Práctica Final de JFLAP 2018-2019

Especificaciones

El objetivo de esta práctica es diseñar e implementar una Maquina de Turing que calcule las soluciones de una familia concreta de polinomios de orden 2. Para la simulación de la máquina se empleará la herramienta JFLAP.

Los polinomios que vamos a *resolver* se representan mediante la ecuación:

$$x^2 - Ax - 1 = 0$$

donde *A* es una constante que puede tomar algún valor entero. Las soluciones que se obtienen para diversos valores de *A* forman parte de los llamados números metálicos. El valor que se obtiene para *A*=1 es Φ=1.618033989…, un número irracional muy conocido en numerosas disciplinas y desde la antigüedad como razón áurea (o número de oro) debido a numerosas propiedades. Para otros valores de *A* se obtienen otros números, algo menos conocidos: *A*=2, 2,4142135623… (número de plata), *A*=3, 3,302775638... (número de bronce).

Obtener la solución exacta de un polinomio cuadrático usualmente implica realizar cálculos con raíces cuadradas, lo cual puede suponer una complejidad excesiva para esta práctica. Por ello proponemos recurrir a un método numérico que obtiene una aproximación en sucesivas iteraciones. Transformando el polinomio en $\boldsymbol{x = A + 1/x}$ obtenemos el precursor de un método iterativo estable que converge hacia la raíz buscada:

$$\boldsymbol{x_{i+1} = A + 1/x_i \text{ donde } x_0 \geq 0 \text{ (por ejemplo, } x_0 = A), i \geq 0}$$

Esto quiere decir que, para un número de iteraciones elevado, utilizando este polinomio, a partir de un valor x_0 introducido, y un valor *A*, obtendremos aproximaciones precisas de los números metálicos.

Para la práctica tendremos en cuenta los siguientes requisitos y propiedades:

- La serie de valores x_i converge en el infinito, por lo que limitaremos el cálculo al valor n-ésimo x_n, obtenido de forma iterativa partiendo del valor inicial x_0. x_n será el último valor que calcule la máquina antes de parar.
- Todos los números se representarán en binario.
- Tendremos como parámetros los siguientes valores:
    - Número de iteraciones a realizar (un número entero no negativo).
    - Valor de la constante A, que será siempre un entero no negativo.
    - Valor inicial de la serie (x_0). Un valor no negativo cuyos dígitos decimales indicaran con qué precisión deberá operar el algoritmo (precisión decimal deseada)

- Una propiedad que puede ser útil es que $x_i < A + 1,\ \forall i > 0,\ A > 0$
- Y otra propiedad que puede ser útil es que $1/x_i < 1,\ \forall i > 0,\ A > 0$

- La estructura inicial de la cinta debe ser:

#	1	1	0	$	1	1	$	1	0	.	0	1	1	1	0	#
	N				A			Valor de X_0 con precisión implícita								

- Usamos el símbolo # en este enunciado para representar las celdas en blanco.
- El símbolo **$** sirve para separar un parámetro de otro.
- Se emplea el punto (.) para indicar la posición de la coma en X_0. El resultado final deberá tener el mismo número de posiciones decimales con valor significativo.
- Cualquiera de los valores podrá ser mayor o igual a cero, no se contemplan números negativos.
- Vamos a considerar que cualquier número puede llevar ceros no significativos por la izquierda. Es decir, podríamos emplear una cinta inicial como:

#	0	1	1	$	0	1	$	0	1	.	0	1	#

- En el instante de parada de la máquina, la cinta debe contener únicamente el último valor de la serie, sin ceros no significativos por la izquierda. Por ejemplo:

#	1	0	.	1	0	1	0	1	#

- No se permite el uso del desplazamiento nulo del cabezal salvo para las transiciones **a un estado final.**
- La Máquina de Turing deberá funcionar en modo transducción.
- Se deberá utilizar la función BuildingBlocks para aportar modularidad.

Tareas, documentación y entrega

Grupos

Esta práctica se realizará en grupos de como máximo 2 personas. **Las entregas las debe realizar sólo uno** de los alumnos de cada grupo.

Tareas a realizar

1. Diseño de la Máquina de Turing a partir de las especificaciones.
2. Elaborar una memoria de describa de forma **modular** y detallada la máquina diseñada.
3. Describir las pruebas realizadas para validar el correcto funcionamiento de la máquina.

Evaluación de la práctica

En la nota de la práctica influirán diversos aspectos que debéis tener en cuenta.

- La memoria debe estar completa, detallada y bien redactada. Es importante que expliquéis bien el trabajo que habéis realizado.
- Las imágenes incluidas deben ser legibles, es decir, los símbolos deben tener un tamaño razonable, y los grafos deben ser fáciles de entender.
- Los ejemplos de prueba que hayáis empleado para validar vuestra máquina deben ser exhaustivos y deben estar escogidos de forma meditada.
- La máquina debe funcionar correctamente en todo tipo de casos.
- En la corrección se emplean algunos procesos automáticos que son muy sensibles a los casos en los que no se cumplen las especificaciones, lo cual incidirá negativamente en la nota.
- Se valorará la originalidad de los diseños que presentéis, cosa que debéis documentar expresamente. Por ejemplo, si consideráis que vuestra máquina está especialmente optimizada en cuanto al número de estados, o del tiempo que requiere para realizar el procesamiento, o que emplea un algoritmo alternativo o mejor (consultad con los profesores).

Normas de entrega

En el entregador de Aula Global, en los grupos *reducidos* deberá entregarse un fichero ZIP con el nombre:

- **Entrega parcial: TALF_PARCIAL_NIA1_NIA2.zip**
- **Entrega final: TALF_NIA1_NIA2.zip**

donde **NIA1** y **NIA2** serán los números identificativos de la universidad de los alumnos que hacen la entrega.

Este ZIP contendrá los siguientes ficheros en una única carpeta:

- Los ficheros jff con las máquinas diseñadas. La máquina definitiva deberá llamarse **mt-final.jff**.
- Un fichero de texto con las palabras de prueba escogidas, llamado **pruebas.txt**.
- No utilicéis espacios, tildes u otros símbolos no alfabéticos en los nombres de los ficheros.

La memoria final se entregará en otro entregador dedicado para la misma, tras ser evaluado por la herramienta Turnitin. Se llamará **TALF_MEMORIA_NIA1_NIA2.pdf**. La primera página del documento deberá identificar correctamente a los miembros del grupo.

Entregas a realizar

1. Entrega parcial (**4 de Diciembre de 2018 hasta las 23:55h**): entrega **obligatoria** de lo realizado hasta la fecha. Se revisará durante la clase de dicho día el progreso de cada grupo.
2. Entrega Final: **10 de Diciembre de 2018 hasta las 23:55h.**
3. Entrega de Recuperación. Se realizará una corrección previa para detectar prácticas que no funcionan o que no se pueden corregir. A partir de la fecha de notificación dispondréis de dos días para solventar posibles problemas. Esta entrega conllevará una penalización de tres puntos en la nota de la práctica (nota máxima = 7).

Sugerencias

Para diseñar la Máquina de Turing completa conviene seguir algunas pautas. El punto de mayor dificultad puede ser implementar la división, en este caso, calcular la inversa de un número. Otras dificultades pueden radicar en dónde situar y cómo tratar con la coma decimal, cómo situar los operandos, etc.

Existen varios algoritmos para aplicar la división. El más habitual, que también es bastante eficiente, se basa en sucesivas restas combinadas con desplazamientos (multiplicar por dos el dividendo). Si se escoge este algoritmo, será necesario implementar y dominar previamente la suma, la resta y la comparación de dos números binarios. Estos a su vez se basan en calcular el incremento y decremento (sucesor y antecesor) de un número.

Es muy recomendable estudiar con detenimiento cómo se combinan estas operaciones, antes de pasar a su implementación. El diseño de la máquina de Turing de división debería ser lo más modular posible. Para ello se deberá usar las opciones de BuildingBlocks en JFLAP. Y a su vez, cada uno de los componentes (incremento, decremento, suma, resta, comparación, …) debería ser lo más compacto posible. Tened en cuenta que cualquier sub-máquina que requiera de muchos estados, tendrá mayor posibilidad de contener algún error difícil de detectar debido a una mayor cantidad de circunstancias que puedan pasar desapercibidas a la hora de diseñarla.

El uso de sub-máquinas que resuelven problemas concretos y bien definidos puede facilitar la construcción global, y permite hacer depuraciones locales. Pero requiere definir bien el contexto en el que va a operar cada una de ellas.

A la hora de escoger nombres para cada una de las sub-máquinas (sus ficheros), procurad evitar nombres largos, con espacios en blanco u otros símbolos no alfanuméricos, dado que JFLAP puede dar errores en estos casos.

El uso de BuildingBlocks con máquinas de más de dos niveles no está bien verificado por ahora.

La división 1/x es el punto de mayor complicación.

Se recomienda empezar por resolver problemas más sencillos, la suma de dos números, la resta…

Y luego por estudiar cómo se opera una división por pasos, en binario, con decimales…

La propiedad $x_i < A + 1$ tiene interés porque implica lo siguiente: $1/x_{i-1}$ será menor que 1, $A + 1/x_{i-1}$ será menor que A+1, por lo que al sumar A a $1/x_{i-1}$ nunca hará falta añadir más bits significativos de los que ya tiene A.

La división se puede implementar basándose en diversos algoritmos:

- Pasando a unario (extremadamente ineficiente).
- Por restas sucesivas (muy ineficiente al tratar decimales de precisión).
- Por restas y desplazamientos, el habitual.
- Otros, más eficientes, que no se suelen dar.

La coma decimal se puede tratar de diferentes formas:

- Explícita, empleando el símbolo de coma.
- Figurada marcando el dígito menos significativo, por ejemplo.
- Representando su posición con un valor numérico adicional.
- Implícita, si se trata de coma fija, siempre estará en la misma posición.