

# Evaluación de las rutinas de multiplicación sobre matrices dispersas

E. Maciel<sup>1</sup>   P. Torres<sup>1,2</sup>   C. Schaerer<sup>1,2</sup>

<sup>1</sup>Facultad Politécnica  
Universidad Nacional de Asunción

<sup>2</sup>Centro de Investigación en Matemática - CIMA

X Workshop de Proyectos Finales de Grado, 2016

# Esquema

## Matriz \* vector

MV

SpMV y SpMM

## CPU y memoria

Procesador y jerarquía de memoria

Costo de procesamiento y transporte de datos

## Métricas

Intensidad Aritmética

Roofline model

IA de MV y otros kernels

## Resultados

SpMV / SpMM

## Multiplicación matriz por vector

Enorme cantidad de aplicaciones científicas y de ingeniería (KSM, procesamiento de señales, motores de búsqueda, entre otros).

Ejemplo de MV:

$$\begin{bmatrix} a & b \\ c & d \\ e & f \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} ax_1 + bx_2 \\ cx_1 + dx_2 \\ ex_1 + fx_2 \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} \quad (1)$$

En general:

$$A^{m \times n}, x^{n \times 1}, y^{m \times 1}$$
$$y_i = \sum_{j=1}^n A_{ij} x_j \text{ for } i \in \{1, \dots, m\} \quad (2)$$

# SpMV y SpMM

## SpMV

Consiste en realizar una multiplicación de la forma

$$y = A * x \quad (3)$$

donde  $A$  es una matriz dispersa de  $m$  por  $n$ ,  $x$  es un vector denso e  $y$  es el vector que resulta de la operación.

## SpMM

Es una generalización de SpMV en donde la matriz dispersa es multiplicada por un conjunto de vectores densos, los cuales forman una matriz densa.

$$C = A * B \quad (4)$$

$A$  es una matriz dispersa de  $m$  por  $n$ ,  $B$  es una matriz densa de  $n$  por  $p$  ( $p \ll n$ ) y  $C$  es la matriz resultante.

# Arquitectura de von Neumann

Las computadoras, a un nivel alto de abstracción, pueden describirse mediante la arquitectura de von Neumann.

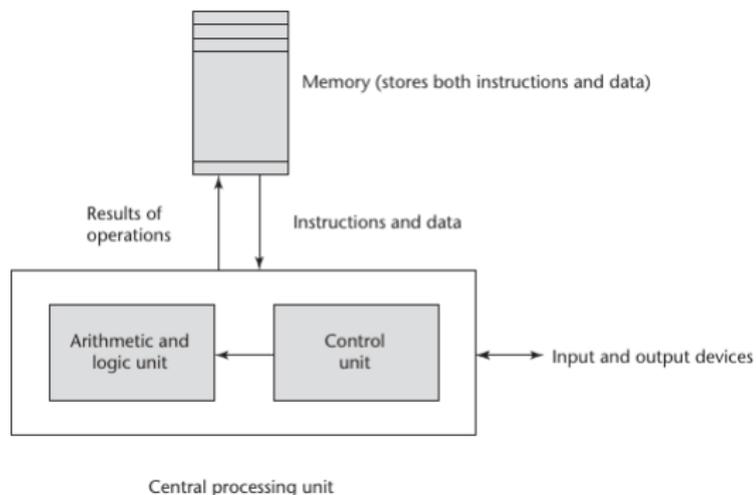


Figure: Arquitectura de von Neumann. Sebesta, 2012

# Jerarquía de memoria

Las computadoras poseen varios niveles de memoria que se encuentran entre la CPU y la memoria principal.

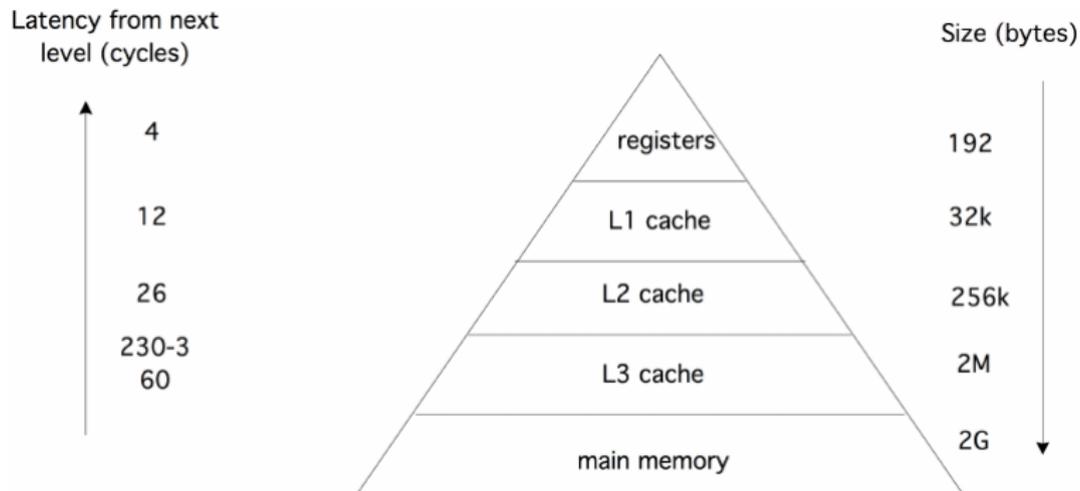


Figure: Jerarquía de memoria de un Intel Sandy Bridge. Eijkhout, 2014

# Procesamiento y transporte de datos

Los datos a ser procesados se cargan en la memoria principal y pasan por cada uno de los niveles de la caché hasta llegar a los registros, donde son procesados.

## Tiempo de ejecución de un programa

$$\begin{aligned} & \#FLOPS * tiempo\_por\_flop + \\ & \#datos / ancho\ de\ banda + \\ & \#transportes * latencia \end{aligned}$$

en donde

$$tiempo\_por\_flop \ll 1 / ancho\ de\ banda \ll latencia$$

# Intensidad Aritmética

## Definición

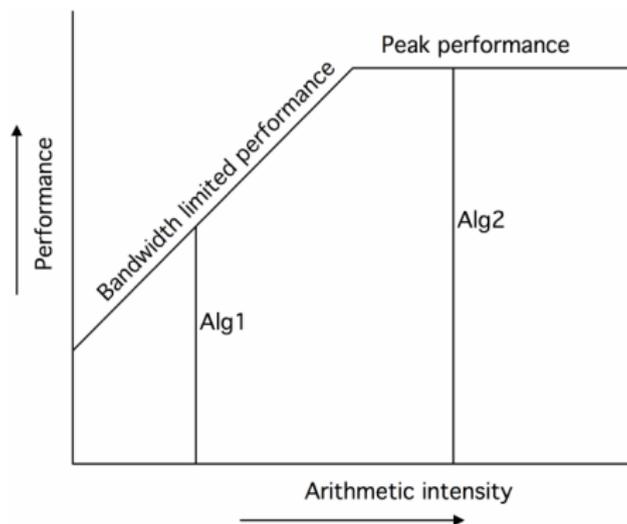
Si  $n$  es la cantidad de datos sobre los cuales un algoritmo opera, y  $f(n)$  la cantidad de operaciones que se realizan, la intensidad aritmética es  $\frac{f(n)}{n}$

Típicamente se expresa en

$$\frac{FLOPS}{Bytes}$$

## Roofline model

Para una intensidad aritmética dada el rendimiento se determina por el punto donde la línea vertical interseca el *roof line*.



**Figure:** Gráfico del *roofline model*. El rendimiento de *Alg1* está limitado por el ancho de banda de la memoria, mientras que *Alg2* está limitado por la capacidad de cálculos del procesador. Eijkhout, 2014

# Intensidad Aritmética en algoritmos

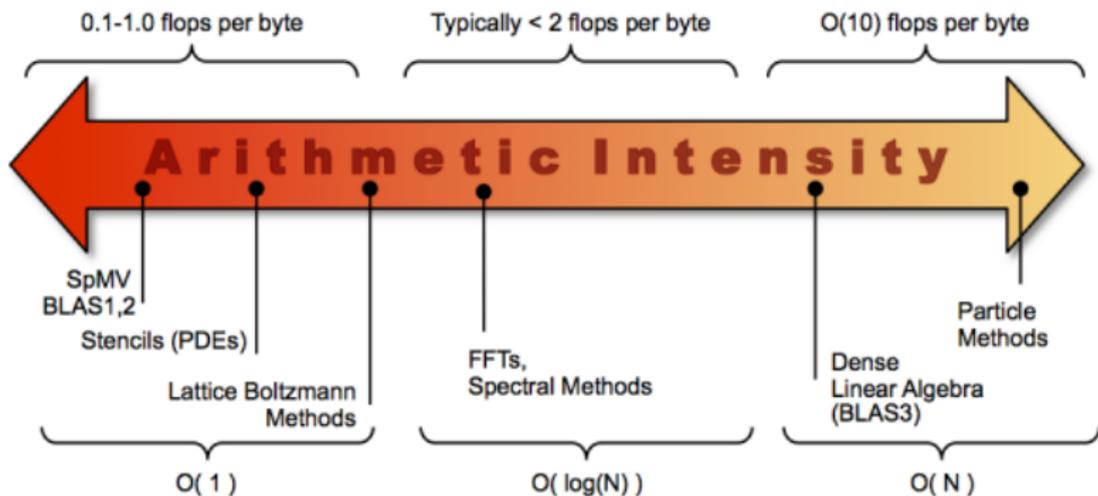


Figure: IA lograda en distintos kernels. Fuente: <https://crd.lbl.gov/departments/computer-science/PAR/research/roofline/>

# Configuraciones para las pruebas

## Hardware

- ▶ AMD Phenom II x4 955
- ▶ Intel Xeon E5530

## Librerías

- ▶ **PAPI** (*Performance Application Programming Interface*):  
Para acceder a los contadores de hardware, como tiempo de uso del procesador, FLOP/s, etc.
- ▶ **Eigen v3**:  
Provee abstracciones para la manipulación y operación sobre matrices y vectores en C++.

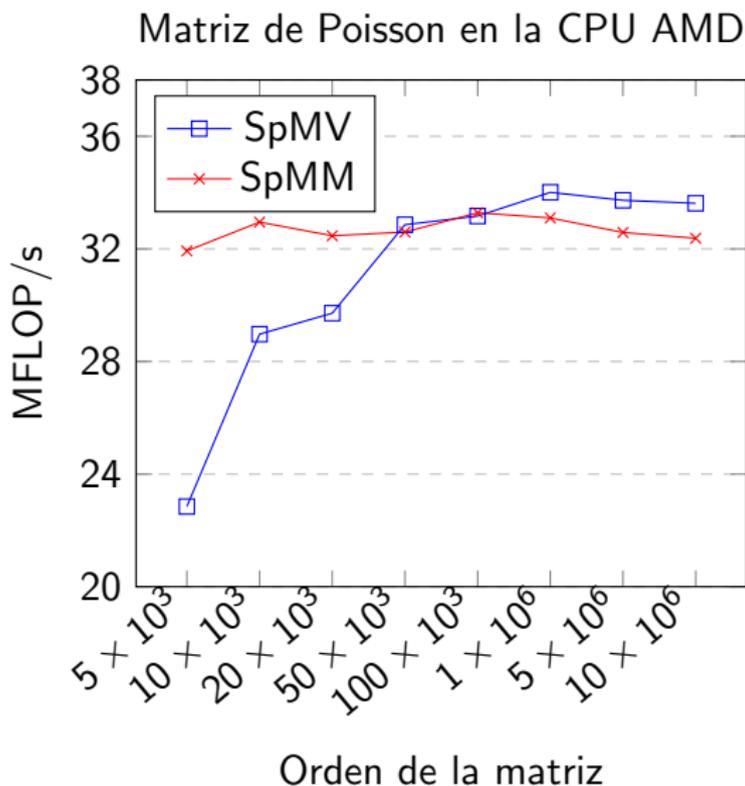
Table: Pruebas en AMD Phenom II x4

| Matriz   | Dimensión   | MFLOP/s |               |
|----------|-------------|---------|---------------|
|          |             | SpMV    | SpMM $p = 10$ |
| finan512 | 74752x74752 | 32.416  | 37.2906       |
| mcfе     | 765x765     | 34.7158 | 48.2067       |
| pwt      | 36519x36519 | 27.9292 | 33.8493       |
| raefsky3 | 21200x21200 | 50.1089 | 51.5167       |
| s3dkq4m2 | 90449x90449 | 46.7826 | 49.4118       |
| sherman5 | 3312x3312   | 28.4741 | 39.2228       |

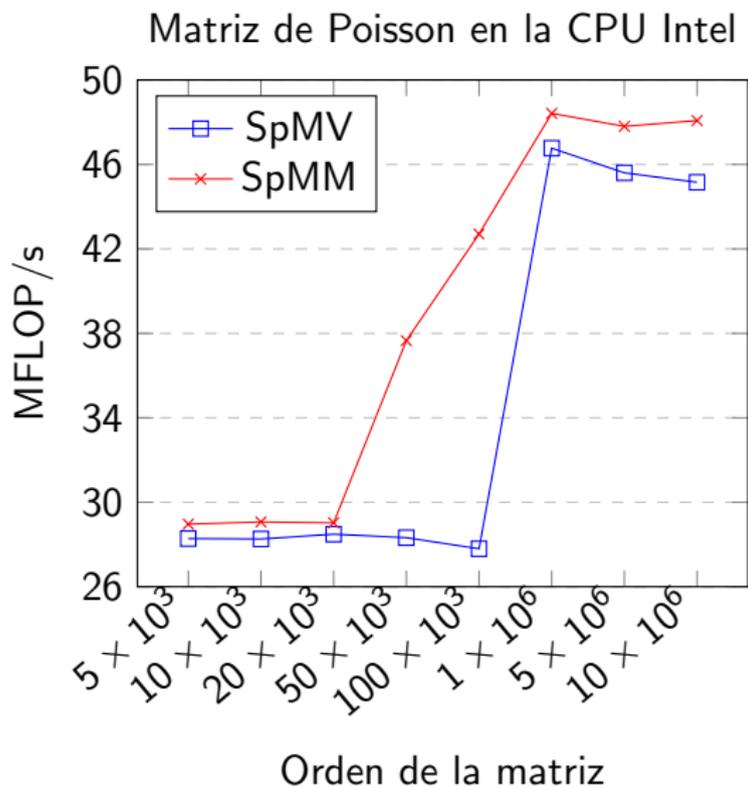
Table: Pruebas en Intel Xeon E5530

| Matriz   | Dimensión   | MFLOP/s |               |
|----------|-------------|---------|---------------|
|          |             | SpMV    | SpMM $p = 10$ |
| finan512 | 74752x74752 | 42.2557 | 53.2811       |
| mcf      | 765x765     | 45.6382 | 66.2403       |
| pwt      | 36519x36519 | 36.2499 | 45.1379       |
| raefsky3 | 21200x21200 | 81.3873 | 83.6199       |
| s3dkq4m2 | 90449x90449 | 73.1416 | 77.6655       |
| sherman5 | 3312x3312   | 32.9084 | 55.4644       |

# SpMV / SpMM (Poisson) (I)



# SpMV / SpMM (Poisson) (II)



# Sumario

- ▶ SpMV y SpMM son kernels con baja IA.
- ▶ Para la optimización se deben realizar estrategias con relación al transporte de datos.
- ▶ SpMM tiene un mayor rendimiento sobre SpMV hasta un cierto orden para la matriz.

## Estado del trabajo

- ▶ Desarrollo de la matemática asociada a los resultados.

## Posibles trabajos futuros

- ▶ Mismo concepto en arquitecturas con múltiples procesadores.

## Referencias y lectura adicional



V. Eijkhout, E. Chow, R. van de Geijn.

*Introduction to High Performance Scientific Computing.*  
lulu.com, 2014.



R. Sebesta.

*Concepts of programming languages.*  
Addison-Wesley, 2012.



G. Ballard, E. Carson, J. Demmel et al.

Communication lower bounds and optimal algorithms for  
numerical linear algebra.

*Acta Numerica*, 23:1–155, 2014.