Servicios de Computación de Altas Prestaciones y Disponibilidad

Tema 4. Evaluación de Prestaciones (Benchmarking)

Guion de Prácticas

Aurora Macías Ojeda

Resumen. Este documento propone una relación de actividades para poner en práctica los conocimientos adquiridos en el *Tema 4* de la asignatura *Servicios de Computación de Altas Prestaciones y Disponibilidad* del MUii.

1 Introducción

La Computación de Altas Prestaciones constituye uno de los campos más dinámicos dentro de la Informática actualmente, así como también sus aplicaciones. Debido a ese dinamismo y a la creciente tendencia al uso de sistemas de ese tipo, la necesidad de su evaluación también gana relevancia.

Este documento recoge diversas actividades propuestas para la práctica de los conceptos vistos en teoría sobre las herramientas de evaluación a las que se hace referencia en las secciones siguientes.

Algunos archivos necesarios para llevar a cabo la práctica pueden encontrarse en el siguiente enlace: *https://github.com/auroramo/SCAPD_T4*.

Es importante hacer constar que no se incluyen instrucciones o recomendaciones especiales debido a que todo lo que es necesario conocer sobre la operativa a seguir en aspectos comunes a los de otras partes de la asignatura ya se ha comentado en sesiones anteriores.

2 Gprof

A continuación, se propone una serie de actividades para afianzar los conocimientos sobre el uso básico de la herramienta de perfilado *Gprof*.

2.1 Ejercicio 1

Obtén el archivo de código fuente *bucles.c* y súbelo al clúster (crea una carpeta para esta sesión).

Una vez tengas el archivo en PK2:

a) Compila el código de manera que se añada información de perfilado para Gprof.
b) Ejecuta el programa creado.
c) Examina el archivo de perfilado y contesta a estas preguntas:
    1. ¿Cuánto tiempo tarda la ejecución?
    2. ¿Cuál es la función que más veces se invoca?
    3. ¿Cuál es la función que más tiempo tarda?
    4. ¿A qué hace referencia "<spontaneous>"?
d) Modifica el código para que cada función (producto y división) se implemente de acuerdo a la definición que nos dieron en el colegio de cada una de ellas. (Se proporcionará ayuda durante la sesión)
e) Compila el código y ejecuta el programa creado.
f) Examina el archivo de perfilado.
   ¿Refleja la tabla algún cambio destacable con respecto a las funciones de producto y división?
g) Agrega una nueva función al código (potencia, por ejemplo) pero no la invoques desde ninguna parte. Compila el código, ejecuta el programa creado, y examina el archivo de perfilado.
   ¿Qué podrías destacar de la nueva salida?

3 Valgrind

En esta sección se recoge una serie de actividades para practicar los conceptos básicos de la herramienta *Valgrind* y su utilidad en la detección de problemas de memoria en los programas.

3.1 Ejercicio 1 (Memcheck)

Obtén el archivo de código fuente *julaMenck3b.c*. Una vez tengas el archivo en PK2:

a) Compílalo para poder usar después alguna de las herramientas de Valgrind.
b) Ejecuta el programa generado con la herramienta por defecto de Valgrind (no olvides habilitar el detector de fugas de memoria -*memory leak*- detallado).
c) Comenta los errores detectados.
d) Enumera los tipos de errores que puede detectar la subherramienta.

3.2 Ejercicio 2

A partir del archivo de código fuente del ejercicio anterior que ya tienes en PK2:

a) Ejecuta otra de las herramientas de Valgrind (por ejemplo, Massif).
b) Intenta explicar, en líneas generales, qué información ofrece.

4 TORQUE

En prácticas anteriores (ver Práctica MPI de Tema 3) se han visto algunas de las directivas de *TORQUE* para especificar las características de un job, y cómo se envía el job para que se lleve a cabo su ejecución. En esta práctica vamos a profundizar más en el uso básico de la herramienta.

Teniendo en cuenta lo visto hasta el momento, realiza los ejercicios siguientes.

4.1 Ejercicio 1

Usando la plantilla de ejemplo de script de envió de un job:

a) Programa un script que muestre un mensaje en la consola indicando que su ejecución está comenzando.
   - Llama al job "TankWarVR".
   - Solicita la asignación de 1 máquina o nodo, 2 procesadores en cada una de ellas, y 256MB en cada procesador.
   - El script deberá posibilitar que se tenga acceso a los archivos de salida tras la ejecución del job.
   - También deberá permitir que se notifiquen los diferentes cambios de estado del job enviado.
   - Solicita, además, un tiempo de ejecución (el tiempo máximo de reloj durante el que se permitirá la ejecución del job) de 15 minutos.

   Incluye el código fuente del script (añade comentarios indicando en qué punto das solución a los diferentes aspectos de lo que pide el enunciado).
   ¿Has recibido notificación alguna?

b) Envía el job para que sea ejecutado (a la cola por defecto) y consulta su estado inmediatamente.
   ¿Cuál es el resultado de los comandos en ambos casos? ¿A qué crees que se debe el resultado de la segunda acción?

c) Consulta los archivos de salida.
   ¿En qué ruta se han generado? ¿Qué nombre tienen? Comenta su contenido.

d) Modifica el script y añade tras el mensaje las instrucciones para ejecutar un programa que no existe (como se hacía en la Práctica 3). Envía el job para su ejecución y consulta los archivos de salida.
   Incluye el nuevo script y los contenidos de los archivos como solución a este apartado.

4.2 Ejercicio 2

Trabaja sobre el script del ejercicio anterior.

a) Añade un retardo de 40 segundos tras el mensaje.

b) Añade otro mensaje tras el retardo indicando que la ejecución está terminando.

c) Envía el job para que sea ejecutado y consulta su estado inmediatamente. Consulta el estado repetidamente hasta obtener la misma respuesta que en el ejercicio anterior.
   Interpreta las salidas. ¿Cuáles son los posibles estados de un job?
d) Consulta los archivos de salida para comprobar el contenido.
   Si no supieses donde encontrarlos, ¿cómo averiguarías su ruta?
e) Vuelve a enviar el job para su ejecución. Espera unos segundos (menos de 40) y elimina el job de la cola. Consulta de nuevo los archivos de salida.
   ¿Cuál es el contenido en este caso?

4.3 Ejercicio 3

Trabaja sobre el script original del Ejercicio 1.

a) Añade un retardo de 20 segundos tras el mensaje, y añade otro mensaje indicando que el job está finalizando.
b) Envía una serie (array) de 3 jobs usando un único comando y un único script de envío.
   Consulta inmediatamente el estado de la cola.
c) Consulta los archivos de salida.
   ¿Cuántos hay? ¿Cuáles son sus nombres?
d) Utilizando la misma filosofía que en b), envía una serie de jobs cuyos nombres sean los de los protagonistas de “Expediente X”.
   Vuelve a consultar los nombres de los archivos de salida.

4.4 Ejercicio 4

Trabaja sobre el primer script del ejercicio anterior y el script original del Ejercicio 1.

a) Cambia el nombre a los jobs correspondientes a cada script: “conductor” en el primer caso y “artillero” en el segundo.
   De considerarse necesario, amplía el tiempo de retardo del primer script.
b) Envía ambos jobs para su ejecución, teniendo en cuenta que la ejecución del segundo job no deberá comenzar hasta que no termine el primero.
c) Consulta el estado de la cola.