IUP MIAGE
Système d'exploitation

TP — Compilation parallèle

Ce TP illustre une utilisation des appels systèmes `fork ()`, `wait ()`, `waitpid ()`, et `exec ()` (avec ses variantes `execlp ()`, `execvp ()`).

Consultez les documents de TD, l'annexe de ce sujet et/ou les pages de manuels pour ces différentes primitives.

1 Compilation

La compilation d'un programme C produit un exécutable à partir d'un ensemble de fichiers source `.c`.

Cette compilation se réalise en deux étapes :

1. À partir de chaque fichier source, on produit un fichier objet `.o` en invoquant le compilateur C (`gcc` par exemple) avec l'option `-c` :

   ```
   gcc -c file.c
   ```

   Notons que l'ensemble des fichiers `.o` peuvent être produits indépendamment les uns des autres.
2. L'éditeur de liens (on utilise à nouveau `gcc`, cette fois sans option) produit un exécutable à partir de l'ensemble des fichiers objets `.o`.

2 Compilation en parallèle

Le programme qu'il vous est demandé d'écrire prend en paramètre une série de fichiers source, les compile chacun séparément et en parallèle puis édite les liens pour produire un exécutable.

Ce programme doit :
- lancer un processus fils pour chacun des noms de fichiers passés en paramètre ;
- chaque fils doit exécuter le programme `gcc -c` sur le fichier dont il s'occupe ;
- le père doit attendre la terminaison de tous ses fils ;
- si l'ensemble des fils ont terminés sans erreur, le père réalise l'édition de liens.

Veillez particulièrement
- à tester les valeurs de retour des appels systèmes et à utiliser la primitive `perror ()` ;
- à ce que votre programme retourne (`exit ()`) une valeur cohérente.

3 Compilation pas trop parallèle

Modifiez votre programme pour qu'il ne lance jamais plus de N (par exemple 10) processus simultanés.

4 Annexe

La fonction `execvp ()`

Id: tpcompilo.tex,v 1.8 2001/02/15 15:25:30 marquet Exp

```
#include <unistd.h>
int execvp (const char *file, char *const argv[]) ;
```

de la librairie se comporte comme l'appel système `execve ()` à la différence près qu'elle recherche l'exécutable `file` dans les répertoires fournis dans la variable d'environnement `$PATH`.