IE–TEC

Instituto Tecnológico de Costa Rica
Escuela de Ingeniería Electrónica
MP-6123 Procesamiento Digital de Imágenes
Profesor: MSc. Carl Michael Gruner Monzon
II Cuatrimestre, 2016

Proyecto 1

Instituto Avanzado de Biotecnología
Comparación de Estrategias de Optimización

1. Problema

El *IAB* o Instituto Avanzado de Biotecnología está incursionando en el estudio de procesamiento digital de señales como parte de su estrategia de investigación. El preprocesamiento de imágenes es una etapa fundamental en nuestros procesos. Típicamente se requieren diferentes etapas de filtrado previo a los algoritmos principales. Como tal, la convolución es un pilar para nuestros procesos.

Se ha decidido contratar a su empresa para realizar una investigación acerca de diferentes técnicas de optimización para el algoritmo de la convolución y realizar una publicación de los resultados en nuestro instituto. Hasta el momento, nuestro equipo tiene conocimiento de la implementación de la convolución a nivel teórico e inclusive ha realizado implementaciones propias a nivel de lenguaje C++. Se requiere que su empresa realice un estudio de implementaciones utilizando técnicas avanzadas de optimización de código que aprovechen la arquitectura del procesador tales como SIMD, técnicas modernas de concurrencia basadas en el GPU tales como OpenCL, CUDA, OpenACC o técnicas basadas en procesadores de múltiples núcleos como OpenMP.

Se espera que con su estudio y publicación nuestro equipo tenga una visión más completa de diferentes técnicas de optimización modernas y avanzadas, junto con una comparación cuantitativa de rendimiento, que puedan ser aplicados en algoritmos futuros para cumplir los requerimientos de un mercado cada vez más exigente.

2. Especificación del Proyecto

2.1. Convoluciones

Se requiere realizar el estudio de la convolución utilizando cuatro tipos específicos de núcleos:

- Núcleo de longitud impar:

$$\mathbf{h} = \begin{pmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{01} & a_{02} \\ a_{20} & a_{21} & a_{22} \end{pmatrix}$$

- Núcleo de longitud par:

$$\mathbf{h} = \begin{pmatrix} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{10} & a_{11} & a_{12} & a_{13} \\ a_{20} & a_{21} & a_{22} & a_{23} \\ a_{30} & a_{31} & a_{32} & a_{33} \end{pmatrix}$$

- Núcleo separable de longitud impar:

$$\mathbf{h} = \begin{pmatrix} a_{00} \\ a_{10} \\ a_{20} \end{pmatrix} (b_{00} b_{01} b_{02})$$

- Núcleo separable de longitud par:

$$\mathbf{h} = \begin{pmatrix} a_{00} \\ a_{10} \\ a_{20} \\ a_{30} \end{pmatrix} (b_{00} b_{01} b_{02} b_{03})$$

2.2. Marcos de Trabajo

Para cada una de las diferentes convoluciones, se deberá realizar el estudio utilizando los diferentes marcos de trabajo:

2.2.1. C/C++ Pleno

Implementación en C/C++ sin ninguna optimización. Esta implementación del algoritmo servirá como referencia para las optimizaciones posteriores.

2.2.2. SIMD

SIMD, o *Single Instruction Multiple Data* por sus siglas en inglés, es una clase de arquitectura que permite la ejecución de una operación sobre mútiples estructuras de datos en paralelo. Dicho paralelismo se logra sin entrar en concurrencia ya que todas las operaciones se realizan en un mismo hilo de ejecución.

La implementación de módulos SIMD es diferente para cada procesador. La Tabla 1 resume algunas de las arquitecturas más populares:

Típicamente la implementación mediante SIMD requiere la ejecución de instrucciones mediante ensamblador directamente, o indirectamente mediante Intrinsics. De manera similar, típicamente se tienen requerimientos especiales en cuanto al alineamiento de memoria utilizada.

Tabla 1: Arquitecturas SIMD más populares

Fabricante	Módulo SIMD
Intel	MMX, SSE, SSE2, SSE3, SSE4, AVX
AMD	3DNow!
ARM	NEON
MIPS	MDMX

2.2.3. Aceleración por GPU

Enfoques más modernos de optimización aprovechan técnicas de programación concurrente para ejecutar una tarea costosa, mediante múltiples tareas de menor tamaño paralelas. Dado el auge de los procesadores gráficos y su gran cantidad de unidades de procesamiento, se ha empezado a utilizar los mismos para optimizar tareas que requieren gran cantidad de recursos computacionales. Los fabricantes de los mismos han modificado dichas arquitecturas para facilitar estas operaciones. Se le ha denominado a este enfoque GPGPU (*General Purpose Graphics Processing Unit*).

Existen diferentes herramientas para codificar algoritmos concurrentes aprovechando las unidades de ejecución del GPU. Si bien, se ha tratado de abstraer el proceso de optimización para diferentes arquitecturas mediante un mismo marco de trabajo, todavía existen marcos de trabajo proprietarios dedicados a una arquitectura en específico. La Tabla 2 resume algunos de los marcos de trabajo más populares, su portabilidad y el hardware de operación.

Tabla 2: Herramientas GPGPU más populares

Heramienta	Portabilidad	Hardware
CUDA	NVidia únicamente	GPU
OpenCL	Portable	GPU, DSP, FPGA
OpenACC	NVidia únicamente	GPU
OpenMP	Portable	CPU (multi-núcleo)

3. Entregables y Fecha Límite

- Código fuente de algoritmo de referencia en C/C++.
- Código fuente de algoritmo utilizando, al menos, una arquitectura SIMD.
- Código fuente de algoritmo utilizando, al menos, una marco de trabajo GPGPU.
- Artículo científico en el formato estándar de la IEEE con extensión de no más de 5 páginas.
- Opcional: Historial de sistema de control de versiones utilizado (extra)

El proyecto tiene una extensión de dos semanas. Dadas las vacaciones del TEC (del 4 al 17 de julio), el proyecto deberá entregarse el lunes 25 de julio antes de las 23:59.