Systèmes embarqué, temps réel et mobile

Les algorithmes d’ordonnancement : cas de l’EDF

AHIDOTE Miguel

BLEOSSI Cédric

MONTCHO Wesley

Sommaire

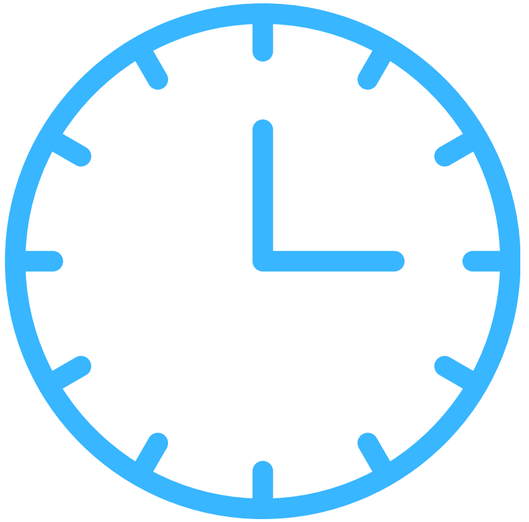
Introduction

Dans les systèmes temps réel, le respect strict des contraintes temporelles est aussi essentiel que la justesse des calculs effectués. L'algorithme Earliest Deadline First (EDF) s'impose comme une solution optimale pour garantir la ponctualité des tâches, en attribuant la priorité à celles dont l'échéance est la plus proche. Cet exposé présente le principe de fonctionnement de l'EDF, sa démonstration d'optimalité, ainsi que ses avantages et limites dans l'ordonnancement des systèmes embarqués.

Rappel sur les systèmes temps réel

Un système temps réel est un système informatique dont le fonctionnement est soumis à des contraintes de temps strictes. Il doit réagir à des événements de son environnement dans un délai précis et garanti, autrement dit, la ponctualité est aussi importante que l'exactitude du résultat.

Le non respect des contraintes de temps peut entrainer de graves conséquences : pertes financières, matérielles, mort...



Paramètres clés liés aux systèmes temps réel

- **Tâche (Task) :** Travail spécifique à accomplir
- **Job :** Instance d'une tâche
- **Deadline (Di) :** Date limite d'exécution de la tâche
- **Période (T) :** Intervalle entre deux activations de tâches
- **Temps d'exécution (Ci) :** Temps nécessaire pour exécuter la tâche i
- **Date d’activation (Si)** de la date i
- **Taux d’utilisation du CPU (U)** (Ui = Ci/Ti pour la tache i)

Paramètres clés liés aux systèmes temps réel

Caractéristiques du système

N : Nombre total de tâches périodiques à exécuter

U = Σ (Ui) : Utilisation globale du processeur

Les algorithmes d’ordonnancement : caractéristiques

Test d’ordonnancement

Formule qui fournit une condition nécessaire et/ou suffisante pour qu’un algorithme satisfasse les contraintes temporelles d’un ensemble de tâches.

Test d’admission

Formule qui vérifie que l’ajout d’une nouvelle tâche préserve le respect des contraintes temporelles de l’ensemble de tâches précédent .

Les algorithmes d’ordonnancement : caractéristiques

Quelques opérateurs

Plafond : Le nombre de fois qu’une tâche interrompt une autre sur une période donnée ($\lceil x \rceil$)

Plancher : Le nombre d’activations déjà passées avant un instant donné ($\lfloor x \rfloor$)

Classification des algorithmes d’ordonnancement

L’ordonnancement des tâches apériodiques

- Serveur à scrutation
- Serveur différé
- Serveur sporadique

L’ordonnancement des tâches périodiques

- Ordonnancement préemptif
- Ordonnancement non-préemptif

Classification des algorithmes d’ordonnancement

Les algos non pré-emptifs

Ils sélectionnent une tâche qui conserve le processeur jusqu'à son achèvement, même si une tâche plus prioritaire arrive entre-temps.

Exemple : FCFS (First Come First Served)

Les algos pré-emptifs

L'arrivée d'une tâche plus prioritaire peut interrompre la tâche en cours d'exécution, qui est remise en attente.

Exemple : Round Robin, Shortest Remaining Time First (SRTF), Rate Monotonic (RM)

L’algo de l’EDF appartient à la classe des algorithmes préemptifs

L’EDF : c’est quoi ?

EDF (Earliest Deadline First) : C’est un algorithme d'ordonnancement temps réel qui attribue une priorité aux tâches en fonction de leur échéance la plus proche. La tâche avec l'échéance la plus proche est la plus prioritaire et est exécutée en premier. C'est un algorithme préemptif à priorité dynamique, ce qui signifie que l'exécution d'une tâche peut être interrompue si une tâche plus prioritaire arrive.

L’EDF : le principe

Hypothèses

- Tâches périodiques, préemptibles et indépendantes
- Activation en début de période (Si = 0) Échéance en fin de période (Di = Ti)
- Pire cas Ci connu à priori

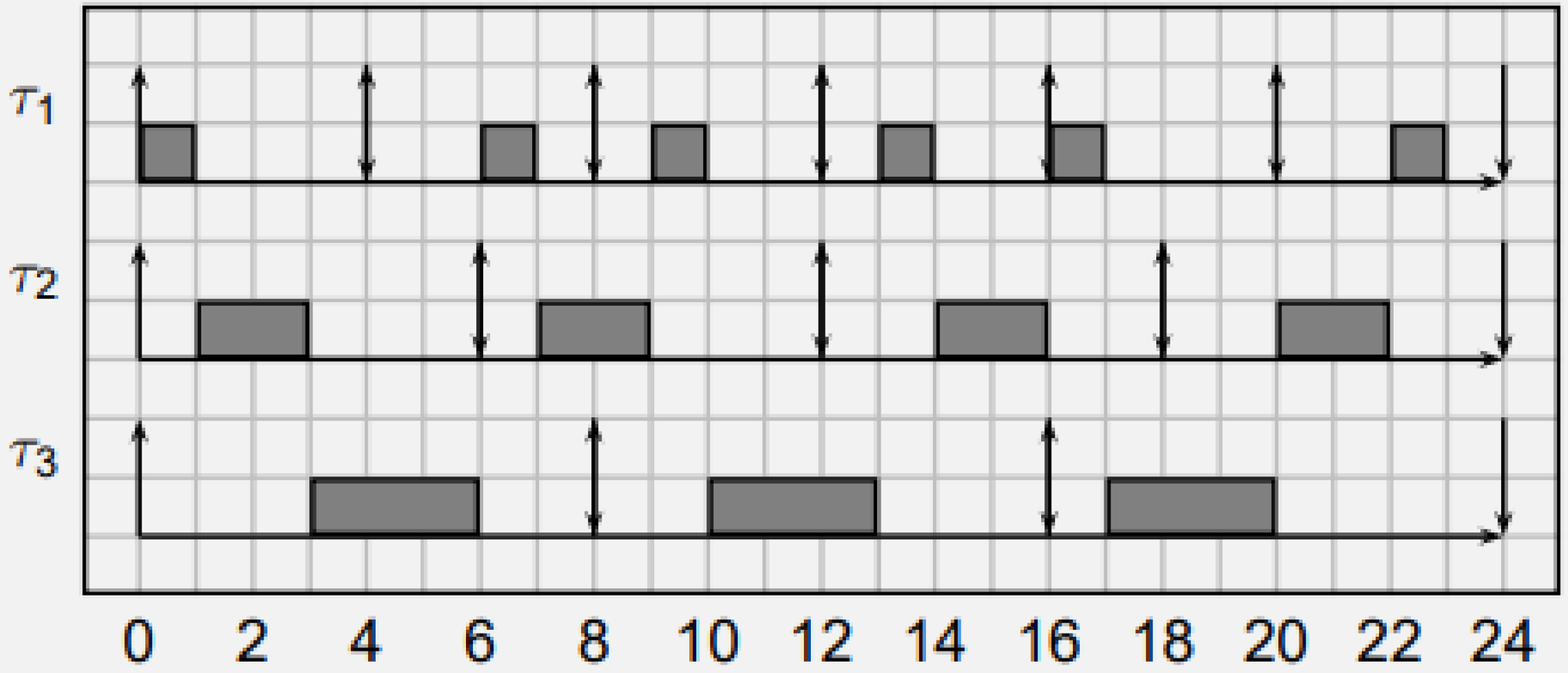
Principe

- Une activation de tâche réveille l’ordonnanceur
- Il choisit la tâche éligible de plus proche échéance

Test d’ordonnancement : Condition nécessaire et suffisante : $\Sigma Ci / Ti \leqslant 1$

L’EDF : comment ça marche ?

- Chaque tâche reçoit une priorité qui augmente à mesure que son échéance approche.



L’EDF : comment ça marche ?

Intervalle de temps	Deadlines des Jobs en attente	Tâche exécutée	Justification EDF
[0, 1]	T1: 4, T2: , T3: 8	**T1**	Deadline la plus proche (4).
[1, 3]	T2: , T3: 8	**T2**	Deadline la plus proche (6).
[3, 6]	T3: 8	**T3**	Seule tâche en attente.
t=4	**Nouveau job T1 arrive** (Deadline à 8)		
[6, 7]	T1: 8, T3: 8	**T1**	**Observation clé** : Deadline de 8 pour les deux.T1 est choisie, probablement à cause de la **priorité par défaut** (la plus petite indice) en cas d'égalité, ou T1 a la plus courte durée restante.
t=6	**Nouveau job T2 arrive (Deadline à 12)**		
[7, 8]	T3: 8, T2: 12	**T3**	Deadline la plus proche (8).

Optimalité de l’algorithme EDF

Le Théorème de Dertouzos (1973)

Si un ensemble de tâches est ordonnançable (satisfaisant toutes ses échéances) par n'importe quel algorithme, alors il est nécessairement ordonnançable par EDF.

- **Optimalité :** Ce théorème prouve qu'EDF est un algorithme optimal pour l'ordonnancement de tâches (jobs) préemptives sur un seul processeur.

- **Implication :** Si un ensemble de tâches ne peut pas être ordonnancé par EDF, cela signifie qu'il n'existe aucun algorithme, fixe ou dynamique, qui pourrait le faire. EDF est le "meilleur" algorithme possible.

Les avantages de l’algorithme EDF

EDF est un algorithme dominant !

L'algorithme EDF est généralement considéré comme supérieur à tout algorithme d'ordonnancement à Priorité Fixe (FP), comme Rate Monotonic (RM), principalement en raison de sa capacité à maximiser l'utilisation du processeur et de sa souplesse.

Il peut ordonnancer tous les ensembles de tâches qui peuvent l'être par les algorithmes à priorité fixe, mais pas l'inverse

Les avantages de l’algorithme EDF

EDF favorise une utilisation maximale du processeur

- EDF est le seul algorithme (parmi les algorithmes dynamiques simples) capable d'ordonnancer tout ensemble de tâches dont l'utilisation du processeur atteint U = 100 %, à condition que toutes les échéances soient respectées.

- En comparaison, les algorithmes FP ne peuvent garantir l'ordonnançabilité que jusqu'à un seuil d'utilisation inférieur à 100 % (ex: 69,3% pour RM dans le cas général).

Les avantages de l’algorithme EDF

Insensibilité aux décalages

- Les décalages (l'instant où la première instance d'une tâche arrive) compliquent énormément l'analyse et l'affectation des priorités dans les systèmes à priorité fixe. Un schéma de priorité optimal sans décalage peut devenir sous-optimal avec des décalages.

- EDF n'est pas affecté par les décalages, car il se concentre uniquement sur l'échéance absolue des jobs actifs, quel que soit leur point de départ.

Les limites de l’algorithme EDF

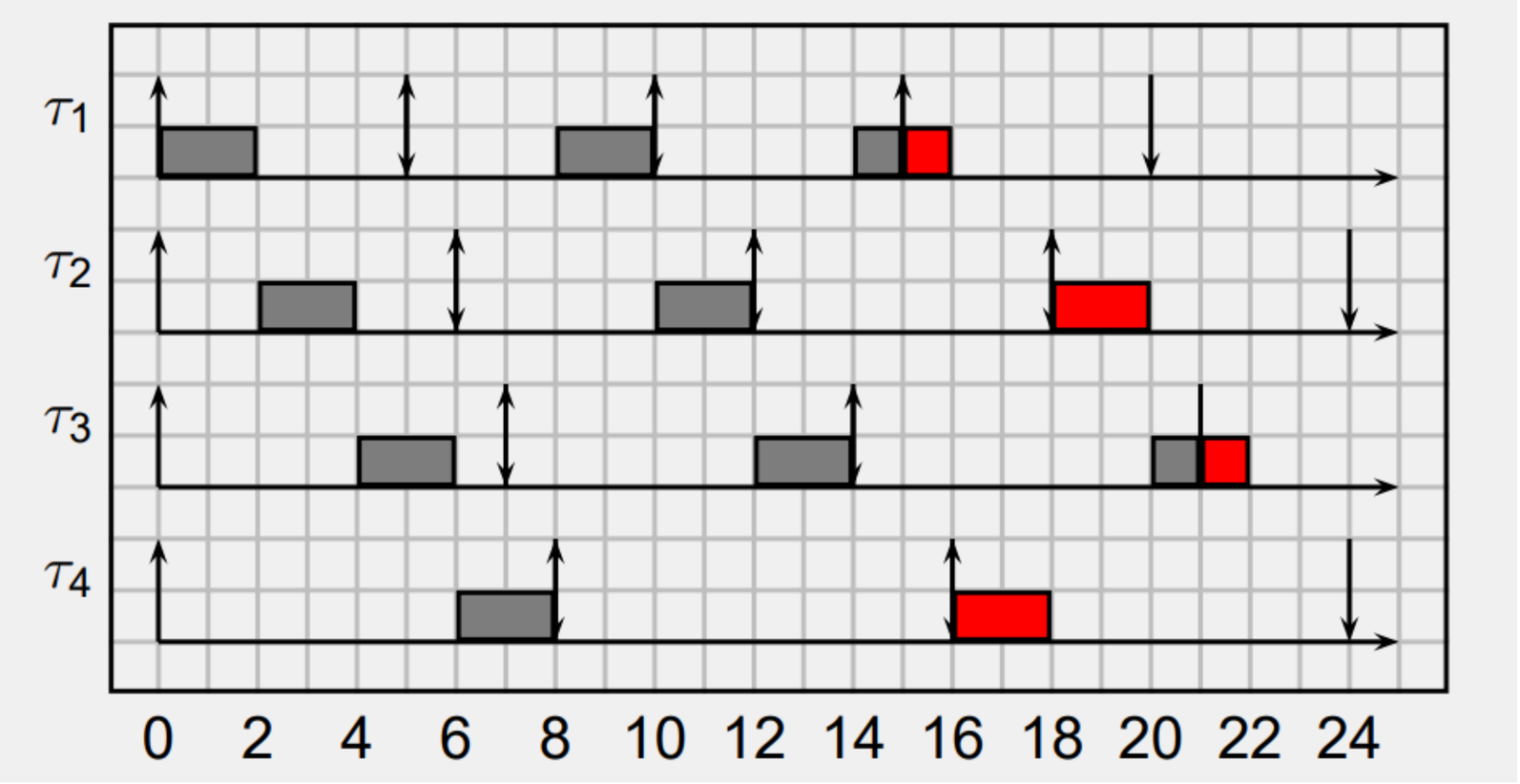
En EDF : La priorité d'une tâche change constamment. Le temps de réponse d'un job de J1 dépendra de la densité des échéances des autres tâches au moment de son arrivée. Un job de J1 pourrait avoir une longue attente si de nombreux jobs de J2 ou J3 ont des échéances plus proches que lui. Cette variabilité réduit la prévisibilité du système.

Si le système est en surcharge, même brièvement, EDF commence à manquer des échéances de manière imprévisible, souvent en laissant passer l'échéance de la tâche qui a la priorité la plus haute (l'échéance la plus proche), et ensuite de proche en proche. Le comportement est chaotique.

Implémenter EDF nécessite de maintenir une file d'attente triée par échéance absolue, qui doit être mise à jour à chaque arrivée de job ou à chaque préemption. Cette gestion est plus complexe et demande plus de ressources (calcul) qu'une file d'attente à priorité fixe simple utilisée par FP.

Les limites de l’algorithme EDF

L’effet domino



- Vers t = 14–16, une tâche (τ_1) termine trop tard (en rouge)
- Cela retarde τ_2, puis τ_3, puis τ_4
- On observe une propagation horizontale du retard

Comparaison EDF vs RATE MONOTONIC

Critère	EDF	Rate Monotonic
Type de priorité	Dynamique	Fixe
Taux d'utilisation max	100%	~69-82%
Optimalité	Optimal	Optimal parmi les algo à priorité fixe
Prévisibilité	Faible (temps de réponse variable)	Haute (temps de réponse stable)
Surcharge	Comportement catastrophique	Dégradation gracieuse
Complexité impl.	O(log n)	O(1)
Sensibilité aux offsets	Insensible	Très sensible

Extensions et variantes d’EDF

- **EDF-DS (Deadline Scheduling) :** Pour deadlines arbitraires (D ≠ T)
- **RUN (Reduction to Uniprocessor) :** Extension d'EDF pour multiprocesseurs
- **CBS (Constant Bandwidth Server) :** EDF + contrôle d'admission pour gérer la surcharge
- **EDF-VD (Value Density) :** Combinaison valeur/deadline pour maximiser la valeur en surcharge

Conclusion

L’algorithme EDF se distingue par son efficacité et son optimalité dans l’ordonnancement des tâches temps réel sur un processeur unique. En adaptant dynamiquement les priorités selon les échéances, il maximise l’utilisation du processeur tout en garantissant le respect des contraintes temporelles. Cependant, sa complexité de mise en œuvre et son comportement imprévisible en cas de surcharge rappellent la nécessité d’une planification rigoureuse dans les systèmes critiques.

Merci !