Open-MP: Generación de Matriz de Covarianzas

Pedro Vladimir Hernández Serrano, José Alfredo Méndez Barrera y Elisa Hernández Rodríguez

I. INTRODUCIÓN

El siguiente documento aborda la descripción de la codificación, compilación y ejecución de una matriz de covarianzas en paralelo y de manera secuencial. La ejecución secuencial se realiza en C. Por otra parte, la ejecución en paralelo es posible a través de la herramienta Open-MP.

El uso de esta matriz, particularmente en *big data*, requiere poder de cómputo y eficiencia en el código. En la siguiente sección se describe la forma en que una matriz de covarianzas puede ejecutarse.

II. INVESTIGACIÓN DEL PROBLEMA

La matriz de Covarianzas es muy utilizada en distintos problemas, ya que ayuda a conocer la relación que existe entre las variables que componen la matriz origen.

La covarianza es una medida estadística de la relación lineal entre dos variables. Si la covarianza entre dos variables es alta, un cambio en una variable significa un cambio lineal similar en la variable relacionada. Cuando la covarianza es muy negativa, las dos variables están inversamente relacionadas entre sí.

En particular, esta matriz juega un papel importante en la minera de datos y para clusterización por medio de **componentes principales basados en las covarianzas**.

- El análisis de componentes principales lo utiliza para ordenar las variables de manera decreciente con respecto a su variabilidad y así el número de variables puede ser reducido a aquellas de mayor alcance.

- La clusterizacin requiere la recuperación de información de grandes bases de datos por lo que es conveniente reducir la dimensionalirdad del conjunto de datos, y la idea principal es la de implementar una técnica para formar una matriz de covarianza de los datos y encontrar los vectores propios. los componentes principales se utilizan para formar un nuevo espacio, más pequeño dimensionalmente con puntos de datos proyectados de la datos originales.

- La matriz de covarianza también se utiliza en la minería de datos. La clasificación es una forma de crear normas que rigen la categoría de un objeto de datos. Estas reglas se construyen mediante el examen de un pequeño conjunto de datos ya categorizados. Los algoritmos utilizan estas reglas para separar los nuevos valores de los datos en clases existente.

De manera formal podemos definir la covarianza matemática dadas las variables X, Y como se muestra en la ecuación 1 de igual manera podemos ver en la ecuación 4 una forma alternativa.

$$\operatorname{cov}(X,Y) = \mathrm{E}\left[(X-\mathrm{E}[X])(Y-\mathrm{E}[Y])\right], \tag{1}$$

En donde E[X] es el valor esperado de X.

$$\begin{aligned}\operatorname{cov}(X,Y) &= \mathrm{E}\left[(X-\mathrm{E}[X])\,(Y-\mathrm{E}[Y])\right] \\ &= \mathrm{E}\left[XY - X\,\mathrm{E}[Y] - \mathrm{E}[X]\,Y + \mathrm{E}[X]\,\mathrm{E}[Y]\right] \\ &= \mathrm{E}[XY] - \mathrm{E}[X]\,\mathrm{E}[Y] - \mathrm{E}[X]\,\mathrm{E}[Y] + \mathrm{E}[X]\,\mathrm{E}[Y] \\ &= \mathrm{E}[XY] - \mathrm{E}[X]\,\mathrm{E}[Y]\,.\end{aligned}$$

Que también puede ser expresado como sigue:

$$\operatorname{cov}(X,Y) = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{1}{2}(x_i - x_j)\cdot(y_i - y_j) \tag{2}$$

$$= \frac{1}{n^2}\sum_{i}\sum_{j>i}(x_i - x_j)\cdot(y_i - y_j) \tag{3}$$

$$= \frac{1}{n}\sum_{i}(x_i - \bar{x})\cdot(y_i - bary) \tag{4}$$

Cada elemento de la matriz es una suma de productos de los elementos en la matriz de entrada tomados de una ventana de datos dinmica.

II-A. PASOS PARA GENERAR LA MATRIZ

El programa desarrollado permite que el usuario incluya una matriz dentro de un documento en formato csv. Las caracterasticas que debe cumplir el archivo son:

1. Nombrarlo file.csv
2. Separar los elementos con comas
3. Cada fila de la matriz debe escribirse en una nueva fila en el archivo

A continuación en I se presenta un ejemplo del file.csv que debe contruirse.

TABLE I

FILE.CSV: EJEMPLO

```
20,10,1,3,4,7,8,4,23,56,844,24
3,12,13,56,1,6,7,98,46,34,21,36
7,4,5,9,30,3,700,6,45,2,93,16
```

II-A.1. SECUENCIAL: El proceso para calcular la matriz de covarianzas de forma secuencial está en un *script* en C. Este proceso imprime a pantalla la matriz origen, la matriz de los productos de las desviaciones y la matriz de covarianzas.

Algorithm 1: Matriz de Varianzas y Covarianzas: Método Secuencial

Result: Matriz de covarianzas de las variables de una matriz (M) de p filas y q columnas implementando un algoritmo secuencial

```
initialization;
M una matriz de datos de p filas y q columnas
Calcular la suma de los valores de cada una de las q variables de M.
while Mientras no se terminen las columnas do
    while Mientras no se terminen las filas do
        suma_j += M(i,j);
    end
end
Procedemos a calcular los valores esperados por variable. while Mientras no se terminen las columnas do
    E(j)=suma_j/p;
end
Procedemos a calcular la matriz de los productos de las desviaciones.
while Mientras no se terminen las columnas do
    while Mientras no se terminen las columnas do
        while Mientras no se terminen las filas do
            restasX_i = M(i,j) - E(j); restasY_i = M(i,z) - E(z); M_multiXY(j,z) += restasX_i*restasY_i
        end
    end
end
Procedemos a calcular la matriz de las covarinzas.
while Mientras no se terminen las columnas do
    while Mientras no se terminen las columnas do
        M_covarianzas(j,z) = M_multiXY(i,j)/q
    end
end
```

II-A.2. PARALELO:

OPEN-MP: El cómputo en paralelo tiene dos arquitecturas principales: *shared memory* y *clústers*. En este proyecto, la matriz es realizada con una architectura de *shared memory*: Open-MP.

Es la API estńdar para la escritura de la memoria compartida de las aplicaciones en paralelo en C, C ++ y Fortran.

En qué consiste?

- Directivas del compilador
- Subrutinas/funciones de tiempo de ejecución
- Variables de entorno

El algoritmo construido tiene algunas caractersticas que el usuario debe distinguir:

1. pragma: el *pragma omp parallel* se utiliza para bifurcar *threads* adicionales para llevar a cabo el trabajo. El *thread* original se denota como *thread* principal con ID 0.

   a) private: los datos de la regin paralela son privados para cada thread, lo que significa que cada thread tendrá una copia local que la usa como variable temporal. Una variable privada no es inicializada y tampoco se mantiene fuera de la regin paralela. Por definición, los contadores de iteraciones en Open-MP es privado.

   b) shared: los datos de la región paralela son compartidos, lo que significa que son visibles y accesibles por todos los threads. Por definición, todas las variables que trabajan en la región paralela son compartidas excepto los contadores de iteraciones.

2. schedule: Las iteraciones se asignan a los thread de acuerdo con el método de programación definido en el *schedule*, en particular en este proyecto se asigno un *schedule static*. En este último todos los thread son asignados antes de que las iteraciones sean ejecutadas.

3. chunk: Siguiendo la elección de *schedule static*, el loop es dividido en cada thread en partes iguales.

En este proyecto es nesario que el usuario considere que el número de threads debe ser divisor del número de columnas.

III. DISCUSIÓN

El cálculo de la matriz de covarianzas a través de Open-MP se realizó con éxito. No obstante, el uso del algoritmo construido requiere que el usuario elimine las instrucciones de impresión para hacer más eficiente el código. Por otra parte, se sugiere incorporar la comparacin entre paralelismo con memoria compartida y métodos como MPI que posibilitan ejecutar paralelismo a través de la formación de clústers.

REFERENCIAS

[1] Chilson J. Parallel Computation of High Dimensional Robust Correlation and Covariance Matrices, Univerity of British Columbia, Thesis, March, 2004.

[2] Open-MP Loop Scheduling, Ronald W Green (Intel). September 4, 2012

Algorithm 2: Matriz de Varianzas y Covarianzas: Mtodo Paralelo

Result: Matriz de covarianzas de las variables de una matriz (M) de p filas y q columnas utilizando paralelismo de cores por medio de la implementacin de multithreading processing a travs de la extensión OpenMP

initialization;
M una matriz de datos de p filas y q columnas
nthreads : Número de threads elegidos
chunk: Número de elementos consecutivos que se le asignan a cada thread dentro de un for
Obtenemos el número de threads que eligió el usuario.
Calculamos el chunk en función al nthreads y a la cantidad de campos (q) que tiene la matriz fuente (M).
Inicializamos la direcitva paralle de pragma omp estableciendo las variables que serán compartidas para todos los threads y las que mantendrán valores en privado.
Iniciamos un for schedule con la cantidad de calculada de chuk y procedemos a calcular la suma de los valores de cada una de las q variables de M.
while *Mientras no se terminen las columnas* **do**
    while *Mientras no se terminen las filas* **do**
        suma_j +=M(i,j);
    end
end
Iniciamos un for schedule con la cantidad de calculada de chuk y procedemos a calcular los valores esperados por variable. **while** *Mientras no se terminen las columnas* **do**
    E(j)=suma_j/p;
end
Iniciamos un for schedule con la cantidad de calculada de chuk y procedemos a calcular la matriz de los productos de las desviaciones.
while *Mientras no se terminen las columnas* **do**
    while *Mientras no se terminen las columnas* **do**
        while *Mientras no se terminen las filas* **do**
            restasX_i = M(i,j) - E(j); restasY_i = M(i,z) - E(z); M_multiXY(j,z) += restasX_i*restasY_i
        end
    end
end
Iniciamos un for schedule con la cantidad de calculada de chuk y procedemos a calcular la matriz de las covarinzas.
while *Mientras no se terminen las columnas* **do**
    while *Mientras no se terminen las columnas* **do**
        M_covarianzas(j,z) = M_multiXY(i,j)/q
    end
end