Inteligencia Artificial (IS) 2017/18
Propuesta de trabajo

Reparación automática de circuitos reconfigurables

Agustín Riscos Núñez

1. Introducción y objetivo

Las plataformas de hardware reconfigurable ofrecen un mundo de posibilidades al combinarlas con sistemas evolutivos como los algoritmos genéticos.

Vamos a utilizar para este trabajo un circuito rectangular de *m* x *n* puertas lógicas (*m* capas con *n* puertas cada una), en el que cada puerta puede ser programada para comportarse de varias formas (OR, AND, NOT, NAND, XOR), y además las conexiones entre las puertas también pueden variar: cada puerta puede recibir uno o dos inputs provenientes de cualquier puerta de las dos capas anteriores, o también puede no tener ningún input asignado, lo que significaría estar desactivada.

El **objetivo principal** de esta propuesta es la implementación de un método de *auto-reparación* que permita al circuito detectar si su hardware ha sufrido algún daño físico y en ese caso activar un proceso evolutivo para reconfigurarse de forma que pueda seguir comportándose igual a nivel global. Para ello será necesario alcanzar los siguientes objetivos **específicos**:

1. Implementar la estructura de datos para el circuito. Elegir un circuito "solución" que será el que supondremos que resulta dañado y hay que restaurar.
2. Implementar un mecanismo que actúe como simulador de circuitos (es decir, que dado un vector de entrada con *n* bits calcule el vector de output del circuito), incluyendo una opción de marcar ciertas conexiones y/o puertas como "defectuosas".
3. Construir una colección de pares de vectores de *n* bits (entrada,salida) que nos servirá de referencia para representar el "comportamiento correcto" del circuito. Implementar un método de auto-diagnóstico que compruebe si hay algún fallo respecto a dicha colección de pares.
4. Implementar un algoritmo genético que dado un circuito dañado, busque "buenos" circuitos alternativos, cambiando la programación de las puertas del circuito y/o las conexiones entre ellas. El algoritmo genético debe trabajar sin conocer qué puertas son las dañadas o defectuosas.
5. Realizar varios experimentos suponiendo que se produce un daño en un subconjunto de puertas y/o de conexiones, y analizando el rendimiento del sistema de auto-reparación en cada caso.
6. Documentar el trabajo en un fichero pdf con formato de artículo científico.
7. Realizar una corta presentación de los resultados obtenidos en la defensa del trabajo.

Para que el trabajo pueda ser evaluado, se deben satisfacer TODOS los objetivos específicos al completo: el trabajo debe ser original, estar correctamente implementado y funcionar perfectamente, los experimentos deben haber sido llevado a cabo y analizados razonadamente, el documento pdf debe ser completo y contener entre 8 y 10 páginas, y se debe realizar la defensa con una presentación de los resultados obtenidos.

2. Descripción del trabajo

A continuación se introduce la metodología a seguir para el correcto desarrollo del trabajo.

2.1. Entrada y Salida del circuito

El trabajo debería implementarse de forma parametrizada, de forma que se pueda elegir libremente cualquier valor para las variables que marcan el tamaño del circuito, *m* y *n*. Sin embargo, es posible realizar un desarrollo ad-hoc fijando valores (aunque supondrá una menor calificación).

Se considerará siempre que el circuito tiene *n* bits de entrada y *n* bits de salida, independientemente de que alguna de las entradas no esté conectada con ninguna puerta. En caso de que alguna puerta del circuito tenga una entrada sin conexión asignada, o con una conexión que ha resultado dañada, se considerará que en la posición correspondiente se recibe un 0.

2.2. Procedimiento evolutivo de reparación

No hay ninguna restricción respecto al tipo de algoritmo genético que debe usarse. Está permitido usar librerías existentes de Python, siempre citando la referencia correspondiente.

Dependiendo del daño producido en el circuito, podría ocurrir que sea imposible repararlo, así que no es recomendable que el test de parada sea lograr cero errores respecto a la colección de pares de referencia.

Sin embargo, sí es muy importante saber si se ha logrado una auto-reparación completa o no, así que al aplicar el algoritmo genético, deberá mostrarse un mensaje indicando el rendimiento (o al menos que se diga si se detectan errores o no) para el circuito que se devuelva como salida.

2.3. Experimentación

No se indica un número mínimo de repetición de cada experimento, pero teniendo presente el componente aleatorio de los algoritmos genéticos, se espera que se realicen una cantidad suficientemente representativa. No se pide que se incorpore a la entrega literalmente la salida de cada una de las ejecuciones (aunque se pueden guardar si se desea ficheros de texto a modo de anexo). Lo importante es que en la fase de experimentación habrá que procesar los datos obtenidos y generar las correspondientes tablas con valores medios y otras estadísticas relevantes. El análisis de los resultados que se pide que se haga en la sección de conclusiones se deberá referir a dichas tablas y estadísticas, nunca a los resultados de una sola ejecución.

2.4. Documentación

En el fichero *plantilla-trabajo.doc* se muestra una sugerencia de estructura y formato de estilo artículo científico correspondiente a la documentación del trabajo. Este formato es el del *IEEE conference proceedings*, cuyo sitio web *guía para autores* [1] ofrece información más detallada y plantillas para Word y Latex.

El artículo deberá tener una extensión **entre 8 y 10 páginas**, y la estructura general del documento debe ser como sigue: en primer lugar realizar una **introducción** al trabajo explicando el objetivo fundamental, incluyendo un breve repaso de antecedentes en relación con la temática del trabajo y con los métodos empleados (mencionar referencias bibliográficas), a continuación describir la **estructura** del trabajo, las **decisiones de diseño** que se hayan tomado a lo largo de la elaboración del mismo, y la **metodología** seguida al implementarlo (nunca poner código, pero sí pseudocódigo), y seguidamente detallar los **experimentos** llevados a cabo, **analizando los resultados** obtenidos. Por último, el documento debe incluir una sección de **conclusiones**, y una **bibliografía** donde aparezcan no sólo las referencias citadas en la sección de introducción, sino cualquier documento consultado durante la realización del trabajo (incluidas las referencias web a páginas o repositorios).

2.5. Mejoras

Aunque no sea obligatorio, sí se tendrá en cuenta en la calificación la incorporación de un interfaz o menú amigable que facilite la experimentación con la herramienta desarrollada. También podrán recibir puntuación adicional otras mejoras o añadidos que se incorporen al trabajo más allá de los requisitos mínimos que se mencionan en la lista de objetivos específicos.

Por ejemplo: (a) incluir un visor que muestre gráficamente el circuito; (b) incluir más tipos de puertas lógicas; (c) incluir varios tipos de operadores genéticos y/o varios tipos de algoritmos y realizar experimentos comparativos; etc.

2.6. Presentación y defensa

El día de la defensa se deberá realizar una pequeña presentación de 5 minutos por alumno. En esta presentación seguirá a grandes rasgos la misma estructura que el documento, pero se hará especial mención a los resultados obtenidos y al análisis crítico de los mismos. Se podrá usar un portátil (personal del alumno), diapositivas y/o pizarra. En los siguientes 10 minutos de la defensa, el profesor procederá a realizar preguntas sobre el trabajo, que podrán ser tanto del documento como del código fuente.

3. Criterios de evaluación

Para que el trabajo pueda ser evaluado, se deberá satisfacer los objetivos concretos descritos en el apartado 1 (todos y cada uno de ellos, si no, el trabajo obtendrá una nota de suspenso). Uno de los alumnos del equipo deberá subir a través del formulario disponible en la página de la asignatura un fichero comprimido .zip, que contenga:

- Una carpeta con el código fuente. Dentro de dicha carpeta tiene que haber un fichero README.txt, que resuma la estructura del código fuente, e indique cómo usar la interfaz (si se ha implementado), o al menos cómo hacer pruebas con los métodos implementados para los objetivos 2, 3, y 4, incluyendo ejemplos de uso. Asimismo se deberá de indicar cómo reproducir los experimentos realizados (objetivo 5) y cómo se diseñarían y ejecutarían nuevos experimentos. Es importante la coherencia de este fichero con la defensa.
- El documento – artículo en formato PDF. Deberá tener una extensión mínima de 8 páginas, y máxima de 10. Deberá incluir toda la bibliografía consultada (libros, artículos, technical reports, páginas web, códigos fuente, diapositivas, etc.) en el apartado de referencias, y mencionarlas a lo largo del documento.

Para la evaluación se tendrá en cuenta el siguiente criterio de valoración, considerando una nota máxima de 3 en total para el trabajo:

- El código fuente (0,8 puntos): se valorará la claridad y buen estilo de programación, corrección y eficiencia de la implementación, y calidad de los comentarios. La claridad del fichero README.txt también se valorará. En ningún caso se evaluará un trabajo con código copiado directamente de internet o de otros compañeros.
- El documento – artículo científico (1,5 puntos):
  - Se valorará el estilo general del documento (por ejemplo, el uso de la plantilla sugerida).
  - Se valorará la cantidad y calidad de los experimentos, los resultados alcanzados y el análisis crítico que se haga de los mismos.
  - Se valorará la claridad de las explicaciones, el razonamiento de las decisiones, el análisis y presentación de resultados, y el uso del lenguaje. Se podrá realizar en castellano o inglés. Igualmente, no se evaluará el trabajo si se detecta cualquier copia del contenido.
- La presentación y defensa (0,7 puntos): se valorará la claridad de la presentación y la buena explicación de los contenidos del trabajo, así como, especialmente, las respuestas a las preguntas realizadas por el profesor.
- Mejoras (hasta 0,5 puntos): se valorará en su caso la incorporación de características adicionales más allá de los requisitos obligatorios, aunque sin poder superar la nota máxima de 3 puntos.

Cualquier **plagio, compartición de código** o uso de cualquier material que no sea original y del que no se cite convenientemente la fuente; que se detecte, significará automáticamente la **calificación de cero** en la asignatura para **todos los alumnos involucrados**. Por tanto, a estos alumnos **no se les conserva**, ni para la actual ni para futuras convocatorias, **ninguna nota** que hubiesen obtenido hasta el momento. Todo ello sin perjuicio de las correspondientes **medidas disciplinarias** que se pudieran tomar.

4. Referencias

[1] Plantilla IEEE. https://www.ieee.org/conferences_events/conferences/publishing/templates.html