An Efficient Algorithm for Scheduling Jobs in Volunteer Computing platforms

A.Essafi, D.Trystram, Z.Zaidi

University of Tunis, University Grenoble-Alpes, University of Tunis,

March 31 - April 4, 2014

Centre CNRS "Paul-Langevin", Aussois, France

Abstract

Desktop Grid

Internet : a reserve of underexploited resources

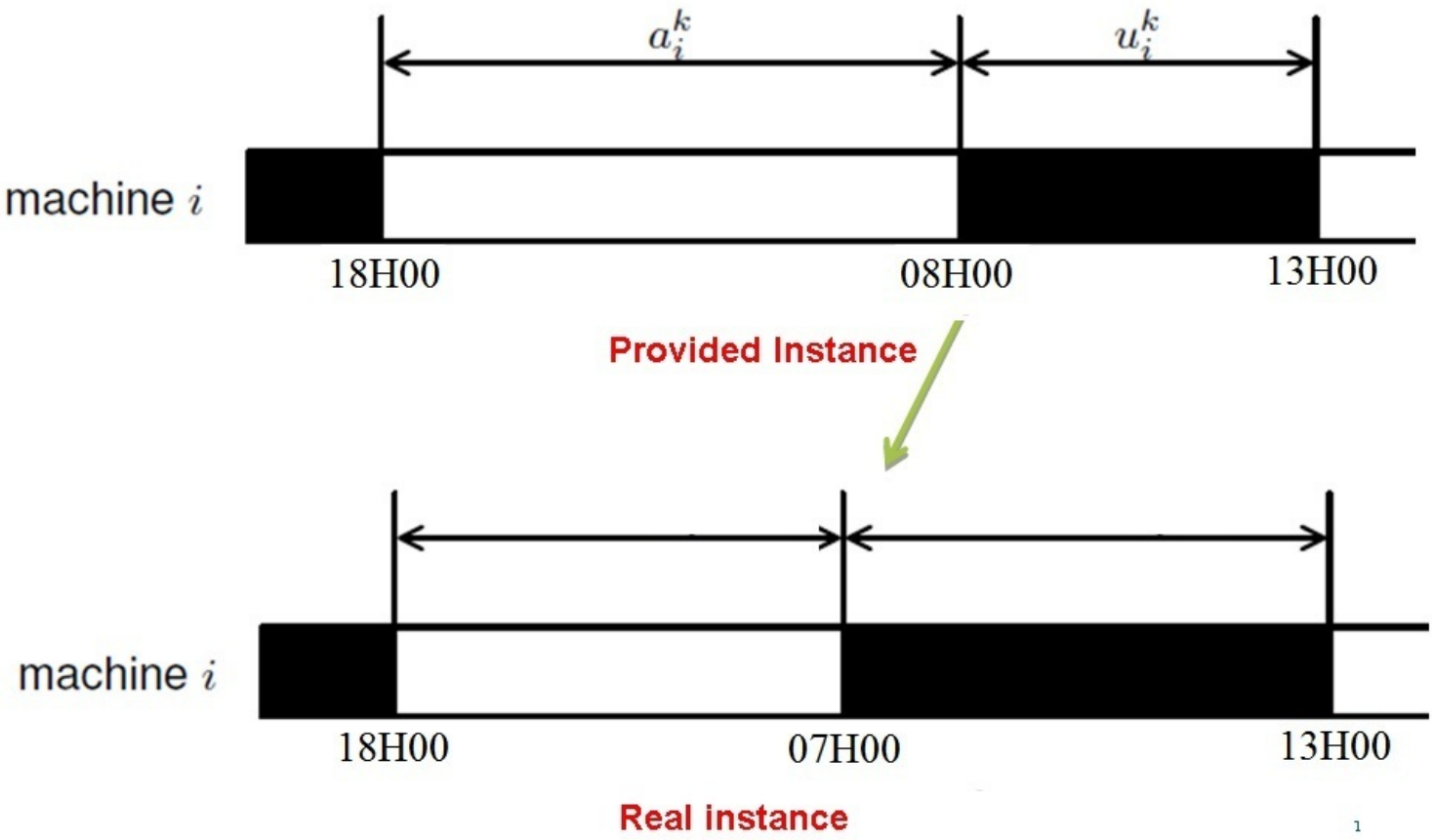
Volunteering :

- Many actors
- Acceptable performance (technological evolution)
- Latent instability
- Short lived

Collaboration :

- Long-run (Institutional)
- More or less – stable performance
- Not always evident to place
- Conflict of interest

Uncertain Platform

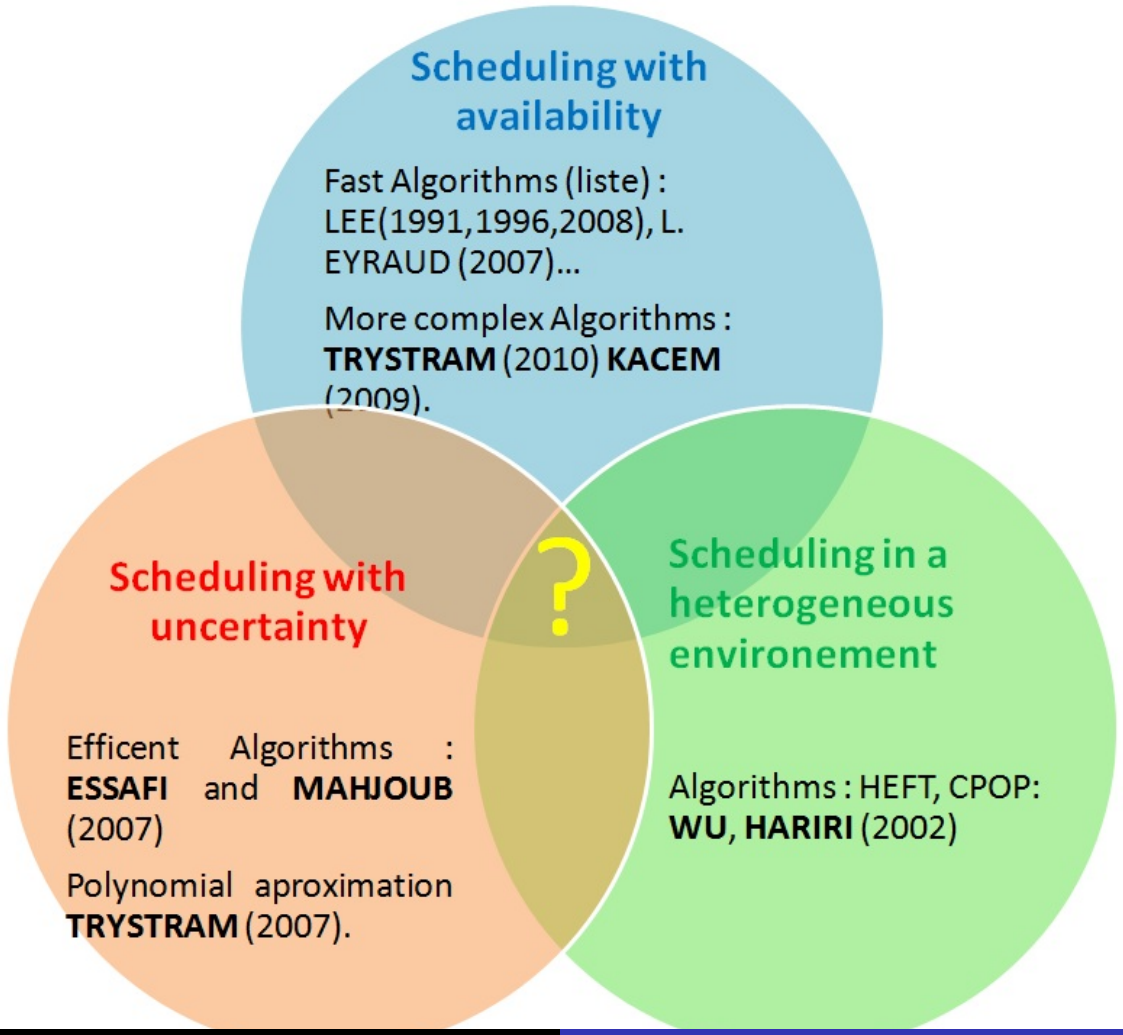


Definitions of the Problem

Tasks, machines and objective :

- Tasks : n non preemptive independent tasks of size p_j
- Machines
  - m machines of speed of S_i
  - Machine i is only available during giving intevals
- Objective
  - Makespan (Taken uncertainty into account)

Related work



HEFT Principle

HEFT principle for DAG

Phase 1

- Calculation of the priority of each task ($rank_u$), which is based on the average calculation and communication costs.
- The task list is generated by sorting the tasks in decreasing order of $rank_u$.

Phase 2

- For most scheduling algorithms, the availability date for a processor p_j is the end of execution of its last task assigned.
- Insertion policy
- Possibility of insertion of a task in an interval of inactivity

HEFT Principle

HEFT principle for DAG

Phase 1

- Calculation of the priority of each task ($rank_u$), which is based on the average calculation and communication costs.
- The task list is generated by sorting the tasks in decreasing order of $rank_u$.

Phase 2

- For most scheduling algorithms, the availability date for a processor p_j is the end of execution of its last task assigned.
- Insertion policy
- Possibility of insertion of a task in an interval of inactivity

HEFT adaptation to availability constraint

- $EST(i,j) = avail(j,i)$: First date in wich j can execute task i
- $EFT(i,j) = EST(i,j) + \frac{P_i}{S_j}$:End date of i task if executed on processor j
- $rank_u(i) = \overline{P_i}$: Priority of the i task

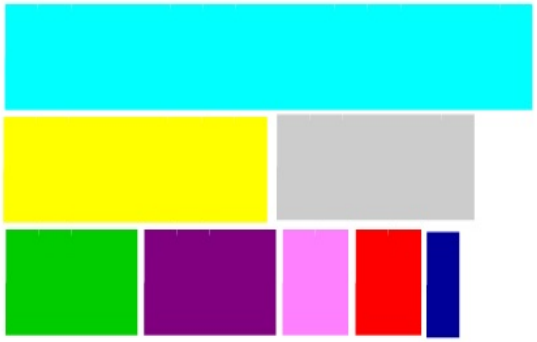
HEFT adaptation to availability constraint

- $EST(i,j) = avail(j,i)$: First date in wich j can execute task i
- $EFT(i,j) = EST(i,j) + \frac{P_i}{S_j}$:End date of i task if executed on processor j
- $rank_u(i) = \overline{P_i}$: Priority of the i task

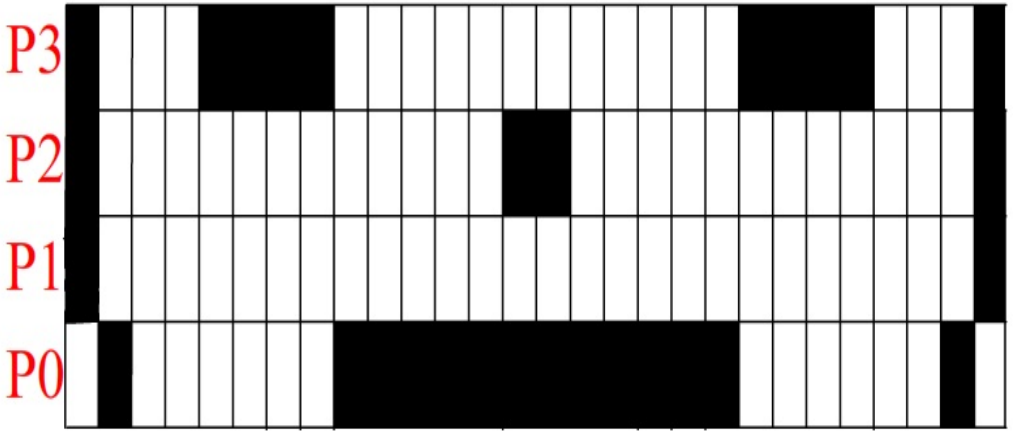
HEFT-A2 Algorithm

1. Compute the $Rank_u$ and the average cost of processing for all tasks
2. Sort all tasks in order of decreasing values of $Rank_u$.
3. **While** there are unscheduling tasks in the list **do**
5. Select the first task, n_i, from the list for scheduling
6. **For** each processor p_k in the processor-set ($p_k \in Q$) **do**
7. Compute the availablity date for n_i (avail[p_k,n_i])
7. Compute EFT (n_i, p_k) value
8. Assign task n_i to the processor p_k that minimize EFT of task n_i.
9. **endwhile**

Exemple for Tasks allocation in HEFT-A2



V0=V1=V2=V3=10



What's wrong with HEFT-A2 ?

Risks

- Can fill an interval availability entirely
- Uncertainty unawareness ?

Improvement

- We must improve the allocation of tasks to machines using the availability model.

Stable algorithm for disturbed environments

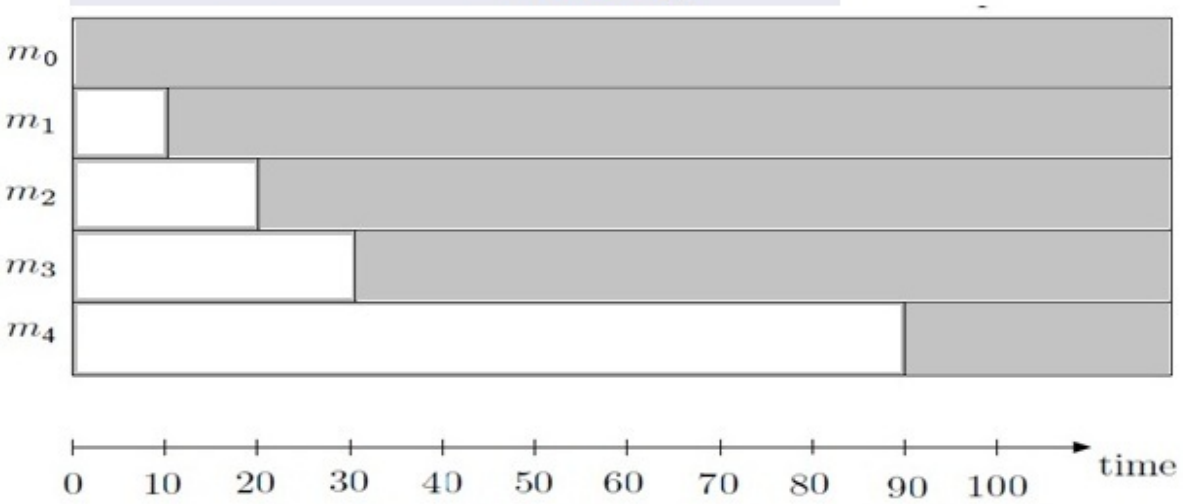
Performance modeling , Kondo et al.

- Identifying correlation between resources based on their availability
- Standard classification algorithm (K-means)
- They were able to identify 5 classes of machines

The average availability of a machine in a grid is a good criterion for the classification

Alpha

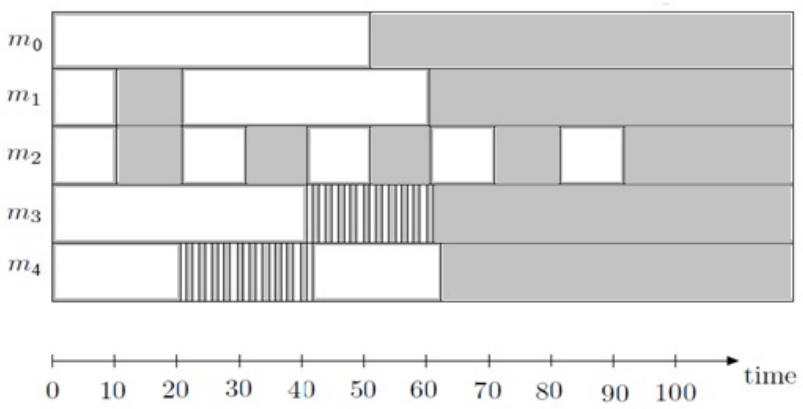
$$alpha[j] = \frac{\sum_{k=1}^{nb_{intervalles[j]}} a_j^k * vitesse[j]}{nb_{intervalles[j]}}$$



Is the average amount of computation in one interval

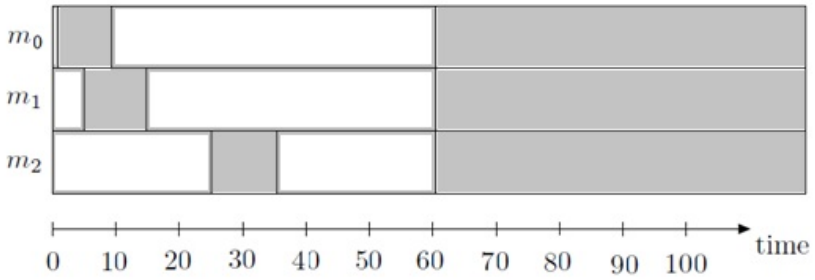
alpha[0]= 0
alpha[1]= 10
alpha[2]= 20
alpha[3]= 30
alpha[4]=90

Drawback whith Alpha



alpha[0]= 50
alpha[1]= 25
alpha[2]= 10
alpha[3]= 50/11 = 4.54
alpha[4]= 50/12 = 4.16

The alpha parameter does not take into account the dispersion pattern of intervals availability.



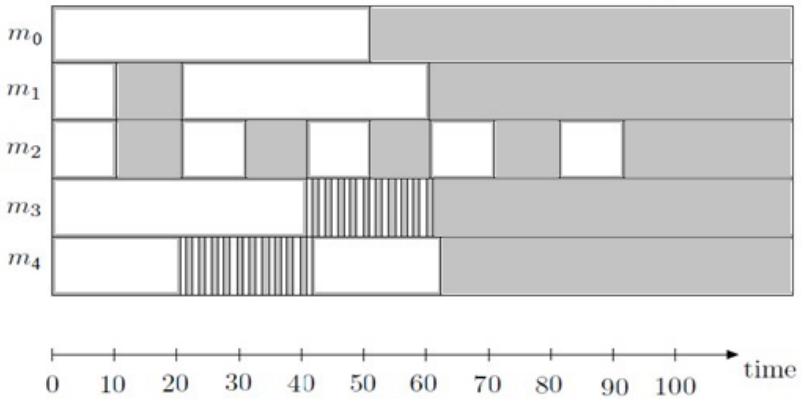
alpha[0]= 25
alpha[1]= 25
alpha[2]= 25

The alpha parameter is insufficient to characterize machines

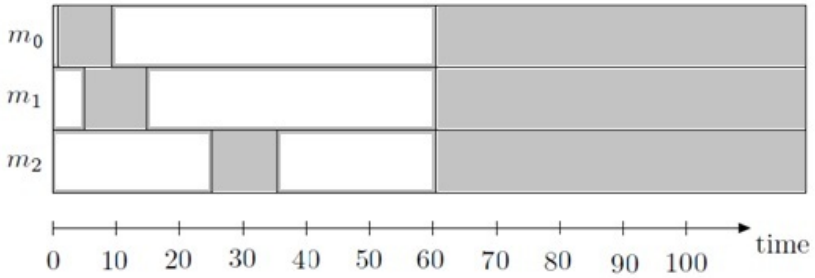
Beta

$$\text{beta}[j]=\sum_{k=0}^{nb\_intervalles[j]}\left(a_j^k-\bar{l}[j]\right)^2$$

Is the variance between availability intervals of machine j



Valeur de alpha	Valeur de beta
alpha[0]= 50	beta[0]=0
alpha[1]= 25	beta[1]=450
alpha[2]= 10	beta[2]=0
alpha[3]= 50/11 = 4.54	beta[3]=138.27
alpha[4]= 50/12 = 4.16	beta[4]=54.69

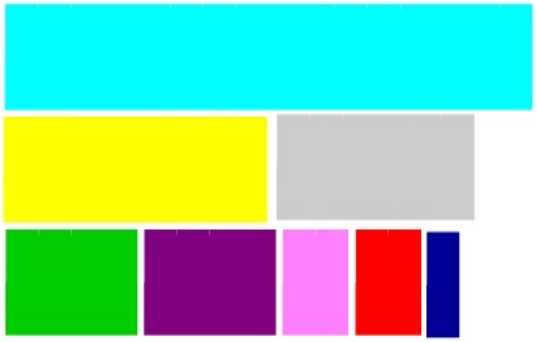


Valeur de alpha	Valeur de beta
alpha[0]= 25	beta[0]=1152
alpha[1]= 25	beta[1]=1058
alpha[2]=25	beta[2]=0

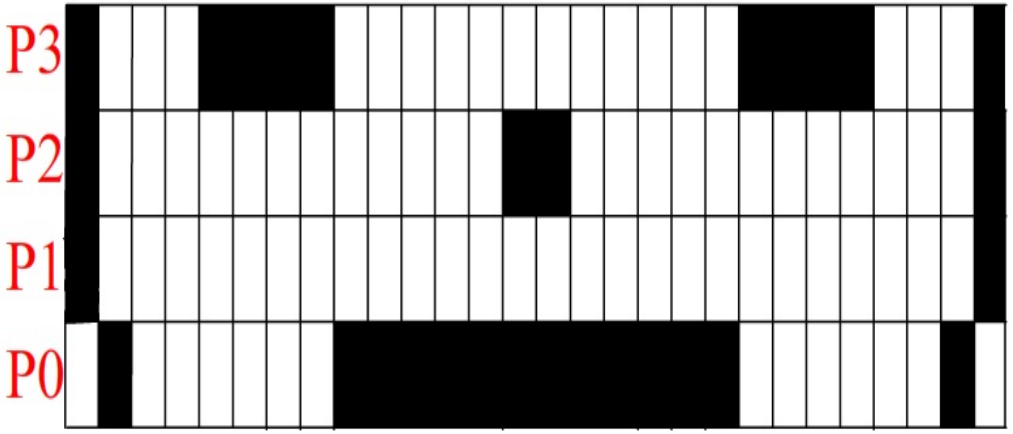
HEFT-ASA Algorithm

1. Compute the $Rank_u$ for all tasks
2. Compute alpha[j] and beta[j] for all processors.
3. Sort all tasks in order of decreasing values of $Rank_u$. (Longest First)
6. **While** there are unscheduled tasks in the list **do**
7. Select the first task, ni from the list
8. Compute $EFT^*(n_i) = Min_{j \in 1..m}(EFT(n_i, p_j))$
9. Let P_{cand} the list of processor p_j that $EFT(n_i, p_j) \leq EFT^*(n_i,) * (1+\tau)$
10. Assign task n_i to processor p_j from P_{cand} such as alpha[j] /beta[j] is the maximum
11. **endwhile**

Exemple for Tasks allocation in HEFT-ASA



V0=V1=V2=V3=10



Execution Scenario

Scenario 1 : (without disturbance) Tasks are executed using the planned dates.

Scenario 2 : (with disturbance) Potential interruptions of tasks are handled by a local re scheduling mechanism on the same processor.

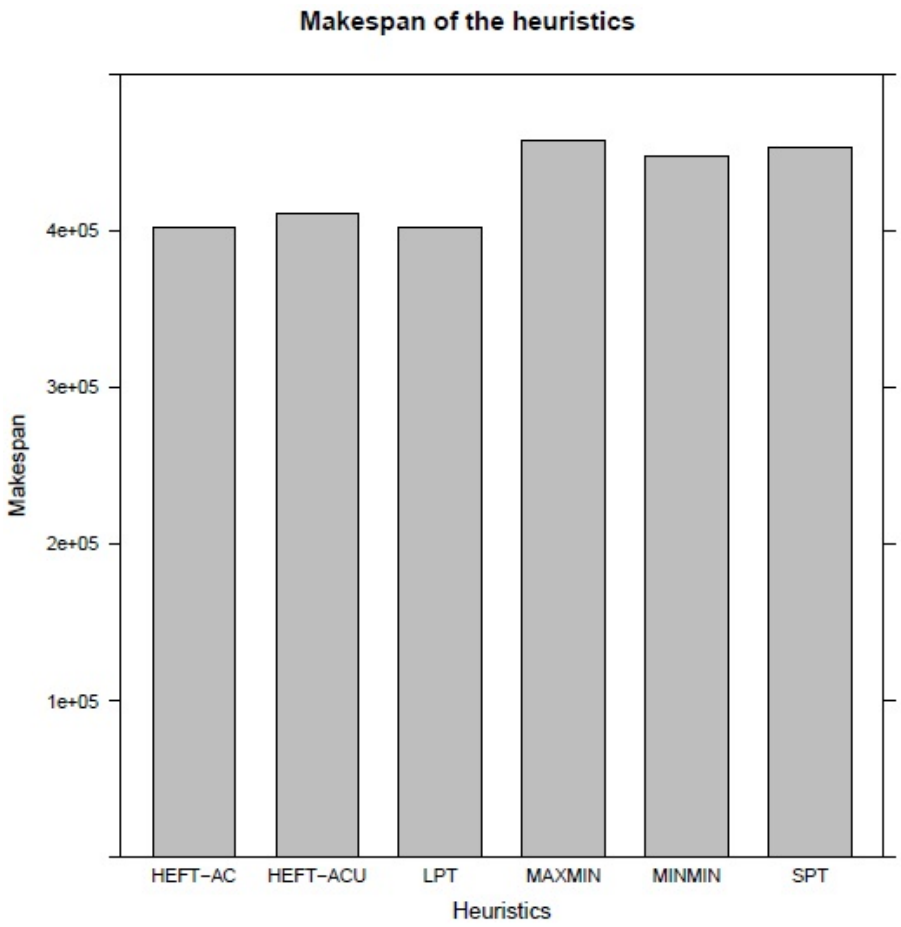
To do that we designed a specific simulator which supports machine profiles, task profile and disturbance in availability intervals

Study of performance of heuristics

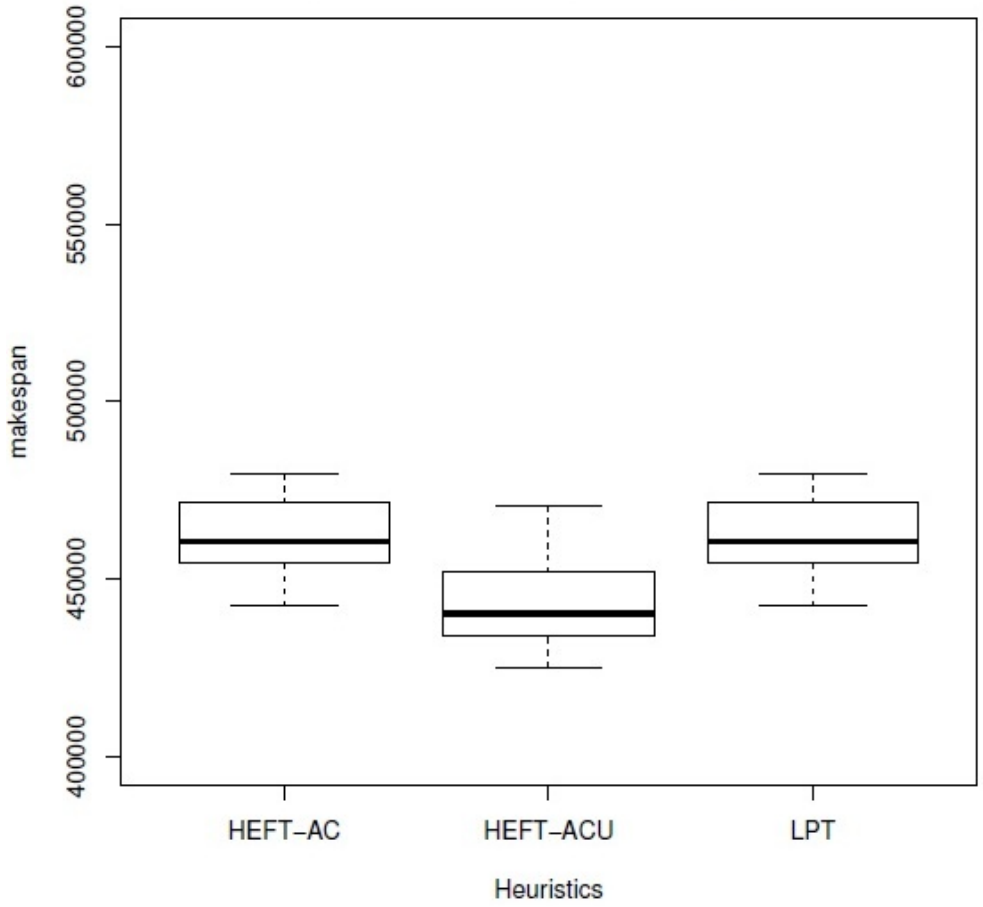
- Six algorithms are studied :
  - LPT
  - SPT
  - MinMin
  - MaxMin
  - HEFT-A2
  - HEFT-ASA

- Test are performed on 10 different instances
  - 10000 tasks
  - 1000 machines
  - $\tau = 0.2$ (empiricaly chosen)
  - Each instance is disturbed 30 times

Performance Comparison between Heuristics without disturbance



Performance Comparison between Heuristics with disturbance



Conclusion and prospect

Conclusion

- We are considering uncertain Desktop grid platforms
- We adapted HEFT to schedule tasks within the schedule instead of only at the end
- We use the average and variance of the length of availability intervals to characterise the most stable machines
- HEFT-ASA is the most stable evaluated algorithm

- Future works

Migration, Duplication...

Thank you for your attention

Thank you for your attention