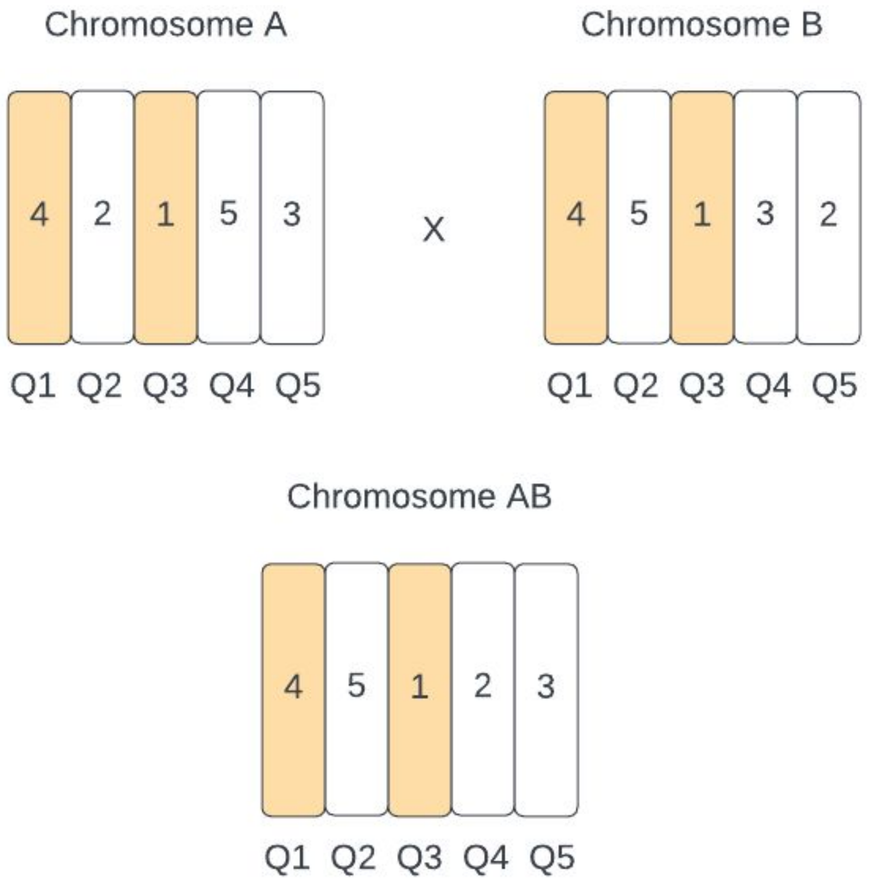
- “fitness” is evaluated for each chromosome

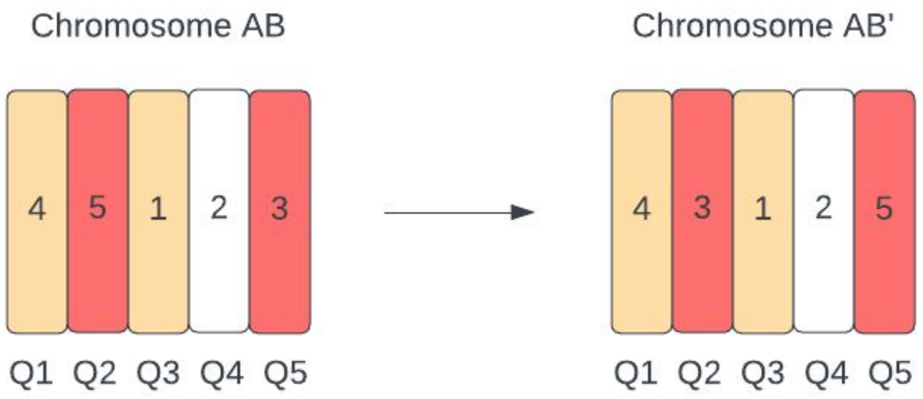
- best performing chromosomes are crossed to create new “generation” (maintain top performers without crossing)

- introduce mutations on subset of new generation

Oregon State University
College of Engineering

Genetic Algorithm





Oregon State University
College of Engineering

Genetic Algorithm

- “fitness” of new generation is determined and the process is repeated
- repeated for N generations where N is predetermined or until the average weight of the top performers doesn’t change from generation to generation
- chromosomes from N-th generation represents highly viable solutions

Oregon State University
College of Engineering

Particle Swarm Algorithm[2]

- meta-heuristic
- Data points referred to as particles simulate the general flight pattern of a flock of birds
- Algorithm behaviors:
  - particles should sense environment and estimate own behavior
  - particle use each other as a comparative reference
  - particles mimic each other

Particle Swarm Algorithm[2]

Oregon State University College of Engineering

“1) Initialize a population of particles

2) Evaluate the desired optimization function at the position given by each particle

3) Compare the evaluation of each particle with its last best value, called **pbest**. If current value is better than **pbest**, adopt it as the new **pbest**

4) Compare the current evaluation with the best value visited by the entire population, called **gbest**. If the current value is better than **gbest**, adopt it as the new **gbest**

5) Update the velocity (v) and the position (x)

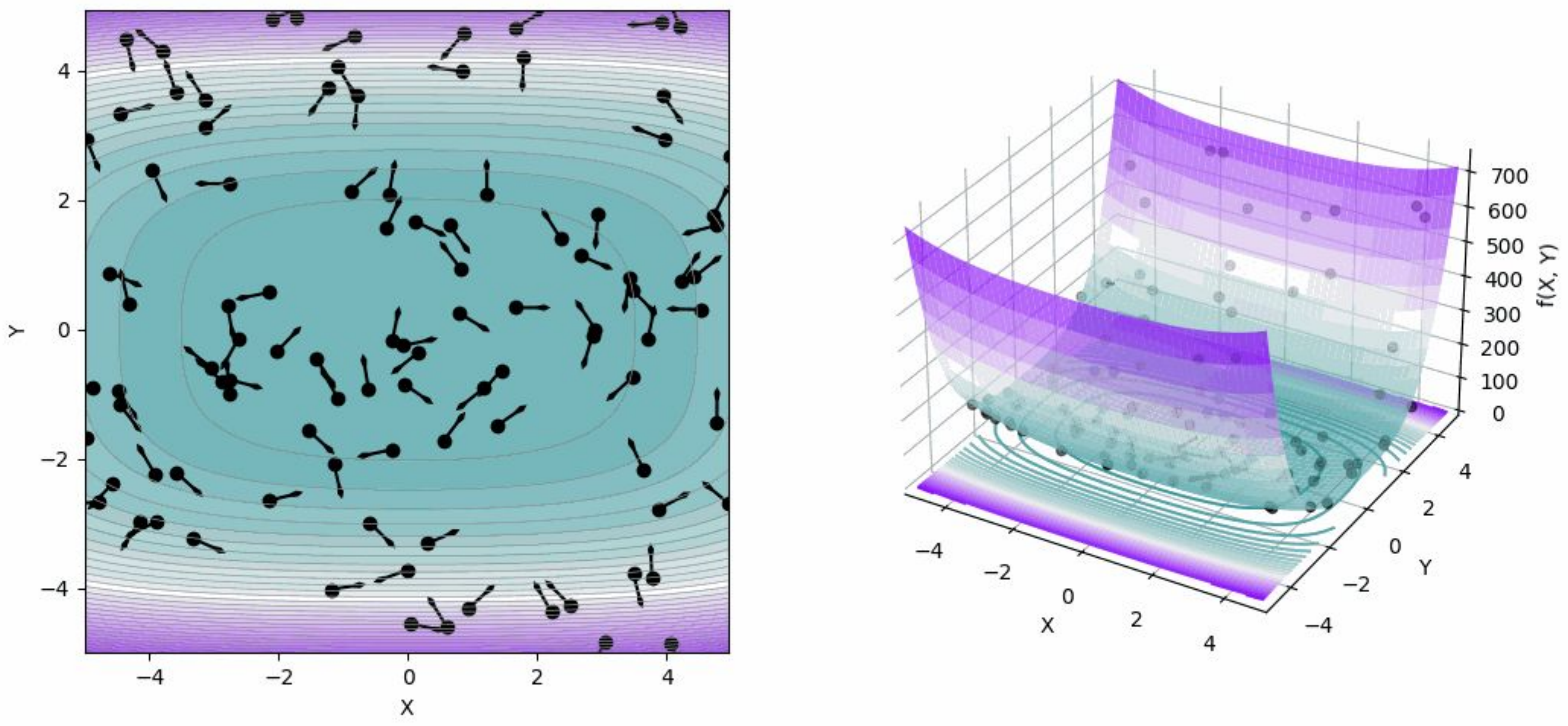
$$v \leftarrow v + c1 \otimes [pbest - x] + c2 \otimes [gbest - x]$$

$$x \leftarrow x + v \quad (2)$$

6) Repeat from step 2”

Particle Swarm Algorithm

Zakharov function - [1/100] w:0.800 - c_1:3.500 - c_2:0.500



Oregon State University
College of Engineering

Simulation Parameters

Vm Parameters:

RAM: 512 MB

CPU Cores: 2

Bandwidth: 1000 MB/s

Image Size: 1000 MB

MIPS: Fixed Array: [500, 100, 1500, 1500, 2500]

No of VMS used: 5

VMM: Xen, OS: Arch

Cloudlet (Task) Params:

Processing Element: 1

MIPS: Random int between 100 and 10000

Cloudlet Numbers (No of tasks):

100, 500, 1000, 1500, 2000

Oregon State University
College of Engineering

GA Setup

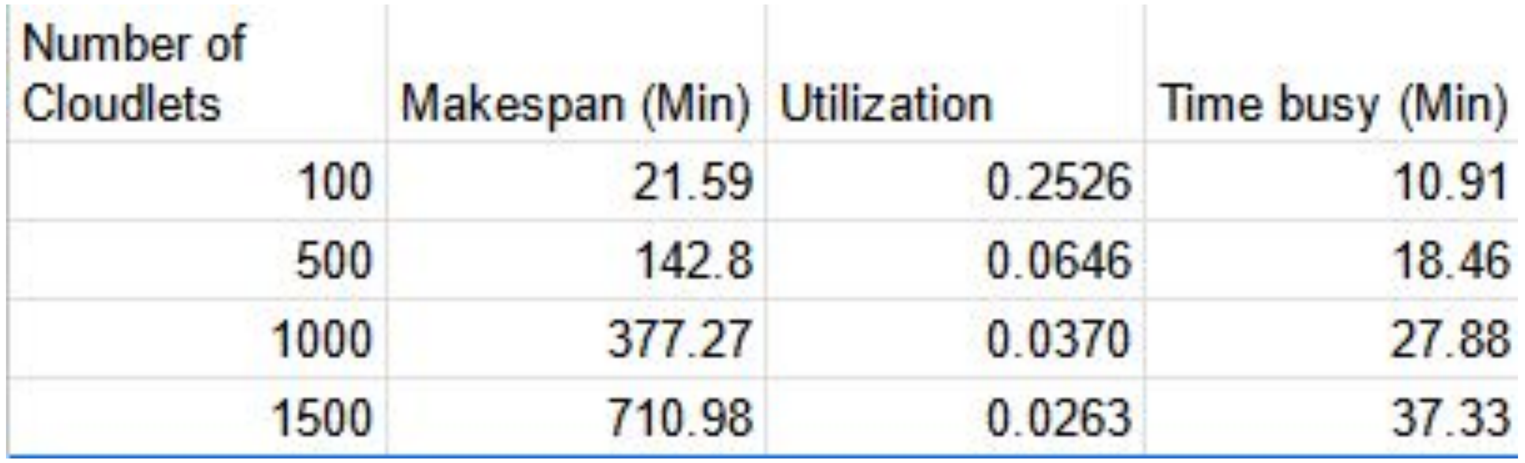
- Population: size 60, run for 50 generations
- Selection Policy: Tournament, size 2, selection prob. 0.75
  - Individuals are randomly sampled from population w/ replacement, individual w/ highest fitness is selected
- Crossover Policy: Single-point
  - Chromosomes are split and recombined at single (random) point

Oregon State University
College of Engineering

GA Setup (Continued)

- Mutation Policy: Swap, mutation prob. 0.9
  - Individual genes in a chromosome are selected and swapped w/ mutation prob.
- Fitness function: Based on makespan
  - Calculated by finding max task execution time

GA Results



Number of Cloudlets	Makespan (Min)	Utilization	Time busy (Min)
100	21.59	0.2526	10.91
500	142.8	0.0646	18.46
1000	377.27	0.0370	27.88
1500	710.98	0.0263	37.33

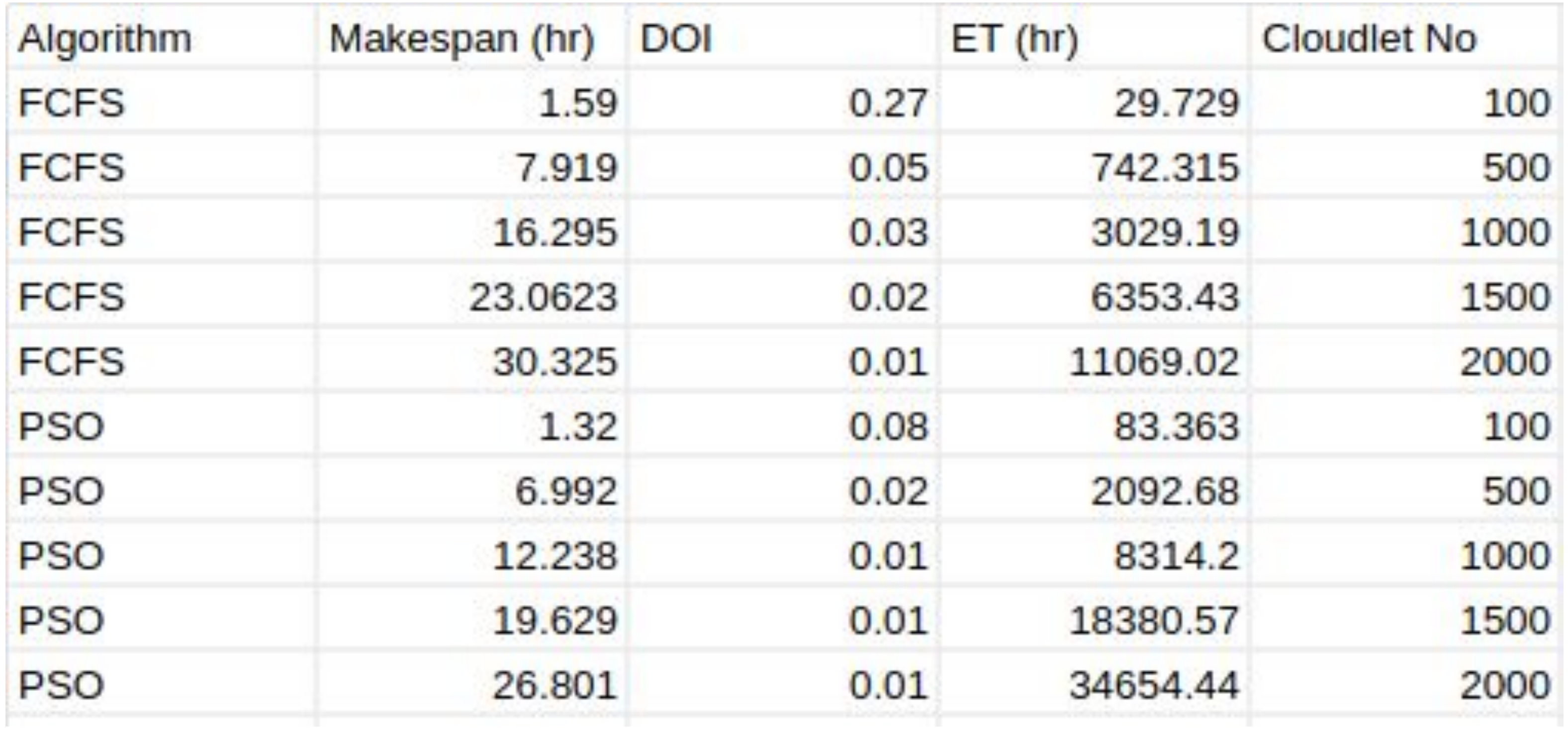
PSO Setup

- Optimizing for Makespan
  - Makespan calculated by getting max task execution time
- Swarm Size: 30
- Iterations: 10000
- Fitness Function:
  - Returns maximum execution time across all vms for a given cloudlet as fitness value

    [3]

$$Fitness = Max[EXC_{VM_1}(j_1), \ldots EXC_{VM_n}(j_m)]$$

PSO VS FCFS: Results



Algorithm	Makespan (hr)	DOI	ET (hr)	Cloudlet No
FCFS	1.59	0.27	29.729	100
FCFS	7.919	0.05	742.315	500
FCFS	16.295	0.03	3029.19	1000
FCFS	23.0623	0.02	6353.43	1500
FCFS	30.325	0.01	11069.02	2000
PSO	1.32	0.08	83.363	100
PSO	6.992	0.02	2092.68	500
PSO	12.238	0.01	8314.2	1000
PSO	19.629	0.01	18380.57	1500
PSO	26.801	0.01	34654.44	2000

Conclusion

- Many more types of scheduling algorithms than covered
- Implementation ultimately comes down to characteristics of cloud system (topology, hardware, expected use, etc.)

References

[1] Z.-H. Zhan *et al.*, “Cloud computing resource scheduling and a survey of its evolutionary approaches,” *ACM Computing Surveys*, vol. 47, no. 4, pp. 1–33, Jul. 2015. doi:10.1145/2788397

[2] P. S. Prampero and R. Attux, "Magnetic particle swarm optimization," 2011 IEEE Symposium on Swarm Intelligence, Paris, France, 2011, pp. 1-7, doi: 10.1109/SIS.2011.5952575.

[3] H.S Al-Olimat, R.C Green, and M. Alam, “Cloudlet Scheduling With Population Based Metaheuristics” in *SummerSim 14’*:, Monterey, CA, USA, Jul. 2014