Evaluación del rendimiento del sistema en un entorno real

Diego Martín Arroyo

Domingo, 25 de mayo de 2015

Resumen

Una adecuada evaluación de las herramientas, protocolos y arquitectura planteadas debe incluir un conjunto de pruebas en entornos en los que la versión final del sistema debe integrarse. Dichas pruebas deben ser preparadas de antemano.

Índice

Marcobootstrap

Se realizarán las siguientes pruebas

Instalación del sistema operativo en un nodo sin asistencia del administrador, evaluando el tiempo de descarga e instalación

Sistemas utilizados

- Una Rasbperry Pi Modelo 2B
- Un servidor Sun
- Imagen creada con las versiones estables más recientes de todas las herramientas, con un tamaño de
- La prueba se realiza en la infraestructura final, conectando los nodos a las tomas de red del Laboratorio de Ingeniería Informática.

Resultados

- **Tiempo de descarga**: 3 minutos, 27 segundos
- **Tiempo de instalación**: 11 minutos, 49 segundos
- **Tiempo de instalación**: 22 minutos, 37 segundos

Actualización del sistema operativo

Se realiza la instalación de una versión alternativa del sistema operativo, comprobando que los componentes no actualizables permanecen intactas tras el proceso.

Deployer

Sistemas utilizados

- 4 Rasbperry Pis con la última versión de las herramientas a utilizar.
- La prueba se realiza en la infraestructura final, conectando los nodos a las tomas de red del Laboratorio de Ingeniería Informática.

Logger

- 4 Raspberry Pis ejecutando un programa que imprime un número elevado de mensajes en un orden dado
- La prueba se realiza en la infraestructura final, conectando los nodos a las tomas de red del Laboratorio de Ingeniería Informática.

Status monitor

Se evalúa el rendimiento de la herramienta con los cuatro equipos, uno de ellos compilando el software MPI simultáneamente. El rendimiento de la aplicación es el esperado, contando con una frecuencia de actualización de un segundo de forma constante.

Compilador

Se realiza como prueba la compilación de la biblioteca openmpi (Versión 1.8.4) con los siguientes parámetros:

- **Tiempo de configuración**: 15 minutos
- **Tiempo de compilación**: 23 minutos
- **Tiempo de instalación**: 6 minutos

MPI

Se han realizado las siguientes pruebas de rendimiento y viabilidad de MPI:

Práctica de la asignatura Arquitectura de Computadores

Se ha ejecutado la solución a la práctica realizada en la asignatura Arquitectura de Computadores durante el curso 2013-2014.
El enunciado de dicha práctica es el siguiente:
Se desea realizar una transformación de un programa que realiza una serie de cálculos de manera secuencial. En este se quieren "adivinarüna serie de números de manera aleatoria. Para ello se emplea la siguiente función, a la que se le pasa el número a descubrir y devuelve el número de intentos que ha necesitado para "adivinarloz el tiempo que ha empleado (se ha forzado la semilla a 1 siempre para que el número de intentos para descubrir un mismo número sea siempre el mismo):
SE PIDE:
Realizar un programa en MPI que permita distribuir el cálculo del ejemplo. Para ello habrá: Un proceso encargado de la E/S que:
El proceso de E/S deberá mostrar por pantalla como mínimo:
Para cada número a "adivinar": El número "adivinado" El número de intentos utilizado El tiempo empleado
No es necesario que la información de los números adivinados aparezca en el mismo orden que en la lista de números. El proceso de E/S mostrará por pantalla el tiempo empleado total y los valores acumulados, tanto de intentos como de tiempos de cálculo. El proceso de E/S puede ser o no calculador. Hay libertad para que el proceso de E/S también calcule o bien sea un mero repartidor de tareas.
N procesos calculadores que: Esperarán a que el proceso de E/S le pase un trabajo, ya sea un número o varios números.
Para los números adivinados deberá pasarle la información necesaria al proceso de E/S para que pueda sacarla por pantalla-
Los procesos podrán devolver los resultados como mejor se considere (uno por uno, en una estructura, etc), lo que hay que respetar son los distintos tipos de datos.
Al emplear la función rand hay que considerar inicializa la semilla en cada proceso con un valor distinto. El hecho de emplear un procedimiento aleatorio para descubrir los números hará que dependiendo del valor de la semilla los resultados puedan variar. Se ha forzado la semilla a 1 siempre para que el número de intentos para descubrir un mismo número sea siempre el mismo
Todos los procesos deberán terminar (no se pueden quedar en un bucle infinito esperando solicitudes).
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, 128...).
Una vez alcanzado el tope de una máquina probar con varias máquinas en red.
Es recomendable emplear un número de iteraciones grandes para que los tiempos de cálculo antes de distribuir sean grandes.

Resultados

Con 2 nodos Intentos Acumulados: 42206944176 Tiempo Acumulado: 10728.003393 Tiempo Total: 1691.146348 Max Intentos: 2798133238 Max Tiempo: 652.491734