Ordonnancement avec Oracle

Algorithmes et Expérimentations

Fanny Dufossé[1]
Vincent Fagnon[1]
Malin Rau[2]
Denis Trystram[1]

{1} Univ. Grenoble Alpes, CNRS, INRIA, Grenoble INP, LIG, Grenoble, France

{2} Universität Hamburg, Germany

Groupe SCALE

Sommaire

Description du problème

On s'interesse à un problème offline avec :

- Des tâches courtes (n_s) de durée p_s
- Des tâches longues (n_l) de durée $p_l = p_s + \Delta$
- Pour lequel on cherche à minimiser ($\sum_{\forall j \in \mathcal{T}}$ Flow-time$_j$)
- On connait le nombre de tâches longues et le nombre de tâches courtes
- A priori, on ne sait pas si une tâche est courte ou longue

Sommaire

État de l'art

- SPT optimal : Richard W Conway, Louis W Miller, and William L Maxwell. Theory of scheduling. Dover, 2003.

- meilleur cas : $\sum_{\mathcal{T}} F_j = \frac{p_s}{2} n(n+1) + \frac{n_L+1}{2} n_L \Delta = \text{OPT}$
- pire cas : $\sum_{\mathcal{T}} F_j = \text{OPT} \boxed{+ \Delta n_L n_S}$
- cas moyen:

$\mathbb{E}_{\forall \text{permutation}}(\sum_{\mathcal{T}} F_j) = \text{OPT} \boxed{+ \frac{\Delta n_L n_S}{2}}$

État de l'art

- SPT optimal : Richard W Conway, Louis W Miller, and William L Maxwell. Theory of scheduling. Dover, 2003.
- Avec preemption : Rajeev Motwani, Steven Phillips, and Eric Torng. Nonclairvoyant scheduling. Theoretical computer science, 130(1):17–47, 1994.

État de l'art

- SPT optimal : Richard W Conway, Louis W Miller, and William L Maxwell. Theory of scheduling. Dover, 2003.
- Avec preemption : Rajeev Motwani, Steven Phillips, and Eric Torng. Nonclairvoyant scheduling. Theoretical computer science, 130(1):17–47, 1994.
- Avec Oracle : Fanny Dufossé, Christoph Dürr, Noël Nadal, Denis Trystram, and Oscar C Vasquez. Scheduling with a processing time oracle. 2020

Notion d'Oracle

On peut faire appel à un oracle qui nous dit si la tâche est longue ou courte

- cout d'appel p_T
- cout d'entrainement (dont on ne parlera pas ici)
- l'oracle a toujours raison (100% fiabilité)

Sommaire

Ne rien tester

- meilleur cas : $\sum_{\mathcal{T}} F_j = \frac{p_s}{2} n(n+1) + \frac{n_L+1}{2} n_L \Delta = \mathrm{OPT}$
- pire cas : $\sum_{\mathcal{T}} F_j = \mathrm{OPT} \boxed{+ \Delta n_L n_S}$
- cas moyen:

$$\mathbb{E}_{\forall \text{permutation}}\left(\sum_{\mathcal{T}} F_j\right) = \mathrm{OPT} \boxed{+ \frac{\Delta n_L n_S}{2}}$$

Tout tester

- meilleur cas : OPT $\boxed{+\ p_T \frac{2n^2+n_s^2-2nn_s+n_s}{2}}$
- pire cas : OPT $\boxed{+\ p_T \frac{2n^2-n_s^2+n_s}{2}}$
- cas moyen:

$$\mathbb{E}_{\forall \text{permutation}}\left(\sum_{\mathcal{T}} F_j\right) = \text{OPT} \boxed{+p_T \frac{2n^2-nn_s+n_s}{2}}$$

Tester les n_T premieres tâches

Se tromper au début fait plus mal que se tromper à la fin.

- meilleur cas : S*L*

- pire cas : L*S*L*S*

- cas moyen (avec x tests) :

$\mathbb{E}(C(n_T)) = \mathrm{OPT}$

$$\boxed{+p_T \cdot n_T\left(n - \frac{n_s}{2n}(n_T - 1)\right) + \frac{n_l \cdot n_s \cdot (n-n_T)}{n(n-1)} \Delta \frac{n-n_T-1}{2}}$$

Si $n_L n_S / n^2 < p_T / \Delta$: $n_T = 0$

Sinon $n_T = \left\lfloor \frac{n \cdot n_S n_L \Delta - (n-1)(n^2+n_S)p_T)}{n_S(n_L\Delta - (n-1)p_T)} \right\rfloor$

Adaptatif

Parfois on a peu de chances de se tromper au début, et plus de chance de se tromper à la fin.

	i $\in \mathcal{S}$	i $\in \mathcal{L}$
Probability	$\frac{\bar{n}_s}{\bar{n}_s+\bar{n}_l-\bar{q}}$	$\frac{\bar{n}_l-\bar{q}}{\bar{n}_s+\bar{n}_l-\bar{q}}$
if i is tested	$SC = p_T * (\bar{n}_s + \bar{n}_l)$	
if not is tested	$SC = 0$	$SC = \Delta \bar{n}_s$

Donc si $(\frac{p_T}{\Delta} < \frac{\bar{n}_l-\bar{q}}{\bar{n}_s+\bar{n}_l-\bar{q}} \cdot \frac{\bar{n}_s}{(\bar{n}_s+\bar{n}_l))})$, on teste la tâche courante

Adaptatif

$SC[\bar{n}_l, \bar{n}_s, \bar{q}]$

test i

$\mathbb{P}_{i\in\mathcal{S}}=\frac{\bar{n}_s}{\bar{n}_s+\bar{n}_l}$ $\quad p_T(\bar{n}_s + \bar{n}_l) + \mathrm{SC}[\bar{n}_l,\bar{n}_s\text{-}1,\bar{q}]$

$\mathbb{P}_{i\in\mathcal{L}}=\frac{\bar{n}_l-\bar{q}}{\bar{n}_s+\bar{n}_l}$ $\quad p_T(\bar{n}_s + \bar{n}_l) + \mathrm{SC}[\bar{n}_l\text{-}1,\bar{n}_s,\bar{q}+1]$

don't test i

$\mathbb{P}_{i\in\mathcal{S}}=\frac{\bar{n}_s}{\bar{n}_s+\bar{n}_l}$ $\quad 0 + \mathrm{SC}[\bar{n}_l,\bar{n}_s\text{-}1,\bar{q}]$

$\mathbb{P}_{i\in\mathcal{L}}=\frac{\bar{n}_l-\bar{q}}{\bar{n}_s+\bar{n}_l}$ $\quad \Delta\bar{n}_s + \mathrm{SC}[\bar{n}_l\text{-}1,\bar{n}_s,\bar{q}]$

$\forall q$: SC[0,0,q]=0 $\quad\forall i,j$: SC[i,0,j]=0 $\quad\forall i,j$: SC[0,i,j]=0

Optimum

$SC[\bar{n}_l, \bar{n}_s, \bar{q}]$

- test i
  - $\mathbb{P}_{i\in\mathcal{S}}=\frac{\bar{n}_s}{\bar{n}_s+\bar{n}_l}$: $p_T(\bar{n}_s + \bar{n}_l) + \text{SC}[\bar{n}_l,\bar{n}_s\text{-}1,\bar{q}]$
  - $\mathbb{P}_{i\in\mathcal{L}}=\frac{\bar{n}_l-\bar{q}}{\bar{n}_s+\bar{n}_l}$: $p_T(\bar{n}_s + \bar{n}_l) + \text{SC}[\bar{n}_l\text{-}1,\bar{n}_s,\bar{q}+1]$
- don't test i
  - $\mathbb{P}_{i\in\mathcal{S}}=\frac{\bar{n}_s}{\bar{n}_s+\bar{n}_l}$: $0 + \text{SC}[\bar{n}_l,\bar{n}_s\text{-}1,\bar{q}]$
  - $\mathbb{P}_{i\in\mathcal{L}}=\frac{\bar{n}_l-\bar{q}}{\bar{n}_s+\bar{n}_l}$: $\Delta\bar{n}_s + \text{SC}[\bar{n}_l\text{-}1,\bar{n}_s,\bar{q}]$

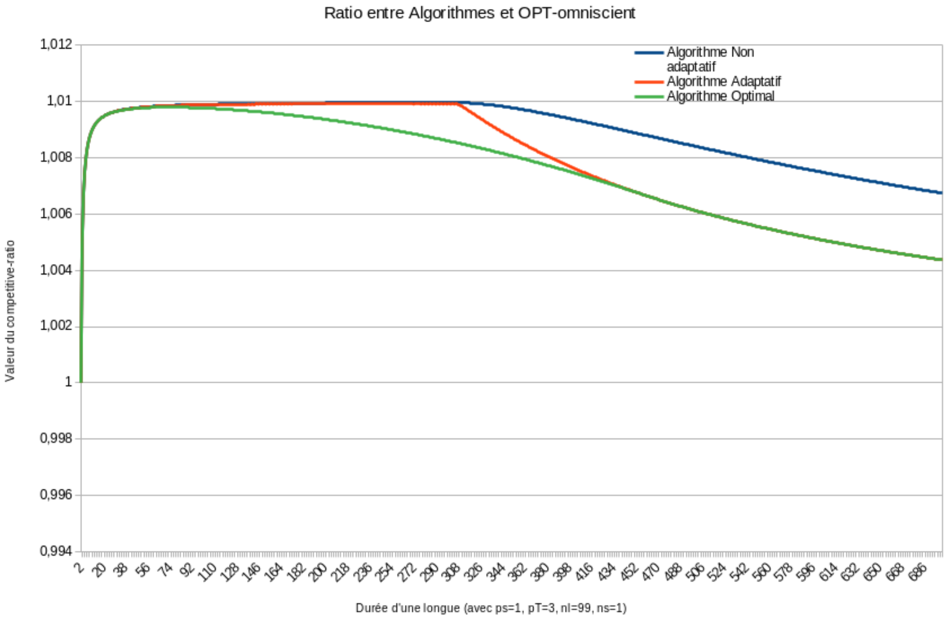
Sommaire

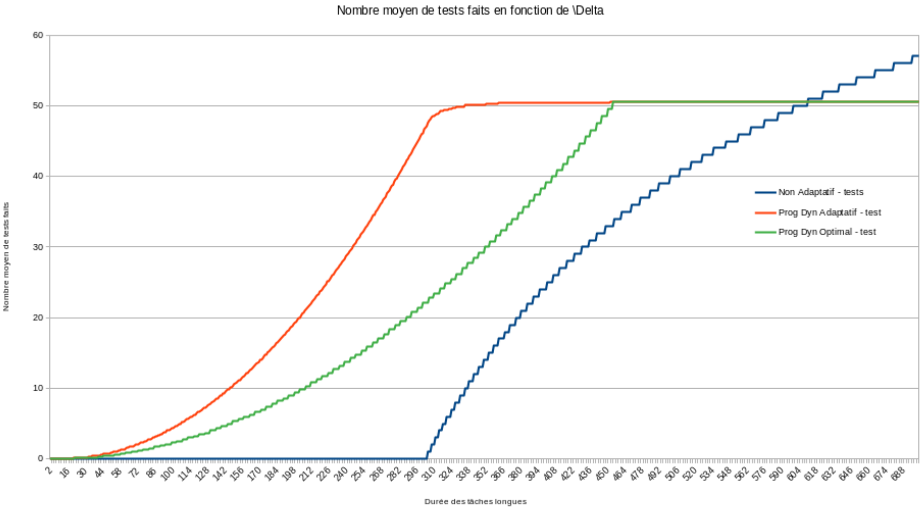
Expérimentations

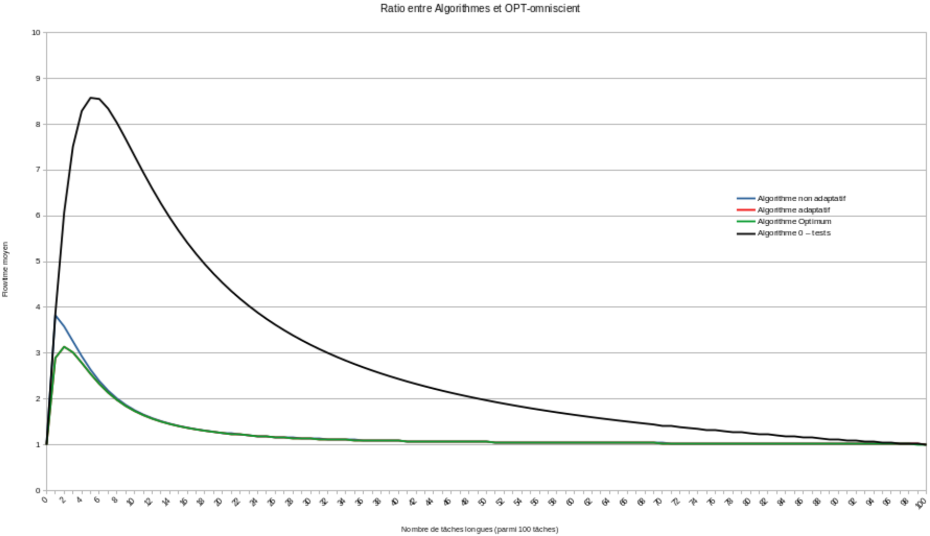
On fixe :

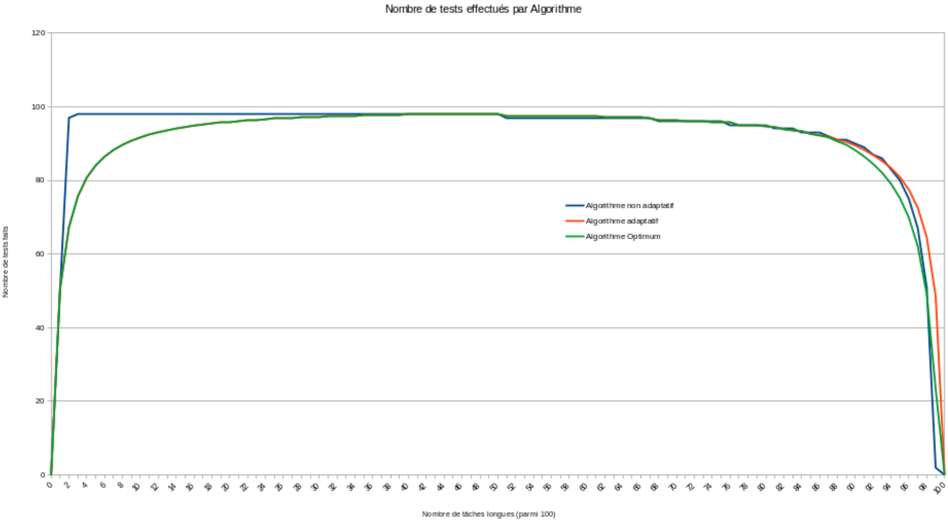
- $p_S = 1$
- $p_T = 3$
- $n = 10$

Puis on fait varier Δ, ainsi que la proportion Longues / Courtes


Ratio entre Algorithmes et OPT-omniscient
Algorithme Non adaptatif
Algorithme Adaptatif
Algorithme Optimal
Valeur du competitive-ratio
1,012
1,01
1,008
1,006
1,004
1,002
1
0,998
0,996
0,994
2 20 38 56 74 92 110 128 146 164 182 200 218 236 254 272 290 308 326 344 362 380 398 416 434 452 470 488 506 524 542 560 578 596 614 632 650 668 686
Durée d'une longue (avec ps=1, pT=3, nl=99, ns=1)


Nombre moyen de tests faits en fonction de \Delta
Nombre moyen de tests faits
Durée des tâches longues
Non Adaptatif - tests
Prog Dyn Adaptatif - test
Prog Dyn Optimal - test
0
10
20
30
40
50
60


Ratio entre Algorithmes et OPT-omniscient
Flowtime moyen
Nombre de tâches longues (parmi 100 tâches)
Algorithme non adaptatif
Algorithme adaptatif
Algorithme Optimum
Algorithme 0 – tests


Nombre de tests effectués par Algorithme
Algorithme non adaptatif
Algorithme adaptatif
Algorithme Optimum
Nombre de tests faits
Nombre de tâches longues (parmi 100)
0
20
40
60
80
100
120

Conclusion

- Algorithme Optimum Glouton
- Des contraintes supplémentaires
- Robustesse des algorithmes
- L'Oracle