High performance linpack en un cluster del único centro de supercomputación en Colombia reconocido por Europa, SC3UIS.

High performance linpack in a single cluster supercomputing center in Colombia recognized by Europe, SC3UIS.

Angie Natalia Arias Gómez
Escuela de Ingeniería de Sistemas
Universidad Industrial de Santander
angie2172017@correo.uis.edu.co

Samuel Yesid Cadena Pinilla
Escuela de Ingeniería de Sistemas
Universidad Industrial de Santander
samuel2171399@correo.uis.edu.co

Juan Pablo Claro Pérez
Escuela de Ingeniería de Sistemas
Universidad Industrial de Santander
juan2181707@correo.uis.edu.co

Juan Sebastián Díaz Gutiérrez
Escuela de Ingeniería de Sistemas
Universidad Industrial de Santander
Juan2180038@correo.uis.edu.co

Mario Andrés Anaya Merchán
Escuela de Ingeniería de Sistemas
Universidad Industrial de Santander
Mario2180016@correo.uis.edu.co

Resumen

En la computación vemos diferentes problemáticas que se han venido estudiando y entre tantos sale a colación la resolución de operaciones matemáticas complejas bastante frecuentada en este ámbito, para ello el High Performance Linpack o mejor conocido como HPL realiza una excelente labor probando diferentes métodos matemáticos, entre ellos se encuentra “Basic Linear Algebra Subprograms” conocido como BLAS el cual consiste en la definición de un conjunto de rutinas las cuales se componen por operaciones con matrices, también esta presente VSLIP, funciones matemáticas escalares, funciones de matemática compleja, aritmética, histogramas y variedad de métodos.

Abstract

In computing, we see different problems that have been studied and among many, the resolution of complex mathematical operations that is quite popular in this area comes up, for this the High Performance Linpack or better known as HPL does an excellent job testing different mathematical methods, among They are "Basic Linear Algebra Subprograms" known as BLAS which consists of the definition of a set of routines which are composed by operations with matrices, VSLIP is also present, scalar mathematical functions, complex mathematical functions, arithmetic, histograms and variety of methods.

I. Introducción

HPL es una implementación de referencia de Linpack de alto rendimiento. El código resuelve un sistema uniformemente aleatorio de ecuaciones lineales e informa el tiempo y la tasa de ejecución de punto flotante utilizando una fórmula estándar para el recuento de operaciones; HPL está escrito en un ANSI C portátil y requiere una implementación MPI, así como una biblioteca BLAS o VSIPL. Esta elección de dependencias de software le da a HPL tanto portabilidad como rendimiento, HPL es a menudo uno de los primeros programas que se ejecutan en grandes instalaciones de computadoras para producir un resultado que se puede enviar a TOP 500 y este articulo vamos a hablar de nuestra experiencia trabajando con HPL Utilizando una máquina virtual con SO Linux, con 4 núcleos y 4 GB de memoria evaluando el rendimiento de una supercomputadora.

II. Marco teórico

Es importante mencionar que **HPL** es un paquete de software que resuelve un sistema lineal denso (aleatorio) en aritmética de doble precisión (64 bits) en computadoras con memoria distribuida. Por lo tanto, se puede considerar como una implementación portátil y de libre acceso del Benchmark Linpack de Computación de Alto Rendimiento.

El **algoritmo** utilizado por HPL se puede resumir con las siguientes palabras clave: Distribución de datos cíclicos en bloques bidimensionales - Variante de la derecha de la factorización LU con pivoteo parcial de fila con múltiples profundidades de anticipación - Factorización de panel recursivo con búsqueda de pivote y difusión de columna combinado - Varias topologías de transmisión de panel virtual - algoritmo de transmisión de intercambio de reducción de ancho de banda - sustitución hacia atrás con anticipación de profundidad, el paquete HPL proporciona un programa de prueba y temporización para cuantificar la **precisión** de la solución obtenida, así como el tiempo que tomó calcularla. El mejor **rendimiento que** puede lograr este software en su sistema depende de una gran variedad de factores. No obstante, con algunos supuestos restrictivos sobre la red de interconexión, el algoritmo descrito aquí y su implementación adjunta son **escalables** en el sentido de que su eficiencia paralela se mantiene constante con respecto al uso de memoria por procesador.

El paquete de software HPL **requiere** la disponibilidad en su sistema de una implementación de la interfaz de paso de mensajes **MPI**, Una implementación de **cualquiera de** los subprogramas básicas de álgebra lineal **BLAS o** La imagen del vector de señal Procesamiento Biblioteca **VSIPL** también es necesaria. Las implementaciones específicas de la máquina y genéricas de MPI , BLAS y VSIPL están disponibles para una gran variedad de sistemas.

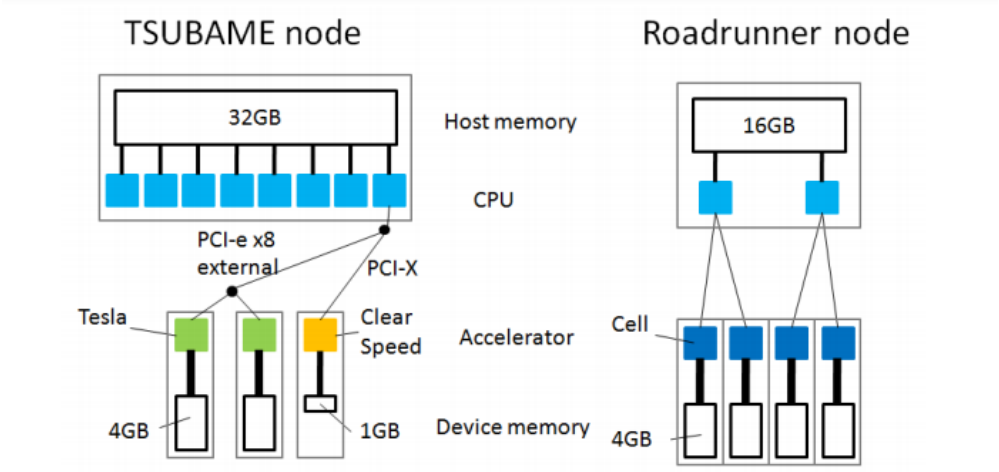


Figura 1. *The overview of architecture of a TSUBAME node with Tesla and a Roadrunner node (Imagén sujeta a derechos de autor)*

	06/'06	11/'06	06/'07	11/'07	06/'08	11/'08	06/'09	11/'09
Speed(TF)	38.18	47.38	48.88	56.43	67.70	77.48	87.01	87.01
Rank	7	9	14	16	24	29	41	56

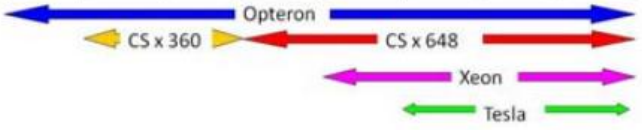


Figura 2. *History of the TSUBAME system in the Top500 ranking. Arrows show the kinds of CPUs/accelerators used in the evaluations ("CS" indicates ClearSpeed accelerators). (Imagén sujeta a derechos de autor)*

Cuando hablamos del proyecto Top500 nos referimos a un ranking de las 500 supercomputadoras con mayor rendimiento del mundo, siento esta lista recopilada por:

- ✓ Hans Meuer, Universidad de Mannheim (Alemania)
- ✓ Jack Dongarra, Universidad de Tennessee (Knoxville)
- ✓ Erich Strohmaier, NERSC/Lawrence Berkeley National Laboratory
- ✓ Horst Simon, NERSC/Lawrence Berkeley National Laboratory

El proyecto se inicia en 1998 y publica una lista actualizada cada seis meses. La primera actualización de cada año se realiza en junio, coincidiendo con la International Conference, y la segunda actualización se realiza en noviembre en la IEEE Supercomputer Conference.

Al inicio de la década de los 90 se necesita una definición del rendimiento de un supercomputador para producir estadísticas comparables. Tras experimentar con diversas métricas basadas en el número de procesadores, en 1992 surge la idea de utilizar un listado de los sistemas en producción como base de la comparativa, en la University of Mannheim.

Un año después, Jack Dongarra se une al proyecto aportando el benchmark Linpack. En mayo de 1993, se crea la primera versión de prueba basada en datos publicados en Internet:

-Serie de estadísticas de supercomputadores de la Universidad de Mannheim (1986-1992).

-Gran cantidad de información recopilada por David Kahaner.

-Listado de los sitios de computación más poderosos del mundo.

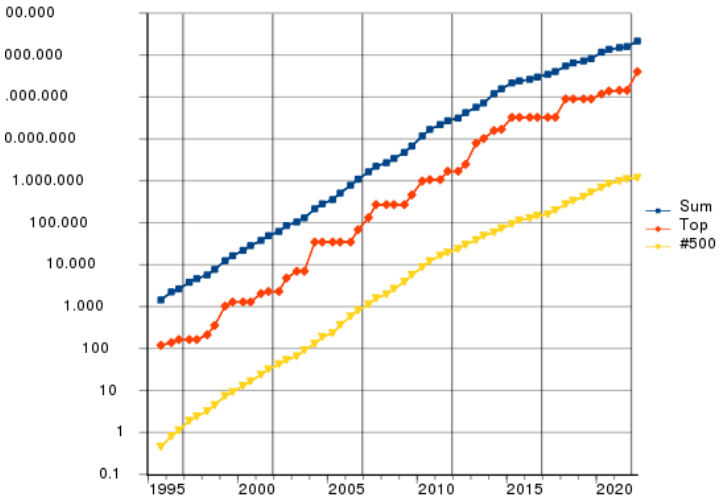


Figura 3.
Crecimiento rápido del rendimiento de las supercomputadoras, basado en los datos del sitio top500.org. (Imagen sujeta a derechos de autor)

Desde 1998, las prestaciones de los sistemas ubicados en el primer puesto han crecido sostenidamente, cumpliendo la ley de Moore y duplicando el rendimiento cada, aproximadamente, 14 meses. En junio de 2013, el sistema más rápido, el Tianhe-2 con un rendimiento pico de 54,9 PFlops, es más de 419.100 veces más rápido que el sistema más rápido en noviembre de 1993, la Connection Machine CM-5/1024 (2048 núcleos) con un pico teórico de 131 GFlops.

Una prueba de rendimiento o comparativa (en inglés benchmark) es una técnica utilizada para medir el rendimiento de un sistema o uno de sus componentes. Más formalmente puede entenderse que una prueba de rendimiento es el resultado de la ejecución de un programa informático o un conjunto de programas en una máquina, con el objetivo de estimar el rendimiento de un elemento concreto, y poder comparar los resultados con máquinas similares. En el ámbito de las computadoras, una prueba de rendimiento podría ser realizada en cualquiera de sus componentes, ya sea la CPU, RAM, tarjeta gráfica, etc. También puede estar dirigida específicamente a una función dentro de un componente, como la unidad de coma flotante de la CPU, o incluso a otros programas. Los benchmarks devuelven información detallada de todas las características que posee y en base a dicha información, se puede evaluar si está completamente optimizado para correr o ejecutar las aplicaciones que necesitamos dependiendo del área.

III. Ítems

- *Openblas*
- *Openmpi*
- *Construcción Hpl*
- *Pruebas Hpl Y Resultados*
- *Comparación De Resultados Con La Supercomputadora Top1*

- ✓ ***Openblas***

Recordando que necesitamos BLAS u alguna version de esta para poder correr el hpl, en nuestro caso utilizamos openblas, es una biblioteca BLAS optimizada basada en la versio´n Gotoblas, que cuenta con muchas optimizaciones hechas a mano para tipos de procesadores especificos, también, agrega implementaciones optimizadas de kernels de álgebra lineal para varias arquitecturas de procesadores.

- ✓ ***Openmpi***

Openmpi es un proyecto de biblioteca de message passing interface (MPI), utilizado por muchas supercomputadoras TOP500, por poner un ejemplo computadoras como la Roadrunner y la computadora K, en años pasados lo han utilizado. Openmpi recolecta lo mejor de otras imple- mentaciones de mpi para destacar en en todas las areas, en nuestro caso este será la variación de mpi con la que trabajaremos.

- ✓ ***¿Cómo se implementa el hpl?***

Para lograr una correcta implementación es necesario instalar las dependencias necesarias pedidas por hpl, que en nuestro caso serán openmpi como variante de mpi y openblas como variante de blas, una vez logrado esto, lo siguiente es descargar y descomprimir el hpl, al hace esto nos quedarán los archivos necesarios, instructivos y demás para la realización de nuestro propio hpl, basándonos en la estructura de cualquier Make en la carpeta setup podemos replicar el nuestro y para ello es necesario saber que función cumple este archivo.

El archivo Make.arch es un archivo que se ubica en el directorio hpl y que contiene los compiladores las librerías con sus paths para poder ser usadas, de otro modo, contiene la ubicación de Blas, mpi, los compiladores, las direcciones locales etc, una vez editado este archivo, ya se puede compilar pues el programa conoce donde se encuentran las dependencias sobre las cuales funciona, así que haciendo un make arch=¡arch¿ se compilara´ el hpl y generará los archivos y carpetas necesarios para modificar el benchmark y poder ejecutarlo.

A partir de este punto lo que queda es localizar la carpeta por nombre arch, generalmente se encuentra dentro de la carpeta hpl/bin, en este directorio se encuentran dos archivos realmente interesantes, el archivo hpl.dat y el archivo xhpl.

Hpl.dat es archivo encargado de especificar las características del problema, tanto el tamaño del problema, como el tipo de algoritmo que se usara´, el número de bloques etc, este archivo.

IV. Trabajando en High performance Linpack

Para medir y conocer las velocidades y capacidades de una infraestructura computacional se realizan benchmarks, son pruebas de rendimiento básicamente, y la verdad hay mucha variedad pues pueden ser específicos a un componente como por ejemplo un benchmark para principalmente conocer el rendimiento de una tarjeta gráfica, el cual no necesariamente pone a prueba otros componentes como el procesador o la memoria, es evidente su utilidad para las compañías que desarrollan hardware pues

gracias a estos dan a conocer sus resultados de rendimiento, permitiéndole al cliente comparar y optar por el producto que mejor satisfaga sus necesidades.

El high performance linpack (hpl) es un benchmark desarrollado y utilizado principalmente para evaluar el rendimiento de las supercomputadoras, y es el principal filtro para poder listar dichas máquinas, de modo que el supercomputador en el puesto 1 es aquel que mejor desempeño y rendimiento obtuvo en el hpl respecto a sus competidores, la razón de su prestigio se debe a que es una implementación portátil y de libre acceso del Benchmark Linpack de la computación de alto rendimiento.

El punto de referencia utilizado en el LINPACK Benchmark es resolver un sistema denso de ecuaciones lineales. Para el TOP500, usamos esa versión del punto de referencia que permite al usuario escalar el tamaño del problema y optimizar el software para lograr el mejor rendimiento para una máquina determinada. Este desempeño no refleja el desempeño general de un sistema dado, como ningún número puede hacerlo nunca. Sin embargo, refleja el rendimiento de un sistema dedicado para resolver un sistema denso de ecuaciones lineales. Dado que el problema es muy regular, el rendimiento alcanzado es bastante alto y los números de rendimiento dan una buena corrección del rendimiento máximo.

Al medir el rendimiento real para diferentes tamaños de problema n, un usuario puede obtener no solo el rendimiento máximo alcanzado Rmax para el tamaño de problema Nmax sino también el tamaño de problema N1 / 2 donde se alcanza la mitad del rendimiento Rmax. Estos números junto con el rendimiento máximo teórico Rpeak son los números dados en el TOP500. En un intento por obtener uniformidad en todas las computadoras en los informes de rendimiento, el algoritmo utilizado para resolver el sistema de ecuaciones en el procedimiento de referencia debe ajustarse a la factorización LU con pivotaje parcial. En particular, el recuento de operaciones para el algoritmo debe ser 2/3 n ^ 3 + O (n ^ 2) operaciones de coma flotante de doble precisión. Esto excluye el uso de un algoritmo de multiplicación de matriz rápida como el "Método de Strassen" o algoritmos que calculan una solución con una precisión menor que la precisión total (aritmética de coma flotante de 64 bits) y refinan la solución utilizando un enfoque iterativo.

En términos generales, el algoritmo tiene las siguientes características:

-Distribución cíclica de datos en bloques 2D
-Factorización de LU utilizando la variante de aspecto correcto con varias profundidades de anticipación
-Factorización de panel recursiva
-Seis variantes de transmisión de panel diferentes
-Algoritmo de transmisión de intercambio de reducción de ancho de banda
-Sustitución hacia atrás con anticipación de profundidad 1.

Se dice que el punto de referencia LINPACK tuvo éxito debido a la escalabilidad de HPL, el hecho de que genera un solo número, lo que hace que los resultados sean fácilmente comparables y la extensa base de datos históricos que tiene asociada. Sin embargo, poco después de su lanzamiento, el punto de referencia LINPACK fue criticado por proporcionar niveles de rendimiento "generalmente inalcanzables para todos menos unos pocos programadores que optimizan tediosamente su código para esa máquina y esa máquina sola", porque solo prueba la resolución de sistemas lineales densos, que no son representativos de todas las operaciones que se realizan habitualmente en la informática científica.

HPL necesita contar con, una interfaz de paso de mensajes (MPI) para realizar la conexión entre nodos y obtener el mejor rendimiento, y una implementación de subprogramas básicos de álgebra linear (BLAS) o también La imagen del vector de señal Procesamiento Biblioteca VSIPL para la parte algebraica y operacional, pues estas librerías permiten realizar este tipo de cálculos.

Los resultados obtenidos con el hpl dependen de varios factores, de hecho el benchmark con diferentes librerías obtiene un resultado distinto, por ejemplo el trabajar con librerías estándares hace que el rendimiento no sea el potencialmente alcanzable por una arquitectura, si tenemos una arquitectura intel o amd, una BLAS optimizada a dicha arquitectura dará paso a mejores resultados, en las supercomputadoras no hay problema con esto pues cada máquina corre el benchmark lo más optimizado posible para explotar el hardware y demostrar de qué están hechas, pero es importante tenerlo en cuenta sobre todo en la parte local.

La manera en la que hpl nos permite conocer el rendimiento de una infraestructura computacional es gracias a los gflops obtenidos y el tiempo que tarda en resolver el sistema,sabiendo esto, vamos a realizar la implementación del hpl en nuestra máquina local, en un par de nodos del supercomputador de nuestra universidad (UIS) y compararemos los resultados.

Para nuestra máquina local contamos con un procesador Ryzen 5 3600, y 16 GB de memoria ram, utilizando un único nodo.

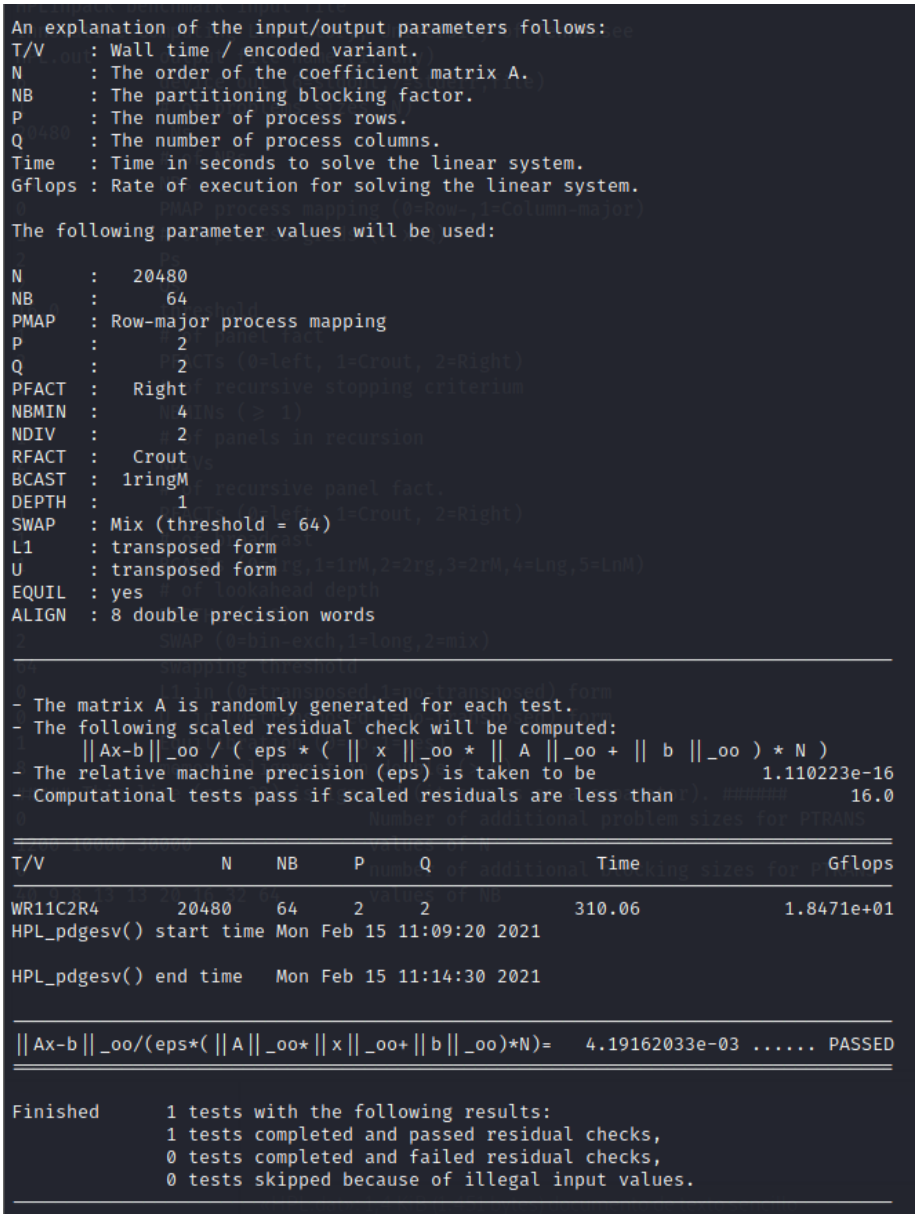
Utilizando una máquina virtual con SO linux, con 4 nucleos y 4 GB de memoria, necesitamos instalar mpi,blas y hpl, una vez realizada la instalación de estos programas debemos crear un Make.<arch> con la estructura dada en hpl, es útil entonces utilizar uno de estos Makes realizados y realizar las modificaciones correspondientes para que nuestro Make.<arch> pueda encontrar la ruta a los demás programas necesarios, mpi, blas el compilador, etc.

Una vez compilado el hpl, es necesario configurar el archivo HPL.dat, este archivo define las características del problema, es con este archivo con el que debemos jugar para conseguir un mejor rendimiento.

Con esta configuración obtenemos un buen rendimiento en el test que realiza nuestro hpl, teniendo en cuenta que es una máquina virtual poco potente, por lo general se obtienen resultados de 1.5 +- 0.3 para distintas configuraciones, siendo esta una de las mejores configuraciones obtenidas, los resultados se encuentran en la figura de abajo.

Se realiza un único test, con un tamaño del problema bastante considerable, y es esta la razón por la cual al sistema le cuesta tanto tiempo resolver el problema, por lo general los Gflops obtenidos se mantienen en el rango especificado anteriormente, pero es bueno ver que aun con un tamaño de problema muy elevado el rendimiento se mantiene dentro de los parámetros calculados.

❖ ***Resultados hpl***



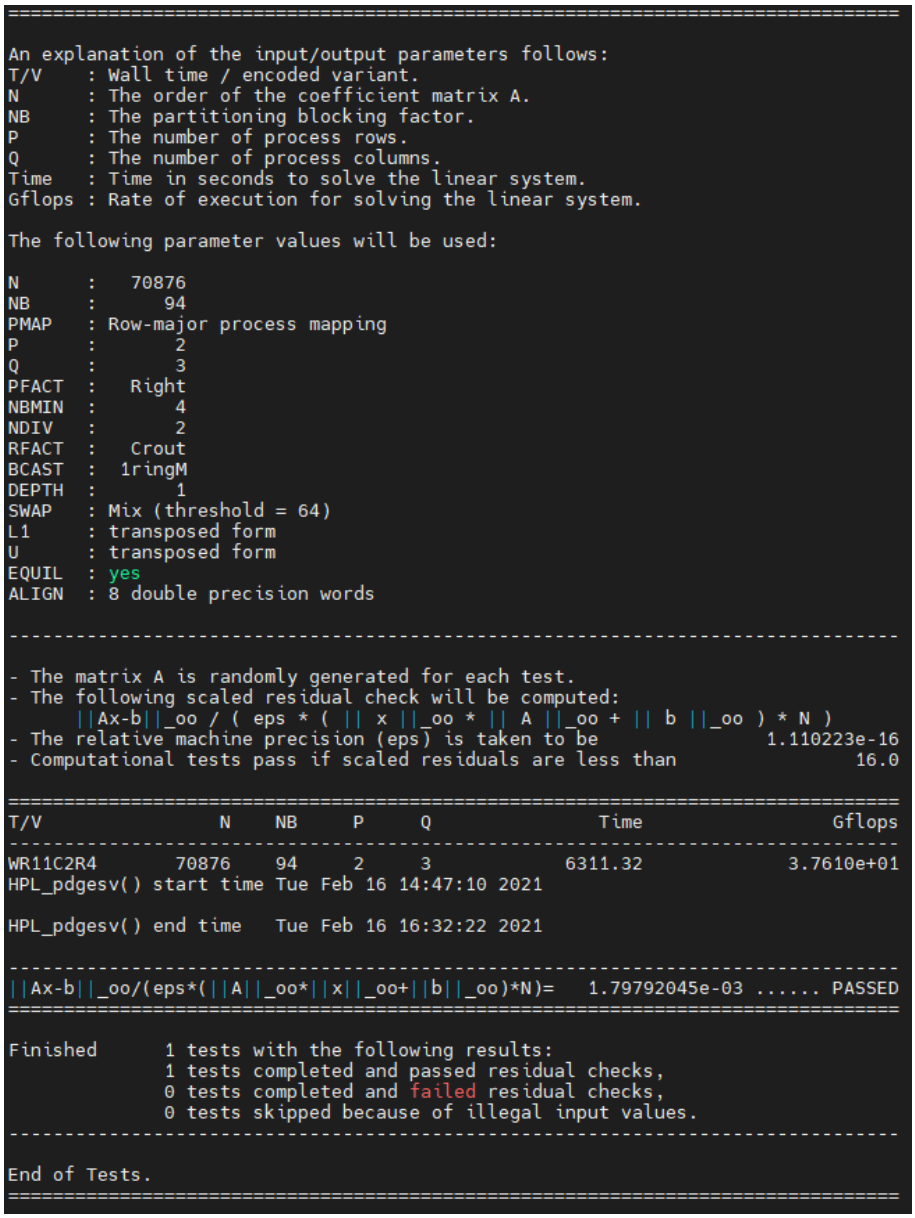
```
An explanation of the input/output parameters follows:
T/V    : Wall time / encoded variant.
N      : The order of the coefficient matrix A.
NB     : The partitioning blocking factor.
P      : The number of process rows.
Q      : The number of process columns.
Time   : Time in seconds to solve the linear system.
Gflops : Rate of execution for solving the linear system.

The following parameter values will be used:

N      :   20480
NB     :      64
PMAP   : Row-major process mapping
P      :       2
Q      :       2
PFACT  :   Right
NBMIN  :       4
NDIV   :       2
RFACT  :   Crout
BCAST  :  1ringM
DEPTH  :       1
SWAP   : Mix (threshold = 64)
L1     : transposed form
U      : transposed form
EQUIL  : yes
ALIGN  : 8 double precision words

--------------------------------------------------------------------------------

- The matrix A is randomly generated for each test.
- The following scaled residual check will be computed:
      ||Ax-b||_oo / ( eps * ( || x ||_oo * || A ||_oo + || b ||_oo ) * N )
- The relative machine precision (eps) is taken to be               1.110223e-16
- Computational tests pass if scaled residuals are less than                16.0

================================================================================
T/V                N    NB     P     Q               Time                 Gflops
--------------------------------------------------------------------------------
WR11C2R4       20480    64     2     2             310.06             1.8471e+01
HPL_pdgesv() start time Mon Feb 15 11:09:20 2021

HPL_pdgesv() end time   Mon Feb 15 11:14:30 2021

--------------------------------------------------------------------------------
||Ax-b||_oo/(eps*(||A||_oo*||x||_oo+||b||_oo)*N)=   4.19162033e-03 ...... PASSED
================================================================================

Finished      1 tests with the following results:
              1 tests completed and passed residual checks,
              0 tests completed and failed residual checks,
              0 tests skipped because of illegal input values.
--------------------------------------------------------------------------------
```

Resultados obtenidos en un un nodo de la uis, el cual no es ni de lejos el mejor computador usable, pero debido a la disponibilidad a fecha de publicación no se alcanzó a probar en el supercomputador de la uis, sin embargo es una maquina más potente que la máquina local por lo tanto es buena comparación, las características principales de esta máquina de un solo nodo son los 6 núcleos que posee, es decir 2 más que en nuestra máquina local, también la gran capacidad de memoria la cual supera las 50 GB, lo cual es unas 12 veces más capacidad que la de nuestra máquina local.

Con esta información presente, modificamos el archivo hpl.dat para que se adapte a las condiciones del hardware, esto significa que algunos valores aumentarán, al poseer esta máquina mayor memoria, el tamaño del problema puede aumentar y es precisamente una de las modificaciones que hicimos, la factorización sigue siendo de la derecha y en este caso las proporciones del p y el q cambian de 2-2 a 2-3, y los tamaños de bloque pasan a ser de 94, y esta configuración realmente puso a prueba la máquina,

cosa que se ve en el tiempo que tardó en resolver el problema pues no fueron exactamente unos cuantos minutos, el Benchmark tardó 1.75 horas en completarse, en cuanto al rendimiento de la máquina obtenemos unos Gflops de 37.6, y acá está lo interesante porque el rendimiento obtenido es poco más del doble que el rendimiento de nuestra máquina local, el equipo de cómputo de más bajas prestaciones del centro de supercomputación UIS es 2 veces superior a nuestra máquina.

Los resultados del Benchmark en la figura de abajo.



Analizando las máquinas top de las top500

Para poder tener un marco de referencia para el hpl, tomaremos en cuenta las supercomputadoras top del top500, el proyecto Top500 es un ranking de las 500 supercomputadoras con mayor rendimiento del mundo, la última lista actualizada es la de noviembre del año pasado y los resultados son realmente fascinantes, por propósitos del proyecto vamos a tener en cuenta únicamente el rendimiento en el Benchmark HPL.

Para dar un poco de contexto acerca del nivel de este top, compararemos los resultados que nosotros obtuvimos con la máquina #500 del top, solo para ver qué tan diferentes serán los resultados. En nuestra máquina obtuvimos un rendimiento de 18.471 Gigaflops por segundo, mientras que la máquina que ocupó el puesto 500 tuvo un rendimiento máximo de 1’316,800.0 Gigaflops por segundo, esto indica que la máquina menos potente del top es 71290.13 veces más potente que la nuestra, ahora bien, comparando los resultados del nodo uis, obtuvimos un rendimiento de 37.6 Gigaflops por segundo, lo cual es el doble de nuestra máquina local, pero en relación con la máquina 500 del top, sigue siendo muy inferior pues la máquina 500 es unas 35645 veces más potente, ahora imaginen la máquina en el puesto 1 del top.

Si Colombia quisiera colocar una máquina dentro de las 10 primeras del Top500 de Junio de 2021, este deberá estar equipado con una increíble cantidad de núcleos para que el procesador pueda ejecutar la mayor cantidad de operaciones a la vez de manera fluida y muchas gigas de memoria que el flujo de datos no se detenga, en el Benchmark HPL lo más importante es la velocidad pico de un procesador y el número de procesadores que trabajan en conjunto, pero una mayor cantidad de núcleos no significa inmediatamente tendrá un pico de velocidad más alto, pues en Noviembre 2020 la máquina #1 tenía 7.630.848 núcleos y el #4 tenía 10.649.600 pero la velocidad máxima fue 442.010,0 Tflops y 93.014.6 Tflops respectivamente, esto demuestra que no solo los núcleos afectarían la velocidad una máquina, sino también la frecuencia por núcleo que maneja cada infraestructura computacional y la manera en la que están conectados. La velocidad máxima que demostró el #10 del Top500 de Noviembre 2020 fue 22.400 Tflops, entonces según la media de aumento en cada edición de la lista Colombia debería como mínimo apuntar a superar esta marca por unos 2000 Tflops, es decir apuntar a lograr un rendimiento máximo de 24.400 Tflops.

V. Conclusiones

- La cantidad de cores influye fuertemente en el rendimiento que tendrá una maquina en el benchmark hpl pero no es el único componente que marca dicho rendimiento, también se deben tener en cuenta otros aspectos como, la frecuencia a la que trabajan los cores de los procesadores, la optimización del software como, el blas, o el openmpi para realizar la conexión entre nodos de manera más eficiente.

- El archivo hpl.dat, que corresponde a las especificaciones del problema a resolver, depende de la arquitectura de la máquina, aspectos como el tamaño del problema están condicionados por la memoria y el blocksize, a su vez el p y el q varían en relación a la cantidad de cores.

- En la lista del top500 se ve como las potencias mundiales suelen aparecer varias veces en el top, mientras que países latinoamericanos son muy escasos por no decir que no están presentes, la supercomputación no se encuentra tan desarrollada en estos países, sin embargo, es bastante enriquecedor saber que en nuestra universidad contamos con el único centro de supercomputación en Colombia reconocido por Europa.

- Es una pena no mostrar resultados de la capacidad de cómputo ofrecida por el supercomputador UIS, sin embargo podemos especular un poco, como estudiantes tenemos acceso máximo a dos nodos, los cuales cuentan con 24 cores, en total unos 48 cores, así como un total de 200GB de memoria, es poco en comparación a instalaciones realmente dedicadas únicamente al benchmark

pero es lo que tenemos, con estas características el hpl.dat tendría alrededor de 140000 de tamaño del problema, el tiempo en resolver el sistema sería de horas y el resultado en Gflops sería de cientos, lo cual aplasta totalmente a nuestras máquinas ya mencionadas en el artículo.

VI. Referencias

- Dongarra, JJ, Luszczek, P., Petitet, A .: The LINPACK Benchmark: Pasado, presente y futuro, concurrencia y computación: práctica y experiencia 15 (2003)Google Académico

- Petitet, A., Whaley, RC, Dongarra, J., Cleary, A .: HPL: una implementación portátil del parámetro de referencia Linpack de alto rendimiento para computadoras con memoria distribuida, http://www.netlib.org/benchmark/hpl /

- Meuer, HW, Strohmaier, E., Dongarra, JJ, Simon, HD: Top500 Supercomputer Sites, 17ª ed., 2 de noviembre (2001). El informe se puede descargar de http://www.netlib.org/benchmark/ top500.html

- Zhang, BL, et al .: Teoría y método de computación paralela numérica. Prensa de la industria de defensa nacional, Beijing (1999)Google Académico

- Lin, CS: método de cálculo numérico (columna A). Prensa científica, Beijing (1998)Google Académico

- Chen, GL: Computación paralela: estructura, algoritmo, programación (versión modificada), p. 8. Prensa de educación avanzada, Beijing (2003)Google Académico

- Sun, ZZ: Análisis numérico, 2ª ed., Pág. 1. Prensa universitaria del sudeste, Nanjing (2002)Google Académico http://www.cs.utk.edu/~dongarra/WEB-PAGES/SPRING-2000/lect08.pdf

- Wenli, Z., Jianping, F., Mingyu, C .: Determinación eficiente del tamaño del bloque NB para la prueba de Linpack en paralelo. En: Actas de la Conferencia Internacional IASTED sobre Sistemas y Computación Paralelos y Distribuidos (PDCS 2004). MIT, Cambridge (2004) (recibido)Google Académico