Soutenance de thèse

Ordonnancement temps réel multiprocesseur pour la réduction de la consommation énergétique des systèmes embarqués

Vincent LEGOUT

Directeurs de thèse : Laurent PAUTET, Mathieu JAN

CEA NanoInnov

Mardi 08 Avril 2014

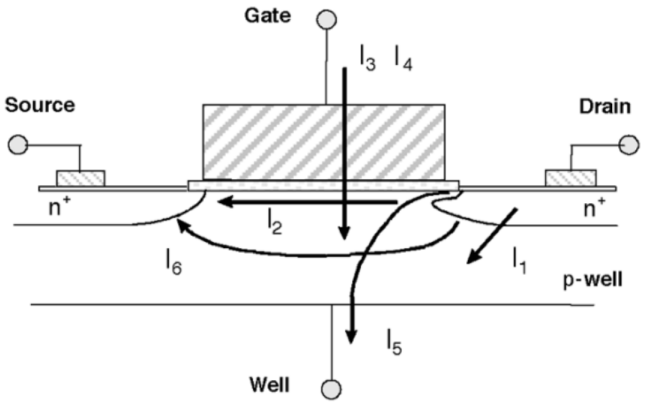
Plan

Systèmes temps réel multiprocesseurs embarqués

Validité d'un système temps réel = f(résultat, date)

- Évolution des systèmes temps réel embarqués
  - Les systèmes multiprocesseurs remplacent les systèmes monoprocesseurs
  - Puissance de calcul accrue
- Temps réel dur : dépassements d'échéances interdits
- Temps réel à criticité mixte : dépassements d'échéances autorisés pour les tâches de faible criticité
  - Application composée de tâches de différents niveaux de criticité

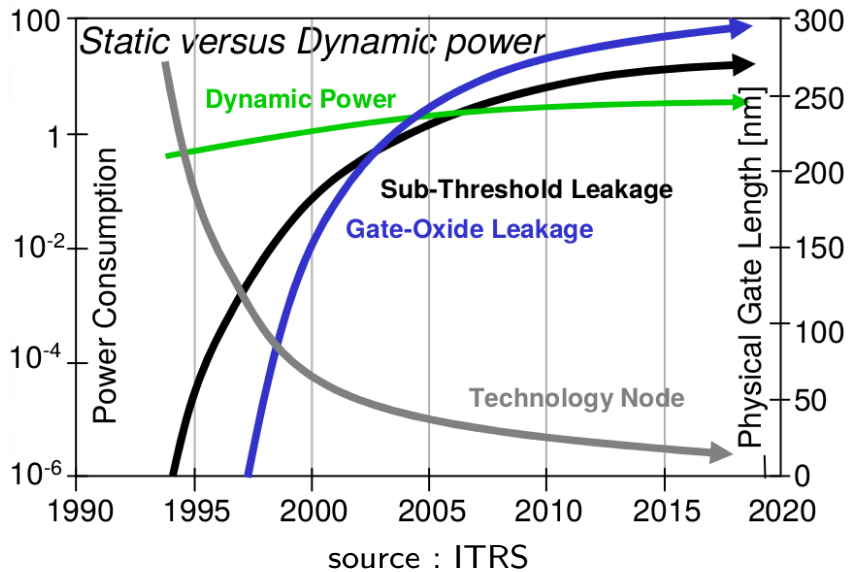
Réduction de la consommation statique des processeurs



source : Roy et al., 2003

- Consommations dynamique et statique
  - Dynamique : dépend de la vitesse du processeur
  - Statique : constante, due aux courants de fuite

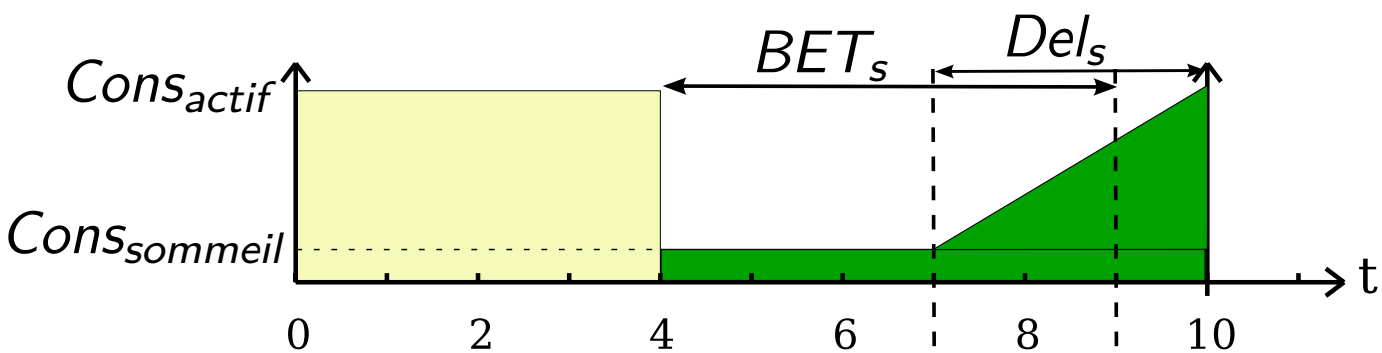
- Évolution des processeurs
  - Miniaturisation
  - Augmentation de la densité
- Puissance statique majoritaire



source : ITRS

Contraintes d'ordonnancement des états basse-consommation

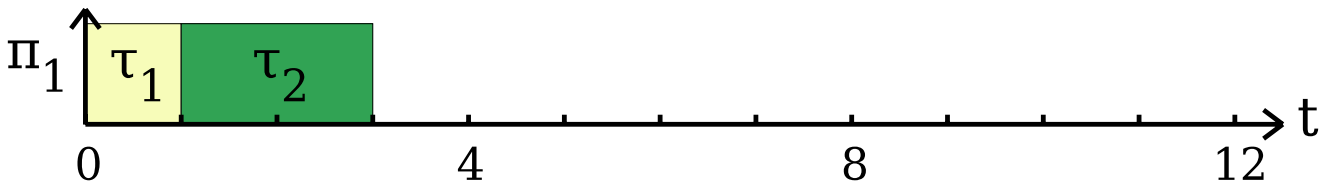
- Délai de transition Del_s pour revenir à l'état actif
  - Pénalité énergétique Pen_s associée
- BET_s (*Break Even Time*) : taille minimale de la période d'inactivité pour activer l'état basse-consommation



Intégration des états basse-consommation dans l'ordonnancement

Illustration avec 2 tâches :

	τ_1	τ_2
WCET	1	2
Période	4	6



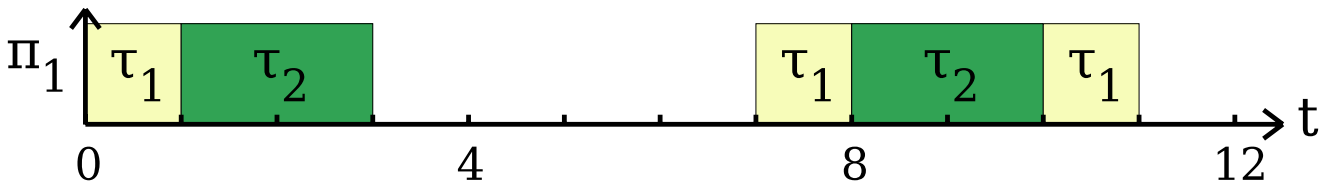
Activation d'un état basse-consommation à $t = 3$, quelle est la date maximale de réveil pour qu'aucune échéance ne soit violée ?

- De Lee et al. (ECRTS 03) à Awan et al. (RTNS 13)

Intégration des états basse-consommation dans l'ordonnancement

Illustration avec 2 tâches :

	τ_1	τ_2
WCET	1	2
Période	4	6



Activation d'un état basse-consommation à $t = 3$, quelle est la date maximale de réveil pour qu'aucune échéance ne soit violée ? $t = 7$

- De Lee et al. (ECRTS 03) à Awan et al. (RTNS 13)

Algorithmes existants pour systèmes multiprocesseurs

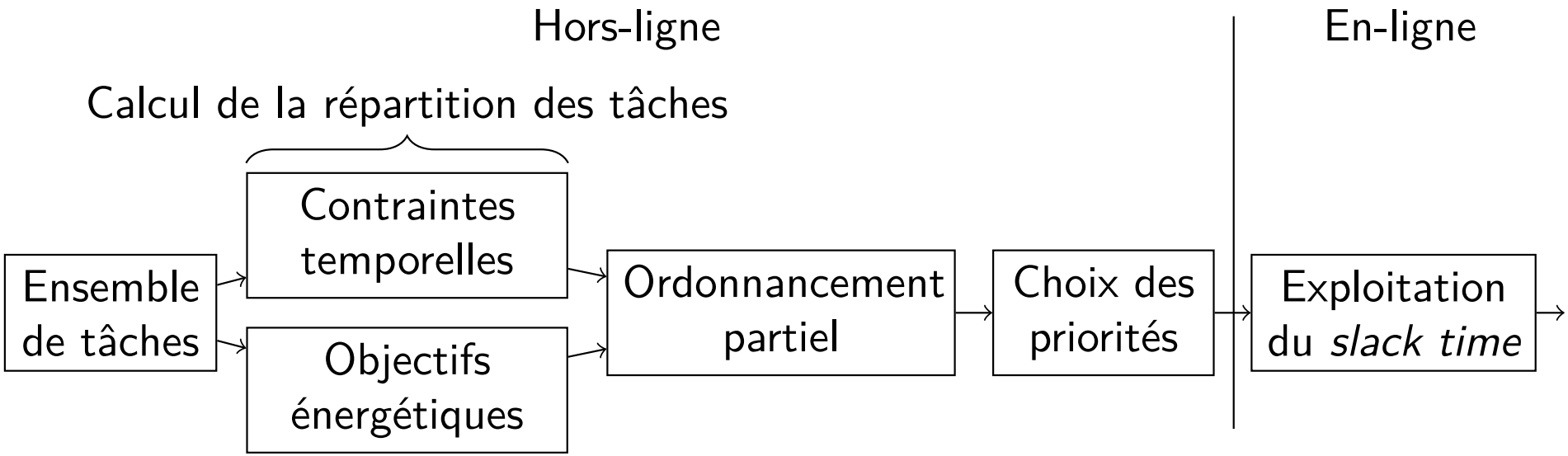
- Approches partitionnées (migrations interdites)
  - Chen et al. (2006) : partitionnement optimisant la taille des périodes périodes d'inactivité
  - Limitations : créer des périodes d'inactivité sur chaque processeur
  - Seo et al. (2008), Huang et al. (2010) : migrations restreintes
- Approches globales (migrations autorisées)
  - AsDPM (Bhatti et al. (2009)) : minimise le nombre de processeurs
  - Limitations : complexité importante ($O(n^3)$ avec n tâches), algorithme en-ligne non optimal au regard de l'utilisation des processeurs

Objectifs pour la réduction de la consommation statique

- Réduire la consommation énergétique des processeurs
- Activer les états basse-consommation les plus économes en énergie
  - Intégrer les contraintes dans l'ordonnancement
- Optimiser la taille de toutes les périodes d'inactivité
  - Autoriser les préemptions et migrations
  - Retarder les exécutions des tâches
- Garder une complexité en-ligne réduite

Plan

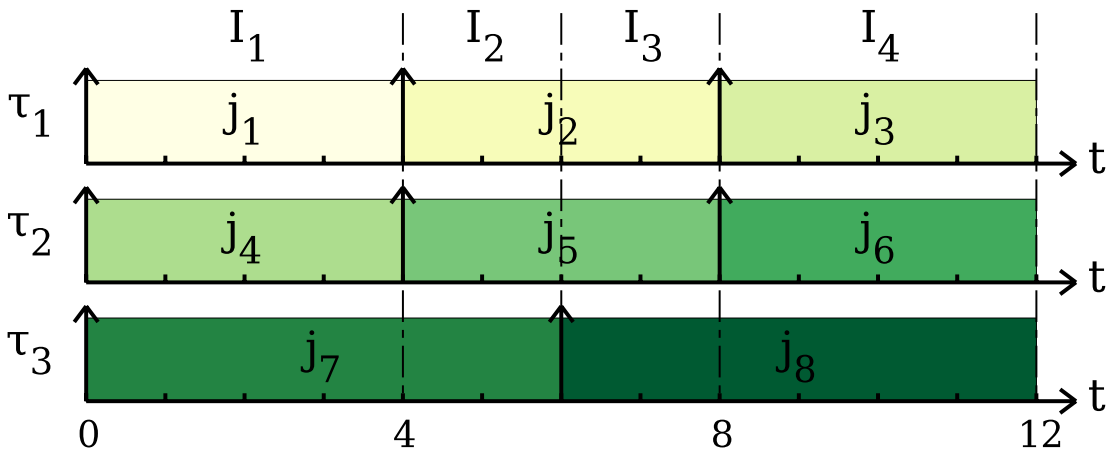
Division hors-ligne et en-ligne



- Hors-ligne : garantir un gain énergétique minimal hors-ligne
- En-ligne : étendre les périodes d'inactivité existantes

Ensemble de tâches périodiques à échéances implicites

- Tâches synchrones caractérisées par leur WCET et leur période
  - Migrations et préemptions autorisées
- Utilisation globale U : $m - 1 < U < m$ (m processeurs)
- Division de l'hyperpériode en intervalles
  - Frontières des intervalles : dates d'activation des travaux
- Distribution des travaux de 3 tâches périodiques



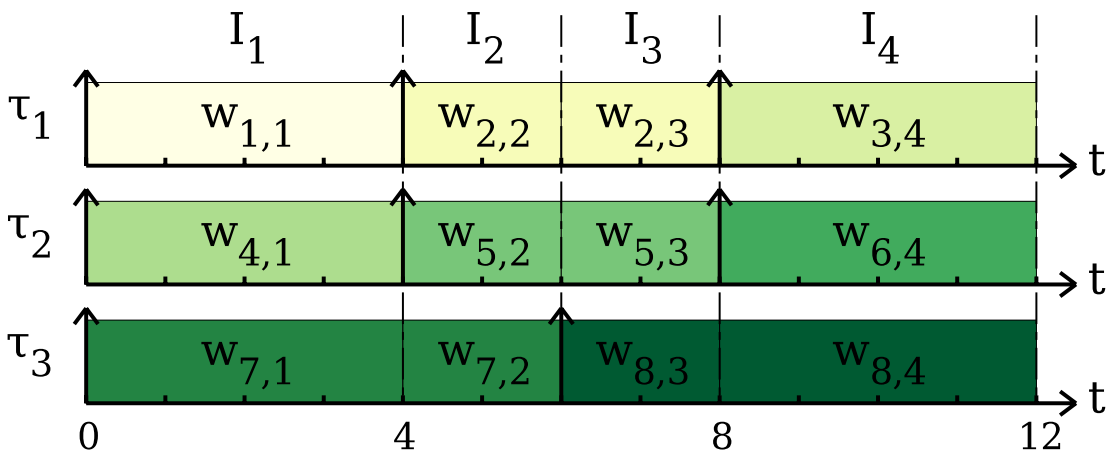
	τ_1	τ_2	τ_3
WCET	0.8	2.4	4
Période	4	4	6

$H = 12$, 4 intervalles

Calcul de la répartition des tâches sur les intervalles

- Calcul du poids de chaque travail sur chaque intervalle
  - $w_{j,k}$: poids du travail j sur l'intervalle k
  - Poids : portion de l'intervalle utilisée par un travail
  - Temps d'exécution sur un intervalle : poids $\times$ taille de l'intervalle

- Distribution des poids :



	τ_1	τ_2	τ_3
WCET	0.8	2.4	4
Période	4	4	6

Traduction des contraintes temporelles

- Limite l'utilisation du processeur dans chaque intervalle :
  - J_k : ensemble des travaux de l'intervalle k

$$\forall k, \sum_{j \in J_k} w_{j,k} \leq m$$

- Limite l'utilisation de chaque tâche dans chaque intervalle :

$$\forall k, \forall j, 0 \leq w_{j,k} \leq 1$$

- Garantit que tous les travaux sont exécutés :
  - $|I_k|$: taille de l'intervalle k
  - E_j : ensemble des intervalles où j est présent
  - $j.c$: WCET du travail j

$$\forall j, \sum_{k \in E_j} w_{j,k} \times |I_k| = j.c$$

Modélisation des périodes d'inactivité

- Nouvelle tâche τ' représentant l'inactivité du système
  - Tâche caractérisée par un WCET et une période
  - Poids de la tâche τ' : w_k

Période	H
Utilisation	$m - U$

- U $= m$, tous les processeurs sont toujours actifs
- Une exécution de τ' représente une période d'inactivité

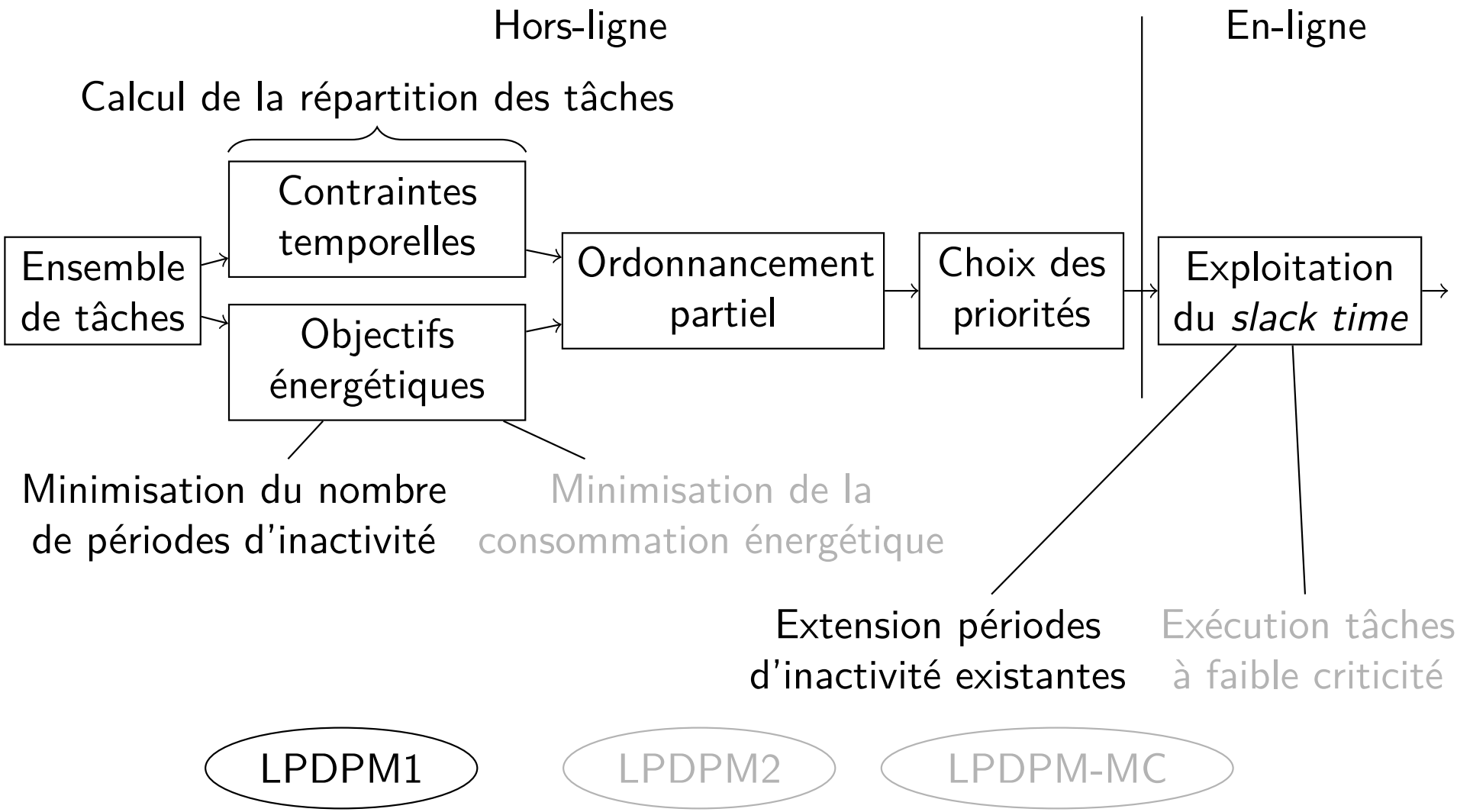
Choix des priorités dans les intervalles

- Utilisation de FPZL (*Fixed Priority with Zero Laxity*)
  - Ordonnancement des tâches suivant leur priorité et lorsque la laxité d'une tâche devient nulle
  - Algorithme d'ordonnancement optimal à l'intérieur des intervalles
- Plus le poids d'une tâche est important, plus sa priorité est importante
- Priorité particulière à τ' pour étendre les périodes d'inactivité

Contraintes et objectifs suivant le cadre d'application

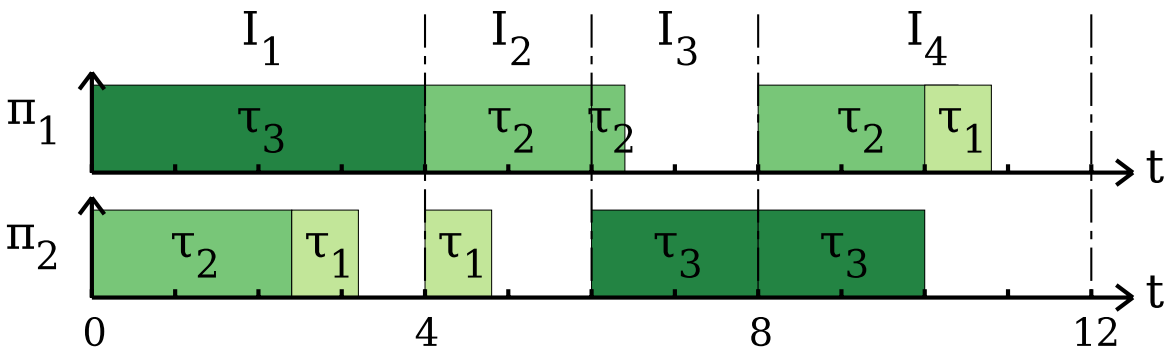
- Temps réel dur : dépassements d'échéances interdits, 2 approches
  - Minimiser le nombre de transitions entre les états basse-consommation et l'état actif
  - Minimiser la consommation énergétique
- Temps réel à criticité mixte : dépassements d'échéances autorisés pour les tâches à faible criticité
  - Hors-ligne : sous-estimer le WCET des tâches à faible criticité
  - En-ligne : compenser cette sous-estimation en exploitant les marges entre le temps d'exécution réel et le WCET

Application : temps réel dur et à criticité mixte



Minimisation du nombre de préemptions de τ'

- Ordonnancement avec les contraintes uniquement, sans tâche τ'

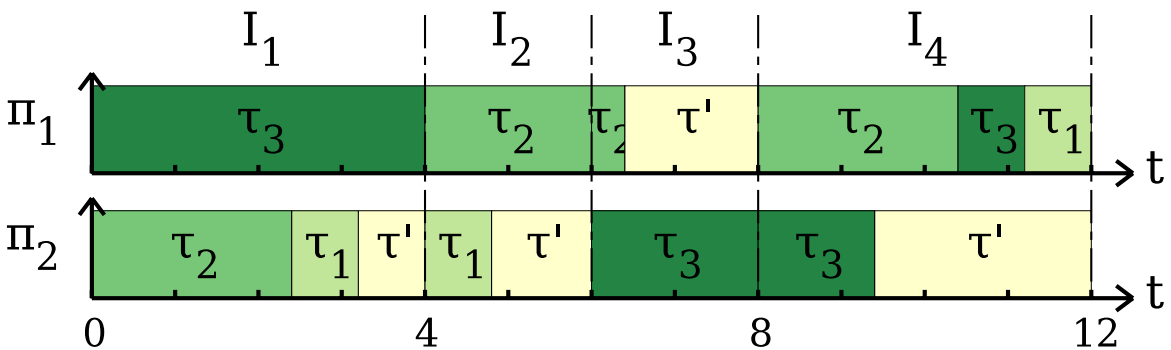


	τ_1	τ_2	τ_3
WCET	0.8	2.4	4
Période	4	4	6

5 périodes d'inactivité

Minimisation du nombre de préemptions de τ'

- Ordonnancement avec les contraintes uniquement, sans objectif
- Suppression d'une période d'inactivité sur I_4 car τ' ne peut être exécutée sur 2 processeurs en même temps



	τ_1	τ_2	τ_3
WCET	0.8	2.4	4
Période	4	4	6

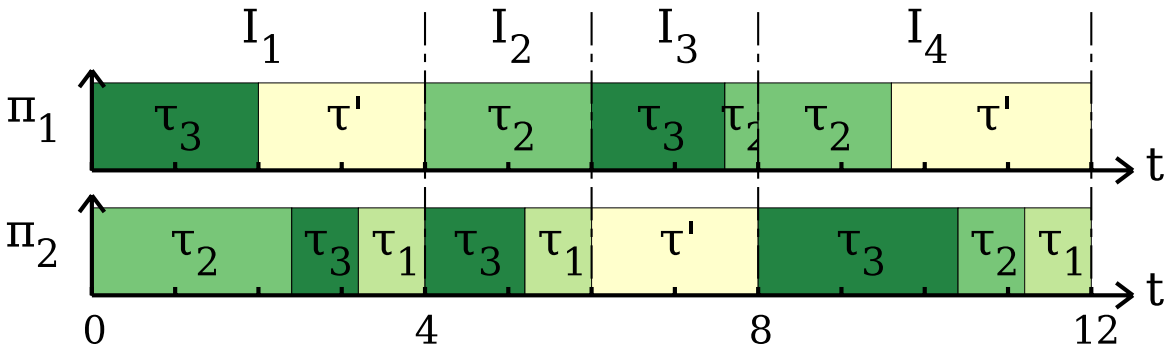
De 5 à 4 périodes d'inactivité

Minimisation du nombre de préemptions de τ'

- Minimisation des préemptions dans les intervalles
  - Favoriser les intervalles *plein* ($w_k = 1$) et *vide* ($w_k = 0$)
  - Ajouter 2 variables binaires f_k et e_k

$$f_k = \begin{cases} 0 & \text{si } w_k = 1 \\ 1 & \text{sinon} \end{cases} \qquad e_k = \begin{cases} 0 & \text{si } w_k = 0 \\ 1 & \text{sinon} \end{cases}$$

Objectif : Minimiser $\sum_k f_k + e_k$



	τ_1	τ_2	τ_3
WCET	0.8	2.4	4
Période	4	4	6

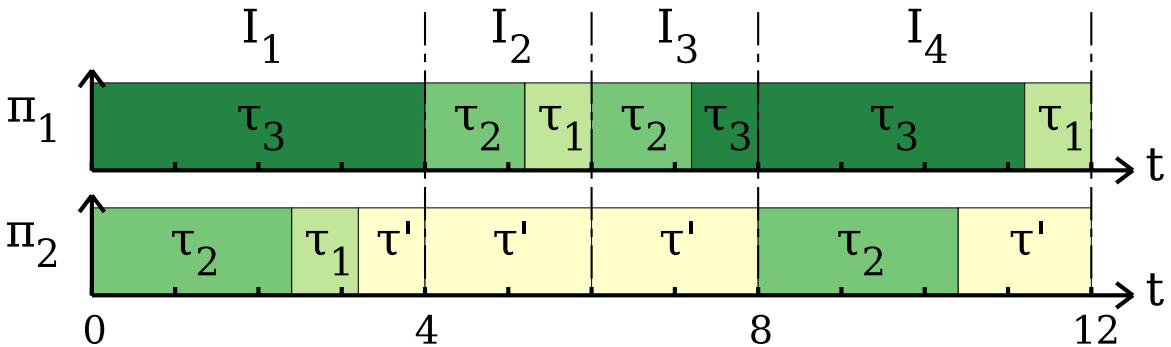
De 5 à 3 périodes d'inactivité

Minimisation du nombre de préemptions de τ'

- Minimisation des préemptions entre 2 intervalles consécutifs
  - Faire que les intervalles où $f_k = 0$ soient consécutifs (resp $e_k = 0$)
  - Ajouter 2 variables binaires fc_k et ec_k

$$fc_k = \begin{cases} 1 & \text{si } f_k = 1 \text{ et } f_{k+1} = 0 \\ 0 & \text{sinon} \end{cases} \qquad ec_k = \begin{cases} 1 & \text{si } e_k = 1 \text{ et } e_{k+1} = 0 \\ 0 & \text{sinon} \end{cases}$$

Objectif : Minimiser $\sum_k f_k + e_k + fc_k + ec_k$

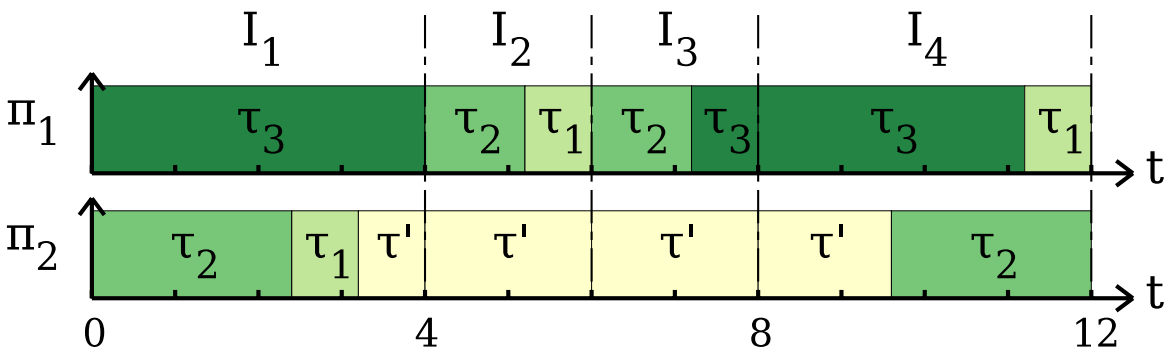


	τ_1	τ_2	τ_3
WCET	0.8	2.4	4
Période	4	4	6

De 5 à 2 périodes d'inactivité

Minimisation du nombre de préemptions de τ'

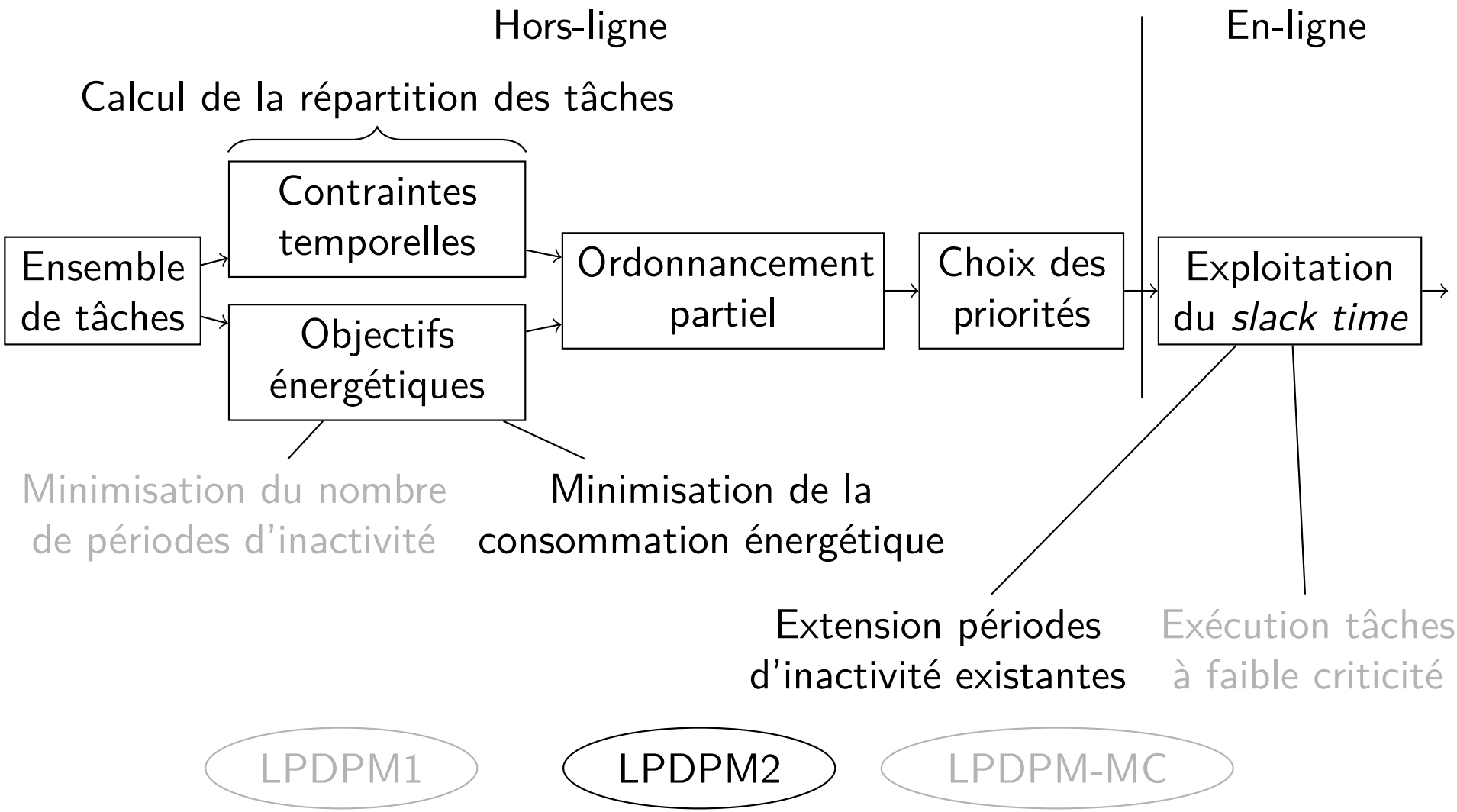
- Choix de la priorité de τ' en début d'intervalle
  - Plus haute priorité si un état basse-consommation est actif
  - Plus faible priorité sinon
- Exploitation du *slack time* pour réduire la consommation énergétique
  - *Slack time* donné à τ' pour étendre la période d'inactivité courante



	τ_1	τ_2	τ_3
WCET	0.8	2.4	4
Période	4	4	6

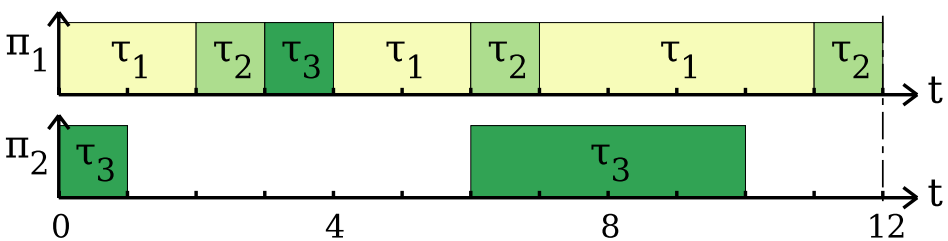
De 5 à 1 seule période d'inactivité !

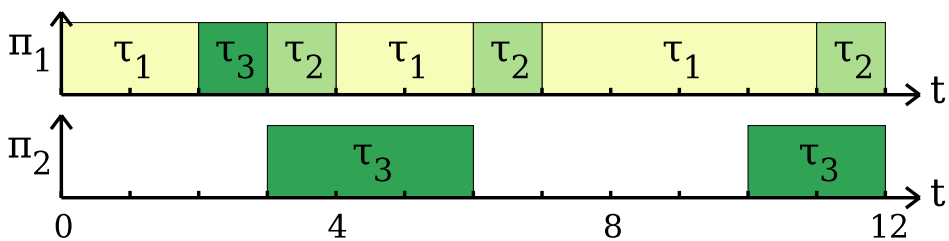
Application : temps réel dur et à criticité mixte



Minimisation de la consommation énergétique

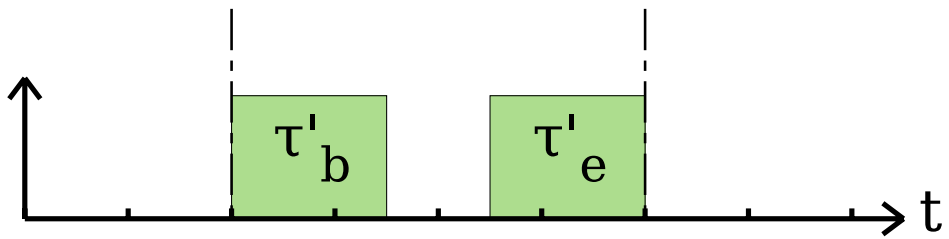
- LPDPM1 : minimisation du nombre de transitions
  - Ne connait pas la taille des périodes d'inactivité hors-ligne
  - Ne tient pas compte des caractéristiques des états basse-consommation, périodes d'inactivité de taille inférieure au BET
- Illustration avec 3 tâches et 2 périodes d'inactivité
  - Si BET $= 3$, impossible d'activer un état basse-consommation dans la 2ème période d'inactivité du scénario de gauche





Modélisation des périodes d'inactivité

- Nécessite de calculer la taille et la consommation énergétique de chaque période d'inactivité
- τ' exécutée en début et fin d'intervalle
  - Division de τ' en deux sous-tâches τ'_b et τ'_e
  - Choix similaire à LPDPM1 fait maintenant hors-ligne



- Une période d'inactivité par intervalle (de taille q_k)
- Complexité dépendant du nombre d'intervalles de manière exponentielle

Calcul de la consommation énergétique

- Consommation énergétique de la période d'inactivité de l'intervalle k avec l'état basse-consommation s :
  - $Cons_s$: consommation de l'état basse-consommation s
  - Pen_s : pénalité énergétique de l'état basse-consommation s

$$P_{k,s} = Cons_s \times q_k + Pen_s$$

Fonction objectif finale : Minimiser $\sum_k P_k$

Comparaison entre LDPDPM1, LPDPM2, RUN et U-EDF

- Critères d'évaluation :
  - Utilisation des états basse-consommation
  - Consommation énergétique
  - Nombre de préemptions
- RUN et U-EDF : algorithmes multiprocesseurs optimaux
  - Meilleurs algorithmes d'ordonnancement multiprocesseurs optimaux pour réduire le nombre de préemptions et de migrations
- Aucun autre algorithme d'ordonnancement multiprocesseur optimal minimisant la consommation statique

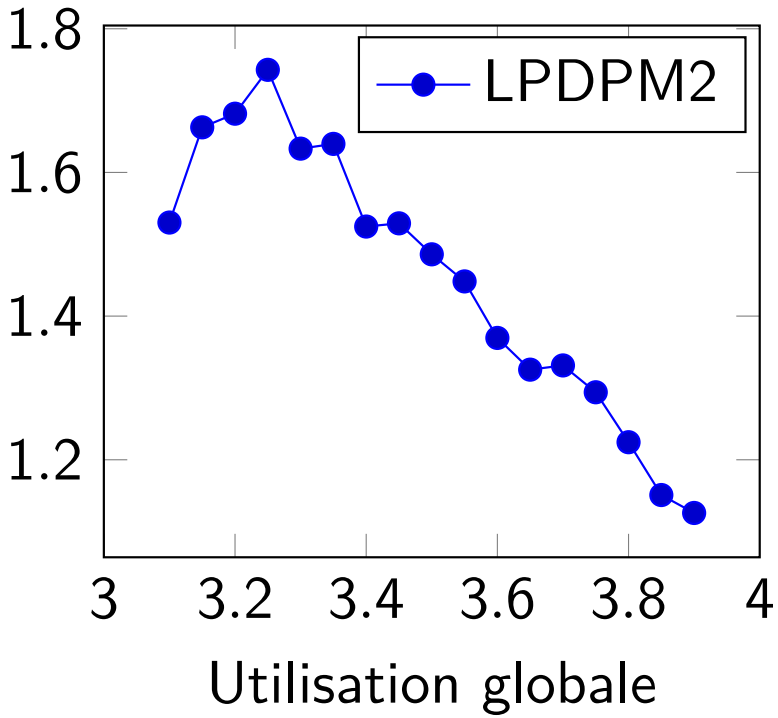
Conditions de simulation

- Ensembles de 10 tâches ordonnancées sur 2 hyperpériodes
- 4 processeurs : utilisation globale entre 3 et 4
- Pour chaque utilisation globale 500 ensembles de tâches aléatoires
- Programme linéaire résolu avec CPLEX (1 min par résolution)
- Calcul de la consommation quand les processeurs sont inactifs
- États basse-consommation :

Mode	Consommation énergétique	Délai de transition
Run	1	
Sleep	0.5	$0.01ms$
Stop	0.1	$2ms$
Standby	0.00001	$10ms$

Consommation de LPDPM1 plus faible que LPDPM2

Consommation énergétique normalisée de LPDPM2 par rapport à LPDPM1

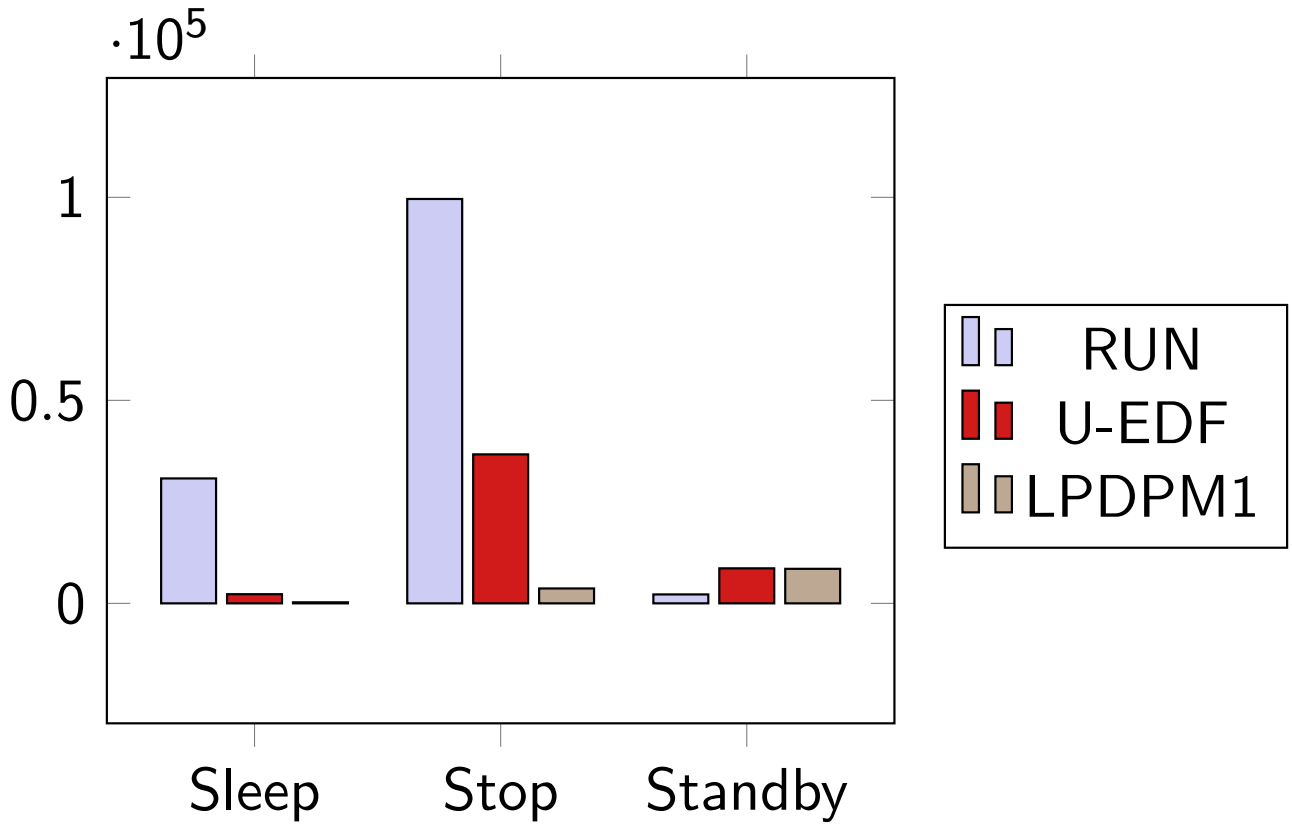


Optimalité des poids (%)

m	LPDPM1	LPDPM2
2	100	23
4	99	0.1
8	88	0

Différence plus importante quand l'utilisation est proche de 3, due à la non-optimalité des solutions de LPDPM2

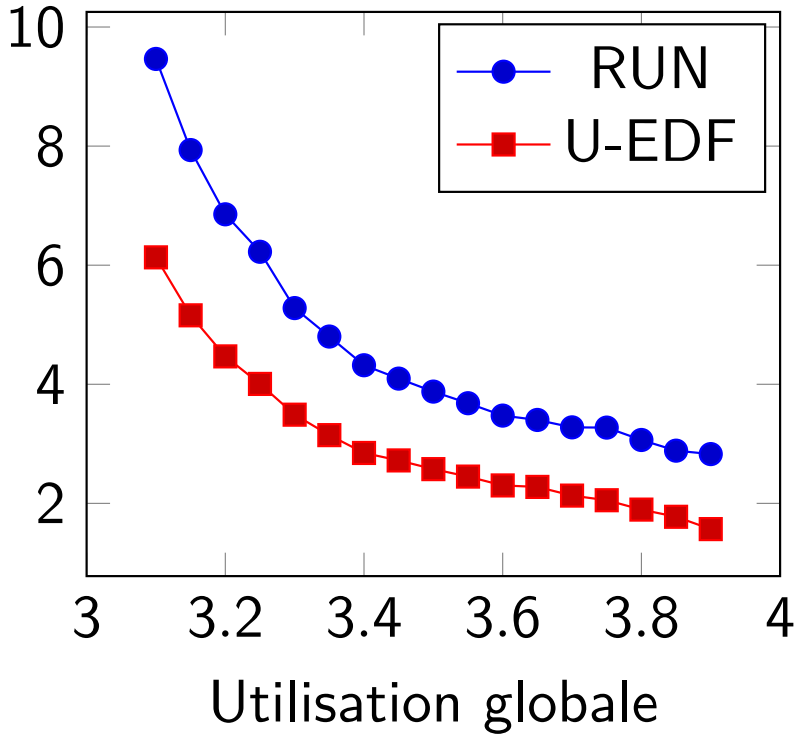
Répartition des activations des états basse-consommation



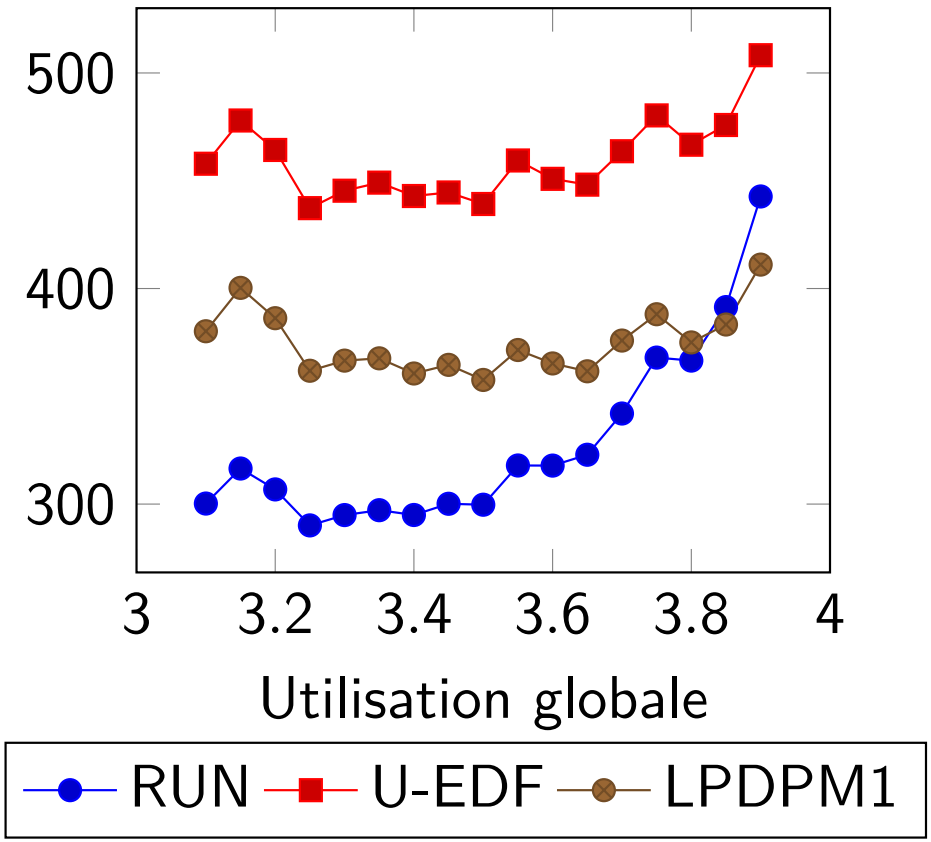
LPDPM1 utilise davantage l'état basse-consommation le plus efficace *Standby*, RUN et U-EDF activent essentiellement les états *Sleep* et *Stop*

Consommation réduite et nombre de préemptions similaire

Consommation énergétique normalisée par rapport à LPDPM

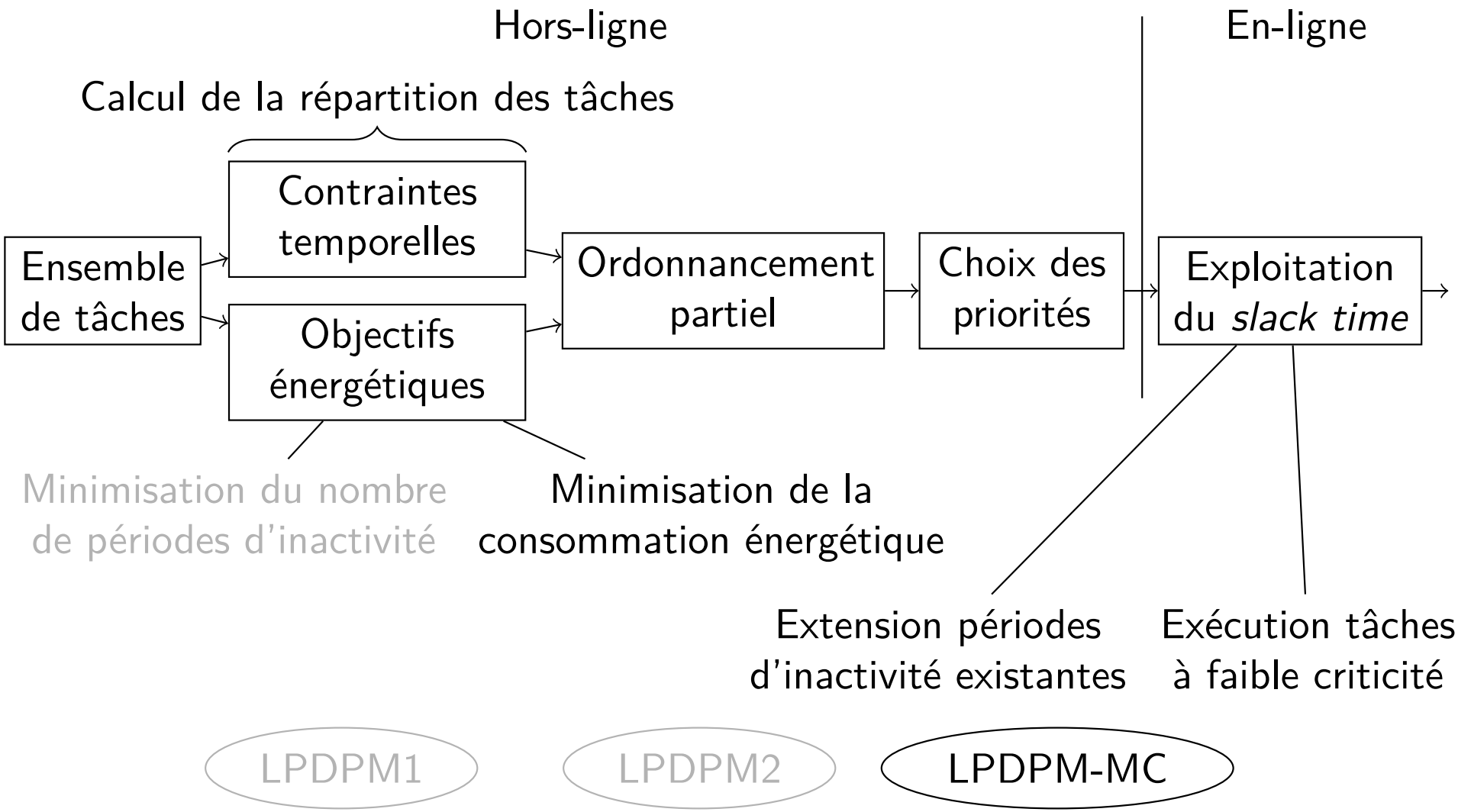


Nombre de préemptions



Consommation énergétique jusqu'à 10 fois plus faible avec LPDPM

Application : temps réel dur et à criticité mixte



Compromis entre consommation et taux de défaillances

- 2 niveaux de criticité :
  - Haute criticité : temps réel dur
  - Faible criticité : dépassements d'échéances autorisés
- Présence d'une marge importante entre temps d'exécution et WCET
- Utilisation de cette marge pour les tâches à faible criticité pour augmenter la taille des périodes d'inactivité
  - Peut entrainer des dépassements d'échéances pour ces tâches à faible criticité

Réduction du temps processeur réservé aux tâches à faible criticité

Ne réserver qu'un pourcentage du WCET pour les tâches à faible criticité, pourcentage représenté par la variable $\alpha \in [0, 1]$

- $\forall j_{\mathrm{LO}}$: ensembles des travaux à faible criticité

$$\forall j_{\mathrm{LO}}, \quad \sum_{k \in E_j} w_{j,k} \times |I_k| = \alpha \times j.c$$

- Possible de désactiver un ou plusieurs processeurs pour que $m - 1 < U < m$ soit toujours vérifié
- Utilisation de la distribution des temps d'exécution pour choisir α (e.g. Gumbel)

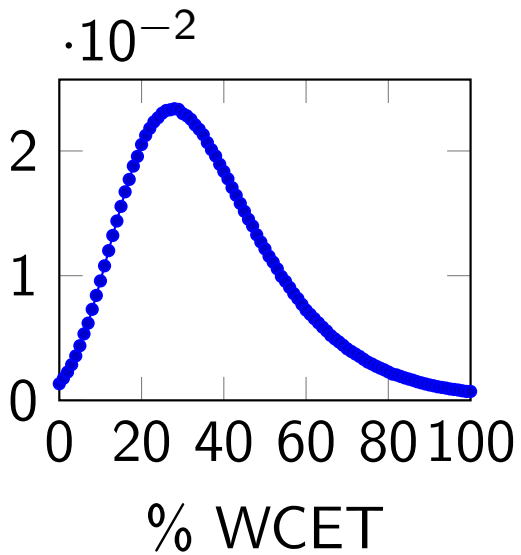
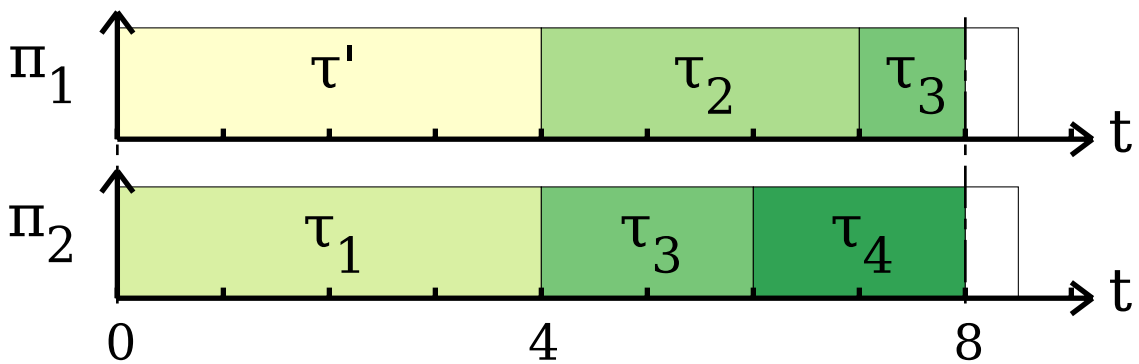


Illustration de l'exploitation du *slack time* en-ligne

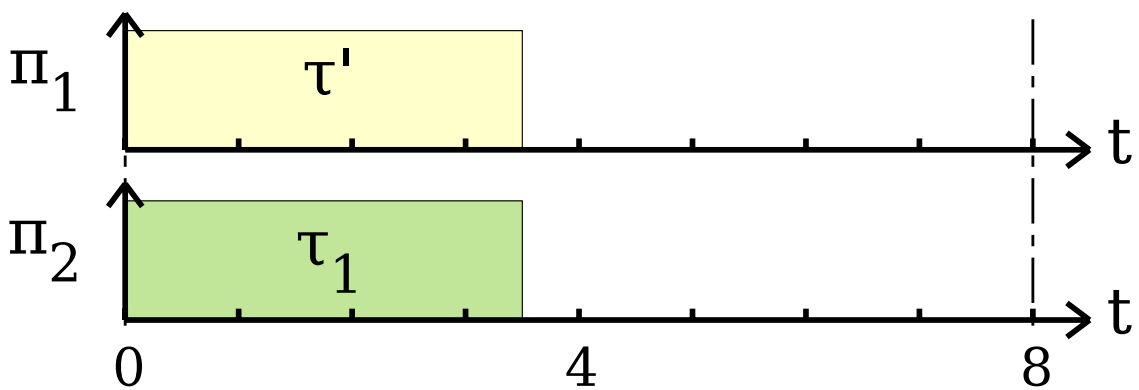
	τ_1	τ_2	τ_3	τ_4
Poids	4	3	3	2
Criticité	Haute	Haute	Faible	Faible



Dépassements d'échéances pour τ_3 et τ_4

Illustration de l'exploitation du *slack time* en-ligne

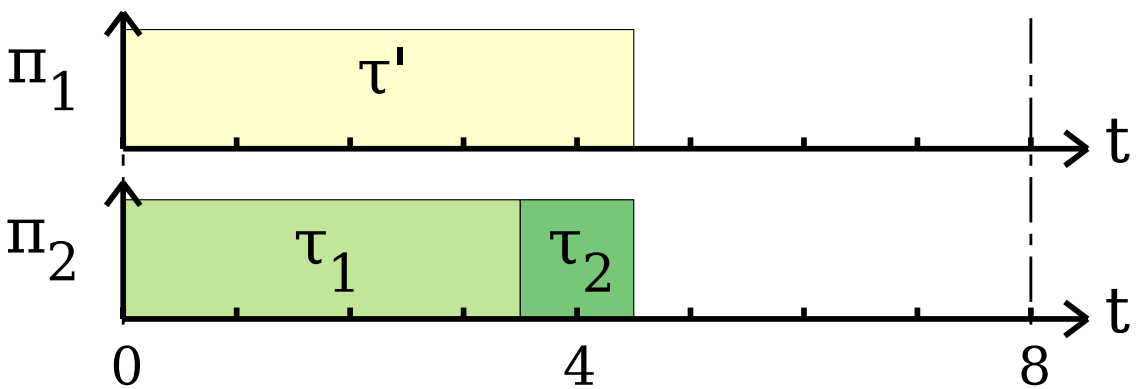
	τ_1	τ_2	τ_3	τ_4
Poids	4	3	3	2
Criticité	Haute	Haute	Faible	Faible



$t = 3.5$, τ_1 termine, 0.5 unité de *slack time* libérée, donnée à τ'

Illustration de l'exploitation du *slack time* en-ligne

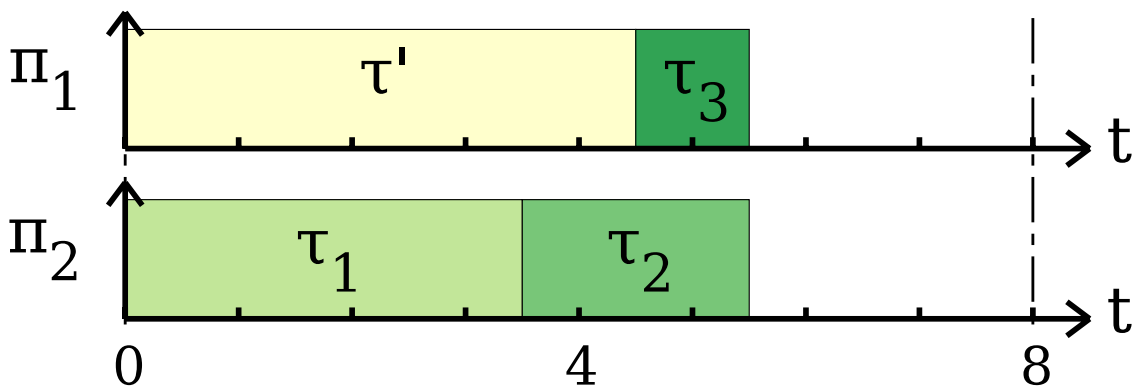
	τ_1	τ_2	τ_3	τ_4
Poids	4	3	3	2
Criticité	Haute	Haute	Faible	Faible



$t = 4.5$, τ' termine, réveil du processeur π_1

Illustration de l'exploitation du *slack time* en-ligne

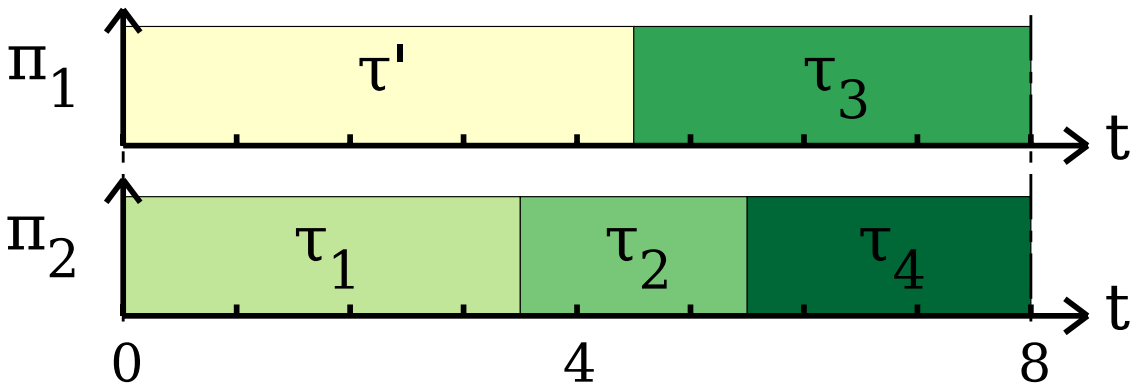
	τ_1	τ_2	τ_3	τ_4
Poids	4	3	3	2
Criticité	Haute	Haute	Faible	Faible



$t = 5.5$, τ_2 termine, 1 unité de *slack time* libérée, donnée à τ_3 et τ_4

Illustration de l'exploitation du *slack time* en-ligne

	τ_1	τ_2	τ_3	τ_4
Poids	4	3	3	2
Criticité	Haute	Haute	Faible	Faible



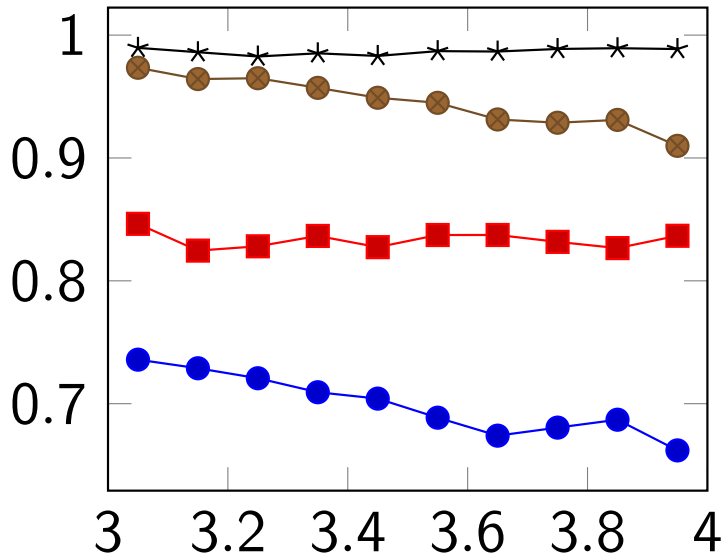
$t = 8$, fin de l'intervalle, aucun dépassement d'échéance pour τ_3 et τ_4, aucune création de période d'inactivité

Conditions de simulation

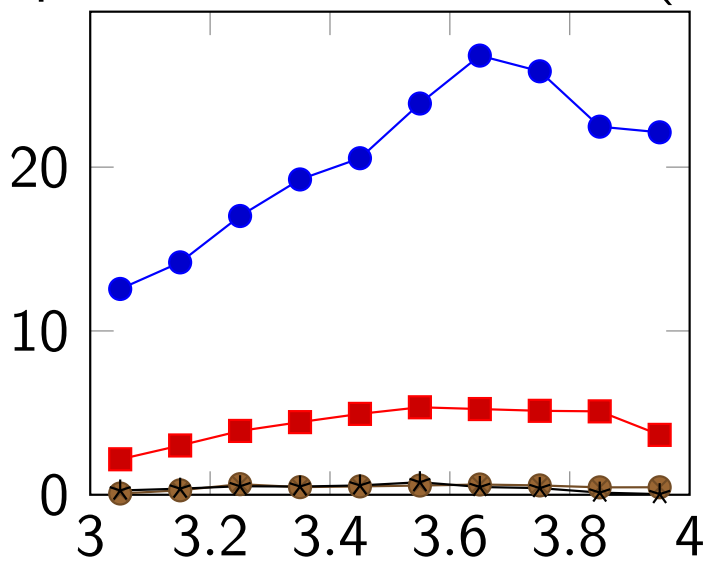
- Différentes valeurs de α : 0.2, 0.4, 0.6 & 0.8
- Environnement de simulation similaire à LPDPM
  - 4 processeurs, 10 tâches
  - Utilisation globale entre 3 et 4
  - 500 ensembles de tâches générés aléatoirement
  - 4 tâches à haute criticité, 6 tâches à faible criticité
- Les tâches à haute criticité utilisent leur WCET
- Les tâches à faible criticité utilisent leur temps d'exécution estimé
  - Calculé en utilisant la distribution de Gumbel

Consommation et taux de défaillances

Consommation énergétique



Dépassements d'échéances (%)



$\alpha = 0.2$	$\alpha = 0.4$	$\alpha = 0.6$	$\alpha = 0.8$

Plus α est faible, plus la consommation énergétique est réduite et plus le taux de défaillances est important ($\alpha = 0.4$ compromis acceptable)

Plan

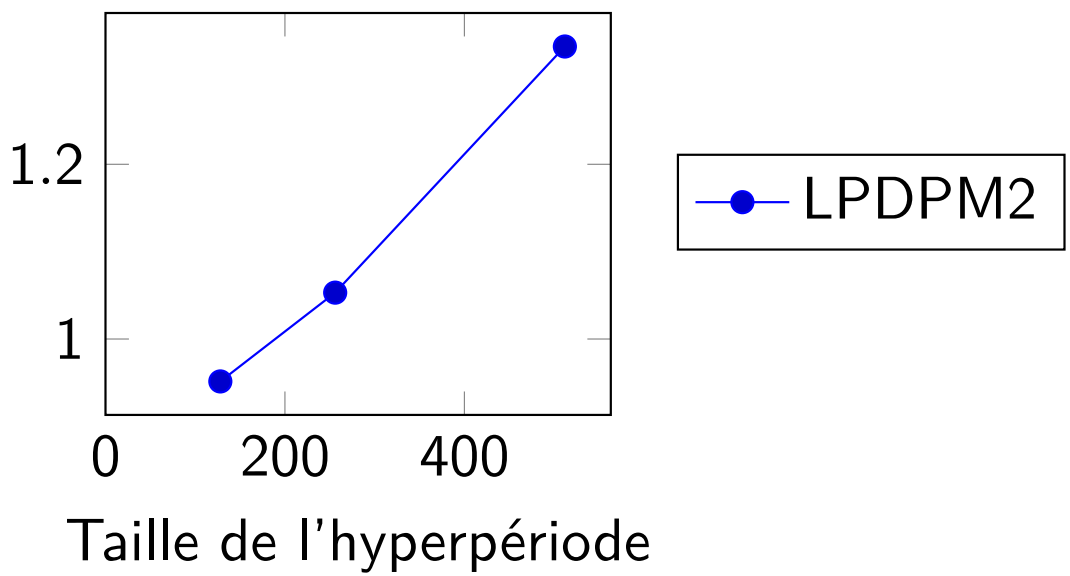
Conclusion

- Approche hybride pour la conception d'algorithmes d'ordonnancement multiprocesseurs optimaux minimisant la consommation statique
  - Premier algorithme d'ordonnancement multiprocesseur de cette classe
  - Programmation linéaire hors-ligne garantissant un gain énergétique minimal
  - Ordonnancement en-ligne dépendant du contexte applicatif
- Temps réel dur : 2 approches minimisant respectivement le nombre de transitions et la consommation énergétique
  - Consommation énergétique jusqu'à 10 fois inférieure et nombre de préemptions similaire aux algorithmes existants
- Temps réel à criticité mixte : exploitation de la marge entre temps d'exécution réel et WCET pour réduire la consommation
  - Consommation réduite de 16%, taux de défaillances de moins de 6%

Perspectives

- Implémentation et calcul de la consommation sur un système réel
- Approfondissement des évaluations entre LPDPM1 et LPDPM2
  - Identification du nombre d'intervalles comme facteur critique
  - Proposition d'une solution hybride entre LPDPM1 et LPDPM2

Consommation énergétique vs taille maximale de l'hyperpériode



Perspectives

- Approche duale pour systèmes temps réel à criticité mixte
  - Gain énergétique fixé et minimisation du taux de défaillances
- Couplage avec les modèles thermiques et de fiabilité
  - Activation d'états basse-consommation sur 2 processeurs simultanément

Publications & Communications

Conférence

- Reducing Static Energy Consumption on Multiprocessor Real-Time Systems
  - w/ Mathieu Jan and Laurent Pautet. 21st International Conference on Real-Time Networks and Systems (RTNS). Nice. Octobre 2013. *Outstanding Paper & Best Student Paper Award*.

Soumission Journal

- Scheduling Algorithms to Reduce the Static Energy Consumption of Real-Time Systems.
  - w/ Mathieu Jan and Laurent Pautet. Journal of Real-Time Systems.

Workshops

- Mixed-Criticality Multiprocessor Real-Time Systems : Energy Consumption vs Deadline Misses
  - w/ Mathieu Jan and Laurent Pautet. 1st workshop on Real-Time Mixed Criticality Systems (ReTiMiCS). Taipei, Taiwan. Août 2013.
- An off-line multiprocessor real-time scheduling algorithm to reduce static energy consumption
  - w/ Mathieu Jan and Laurent Pautet. 1st Workshop on Highly-Reliable Power-Efficient Embedded Designs (HARSH). Shenzhen, China. Février 2013.

École d'Été

- Réduction de la consommation statique des systèmes temps réel multiprocesseurs
  - w/ Mathieu Jan and Laurent Pautet. École d'Été Temps Réel. Toulouse. Août 2013.

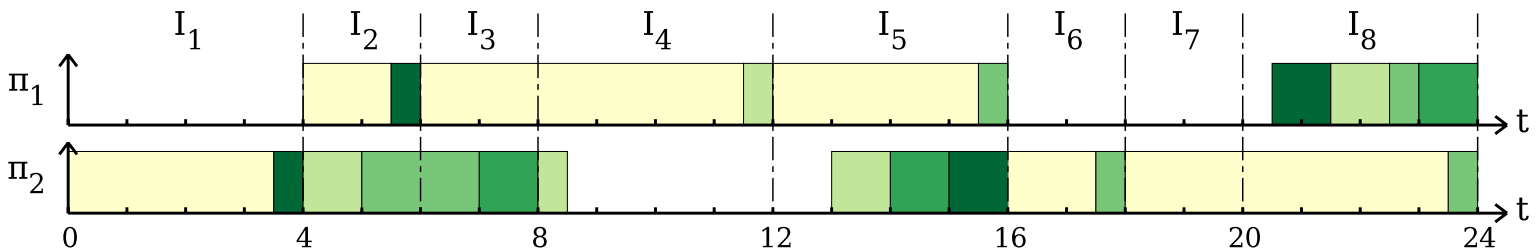
Séminaire

- Séminaire MeFoSyLoMa (Méthodes Formelles pour les Systèmes Logiciels et Matériels).
  - ENS Cachan. Mars 2013.

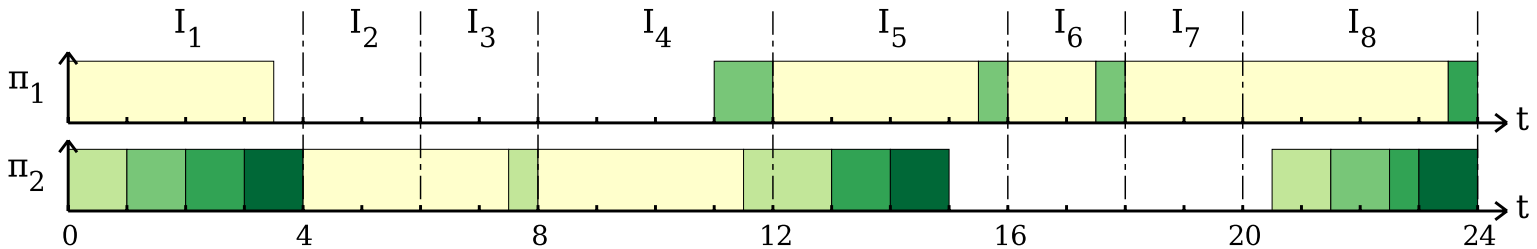
Apport de la modélisation LPDPM2 par rapport à LPDPM1

	τ_1	τ_2	τ_3	τ_4	τ_5
WCET	3.5	1	1	1	1
Période	4	6	8	8	8

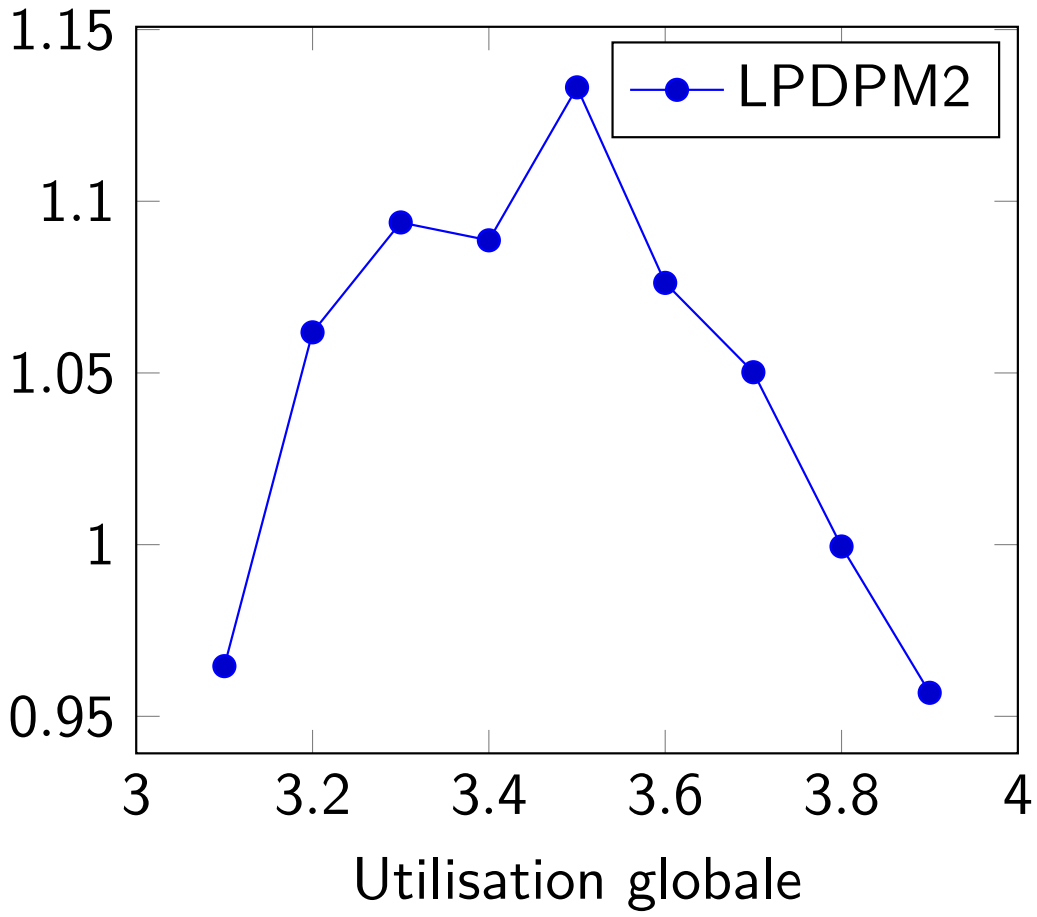
LPDPM1



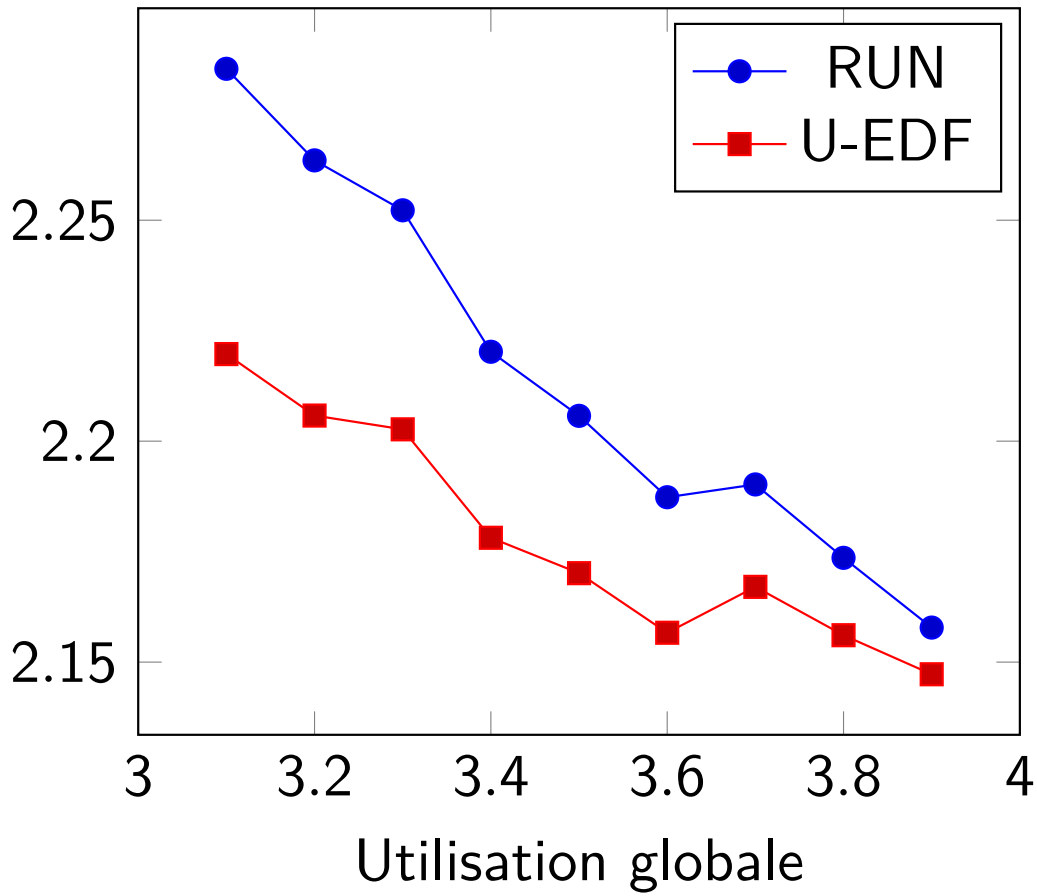
LPDPM2



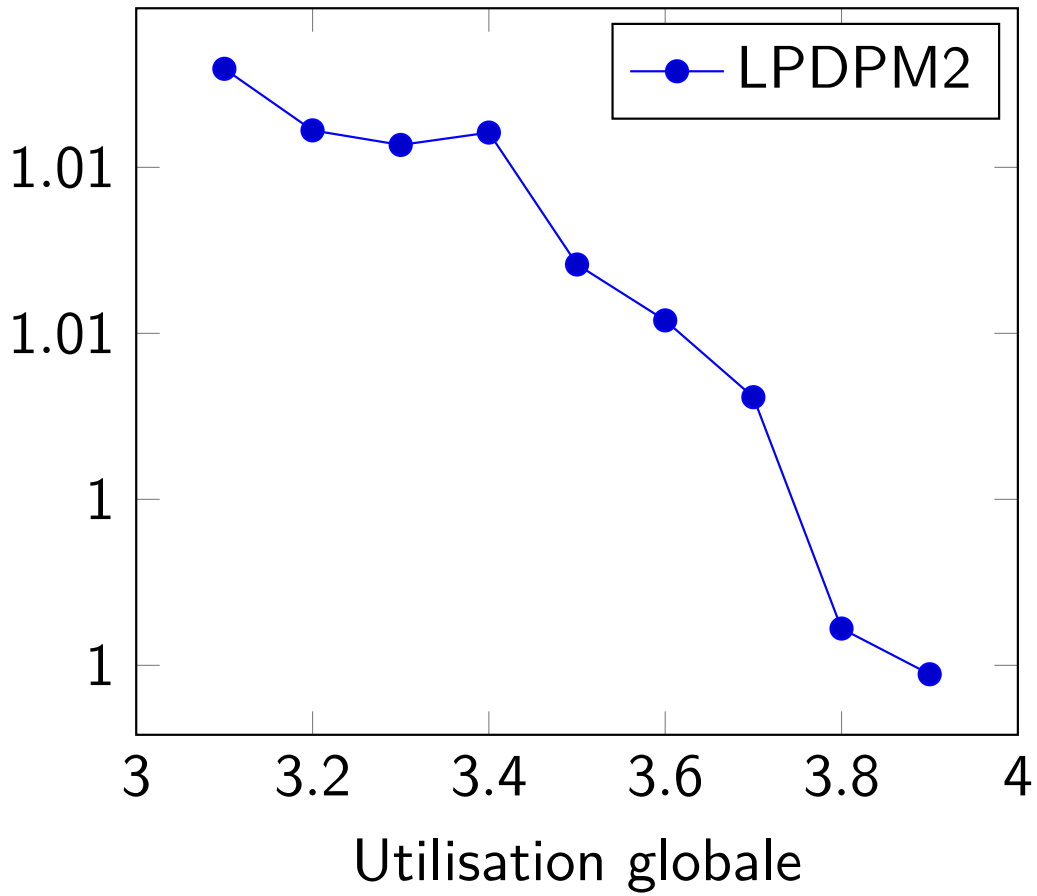
Consommation énergétique normalisée de LPDPM2 vs LPDPM1



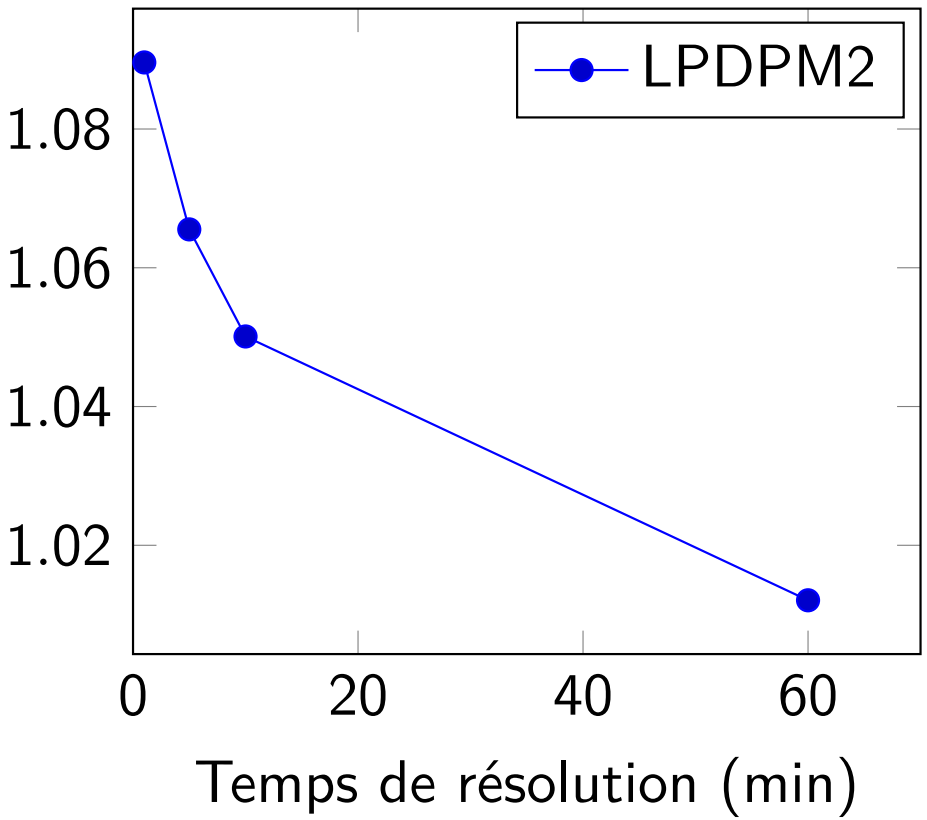
Consommation énergétique normalisée en utilisant distribution Gumbel



Consommation énergétique normalisée de LPDPM2 vs LPDPM1



Consommation énergétique suivant le temps de résolution



% optimalité suivant temps de résolution (min)

Temps	LPDPM1	LPDPM2
1	100	0
5	100	0.02
10	100	0.02
60	100	0.03