PPCD Tarea 1

Cerritos Lira Carlos

29 de Octubre del 2020

2. Comandos en C para porcesos e hilos

Procesos

1. fork() - Crea un proceso hijo.
2. wait() - Bloque la ejecución del proceso padre hasta que uno de sus procesos hijos termine.
3. exit() - Termina la ejecución del prcoeso.
4. pipe(int fds[2])
   Crea un canal de comunicación entre procesos, para leer datos del canal se utilizara fd[0] y para escribir fd[1]

Hilos

1. pthread_create(pthread_t *thread, const pthread_attr_t *attr,void *(*start_routine) (void *), void *arg)
   Crea un nuevo hilo que ejecuta la función start_routine, donde los argumentos arg, son las variables que pasan a esta función.
2. pthread_join (pthread_t thread)
   Bloque la ejecuión del hilo actual hasta que el processo con id thread termine.
3. Los hilos comparten memoria por lo tanto no es necesario usar pipe() como en los procesos.

3. Conceptos multiprocessing

Global Interpreter Lock (GIL)

El Global Interpreter Lock o GIL, es un bloqueo que solo permite que un hilo tenga control en un interpretador de python.
Las desventajas de tener GIL son notorias cuando se ejecutan múltiples hilos.

Ley de Amdahal

La ley de Amdahal nos dice cuanto acelera la ejecución de un programa cuando los recurso del sistema mejoran.
En programación paralela es frecuentemente usada cuando el número de procesadores incrementa.

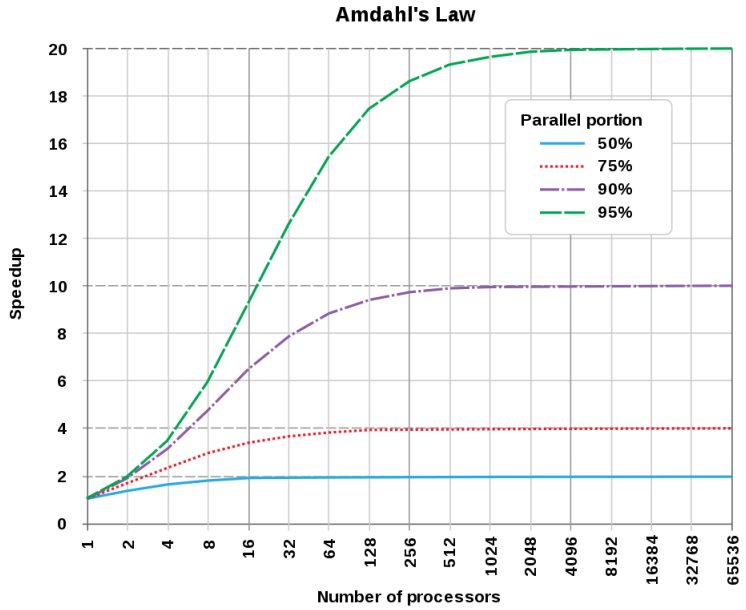

Figure 1: Ley de Amdahal aplicada a la paralelización de un programa

Multiprocessing

Multiprocessing es el uso de dos o más unidades de CPU en una computadora. El término tambien se puede referir a la habilidad del sistema de soportar mas de un proceso al mismo tiempo.

4. Creación de procesos en python

La creación de procesos en python es muy simple gracias a la librería multiprocessing. Las funciones principales son:

1. multiprocessing.Process(target, args)
   Crea un proceso donde target es la función que ejecutara y args las variables que se pasan a esta función.

2. multiprocessing.Process.start()
   Inicializa el procesos hijo.

3. multiprocessing.Process.join()
   Detiene la ejecución del proceso hasta que el proceso hijo termine.

Ejemplo

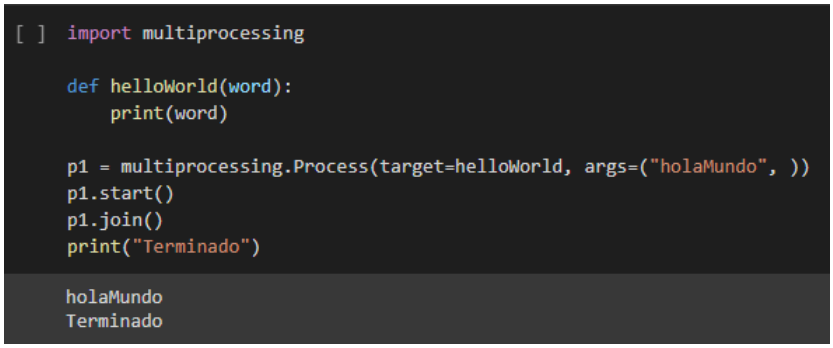

Figure 2: Ejemplo creación de proceso

Ahora bien, lo complicado de crear procesos es determinar la forma en la que se relacionan.

```python
def info():
    name = __name__
    cid = os.getpid()
    id = os.getppid()
    print('nombre proceso:', __name__)
    print('proceso padre:', os.getppid())
    print('proceso actual:', os.getpid())
    print('-----')

def solve(f, x):
    ans = f(*x)
    print('result: ', ans)
    info()
    return ans

a, b, c = 1, 1, 1
abcSum = lambda a,b,c: a*b*c
abcMult = lambda a,b,c: a+b+c
abcDiv = lambda a,b,c: (a+b)/c

for f in (abcSum, abcMult, abcDiv):
    p = mp.Process(target=solve, args=(f, (a, b, c)))
    p.start()
    p.join()
```

```
result:  1
nombre proceso: __main__
proceso padre: 103
proceso actual: 1034
-----
result:  3
nombre proceso: __main__
proceso padre: 103
proceso actual: 1045
-----
result:  2.0
nombre proceso: __main__
proceso padre: 103
proceso actual: 1056
-----
```

Figure 3: Ejemplo utilización de multiprocessing con 3 procesos

Bibliografía

- Linux man pages. Retrived October 29 from https://www.man7.org/linux/man-pages/index.html.
- Amdahl's law. Retrived October 29 from https://en.wikipedia.org/wiki/Amdahl%27s_law.
- What Is the Python Global Interpreter Lock (GIL)? https://realpython.com/python-gil/

- Multiprocessing Python. Retrived October 29 from https://docs.python.org/3/library/multiprocessing.html#module-multiprocessing.