Instituto Tecnológico de Costa Rica

Área Académica de Ingeniería en Computadores
(Computer Engineering Academic Area)

Programa de Licenciatura en Ingenería en Computadores
(Licentiate Degree Program in Computer Engineering)

Diseño asistido de aplicaciones aproximadas para sistemas computacionales personalizables

(Assisted design of approximate applications for customized computational systems)

Informe del Anteproyecto para el Trabajo Final de Graduación
(Report of Pre-project for a Graduation Work in fulfillment of the requirements for the degree of Licentiate in Computer Engineering)

Daniel Esteban Moya Sánchez

Cartago, mayo, 2018

Índice

1. Palabras Clave

Arquitectura heterogénea, Multi-aceleradores, Multi-núcleo, Caracterización, Calendarización, Computación aproximada, Calidad

2. Introducción

Los sistemas de Tecnologías de Información (TI) buscan dar una mejor calidad de vida a las personas. En esta tarea, estos sistemas han tenido que enfrentar ciertos problemas entre los que se puede mencionar el costo en área, potencia y tiempo de ejecución, las cuales son variables que restringuen 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 solo se enfoca en contar con más desempeño si no en tener un manejo de recursos apropiado. No obstante, algunos desafíos en este campo están dados por limitaciones físicas, por ejemplo:

- las características eléctricas de los transistores CMOS, las cuales restringen el consumo de energía en sistemas embebidos y lo cual es un aspecto que deben considerar los diseñadores de componentes para propósito específico en procesadores;
- 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 es necesario identificar aplicaciones tolerantes a errores y determinar, 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.

Este documento busca explicar los detalles relacionados a la propuesta de un anteproyecto para el diseño asistido de aplicaciones aproximadas en sistemas computacionales personalizables, considerando plataformas de hardware multi-núcleo y multi-acelerador. Con la realización del proyecto se espera constribuir a la investigación en el campo de la computación aproximada, especialmente en el Instituto Tecnológico de Karlsruhe en Alemania, así como en el de área de ingeniería en computadores en general.

En la siguiente sección se presenta el contexto y los antecedentes del proyecto, que incluye una descripción de la institución donde se realizará el trabajo y el área de negocio específica de este. Seguidamente, se realiza la justificación del problema, la especificación de los objetivos, la mención de los beneficios y beneficiarios con los resultados del proyecto, los supuestos

y limitaciones sobre los que parte el proyecto y un análisis de riesgos del mismo. Finalmente, se detalla la metodología que se seguirá en el proyecto, la cual incluye la tipificación del trabajo, descripción de las tareas que se realizarán, las herramientas que se utilizarán, los entregables que se fijan para cumplir con los objetivos propuestos, las estrategias de verificación y validación que se seguirán y, por último, el cronograma de trabajo propuesto.

3. Contexto y Antecedentes

3.1. Descripción de la Institución

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-Wüttemberg, al suroeste de Alemania.

El KIT es una de las universidades técnicas más prestigiosas de Alemania, la cual se especializa en ciencias de la ingeniería. Según [1] para el 2017 contó con 25.495 estudiantes y 9.297 empleados. De acuerdo con [2] el KIT está divido en cinco divisiones:

- División I: Biología, Químnica e Ingeniería de Procesos.
- División II: Informática, Economía y Sociedad.
- División III: Ingenierías Mecánica y Eléctrica.
- División IV: Ambiente Natural y Construido.
- División V: Física y Matemática.

Las divisiones trabajan en aspectos de investigación, enseñanza e innovación. Los programas de investigación se organizan en programas Helmholtz, donde se le da apoyo a las investigaciones multidisciplinarias. Los departamentos en el KIT son los responsables de la educación universitaria. En la figura 1 se resume la organización en el campo de ciencia que posee el KIT.

El Instituto de Ingeniería en Computadores del KIT incluye 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 presente proyecto será desarrollado en el CES bajo la dirección 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.

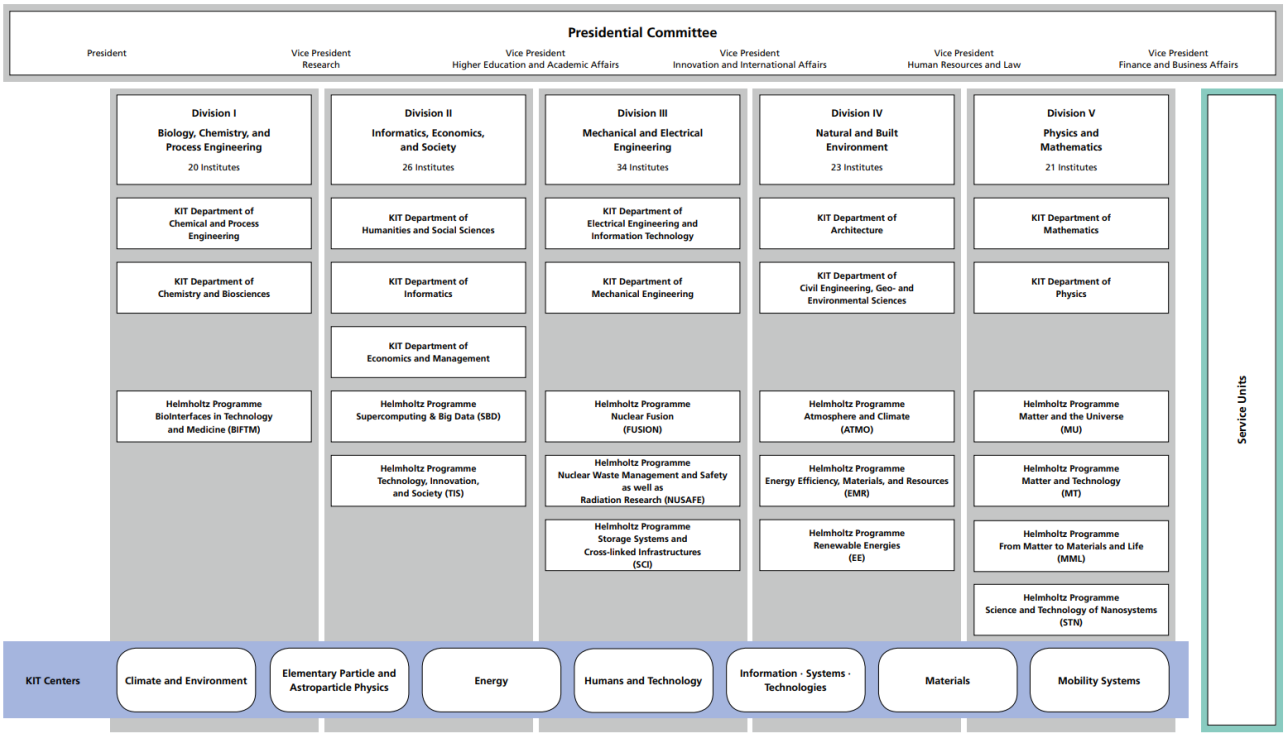

Figura 1: Organización científica en el KIT. Tomado de [2].

3.2. Área de Negocio

El área de conocimiento principal del proyecto es computación aproximada (explicada brevemente en la sección de Contexto del Problema), la cual incluye aspectos tanto de software como de hardware; esta se encuentra dentro de las área de investigación del CES. Las áreas de conocimiento de ingeniería específica que se tratan son: conocimiento sobre aspectos de arquitectura y micro-arquitectura de procesadores, computación de alto rendimiento, sistemas operativos, desarrollo de compiladores y comprensión de posibles aproximaciones de algoritmos en código de bajo nivel (por ejemplo, en el lenguaje de programación C).

4. Descripción de la Propuesta

4.1. Justificación y Definición del Problema

4.1.1. Contexto del Problema

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 2 se muestra un esquema que puede ser aplicado a sistemas tolerantes a errores para incluir en estos la computación aproximada [**?**].

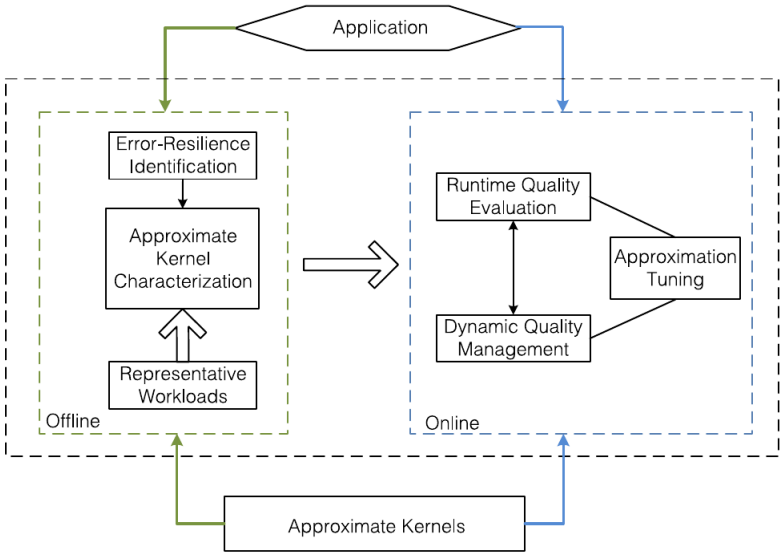

Figura 2: Un marco de trabajo para el uso de computación aproximada. Tomado de [**?**].

Los elementos clave de la Figura 2 son *kernels* aproximados, los cuales representan la implementación (técnicas) de las funciones aproximadas, estas puede ser realizadas a nivel de hardware o de software; la identificación de la 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 utilizan 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.

4.1.2. Especificación del Problema

Al contar con una aplicación que presenta 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 error máximo 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 posibles combinaciones posibles de versiones aproximadas, por lo que es importante utilizar un esquema de trabajo diferente.

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 3 muestra como ejemplo una aplicación genérica donde ambas situaciones pueden ocurrir.

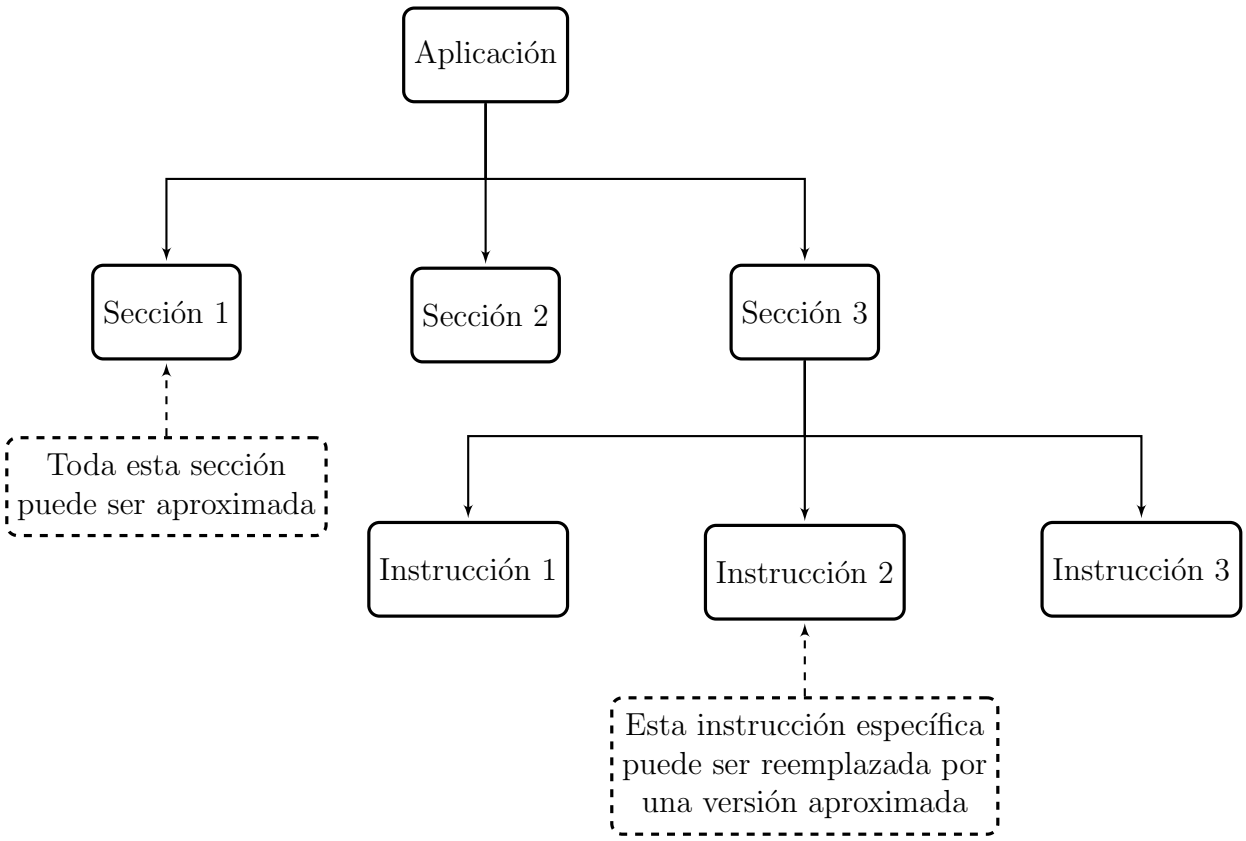

Figura 3: Una posible aplicación aproximada genérica analizada en este proyecto.

Como se muestra en la Figura 3, esta aplicación posee tres secciones, de las cuales la primera (por ejemplo, una etapa de preprocesamiento) puede ser completamente aproximada, la segunda no puede ser aproximada del todo (por ejemplo, una sección crítica de la aplica-

ción) y, finalmente, la tercera tiene tres instrucciones específicas, de las cuales únicamente la segunda posee una versión aproximada.

En el caso hipotético de la Figura 3 donde se tenga más de una versión aproximada para la sección 1 y para la instrucción 2 de la sección 3, es complicado (dicha complejidad aumenta con la cantidad de versiones aproximadas) determinar qué combinación de versiones aproximadas produce la mejor aplicación aproximada final, debido a que, por ejemplo, un cambio en la sección 1 puede impactar severamente las secciones 2 y 3; inclusive, puede que la versión aproximada de la instrucción 2 determine qué tipo de versiones aproximadas son las más convenientes (según las especificaciones del usuario) en la sección 1 para no impactar en gran medida la calidad de la aplicación.

4.1.3. Justificación de la Necesidad

Debido a la creciente necesidad por un consumo eficiente de recursos, ya sea energía, área o tiempo de ejecución, la computación aproximada, como alternativa de solución a este problema, se considera considerablemente importante. El diseño de un marco de trabajo para la realización de aplicaciones personalizadas mediante el uso de computación aproximada puede traer nuevos conocimientos a esta área de investigación y potenciar la creación de más aplicaciones, especializadas según los requerimientos específicos de los usuarios.

4.2. Enfoque de la solución

Se busca desarrollar una herramienta de software que pueda escoger entre diferentes versiones para una aplicación aproximada (cada versión dada por una combinación diferente de versiones específicas para cada sección aproximable), según el criterio de usuario que especifique cuáles recursos son críticos en la aplicación y cuál es la cantidad máxima de error permitido. La Figura 4 muestra una abstracción de la implementación de esta herramienta.

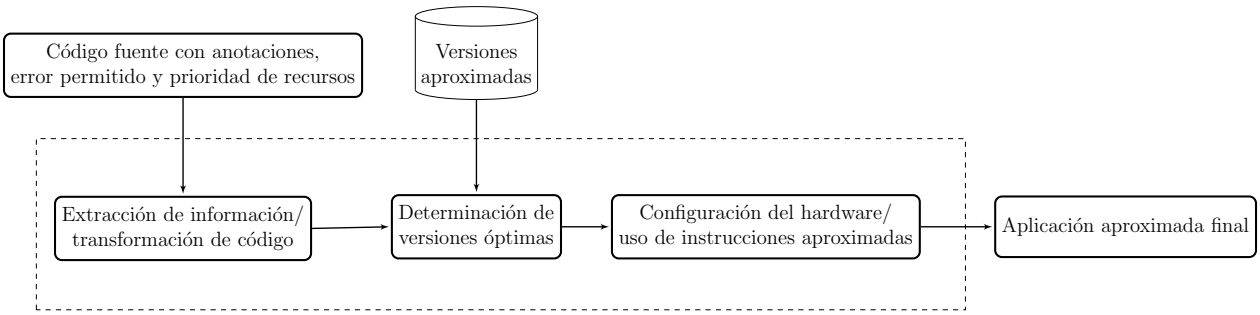

Figura 4: Esquema general de la solución propuesta.

Como se muestra en la Figura 4, 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 es transformado a una representación

intermedia. El algoritmo que se pretende desarrollar busca poder escoger cuáles versiones aproximadas de las posibles que existen se ajustan para cumplir a cabalidad con el error permitido y, de la mejor manera, con la priorización de recursos. Una vez identificadas las versiones que serán utilizadas, se procederá a utilizar el hardware específico que cuente con el soporte para las instrucciones aproximadas correspondientes, para finalmente entregar una aplicación final aproximada.

4.3. Especificación de Objetivos

4.3.1. 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.

4.3.2. 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.

4. Verificar que la aplicación final aproximada cumpla con el funcionamiento de la aplicación original y las restricciones dadas por el usuario y por el la aplicación

4.4. Beneficios y Beneficiarios con la Propuesta

1. Daniel Esteban Moya Sánchez: Se podrán fortalecer los conocimientos de arquitectura en computadores, compiladores, y aproximaciones a nivel de hardware y software, todo esto en un ambiente internacional, el cual aumentará el panorama cultural que se tiene, con los beneficios personales que esto conlleva. Se espera cunplir con los requerimientos del curso CE5600 Trabajo Final de Graduación, para así completar el plan de estudios 2100 de la carrera de Ingeniería en Computadores y poder graduarse para el año 2019.

2. Jorge Alberto Castro Godínez: Profesor tutor en el KIT, Alemania. Es investigador y estudiante de doctorado, a cargo de varios proyectos en el CES. Dentro de los beneficios está el conocimiento que se genere dado a el intercambio de ideas y resultados del

proyecto, que ampliarán las habilidades técnicas tanto a nivel de hardware como de software.

3. Chair for Embedded Systems (CES): Corresponde al lugar específico en el KIT donde se estará llevando a cabo el proyecto. El proyecto propuesto podrá formar parte de la investigación en el CES, lo cual beneficia parte de las áreas de investigación que trata. Se espera que los resultados del proyecto incentiven nuevos proyectos y aumenten así el conocimiento en el área de computación aproximada.

4. Instituto Tecnológico de Costa Rica: Como parte de los principios de investigación y extensión, para el TEC es suammente importante la presencia de estudiantes en el exterior. Los conocimientos que se generan a partir del proyecto propuesto podrán mejor la investigación en el TEC, e incentivar el área de computación aproximada.

4.5. Supuestos y Limitaciones

1. Limitación de tiempo: El proyecto se debe completar en un periodo menor a 5 meses, específicamente del 1 de Julio al 20 de Noviembre del 2018, dado que se necesita regresar al país para realizar la defensa presencial del proyecto en el TEC, además de realizar los trámites correspondientes para la graduación del 2019.

2. Disponibilidad de recursos: El proyecto se realizará utilizando herramientas de software libres. Para investigar se utilizará internet y el material disponible en el CES. Cualquier material físico del proyecto (como placa FPGA de desarrollo) será provisto por el CES.

3. Disponibilidad de versiones de aplicaciones aproximadas: El proyecto parte de la existencia de diferentes versiones aproximadas de algoritmos populares, que se utilizarán como base para el trabajo de personalización de aplicaciones, es decir, selección de específicas versiones aproximadas.

4.6. Análisis de Riesgos

Tabla 1: Posibles riesgos del proyecto.

ID	Categoría	Descripción	Probabilidad de ocurrencia	Impacto (horas)	Plan de Acción
1	Personal	Descripción	0.3	10	Afrontar
1	Herramientas	Carencia de materiales necesarios para el proyecto	0.3	16	Atacar

5. Propuesta Metodológica

5.1. Tipificación del Trabajo

El proyecto se clasifica como un trabajo de investigación aplicada, con alto porcentaje de experimentación.

5.2. Descripción del Proceso

La Figura 5 resume el proceso que se realizará durante el proyecto. Como se puede observar, 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 que permita 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.

Finalmente, se debe realizar la verificación de la aplicación aproximada final, de forma que se garantice un cumplimiento en el nivel de error y una optimización apropiada de los recursos. Para esto se realizarán simulaciones y pruebas unitarias en plataformas como ModelSim. Si por alguna razón se detectaran fallas, se revisará el algoritmo desarrollado con el fin de poder corregirlo.

La documentación del proyecto se trabajará a lo largo de todo el proceso de desarrollo, de forma que al final se genere un artículo científico y demás documentos propios de un trabajo final de graduación.

5.3. Herramientas

1. Lenguaje de programación: El software desarrollado utilizará C/C++ como lenguaje de programación.

2. Sistema operativo: Se utilizará Ubuntu 17.10.

3. Compiladores: Se utilizará la versión 7.2.0 para GCC/G++

4. Editor de texto: Para la documentación se utilizará LaTeX, con el ambiente de desarrollo Kile y compartido a través de Git. Para programar, se utilizará principalmente el editor

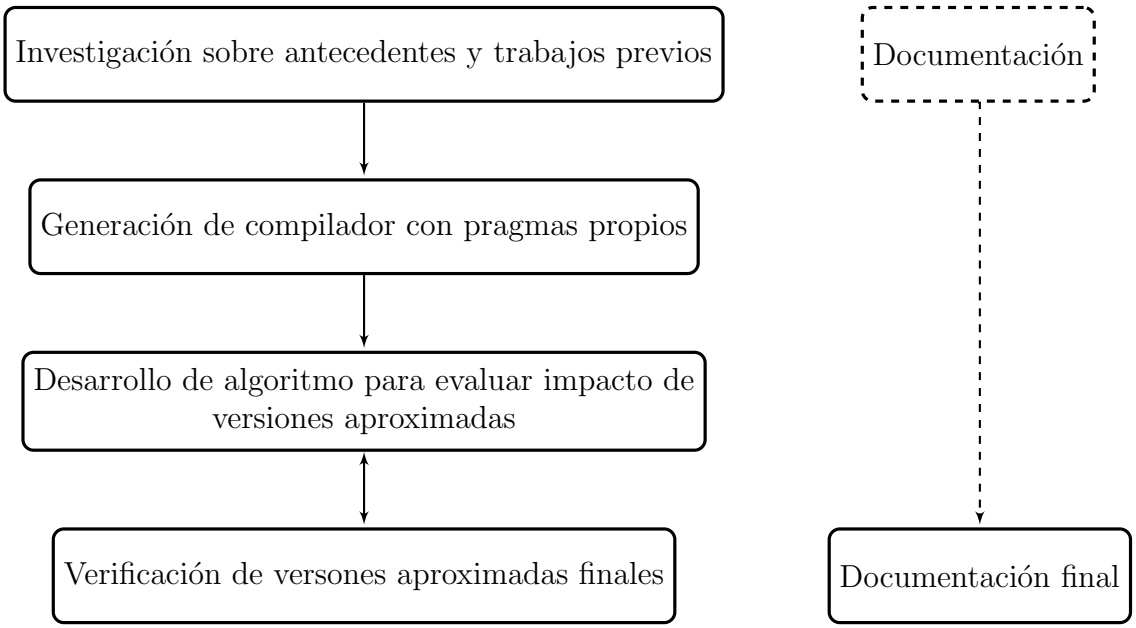

Figura 5: Proceso a realizar en este proyecto.

en terminal Vim.

5.4. Descripción de Entregables

La tabla 2 presenta la asociación entre entregables y objetivos del proyectos, según los apartados mencionados anteriormente.

Tabla 2: Entregables del proyecto

Objetivo	Entregable
Generar una representación del código de entrada que sea manipulable a partir de las anotaciones dadas por un usuario	Generación de compilador con pragmas propios
Evaluar las versiones aproximadas de las secciones de una aplicación para saber cuál es su impacto en la aplicación final	Documentación y desarrollo del algoritmo
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	Desarrollo de algoritmo para evaluar impacto de versiones aproximadas
Validar que la aplicación final aproximada cumpla con el funcionamiento de la aplicación original y las restricciones dadas por el usuario y por el la aplicación	Aplicación aproximada final y documentación

5.5. Estrategias de Verificación y Validación

5.6. Cronograma de Trabajo Propuesto

Referencias

[1] Karlsruhe Institute of Technology. Data and facts, 2018.

[2] Karlsruhe Institute of Technology. Tasks and structure, 2018.

Ingeniería en Computadores
Ficha de contactos del proyecto

Datos del estudiante

Nombre	Daniel Esteban Moya Sánchez
Correo electrónico	danielmscr1994@gmail.com
Teléfonos	(+506) 8325 9730

Datos del proyecto

Nombre	Diseño asistido de aplicaciones aproximadas para sistemas computacionales personalizables
Breve descripción	
Fecha de inicio	Lunes 2 de Julio del 2018

Datos de la empresa u organización

Nombre	Chair for Embedded Systems (CES), Instituto Tecnológico de Karlsruhe (KIT), Alemania
Nombre contacto	Jorge Alberto Castro Godínez, M.Sc.
Correo electrónico	jocastro@itcr.ac.cr
Teléfonos	+49 721 608 48780