Scheduling Tasks on a Parallel Platform Powered by Renewable Energy Sources

Ayham KASSAB, Jean-Marc NICOD, Laurent PHILIPPE, Veronika REHN-SONIGO

FEMTO-ST Institute - University of Bourgogne Franche-Comté - ENSMM

GreenDays - July 2018"

HPC in green data centers

Advantages

- HPC can tolerate delay
- HPC's Carbon footprint
- Electricity bills

How

- Power restricted task placing
- Green sources only
- Finish as soon as possible

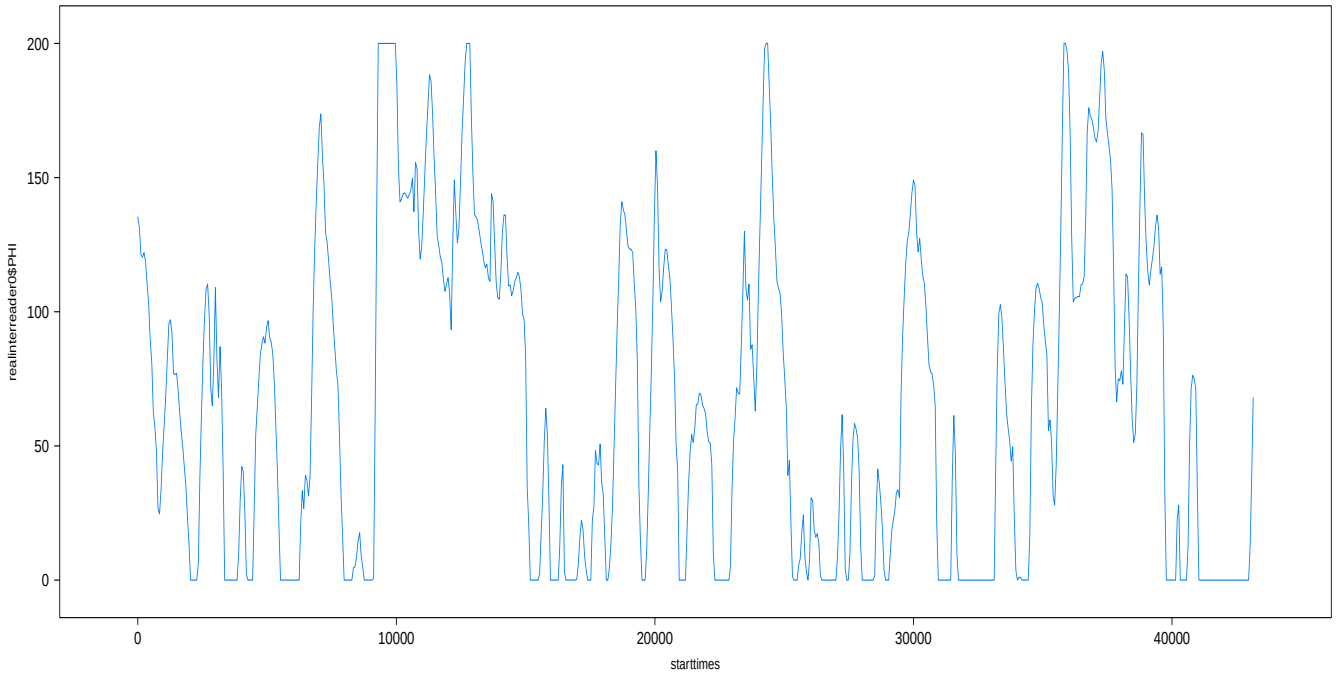
Related Work

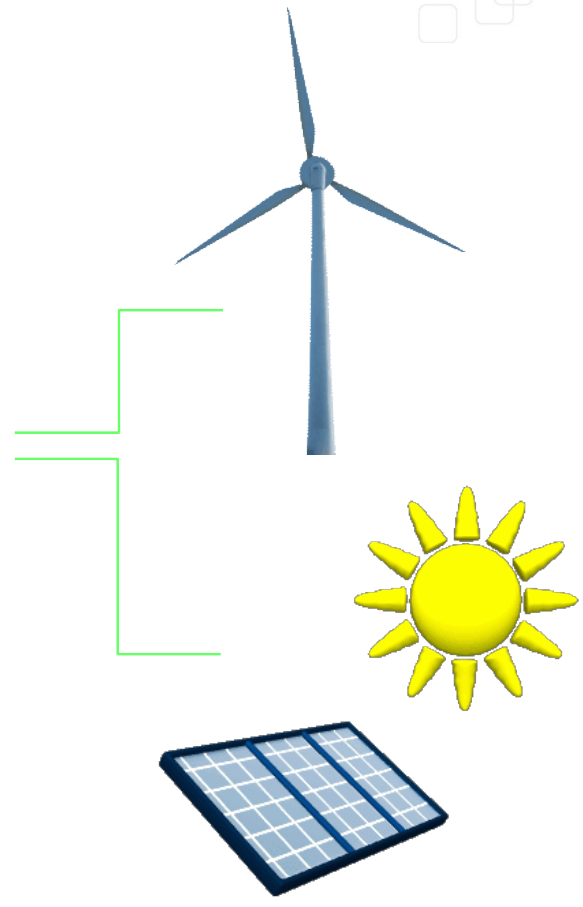
- Dynamic voltage and frequency scaling **DVFS** [Wu2014]
- Machine shutdown [Baptiste2006]
- Cooling [Oró2014]
- Multi-processor scheduling [Ahmad1996]
- Bi-objective evolutionary algorithms (Energy Consumption and Makespan) [Kessaci2011]
- Renewable energy usage in data centers [Lei2016]

Our approach

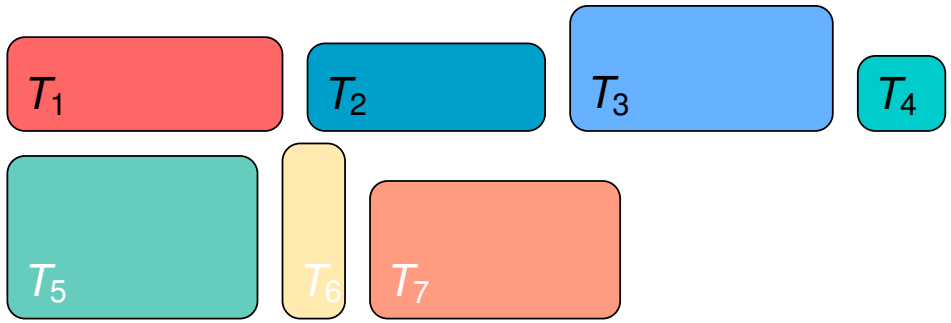
- Green energy turns power into constraints, one criterion optimization problem (makespan minimization)

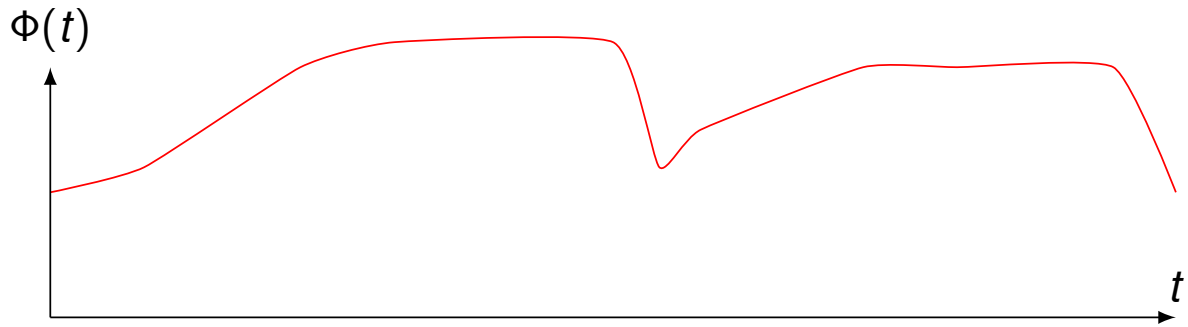
Principle


realinterreader0$PHI
200
150
100
50
0
0
10000
20000
30000
40000
starttimes

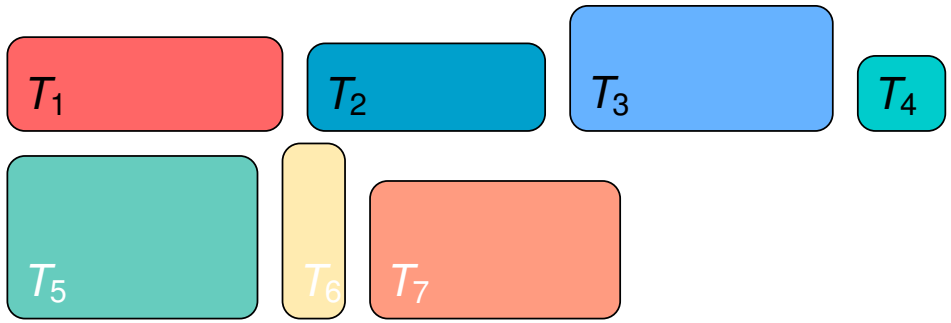


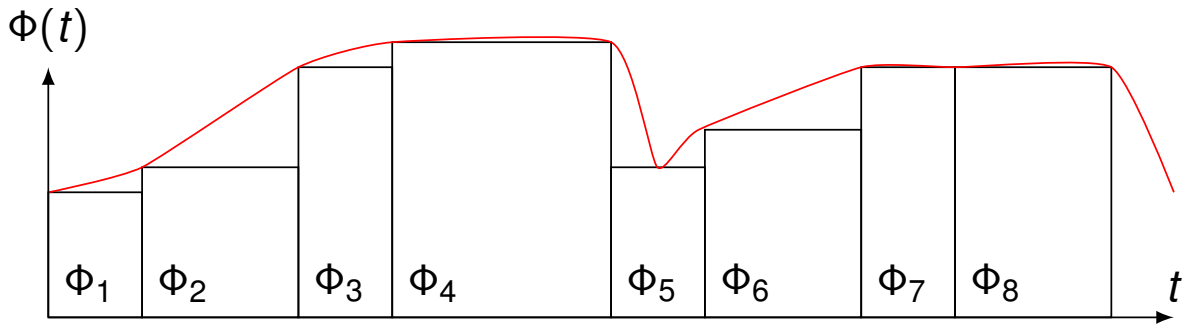
Principle





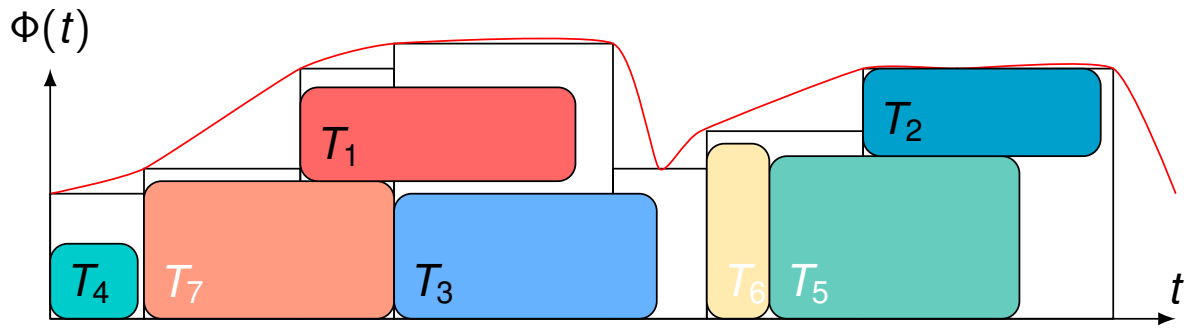
Principle


T_1
T_2
T_3
T_4
T_5
T_6
T_7


$\Phi(t)$
Φ_1
Φ_2
Φ_3
Φ_4
Φ_5
Φ_6
Φ_7
Φ_8
t

Principle

Resulting schedule



Constraints

- $\Phi_x \geq \varphi_i$
- Enough time
- Processor availability

Complexity Results

Without power constraint

	1 Machine		N Machines	
	no-pmtn	pmtn	no-pmtn	pmtn
C_{max}	P	P	NP-complete	P
Flowtime	P	P	P	P

With power constraints

	1 Machine		N Machines	
	no-pmtn	pmtn	no-pmtn	pmtn
C_{max}	NP-complete	P	NP-complete	NP-complete
Flowtime	NP-complete	NP-complete	NP-complete	NP-complete

Complexity Results

Without power constraint

	1 Machine		N Machines	
	no-pmtn	pmtn	no-pmtn	pmtn
C_{max}	P	P	NP-complete	P
Flowtime	P	P	P	P

With power constraints

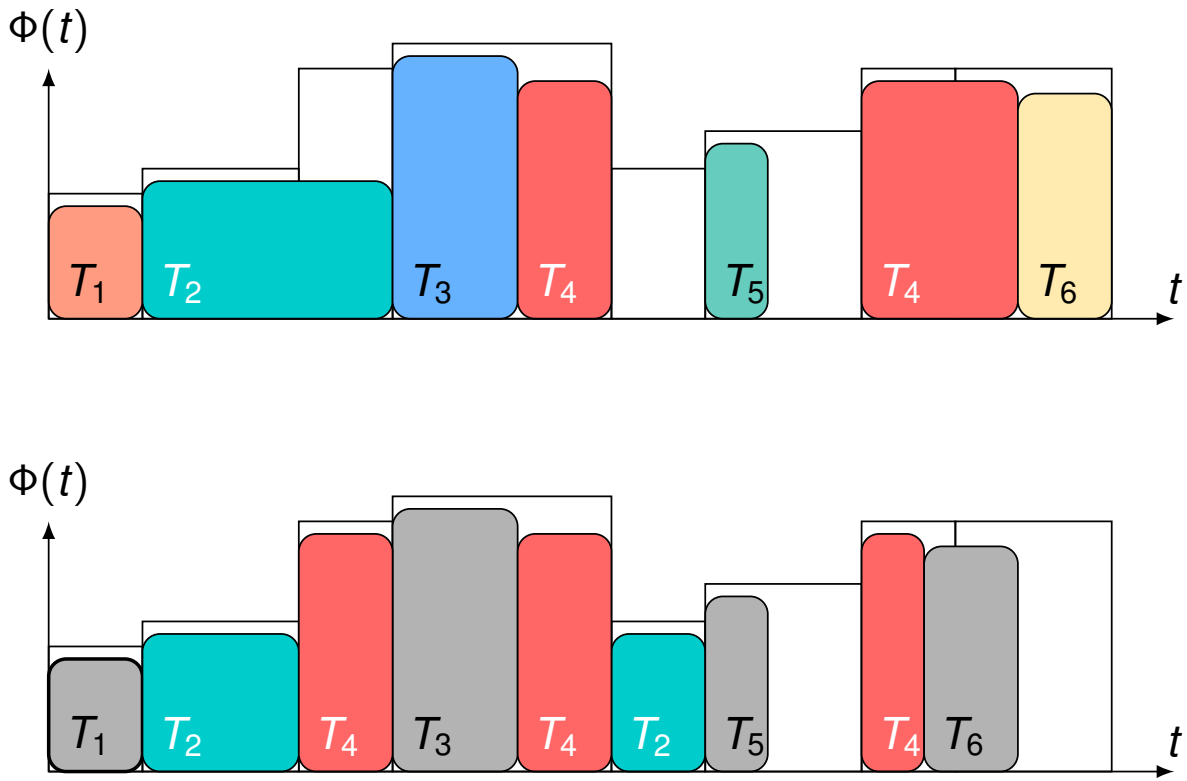
	1 Machine		N Machines	
	no-pmtn	pmtn	no-pmtn	pmtn
C_{max}	NP-complete	P	NP-complete	NP-complete
Flowtime	NP-complete	NP-complete	NP-complete	NP-complete

$1|\varphi_i < \Phi_x, pmtn|C_{max}$

Makespan optimization with preemption on 1 machine

LWRT

Less Wasting Remaining Task

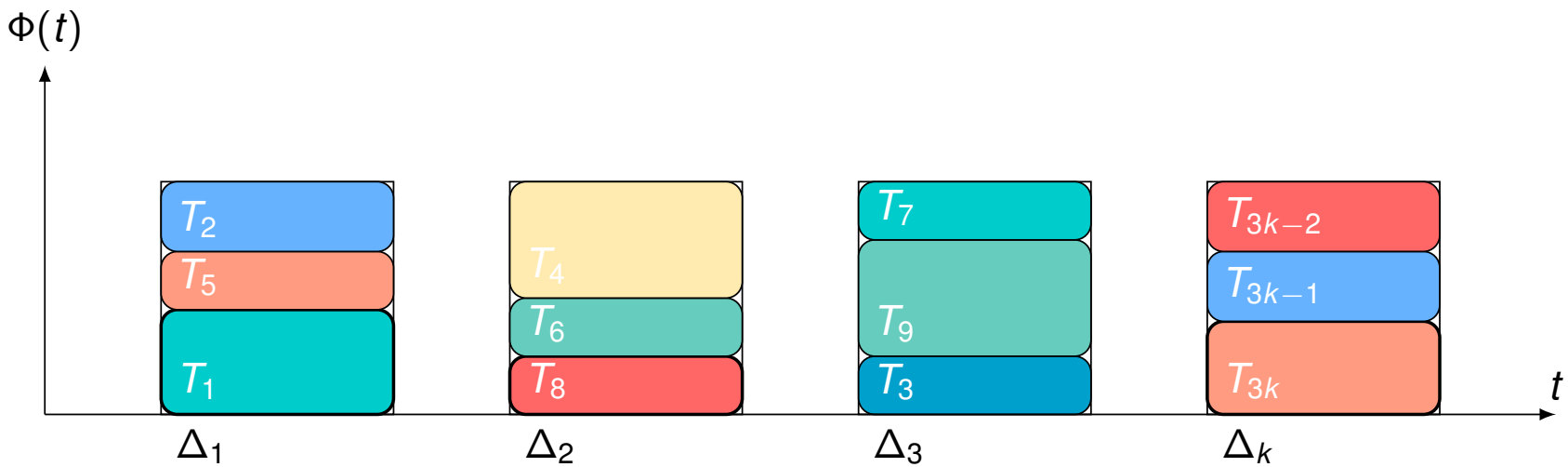


$P|\sum \varphi_i < \Phi_x, pmtn|C_{max}$

Makespan optimization with preemption on N machines

NP-complete
3-PARTITION

- k time slots: δ_k = 1 time unit and unified Φ
- 3k tasks: p_i = 1 time unit and different φ_i, $\Phi/4 < \varphi_i < \Phi/2$, $\sum \varphi_i = k \times \Phi$
- > 3 cores



Heuristics

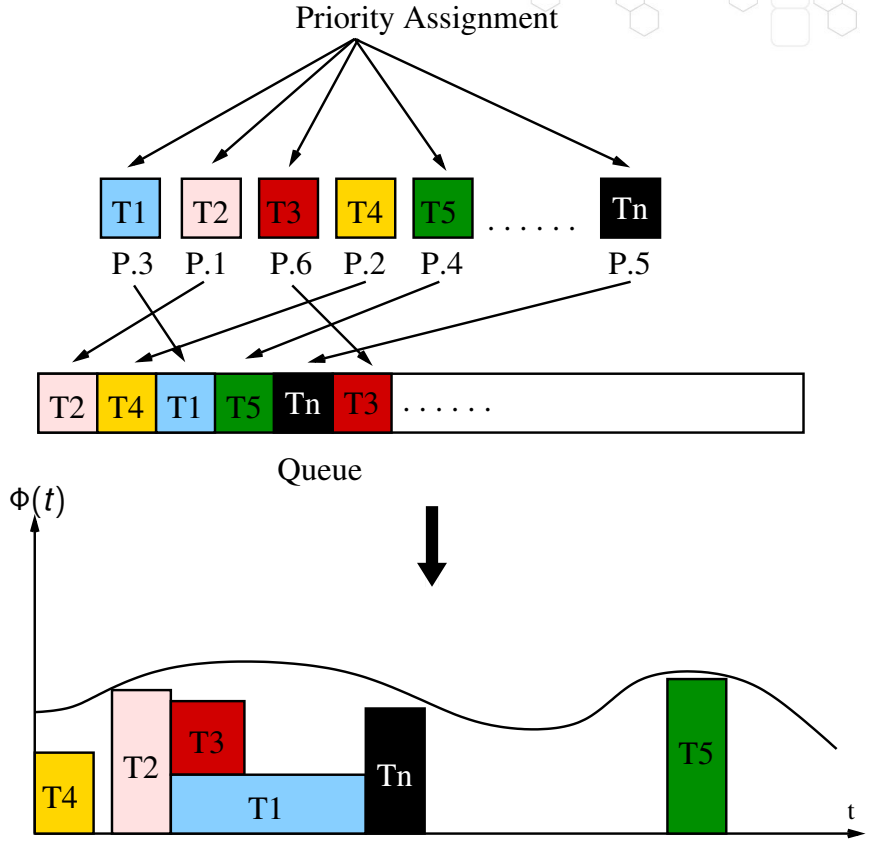
- List Scheduling Algorithm
- Binary Search Algorithm
- Evolution strategy

Note:
All heuristics are designed for the Multi Core Machine case with no preemption

List Scheduling Algorithm

Two steps

1. Priority assignment
2. Greedy scheduling taking power constraints into account



Two dimensional problem

Two properties: p_i and φ_i

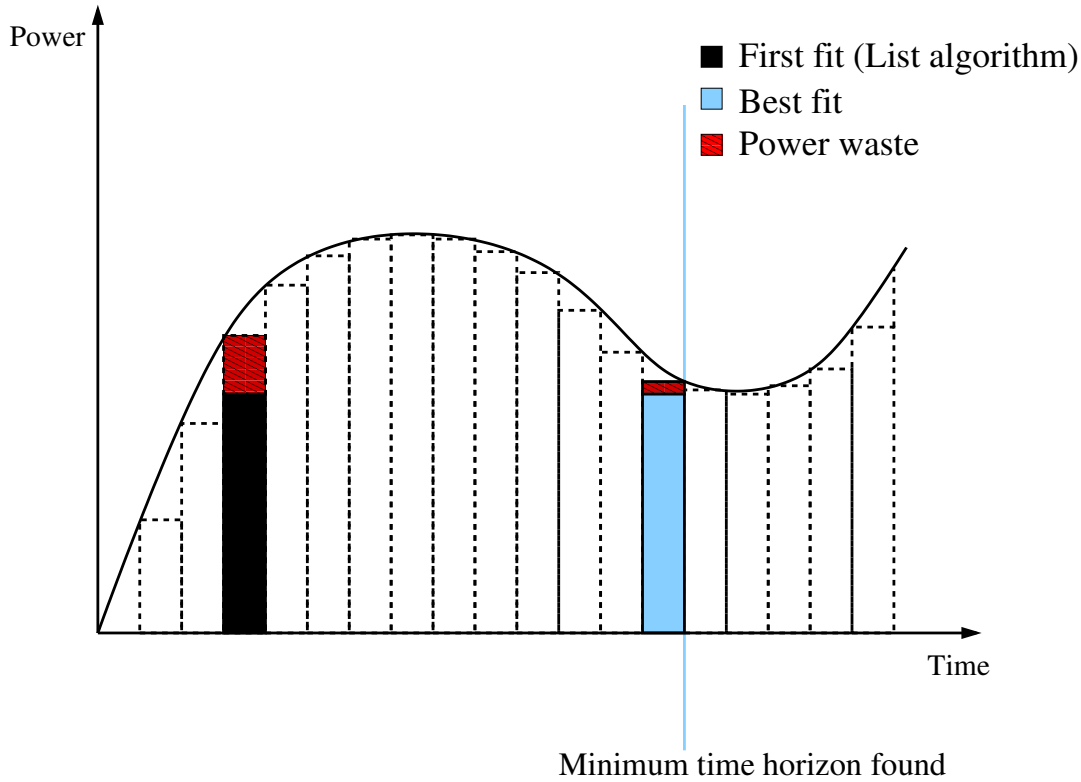
Binary Search Algorithm

Possible solution

Time slot with minimum $\Phi_x - \varphi_i$ (best fit)

instead of the earliest sufficient Φ_x (first fit)

without increasing C_{max}



Binary Search Algorithm

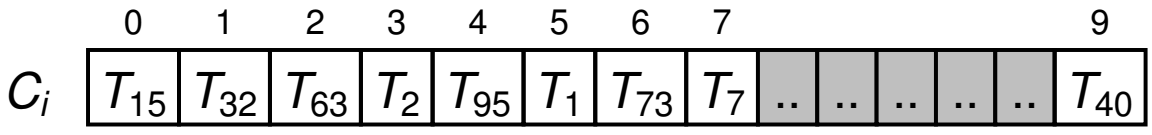
- Uses binary search technique to find the minimum makespan
- Inside each found makespan:
  - Best fit instead of first fit
  - Better exploiting the available power
  - Leading to minimizing the makespan

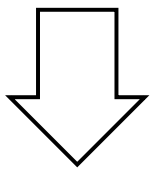
Properties

- Power waste consideration
- Increases the complexity comparing to list scheduling algorithm

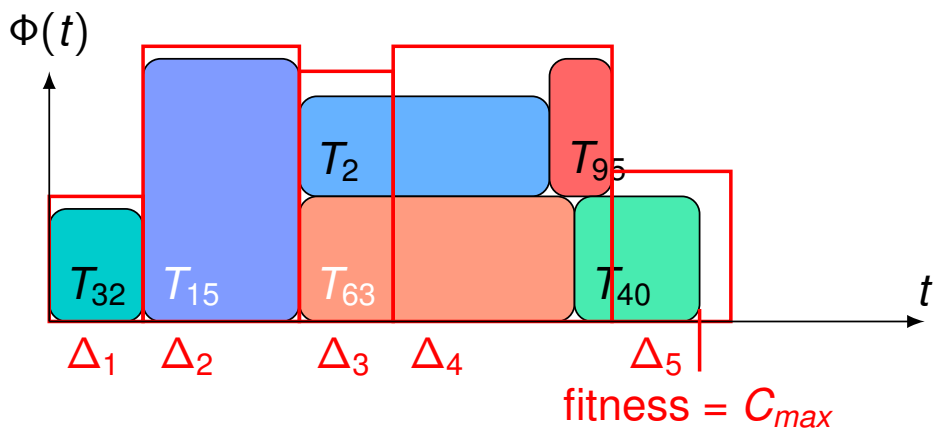
Evolution strategy

Chromosome representation





list scheduling

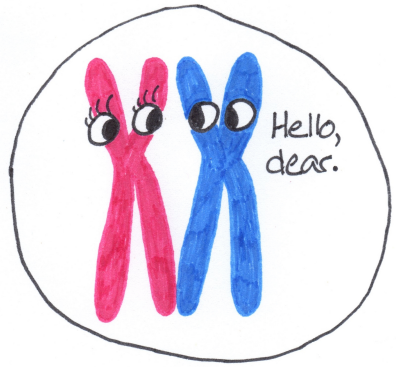


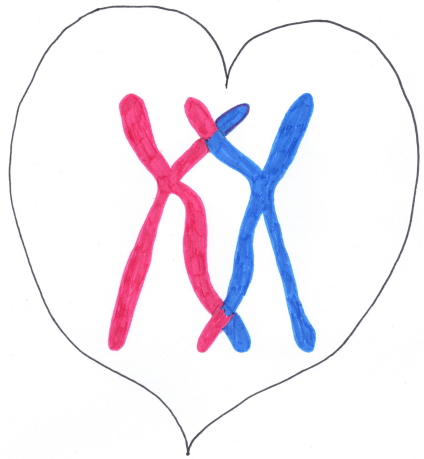
Model

- Independent tasks: p_i, φ_i, $r_i = 0$
- Time slots with various Φ_x
- Multi-Core
- Makespan C_{max}
  - PerMAK = $(C_{max} - useless) \div \sum p_i$
- Flowtime $\sum C_i$
  - PerFLOW = $\sum (C_i - useless_i) \div \sum p_i$
- No preemption

Crossovers

- 1 point crossover 1pX
- 2 points crossover
  - MiddleX
  - OX
- Only mutation NoX





1pX

OnePointCrossOver(C1, C2)

C1 p

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20

C2

5	18	4	8	14	7	17	12	15	11	1	9	20	16	19	6	13	3	10	2

C2\C1[0:p]

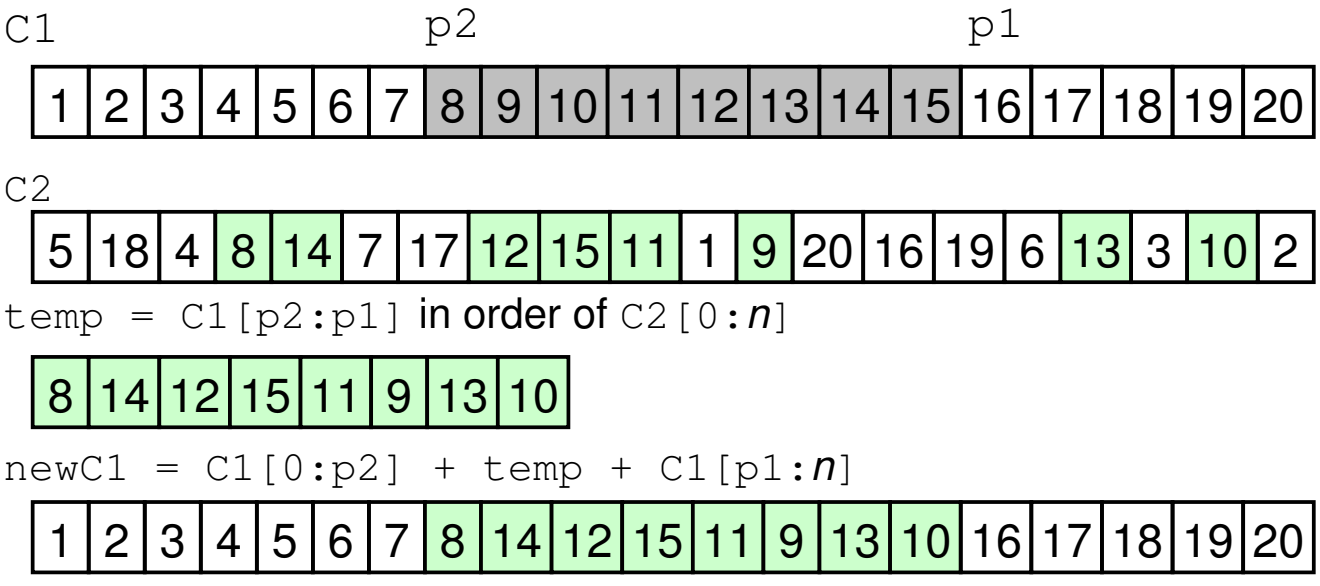
18	14	17	12	15	11	20	16	19	13

newC1 = C1[0:p] + C2\C1[0:p]

1	2	3	4	5	6	7	8	9	10	18	14	17	12	15	11	20	16	19	13

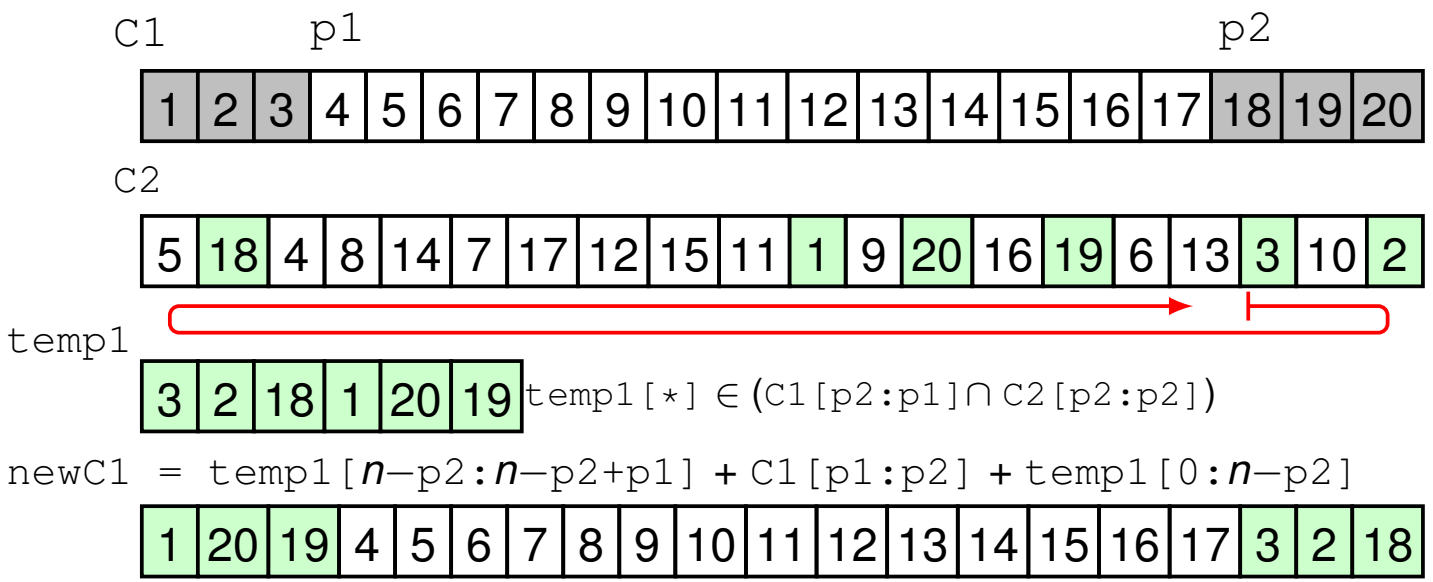
MiddleX

MiddleCrossOver(C1, C2)



OX

OrderCrossOver(C1, C2)

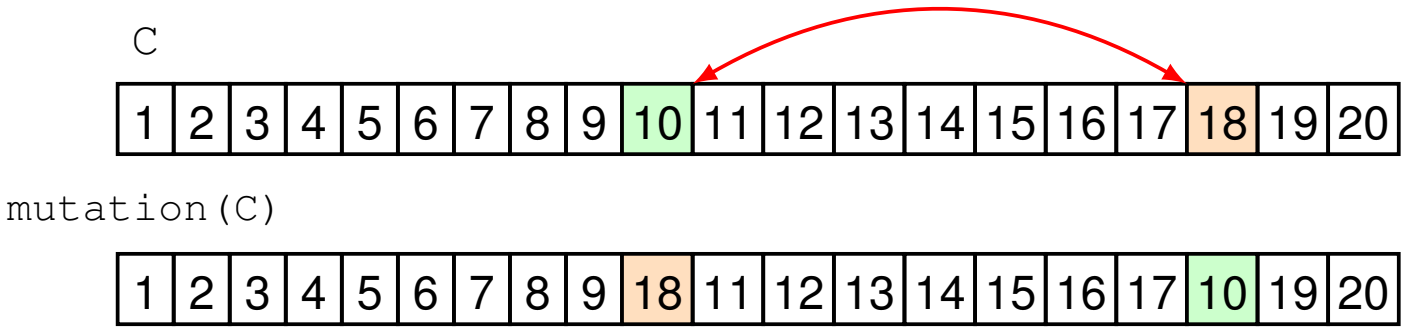


Mutation

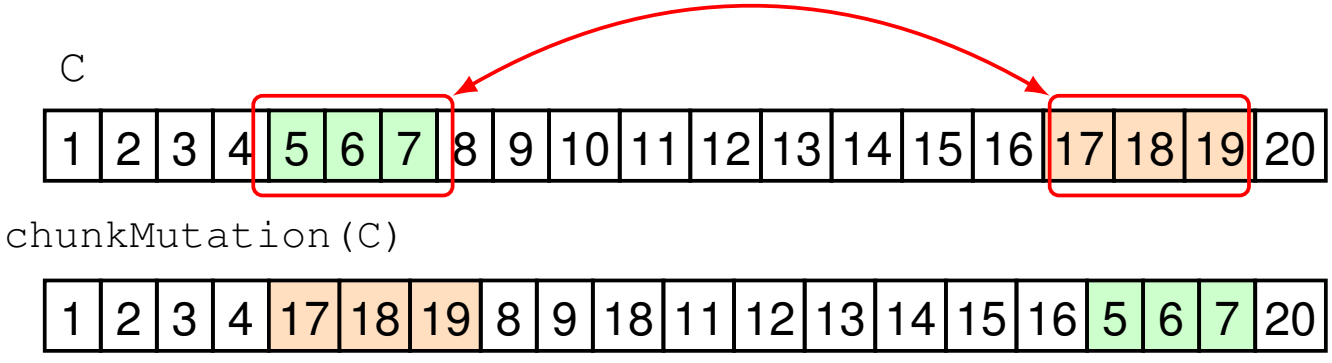
Mutation

Mutation

mutation(C)



chunkMutation(C)



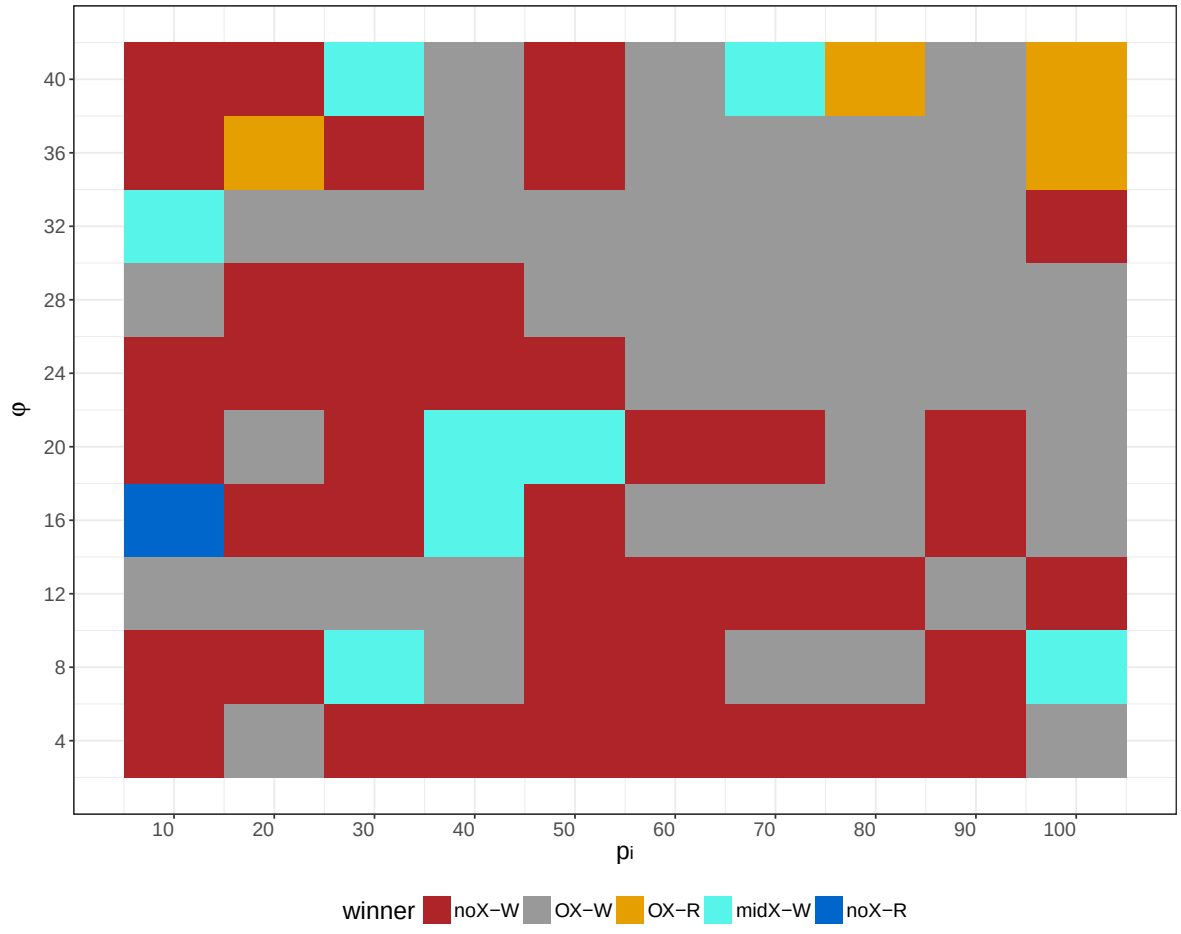
ES parameters

- Population size = 60
- Stopping point : 50 iterations with no improvement
- Wheel selection:
  - Elitism: 10/60
  - Mutation: 15/60
  - Chunk-Mutation: 15/60
  - Crossover: 20/60

4 good seeds

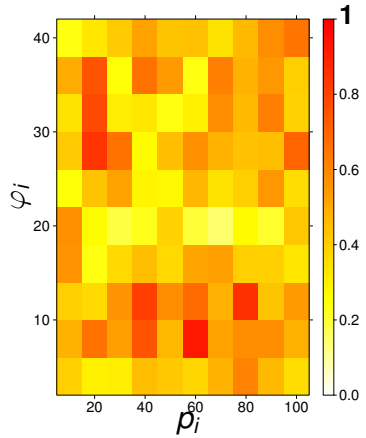
- LPT (Largest Processing Time)
- LPN (Largest Power Need)
- LPTPN (Largest Processing Time Power Need)
- 2Qs (Two Queues)

Experimental results: PerMAK

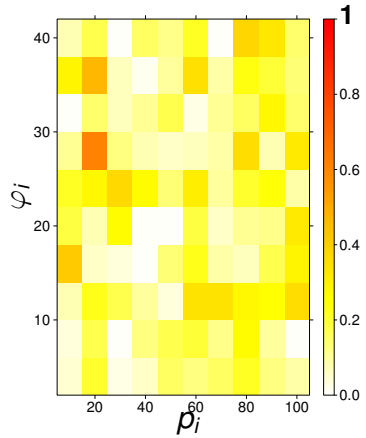


pmax ranging from 10 to 100 and φmax ranging from 4 to 40

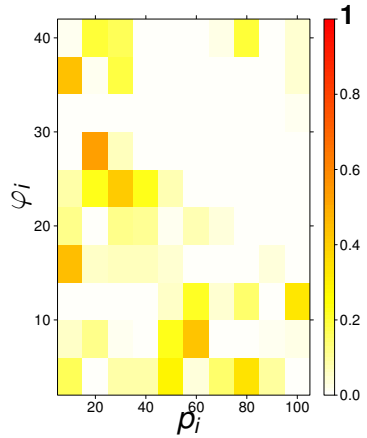
Distance from the best



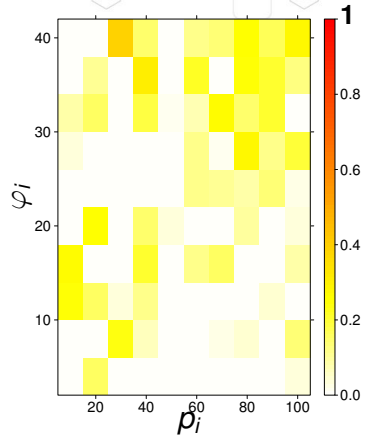
(a) 1pX-W



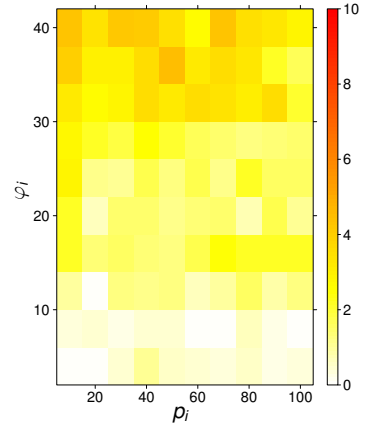
(b) MiddleX-W



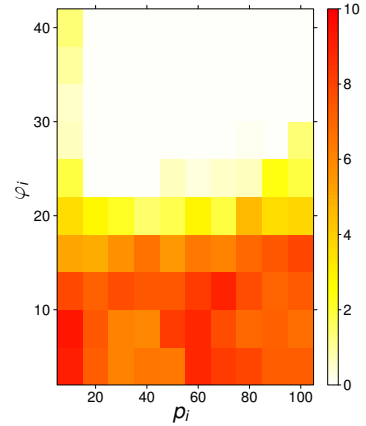
(c) OX-W



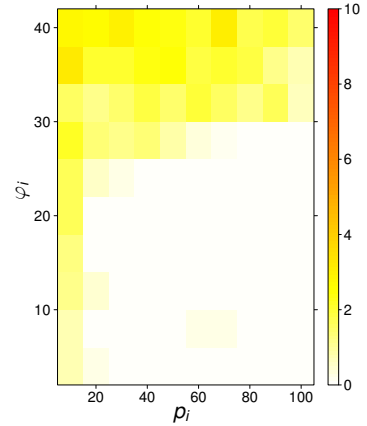
(d) NoX-W



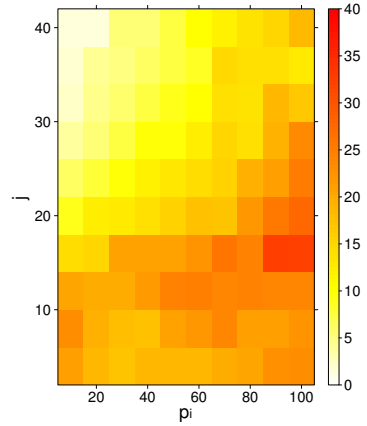
(e) LPT



(f) LPTPN



(g) 2Qs



(h) LPN

Summary

- HPC in green data centers
- Different crossovers
- Wheel selection vs Random selection
- Extensive simulations

Conclusion

- Evolution strategies outperform list algorithms
- Much higher computation time
- Evolution strategies are closer to the optimal still within acceptable time limits
- Wheel selection

Next Step

In Progress

- Realistic power envelopes
- Multi machines
- Distance from optimal {TFI & IFT}

Future Work

- Real workloads
- Flowtime for ES
- Energy storage
- Taking uncertainty on φ_i and p_i into account