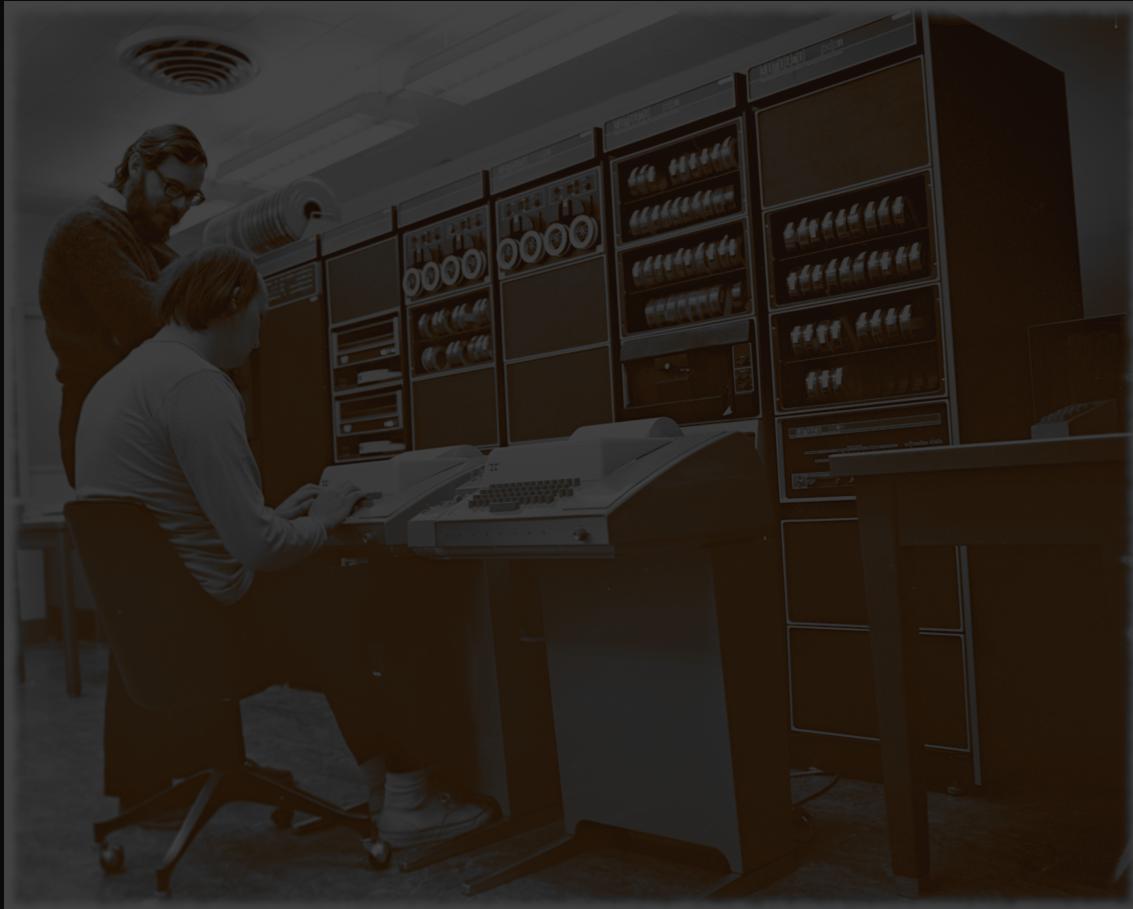


Systèmes d'exploitation

TP n° 1 : génération d'une courbe

Alain Lebret

2024-2025



Objectif

Mettre en oeuvre les processus à l'aide des appels système « fork() » et « exec() ».

Pré-requis : chapitre sur les processus

Durée estimée : 1 séance

Présentation

Le logiciel *Gnuplot* (<http://www.gnuplot.info>) permet de représenter graphiquement des données numériques simples sous forme d'un diagramme, d'une courbe ou encore d'un histogramme. Il est accessible en ligne de commande à partir du terminal :

```
$ gnuplot -persist commandes(gp)
```

où « commandes .gp » est le fichier de commandes utilisé par Gnuplot et l'option -persist permet à la fenêtre de visualisation de rester ouverte.

Voici un exemple de fichier de commandes permettant à Gnuplot de visualiser la courbe du sinus cardinal sur l'intervalle [0, 50] :

```
set samples 500
set title "Sinus cardinal"
set xlabel "x"
set ylabel "sin(x)/x"
set xrange [0:50]
set border
set grid
plot sin(x)/x
```

Travail à réaliser

On se propose de créer un programme de tracé de courbe. Ce programme attend en permanence que l'utilisateur indique une borne B permettant de définir l'intervalle de visualisation de la courbe à afficher (vous fixerez au choix un intervalle $[0, B]$ ou $[-B, B]$). Dès que la borne est entrée par l'utilisateur, le processus parent crée un fichier « commandes .gp », puis un processus enfant qui sera chargé d'exécuter *Gnuplot* pour afficher la courbe dans le nouvel intervalle. Une fois la courbe affichée, le processus parent se remet en attente d'acquisition d'une nouvelle borne.

1. Créez un dossier « tpo1 » dans votre dossier de travail « os » puis déplacez-vous dans celui-ci.
2. Codez le programme « iplot » qui réalise les opérations précédemment décrites. Celui-ci sera construit à partir des fichiers « iplot.c », « iplot.h » et « main.c ». L'appel système « execlp() » pourra être utilisé afin de lancer *Gnuplot*.



Afin de visualiser le mécanisme de couverture (*overlay*), ouvrez deux terminaux et dans l'un deux, entrez la commande « htop » qui affiche la charge processeur en temps réel. Dans « htop », filtrez les processus « iplot » et « gnuplot », puis ajoutez un appel à la fonction `sleep()` dans le processus enfant avant son appel à `execlp()`. Exécutez le programme et notez la création du processus enfant, puis au bout du temps imparti, sa disparition et l'apparition du processus « gnuplot ».

Question : l'enfant et *Gnuplot* ont-ils le même PID ?

Exercice complémentaire

Créez un programme qui crée quatre enfants. Ces quatre enfants utiliseront l'appel système « execvp() » pour lancer différents programmes.

- Le premier enfant exécutera « firefox » de manière à ce que la navigateur s'ouvre dans une fenêtre de taille 800x1200 et avec deux onglets : un pointant sur « <https://foad.ensicaen.fr> » et l'autre sur « <https://gitlab.ecole.ensicaen.fr> ».
- Le deuxième enfant exécutera aussi « firefox » de manière à ce qu'une recherche sur « Codeium » et « Vim » soit réalisée par le moteur de recherche par défaut du navigateur.
- Le troisième enfant exécutera « gedit » (l'éditeur par défaut) de manière à ce qu'il ouvre automatiquement un fichier de votre choix dont le nom aura été passé en argument lors de l'appel du programme.
- Le quatrième enfant exécutera le visualiseur de documents (« evince ») de manière à ce qu'il ouvre automatiquement ce fichier PDF.

Les options de Firefox peuvent être trouvée ici : <https://wiki.mozilla.org/Firefox/CommandLineOptions>.

Livrable

En fin de séance, déposer sur la plateforme Moodle une archive du dossier « tpo1 » :

- un fichier « README.md » indiquant :

- un titre et vos informations (nom, prénom, etc.) ;
 - les instructions pour compiler les programmes à l'aide de make (quelles cibles utiliser).
 - les instructions pour exécuter les éventuels tests et le programme.
- le(s) fichier(s) source(s) « .c » et « .h » ;
 - le fichier « Makefile » permettant de construire tout le projet.

Résumé

Dans ce TP vous avez mis en oeuvre plusieurs processus, que ce soit par clonage à l'aide de l'appel système fork() ou à l'aide du mécanisme de couverture avec la famille d'appels système exec().