1. Taula de l'apartat 1

Temps obtingut a partir de 10 milions d'execucions de cada funció i fer la mitja aritmètica.

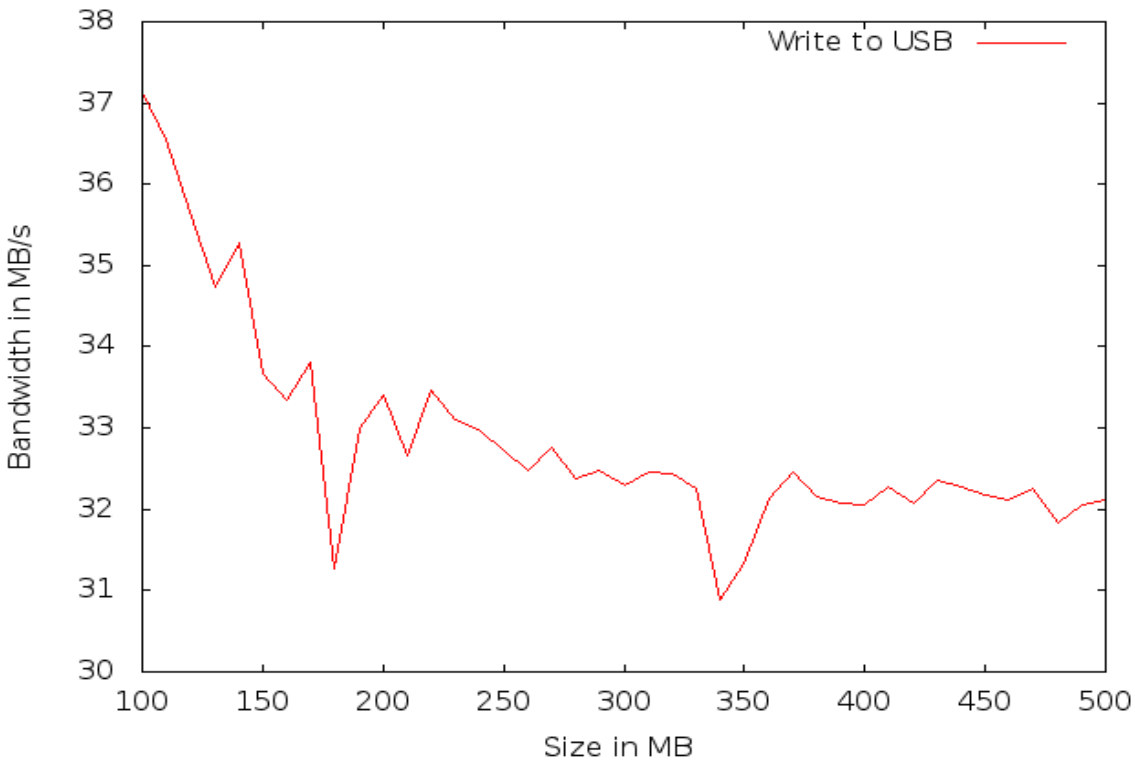
Syscall	sbrk(0)	sbrk(inc)	sched_yield()	getpid()	Fork()/waitpid()
Temps d'execució (ms)	7524,1	219364,9	290069,4	6617,9	110382,8

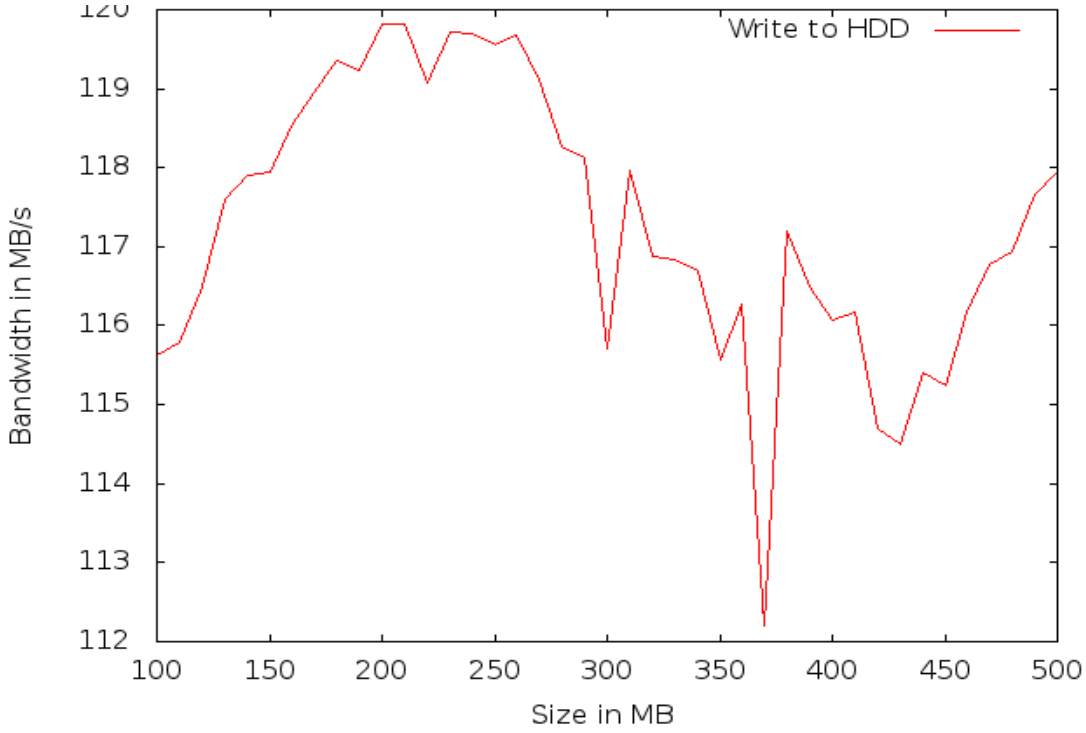
Podem observar clarament com el temps del sbrk amb increment és major que el de la mateixa funció però sense increment. Això és degut a que sense increment només s'ha retornar el valor del SP.
Per altra banda, dir que el temps del sched_yield és tan elevat degut a que s'ha de cambiar de procés (potencialment), i això és una operació molt costosa.

Podem veure també com el getpid és bastant similar al sbrk sense increment, ja que només ha de retornar un valor del PCB.

El fork i el waitpid té un cos bastant inferior al sched_yield ja que, encara que es fa un canvi de context, no s'ha d'invalidar el TLB ja que l'espai d'adreces serà el mateix degut a les polítiques de reemplaçament de pàgines del sistema operatiu Linux. Però, en el cas d'haver de crear un nou espai d'adreces pel fill, llavors el cost hauria de ser major que pel sched_yield.

2. Gràfiques de l'apartat 2





Mirant a les gràfiques podem veure com el bandwidth aconseguit al escriure al HDD (112-120MB/s) és molt superior al aconseguit amb el USB (30-38MB/s). Això és normal ja que estem tractant amb dos busos diferents (SATA II i USB 2) els quals tenen diferents bandwith màxims. En el cas de SATA II, és de 300MB/s, en el USB 2 és de 60MB/s. En ambdos casos, el bandwidth

aconseguit és molt inferior al màxim teòric, però ja és normal ja que aquest màxim teòric és el bandwidth que es podria aconseguir en condicions òptimes i sense cap tipus d'overhead, com pot ser el sistema de fitxers, canvis de context, temperatures no ideals, soroll, etc.