Documentation sur les impacts énergétiques du projet et de ses retombées

Valentin Laclautre, Anthony Dard, Damien Trouche, Martin Gangand, Basel Darwish Jzaerly

Contents

1 Introduction

Au cours de ce projet, nous avons été amené à évaluer l'impact énergétique de notre compilateur deca et de son implémentation. On peut distinguer trois étapes lors desquelles ce compilateur deca consomme de l'énergie. La première étape source de consommation est l'implémentation: le fait de coder ce compilateur demande d'utiliser des ordinateurs pendant une longue durée et de faire des tests. Tout cela a un impact non-négligeable. La deuxième étape est celle de la compilation du code. Il s'agit de transformer le langage deca en assembleur ima, transformation qui demande des calculs et donc de l'énergie. La dernière étape est l'exécution de l'assembleur sur une machine virtuelle ima. Dans ce rapport, nous allons nous intéresser plus précisément à la première et la troisième étape en essayant de quantifier cette consommation d'énergie et en proposant des pistes de solution pour la limiter. Nous avons volontairement choisi de ne pas nous attarder sur la consommation d'énergie due à la compilation car les optimisations que nous proposons ne sont pas orientées vers cette étape.

2 Consommation d'énergie durant l'implémentation

Pour commencer, penchons-nous de plus prêt sur notre consommation durant les trois semaines de développement.

2.1 Consommation due à l'utilisation des ordinateurs

Un ordinateur a besoin d'une certaine puissance électrique pour fonctionner. On estime à 100 W la puissance d'un ordinateur portable et à 200 W la consommation d'un ordinateur fixe. Lors de ce projet, nous avons utilisé cinq ordinateurs : un ordinateur fixe et quatre ordinateurs portables. On obtient donc une puissance consommée cumulée de 600 W. En faisant l'hypothèse assez réaliste que l'on travaille 8 heures par jour, on obtient une énergie consommée quotidienne de:

$$E_{quotidien} = 0,6 \times 8 = 4,8\ kWh \tag{1}$$

D'autre part, on estime le nombre de jours total de développement à 20 jours, ce qui donne :

$$E_{total} = 4,8 \times 20 = 96\ kWh \tag{2}$$

Cette énergie est comparable à celle consommée par un foyer français de deux personnes pendant 21 jours. Malheureusement, il n'est pas chose aisée de diminuer cette consommation car on ne peut pas travailler sans ordinateur.

2.2 Consommation due aux tests

Dans le but de couvrir tous les cas de compilation possibles, nous avons mis en place une base de test contenant environ cinq cents fichiers deca (tests pour l'extension compris). Nous les avons régulièrement exécutés pour suivre de la meilleure des manières l'avancée du développement. Mais la compilation et l'exécution des tests a aussi une consommation énergétique. Pour mieux s'en rendre compte, nous avons utilisé la commande "/usr/bin/time -v" en passant en argument la commande "mvn verify". Nous avons fait cette expérience sur 3 ordinateurs différents, voici les résultats:

- **Ordinateur de l'école**: Le processeur est un Intel Core i5 qui possède une puissance de dissipation thermique de 65 W[1]. 145% du CPU est utilisé (sachant que le maximum est de 600% car il s'agit d'un processeur 6 cœurs). On obtient alors une puissance consommée de :

$$P = \frac{145}{600} \times 65 = 15\ W \tag{3}$$

  Étant donné que le temps d'exécution est de 4:28, cela implique l'énergie consommée suivante :

$$E = P \times T = 15 \times 268 = 4020\ J = 1.11 \times 10^{-3}\ kWh \tag{4}$$

- **Ordinateur d'Anthony** : Il s'agit d'un ordinateur plutôt haut de gamme avec un processeur Core i7 avec une puissance de dissipation thermique de 15 W[2]. 206% du CPU est utilisé (sachant que le maximum est de 400% car il s'agit d'un processeur 4 cœurs). On obtient alors une puissance consommée de :

$$P = \frac{206}{400} \times 15 = 7.8\ W \tag{5}$$

  Étant donné que le temps d'exécution est de 3:34, cela implique l'énergie consommée suivante :

[1]PDT donné par le site d'Intel
[2]PDT donné par le site d'intel

$$E = P \times T = 7.8 \times 214 = 1669\ J = 4.63 \times 10^{-4}\ kWh \qquad (6)$$

- **Ordinateur de Damien**: Il s'agit d'un ordinateur de milieu de gamme, utilisé par l'intermédiaire d'une machine virtuelle, les ressources sont donc beaucoup plus faibles que l'ordinateur précédent. Le processeur est un Core i5 avec une puissance de dissipation thermique de 15 W[3]. 188% du CPU est utilisé (sachant que le maximum est de 400% car il s'agit d'un processeur 4 cœurs). On obtient alors une puissance consommée de :

  $$P = \frac{188}{400} \times 15 = 7.1\ W \qquad (7)$$

  Étant donné que le temps d'exécution est de 10:12, cela implique l'énergie consommée suivante :

  $$E = P \times T = 7.8 \times 612 = 4312\ J = 0,00120\ kWh$$

  (8)

Bien sûr, il ne s'agit ici que d'une estimation assez grossière car les conditions de calcul de la PDT restent assez obscure de la part d'Intel et il y a sans doute d'autres paramètres auxquels nous n'avons pas pensé. Néanmoins, on peut déduire plusieurs enseignements de cette petite expérience: pour un même programme, un processeur plus récent et haut de gamme comme le i7 consomme moins d'énergie qu'un processeur de milieu ou d'entrée de gamme comme le i5 car il peut réaliser un plus grand nombre d'opérations par secondes tout en conservant la même PDT. D'autre part, la consommation énergétique de l'exécution de l'ensemble des tests reste assez faible. Même en estimant que l'on a exécuté ces tests une quarantaine de fois au cours du projet, cela reste peut (moins de $0,01\ kWh$) en comparaison avec les valeurs calculées en première partie. Il est donc plus instructif de s'intéresser à un déploiement à grande échelle auprès des utilisateurs de notre compilateur deca. Pour cela, nous allons regarder de plus prêt certains cas d'usages.

[3]PDT donné par le site d'Intel

3 Consommation d'énergie due à l'exécution dans ima

3.1 Cas d'usage 1 : Téléchargement d'un fichier deca sur internet

Imaginons le cas où l'on cherche à télécharger un fichier exécutable IMA sur internet. Il y a alors un intérêt énergétique et d'efficacité à avoir le fichier le plus léger possible. En effet, le conseil américain pour l'efficacité énergétique (ACEEE) établit que le téléchargement d'un gigaoctet de données coûte en moyenne 5,12 kWh. On se rend bien compte que l'accumulation des téléchargements du fichier IMA aura une forte empreinte énergétique. Pour cela nous avons mis en place une technique d'optimisation afin de diminuer le nombre de lignes d'assembleur tout en conservant le même résultat. Cette technique se base sur la suppression de code mort. Le code mort est un code généré qui ne peut être atteint lors de l'exécution. Un code assembleur généré de façon naïve peut en contenir plusieurs lignes. Par exemple, cela peut provenir d'une condition trivialement vraie ou fausse dans un conditionnel. Cela est expliqué plus en détail dans la documentation de conception. Nous avons pris soin de supprimer automatiquement ces lignes afin que notre fichier IMA ne se limite qu'au code utile.

3.2 Cas d'usage 2 : Exécution d'un fichier gourmand en ressources

Un fichier IMA peut parfois demander une grosse puissance de calcul, il est donc important d'optimiser la façon dont ces calculs sont effectués pour ne pas rallonger inutilement l'exécution du code et ainsi consommer plus d'énergie. Pour cela, nous avons mis en place les optimisations suivantes:

- Les expressions arithmétique qui sont des constantes sont pré-calculées à la compilation.
- Lorsque l'on multiplie une variable entière par une puissance de 2, on réalise un décalage vers la gauche des bits de l'expression et inversement pour une division d'un entier par une puissance de 2. Lorsque le facteur dépasse 2^{20}, on repasse à une multiplication/division classique car cela coûte moins de cycles.

  Outre les économies d'énergie, ces optimisations nous permettent d'assurer une bonne performance au palmarès.

4 Conclusion

La création et l'utilisation d'un compilateur possède une empreinte énergétique non négligeable, que ce soit au cours du développement comme au cours de l'utilisation. Cependant comme nous l'avons vu dans la dernière partie, il existe des optimisations permettant à l'utilisateur de consommer moins d'énergie lors de l'exécution d'un programme généré par ce compilateur. Par la même occasion l'efficacité du code est augmentée, dans l'intérêt de l'utilisateur. La prochaine étape consiste alors à s'intéresser de plus prêt à la compilation du code deca pour jauger au mieux son impact énergétique et proposer des optimisations propres à cette partie-là.