Projet du cours *Informatique pour le Génie Industriel* Optimisation de la consommation énergétique d'un système informatique embarqué

École des Mines de Nancy 2A

Frédéric Sur
frederic.sur@loria.fr
www.loria.fr/~sur/enseignement/projetIGI/

1 Contexte

Les systèmes informatiques embarqués (sur une sonde spatiale, une automobile...) nécessitent l'utilisation de processeurs de plus en plus puissants, tout en restant limités par les ressources énergétiques disponibles. Dans ce cadre, l'industrie propose des processeurs spécifiques pouvant adapter leur fréquence de fonctionnement, et donc leur consommation énergétique, de manière dynamique. Un des problèmes rencontrés est de régler la fréquence du processeur au cours du temps, de manière à minimiser la consommation énergétique du système, mais sous la contrainte que les tâches à exécuter doivent se terminer à une échéance donnée. Fixer une échéance d'exécution est nécessaire pour des systèmes fonctionnant en temps réel. Le but de ce projet est d'implanter et comparer différents algorithmes permettant de résoudre ce problème sous une forme simplifiée.

2 Modalités pratiques du projet

- Travail de programmation VB.net personnel ou par groupe de deux, au choix (merci de me préciser le choix dès les premières séances).
- Évaluation sur la base d'un rapport écrit de quelques pages, décrivant les algorithmes étudiés et la modélisation adoptée. Vous comparerez les performances relatives des différents algorithmes (en terme de consommation énergétique de la solution trouvée et de temps de recherche de cette solution) et veillerez à présenter clairement vos conclusions. N'attendez pas la fin du projet et rédigez au fur et à mesure de la progression. Une démonstration sur machine sera également à prévoir.

3 Modélisation

On suppose que le système embarqué considéré nécessite l'exécution de n tâches à un instant donné. La tâche T_i (i variant entre 1 et n) a besoin pour s'exécuter de c_i cycles de processeurs, et doit être exécutée avant une date (échéance) d_i (exprimée en *secondes*).

Le processeur a une vitesse de travail (exprimée en *nombre de cycles par seconde*, il s'agit d'une fréquence) variant entre F_{min} et F_{max} par pas de F_{pas}. Une seule tâche peut s'exécuter à la fois sur le processeur. On notera $F_j = F_{min} + (j-1) * F_{pas}$ pour j variant entre 1 et m.

On supposera de plus que les tâches s'exécutent à une vitesse constante (*i.e.* la fréquence du processeur n'évolue pas au cours de l'exécution d'une tâche donnée).

Chaque tâche a un profil de consommation indiquant sa consommation selon la vitesse à laquelle elle est exécutée. Cette consommation dépend de la vitesse du processeur mais aussi de la nature de la tâche : un accès au disque dur consomme plus d'énergie qu'un calcul à stocker en mémoire vive par exemple. On dispose

donc d'un tableau $(e_{i,j})$ où $e_{i,j}$ est la consommation énergétique de la tâche T_i lorsqu'elle est exécutée à la fréquence F_j.

On suppose que le système demande l'exécution de ces n tâches à l'instant $t = 0$. On supposera dans la suite que les tâches sont ordonnées selon l'ordre croissant de leur date d'échéance d_i (ordre EDF, Earliest Deadline First). On ne s'intéressera donc pas dans ce projet au problème d'ordonnancement à proprement parler. Tout le problème est d'affecter des fréquences F_j aux tâches T_i de manière à minimiser l'énergie totale consommée tout en respectant les contraintes d'échéance d_i.

Étape 1 (Modélisation mathématique)
Modélisez le problème dans le cadre de la programmation linéaire.
On introduira des variables booléennes.

4 Implantation en Visual Basic.net

Vous implanterez les algorithmes des sections 5 (résolution exacte) et 6,7,8 (heuristiques) sous VB.net.

Étape 2 (Interface)
Vous pouvez commencer par construire une interface graphique avec des boutons pour chacun des algorithmes et des fenêtres pour visualiser les données ainsi que la solution trouvée.

Dans un premier temps, vous comparerez la performance de vos algorithmes sur le jeu de données suivant.

- Paramètres des tâches c_i et d_i, et tableau des consommations en énergie $(e_{i,j})$:

T_i	c_i	d_i
1	6	3
2	6	4
3	10	4
4	11	7
5	13	7

T_i	F_1	F_2	F_3	F_4	F_5
1	301	375	455	537	620
2	376	469	569	671	775
3	414	516	626	738	852
4	434	542	657	776	895
5	361	451	546	644	744

- Vitesses possibles pour le processeur (en cycles/seconde) :
$F_1 = 1,\ F_2 = 3,\ F_3 = 5,\ F_4 = 7,\ F_5 = 10.$

5 Résolution exacte

Dans le cas où il y a peu de tâches et peu de paliers (n et m petits), on peut envisager de chercher les solutions optimales de manière exhaustive.

Étape 3 (Résolution exacte)
Implantez cette recherche exhaustive.

On peut également se dispenser d'explorer l'intégralité des solutions réalisables à l'aide de divers algorithmes. On n'implantera pas ce type de résolution en VB.net dans ce projet.

Étape 4 (Facultatif)
Trouvez les solutions optimales à ce problème de programmation linéaire en variables booléennes (ou *bivalentes*) à l'aide d'un logiciel installé à l'école.

6 L'algorithme du « palier constant »

On peut chercher une solution sous-optimale au problème en supposant que toutes les tâches sont exécutées à la même vitesse. Le problème se réduit donc à trouver la valeur optimale de cette vitesse.

Étape 5 (Palier constant)
Proposez et implantez un algorithme permettant de trouver une vitesse constante de processeur minimisant l'énergie consommée.

7 L'algorithme de la « descente en cascade »

On commence par fixer toutes les tâches à la vitesse maximale du processeur. Les tâches sont ensuite classées par ordre décroissant de « saut d'énergie » (égal à la diminution d'énergie provoquée par la diminution de la vitesse du processeur d'un palier). Si cette transformation ne permet pas de respecter les contraintes du problème, alors elle est rejetée. On arrête l'algorithme lorsqu'aucune modification n'est plus possible.

Étape 6 (Descente en cascade)
Implantez cet algorithme.

8 Recuit simulé

L'algorithme du recuit simulé est un classique de l'optimisation combinatoire.

Étape 7 (Recuit simulé)
Proposez une formulation du recuit simulé pour ce problème, et implantez-là. Vous testerez différentes méthodes d'initialisation de la recherche, et l'influence des paramètres de l'algorithme.

9 Question ouverte

Étape 8 (Initiative personnelle)
Proposez votre propre algorithme, et comparez-le aux algorithmes précédents.

10 Jeu d'essai

Vous expérimenterez également vos algorithmes sur le jeu d'essai disponible sur la page web du projet. Il faudra donc écrire une procédure de lecture de ce fichier. Ce jeu de données correspond à un problème à 20 tâches, pour un processeur à 8 paliers de vitesse.

Le format des données est le suivant :

- Tout d'abord une ligne de 8 entiers pour les 8 fréquences F_j de fonctionnement que peut prendre le processeur (en *cycles par seconde*).
- Ensuite 20 lignes où figurent deux valeurs : le nombre de cycle c_i nécessaires à l'exécution de la tâche T_i et l'échéance d_i (en *secondes*) à laquelle cette tâche doit être terminée. Les tâches sont déjà classées par échéances d_i croissantes.
- Enfin 20 lignes de 8 valeurs pour la consommation de la tâche T_i lorsqu'elle est exécutée à la fréquence de processeur F_j.

11 Bibliographie

Ce projet est en partie inspiré de l'article suivant :

S. Jeannenot, P. Richard, F. Ridouard, *Ordonnancement temps réel avec profils variables de consommation d'énergie*, Real-Time Embedded Systems, 2004.

Attention, la modélisation proposée dans ce projet est sensiblement différente.