Análisis del Uso de un Cluster de Raspberry Pi para Cómputo de Alto Rendimiento

Sebastián Rodríguez Eguren, Franco Chichizola, and Enzo Rucci

III-LIDI, CEA-CIC, Facultad de Informática, Universidad Nacional de La Plata
La Plata (1900), Buenos Aires, Argentina
Email: {seguren,francoch,erucci}@lidi.info.unlp.edu.ar

Resumen En la actualidad, la mejora de la eficiencia energética se ha vuelto uno de los principales desafíos para el cómputo de alto rendimiento (HPC). La Raspberry Pi (RPi) es una *Single Board Computer* de bajo costo, la cual cuenta con múltiples núcleos, capacidad para conectarse en red y soporte a sistemas operativos y herramientas de programación paralela tradicionales. Aunque sus núcleos no son potentes, el bajo consumo de potencia las vuelven una opción atractiva desde el punto de vista energético. En este trabajo se analiza la viabilidad de usar un cluster de placas RPi para HPC. Mediante la ejecución de diferentes aplicaciones paralelas con alta demanda computacional, se identificaron fortalezas y debilidades del cluster RPi. Adicionalmente, se comparó su rendimiento y eficiencia energética con el de un cluster x86.

Keywords: Raspberry, RPi, x86, HPC, Eficiencia energética

1. Introducción

En la actualidad, los grandes sistemas HPC son dominados por procesadores de propósito general con dos conjuntos de instrucciones (x86 y Power) de tres empresas vendedoras (Intel, AMD e IBM) [5]. Estos procesadores han sido diseñados especialmente para ofrecer un alto rendimiento por núcleo, pero con costo económico (de cientos a miles de dólares) y demanda de potencia acordes (algunos puede requerir más de 150W). En la búsqueda de construir sistemas más grandes que nos permitan llegar a la escala de los Exaflops, resulta fundamental explorar otras estrategias que ofrezcan tasas de rendimiento costo-eficientes y que sean capaces de reducir el consumo energético de estos sistemas [24].

Durante décadas, el desarrollo de plataformas HPC estuvo focalizado casi únicamente en mejorar el rendimiento de las mismas. Esto provocó un crecimiento exponencial en los requerimientos de potencia de estos sistemas (no sólo para alimentarlos sino también para refrigerarlos), lo que a su vez repercutió en el costo económico de los mismos. Es por eso que, hoy en día, la reducción del consumo energético se ha vuelto uno de los principales desafíos para la comunidad HPC.

Entre las diferentes alternativas actualmente exploradas, se encuentra el uso de procesadores de bajo consumo como los ARM. Esta clase de procesadores ha sido especialmente diseñada para el mercado de dispositivos móviles y los sistemas embebidos. Recientemente, han entrado al mercado de PCs y servidores y se espera que en el futuro incrementen significativamente su poder de cómputo manteniendo el bajo consumo de potencia [18]. Para la comunidad HPC, este escenario presenta una oportunidad de emplearlos para construir sistemas de alto rendimiento. En los últimos años, el uso de procesadores ARM se ha vuelto popular entre desarrolladores y entusiastas de la programación debido al surgimiento de *Single Board Computer*'s (SBC) como la Beagleboard [4], la Raspberry Pi [7] y la Pandaboard [3]. Esto ha llevado a la formación de varias comunidades de desarrollo que invierten tiempo y esfuerzo en portar una amplia gama de sistemas operativos, aplicaciones, librerías y herramientas a los sistemas ARM más populares [9].

La RPi es una SBC de bajo costo desarrollada en el Reino Unido por la Fundación Raspberry Pi, con el objetivo de estimular la enseñanza de las ciencias de la computación en las escuelas. Desde su surgimiento, la RPi ha sido adoptada masivamente para una amplia gama de aplicaciones debido a su alto cociente rendimiento/precio, su tamaño reducido (similar al de una tarjeta de crédito) y su versatilidad [11]. En particular, desde el punto de vista del procesamiento paralelo, las RPi tienen múltiples núcleos, pueden conectarse en red y dan soporte a sistemas operativos y herramientas de programación tradicionales de HPC. Aunque sus núcleos no son potentes, el bajo consumo de potencia de las RPi (˜3.5W) las vuelven una opción atractiva desde el punto de vista energético. Es por eso que resulta interesante analizar la viabilidad del uso de un cluster de RPi para HPC.

El resto del documento se organiza de la siguiente forma: la Sección 2 describe las características del cluster de RPi utilizado en este estudio mientras que la Sección 3 detalla las aplicaciones benchmark seleccionadas. La Sección 4 analiza los resultados obtenidos, seguido de la Sección 5 que estudio los trabajos relacionados. Finalmente, la Sección 6 presenta las conclusiones y las posibles líneas de trabajo futuro.

2. Cluster de RPi

Existe en el mercado una amplia gama de variantes de las placas RPi. La familia original tiene 3 versiones. La primera RPi (versión 1) tuvo varios modelos: A, B y B+, siendo B el más utilizado. La versión 2 sólo tuvo un modelo (B). Por su parte, la versión 3 también tuvo durante un largo tiempo un único modelo (B), siendo presentado recientemente su sucesor (B+) en marzo de 2018.

Para esta investigación se ha desplegado un cluster de 20 placas RPi 3 modelo B, las cuales se encuentran interconectadas por un switch 3COM 3C16475BS y alimentadas por una fuente estándar para PC de 250W. Como software de base, las placas cuentan con el sistema operativo Raspbian Stretch Lite. En la Figura 1 se presenta una RPi 3 B mientras que en la Figura 2 se muestra una foto del cluster utilizado en este trabajo.

La RPi 3 está conformada por un *System-on-Chip* (SoC) especialmente desarrollado para este modelo (Broadcom BCM2837), el cual incluye 4 núcleos de procesamiento ARM Cortex-A53 de alto rendimiento que funcionan a 1.2GHz con 32Kb de cache de nivel 1 y 512Kb de nivel 2. Además, cuenta con un procesador gráfico VideoCore IV que se encuentra conectado a un módulo de memoria LPDDR2 de 1 GB en la parte posterior de la placa.

Este modelo comparte el mismo chip SMSC LAN9514 que su predecesor, la RPi 2, agregando conectividad Ethernet 10/100 y cuatro canales USB a la placa. El chip SMSC se conecta al SoC a través de un único canal USB, que funciona como un adaptador USB-a-Ethernet y un concentrador USB. También se puede encontrar el chip Broadcom BCM43438 que proporciona conectividad LAN inalámbrica de 2.4GHz 802.11n y Bluetooth 4.1 de bajo consumo.

3. Aplicaciones Benchmark

Para poder realizar el análisis de rendimiento y eficiencia energética, se seleccionaron 3 aplicaciones que se pueden considerar clásicas y representativas de aplicaciones científicas, que poseen alta demanda computacional pero con diferentes características entre sí. Las implementaciones fueron desarrolladas en lenguaje C con la librería MPI. A continuación, se describen las características distintivas de cada una de las aplicaciones junto a las técnicas de programación y optimización realizadas en cada una de las aplicaciones paralelas.

Multiplicación de Matrices (MM). La implementación paralela sigue el modelo *master-worker*, donde uno de los procesos es responsable de la distribución del trabajo y la recuperación de los resultados parciales. Para computar $C = A \times B$, el *master* envía a cada proceso (inclusive a sí mismo) un bloque de filas de la matriz A y la matriz B en forma completa. Luego, cada proceso será responsable de calcular un bloque de filas de C computando las celdas de a bloques cuadrados para maximizar la localidad de datos. Por último, resulta importante mencionar que los procesos emplean operaciones de comunicación colectiva para el intercambio de datos (`MPI_Scatter` para distribuir A, `MPI_Bcast` para enviar B y `MPI_Gather` para recuperar los resultados parciales de C).

Sintéticamente, MM es una aplicación que demanda gran cantidad de cómputo regular y que requiere poca cantidad de comunicaciones al inicio y al final de cada tarea, pero de gran tamaño.

Figura 1: RPi 3 B

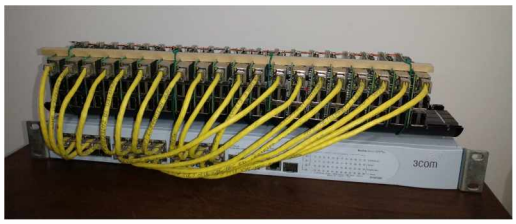

Figura 2: Cluster de RPi 3 B

Método de Jacobi. Jacobi es un método iterativo para resolver sistemas de ecuaciones lineales. Durante un número determinado de pasos, se actualizan dos matrices (*M1* y *M2*) en forma alternada. Esto significa que en el paso *i*, *M1* se actualiza leyendo los valores de *M2* mientras que, en el paso *i+1*, *M2* se actualiza a partir de los valores de *M1*. El nuevo valor de una celda está definido por el promedio de cuatro vecinos (el de arriba, el de abajo, el de la izquierda y el de la derecha).

La implementación paralela sigue el modelo *Single Program Multiple Data* (SPMD). Cada proceso maneja un bloque de filas de cada matriz y necesita de espacio adicional para mantener las filas de los límites de sus vecinos. En cada paso, cada proceso actualiza el bloque de filas correspondiente y, a continuación, intercambia con sus vecinos las filas de los límites de sus bloques.

En resumen, Jacobi es una aplicación regular con alto grado de paralelismo y de dependencia a nivel de datos entre las distintas tareas y etapas de ejecución. Esta gran cantidad de sincronización entre las tareas genera una considerable cantidad de comunicaciones de moderado tamaño a lo largo de toda la ejecución.

N-Reinas. Este problema se basa en determinar la cantidad de formas diferentes de ubicar N reinas en un tablero de N×N sin que se amenacen (no pueden convivir dos reinas en una misma fila, columna o diagonal). En la implementación paralela se generan todos los posibles tableros iniciales (tableros con las 4 primeras reinas ubicadas) y se los distribuye bajo demanda entre los procesos utilizando el modelo *master-worker*. El proceso *master* se encarga únicamente de distribuir estos tableros y recibir los resultados. Cuando un *worker* recibe un tablero inicial para procesar, calcula todos las formas válidas de completar las N reinas en ese tablero (el trabajo para realizar ese cálculo varía considerablemente dependiendo del tablero inicial) y le responde al *master*.

En síntesis, N-Reinas es una aplicación con alto grado de paralelismo, pero donde las tareas son irregulares y de gran tamaño, lo que obliga a realizar la distribución dinámica de las mismas entre los procesos. Esto genera una gran cantidad de comunicaciones no estructuradas de tamaño pequeño.

4. Resultados Experimentales

4.1. Diseño Experimental

Para realizar la experimentación se utilizó el cluster de RPi descrito en la Sección 2. Además, también se usó un cluster de computadoras x86, donde cada nodo cuenta con un procesador Intel Core i5-4460 3.20Ghz, 8 GB RAM y conexión de red Gigabit Ethernet. Las pruebas fueron realizadas en tres etapas:

- El objetivo de la primera etapa fue determinar los tiempos de ejecución de los algoritmos secuenciales, variando el tamaño de problema. En el caso particular de MM, además se probaron diferentes tamaños de bloque a fin de obtener el óptimo, el cual se utilizó posteriormente para correr las pruebas paralelas.

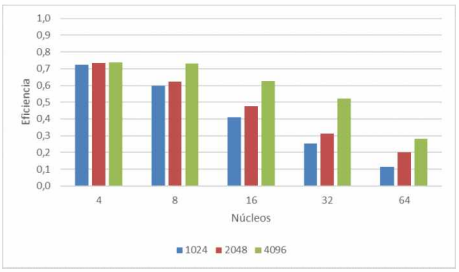

Figura 3: Valores de Eficiencia obtenidos para MM en el cluster RPi

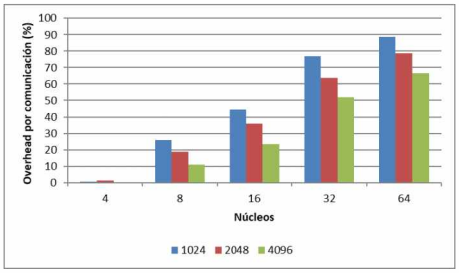

Figura 4: Valores de Overhead obtenidos para MM en el cluster RPi

- La segunda etapa de pruebas se basó en determinar los tiempos de ejecución de los algoritmos paralelos en el cluster de RPi.
- En la tercera etapa se ejecutaron los algoritmos paralelos en el cluster de computadoras x86.

Cada prueba fue repetida 10 veces y los valores reportados corresponden a la media de los datos obtenidos.

4.2. Resultados en el cluster RPi

Para evaluar el comportamiento de las aplicaciones benchmark al escalar el problema (N) y/o la arquitectura (P) se analiza la Eficiencia. En la Figura 3 se grafican los valores de Eficiencia obtenidos para MM. Se puede apreciar como la Eficiencia se incrementa a medida que crece el tamaño del problema para un determinado número de núcleos y se degrada al incrementar el numero de núcleos para un tamaño de problema dado. Este último decremento se debe principalmente al overhead que generan las comunicaciones, ya que a medida que aumenta P, se incrementan las interacciones entre procesos.

Resulta interesante analizar la incidencia de las comunicaciones en el rendimiento final considerando que la interacción entre procesos limita la Eficiencia alcanzada en las pruebas realizadas. En la Figura 4 se ilustran los porcentajes de tiempo de ejecución dedicados a comunicación para cada conjunto de pruebas realizadas de la aplicación MM. Se puede notar que, al aumentar P y mantener fijo N, el porcentaje de comunicación se incrementa. A la inversa, el porcentaje de comunicación disminuye cuando N aumenta y P se mantiene. También se observan porcentajes elevados de overhead que obviamente penalizan el rendimiento. Esto se debe a la velocidad de la interfaz de red de la RPi (100Mbit), lo cual impone una fuerte limitación al rendimiento final de la aplicación considerando el volumen de las comunicaciones. La excepción es cuando $P = 4$, ya que no se genera comunicación *real* por la red (se utiliza una única RPi).

Resulta importante mencionar que, aunque se probó con tamaños de matrices más grandes, no fue posible reportar sus resultados debido a que el requerimiento de memoria supera ampliamente el espacio disponible en cada nodo del cluster, lo que produce fallas en la ejecución.

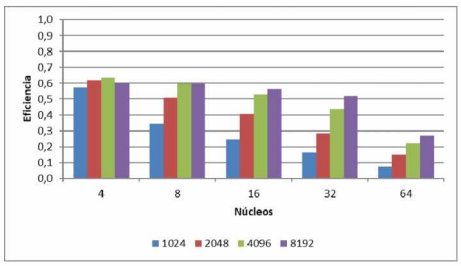

Figura 5: Valores de Eficiencia obtenidos para Jacobi en el cluster RPi

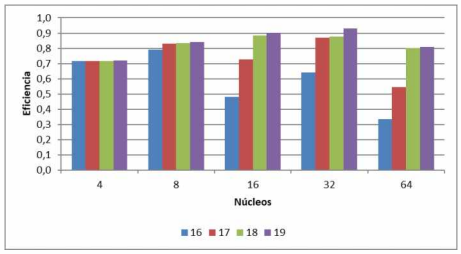

Figura 6: Valores de Eficiencia obtenidos para N-Reinas en el cluster RPi

En la Figura 5 grafican los valores de Eficiencia obtenidos para Jacobi. De los resultados se aprecia un comportamiento similar al caso MM. Al mantener fijo el tamaño de la matriz y aumentar la cantidad de procesadores, la Eficiencia disminuye debido al overhead de comunicación que se al comunicar los bordes de la matriz a cada proceso y, en menor medida, a la distribucin inicial de datos. Al escalar N manteniendo fijo P, la Eficiencia mejora en general, dado que el overhead mencionado anteriormente tiene menor peso en el tiempo total de ejecución.

Se debe notar que los valores de Eficiencia obtenidos con Jacobi son inferiores a los correspondiente a MM. Esto es debido a que, aunque manejen estructuras de datos más pequeñas, deben realizarse más comunicaciones (de pocos datos) y la relación cómputo-comunicación es mucho mas baja. Además, también es importante mencionar que existen muchos puntos de sincronización que producen overhead (espera ociosa).

En la Figura 6 se presentan los valores de Eficiencia obtenidos para N-Reinas. Es posible notar un incremento de la Eficiencia a medida que aumenta N, lo cual se debe a que el componente paralelo del algoritmo crece a un ritmo mayor que el componente serie al aumentar el tamaño del tablero. También se logra un incremento en la Eficiencia -más suave- al incrementar el número de núcleos hasta 32, que luego se reduce (P= 64). Este comportamiento no se da en todos los casos debido a la irregularidad propia de la aplicación.

Cabe recalcar que de las pruebas realizadas, N-Reinas fue la que mejores resultados arrojó (valores de Eficiencia cercanos a 1) y sus causas son múltiples. En primer lugar, la aplicación trabaja con un conjunto reducido de datos, de tamaño mucho menor al del resto de los benchmarks empleados. En segundo lugar, si bien requiere una gran cantidad de comunicaciones, las mismas son de pocos datos y no representan un porcentaje de tiempo significativo respecto al tiempo total de la aplicación. Por último, el algoritmo trabaja con números enteros a diferencia de los anteriores que computan en punto flotante.

4.3. Comparación con cluster x86

Para poder comparar el rendimiento entre los diferentes clusters se requiere una métrica común. Es por eso que se seleccionó la métrica GFLOPS para MM y

P	N		
	1024	2048	4096
4	0,43	0,51	0,51
8	0,72	0,87	1,01
16	0,98	1,32	1,74
32	1,21	1,73	2,88
64	1,09	2,24	3,12
GFLOPS			

(a) MM

P	N			
	1024	2048	4096	8192
4	0,74	0,68	0,67	0,64
8	0,89	1,12	1,27	1,27
16	1,27	1,79	2,23	2,39
32	1,70	2,51	3,71	4,41
64	1,53	2,64	3,71	4,61
GFLOPS				

(b) Jacobi

P	N			
	16	17	18	19
4	69,95	70,06	70,22	70,66
8	154,63	162,51	162,88	164,92
16	188,12	284,56	345,50	353,21
32	501,28	680,00	687,74	728,97
64	522,59	852,10	1257,04	1266,97
MQUEENS				

(c) N-Reinas

P	N		
	1024	2048	4096
4	7,85	7,99	7,99
8	7,19	12,09	13,86
16	7,17	13,73	17,43
32	7,78	11,32	23,58
64	5,02	12,95	32,62
GFLOPS			

(d) MM

P	N			
	1024	2048	4096	8192
4	6,88	3,76	3,84	3,84
8	6,88	7,48	7,61	7,67
16	6,61	20,01	23,79	23,83
32	5,36	27,13	45,18	47,96
64	5,73	27,92	75,07	86,71
GFLOPS				

(e) Jacobi

P	N			
	16	17	18	19
4	327,88	327,27	329,12	331,14
8	679,39	750,89	766,08	770,29
16	888,43	1284,37	1347,85	1358,97
32	1035,69	2259,81	2697,68	2782,20
64	554,17	2753,42	5014,06	5623,87
MQUEENS				

(f) N-Reinas

Tabla 1: Rendimiento de las aplicaciones benchmark en el cluster RPi (a-c) y en el x86 (d-f).

Jacobi. Sin embargo, en el caso de N-Reinas, al ser una aplicación que computa con números enteros, se empleó una métrica *ad-hoc*. Esta métrica considera el número total de reinas posicionadas y se la denominó MQUEENS (millones de reinas posicionadas).

Como se puede apreciar en la Tabla 1, en ambos cluster y para todas las aplicaciones, el rendimiento global del sistema mejora al aumentar tanto el tamaño del problema como de la arquitectura. Como era de esperar, el cluster x86 obtiene rendimientos superiores en las 3 aplicaciones. Esto se debe a que sus procesadores son más potentes y sofisticados, sus memorias RAM son más rápidas, su interfaz de red es más veloz, entre otras características más avanzadas de este cluster. Por otra parte, a nivel de aplicación, la mayor diferencia se encuentra en MM con una diferencia promedio de 10.4× y una máxima de 18.2×, seguido por Jacobi con promedio de 9× y máxima de 18.8×; y por último, N-Reinas donde la diferencia es mucho menor, promedio de 4× y máxima de 4.7×.

En forma similar al análisis anterior, para poder evaluar la eficiencia energética se empleó la métrica FLOPS/Watt para MM y Jacobi, mientras que MQUEENS/Watt fue utilizada para N-Reinas. Para estimar los Watt consumidos por el cluster x86, se empleó el valor observado de consumo de su procesador en diferentes estudios realizados [10,17], el cual a su vez coincide con la Potencia de Diseño Térmico (84W). Por otro lado, en el caso del cluster de RPi, se optó por emplear el valor observado como máximo en diferentes reportes disponibles (3.7W) [22,16,21].

La Tabla 2 muestra la eficiencia energética de las aplicaciones benchmark en ambos clusters. A diferencia del análisis de rendimiento, es el cluster de RPi quien obtiene los mejores cocientes de eficiencia energética para todos las combinaciones (N,P). Esto se debe en gran medida a su bajo consumo de potencia. En particular, obtiene una diferencia promedio de 2.5× y una máxima de 5× para MM, 3.1× con una máxima de 7.2× para Jacobi y 6.5× con máxima de 21.4× para N-Reinas. Es claro que la mayor diferencia se da en N-Reinas, ya que justamente es la aplicación en la que el cluster de RPi obtiene el mejor rendimiento y la menor diferencia con el cluster x86.

Resulta importante mencionar 2 cuestiones que podrían incidir en los resultados encontrados. La primera es desfavorable para el cluster RPi debido a que existen procesadores x86 con mejores tasas de eficiencia energética (por ejemplo, familia Xeon de Intel). Por el contrario, la segunda cuestión es beneficiosa para el cluster de RPi, ya que en el análisis realizado se consideró el consumo completo de la placa, mientras que en el de x86, sólo el del procesador.

5. Trabajos Relacionados

Existen varios trabajos que realizan análisis de rendimiento y/o eficiencia energética de cluster de RPi's, además de comparaciones con otras arquitecturas.

Kiepert [19] describe el armado de un clúster de 32 RPi's. Para evaluar el rendimiento del mismo, ejecutó una aplicación MPI que estima el valor del número Pi. Al ser una aplicación *embarazosamente paralela*, el cluster obtiene aceleraciones muy cercanas al número de nodos empleados.

Die [15] desplegó un cluster de 8 RPi's, ejecutando OpenFOAM [2] y High-Performance Linpack (HPL) [1] para el análisis de rendimiento. El cluster armado mostró buenos rendimiento para OpenFOAM en los casos en que el volumen de comunicaciones era bastante menor que el de cómputo. Respecto a HPL, Die encontró niveles de prestaciones aceptables mientras los requerimientos de memoria no superaron la memoria RAM disponible. En este estudio se observaron resultados similares, aunque al contar con más memoria RAM en cada placa, fue posible procesar problemas de mayor tamaño.

Cox et al [14] armaron un cluster de 64 RPi modelo B (usando uno de ellas como *frontend*). Al igual que el caso anterior, emplearon HPL para evaluar el rendimiento alcanzando 1.14 GFLOPS con el sistema completo. En forma similar a este trabajo, observaron que la capacidad de memoria de las placas, impiden computar problemas más grandes.

P	N		
	1024	2048	4096
4	0,12	0,14	0,14
8	0,10	0,12	0,14
16	0,07	0,09	0,12
32	0,04	0,06	0,10
64	0,02	0,04	0,05
GFLOPS/Watt			

(a) MM

P	N			
	1024	2048	4096	8192
4	0,20	0,18	0,18	0,17
8	0,12	0,15	0,17	0,17
16	0,09	0,12	0,15	0,16
32	0,06	0,08	0,13	0,15
64	0,03	0,04	0,06	0,08
GFLOPS/Watt				

(b) Jacobi

P	N			
	16	17	18	19
4	18,90	18,94	18,98	19,10
8	20,90	21,96	22,01	22,29
16	12,71	19,23	23,34	23,87
32	16,94	22,97	23,23	24,63
64	8,83	14,39	21,23	21,40
MQUEENS/Watt				

(c) N-Reinas

P	N		
	1024	2048	4096
4	0,09	0,10	0,10
8	0,04	0,07	0,08
16	0,02	0,04	0,05
32	0,01	0,02	0,04
64	0,00	0,01	0,02
GFLOPS/Watt			

(d) MM

P	N			
	1024	2048	4096	8192
4	0,08	0,04	0,05	0,05
8	0,04	0,04	0,05	0,05
16	0,02	0,06	0,07	0,07
32	0,01	0,04	0,07	0,07
64	0,00	0,02	0,06	0,06
GFLOPS/Watt				

(e) Jacobi

P	N			
	16	17	18	19
4	3,90	3,90	3,92	3,94
8	4,04	4,47	4,56	4,59
16	2,64	3,82	4,01	4,04
32	1,54	3,36	4,01	4,14
64	0,41	2,05	3,73	4,18
MQUEENS/Watt				

(f) N-Reinas

Tabla 2: Eficiencia energética de las aplicaciones benchmark en el cluster RPi (a-c) y en el x86 (d-f).

Un cluster de 25 RPi modelo B fue construido por Pfalzgraf y Driscoll [23]. Usando 3 aplicaciones de álgebra lineal (triada de vector, multiplicación de matriz-vector y multiplicación de matrices) analizaron el rendimiento del cluster armado. En el caso de la multiplicación de matrices, logran un pico de 1.1 GFLOPS con 24 placas. En esta investigación, se alcanza un pico de 3.12 GFLOPS con 16 placas, aunque obviamente se debe tener en cuenta la mayor potencia de las RPi 3 comparado a la RPi modelo B.

Cloutier et al [13] armaron un cluster de 25 RPi 2 modelo B. Una vez más, HPL fue empleado para el análisis de rendimiento. En este caso además, evaluaron la eficiencia energética del cluster midiendo el consumo a través de un medidor WattsUp Pro. En la comparación con un sistema basado en un Intel Haswell EP de 16 núcleos, observan que el cluster de RPi se ve superado por el sistema x86 en forma opuesta a lo observado al momento. Los autores indican que entre las principales causas se encuentra el profundo nivel de optimización de HPL como benchmark para medir rendimiento y la alta tasa de operaciones en punto flotante, lo que favorece a los procesadores x86 tradicionales.

Cicirello [12] construyó un cluster de RPi 3 modelo B, seleccionando como casos de estudio la estimación del número Pi y la multiplicación de matrices. Lamentablemente, la ausencia de tiempos de ejecución o FLOPS impide la comparación directa con los resultados de este estudio. Más allá de eso, y en sentido similar a este trabajo, Cicirello detecta que el rendimiento del cluster está significativamente limitado por la velocidad de la interfaz de red en aquellas aplicaciones que tienen comunicaciones de gran volumen. Además, encuentra aceptables tasas de aceleraciones en aplicaciones donde la cantidad y el volumen de comunicaciones tienen mucho menor peso que la proporción de cómputo.

Por último, los mismos autores de [13] desplegaron un cluster de RPi 3 B en [20]. Al igual que en su trabajo previo, el uso de HPL como caso de estudio los lleva a encontrar que los sistemas x86 resultan superiores no sólo en rendimiento pico sino también en rendimiento/watt.

6. Conclusiones y Líneas de Trabajo Futuro

En la actualidad, la reducción del consumo energético se ha vuelto uno de los principales desafíos para la comunidad HPC. Entre las diferentes alternativas actualmente exploradas, se encuentra el uso de procesadores ARM. En este trabajo se estudia la viabilidad del uso de un cluster de RPi. Del análisis realizado sobre los resultados obtenidos se puede concluir que:

- Aquellas aplicaciones que demandan comunicaciones de gran volumen, probablemente sufran una degradación importante en su rendimiento debido a la limitada velocidad de la interfaz de red que tiene la RPi, como se observó en MM.
- Aquellas aplicaciones que requieran una gran cantidad de comunicaciones de tamaño moderado, también penalizarán su rendimiento por la misma causa que el caso anterior, como se apreció en Jacobi.
- Aplicaciones que trabajen con volúmenes de datos muy superiores a la capacidad disponible de la memoria RAM de las RPi, pueden sufrir errores en ejecución, tal como se vio en MM.
- Aplicaciones que demanden mucho cómputo pero con pequeños datos, que no tengan una incidencia significativa de comunicaciones, probablemente tengan muy buen rendimiento. Esto se ve reflejado en el caso de N-Reinas, donde sus resultados muestran niveles altos de eficiencia.

Como era de esperar, desde el punto de vista del rendimiento, el cluster RPi presenta tasas de prestaciones más bajas en comparación a un cluster x86 (entre 4 y 10.4× de diferencia promedio). Sin embargo, desde la perspectiva de la eficiencia energética, el cluster de RPi alcanza mejores resultados (entre 2.5 y 6.5× de diferencia promedio). Considerando lo analizado, es posible decir que los cluster de RPi ofrecen capacidades para cómputo paralelo y pueden obtener tasas de eficiencia energética superiores a los clusters x86. Sin embargo, debido a la gran diferencia de rendimiento pico que tiene con estos últimos, no representan hoy una opción viable para HPC si el objetivo principal es la reducción del tiempo de ejecución.

Algunas de las líneas de trabajo futuro que se desprenden de esta investigación pueden ser:

- Considerando que existen placas más avanzadas que la RPi 3 (como pueden ser la Banana Pi M3 [6] o la ROCK64 [8]), interesa replicar el estudio realizado para continuar la búsqueda de alternativas viables basadas en procesadores ARM.
- Como el consumo real puede diferir del estimado, sería interesante medir realmente el consumo en ambos clusters para hacer un estudio más preciso del balance rendimiento/energía.

Referencias

1. Hpl. http://www.netlib.org/benchmark/hpl
2. Openfoam. https://www.openfoam.com
3. PandaBoard. https://en.wikipedia.org/wiki/PandaBoard (2017)
4. Sitio Web Beagleboard. https://beagleboard.org/ (2017)
5. TOP500. https://www.top500.org (2017)
6. Banana Pi M3. http://www.banana-pi.org/m3.html (2018)
7. Raspberry. https://www.raspberrypi.org (2018)
8. ROCK64. https://www.pine64.org/?page_id=7147 (2018)
9. Balakrishnan, N.: Building and benchmarking a low power ARM cluster. Ph.D. thesis, University of Edinburgh (2012)
10. Benchmark-tests: Intel core i5-4460. https://tinyurl.com/ybduwtu2
11. Chaudhari, H.: Raspberry pi technology: A review. International Journal of Innovative and Emerging Research in Engineering (IJIERE) pp. 83–87 (2015)
12. Cicirello, V.A.: Design, configuration, implementation, and performance of a simple 32 core raspberry pi cluster (08 2017)
13. Cloutier, M.F., Paradis, C., Weaver, V.M.: Design and analysis of a 32-bit embedded high-performance cluster optimized for energy and performance. In: 2014 Hardware-Software Co-Design for HPC. pp. 1–8 (Nov 2014)
14. Cox, S., Cox, J., Boardman, R., et al: Iridis-pi: a low-cost, compact demonstration cluster. Cluster Computing 17(2), 349358 (Jun 2014)
15. Die, B.: Distributed computing with the Raspberry Pi. Ph.D. thesis, Kansas State University (2014)
16. Dramble, R.P.: Power consumption benchmarks. https://tinyurl.com/jy7fvzc
17. Hardware.Info: Amd vs intel: 57 processor megatest. https://goo.gl/hxZkTX
18. Jarus, M., Varrette, S., Bouvry, A.O..P.: Performance evaluation and energy efficiency of high-density hpc platforms based on intel, amd and arm processors. Lecture Notes in Computer Science 8046, 182–200 (2013)
19. Kiepert, J.: Creating a raspberry pi-based beowulf cluster. https://tinyurl.com/y778qkvx
20. M. F. Cloutier, C.P., Weaver, V.M.: A raspberry pi cluster instrumented for fine-grained power measurement. Electronics 5(4) (2016)
21. Magazine, T.M.: Raspberry pi 3 is out now! specs, benchmarks & more. https://www.raspberrypi.org/magpi/raspberry-pi-3-specs-benchmarks
22. McDonald, S.: Raspberry pi 3 a first look. https://tinyurl.com/y9z4roje
23. Pfalzgraf, A.M., Driscoll, J.A.: A low-cost computer cluster for high-performance computing education. In: IEEE International Conference on Electro/Information Technology. pp. 362–366 (June 2014)
24. Wolfe, M.: Arm yourselves for exascale. https://tinyurl.com/y7gus9ru (2011)