Ordonnancement de graphes orientés acycliques (DAG) sur des systèmes multiprocesseurs

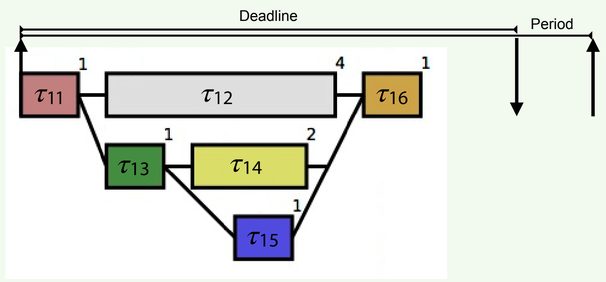
1. Systèmes temps réel

Un système temps réel doit traiter l'information et produire un résultat correct dans un délai contraint, sinon le système risque des conséquences et des échecs.

Contraintes de temps : échéance, période, gigue d'activation, offset (décalage).

Applications : systèmes de transport et avionique, services de communication, …

2. Modèle de tâches : DAG



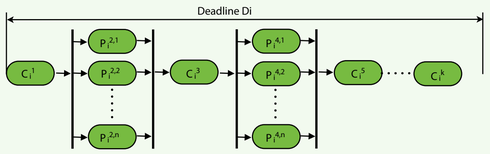
Il y a deux types de parallélisme dans le modèle DAG :

- *le parallélisme inter-jobs/tâches* dû aux dépendences entre les jobs/tâches,
- *le parallélisme intra-jobs/tâches* dû au parallélisme intrinsèque des jobs/tâches.

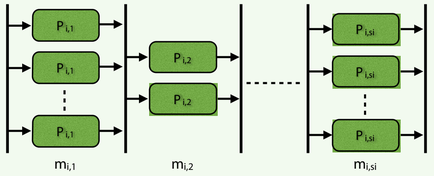
3.1. Ordonnancements utilisant des transformations DAG

Objectif : transformer les DAG dans d'autres modèles de tâches indépendentes pour utiliser les algorithmes d'ordonnancement classiques de tâches séquentielles indépendentes, telles que DM, EDF, LLF, …

Problème : la transformation utilise des approximations et produit des pertes de généralité du modèle initial.



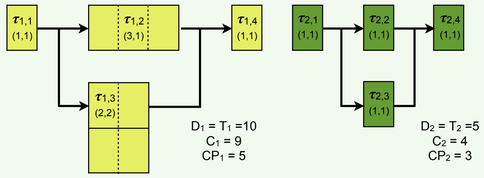
Modèle Fork/Join



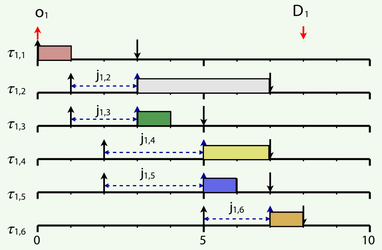
Modèle *Multi-threaded Segment*

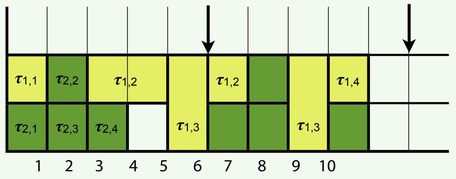
3.2. Ordonnancement direct de DAG

Objectif : ordonnancer les tâches DAG directement sur les systèmes multiprocesseurs et adapter les algorithmes et les tests d'ordonnancement pour considérer les dépendences entre les sous-tâches.



Problème :
les algorithmes sont plus difficiles à analyser : ils nécessitent de connaître de nouveaux paramètres temporels.





Un exemple d'ordonnacement de DAG utilisant l'algorithme LLF

Contact : Manar Qamhieh, Serge Midonnet, Laurent George

LIGM, avril 2014