Ordonnancement temps réel multiprocesseur et réparti

F. Singhoff

10 septembre 2018

Exercice 1 : ordonnancement global vs partitionné

Un ingénieur souhaite tester une configuration de 5 tâches définies par les paramètres suivants :

Tâche	Capacité	Période
T1	3	6
T2	2	24
T3	2	10
T4	1	6
T5	2	5

Les échéances sont égales aux périodes. La date de première activation de toutes les tâches est identique et est égale à zéro. On souhaite appliquer un ordonnancement Rate Monotonic préemptif. L'ingénieur cherche à ordonnancer ce jeu de tâches sur une architecture comportant deux processeurs identiques. Il hésite à utiliser soit une approche par partitionnement, soit une approche d'ordonnancement global.

Question 1

L'ingénieur commence par appliquer une méthode par partitionnement. On suppose que T3 et T5 sont placées sur un processeur et que les autres tâches sont placées sur le second. Calculer le chronogramme d'ordonnancement sur les 24 premières unités de temps. A partir du chronogramme, lire les pires temps de réponse de chaque tâche.

Question 2

Dans un deuxième temps, l'ingénieur applique une approche d'ordonnancement global. Donnez le chronogramme d'ordonnancement sur les 24 premières unités de temps. Les migrations de tâches sont autorisées à tout instant. A partir du chronogramme, lire les pires temps de réponse de chaque tâche.

Question 3

Pour ce jeu de tâches, quelle est la méthode qui minimise les temps de réponse ? De façon générale, c'est-à-dire, pour tout ensemble de tâches périodiques similaire au jeu de tâches définies ci-dessus, quelle est l'approche d'ordonnancement optimale ?

Exercice 2 : ordonnancement global

Un ingénieur souhaite valider un jeu de tâches défini de la façon suivante :

Tâche	Capacité	Période
T1	2	5
T2	8	20
T3	2	25
T4	3	6

Les échéances sont égales aux périodes. La date de première activation de toutes les tâches est identique et est égale à zéro. L'ingénieur souhaite appliquer un ordonnancement préemptif à priorités fixes et affecte les priorités selon l'algorithme Rate Monotonic.

Question 1

Sans calculer l'ordonnancement, montrez que ce jeu de tâches ne peut être exécuté sur un seul processeur.

Question 2

Vérifier que des échéances sont effectivement manquées en calculant l'ordonnancement de ce jeu de tâches pendant les 20 premières unités de temps.

Question 3

Afin de respecter les échéances, l'ingénieur souhaite appliquer un ordonnancement global sur deux processeurs identiques. On suppose que les migrations ne peuvent intervenir qu'à la terminaison d'un travail périodique : les tâches ne peuvent plus migrer lorsqu'elles ont commencé l'exécution d'un travail périodique. Elles ne peuvent donc migrer qu'entre deux réveils périodiques. Calculer l'ordonnancement de ce jeu de tâches sur les 20 premières unités de temps.

Question 4

On souhaite regarder si l'application d'une stratégie différente de migration permet encore de réduire les temps de réponse. On suppose maintenant que les migrations peuvent intervenir à n'importe quel instant. Calculer l'ordonnancement de ce jeu de tâches sur les 20 premières unités de temps. Comparer les ordonnancements calculés dans les questions 3 et 4. Quelle est la stratégie de migration avec ce jeu de tâches qui minimise les temps de réponse ? Peut on généraliser à tout ensemble de tâches périodiques similaires au jeu de tâches défini ci-dessus ? Justifier la réponse.

Exercice 3 : ordonnancement de messages périodiques

Dans les exercices suivants, nous supposons connu l'ordonnancement des messages périodiques. Dans cet exercice, nous regardons comment cet ordonnancement peut être calculé.

On rappelle que le temps de communication d'un message est constitué des éléments suivants (cf. [COT 00]) :

- Le délai de traversée des couches (fixe).
- Le délai de transmission (variable car dépendant de la quantité d'information à transmettre).
- Le délai de propagation (variable car dépendant de la distance à parcourir).
- Le délai d'accès qui dépend du protocole MAC (variable).
- Le délai de réception (fixe).

Message	Période (en ms)	Taille (en octets)	FIP (en μs)	CAN (en μs)
M1	5	2	100	100
M2	10	4	200	120
M3	10	6	250	150
M4	20	8	400	200

Dans cet exercice, on suppose que les délais de propagation, de réception et de traversée des couches du réseau sont négligeables. Les délais de transmission sont données dans le tableau ci-dessus dans les cas oú un réseau FIP et un CAN sont utilisés. La taille des messages est donnée à titre indicatif.

Question 1 : application au bus de terrain CAN

On rappelle qu'un bus CAN [UPE 94] utilise le protocole CSMA/CA (évitement de collision). Chaque message est identifié uniquement. L'identificateur constitue en fait une priorité d'accès au médium. L'arbitrage s'effectue grâce à une émission bit à bit de l'identificateur (cf. bit dominant et bit récessif). En cas de contention, les coupleurs émettant un message de plus faible priorité passent en mode lecture.

1. Le protocole CSMA le plus courant est celui utilisé par Ethernet (CSMA/CD). Quelles sont les différences entre CSMA/CD et CSMA/CA en terme de fonctionnement et de délais de communication.
2. Proposer une solution permettant d'ordonnancer les messages. Les messages peuvent il respecter leurs contraintes temporelles ?
3. Déterminer le temps d'accès au réseau pour chaque message. En déduire le temps de communication des messages.

Question 2 : application au bus de terrain FIP

L'allocation du médium dans un réseau FIP [CIA 99] est réalisé par un arbitre centralisé. L'arbitre possède un fonctionnement cyclique. Chaque cycle est constitué :

— D'une phase d'émission des informations périodiques (nommées variables dans FIP).

— D'une phase d'émission des informations apériodiques.

Pour la phase cyclique l'arbitre exploite une table construite hors-ligne. L'unité de base utilisée dans cette table est la période minimale (appelé microcycle) du jeu de messages périodiques. La taille de la table est égale au PPCM des périodes des messages (macrocycle).

1. Donner la valeur du microcycle et du macrocycle.
2. Proposer une table d'arbitrage permettant d'ordonnancer correctement les différents messages. Les messages peuvent il respecter leurs contraintes temporelles ?
3. A quelle technique d'ordonnancement s'inspire la technique de la table d'arbitrage.
4. Quel(s) algorithme(s) aurions nous pu appliquer pour construire automatiquement cette table.
5. Déterminer le temps d'accès au réseau pour chaque message. En déduire le temps de communication des messages.

Exercice 4 : prise en main d'un outil de simulation

L'objectif de cet exercice est de vous présenter un outil de simulation qui automatise certaines des techniques vues en cours. En particulier, il fournit un certain nombre d'ordonnanceurs de base accompagnés de leurs tests de faisabilité.

L'outil peut être récupéré à cet endroit :

`http://beru.univ-brest.fr/svn/CHEDDAR/trunk/releases/`

Une fois l'outil lancé, on vous demande de le tester :

1. Ajouter un processeur qui utilise un ordonnanceur **POSIX.1003 Highest Priority First Protocol** grâce au menu *"Edit/Hardware/Core"*.
2. Créer 3 tâches grâce au menu *"Edit/Software/Task"*. Les paramètres de ces tâches sont : $S_1 = S_2 = S_3 = 0, P_1 = 30, C_1 = 6, P_2 = 5, C_2 = 3, P_3 = 10, C_3 = 2$. Les délais critiques sont égaux aux périodes. N'oubliez pas de déterminer un niveau de priorité selon la période des tâches : on souhaite appliquer la méthode d'affectation de priorité Rate Monotonic. Avec cet outil, les niveaux de priorité vont de 1 à 255 (255 est le niveau de priorité le plus élevé).
3. Ordonnancer le jeu de tâches en appuyant sur l'icone *"Scheduling simulation"*.
4. Appliquer les tests de faisabilité en appuyant sur l'icone *"Scheduling feasibility"*.

Exercice 5 : temps de réponse de bout en bout

On étudie le système constitué de deux processeurs. Le processeur a héberge les tâches $T1$ et $T2$. Le processeur b héberge les tâches $T3$, $T4$ et $T5$. Certaines tâches communiquement par échange de messages périodiques :

— La tâche $T1$ émet le message périodique $M1$ à destination de $T3$.

— La tâche $T5$ émet le message périodique $M2$ à destination de $T2$.

Les tâches et messages sont définis par les paramètres suivants :

Message	Période (en ms)	Temps de communication (en ms)
M1	120	3
M2	50	1

Notez que la priorité 3 est le niveau de priorité le plus fort et 1 le niveau de priorité le plus faible. Le temps de communication inclut les temps de propagation, d'accès, de transmission et de traversée des couches. Le temps de réponse final d'un message i est obtenu par $TR(M_i) = J_i +$ temps de communication de i.

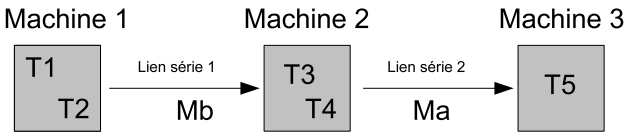
Tâche	Période (en ms)	Capacité (en ms)	Priorité	Processeur
T1	120	3	1	a
T2	50	2	2	a
T3	120	7	2	b
T4	70	1	1	b
T5	50	3	3	b

Le temps de réponse des âches est calculé avec la méthode de Joseph et Pandia étendue avec les *Jitters*.

On souhaite calculer les délais de bout en bout de chaque tâche du système. Pour ce faire, nous allons appliquer la méthode Holistique :

1. On pose $\forall i : J_i = 0$. Quels sont les temps de réponse des tâches et des messages ?
2. Pour quels tâches/messages doit on modifier le paramètre *Jitter*. Donner les valeurs du *Jitter* pour les tâches/messages concernés.
3. Recalculer les temps de réponse aprés avoir modifié les *Jitter*. Quels sont les nouveaux temps de réponse ?
4. Réitérer les opérations 3 et 4 jusqu'à convergence des temps de réponse. Quels sont les délais de bout en bout des chaines de traitement T1/T3 et T5/T2 ?

Exercice 6 : temps de réponse de bout en bout



On vous demande d'assister un ingénieur qui doit analyser une chaine de production de casseroles. Cette chaine est constituée de trois tâches, hébergées sur 3 machines (cf. figure ci-dessus) :

1. Dans la première machine (machine 1), la tâche T1 produit par emboutissage les fonds de casseroles. 2. Puis, la tâche T3 de la seconde machine (machine 2) soude les manches aux fonds de casseroles. 3. Enfin, la dernière machine (machine 3) emballe le produit terminé. Ce dernier traitement est effectué par la tâche T5.

Sur les machines hébergeant T1 et T3, d'autres tâches existent, mais elles ne participent pas à la fabrication des casseroles. Enfin les trois machines sont connectées par deux liens séries. Les liens séries ne sont pas partagés, ainsi : 1. Le lien série entre la machine 1 et la machine 2 est uniquement destiné à transmettre le message Mb. Ce message est expédié lorsque T1 termine son exécution afin d'avertir la tâche T3 qu'elle peut commencer son exécution. 2. Le lien série entre la machine 2 et la machine 3 est uniquement destiné à transmettre le message Ma. Ce message est expédié lorsque T3 termine son exécution afin d'avertir la tâche T5 qu'elle peut commencer son exécution.

L'ingénieur souhaite connaître le pire temps de fabrication d'une casserole : c'est à dire le délai entre le réveil de la tâche T1 et la terminaison de la tâche T5. On vous demande de l'aider.

Question 1 :

Il est possible d'utiliser l'approche holistique de Tindell pour déterminer le pire temps de fabrication des casseroles. Expliquez à l'ingénieur comment fonctionne la méthode holistique.

Question 2 :

Vous avez effectué quelques mesures sur la chaîne de production et vous en avez déduit que :

1. Les messages Ma ont un temps de communication inférieur à 1 seconde. Il en est de même pour Mb. Ces temps de communication incluent les temps de transmission sur le lien série, les temps de propagation ainsi que les temps de traversée des couches logicielles et matérielles.
2. Les tâches sont caractérisées par le tableau ci-dessous. Notez que la priorité 1 est le niveau de priorité le plus fort et 2 le niveau de priorité le plus faible.

Tâche	Priorité	Capacité	Période
T1	1	12	500
T2	2	4	500
T3	2	6	500
T4	1	20	500
T5	1	100	500

Appliquez l'approche holistique au jeu de tâches ci-dessus afin de calculer le pire temps de production d'une casserole, c'est à dire le pire temps de réponse de T5.

Question 3 :

Une autre solution classique pour relier les trois machines consiste à les connecter par un bus de terrain. Le bus est alors partagé par les messages Ma et Mb. Quelle est l'implication de ce changement d'architecture sur l'analyse holistique effectuée dans la question 2.

Références

Ces exercices sont inspirés de [COT 00]. Exercices complémentaires : [COT 00, KRI 97].

[CIA 99] CIAME. *Réseaux de terrain*. Edition Hermès, 1999.

[COT 00] F. Cottet, J. Delacroix, C. Kaiser, and Z. Mammeri. *Ordonnancement temps réel*. Hermès, 2000.

[KRI 97] C. M. Krishna and K. G. Shin. *Real-Time Systems*. Mc Graw-Hill International Editions, 1997.

[UPE 94] P. Upendar and P. J. Koopman. « Communication Protocols for Embedded Systems ». *Embedded Systems Programming*, 7(11) :46–58, November 1994.