Analyse de l'Impact Energetique du Compilateur Deca

Auteurs :
Mehdi El Idrissi El Fatmi
Khadija El Adnani
Alaa Jennine
Badr El Khoundafi
Yassine El Kouri

January 23, 2025

1 Introduction

Ce document présente l'analyse de l'impact énergétique du compilateur Deca en mesurant la consommation d'énergie et le temps d'exécution pour trois fichiers de test : `AgetX.deca`, `AddPrintDot.deca`, et `EstGrand.deca`. Le but est d'évaluer l'efficacité énergétique du compilateur.

2 Configuration du Système

Le travail a été réalisé sur un ordinateur avec les caractéristiques suivantes : **AMD Ryzen 7 8845HS avec Graphiques Radeon 780M**, 16 cœurs (8 cœurs physiques avec 2 threads par cœur), **Fréquence de mise à l'échelle du CPU** : 41%, **Boost de fréquence** : activé, et une architecture **x86_64**.

3 Fichiers de Code

Pour évaluer la consommation d'énergie et le temps d'exécution, trois fichiers Deca ont été utilisés. Le premier fichier, `AgetX.deca`, teste l'accès à un membre `x` d'une classe `A` via la méthode `getX()`. Le second fichier, `AddPrintDot.deca`, crée une classe `A` avec une méthode `m()` qui incrémente une variable `z`. Enfin, `EstGrand.deca` implémente un jeu de devinette où l'utilisateur doit trouver un nombre cible en un nombre limité d'essais. Ces trois fichiers sont représentés ci-dessous :

AgetX.deca

```
class A {
   protected int x ;
   int getX() {
      return x ;
   }
```

```
}
{
    A a = new A() ;
    println("a.getX() = ", a.getX()) ;
}
```

AddPrintDot.deca

```
class A {
    int z = 6;
    void m(int a){
        z = z + 1;
    }
}
{
   A a = new A();
   a.m(1);
   println("a.z = " , a.z);
}
```

EstGrand.deca

```
int target = 42; // Nombre cible
int maxTries = 5; // Nombre maximum d'essais
int guess = 0;
boolean guessed = false;
int tries = 1; // Initialisation du compteur d'essais

println("Devinez le nombre (entre 1 et 100) !");

while (tries <= maxTries && !guessed) {
    println("Essai " , tries , ": ");
    guess = 50; // Simule une entr e (modifiez pour tester)

    if (guess < target) {
        println("Trop petit !");
    } else if (guess > target) {
        println("Trop grand !");
    } else {
        println("Bravo ! Vous avez trouv  le nombre.");
        guessed = true;
    }

    tries = tries + 1; // Incr menter le compteur d'essais
}

if (!guessed) {
```

```
26      println("D sol , vous n'avez pas devin . Le nombre
        tait : " , target);
27  }
```

4 Script utilisé

Le script suivant a été utilisé pour exécuter le compilateur sur les fichiers de test et mesurer la consommation d'énergie ainsi que le temps d'exécution. Nous avons utilisé des couleurs et une présentation soignée pour rendre le code plus lisible.

```
#!/bin/bash

# Tableau des fichiers de test
files=("AgetX.deca" "AddPrintDot.deca" "EstGrand.deca")

# It rer sur chaque fichier
for file in "${files[@]}"
do
    echo "Testing $file..."
    # Commande pour mesurer la consommation d' nergie et le
    temps d'ex cution
    echo "Running perf and time for $file..."
    execution_time=$(time (perf stat -e power/energy-pkg/ ./
    decac $file) 2>&1 | grep real | awk '{print $2}')
    energy_consumption=$(perf stat -e power/energy-pkg/ ./
    decac $file 2>&1 | grep "power/energy-pkg" | awk '{print
    $1}')

    # Afficher les r sultats
    echo "Execution time for $file: $execution_time"
    echo "Energy consumption for $file: $energy_consumption
    Joules"
    echo "-------------------------------------"
done
```

Listing 1: Script pour mesurer la consommation d'énergie et le temps d'exécution

5 Résultats

Les résultats obtenus après l'exécution des tests sont les suivants :

- **AgetX.deca** :

  - Temps d'exécution : 0.161448259 secondes
  - Consommation d'énergie : 3.98 Joules

- **AddPrintDot.deca** :
  - Temps d'exécution : 0.167626053 secondes
  - Consommation d'énergie : 3.77 Joules

- **EstGrand.deca** :
  - Temps d'exécution : 0.174375199 secondes
  - Consommation d'énergie : 3.79 Joules

6 Analyse

Les résultats montrent que le compilateur Deca est relativement efficace avec des temps d'exécution similaires (autour de 0.16 à 0.17 secondes) et une consommation d'énergie modeste (autour de 3.77 à 3.98 Joules) pour ces fichiers de test.

Les différences entre les fichiers de test sont faibles, ce qui peut indiquer que les fichiers traités sont relativement simples en termes de taille ou de complexité.

7 Test de performance pour 71 fichiers

Nous avons effectué un test sur 71 fichiers en exécutant le script `test_codegen`. La commande utilisée pour mesurer les performances est la suivante :

```
perf stat -e power/energy-pkg/ ./test_codegen.sh
```

Les résultats obtenus sont les suivants :

Consommation d'énergie totale : 288.13 Joules
Temps d'exécution total : 11.172464472 secondes

Ces résultats montrent une consommation d'énergie de 288.13 Joules pour un temps d'exécution total de 11.17 secondes. Dans l'ensemble, ces résultats sont assez bons pour un grand nombre de tests exécutés simultanément. Cependant, des optimisations peuvent être envisagées pour réduire davantage la consommation d'énergie tout en maintenant ou en réduisant le temps d'exécution.

Analyse : Bien que la consommation d'énergie soit raisonnable pour 71 tests, il est toujours possible d'optimiser l'efficacité énergétique. Une analyse plus approfondie du code généré par le compilateur pourrait permettre de réduire l'empreinte énergétique. De plus, des techniques comme l'amélioration de la gestion des ressources matérielles lors de l'exécution des tests pourraient contribuer à des gains significatifs. Enfin, la configuration matérielle pourrait aussi influencer la consommation d'énergie, et des tests sur d'autres plateformes pourraient révéler des moyens supplémentaires pour réduire la consommation.

8 Conclusion

Le compilateur Deca présente des résultats satisfaisants en termes d'efficacité énergétique pour les fichiers testés. Toutefois, pour évaluer l'impact énergétique de manière plus approfondie, il serait utile d'effectuer des tests sur une gamme plus large de fichiers avec des caractéristiques variées, telles que des programmes plus volumineux ou plus complexes.