Mejoras en el planificador de procesos de Linux en la versión 2.6.0

Linux 2.6.0 he hecho mejoras significativas en el planificador de procesos en la versión 2.6.0. No solamente los procesos son planificados mas eficientemente, sino que el planificador ha sido rediseñado para ser más escalable cuando se incrementa la cantidad de procesos en la máquina. En 2.4, la escalabilidad no es tan prominente.

Un beneficio adicional de las mejoras del planificador es el uso más eficiente de los recursos, tales como la creación de procesos, y con una configuración del kernel estándar, pueden ejecutarse más procesos en el sistema que antes.

Ejecuciones de la prueba de performance Hackbench

Las pruebas de performance fueron realizadas usando un programa llamado hackbench. Las pruebas se corrieron para comparar un Linux 2.4.18 contra el kernel 2.6.0-test9. Las pruebas se ejecutaron en cuatro sistemas distintos: un sistema de un único procesador, un sistema con 2 procesadores, un sistema con 4 procesadores y un sistema con 8 procesadores. Las configuraciones de los sistemas de detallan en los siguientes enlaces (en inglés):

Sistema de un único procesador
Sistema de dos procesadores
Sistema de cuatro procesadores
Sistema de ocho procesadores

¿Qué es hackbench?

La prueba hackbench es un benchmark para la medida de performance, sobrecarga y escalabilidad del planificador de Linux. Creado por Rusty Russell, utiliza procesos cliente y servidor agrupados para enviar y recibir datos de manera de simular las conexiones establecidas en una sala de chat. Cada cliente envía un mensaje a cada servidor en el grupo.

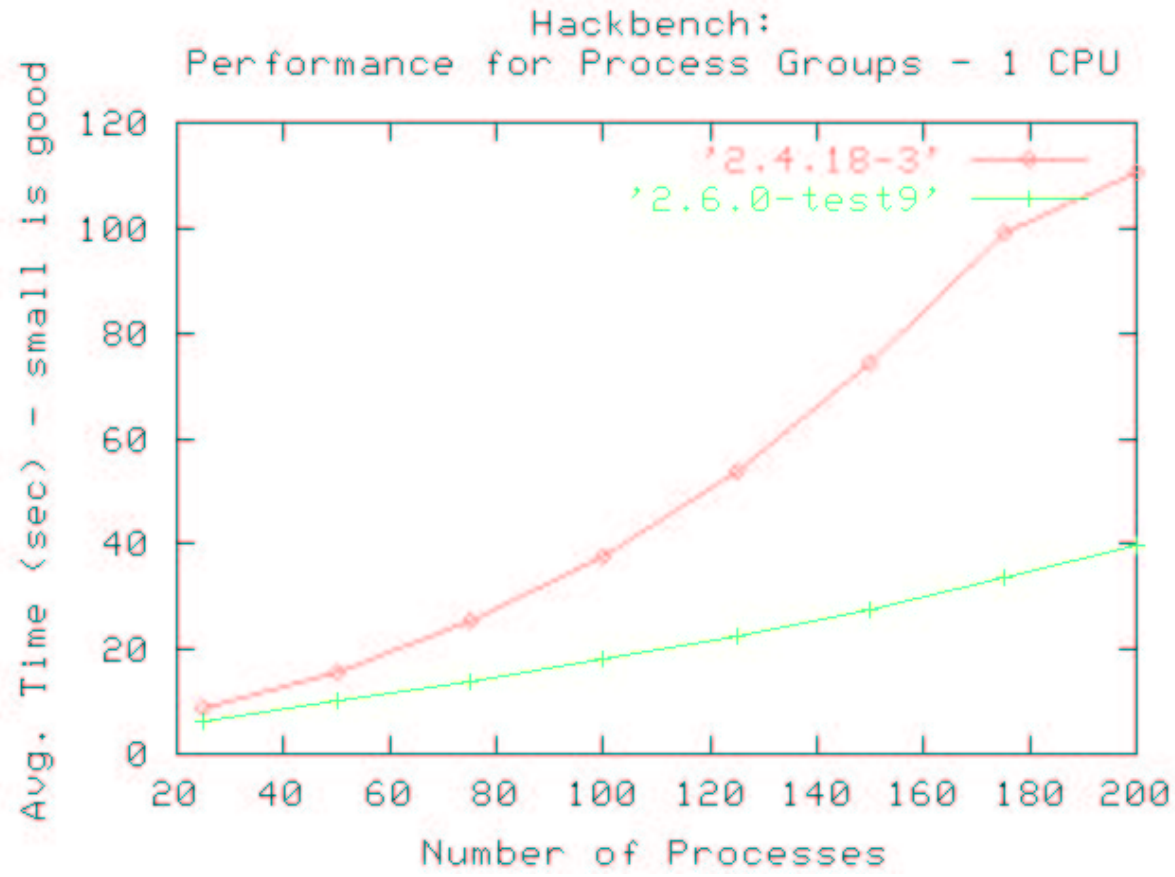
La prueba crea un grupo de procesos agrupados en una cantidad definida por el usuario (para este reporte, fue fijada en 25). Cada par cliente/servidor escucha en un socket; el emisor escribe 100 mensajes en cada socket y el receptor escucha en el socket. Por lo tanto, en este caso, la cantidad total de mensajes enviados son 100 por la cantidad de procesos especificados. El archivo fuente del programa de prueba es hackbench.c y es llamado desde un script perl que es ejecutado en la OSDL’s Scalable Test Platform. El script wrapper es runit.pl

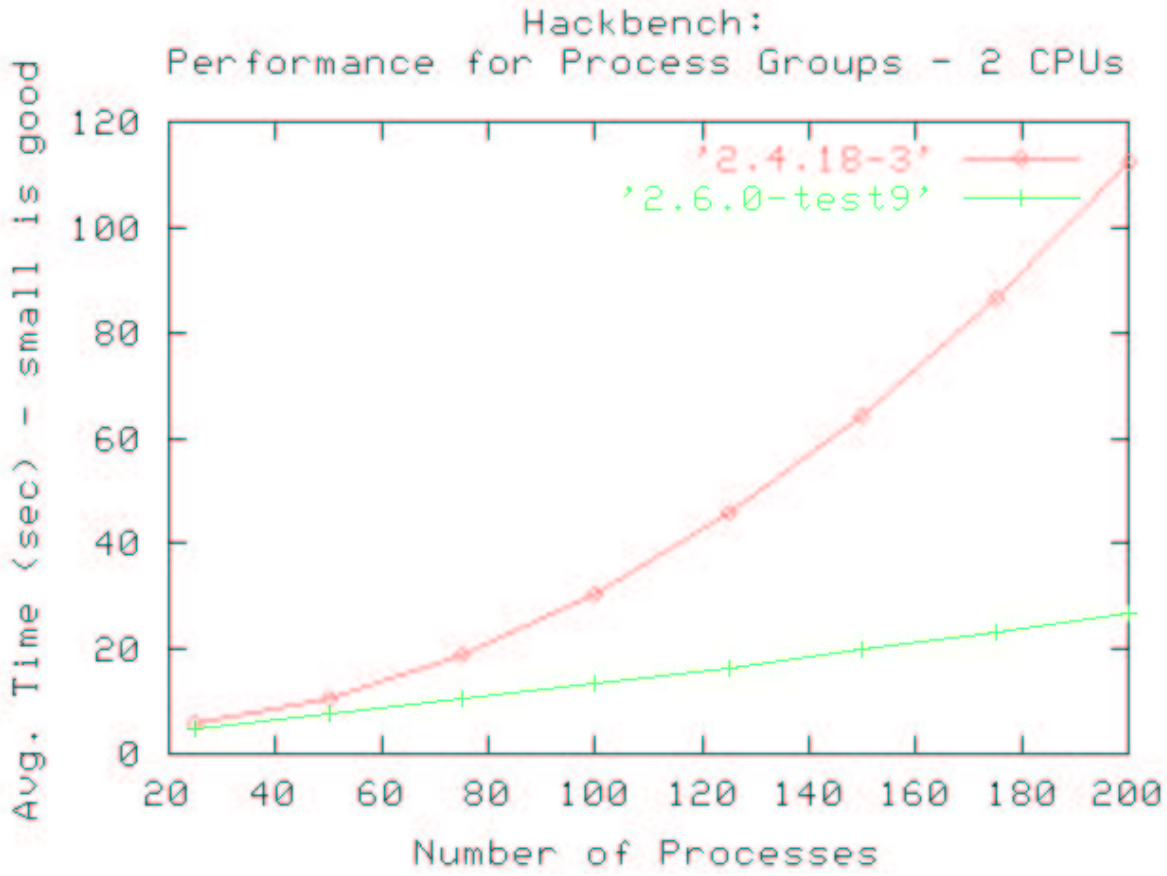
Resultados de las ejecuciones

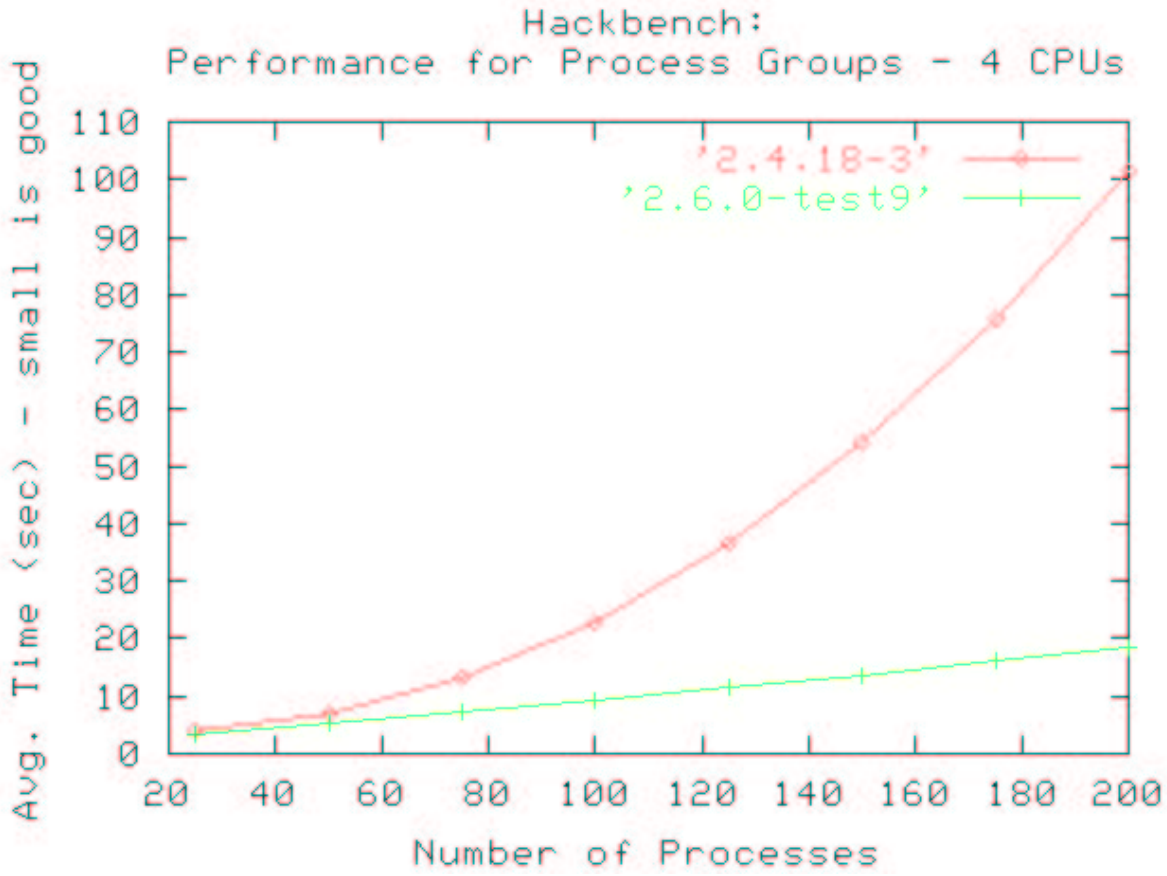
Cada prueba individual ejecuta una cantidad de 25 procesos, se incrementa a la siguiente cantidad y se vuelve a ejecutar el benchmark. Esto continúa hasta que se alcanza el nivel máximo definido por el usuario. Por ejemplo, el usuario quiere probar el sistema corriendo grupos de 25 procesos hasta que se alcanza un máximo de 100 procesos. Por lo tanto el benchmark correrá pruebas para 25, 50, 75 y 100 procesos antes de terminar. Cada conjunto de procesos es ejecutado 5 veces, y luego se informa como resultado el tiempo promedio. El reporte de la prueba tiene un gráfico del tiempo promediado para cada conjunto.

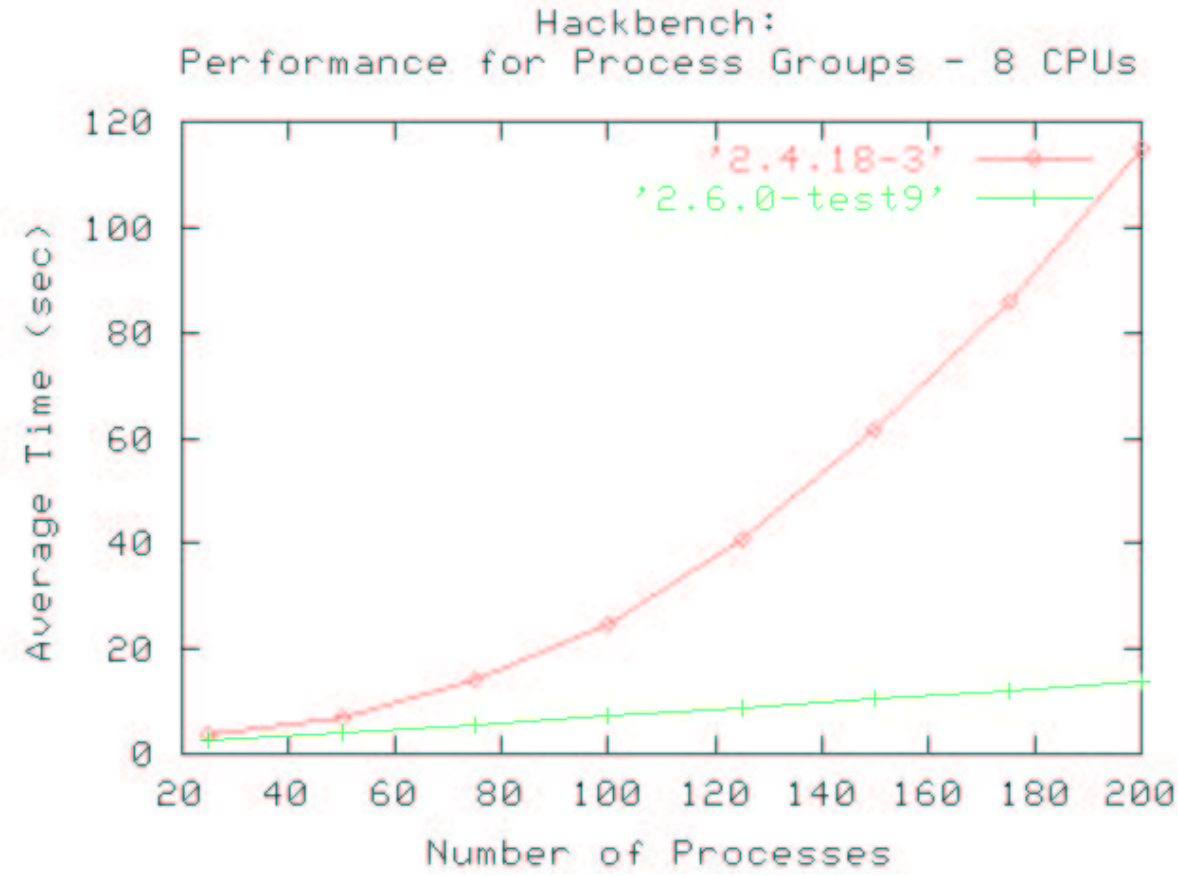
Observación 1: 2.6.0 es más eficiente que 2.4.18

El primer conjunto de pruebas fueron realizadas usando un valor máximo de 200 grupos de emisores y receptores. Las pruebas fueron realizadas en sistemas con un número creciente de procesadores. Los gráficos muestran los tiempos promedio que tardó en completarse hackbench dada la cantidad especificada de grupos de procesos en cada paso.




Hackbench:
Performance for Process Groups - 2 CPUs
Avg. Time (sec) - small is good
Number of Processes
'2.4.18-3'
'2.6.0-test9'
0
20
40
60
80
100
120
140
160
180
200


Hackbench:
Performance for Process Groups - 4 CPUs
Avg. Time (sec) - small is good
Number of Processes
'2.4.18-3'
'2.6.0-test9'
0
10
20
30
40
50
60
70
80
90
100
110
20
40
60
80
100
120
140
160
180
200

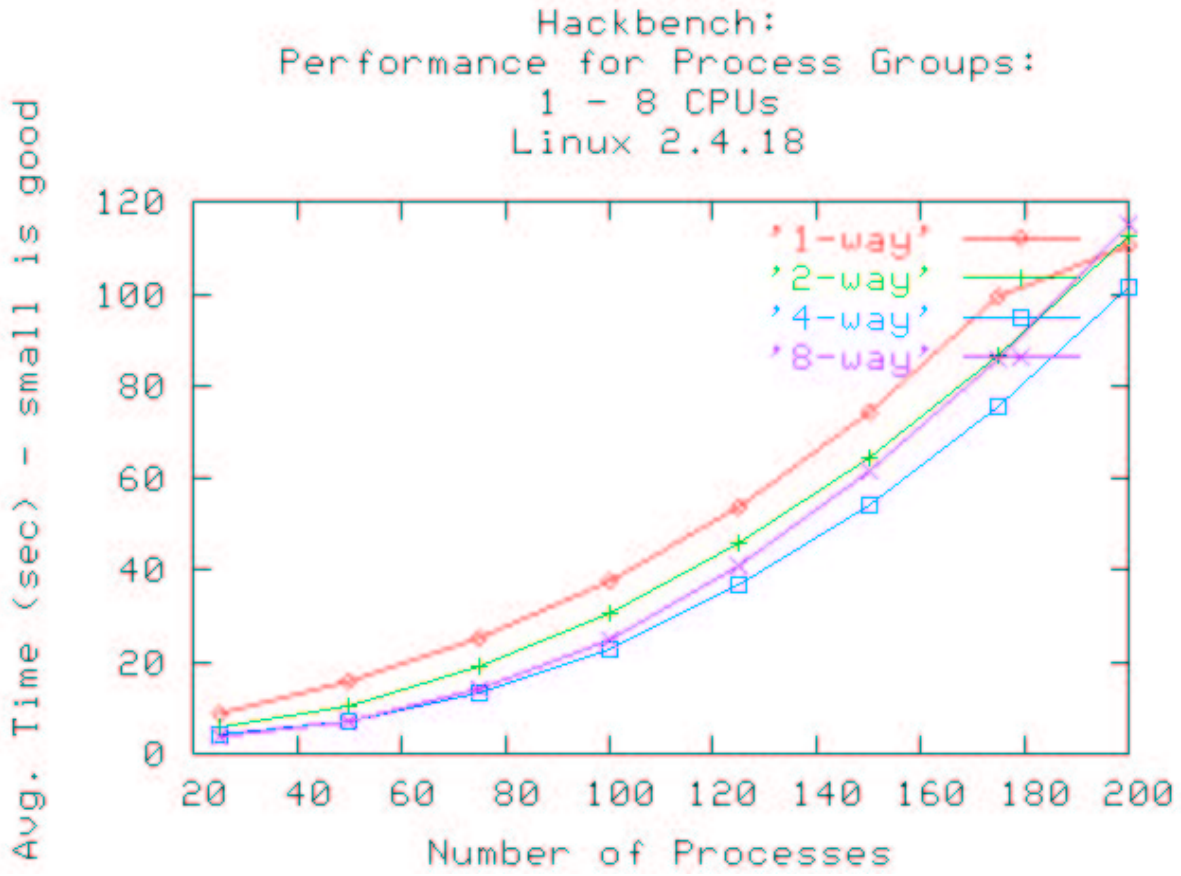


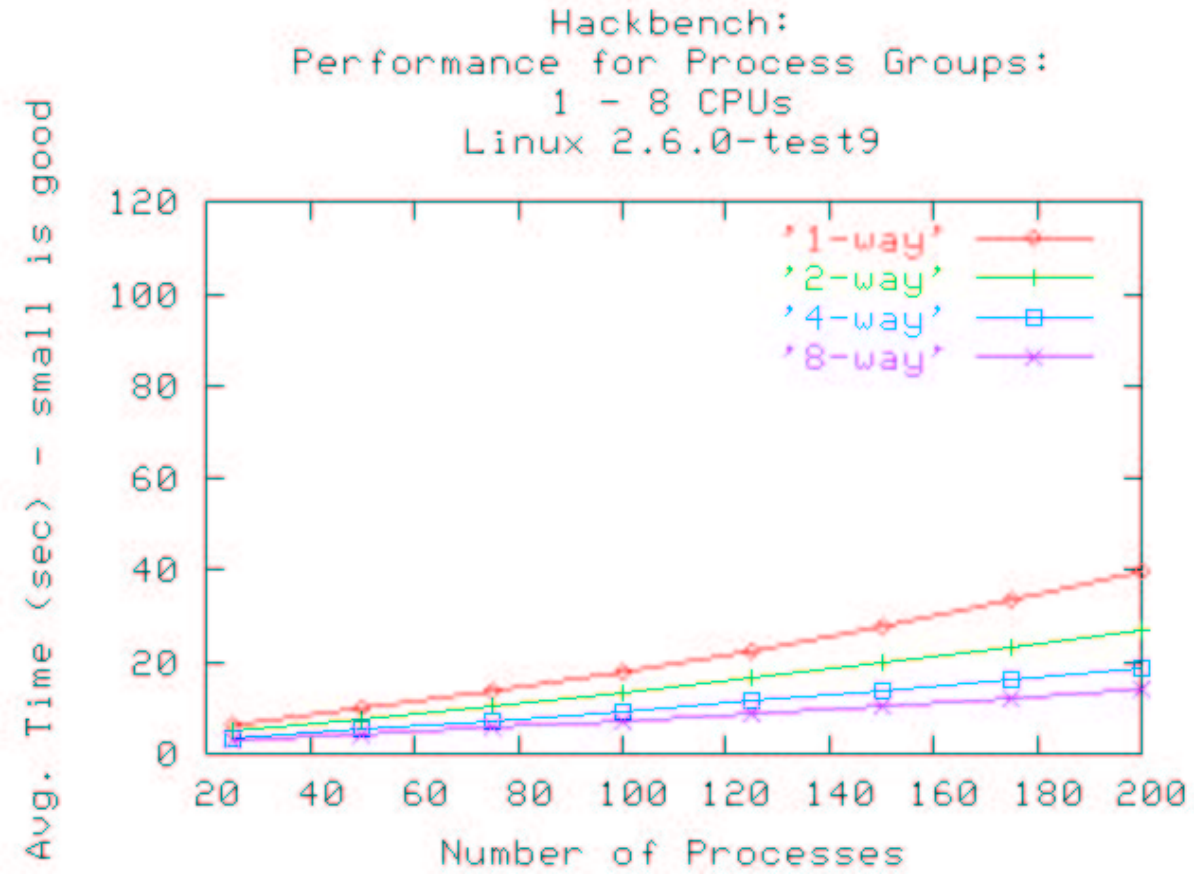
La comparación entre 2.4.18 y 2.6.0-test9 muestra que 2.4.18 tiene una curva parabólica, indicando que toma logarítmicamente más completar la prueba cuando se incrementa la cantidad de grupos (emisores y receptores). Cada emisor y receptor es creado como procesos separados; el planificador de procesos de 2.4.18 eventualmente será incapaz de administrar eficientemente los procesos cuando su número exceda determinado límite.

En contraste, el kernel 2.6.0-test9 mejorado muestra un trazo más lineal, indicando que el nuevo planificador de procesos puede administrar un gran número de grupos creados por la prueba hackbench. La curva también muestra que le demanda mucho menos tiempo el completar la prueba que al 2.4.18, fijado el número de grupos a ejecutar.. El resultado final es que las mejoras en el planificador de procesos de 2.6.0 proveen mejor performance y administran los procesos de manera más eficiente.

Observación 2: 2.6.0 es más escalable que 2.4.18

Los siguiente gráficos muestran los resultados de 2.4.18 para todos los sistemas probados, en un gráfico, y los resultados de 2.6.0 en el otro.


Hackbench:
Performance for Process Groups:
1 - 8 CPUs
Linux 2.4.18
Avg. Time (sec) - small is good
Number of Processes
'1-way'
'2-way'
'4-way'
'8-way'
0
20
40
60
80
100
120
20
40
60
80
100
120
140
160
180
200



Los resultados en el gráfico de 2.4.18 muestran que la performance es casi indistinguible en un sistema de 2 vías y en un sistema de 8 vías cuando la cantidad de grupos de procesos se acerca a 150. Esto indica que el planificador de procesos en 2.4.18 no escalará demasiado bien cuando hablamos de grandes sistemas (grandes cantidades de procesadores). Aparece, para esta prueba, como más eficiente el sistema compuesto por cuatro procesadores.

El kernel 2.6.0-test9 muestra una clara diferencia entre la performance y la cantidad de procesadores usados en el sistema. Esto implica que la escalabilidad es mejor cuando aumenta la cantidad de procesadores. El planificador de procesos (y otros componentes del kernel) pueden utilizar un número mayor de procesadores en el sistema. Por lo tanto, aquellos sistemas que ejecuten un mayor número de procesos pueden beneficiarse escalando hacia un sistema más grande con más procesadores.

Observación 3: 2.6.0 permite un uso más eficiente de los recursos

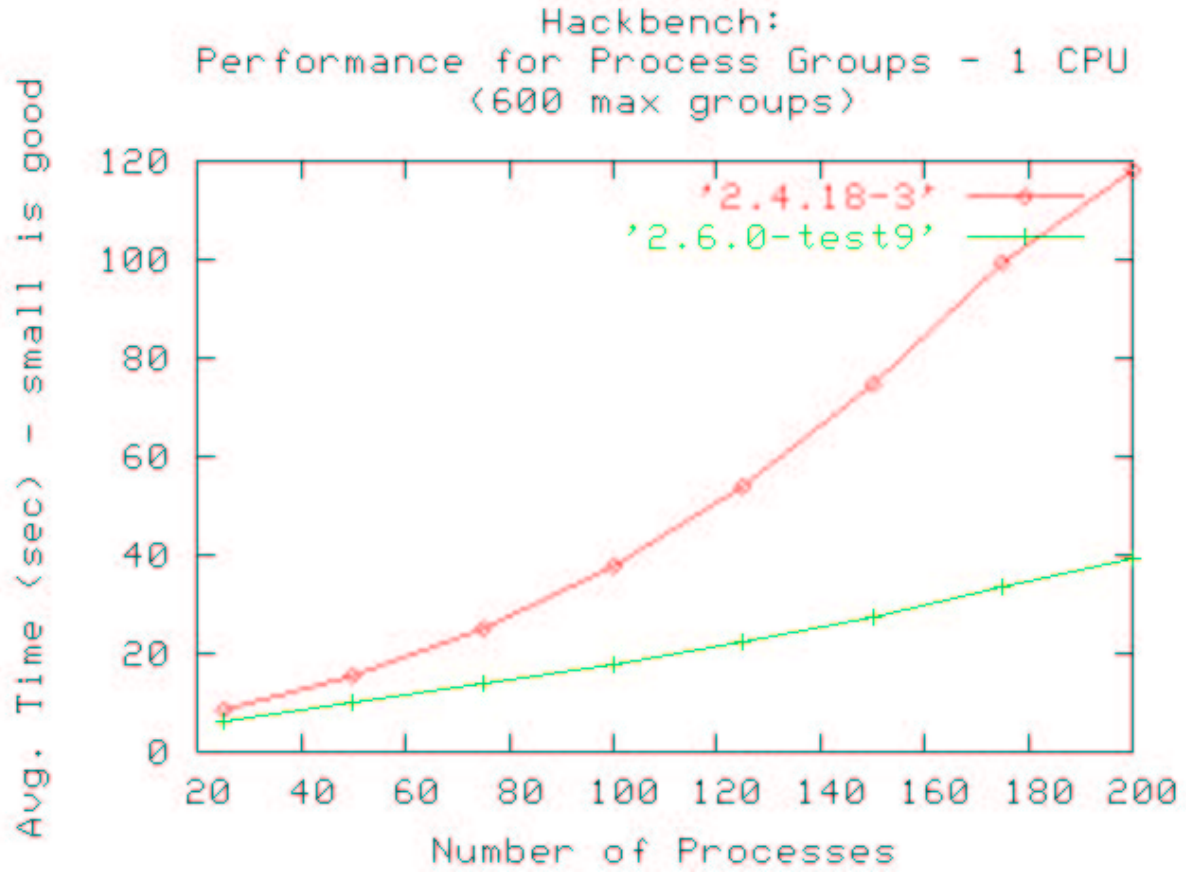
El siguiente conjunto de pruebas fueron realizadas para determinar la cantidad máxima de grupos que pueden ser ejecutados antes que el sistema encuentre una limitación de recursos. No se realizó ningún afinamiento del kernel, las pruebas fueron ejecutadas utilizando los parámetros de configuración por defecto. Las pruebas fueron invocadas con un valor incremental para la cantidad máxima de grupos de procesos y fueron ejecutados hasta que un mensaje de error indicara una limitación en los recursos. El mensaje de error producido fue el siguiente:

groupsfork() (error: Resource temporarily unavailable)

En ese momento, las pruebas se abortaron y se determinó el último conjunto completado. Los gráficos que siguen muestran la cantidad máxima de grupos de procesos para los cuales la ejecución de hackbench se completó exitosamente, dado que las pruebas se ejecutaron en sistemas con un número creciente de procesadores.

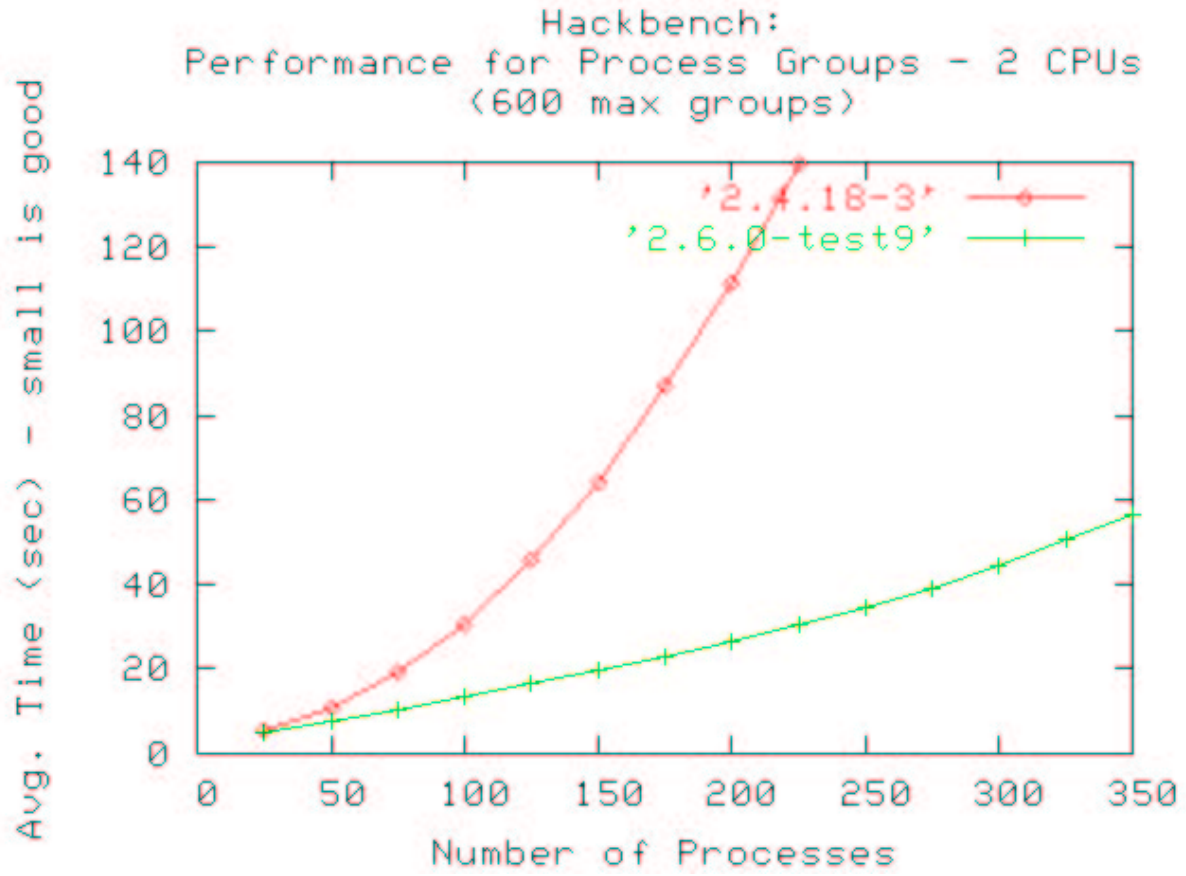
CPU de 1 vía

El siguiente gráfico muestra la ejecución de un máximo de 600 grupos. En un sistema de una única CPU, ambas versiones de Linux fallaron despues de ejecutarse con 200 grupos.



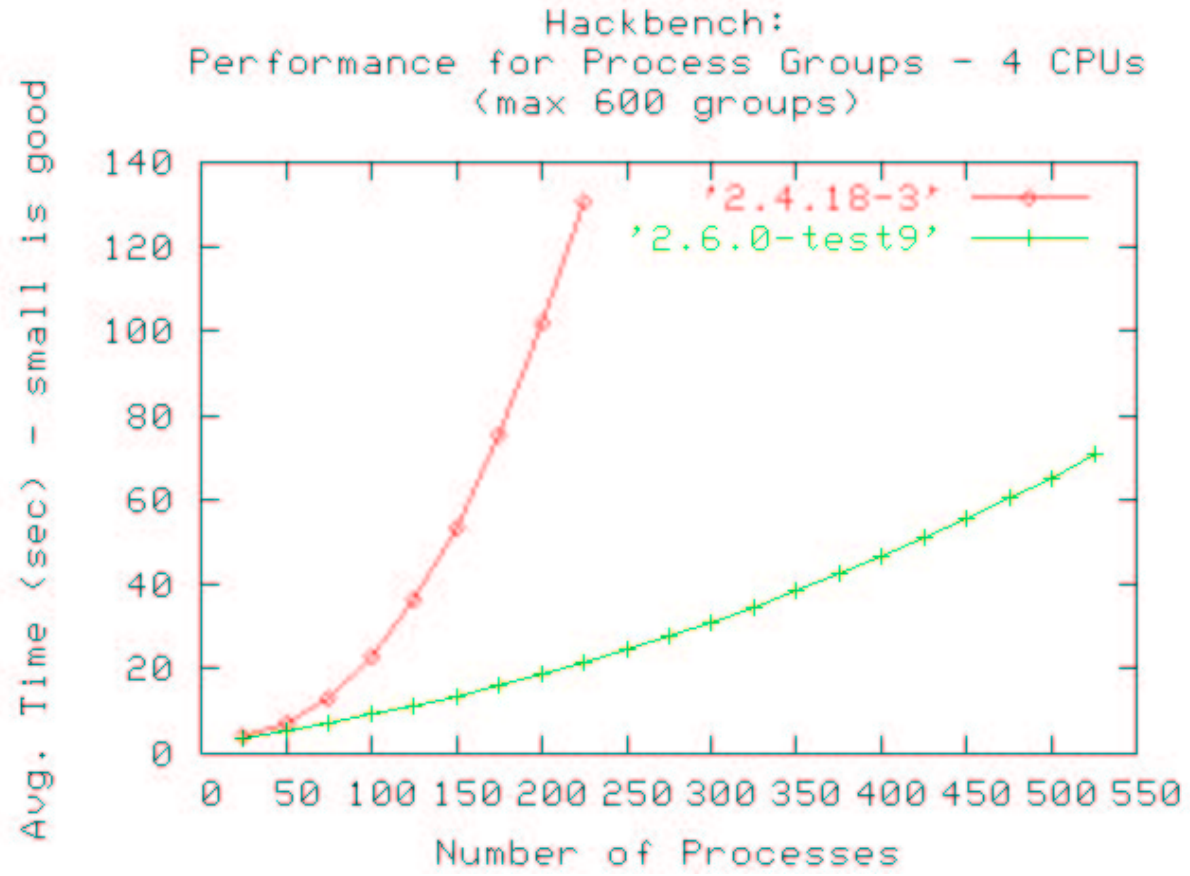
CPU de 2 vías

El siguiente gráfico muestra los resultados de un sistema de dos procesadores. Puede verse que bajo 2.4.18 la cantidad máxima de procesos que pudieron ejecutarse fue 225, pero bajo 2.6.0-test9, la cantidad de procesos que pudieron ejecutarse fue mayor; la prueba fue capaz de ejecutar hasta 350 grupos de procesos.



CPU de 4 vías

El siguiente gráfico muestra los resultados para un sistema de 4 vías. Bajo 2.4.18 la cantidad máxima de procesos capaces de ser creados con este benchmark fue 225, la misma que en un sistema de 2 vías. Bajo 2.6.0-test9, la cantidad de procesos que pudieron ejecutarse con este benchmark se incrementó significativamente a 525.



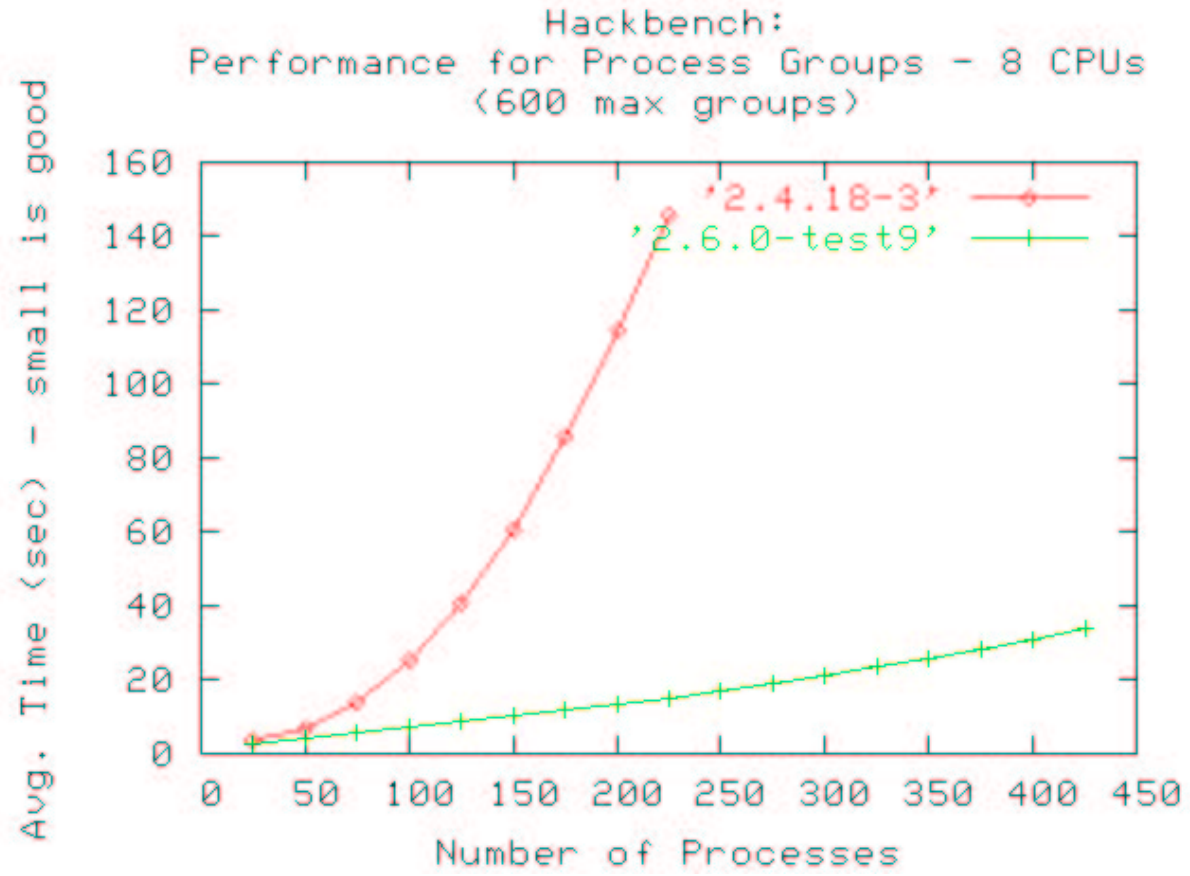
CPU de 8 vías

El siguiente gráfico muestra los resultados para un sistema de 8-vías. Al igual que en el sistema de 4 vías, la cantidad máxima de grupos de procesos bajo 2.4.18 sin llegar a la exhaustividad de recursos fue de 225. Bajo 2.6.0-test9, la cantidad de grupos de procesos que pudieron ejecutarse fue 425. La misma es mejor que en un sistema de 4 vías corriendo 2.6.0-test9, pero es aún superior que con 2.4.18 corriendo en 8 vías.

El resultado de ejecutar la prueba hackbench en un sistema de 8 vías en 2.6.0-test9 con una cantidad mayor que 425 procesos causó un mensaje de error diferente:

SENDER: write (error: No buffer space available)

Este error es reportado por el programa hackbench, mientras intenta escribir en un buffer como preparación para enviar los datos al receptor. Se cree que este error es causado por un espacio de buffer demasiado pequeño asignado en el inicio del sistema de 8 vías. Nuestro sistema de 8 vías tiene más discos y otros dispositivos periféricos que el de 4 vías, lo cual ocasiona que se asigne más espacio al kernel del sistema. Esto deja menor espacio de buffers para el programa hackbench. Asumimos que la prueba podría correr con más grupos de procesos si la configuración del kernel fuera afinada. Los resultados pueden varias dependiendo del hardware específico en el cual se realice la prueba. Sin embargo, los resultados muestran que 2.6.0-test9 en un sistema de 8 vías es aún preferible a 2.4.18.



Conclusión

Linux 2.6.0-test9 es más eficiente con respecto a la planificación de procesos, permite una mayor escalabilidad si se utilizan grandes sistemas para ejecutar un alto número de procesos, y la eficiencia del planificador combinada con otras mejoras permiten que sean ejecutados más procesos en el sistema antes de agotar los recursos.

Nota del traductor

La versión original de este artículo puede encontrarse en http://developer.osdl.org/craiger/hackbench/. Tal como la nota de Copyright precedente lo manifiesta, es posible que al realizar esta traducción esté violando los derechos de OSDL. En mi descargo sólo puedo decir que me tomé la libertad de traducir este artículo para ponerlo al alcance de otras personas que no leen inglés. En todo momento he tratado de ser fiel al texto original.

Javier Smaldone
http://www.smaldone.com.ar