Single Machine Scheduling with Solar Panels and Batteries

V. Fagnon[1], I. Kacem[2], G. Lucarelli[2], J.-M. Nicod[1], V. Sonigo[1]

[1]Institut FEMTO-ST, CNRS, UMLP, SUPMICROTECH, Besançon

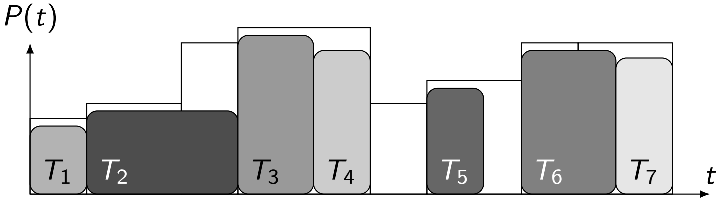
[2]LCOMS, Univ. Lorraine, Metz

Scheduling Workshop 2026 – Fréjus

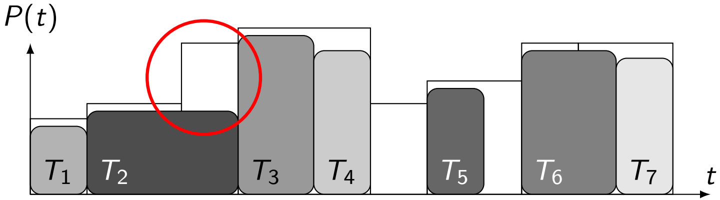
Motivation

- HPC carbon impact
- Using renewable energy is more sustainable
- Scheduling policy depending on energy intermittency

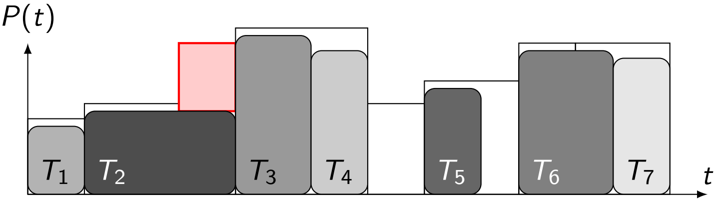
Motivating example (using a multiprocessor platform)



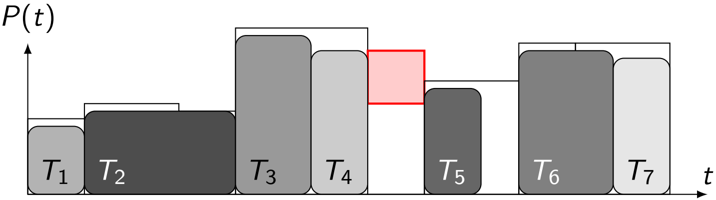
Motivating example (using a multiprocessor platform)



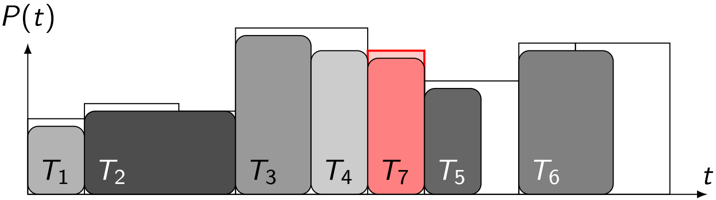
Motivating example (using a multiprocessor platform)



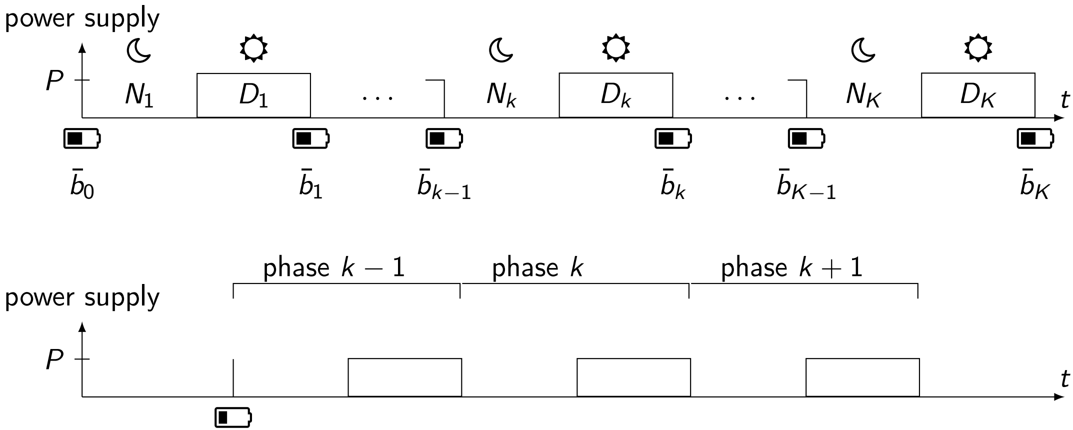
Motivating example (using a multiprocessor platform)



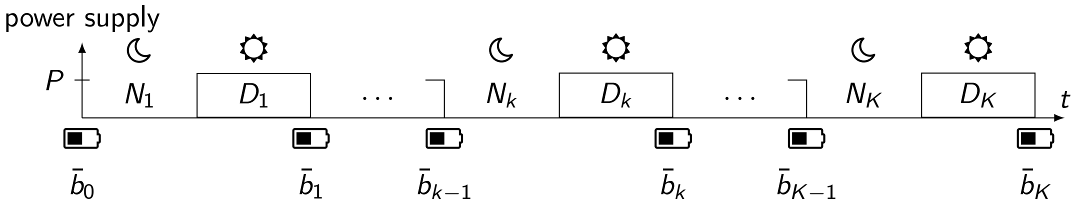
Motivating example (using a multiprocessor platform)

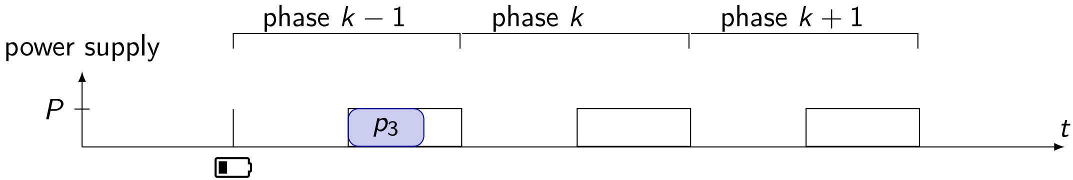


Framework considering one machine with one battery

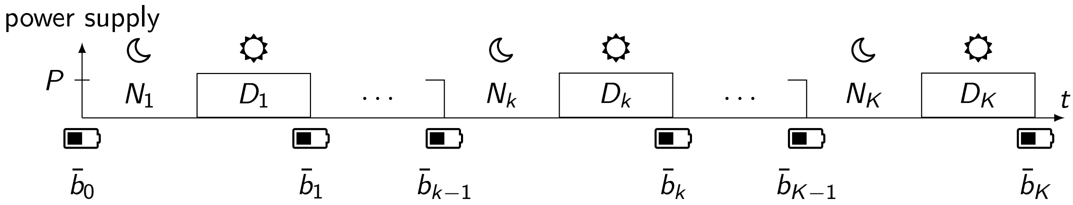


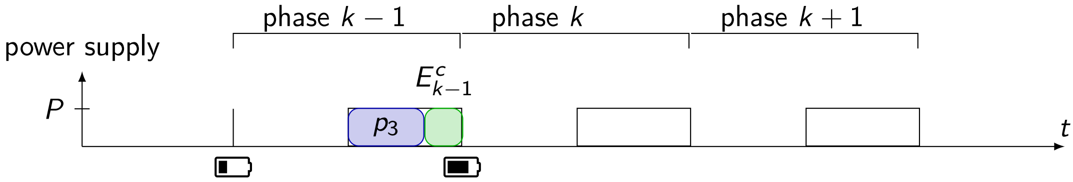
Framework considering one machine with one battery





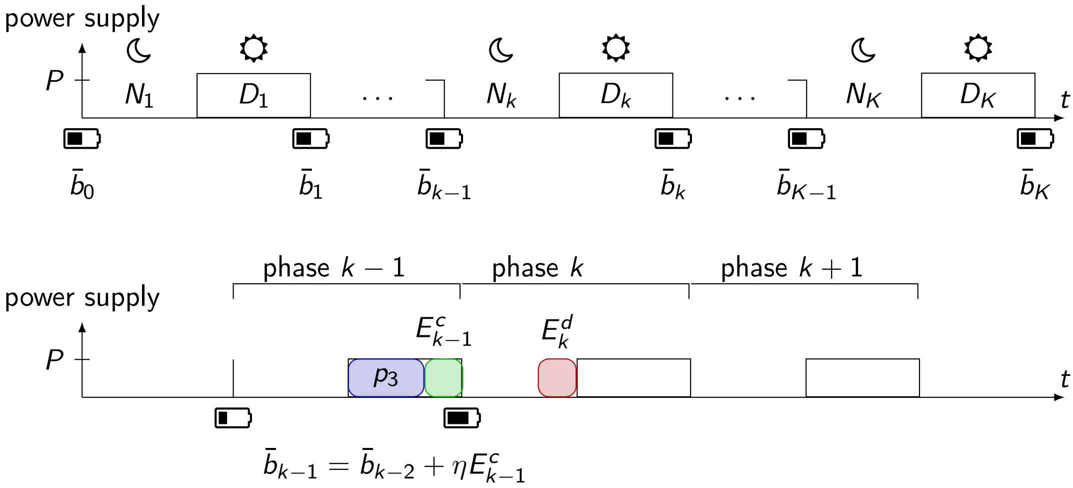
Framework considering one machine with one battery



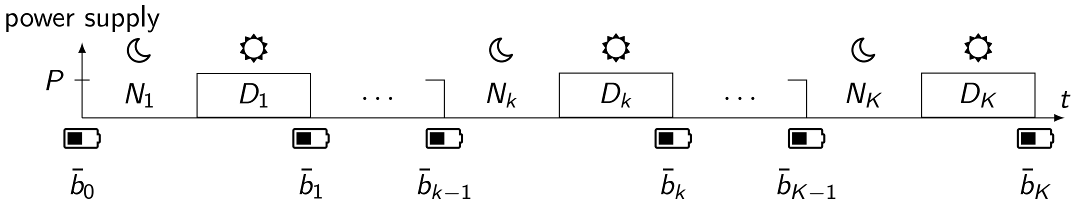


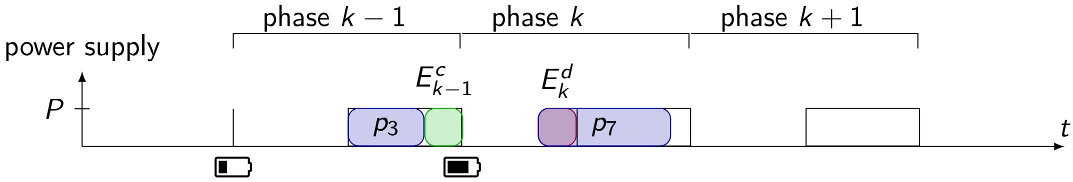
$$\bar{b}_{k-1} = \bar{b}_{k-2} + \eta E^c_{k-1}$$

Framework considering one machine with one battery



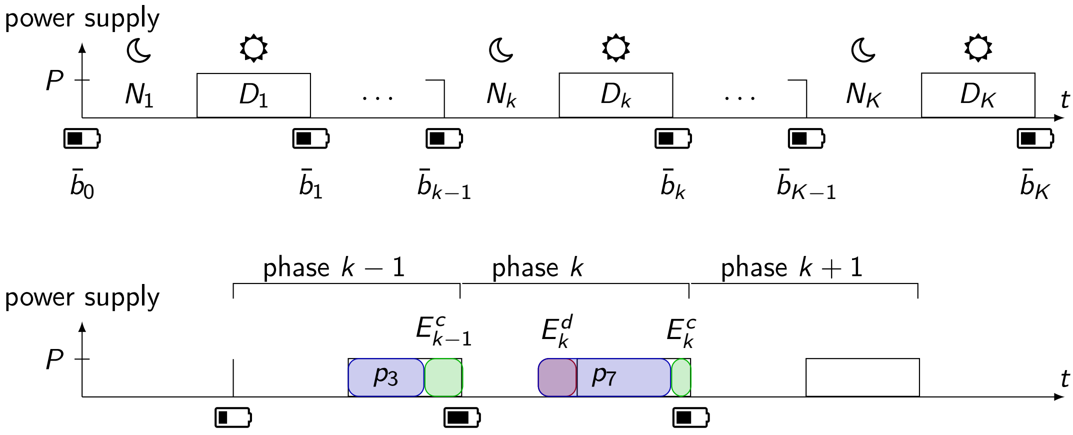
Framework considering one machine with one battery





$$\bar{b}_{k-1} = \bar{b}_{k-2} + \eta E^c_{k-1}$$

Framework considering one machine with one battery



$$\bar{b}_{k-1} = \bar{b}_{k-2} + \eta E^c_{k-1} \qquad \bar{b}_k = \bar{b}_{k-1} + \eta E^c_k - \frac{E^d_k}{\eta}$$

Framework

- $\mathcal{T}$ the set of n non preemptive tasks j and p_j its processing time
- The time horizon composed by 2 alternative periods, day and night, with and without solar energy:
  - D_k the duration of day k
  - N_k the duration of night k
  - P the available energy during D_k, needed to run the machine
- Phase k the k^{th} coupled period, night N_k and day D_k

Framework and objective

- B (Wh) the capacity of the battery ($B \geq P \times \max(p_j)$)
  - $\eta_c P \times t$ the energy charged within the battery during a period t during the day
  - $\eta_d P \times t$ the energy discharged during a period t during the night
  - $\eta_c = \eta_d = \eta \leq 1$ the efficiency to charge or discharge the battery and $(1 - \eta)P \times t$ being the energy loss by each battery usage
  - $\bar{b}_k$ the energy within the battery at the end the k^{th} phase ($\bar{b}_0$ the initial battery level at the beginning)
  - E_k^c and E_k^d the energy charged or discharged during resp. the k^{th} day and night
- Evolution of the energy stored within the battery:

$$\bar{b}_k = \bar{b}_{k-1} + \eta_c \cdot E_k^c - \frac{E_k^d}{\eta_d}.$$

Objective

- Minimize the number of periods used $\sim C_{max}$

Contribution

- The problem depends on the battery feature
  - Without energy loss
    - $B = 0$: Problem NP hard (trivial from bin packing), $\frac{3}{2}$ approximation algorithm exists in the literature (First-Fit-Decreasing)
    - $B = +\infty$ or $B \geq \max_j(p_j)$: we show that the problem is polynomial (ASAP algorithm, next slide).
    - For any B (even close to 0), we propose a (close to) $\frac{11}{9}$-approximation algorithm (G-FFD)
  - With energy loss
    - We can choose to not use the battery, which means to solve the Bin-Packing problem, FFD gives a $\frac{3}{2}$-approximation.
    - We have a counterexample where, if we use the battery, we get a $\frac{3}{2}$-hardness

Battery without energy loss

ASAP – an optimal greedy algorithm using a large battery

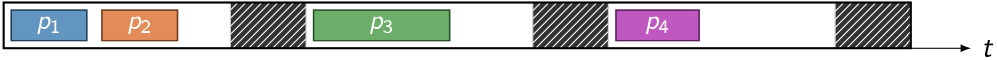
- $B = +\infty$ (unrealistic assumption)
  - Charge enough energy to complete the entire workload without interruption
  - No energy loss

  $\Longrightarrow$ Optimal ($\sim$ to complete the workload with preemption)
- $B \geq \max_j(p_j)$: ASAP algorithm
  - Execute the first day as many tasks as possible and store the unused energy at the end of that day
  - Start the next task as soon as possible before dawn the next day, using all or part of the energy in the battery. At the end of that day, save any unused energy and repeat the process the following days...
  - No energy loss

  $\Longrightarrow$ Optimal ($\sim$ to complete the workload with preemption)

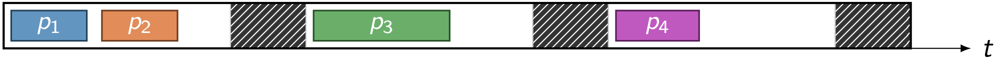
And if the battery is tiny ?

If $B = 0$

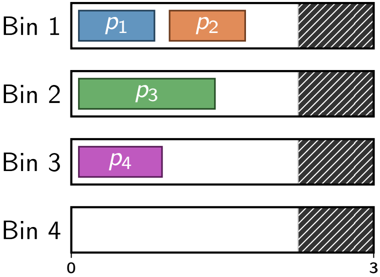


And if the battery is tiny ?

If $B = 0$



$\Longrightarrow$ The problem to solve is a Bin Packing problem



G-FFD (Guess First Fit Decreasing) – Principle

Hypothesis

- $p_j < \min_k D_k$ and B is very small

G-FFD works in two steps:

- Step 1: The Guessing Phase of OPT(bin packing without battery)
  - Guess OPT(Bin Packing)= 1: easy to verify
  - Guess OPT(Bin Packing)= 2: use PTAS to solve $P2||C_{max}$ with $\varepsilon = \frac{B}{(2-1)D}$
  - If previous phase didn't pay, guess OPT(Bin Packing)= 3: use PTAS to solve $P3||C_{max}$ with $\varepsilon = \frac{B}{(3-1)D}$
  - ...
- Step 2: If the guessing step didn't pay, then run FFD without using the battery

G-FFD (Guess First Fit Decreasing) – Principle

Hypothesis

- $p_j < \min_k D_k$ and B is very small

G-FFD works in two steps:

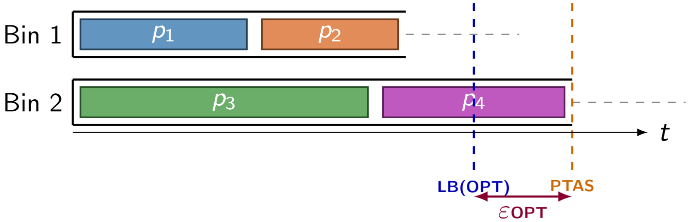
- Step 1: The Guessing Phase of OPT(bin packing without battery)
  - Guess OPT(Bin Packing)$=1$: easy to verify
  - Guess OPT(Bin Packing)$=2$: use PTAS to solve $P2||C_{max}$ with $\varepsilon = \frac{B}{(2-1)D}$
  - If previous phase didn't pay, guess OPT(Bin Packing)$=3$: use PTAS to solve $P3||C_{max}$ with $\varepsilon = \frac{B}{(3-1)D}$
  - ...
- Step 2: If the guessing step didn't pay, then run FFD without using the battery

Natural improvement: start by guessing the number of bins $= \left(\frac{1}{\min_k D_k} \sum_{j \in \mathcal{T}} p_j \right)$

G-FFD Algorithm

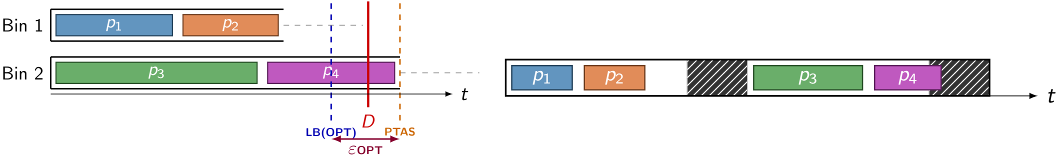
- Guessing $\text{OPT}_{\text{Bin Packing}} = 1$
  - If $P1||C_{max}$ does not exceed one day, a schedule exists with one bin $\Rightarrow$ END
  - Otherwise $\text{OPT}_{\text{Bin Packing}} > 1 \Rightarrow$ next guess
- Guessing $\text{OPT}_{\text{Bin Packing}} = 2$:
  - Solve $P2||Cmax$ using a PTAS with $(1+\varepsilon)\text{OPT}$

We assume that $\varepsilon\text{OPT}(P2||C_{max}) \leq B$



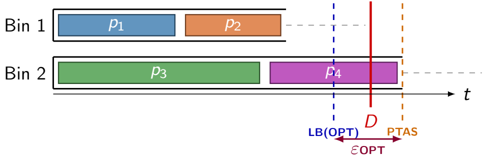
G-FFD Algorithm: guessing $\text{OPT}_{\text{Bin Packing}} = 2$

- Case 1: $D > LB(\text{OPT})$: A feasible schedule exists with the battery when selecting bins in non-decreasing order of fullness

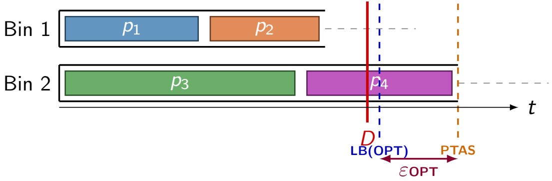


G-FFD Algorithm: guessing $\text{OPT}_{\text{Bin Packing}} = 2$

- Case 1: $D > LB(\text{OPT})$: A feasible schedule exists with the battery when selecting bins in non-decreasing order of fullness



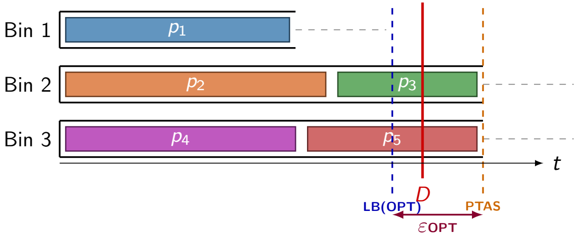
- Case 2: $D < LB(\text{OPT})$:
  - $\text{OPT}(P2||C_{max}) > D \Rightarrow \text{OPT}_{BinPacking} > 2$: There is no schedule with 2 bins



G-FFD Algorithm: Guessing $\text{OPT}_{\text{Bin Packing}} = 3$

Same strategy by solving PTAS for $P3||Cmax$ and using $\varepsilon = B/D(3-1)$, 2 cases:

- Case 1: A feasible schedule exists using 3 nights and days considering another value of ε than the guess before



- Case 2: $OPT_{BinPacking} > 3$ $(LB(OPT) > D)$
  - $\Rightarrow$ next guess...

G-FFD Algorithm: run FFD when guessing does not pay

- We solve FFD (First Fit Decreasing)
- Even if FFD is a 3/2-approx (solution with 3 bins and OPT=2), we know that $OPT > 2$ at this step, and even more with $OPT > \frac{1}{\min_k D_k} \sum_{j \in \mathcal{T}} p_j$.

$$FFD \leq \frac{11}{9} OPT + \frac{6}{9} \leq \frac{11}{9} OPT + \frac{6}{9} \cdot \frac{OPT}{x+1} = \frac{11 + \frac{6}{x+1}}{9} OPT$$ [1]

[1] Dósa, György, *The tight bound of first fit decreasing bin-packing algorithm is FFD (I)≤ 11/9 OPT (I)+ 6/9* in First International Symposium, ESCAPE 2007, Hangzhou, China, April 7-9, 2007

Conclusion and perspectives

Summary

- Battery can be considered as a credible resource augmentation
- Considered battery with or without loss

Perspectives

- Implement G-FFD with a battery and prove by experimentation that this is a better solution than using FFD alone
- Propose heuristics for the battery with loss model
- Extend the model to a multiprocessor architecture