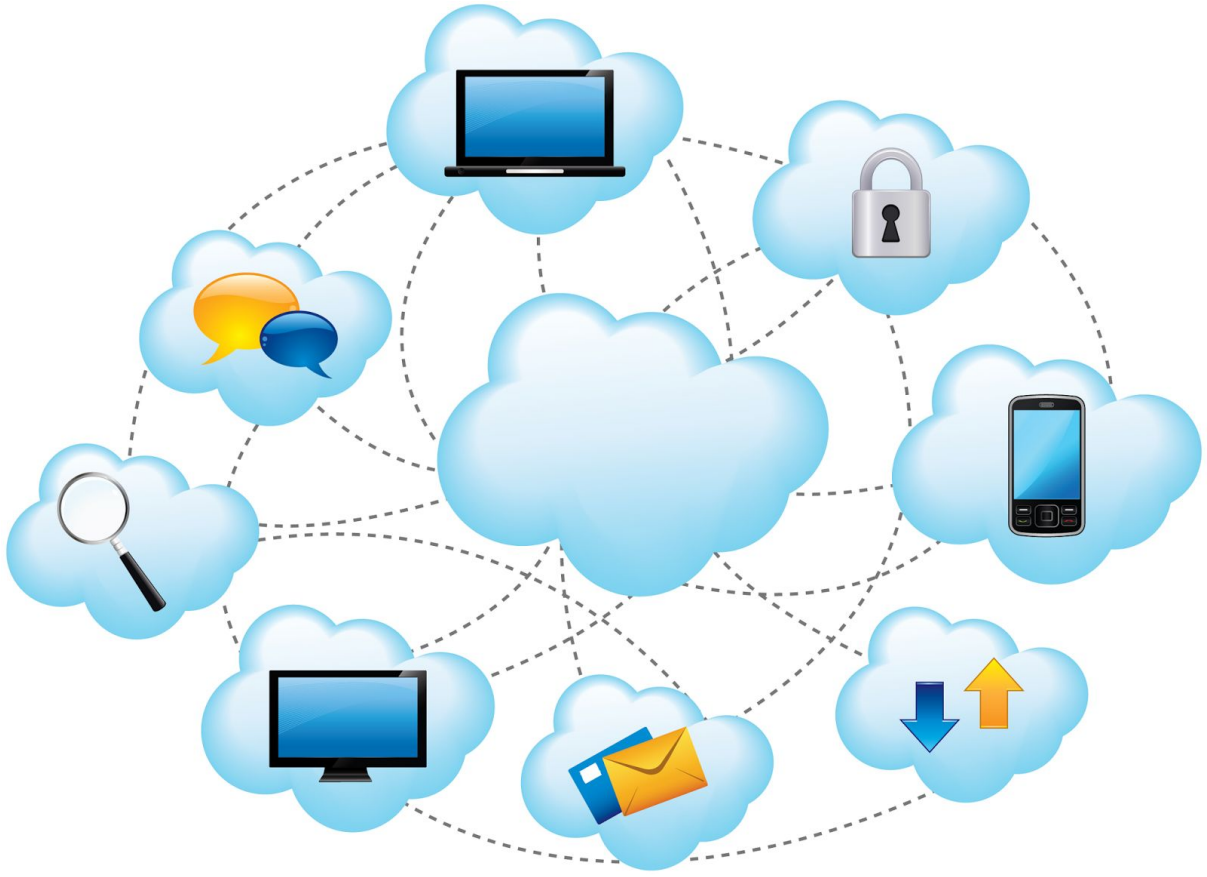
Reporte T3
Programación concurrente.



Francisca Rebolledo Martínez

Compilación del programa.

El programa se debe compilar con los comandos dados, primero para compilar el programa serial:

- mpicc -o t3_serial t3_serial.c -lm

y para el paralelo:

- mpicc -o t3_parallel t3_parallel.c -lm

Luego para correr cada programa:

- serial: mpirun -mca btl ^openib --oversubscribe -host trauco -np 1 ./t3_serial
- paralelo: mpirun -mca btl ^openib --oversubscribe -host trauco -np x ./t3_parallel

donde x es el número de procesos, si se quiere agregar nodos, agregar los nombres tal como sale en el enunciado.

Principales funcionalidades

Lo que hace mi tarea es un loop verificando que la suma de los cuadrados de ambos números randoms sean menor o igual a 1, hasta que la diferencia entre pi estimado y pi real sea muy pequeña, en ese momento retorna.

Para el caso distribuido y con paso de mensajes, asumí un proceso que calcula la aproximación de pi cada vez que los otros procesos(cuando se completa una ronda) le envían un contador(cantidad de veces que la suma de los cuadrados ha sido menor o igual que uno) y una cantidad(cantidad de iteraciones hechas por el proceso), sumando todo lo enviado y haciendo las respectivas operaciones, calcula la estimación de pi y la envía a los otros procesos para que sepan si deben dejar el loop o no.

Reporte

a)

Tipo	cantidad procesos	tiempo
secuencial	1	0.000059
paralelo local	2	0.027
paralelo local	4	0.039
paralelo local	8	0.052
paralelo local	16	0.131
paralelo distribuido	2	0.234
paralelo distribuido	4	0.257
paralelo distribuido	8	0.338
paralelo distribuido	16	0.374
paralelo distribuido	32	0.564

b) Al parecer la comunicación entre procesos locales es bastante más rápida que entre procesos distribuidos, si miramos los tiempos de ejecución con la misma cantidad de procesos es bastante mayor una respecto a la otra, sin embargo en la comunicación distribuida los tiempos no crecen tan rápido como en la comunicación local.

c) <u>Respuestas</u>

1) La manera en que MPI distribuye los procesos en más de un nodo es la siguiente, se ejecuta el programa en todos los nodos necesarios, las veces que le sean indicadas, de esta manera a cada ejecución del programa se le asigna una id, por lo que dentro el programa debe tener un if(mi id) y asignarle la tarea correspondiente.

2) Usar MPI puede ser beneficioso para ejecuciones de tareas que requieran un gran computo de operaciones, aproximar pi hasta los 4 decimales no lo és y por eso el tiempo es tan malo en comparación al serial, si hubiese sido con muchos más decimales probablemente hubiese sido de utilidad.

3) Respecto a las diferencias en velocidades de comunicación, lo mismo que se comentó en el punto b luego de la tabla, los procesos paralelos pero locales se comunican más rápido hasta cierto punto, ya que pertenecen al mismo nodo, por lo que la información deberá viajar más tiempo, sin embargo, mientras más mensajes se envían este empieza a “colapsar”, para los distribuidos sería lo contrario, cuando son pocos, debido a la distancia que recorre el mensaje, el tiempo es más largo, pero mientras más procesos haya, como los procesos locales “colapsan” se ve más rápida la velocidad entre procesos distribuidos.