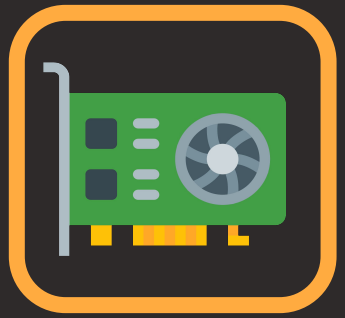
GPU Computing: una mirada al pasado, presente y futuro

Charla Invitado Universidad del Bio Bio

Dr. Cristóbal A. Navarro
Instituto de Informatica
Facultad de Ciencias de la Ingeniería
Universidad Austral de Chile

Facultad de Ciencias de la Ingeniería UACh

temporal



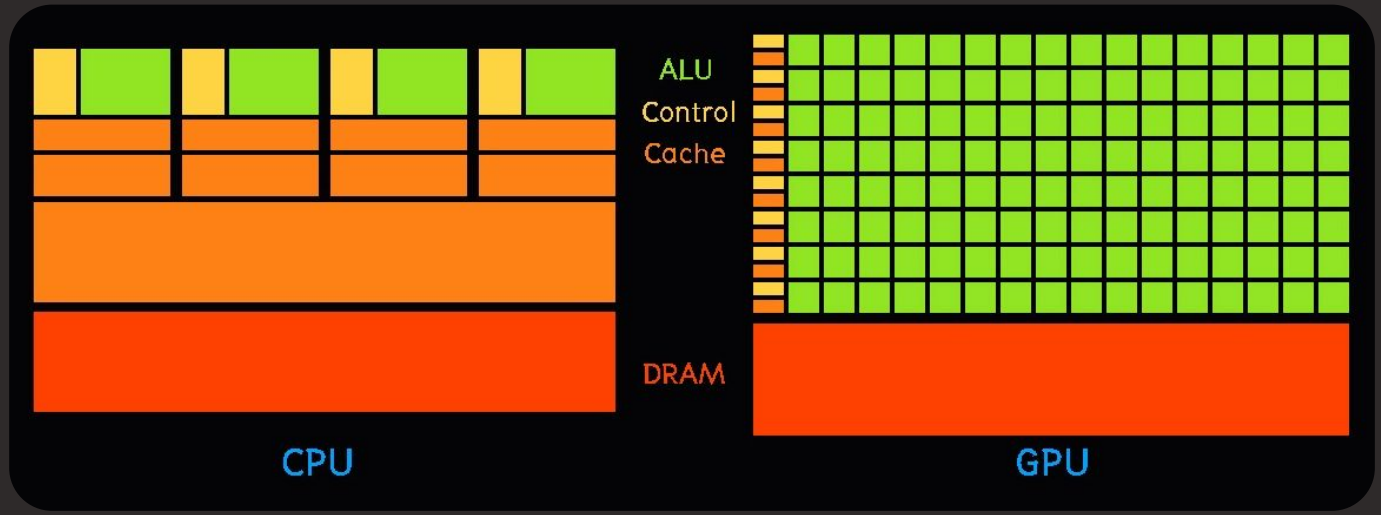
Introducción

Desafíos

Grupo Temporal

Graphics Processing Unit (GPU)

CPU y GPU: Qué las diferencia?

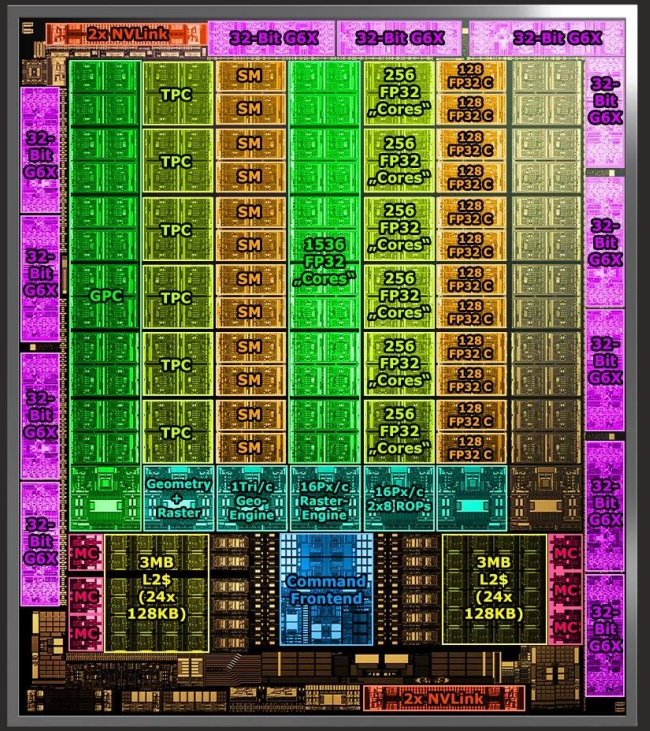


Facultad de Ciencias de la Ingeniería UACh

temporal

Chips en vista IR

CPU (Alder Lake 2021)

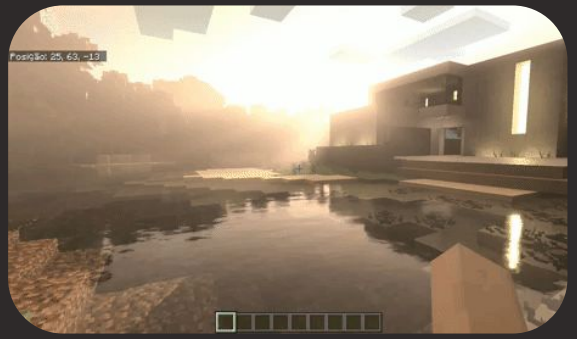


GPU (Ampere, GA102)

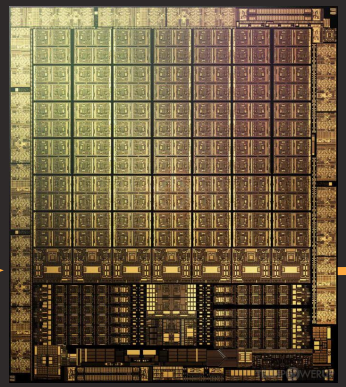
Facultad de Ciencias de la Ingeniería UACh

temporal

Videojuegos cumplen un rol clave



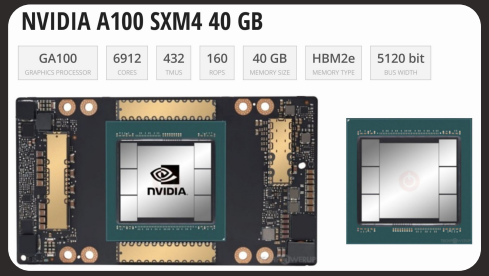
GPU: Arquitectura y Organización



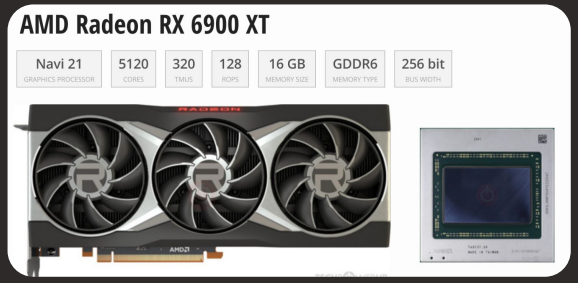
Tarjeta | Chip | Chip (zoom físico) | Chip (zoom lógico)

GPU: Arquitectura y Organización

Algunas GPUs gama alta 2021-2022



~19.5 TFLOPS (FP32)



~23 TFLOPS (FP32)

Intel Xe-HP (2022)

~42 TFLOPS (FP32)

~36 TFLOPS (FP32)

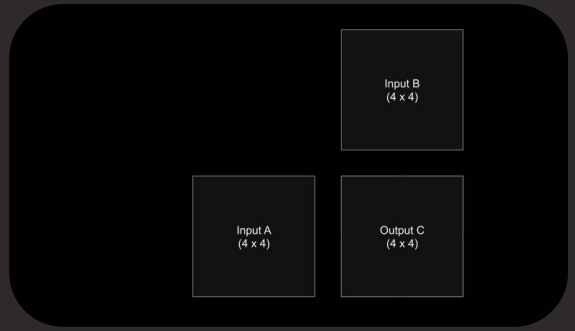
~23 TFLOPS (FP32)

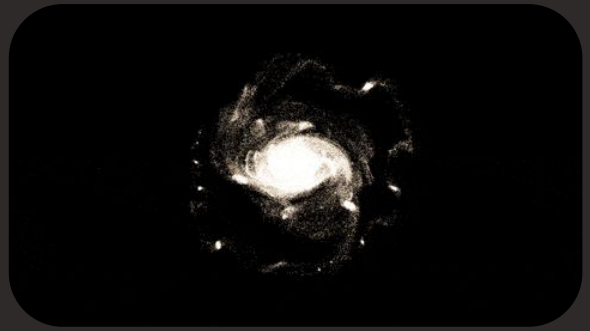
~2.3 TFLOPS (FP32)

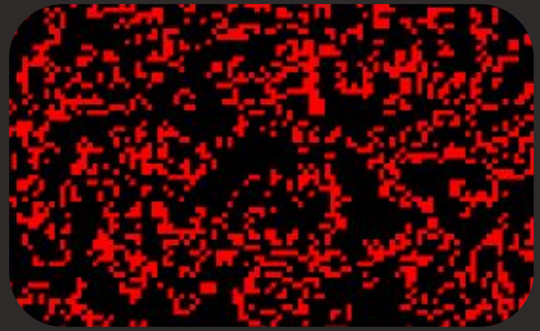
Facultad de Ciencias de la Ingeniería UACh

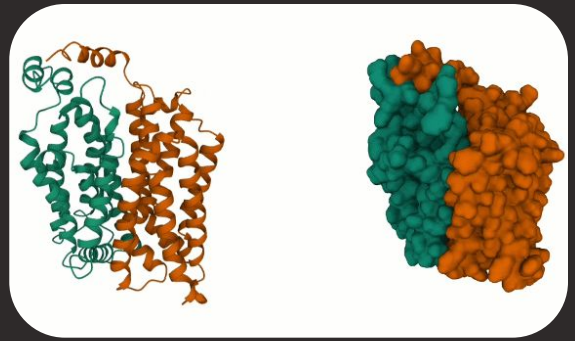
temporal

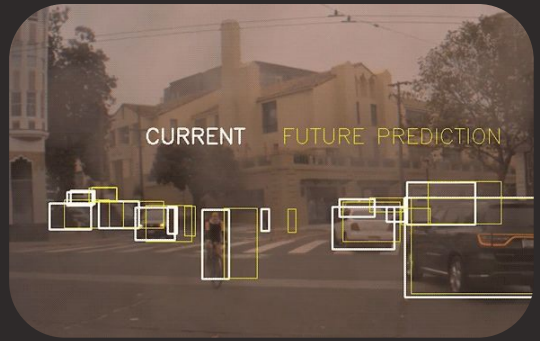
GPU Computing - Aplicaciones



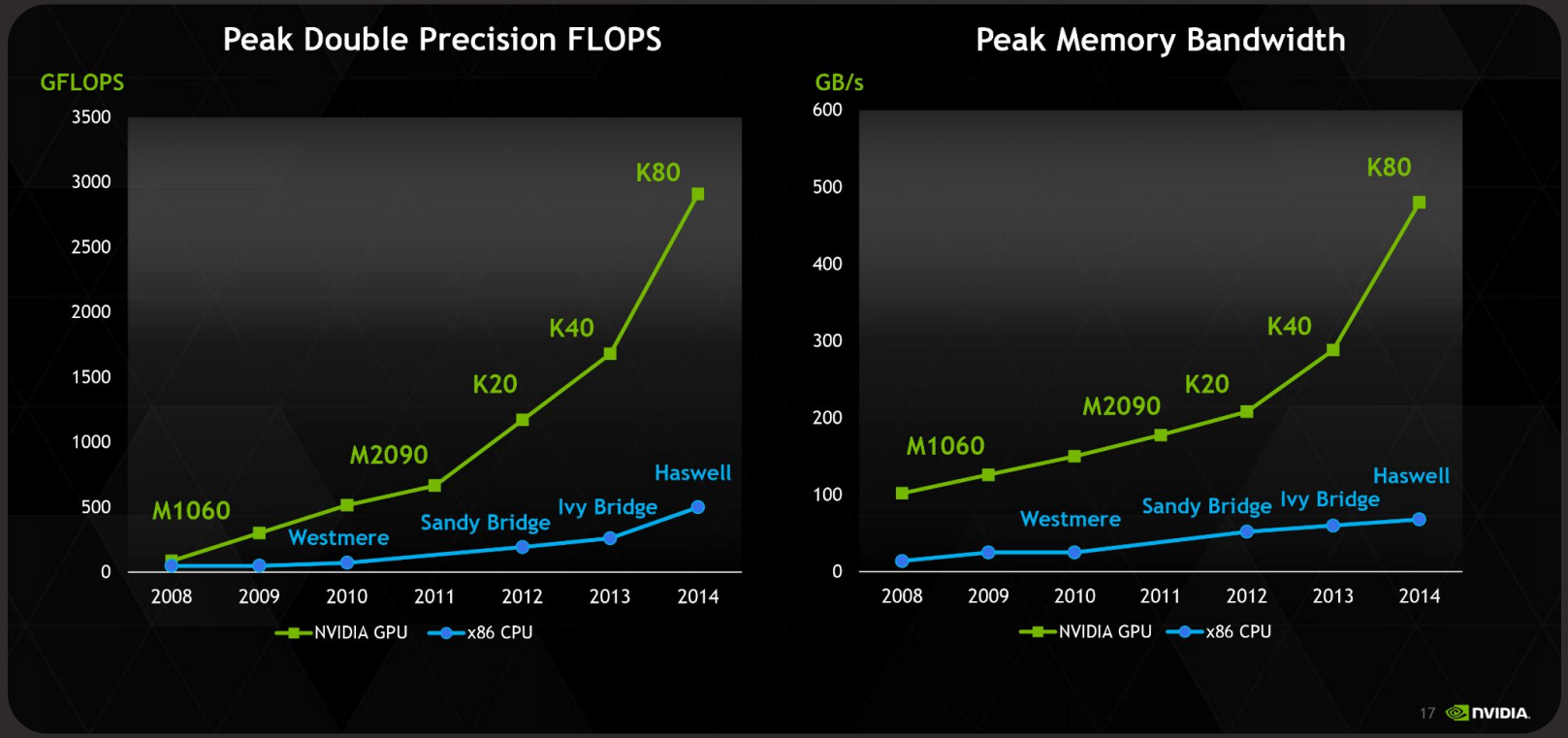




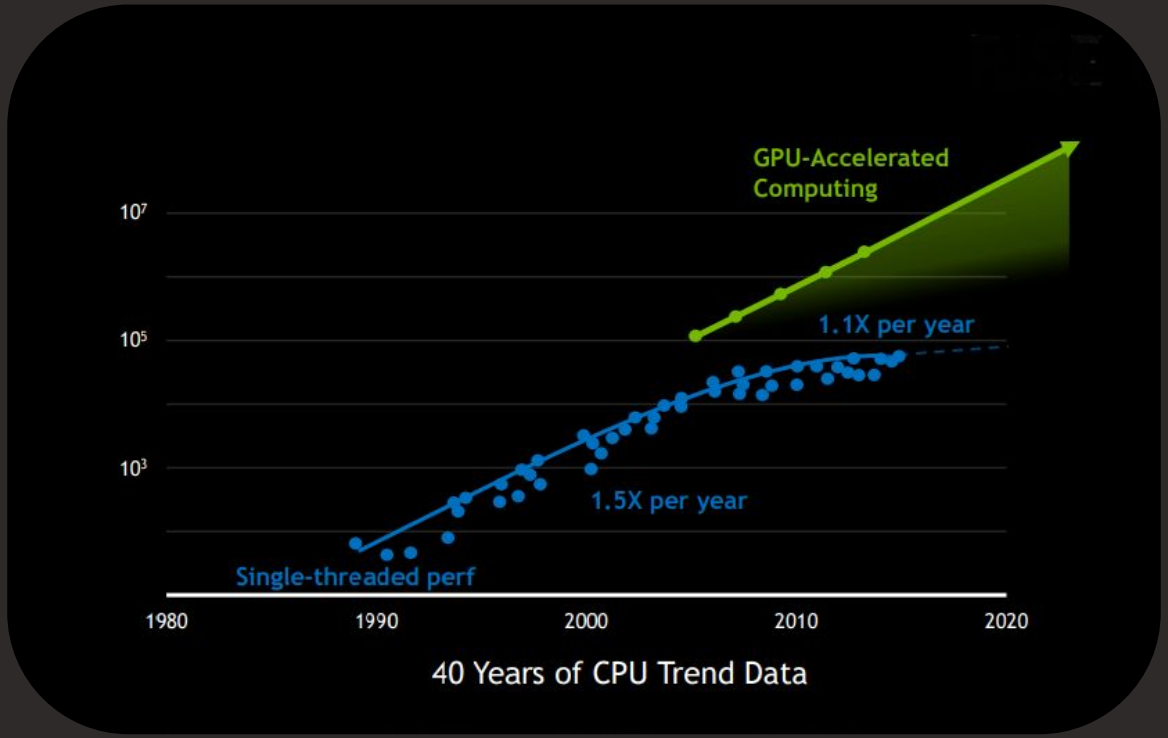




GPU Computing (motivación)



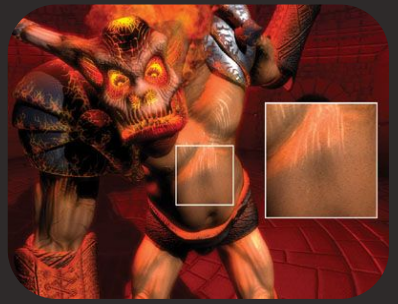
GPU Computing (proyección)

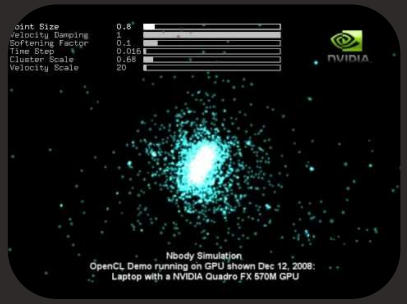


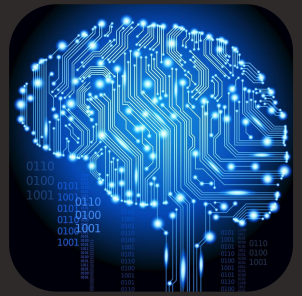
Facultad de Ciencias de la Ingeniería UACh

temporal

Estamos en la 3ª era de GPU Computing







2003

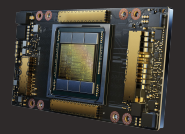
1ª era
(GPU Hacking)

2007

2ª era
(Programacion General GPU)

2017

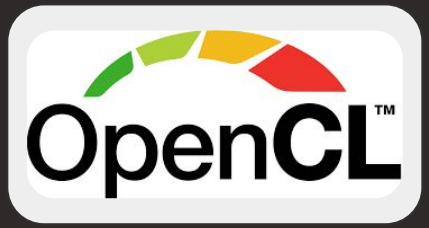
3ª era
(Programacion ASIC)



Facultad de Ciencias de la Ingeniería UACh

temporal

Programación en GPUs:



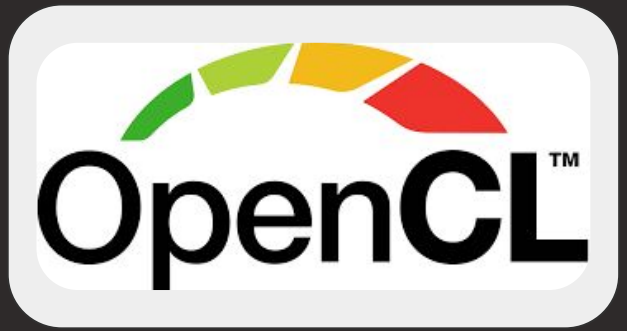
Abstracción específica

Abstracción HPC General

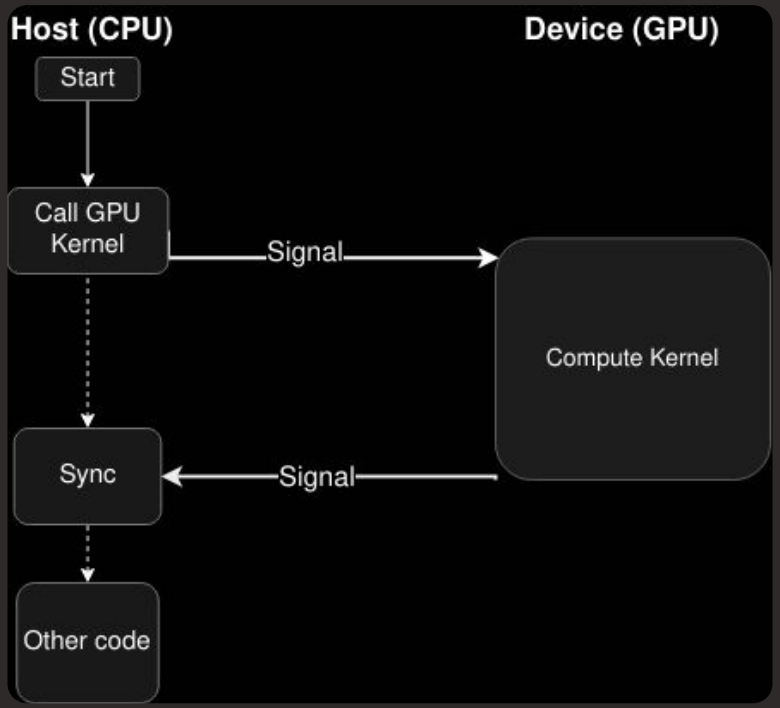
Facultad de Ciencias de la Ingeniería UACh

temporal

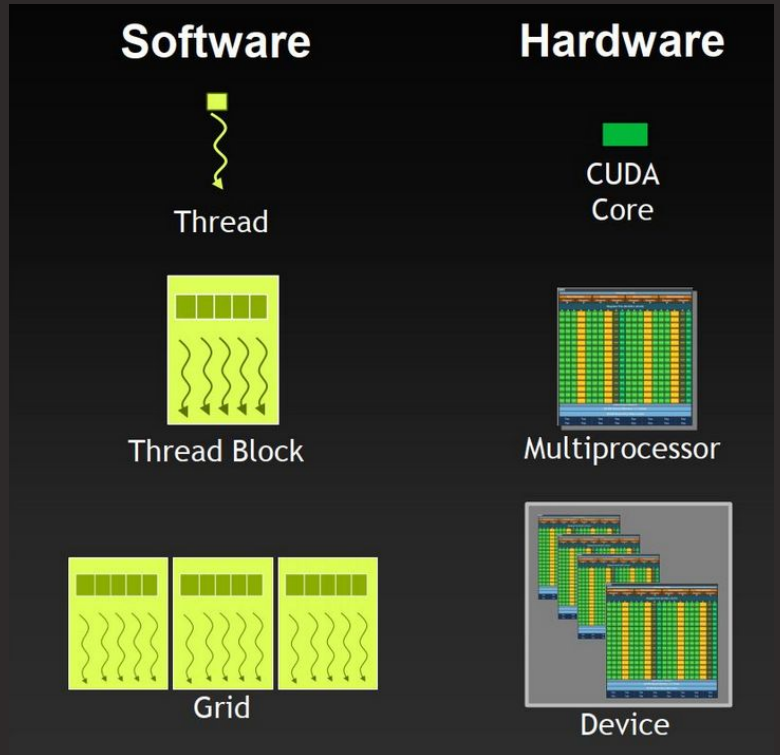
Modelo de programación GPU



Modelo de programación GPU #1

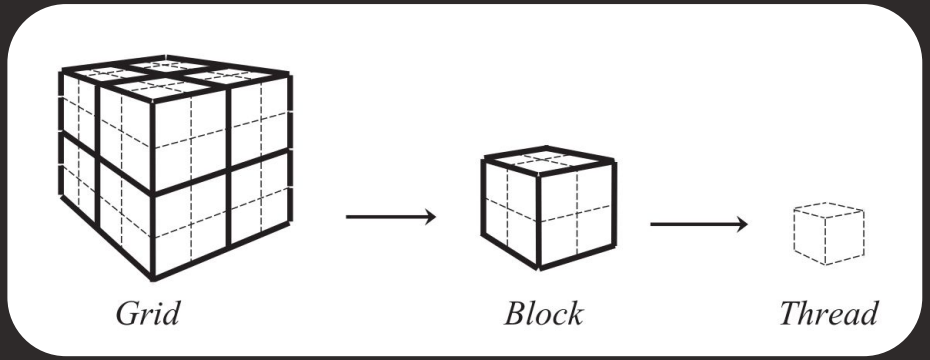


Modelo de programación GPU #2



Modelo de programación GPU #3

- El desarrollador se encarga de:
  - Mover datos Host ←→ Device
  - Definir Grid, block y threads
    - 1D, 2D o 3D.
  - Escribir Kernel
    - Código *per-thread.*
    - *SIMD a nivel micro.*
      - Warp (32 threads)
      - Sincronos por bloque.
    - *SPMD* a *nivel macro.*
      - Blocks asíncronos



Modelo de programación GPU #4

Ejemplo CUDA: `main.cu`

```
__global__ void saxpy_kernel( float a, float* x, float* y, int n ){
  int i;
  i = blockIdx.x*blockDim.x + threadIdx.x;
  if( i <= n ) x[i] = a*x[i] + y[i];
}
```

→ Parte Device (Kernel)

```
void saxpy( float a, float* x, float* y, int n ){
  float *xd, *yd;
  cudaMalloc( (void**)&xd, n*sizeof(float) );
  cudaMalloc( (void**)&yd, n*sizeof(float) ); cudaMemcpy( xd, x,
   n*sizeof(float),
                    cudaMemcpyHostToDevice );
  cudaMemcpy( yd, y, n*sizeof(float),
                    cudaMemcpyHostToDevice );
  saxpy_kernel<<< (n+31)/32, 32 >>>( a, xd, yd, n );
  cudaMemcpy( x, xd, n*sizeof(float),
                    cudaMemcpyDeviceToHost );
  cudaFree( xd ); cudaFree( yd );
}
```

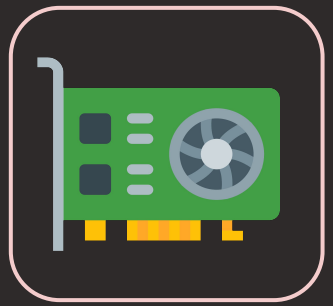
→ Parte Host

Compilacion:

```
nvcc -O3 main.cu -o prog
```

Facultad de Ciencias de la Ingeniería UACh

temporal



Introducción

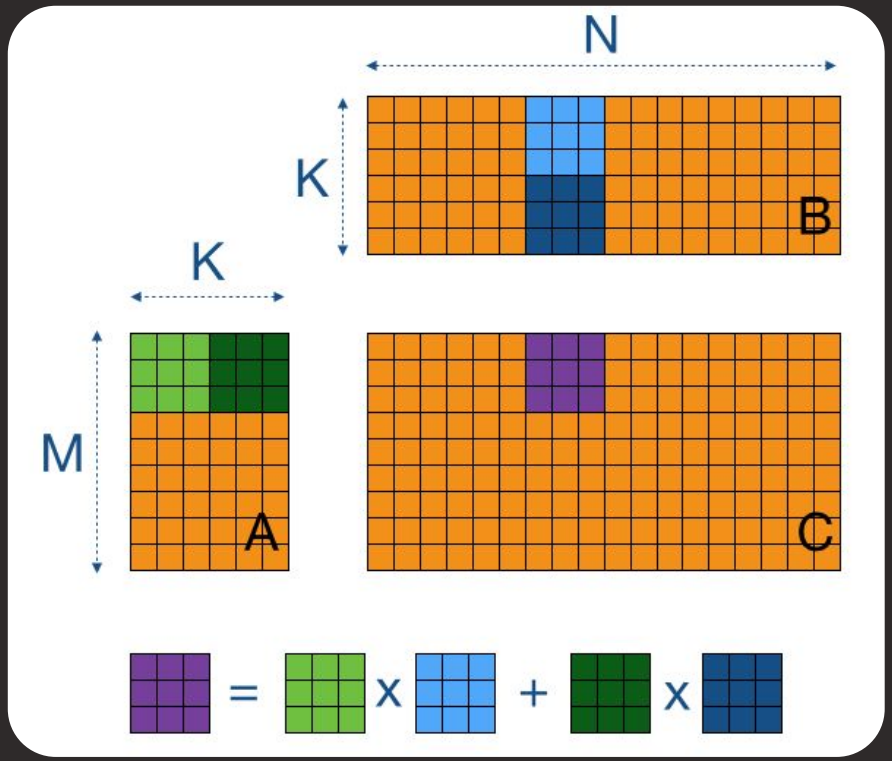
Desafíos

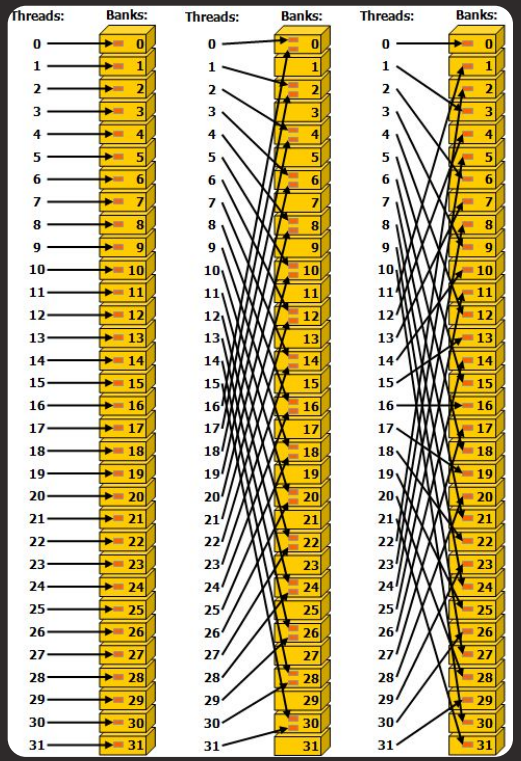
Grupo Temporal

Desafíos clásicos en GPU Computing

- Desafíos de computación paralela:
  - Encontrar casos de paralelización masiva
    - Algebra lineal (matrix multiply).
    - N-body problem.
    - Molecular dynamics.
    - PDEs (stencils).
  - Usos eficientes de caché L1 programmable.
    - Encontrar localidad
    - Evitar bank conflicts
  - Implementaciones multi-GPU eficientes.
  - Estructuras de datos jerárquicas eficientes en GPUs:
    - QuadTree, octree, BVH, kd-tree, k2-tree, etc
    - Usos exitosos de dynamic-parallelism (feature GPU)

Usos eficientes de caché L1 programable





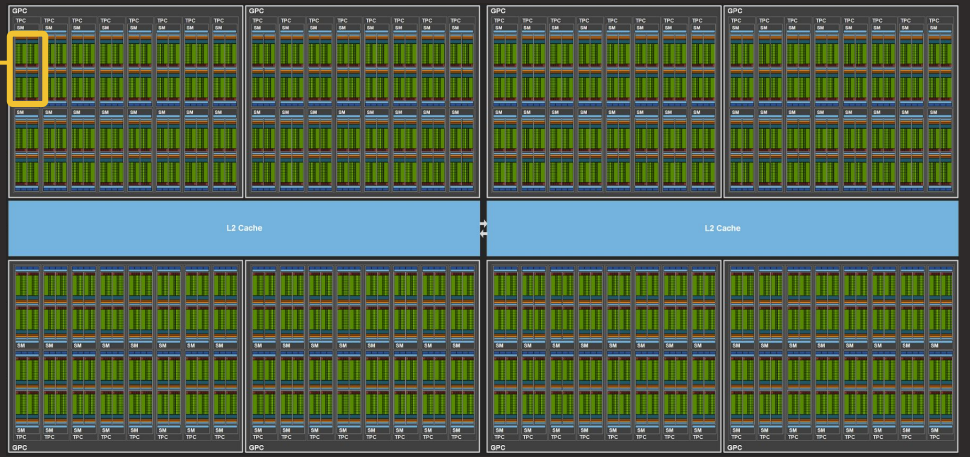
Desafíos del presente en GPU Computing

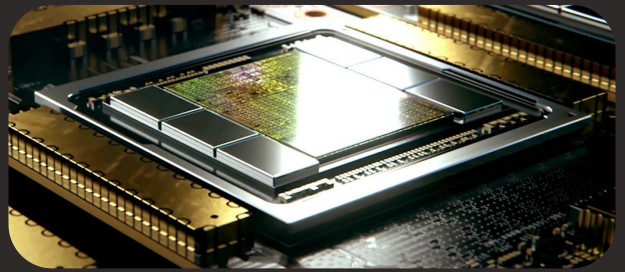
- Desarrollo de generadores / optimizadores de código paralelo
  - Full automaticos:
  - Semi-automaticos (directivas):
    - OpenACC, OpenMP, oneAPI

- Desarrollo y masificación de librerías / software:
  - cuBLAS
  - Thrust
  - Pytorch / TensorFlow
  - Gromacs

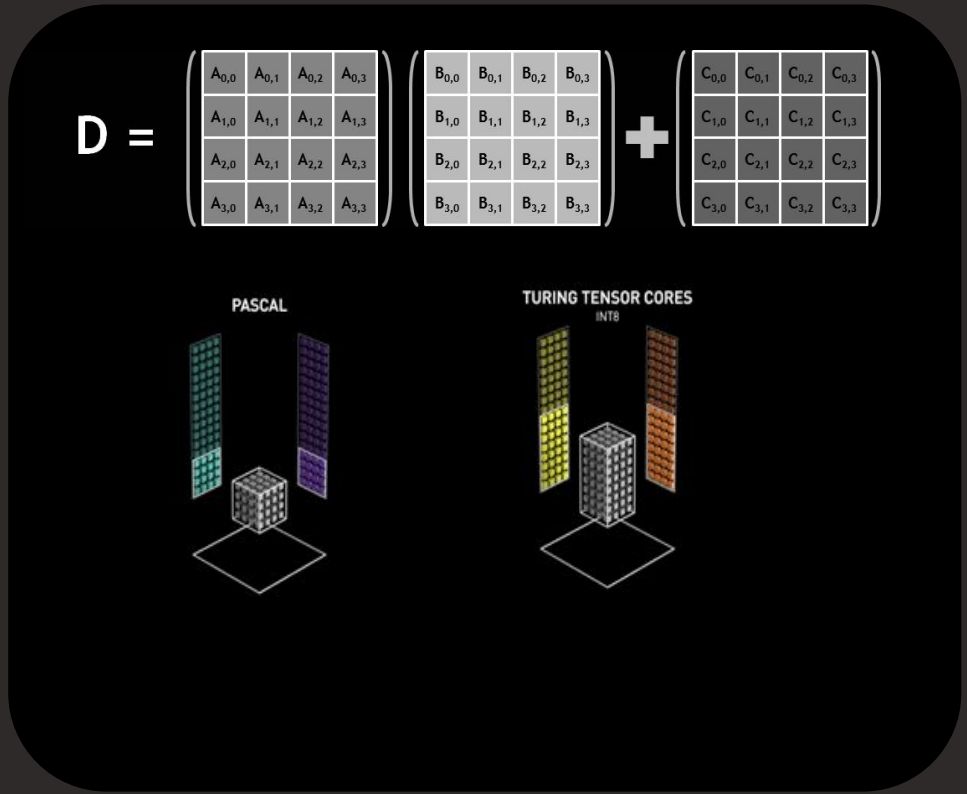
- Encontrar nuevos usos a los ASICs de GPU
  - Tensor-cores
  - RT cores.

Tensor Cores





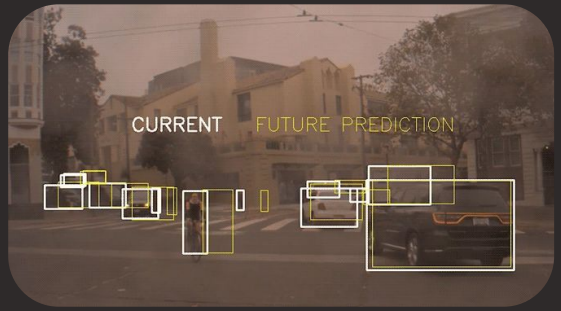
1 tensor core → 1 producto de matrices pequeñas (MMA)

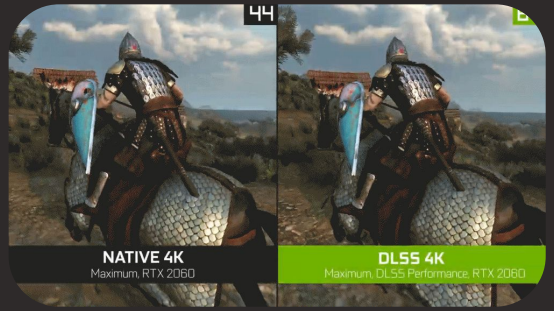


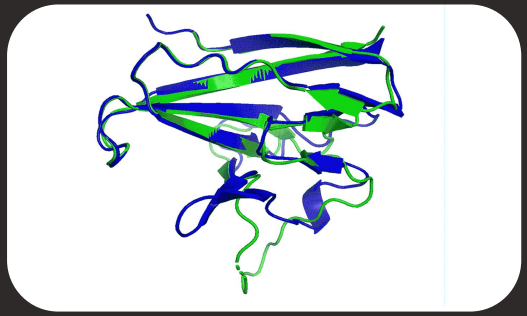
Facultad de Ciencias de la Ingeniería UACh

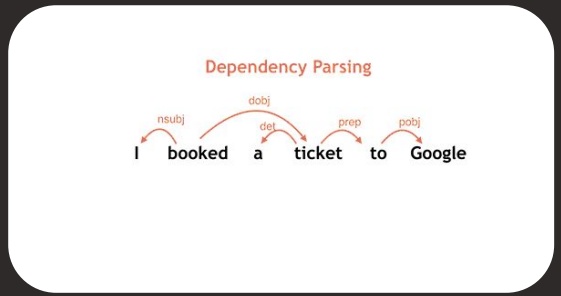
temporal

Uso natural de tensor cores









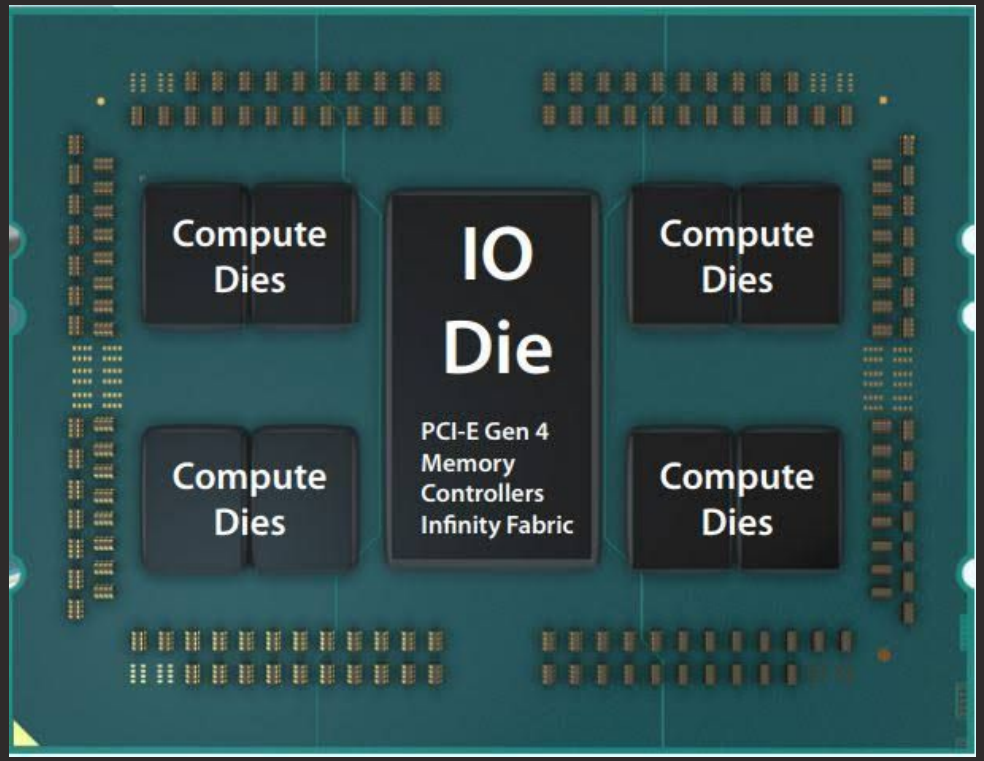
Desafíos del futuro en GPU Computing

- Desarrollar algoritmos energéticamente eficientes
  - Sacrificar un poco de rendimiento por eficiencia energética.

- Combinar el uso de ASICs de GPU.
  - Cores-regulares + tensor-cores + RT-cores.
  - Las aplicaciones guiarán el desarrollo de futuros ASICs.
    - Anticiparse a posibles nuevos ASICs.

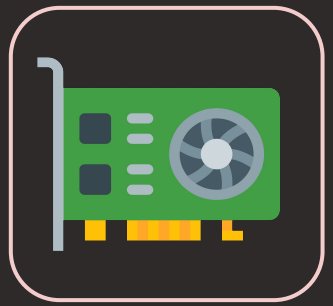
- Estudiar la eficiencia de los *multi-chip-module (MCM)* en el diseño de nuevos algoritmos paralelos.

Multi-chip-module (MCM)



Facultad de Ciencias de la Ingeniería UACh

temporal



Introducción

Desafíos

Grupo Temporal

Grupo Temporal

Core Researchers

Dr. Cristóbal A. Navarro
Instituto de Informática, UACh, Chile

Dr. Nancy Hitchfeld
DCC, Universidad de Chile, Chile

Dr. Hector Ferrada
Instituto de Informática, UACh, Chile

Collaborators

Dr. Ricardo Barrientos
Catholic University of Maule, Chile

Dr. Benjamin Bustos
DCC, Universidad de Chile, Chile

Dr. Miguel Romero
Adolfo Ibáñez University, Chile

Dr. Mario Gonzalez
Instituto Acustica, UACh, Chile

Dr. Benoit Crespin
University of Limoges, France

Students

Roberto Carrasco
PhD Student at DCC, Uchile, Chile.

Felipe Quezada
MSc student at MIN, UACh, Chile.

Roberto Melita
MSc student at MIN, UACh, Chile.

Rodrigo Stevenson
MSc student at MIN, UACh, Chile.

Enzo Meneses
MSc student at MIN, UACh, Chile.

Carlos Schoenfeldt
Undergrduate, UACh, Chile.

Diego Rojas
Undergraduate, UACh, Chile.

Facultad de Ciencias de la Ingeniería UACh

temporal

http://temporal.uach.cl

- Investigación en *GPU Computing*:
  - Aceleración de algoritmos de geometría/simulación.
  - Mapeo eficiente de threads.
  - Paralelismo eficiente en memoria dinámica.
  - Eficiencia energetica en GPUs.

Universidad Austral de Chile
Conocimiento y Naturaleza

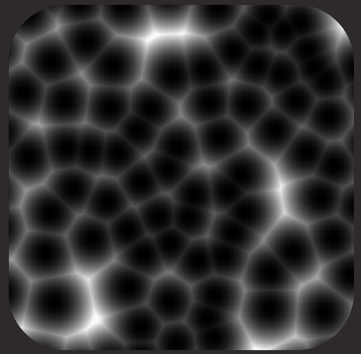
UNIVERSIDAD DE CHILE

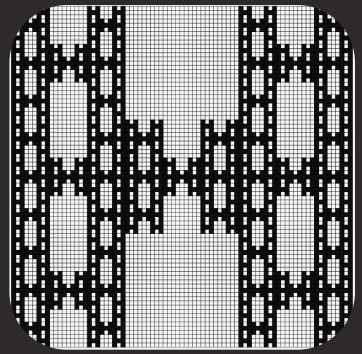
Facultad de Ciencias de la Ingeniería UACh

temporal

Puede el cómputo no-IA también beneficiarse de los tensor cores?

$$\sum_{i=1}^{n} x_i$$





[2018-2021] *Fondecyt iniciación Nº 11180881*

Gracias

Repositorio Temporal: https://github.com/temporal-hpc

Investigación #1

Sumatoria por tensor core

$$\sum_{i=1}^{n} x_i$$

Idea clave #1: un par de MMAs pueden sumar m^2 datos.

$$D = J_m \times \begin{bmatrix} x_{11} & \cdots & x_{1m} \\ \vdots & \ddots & \vdots \\ x_{m1} & \cdots & x_{mm} \end{bmatrix} + 0_m \tag{9}$$

$$= \begin{bmatrix} \sum_{i=1}^{m} x_{1i} & \cdots & \sum_{i=1}^{m} x_{1i} \\ \vdots & \ddots & \vdots \\ \sum_{i=1}^{m} x_{mi} & \cdots & \sum_{i=1}^{m} x_{mi} \end{bmatrix} \tag{10}$$

MMA #1

$$D' = \begin{bmatrix} \sum_{i=1}^{m} x_{1i} & \cdots & \sum_{i=1}^{m} x_{1i} \\ \vdots & \ddots & \vdots \\ \sum_{i=1}^{m} x_{mi} & \cdots & \sum_{i=1}^{m} x_{mi} \end{bmatrix} \times J_m + 0_m \tag{11}$$

$$= \begin{bmatrix} \sum_{i=1}^{m} \sum_{j=1}^{m} x_{ij} & \cdots & \sum_{i=1}^{m} \sum_{j=1}^{m} x_{ij} \\ \vdots & \ddots & \vdots \\ \sum_{i=1}^{m} \sum_{j=1}^{m} x_{ij} & \cdots & \sum_{i=1}^{m} \sum_{j=1}^{m} x_{ij} \end{bmatrix} \tag{12}$$

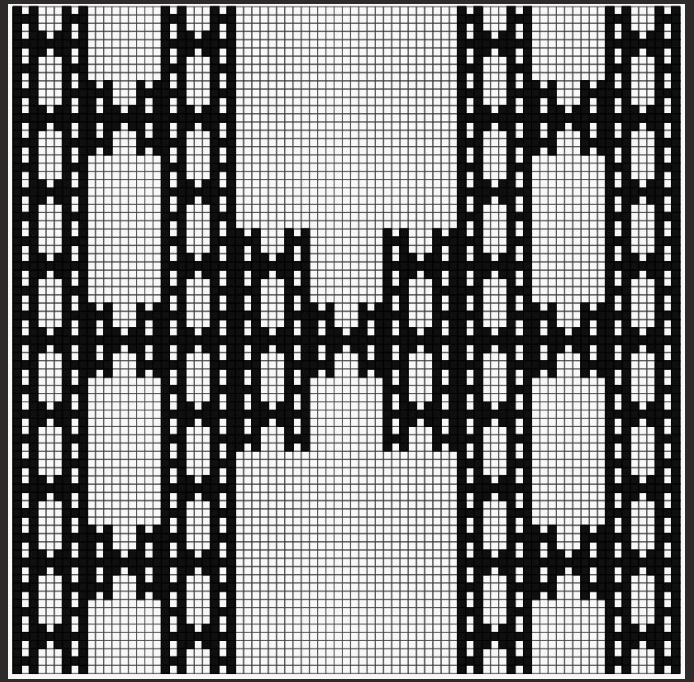
MMA #2

Facultad de Ciencias de la Ingeniería UACh

temporal

Investigación #2

Mapeo de *threads* de GPU a geometría fractal discreta



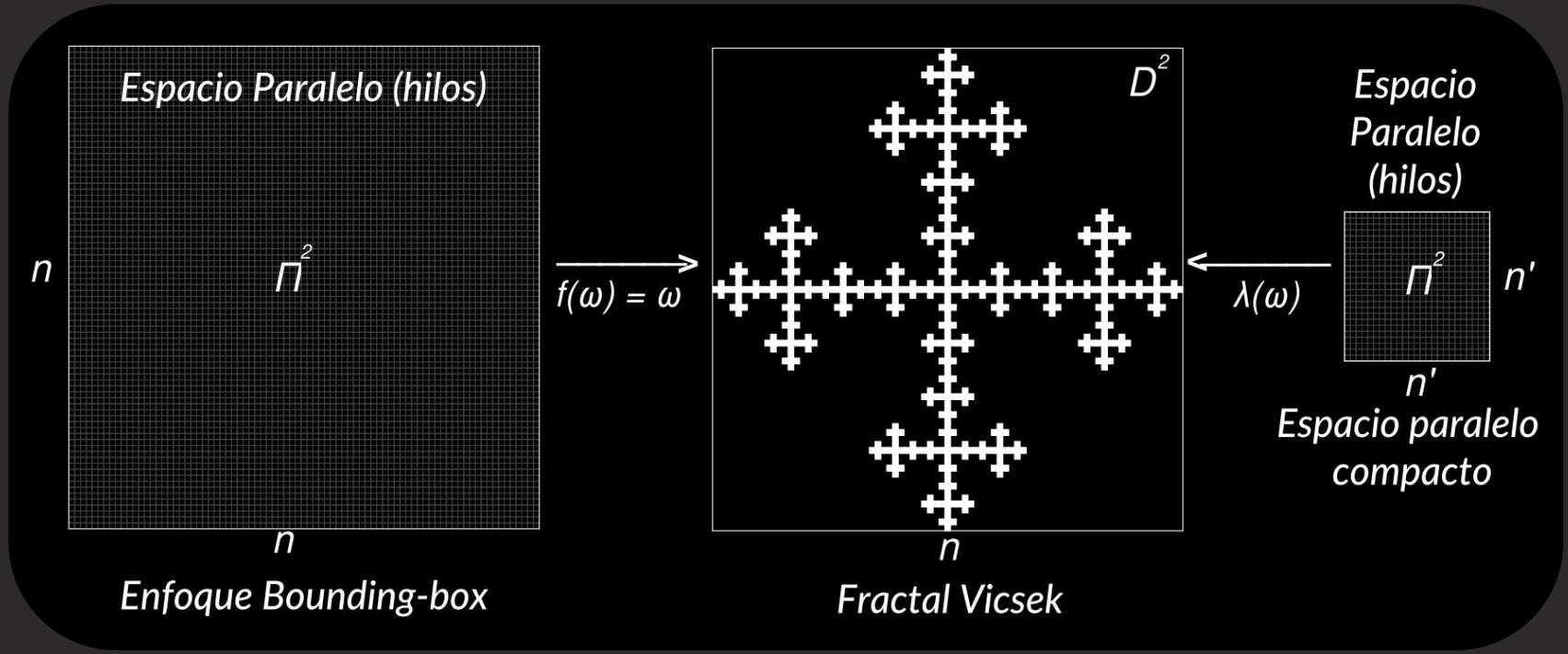
Felipe Quezada S.

Facultad de Ciencias de la Ingeniería UACh

temporal

Idea estandar

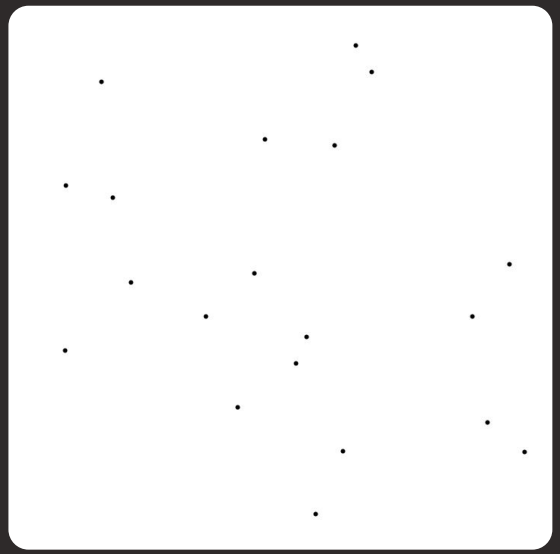
Idea propuesta

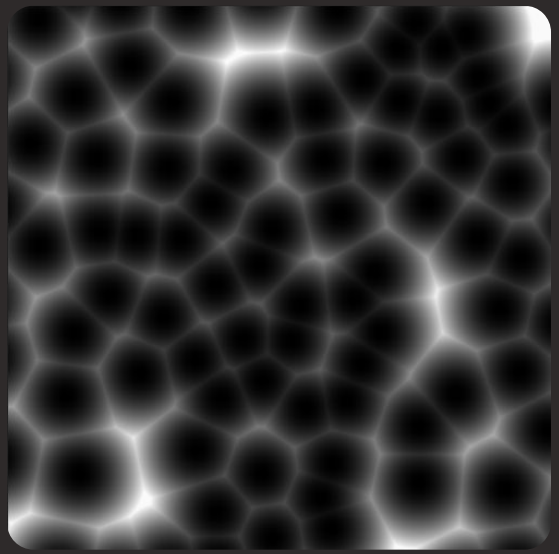


Felipe Quezada S.

Investigación #3 (en curso)

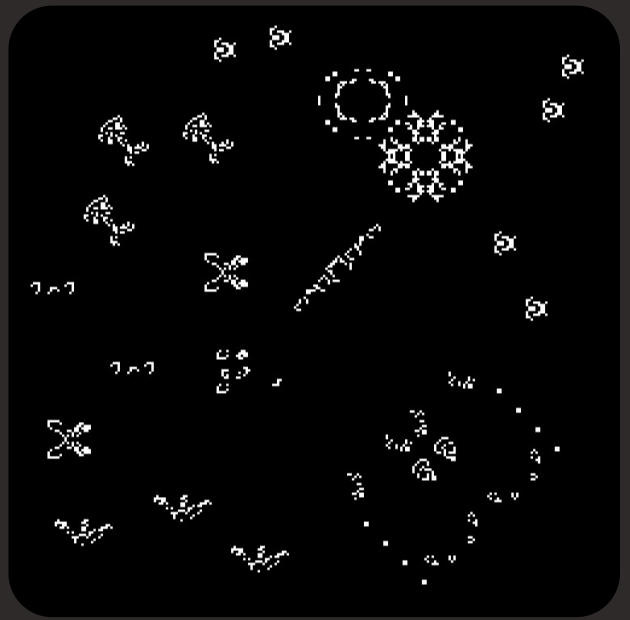
Diagramas de Voronoi dinámicos acelerados por GPU





Investigación #4 (en curso)

Simulación de autómatas celulares con tensor cores



Investigación #5 (en curso)

Manejo eficiente de memoria dinámica en GPU

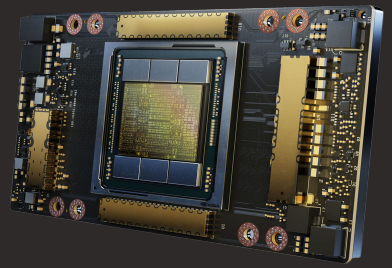
Approach #1: Rápido ✓ | Mucha memoria ✗ | Sparse ✗

Approach #2: Lento ✗✗ | Minima Memoria ✓ | Compacto ✓

Investigación #6 (en curso)

Caracterización de la eficiencia energética de GPUs.

- Resultados preliminares confirman alta eficiencia energética con GPU.
- Actualmente obteniendo resultados experimentales con/sin tensor cores.



Carlos Schoenfeldt Arriaza