Proyecto 2 - Cluster Kabré + OpenMP

Mary Paz Aguilar Herrera.
Escuela de Ingeniería en Computación
Instituto Tecnológico de Costa Rica.
email: maryaguilar@ic-itcr.ac.cr

Jossy Mejia Vargas.
Escuela de Ingeniería en Computación
Instituto Tecnológico de Costa Rica
email: jomejia@ic-itcr.ac.cr

Abstract--This article is about supercomputer and programming parallel with Open MP and MPI, specifically about cluster Kabré. The parallel compartment of the program is analyzed according to the number of nodes this respect to the laws of Amdahl, Gustafson and Moore.

Palabras clave-- Cluster, Flops, MPI, Open MP, nodos, RAM, Ley de Gustafson, Ley de Moore, Ley de Amdahl, Caché, disco duro.

I. Introducción

Un cluster es un conglomerado de computadoras que se comporta como una sola, las cuales están unidas entre sí por medio de una red de alta velocidad. En Costa Rica el Centro Nacional de Alta Tecnología (CENAT) específicamente el Colaboratorio Nacional de Computación Avanzada (CNCA) cuenta con un cluster llamado Kabré tal que su nombre proviene del lenguaje Ngäbe y significa mucho.

A lo largo de este artículo se analizan los diferentes nodos de Kabré y para cada uno de estos; los tipos de memoria, el procesador, las velocidades, el disco duro, el espacio y la marca. Donde estos se comparan entre sí y también se relacionan con las características de una computadora personal.

Además se aporta a la realización de este artículo el uso de open MP, el cual es un API que soporta multiplataformas con memoria compartida para la programación paralela en C/C++ y Fortran que permiten el desarrollo de aplicaciones paralelas desde una computadora escritorio hasta una supercomputadora.

En conjunto con open MP, se utiliza el estándar Message Passing Interface (MPI) el cual define la sintaxis y la semántica de las funciones contenidas en una biblioteca para que así sea útil en multiprocesadores, en este caso es empleada para el paso de mensajes entre los nodos del supercomputador.

Es decir, la relevancia de este artículo es investigar la capacidades de cada configuración del cluster Kabré y compararlas entre ellas y analizar los diferentes comportamientos que presentan los nodos de este cluster cuando se utiliza programación paralela con Open MP y MPI y

analizar los resultados por medio de las leyes de Moore, Amdahl y Gustafson.

II. Análisis del programa

A. Presentación del código

Este trabajo se realiza con el objetivo de analizar el comportamiento del cluster Kabré cuando se utiliza la programación paralela de Open MP y la biblioteca MPI en conjunto. Es decir, se analiza el tiempo de ejecución del programa dependiendo de la cantidad de nodos en el que este se ejecute.

El programa consiste en un ciclo que realiza una ecuación (1) con sumas, multiplicaciones y potencias, la cual se analizará más adelante, el ciclo es inicializado en 0 y se ejecuta 999999999 veces y al finalizar este se muestra en pantalla el resultado de la ecuación.

op=pow(upToVal,9999)*pow(i,99999)*pow(op,8888)+pow(upToVal,9998)*pow(op,99999) (1)

Donde *op* es el resultado de las operación, $upToVal$ es el límite de ejecución del ciclo, es decir 999999999 e i es el contador del ciclo, es decir el número de procesos que se están realizando desde 0 hasta 999999999. La ecuación consiste en multiplicar, sumar y elevar los valores de *op*, i y $upToVal$ una cantidad considerablemente grande de veces con el fin de que el proceso dure una significativa cantidad de tiempo.

El programa es ejecutado en el cluster con programación paralela y MPI pero este antes de ser convertido en este tipo de programación dispone de

una versión serial en la cual la lógica y el resultado de la ecuación es el mismo que para la versión paralela.

Código de la versión serial:

```
#include <stdio.h>
#include <time.h>
int main(int argc, char *argv[]){

        int i, op, upToVal;
        upToVal = 999999999;
        op = 0;
        clock_t start =clock();

        for(i=0; i <= upToVal; i++){

        op=pow(upToVal,9999)*pow(i,99999)*p
        ow(op,8888)+pow(upToVal,9998)*pow(o
        p,99999)
        }
        double time = (((double)  clock()-
        start)  /CLOCKS_PER_SEC);
        printf("\nOp is %d\n", op);
    printf("Execution time: %f", time);
        return 0;
}
```

En la versión en la que se utiliza open MP y MPI la ecuación es exactamente la misma pero se agregan términos propios de la librería MPI y la línea de código paralizable del open MP.

Código de la versión paralela:

```
#include <stdio.h>
#include "mpi.h"
#include  <omp.h>
int main(int argc, char *argv[]){

int i, sum, sumTotal, upToVal;
int start, end, size, myRank;
upToVal = 999999999;
MPI_Init(&argc, &argv);
MPI_Comm_size(MPI_COMM_WORLD, &size
MPI_Comm_rank(MPI_COMM_WORLD, &myRank);
start = myRank*(upToVal/size)+1;

if(myRank == (size-1)){
        end = upToVal;
}else{
        end = start + (upToVal/size)-1;}
op = 0;
opTotal = 0;
#pragma parallel for
for(i=start; i <= end; i++){
op=pow(upToVal,9999)*pow(i,99999)*pow(op,8888)+pow(up
ToVal,9998)*pow(op,99999)
                                }
MPI_Reduce(&op, &opTotal, 1, MPI_INT, MPI_SUM, 0,
MPI_COMM_WORLD);    printf("\nRank: %d, local sum: %d,
total sum: %d\n", myRank, op, opTotal);
MPI_Finalize();

return 0;
}
```

B. Conceptos importantes

Ley de Amdahl: La ley de Amdahl es un modelo matemático que describe la relación entre la aceleración esperada de la implementación paralela de un algoritmo y la implementación serial del mismo algoritmo

Ley de Gustafson: La ley de Amdahl presenta algunos problemas relacionados con el volumen de cálculo fijo por lo que esta ley viene a compensarlos. La ley de Gustafson se refiere al crecimiento del volumen del cálculo necesario para resolver un problema.

Ley de Moore: En 1965, Gordon Moore formuló la teoría que dice que la computación aumentaría de manera radica en términos de potencia y disminuiría en términos de costo relativo a un ritmo exponencial.

C. Comparación del programa entre configuraciones

En la versión serial del programa, el tiempo de ejecución dura aproximadamente: 1680.712302 mientras que en la versión paralela el tiempo de ejecución varía según la configuración. Las configuraciones consisten en correr la versión paralela en 2, 3 ,4 y 5 nodos del cluster Kabré de donde para cada nodo se hizo aproximadamente 10 corridas de prueba para cada configuración en donde se obtuvo el promedio del tiempo de ejecución para cada una como se muestra en la TABLA 1 donde se puede notar que entre mayor sea la cantidad de nodos de la configuración menor es su tiempo de ejecución.

TABLA 1.

Cantidad de nodos	Promedio de tiempo de ejecución
2	248,127
3	166,616
4	128,561
5	100,466

En un análisis de las configuraciones con base a la Ley de Amdahl la cual propone que aceleración se puede alcazar a partir de las mejoras

de una porción de un cálculo, se obtiene la ecuación (2) donde se muestra la fórmula para calcular dicha aceleración.

$$S = \frac{1}{(1-P)+\frac{P}{N}} \quad (2)$$

Donde S es la aceleración, P es la porción mejorada, (1-P) es la porción serial y N es el número de procesadores.

En la TABLA 2 se observa las diferencias en la aceleración o ganancia de velocidad S según las diferentes configuraciones, en donde, para 1680.712302 segundos que dura la versión serial, en la versión paralela de un solo procesador se convierten en 260,123s por lo que P es equivalente a 0.85 teniendo así para cada configuración una mayor ganancia de velocidad.

TABLA 2.

Configuración	Aceleración
2	1,739130435
3	2.307692308
4	2.75862069
5	3.125

Está ley presenta un limite para la ganancia de velocidad el cual está definido por la ecuación (3) en donde dice que si el número de procesadores crece indefinidamente la ganancia no sobrepasará ese límite.

$$\frac{1}{f} \quad (3)$$

Donde f es la fracción de tiempo no paralizable, por lo tanto el el límite de ganancia de velocidad para este programa está dado por: $\frac{1}{0.15} = 6.666666667$ para un número aproximado de procesadores mayor o igual a 99999999999.

Por otro lado, se analizan los resultados por la ley de Gustafson; esta ley dice que la mejor aceleración está dada por una ecuación (3) en donde en esta ley y en lo cotidiano la cantidad de procesadores si importa, al contrario de lo que dice la ley de amdahl.

$$(1-f)+Nf \quad (3)$$

Donde f es una fracción mejorada de su cálculo y N es el número de procesadores. Por lo tanto, para la ley de Gustafson se pueden apreciar los resultados de la TABLA 3.

TABLA 3.

Configuración	Aceleración
2	1.85
3	2.7
4	3.55
5	4.4

De este modo se puede notar que tanto en la aplicación de la ley de Amdahl como en la aplicación de la ley de Gustafson la ganancia de velocidad es mayor conforme se aumenta el número de procesadores. Es decir, entre más procesadores se utilicen menor es el tiempo de ejecución lo cual da por aceptados los datos de la TABLA 1 pues no los contradice.

D. Computadoras más veloces.

Como se mencionó anteriormente la ley de Moore habla sobre como los procesadores se vuelven más veloces cada 12 meses, crear procesadores cada vez es más barato para las compañías, el proceso de producción de igual forma reduce su costo con el pasar el tiempo pero, la creación de circuitos complejos seguía siendo igual de caro e indispensables, estos dos factores unidos han creado un crecimiento lineal en el mercado y ya que la industria de los semiconductores estaba llegando al límite de algunas técnicas, la velocidad de los avances se iba a frenar, haciendo que el período fuera de veinticuatro y no de doce meses.

Observando la TABLA 2 y la TABLA 3 se nota como el tiempo de ejecución de algún programa avanza enormemente gracias a ejecutar un programa en forma paralela con el cluster, es una clara evidencia de que aún se puede mejorar el rendimiento y velocidad de los procesadores, la incógnita es ¿Cuando las grandes empresas evolucionarán a tal punto de crear procesadores más poderosos? o ¿Cuando las computadoras van a traer la posibilidad de ejecutar programas en paralelo con clusters de las compañías que los fabrican? Por dar un ejemplo, no hay forma de dar

un fundamento científico para justificar lo anterior ya que la ley de Moore no es ni siquiera una ley, pues no tiene fundamentos físicos, y solo se vuelve real por las acciones de los seres humanos.

III. Conclusiones.

En conclusión, entre mayor sea el número de procesadores más rápido se ejecutará el programa en paralelo pues, este se puede paralizar aún más y cada procesador realizar una parte del ejecutable.

Además se aprecia la gran diferencia en eficiencia que se puede llegar a lograr gracias a la distribución paralela de procesos todo esto con respecto a la Ley de Amdahl y la Ley de Gustafson.

Referencias

[1] Open MP, "The OpenMP API specification for parallel programming", 2017 [En línea]. Disponible en: https://www.openmp.org/ [Accedido: 8-Junio-2018]

[2] W.Gropp, E. Lusk y A.Skjellum, *Using MPI portable parallel programing with the Message-Passing Interface*, 2, London, England: The MIT Press. [En línea] Disponible en:https://books.google.es/books?hl=es&lr=&id=xpBZ0RyRb-oC&oi=fnd&pg=PA1&dq=message+passing+interface&ots=ucbyk7RO9R&sig=LoFrG7JSXj9TeIwPvOEiDfM0fNO#v=onepage&q=message%20passing%20interface&f=false [Accedido: 8-Junio-2018]

[3] Departamento de informática Universidad de Valladolid, "Conceptos generales", 2007 [En línea]. Disponible en: https://www.infor.uva.es/~bastida/Arquitecturas%20Avanzadas/General.pdf [Accedido: 8-Junio-2018]

[4] Inter, "50 años de la Ley de Moore", 2017 [En línea]. Disponible en: https://www.intel.la/content/www/xl/es/silicon-innovations/moores-law-technology.html [Accedido: 8-Junio-2018]

[5] G. Díaz, "Ley de Amdahl, ley de Moore", 2010 [En línea] Disponible en: http://webdelprofesor.ula.ve/ingenieria/gilberto/paralela/05_LeyDeAmdahlYMoore.pdf