A Multi-objective Approach for the Multi-level Scheduling of Large Workloads in Multicore Distributed Systems

Santiago Iturriaga[1], Bernabé Dorronsoro[2], Andrei Tchernykh[3], Sergio Nesmachnow[1], Pascal Bouvry[4]

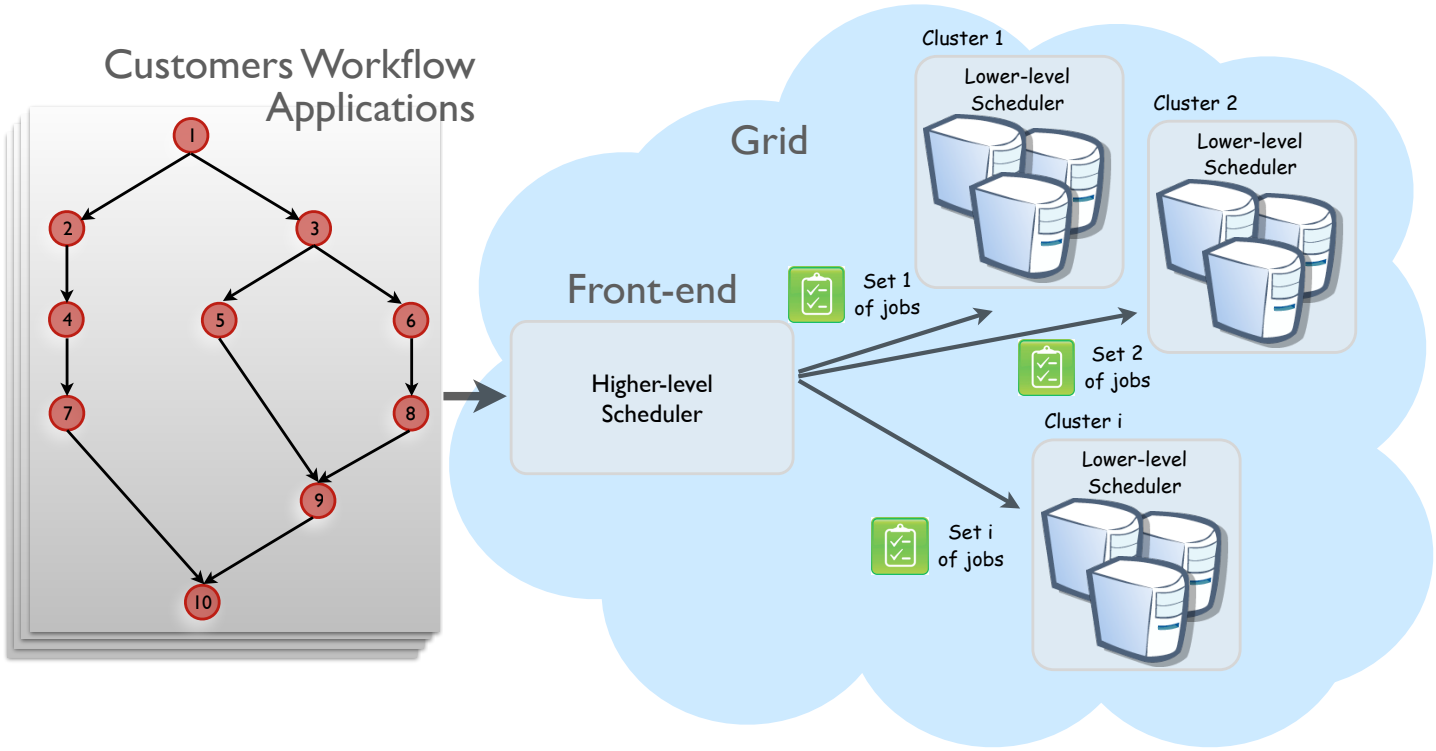
[1] Univ. de la República - UY
[2] University of Cadiz - ES
[3] CICESE Ensenada - MX
[4] University of Luxembourg - LU

B. Dorronsoro, S. Nesmachnow, J. Taheri, A. Y. Zomaya, E.-G. Talbi, P. Bouvry, A hierarchical approach for energy-efficient scheduling of large workloads in multicore distributed systems, *Sustainable Computing: Informatics and Systems*, 4(4):252-261, 2014.

Introduction

- Resource management in large-scale geographically distributed computing centers
  - Big supercomputers
  - High performance computing centers
  - Cloud infrastructures
- Minimize energy consumption in conflict with
  - Performance
  - QoS (overdue deadlines)
- Two-level approach
  - Higher level: decides the mapping between jobs and data-centers
  - Lower level: schedule jobs within each data-center with energy consumption considerations
- Use of multi-objective algorithms to analyze the quality of our two-level schedulers

The Problem



Minimize

- Makespan
- Energy Consumption
- Penalization Cost due to missed deadlines

Applications
- Heterogeneous
- Deadlines

Clusters
- Homogeneous
- Clusters may be different

Makespan Minimization

$$f_M(\vec{x}) = \max_{0 \le p \le k} CT_p$$

$\vec{x}$ represents an allocation

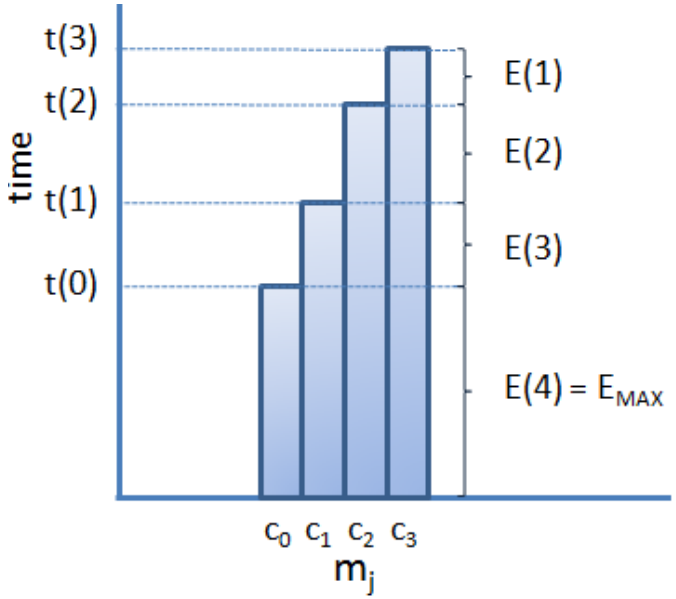
k is the number of available CNs

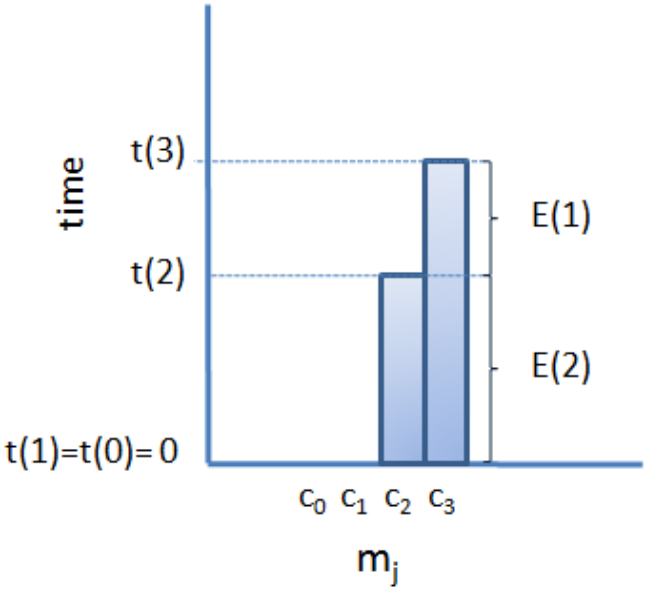
CT_p is the completion time of CN_p

Energy Minimization

$$f_E(\vec{x}) = \left\{ \sum_{r \in CN} \sum_{\substack{jq \in J: \\ f_1(jq)=CN_r}} \sum_{\substack{t_{qi} \in Tq: \\ f_2(t_{qi})=p_j}} EC(t_{qi}, p_j) \right\} + \left\{ \sum_{p_j \in CN} EC_{IDLE}(p_j) \right\}$$

$$E(UC) = E_{IDLE} + (E_{MAX} - E_{IDLE}) \times \frac{UC}{TC}$$





S. Nesmachnow, B. Dorronsoro, J. Pecero, P. Bouvry, Energy-aware scheduling on multicore heterogeneous grid computing systems, The Journal of Grid Computing 11(4):653-680, 2013.

Penalization Cost Minimization

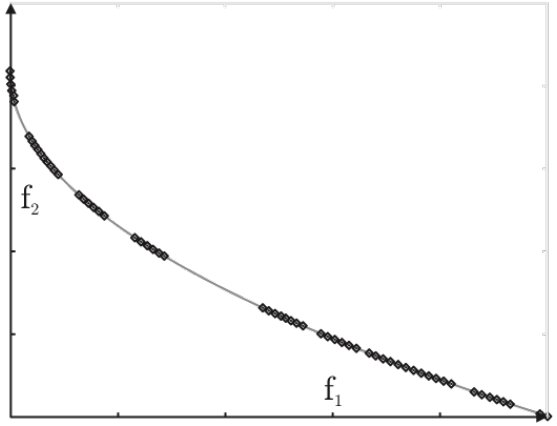
$$f_C(\vec{x}) = \sum_{0 \le q \le n} Cost(\vec{x}, q)$$

$$Cost(\vec{x}, q) = \begin{cases} 0 & if \quad F_q < D_q \\ Penalty_q(F_q) & otherwise \end{cases}$$

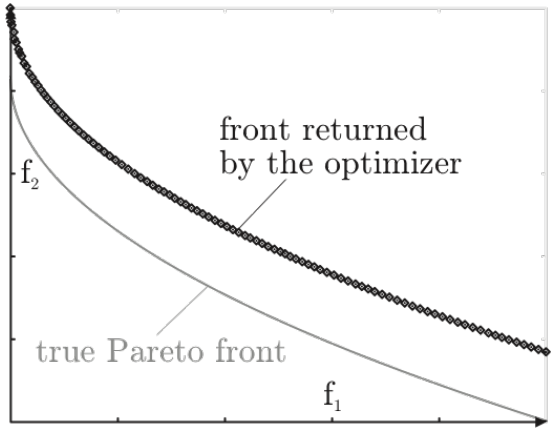
$$Penalty_q(F_q) = \begin{cases} SQRT & = \sqrt{F_q - D_q} \\ LIN & = F_q - D_q \\ SQR & = (F_q - D_q)^2 \end{cases}$$

Multi-objective Optimization

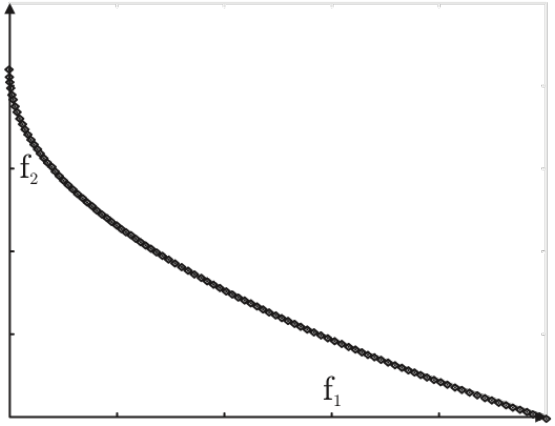
- The goal is to find the Pareto front
- Two key features to measure the quality of solutions
  - Convergence
  - Diversity



Bad diversity



Bad convergence



Ideal case

Algorithms

NSGA-II

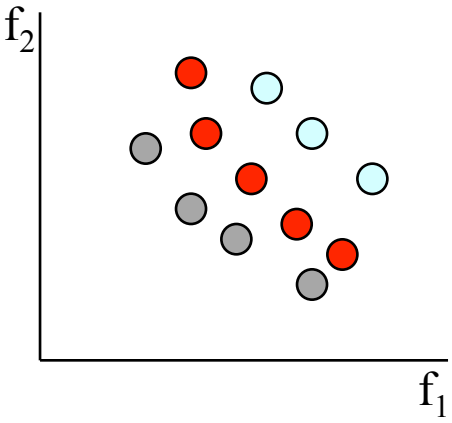
- Reference algorithm
- Panmictic population
- Selection of solutions
  - Ranking
  - Crowding

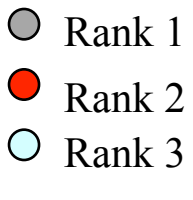
MOCell

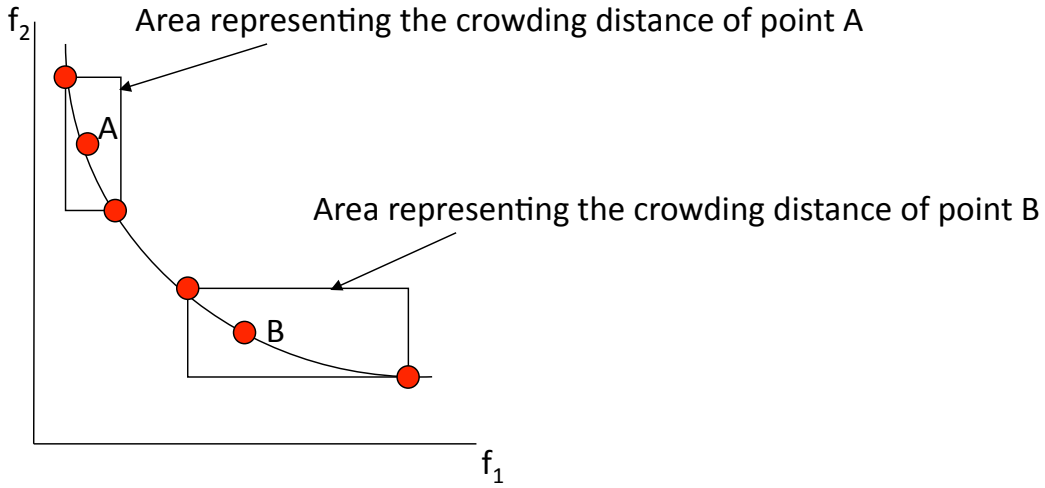
- Cellular population
  - Only next individuals can interact
- External archive
  - Feedback to population

NSGA-II

- Non-dominated Sorting Genetic Algorithm
- Proposed by K. Deb (2002)
- The most popular metaheuristic for multi-objective optimization
- Features
  - Ranking using non-dominated sorting
  - Crowding distance as density estimator

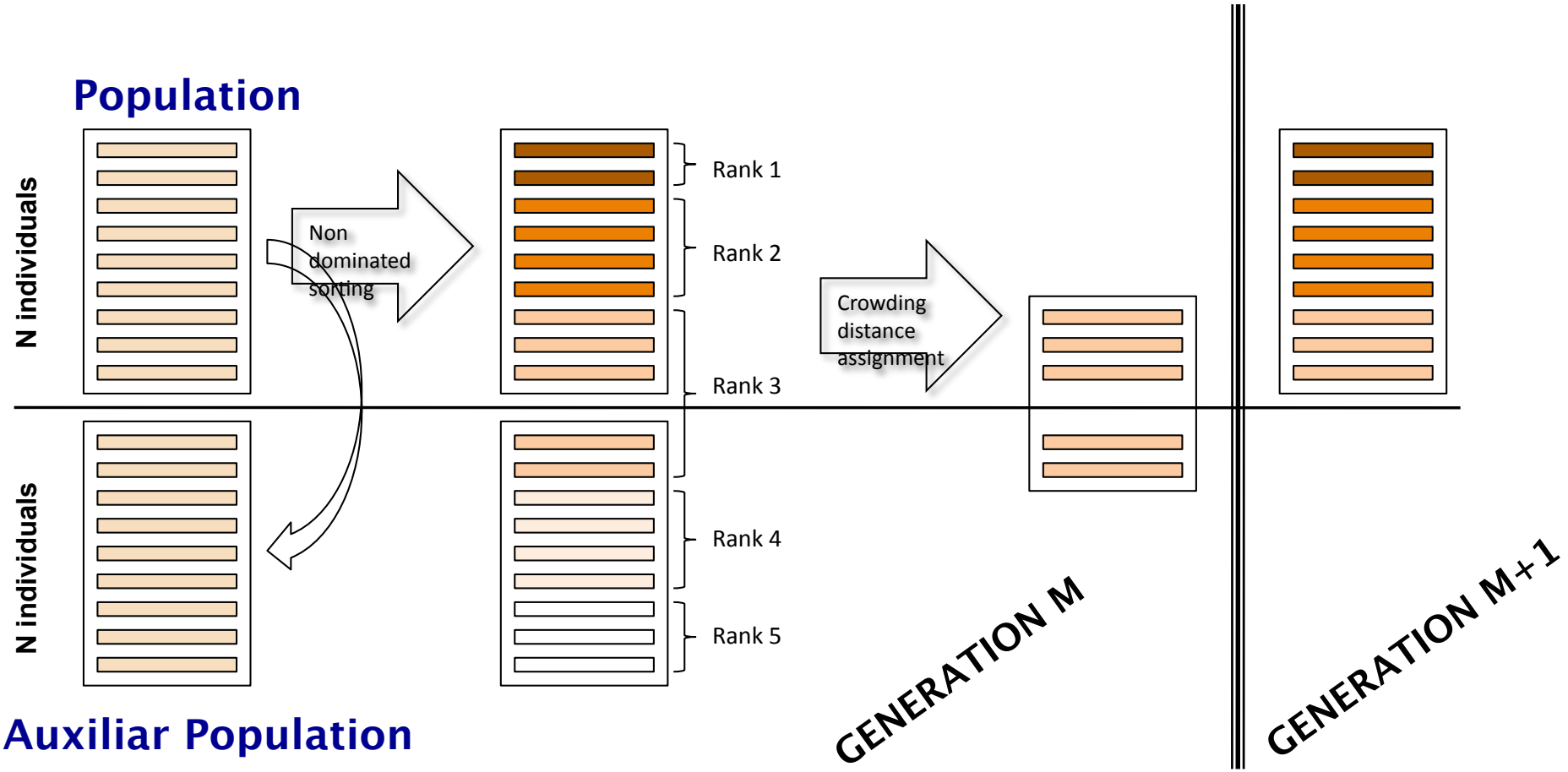




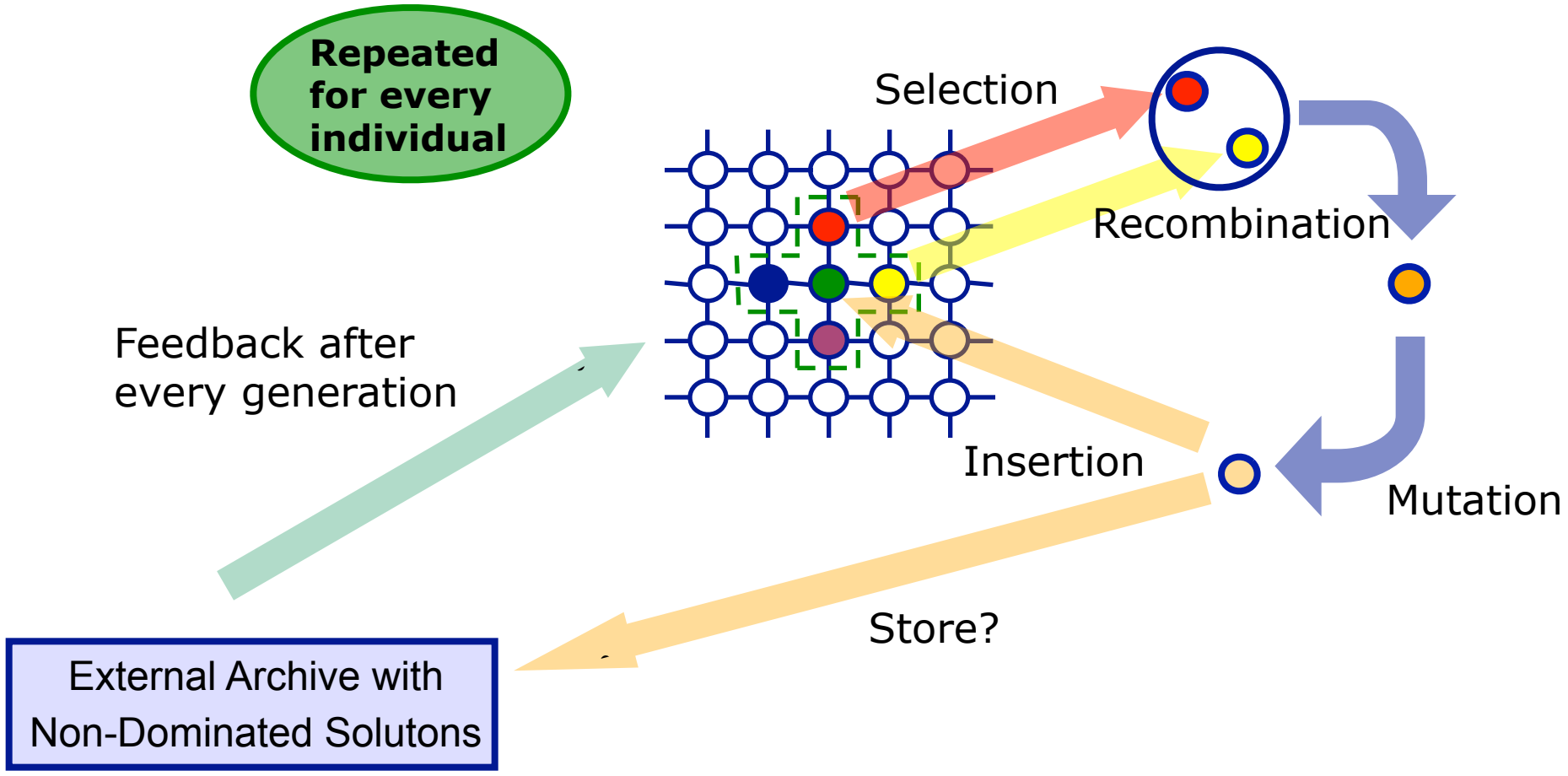


Point B is in a less crowded region than point A

NSGA-II


Population
N individuals
Non dominated sorting
Rank 1
Rank 2
Rank 3
Rank 4
Rank 5
Crowding distance assignment
N individuals
Auxiliar Population
GENERATION M
GENERATION M+1

MOCell


Repeated for every individual
Selection
Recombination
Mutation
Insertion
Store?
Feedback after every generation
External Archive with Non-Dominated Solutons

Problem Instances

- 125 randomly generated workflow batches
  - More than 1000 jobs
  - CNs: between 8 and 64 processors; 1 to 6 cores each

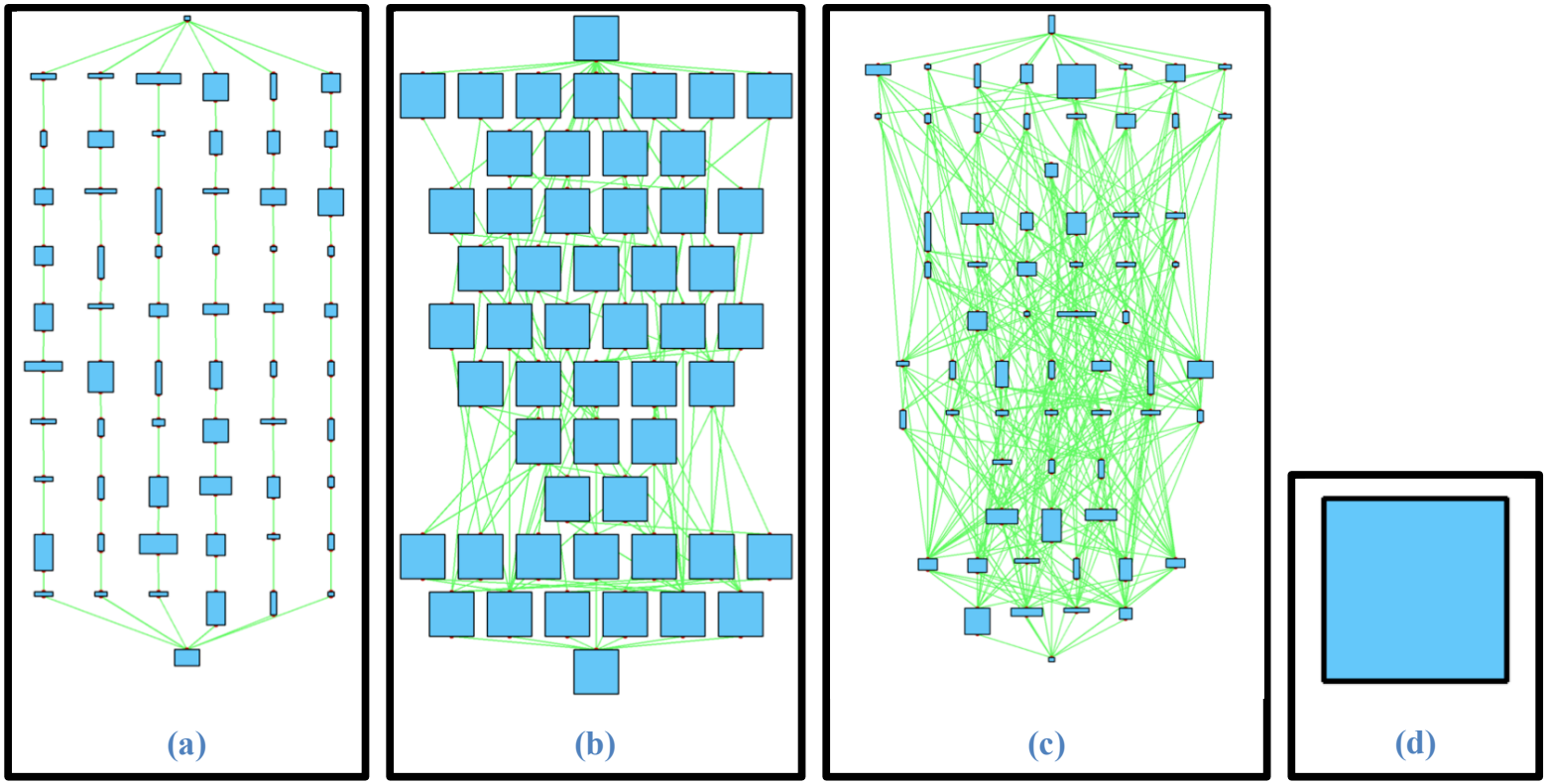


Fig 3. Jobs' shapes: (a) Serious-Parallel, (b Homogenous-Parallel, (c) Heterogeneous-Parallel, and (d) Single-Task

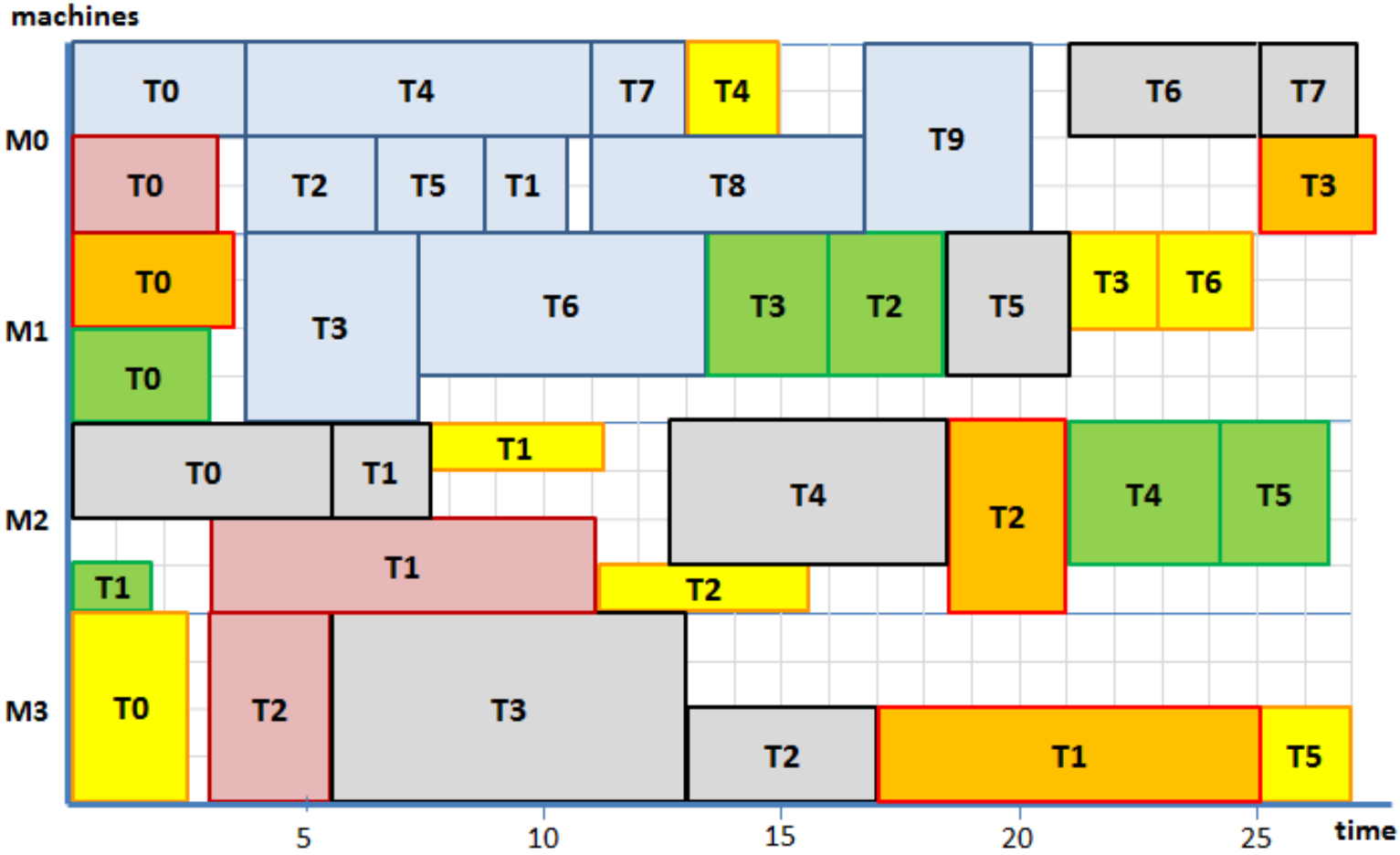
Experiments

- NSGAII and MOCell
  - Map jobs into CNs and establish scheduling order
  - Lower level: EFTH
    - From previous work
- Average results on 25 instances
- Number of independent runs: 30
- Statistical test: Wilcoxon

processor	*frequency*	*cores*	*GFLOPS*	E_{IDLE}	E_{MAX}	*GFLOPS/core*
Intel Celeron 430	1.80 GHz	1	7.20	75.0W	94.0W	7.20
Intel Pentium E5300	2.60 GHz	2	20.80	68.0W	109.0W	10.40
Intel Core i7 870	2.93 GHz	4	46.88	76.0W	214.0W	11.72
Intel Core i5 661	3.33 GHz	2	26.64	74.0W	131.0W	13.32
Intel Core i7 980 XE	3.33 GHz	6	107.60	102.0W	210.0W	17.93

Lower Level Scheduler: EFTH

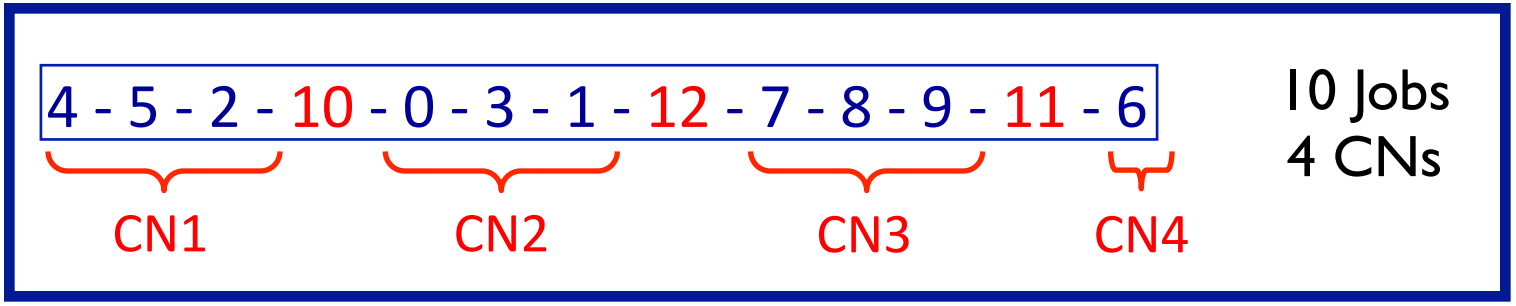
- Based on HEFT
  - Adapted for multicore case: backfilling
- The whole that minimizes task finish time is selected



Problem Resolution

- Problem:
  - Allocate jobs to computing nodes
  - Specify the jobs execution order in every CN
  - Evaluation based on EFTH

- Problem representation
  - Permutation of size $N_{jobs} + N_{CN} - 1$
  - Job IDs $\in [0, N_{jobs}-1]$
  - Splitters $\in [N_{jobs}, N_{CN}-2]$



Operators

- Partially Matched Crossover (PMX)

Parent 1

4 - 5 - 2 - 10 - 0 - 3 - 1 - 12 - 7 - 8 - 9 - 11 - 6

Parent 2

5 - 0 - 6 - 4 - 10 - 7 - 2 - 9 - 12 - 1 - 3 - 8 - 11

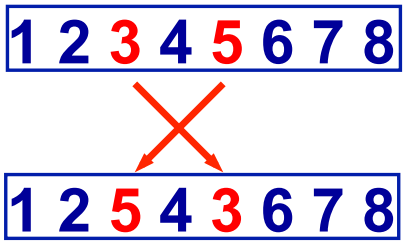
Offspring 1

4 - 5 - 10 - 8 - 0 - 7 - 2 - 9 - 12 - 1 - 3 - 11 - 6

Offspring 2

5 - 0 - 6 - 4 - 10 - 3 - 1 - 12 - 7 - 8 - 9 - 8 - 11

- Swap Mutation

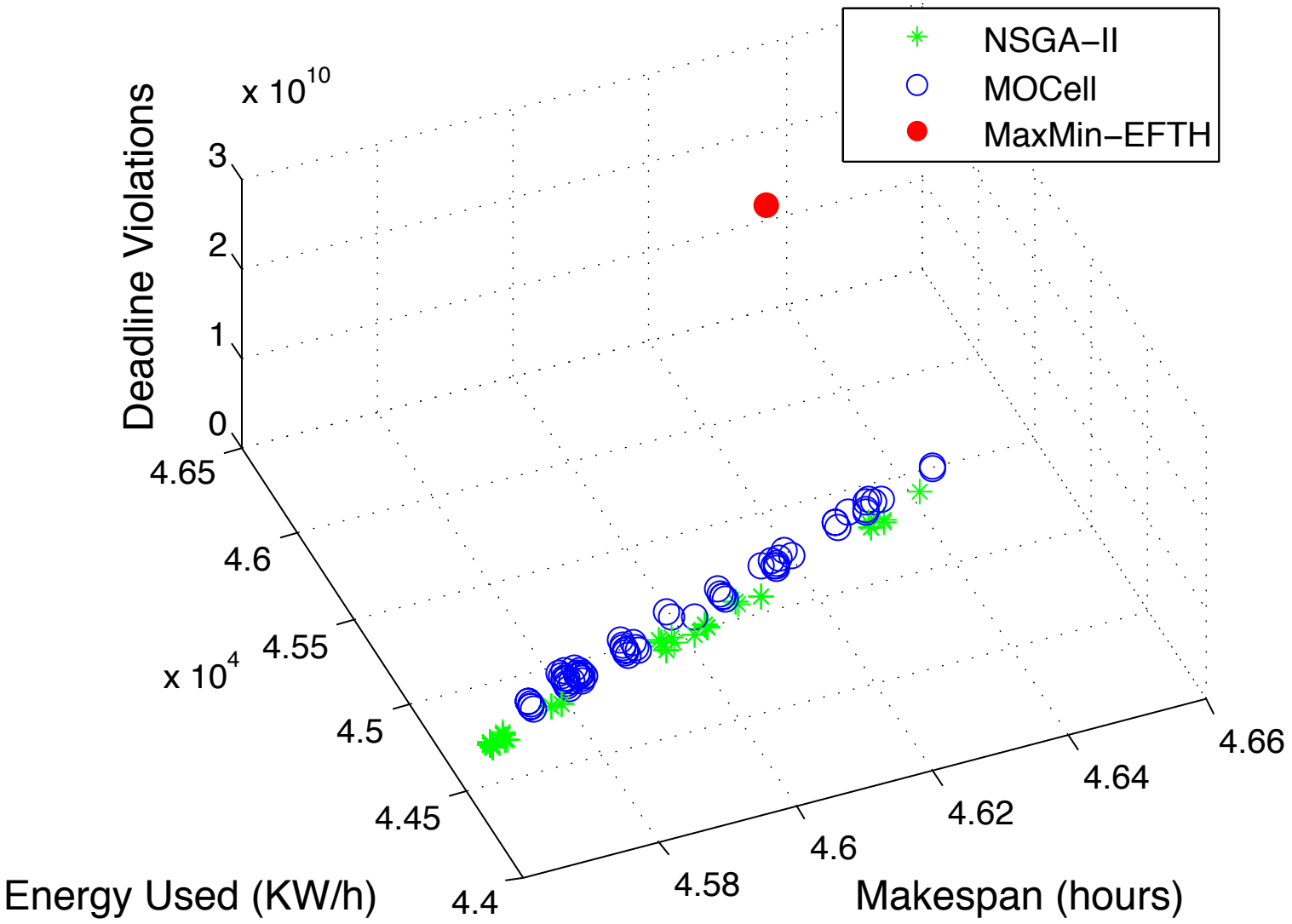


- Solutions Repair: can all CNs execute all assigned jobs?
  - Number of cores

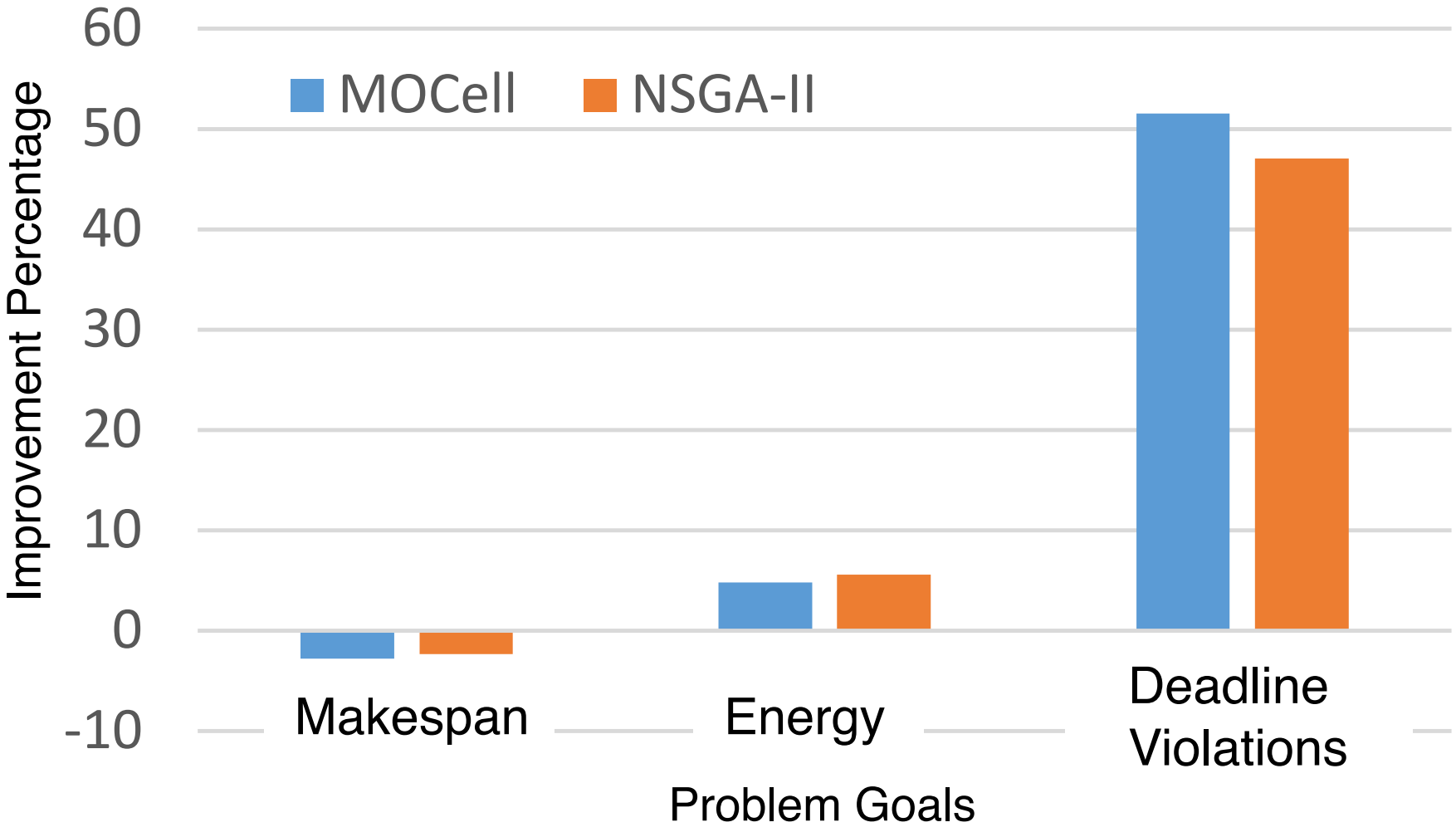
Comparison of the Performance of MOEAs

Instance	HV		spread		IGD	
	MOCell	**NSGA-II**	**MOCell**	**NSGA-II**	**MOCell**	**NSGA-II**
Heterogeneous	0,43±0,06	0,59±0,07	1,00±0,02	1,32±0,03	0,06±0,03	0,04±0,01
Homogeneous	0,57±0,07	0,65±0,06	0,80±0,05	0,90±0,02	0,04±0,01	0,03±0,01
Serious Par.	0,57±0,07	0,65±0,06	0,95±0,03	1,35±0,02	0,05±0,02	0,04±0,01
Single Task	0,57±0,07	0,65±0,06	0,87±0,02	0,98±0,04	0,04±0,01	0,03±0,01
Mix	0,49±0,01	0,58±0,09	0,96±0,04	1,08±0,04	0,04±0,01	0,03±0,01

Sample Solutions for Heterogeneous Inst.



Comparison of MOEAs vs Heuristic



Conclusions

- Resource management in large geographically distributed data-centers with multi-core architectures
- Multiple objectives: Energy; Makespan; QoS
- Solution techniques
  - Two-levels schedule
  - Two MOEAs
- Results
  - NSGAII outperforms MOCell (accuracy)
  - Both outperform best known heuristic
    - Slightly worse in makespan (2.5%)
    - Better in energy (5%) and penalizations cost (50%)
- Future work
  - Enhance MOEAs with population initialization and LS
  - Design better heuristics

Thank you