PRÁCTICA 1: MPI

Se desean realizar unas "medidas de rendimiento" de operaciones aritméticas con distintos tipos de datos. Para ello se dispone de una serie de funciones (incluidas en el fichero "test.c") en las cuales se indican el número de iteraciones a realizar y se obtiene un resultado y el tiempo empleado.

……

```
double test_long_add_sub(unsigned long  iMax, long *vSal);
double test_long_mul_div(unsigned long  iMax, long *vSal);
double test_long_long_add_sub(unsigned long  iMax, long long *vSal);
double test_long_long_mul_div(unsigned long  iMax, long long *vSal);
double test_double_add_subunsigned long  iMax, double *vSal);
double test_double_mul_div(unsigned long  iMax, double *vSal);
double test_long_double_add_sub(unsigned long  iMax, long double *vSal);
double test_long_double_mul_div(unsigned long  iMax, long double *vSal);
```

……

Se desea crear un programa que permita lanzar los distintos test en distintos procesos y obtener los resultados de la ejecución a fin de observar la ganancia frente a la ejecución en un proceso.

SE PIDE:

- **Realizar un programa en MPI que permita distribuir el cálculo.** Para ello habrá:
  - Un proceso encargado de la E/S que:
    - Mostrará un menú por pantalla que permitirá:
      - Seleccionar el número de iteraciones y notificársela al resto de procesos.
      - Seleccionar el test a realizar:
        - Notificárselo al resto de proceso
        - En caso de ser también calculador calculará las iteraciones que le correspondan del test seleccionado
        - Recogerá los datos del resto de procesos y los mostrará por pantalla
        - Volverá al menú principal
      - Finalizar:
        - Notificará al resto de procesos que no se les va a pedir realizar nada más y que pueden terminar.
  - N procesos que:
    - Esperarán a que el proceso de E/S les dé una orden:
      - En caso de recibir el número de iteraciones lo almacenarán para cuando tengan que realizar un test y volverán a esperar una orden del proceso de E/S.
      - En caso de recibir la orden de realizar un test ejecutarán las iteraciones que les correspondan y una vez terminado notificarán los resultados y volverán a esperar una orden del proceso de E/S.
      - En caso de recibir la orden de finalizar finalizarán.
  - En la página 4 hay un ejemplo de una posible salida por pantalla de la ejecución.
  - Mientras no se calcula la carga de CPU debe de ser prácticamente nula (ver página 5).

- **Una vez realizado el programa realizar un estudio de rendimiento en el cual se vea que:**
  - Aumentando el número de procesos se reduce el tiempo de cálculo hasta un cierto momento (por ej.: 1 (secuencial), 2, 4, 8, 16, 64,…..).
  - Una vez alcanzado el tope de una máquina probar con varias máquinas en red.
  - Interpretar los resultados obtenidos.
  - Es recomendable emplear un número de iteraciones grandes para que los tiempos de en secuencial sean grandes.

SE APORTA:

- El fichero fuente (test.c/test.h) con las funciones de cálculo que hay que emplear.

SE DEBERÁ ENTREGAR:

- Código fuente del programa realizado.
- Presentación empleada en el Seminario
    - Breve descripción del trabajo realizado:
        - Aspectos relevantes
        - Decisiones a la hora de distribuir la carga
        - Estructura de comunicación empleada
        - …
    - Estudio del rendimiento obtenido
    - Todo aquello que se estime oportuno para explicar el trabajo realizado

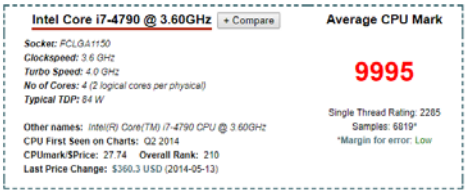
NOTA:

- Se deberán realizar pruebas en las que el resultado de la distribución sea palpable, es decir, que en secuencial los tiempos de ejecución sean grandes, y se observe la mejora de rendimiento
- Realizar pruebas tanto en una máquina como en varias.
- Hay que tener cuidado con los tipos de datos empleados y que se transmiten
- Las funciones de cálculo no calcula nada específico, simplemente realizan una serie de operaciones para un tipo de datos determinado.
- La práctica no trata de optimizar algoritmos ni de emplear las opciones de optimización del compilador, trata de emplear fuerza bruta.
- La práctica se realizarán preferiblemente en grupos de 4 personas
- Se deberá subir al Studium en un único archivo, un único componente del grupo.
- La presentación de la práctica se realizará en la sesión marcada como Seminario, teniendo que subir la presentación en las fechas marcadas en Studium.
- Cualquier modificación del enunciado de publicará en Studium.
- La detección de copia parcial o total de la práctica conllevará la suspensión de las prácticas, y por tanto de la asignatura.

NOTA: Son solamente ejemplos de pruebas incompletas.

Máquina empleada:

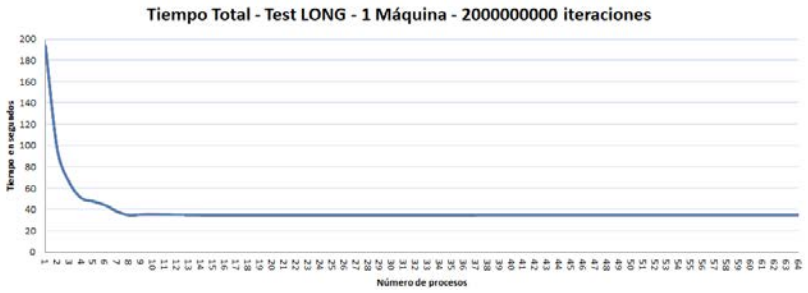
- Intel(R) Core(TM) i7-4790 CPU @ 3.60GHz

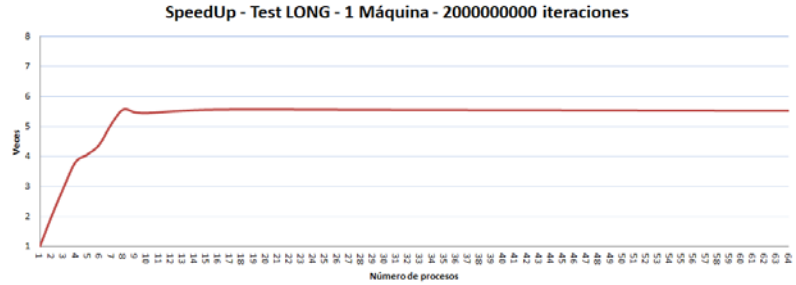


https://www.cpubenchmark.net/cpu.php?cpu=Intel+Core+i7-4790+%40+3.60GHz&id=2226

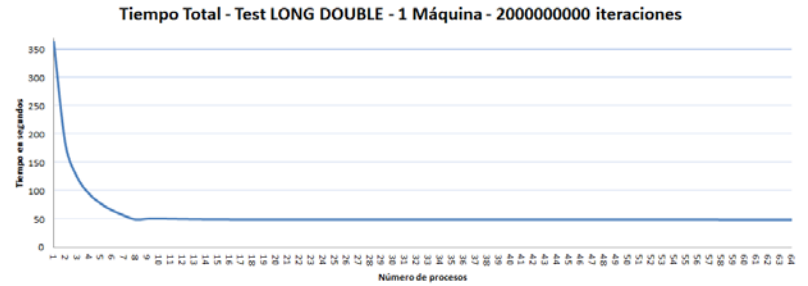
- 8GB de memoria
- Debian GNU/Linux 7.11 (wheezy) 32bits
- Open MPI 1.4.5 (Language: C)

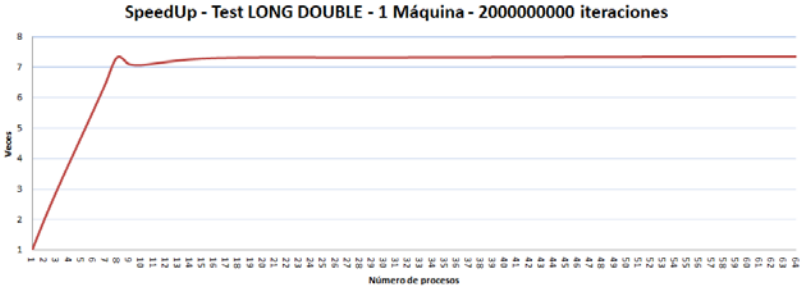
Prueba realizada: Test LONG en 1,2,3,4,5,6,7,8,9,10,16,32 y 64 procesos para 2.000.000.000 iteraciones en una máquina





Prueba realizada: Test LONG DOUBLE en 1,2,3,4,5,6,7,8,9,10,16,32 y 64 procesos para 2.000.000.000 iteraciones en una máquina

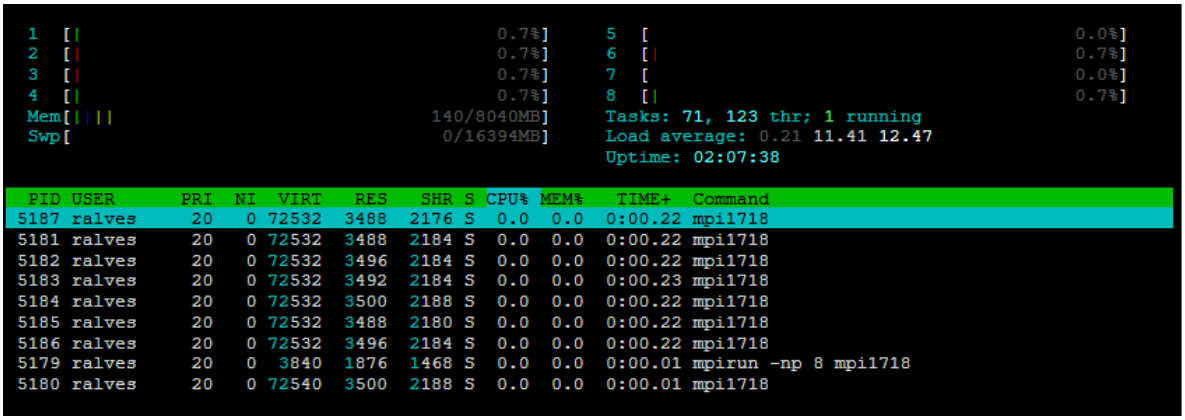




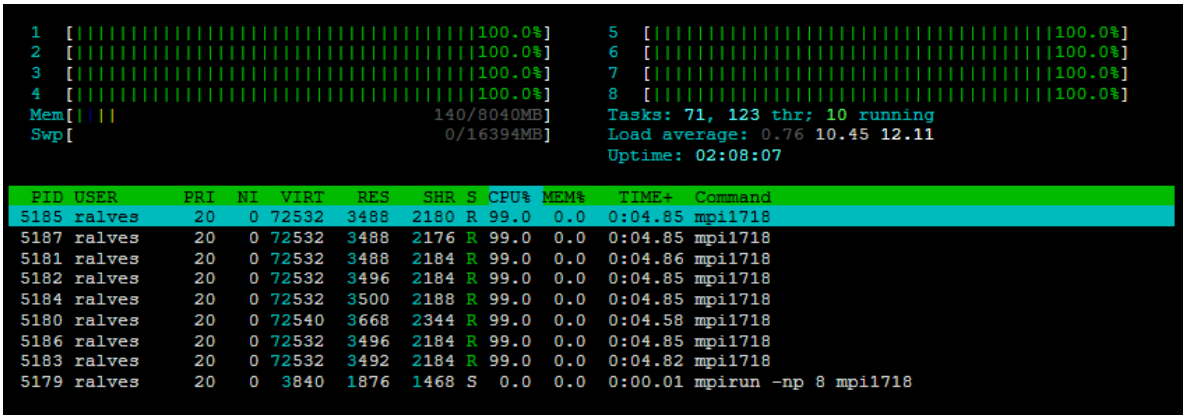
Una posible salida por pantalla:

```
mpirun -np 8 t

OBTENIENDO DATOS SOBRE LOS PROCESOS:
Proceso 0 en lipc29
Proceso 1 en lipc29
Proceso 2 en lipc29
Proceso 3 en lipc29
Proceso 4 en lipc29
Proceso 5 en lipc29
Proceso 6 en lipc29
Proceso 7 en lipc29

.................................

ESTABLECIENDO NUMERO DE ITERACIONES POR DEFECTO: 2000000000

BCAST: Iteraciones por defecto

Seleccione opciOn:
1) Introducir iteraciones (actual: 2000000000)
2) Test LONG
3) Test LONG LONG
4) Test DOUBLE
5) Test LONG DOUBLE
S) Salir
1

Iteraciones (1-4000000000): 1000000000

NOTIFICANDO ITERACIONES 1000000000
Seleccione opciOn:
1) Introducir iteraciones (actual: 1000000000)
2) Test LONG
3) Test LONG LONG
4) Test DOUBLE
5) Test LONG DOUBLE
S) Salir
2

CALCULANDO.....................

<----------TEST LONG---------->
Proceso: 0) Iteraciones: 125000000, Result: 125000001, Tpo +-: 6.069591, Tpo */: 11.351930
Proceso: 1) Iteraciones: 125000000, Result: 125000001, Tpo +-: 6.057560, Tpo */: 11.351932
Proceso: 2) Iteraciones: 125000000, Result: 125000001, Tpo +-: 6.047309, Tpo */: 11.351931
Proceso: 3) Iteraciones: 125000000, Result: 125000001, Tpo +-: 6.042227, Tpo */: 11.351929
Proceso: 4) Iteraciones: 125000000, Result: 125000001, Tpo +-: 6.048620, Tpo */: 11.351934
Proceso: 5) Iteraciones: 125000000, Result: 125000001, Tpo +-: 6.059095, Tpo */: 11.351929
Proceso: 6) Iteraciones: 125000000, Result: 125000001, Tpo +-: 6.040889, Tpo */: 11.351936
Proceso: 7) Iteraciones: 125000000, Result: 125000001, Tpo +-: 6.059471, Tpo */: 11.351931
RESULTADO FINAL:
Iteraciones totales          : 1000000000
Tiempo Total  + y - acumulado: 48.424762
Tiempo Total  * y / acumulado: 90.815453
Resultado Total: 1000000008
Tiempo Total: 17.465235
Iteraciones por segundo: 57256601

Seleccione opciOn:
1) Introducir iteraciones (actual: 1000000000)
2) Test LONG
3) Test LONG LONG
4) Test DOUBLE
5) Test LONG DOUBLE
S) Salir
s

NOTIFICANDO SALIDA
```

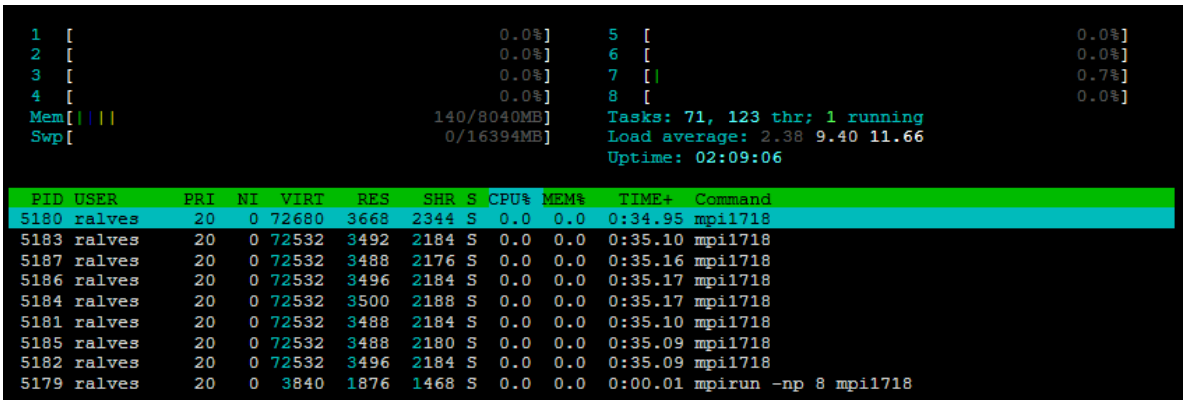
Estado de la CPU:



Esperando selección de opción.



Ejecutando el Test.



Una vez finalizado el Test esperando opción.