Optimisation du code par GCC

Evan VOYLES, Amaury RODRIGUEZ, Stefan GAŁKIEWICZ

19 janvier 2022

Notions de compilation

Il y a une particuliere puissance caché derrière les trois lettres gcc - signifiant à la fois la 'Gnu Compiler Collection' et la commande à lancer dans le terminal pour compiler un program C. Enfin pas que. Quand on appelle une commande de la forme

```
gcc -o hello hello_world.c
```

on lance effectivement une multitude de processus qui travaillent scrupuleusement en silence. Il s'agit de :

1. Préprocesser le fichier .c
2. Compiler les fichiers processés pour créer des fichier d'objet (.o)
3. Relier des fichiers d'objet dans une éxécutable

Ca veut dire quoi, préprocesser un fichier .c? En effet, à chaque fois qu'on écrit `#include <stdio.h>` pour inclure un fichier d'entête, avant la compilation un outil dit le préprocesser va remplacer une ligne d'include avec tout le contenu du fichier d'entête. Alors la directive préprocesseur `#include` consiste en une opération de copier et coller. Il y a plusieurs d'autres directives qui sont processées avant de compiler, par exemple le lecteur reconnaitra peut-etre les directives

```
#if, #ifdef, #ifndef, #else, #elif
```

qui sont employées pour la compilation conditionnel (par exemple, inclure un fichier spécifique pour Windows si le système est Windows) où bien pour éviter d'inclure le même fichier plusiers fois :

```
#ifndef MONFICHIER_H
#define MONFICHIER_H

/* Contenu du fichier monfichier.h */

...

#endif
```

Après que nos fichiers sont préprocessés, ils sont prêts pour être compilés. La compilation traduit des fichiers en C à des instructions de machine en binaire, dit des fichiers d'objet qui portent l'extension .o. Pour experimenter chez vous, on peut donner l'option `-c` à gcc pour arrêter après le compilation et créer des fichiers d'objet. Le dernier étape s'agit de regrouper tout les fichiers d'objet pertinents et de les emballer dans une seule exécutable. C'est quoi la différence concrète entre un 'program' C et une 'bibliothèque'? Une bibliothèque c'est une collection des fonctions et leur définitions tandis ce que un program contient la fonction spécial `main`, qui sert comme une porte d'entrée de l'exécution d'un program.

Alors finalement l'outil `ld`, dit le 'linker' en anglais, relie tous les fichiers .o dans un executable. Son travail est compliqué mais fondamentale. Grosso modo, `ld` trouve exactement où elle sont définies les fonctions extérieures qu'on appelle dans un program. Par exemple, pour un program simple HelloWorld on va utiliser la fonction `printf` qui est défini d'ailleurs. Au moment de créer l'exécutable, `ld` va chercher la définition de `printf` dans la bibliothèque standarde et mettre les instructions binaires dans l'exécutable. La magie de `gcc` c'est qu'il a fait tout cela pour vous en coulisse - un vrai emploi ingrat.

L'Optimisation

-O0, -O1, -O2, -O3

Your text goes here.

A subsection

More text. Even more text.
Here is playing around with some of greek symbols :

Sup boy