

INSTITUTO TECNOLÓGICO DE COSTA RICA
ÁREA ACADÉMICA DE INGENIERÍA EN COMPUTADORES
PROYECTO DE DISEÑO EN INGENIERÍA EN COMPUTADORES



Computación aproximada en aplicaciones tolerantes a errores con una plataforma multinúcleo y multi-acelerador

Chair for Embedded Systems (CES)
Karlsruhe Institute of Technology (KIT)

Propuesta de anteproyecto

DANIEL MOYA SÁNCHEZ

1 de marzo de 2018

1. Introducción

Los sistemas de Tecnologías de Información (TI) buscan dar una mejor calidad de vida a las personas, en esta tarea han tenido que enfrentar ciertos problemas, entre los más comunes se puede mencionar el costo en área, potencia, tiempo de ejecución, variables que restringen el rendimiento de un chip. Idealmente, una aplicación debe ajustarse a las necesidades reales del usuario y, en general, del área de aplicación, de forma que se dé un uso óptimo de los recursos. Actualmente, el diseño de procesadores no se enfoca tanto en tener más desempeño si no tener un manejo de recursos apropiado; no obstante, algunos desafíos en este campo vienen dados por limitaciones físicas, por ejemplo, las características eléctricas de los CMOS, las cuales restringen el consumo de energía en sistemas embebidos y es un aspecto que deben considerar los diseñadores de componentes para propósito específico; la pared de memoria, que corresponde a la diferencia entre el crecimiento de la capacidad de procesamiento contra la velocidad de obtención de datos desde memoria; y la pared de utilización, la cual limita el uso máximo de hardware simultáneo debido a las capacidades de disipación de calor de un sistema.

Para poder atacar los problemas mencionados anteriormente, una de las áreas de investigación actuales corresponde a la Computación Aproximada, un paradigma de diseño que propone una reducción en la precisión o exactitud de la computación para obtener oportunidades de mejora en cuanto al consumo de área, potencia y tiempo de ejecución. Para aplicar dicho paradigma se ocupa la identificación de aplicaciones tolerantes a errores, y más específicamente, cuáles secciones o funciones dentro de estas pueden ser sustituidas por versiones aproximadas, de forma que se pueda generar un balance entre la calidad de la salida y el consumo general de recursos.

1.1. Descripción del problema

A la hora de tener una aplicación con una estructura en pipeline, es decir, que posee una serie de etapas, donde cada etapa recibe su entrada de una etapa anterior y produce una salida para la etapa siguiente, y donde una o más etapas pueden ser aproximables con más de una versión aproximable (una versión se puede concentrar en mejorar el consumo de potencia, mientras que otra el tiempo de ejecución, por ejemplo, resulta complejo determinar qué combinación de versiones aproximadas utilizar de forma que no se sobrepase el máximo error permitido y a la vez se reduzca de manera óptima, el uso de ciertos recursos. Dicho proceso podría tomar una cantidad considerable de tiempo si se decide probar todas las combinaciones posibles de versiones aproximadas, por lo que es importante utilizar un esquema diferente de trabajo.

Una aplicación puede tener un comportamiento aproximado si alguna de sus etapas se puede aproximar, ya sea toda una sección o únicamente una instrucción (dentro de una sección). La figura 1 muestra una aplicación genérica donde ambas situaciones pueden ocurrir.

Como se ve en la figura 1, esta aplicación posee tres secciones, de las cuales la primera (ej.

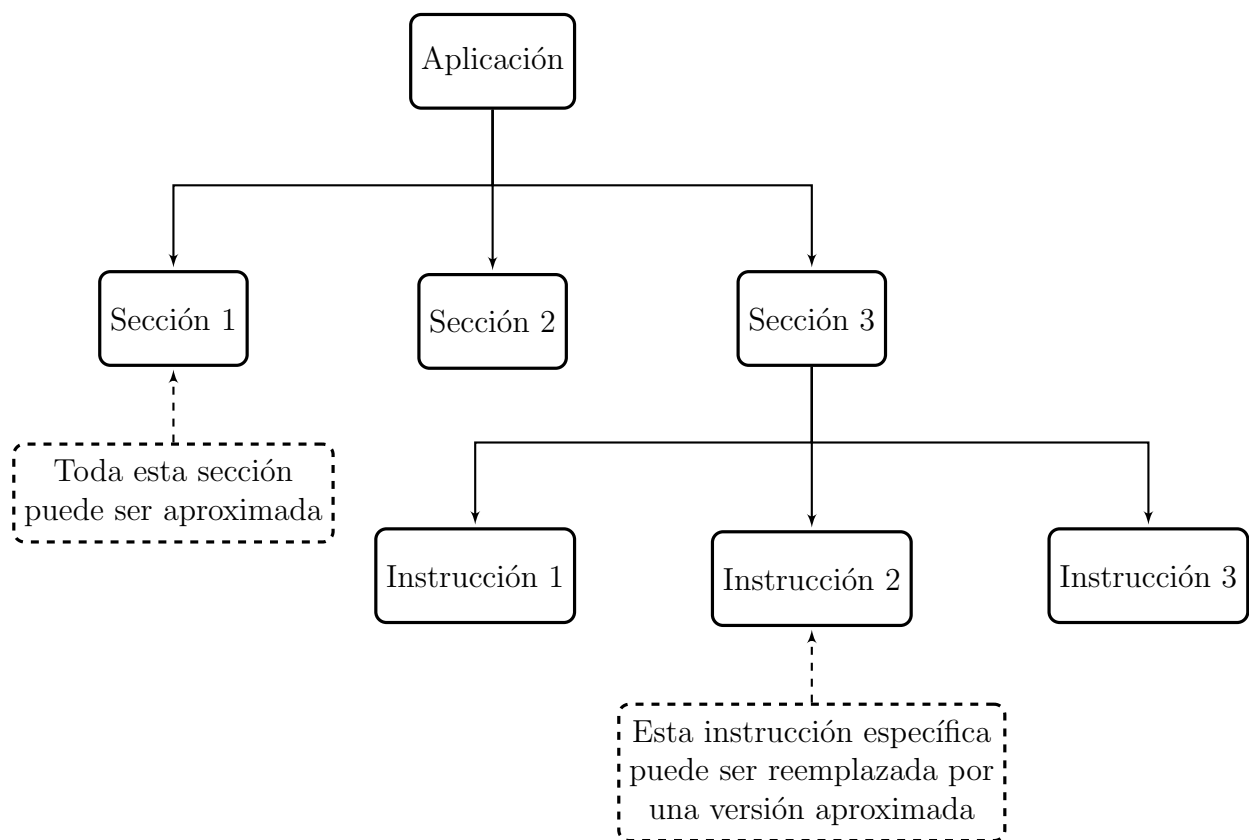


Figura 1: Una posible situación a resolver con este proyecto.

una etapa de preprocesamiento) puede ser completamente aproximada, la segunda no puede ser aproximada del todo (ej. una sección crítica de la aplicación) y, finalmente, la tercera tiene tres instrucciones específicas, de las cuales únicamente la segunda posee una versión aproximada.

2. Enfoque de la solución

Se busca desarrollar una herramienta de software que pueda escoger entre diferentes versiones una aplicación aproximada (cada versión dada por una combinación diferente de versiones específicas para cada sección aproximable), según el criterio de un usuario que especifique cuáles recursos son críticos en la aplicación y cuál es la cantidad máxima de error permitido. La figura 2 muestra a grandes rasgos el esquema de implementación de esta herramienta.

Como se muestra en la figura 2, se espera recibir un código fuente de una aplicación aproximable, donde previamente el usuario ha indicado, a través de pragmas propios, qué funciones del código son tolerantes a errores. Este código es compilado a una representación intermedia, a través del algoritmo que se busca desarrollar, para poder escoger cuáles versiones aproximadas de las posibles que existen se ajustan para cumplir a cabalidad con el error

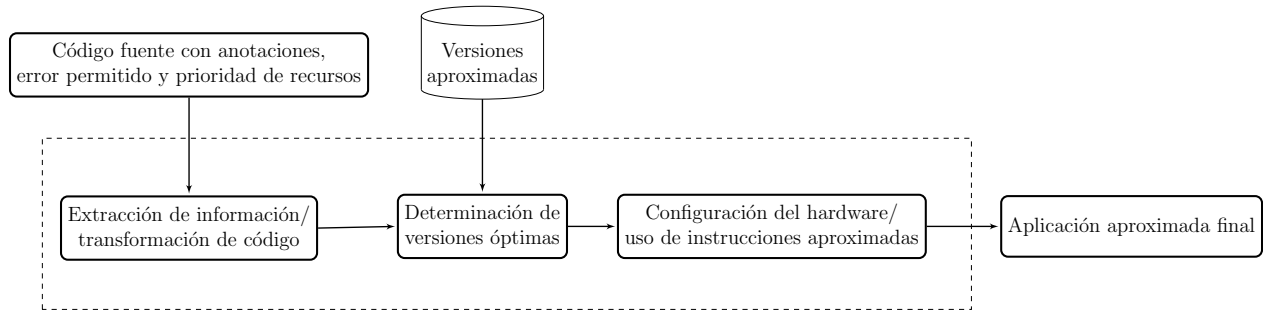


Figura 2: Esquema general de la solución propuesta.

permitido y, de la mejor manera, con la priorización de recursos. Una vez identificadas las versiones que serán utilizadas, se procede a utilizar el hardware específico que soporte las instrucciones aproximadas correspondientes, para finalmente entregar una aplicación final aproximada.

2.1. Objetivos

2.1.1. Objetivo General

Desarrollar una herramienta que, a partir de información sobre diferentes versiones de secciones aproximadas de una aplicación tolerante a errores, pueda determinar cuál combinación de versiones genera un mejor resultado en términos de ahorro de recursos y el nivel de error máximo que un usuario estableció como permitido.

2.1.2. Objetivos Específicos

1. Generar una representación del código de entrada que sea manipulable a partir de las anotaciones dadas por un usuario.
2. Evaluar las versiones aproximadas de las secciones de una aplicación para saber cuál es su impacto en la aplicación final.
3. Desarrollar un algoritmo que permita, a partir de funciones indicadas por un usuario, la escogencia de secciones tolerantes a errores en una aplicación según las posibles versiones aproximadas existentes.

3. Propuesta Metodológica

3.1. Tipificación del trabajo a realizar

El proyecto se clasifica como un trabajo de investigación cualitativa, donde se deben aplicar conocimientos en temas relacionados con sistemas operativos, arquitectura de computadores, computación de alto rendimiento y compiladores.

3.2. Descripción del proceso a realizar

El proyecto iniciará con una etapa de investigación sobre trabajos realizados por varios autores en el área de computación aproximada, relacionados con la caracterización de sistemas o aplicaciones en las cuales una o varias secciones son aproximables. Se investigará sobre maneras de generar un compilador para un cierto lenguaje y con el reconocimiento de pragmas o anotaciones en el código.

Seguidamente, se implementará un algoritmo que, a partir de una información dada (gracias a una base de datos) determine qué combinación de funciones aproximadas se deben colocar en un sistema en pipeline de tal forma que el resultado al final de todas las etapas se mantenga en un nivel de error aceptable. Para esto se tomará información de una base de datos del KIT sobre secciones aproximadas independientes, para posteriormente evaluar el impacto final de cada una de ellas en una aplicación completa.

La documentación del proyecto se trabajará a lo largo de todo el proceso de desarrollo, sin embargo, la redacción de un artículo científico y demás documentos propios de un trabajo final de graduación, se desarrollarán como una etapa adicional.

4. Antecedentes y trabajos relacionados

En la actualidad, dada la gran cantidad de aplicaciones complejas (por ejemplo sistemas GPS, reconocimiento de voz, etc.) la computación aproximada ayuda a mantener una salida aceptable mientras se logra que ciertas métricas como tiempo de respuesta o eficiencia energética se mejoren. En general, la computación aproximada provee la libertad de escoger entre un cierto nivel de error o degradación de la calidad en la salida final de una aplicación (por ejemplo ruido en la señal de la salida) para mejorar el consumo de energía, el área o el tiempo de ejecución; esto sirve como herramienta a un investigador para que ajuste una aplicación dada a las necesidades reales y específicas de esta. En la figura 3 se muestra un esquema que puede ser aplicado a sistemas tolerantes a errores para incluir en estos la computación aproximada [1].

Los elementos clave de la figura 3 son *kernels* aproximados, los cuales representan la implementación (técnicas) de las funciones aproximadas, estas pueden ser realizadas a nivel

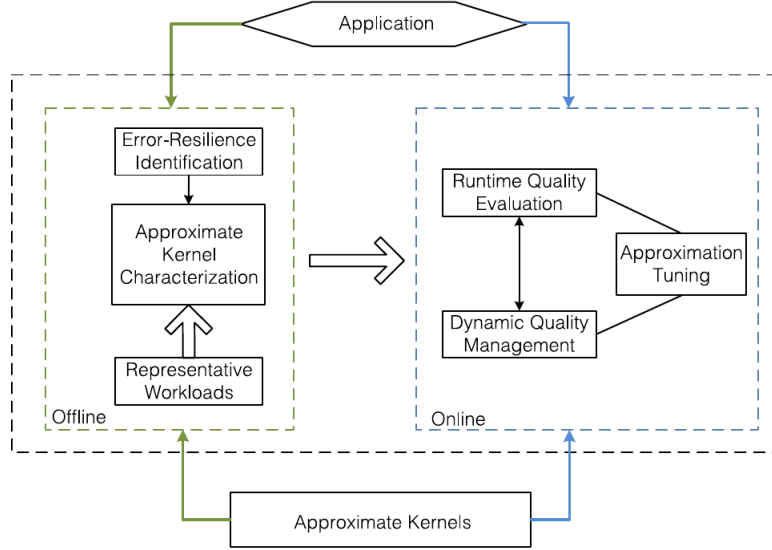


Figura 3: Un marco de trabajo para el uso de computación aproximada [1].

de hardware o de software; la identificación de las secciones tolerantes a errores y sus características particulares (análisis de impacto); y el manejo de la calidad, el cual implica una evaluación continua para determinar si la aplicación logra los requerimientos deseados.

Como se mencionó, la computación aproximada puede ser implementada tanto a nivel de software como de hardware. En software una implementación típica es a través de *Loop Perforation*, en la cual ciertos ciclos (usualmente con un patrón dado, como por ejemplo las pares) no son computados, lo cual, por ejemplo en una aplicación de cálculo numérico, reduciría la precisión del valor final calculado. A nivel de hardware, se pueden utilizar módulos especializados, por ejemplo aceleradores para programas aproximados utilizando redes neuronales.

El graduado de la carrera Ingeniería en Computadores Juan Carlos Cruz, realizó un trabajo sobre la computación aproximada, donde él se dio la tarea de caracterizar y calendarizar programas tolerantes a errores en una plataforma multi-acelerador. Parte del actual proyecto busca partir de los resultados generados por Cruz, de forma que se pueda utilizar el conocimiento generado sobre secciones ya aproximadas, para poder desarrollar el algoritmo que seleccionará cuál de todas ellas es la mejor según las especificaciones de un usuario.

Referencias

- [1] Qiang Xu, Todd Mytkowicz, and Nam Sung Kim. Approximate computing: A survey. *IEEE Design & Test*, 2018.

5. Anexos

5.1. Descripción de la empresa

El Instituto Tecnológico de Karlsruhe (KIT) surge en 2009 a partir de la unión de la Universidad de Karlsruhe, fundada en 1825 como Universidad Fridericiana, y el Centro de Investigación de Karlsruhe. Se ubica en Karlsruhe, en el estado de Baden-Württemberg, al suroeste de Alemania.

El Instituto de Ingeniería en Computadores en el KIT abarca grupos de trabajo que abarcan los diferentes niveles de abstracción de sistemas computacionales. En el Chair for Embedded Systems (CES) se investigan diversos aspectos relacionados con el diseño de sistemas embebidos, desde la confiabilidad de circuitos hasta el manejo de potencia en sistemas multinúcleos.

El proyecto como tal será desarrollado en el CES bajo la tutela del M.Sc. Jorge Alberto Castro Godínez, ingeniero en electrónica, investigador y estudiante de doctorado, quien es egresado del Tecnológico de Costa Rica y posee más de dos años y medio como investigador en el Instituto Tecnológico de Karlsruhe.