Definición y evaluación de un Autómata Finito Determinista Bidireccional con memoria Lifo/Fifo

Giró Juan, Vázquez Juan, Meloni Brenda y Constable Leticia
Universidad Tecnológica Nacional, Facultad Regional Córdoba
Departamento de Ingeniería en Sistemas de Información

Resumen
Con el objetivo de explorar el desempeño de máquinas abstractas, de capacidad inferior a la Máquina de Turing y mayor a la del Autómata Finito, se abordaron sucesivas tareas: reconocer y estudiar las principales máquinas disponibles, proponer una máquina específica a ser considerada, definir e implementar un simulador que posibilite el estudio de su comportamiento, seleccionar casos de estudio y analizar sus resultados. En el documento que se presenta se centra la atención en la definición del nuevo autómata y en los resultados obtenidos con un caso de estudio. Las pruebas se orientaron a evaluar la complejidad temporal y la sensibilidad de este indicador a variantes en las cadenas de datos. También inspiraron otras máquinas a ser estudiadas y confirmaron el enorme valor técnico y pedagógico de los procesos de simulación.

Palabras Clave
Máquinas abstractas, complejidad, simulación.

INTRODUCCIÓN

El campo de la *Ciencia de la Computación* se estableció y consolidó a partir de las máquinas abstractas, algunas de las cuales anticiparon conceptualmente muchos de los progresos de la computación práctica que hoy son realidades, y hay otras que esperan hacerlo en algún futuro, acompañando los avances de la tecnología.

Al tratarlas, resulta conveniente comenzar por la Máquina de Turing, que representa el límite de lo computable. Luego, solo restringiendo con diferentes recursos el acceso a la memoria, que está representada como una cinta infinita de acceso secuencial, se obtiene una enorme y variada diversidad de máquinas de capacidades progresivamente inferiores, algunas muy poco conocidas.

Cabria entonces preguntarse el sentido que tienen tales restricciones. La respuesta es simple: con máquinas bien pensadas, de menor capacidad y más específicas, se ganaría en eficiencia, lo que significa menor complejidad operativa. Es decir, reducción del tiempo de proceso y/o espacio demandado para resolver un mismo problema, lo que en muchos casos hace posible su viabilidad práctica. Esto explica el esfuerzo que realiza la *Ciencia de la Computación* en este amplio campo de investigación.

En este contexto se inscribe este trabajo, que está organizado de la siguiente manera: se comienza por seleccionar, proponer y describir una máquina abstracta "no convencional", se selecciona un caso testigo y se muestra el desempeño de la nueva máquina confrontándola con una Máquina de Turing. Finalmente se discuten los resultados obtenidos y se presentan las conclusiones.

DESCRIPCIÓN DEL AUTÓMATA

En la búsqueda de una máquina interesante y novedosa, que justifique su estudio, se revisaron las diversas opciones propuestas y descriptas en la literatura. En la selección se dio preferencia a autómatas sin capacidad para alterar la información de la cinta de entrada, orientando el trabajo hacia máquinas reconocedoras de lenguajes. Luego, se consideró la conveniencia de disponer de memoria auxiliar con un tipo de acceso específico y control del movimiento del cabezal de entrada en dos sentidos, todo lo cual brinda opciones para trascender el reconocimiento de las gramáticas tipo 3.

En resumen, la máquina buscada se obtiene tomando como base un Autómata Finito Determinista Bidireccional (*AFDB*), al que se lo dota de una memoria dual de tipo Lifo/Fifo. La primera propuesta tiene numerosos antecedentes, al igual que las máquinas con memorias Lifo o Fifo, no siendo habitual la coexistencia de ambos

tipos de accesos a la memoria. A partir de sus componentes se adopta la designación *AFDB-LF* (Autómata Finito Determinista Bidireccional Lifo/Fifo) para el autómata.

Como resultado de la búsqueda de antecedentes se citan a continuación algunos de los principales y más reconocidos: el *AFDB* fue estudiado por Rabin y Scott [1] y Sheperdson [2], el autómata con pila (memoria Lifo) por Oettinger [3] y Schutzenberger [4], finalmente el autómata con cola (memoria Fifo) fue estudiado por Cherubini [5]. Caben aquí agregar los aportes de Petersen [6] y de Rosenberg [7], que demostraron que un autómata con memoria Fifo podía emular una Máquina de Turing, al igual que lo puede hacer el autómata con dos pilas de Koslowski [8]. También deben citarse los trabajos de Aho [9], Hopcroft y Ullman [10], Gluck [11] y Alonso [12] referidos al autómata con pila bidireccional y los de Bhattacharjee [13] y Asha latha [14] que estudiaron un autómata con memoria dual Lifo/Fifo. Este último es denominado en la literatura *Deque Automata*.

A partir de un análisis exhaustivo de estas propuestas y de otras similares, se definieron las características del *AFDB-LF* y se adoptó el criterio de utilizar el formalismo, que es clásico en la literatura, para definir este tipo de máquinas. El autómata propuesto queda establecido como:

$$AFDB\text{-}LF = (\Sigma_E, \Sigma_C, \Gamma, Q, q_o, \#, A, f)$$

donde:

Σ_E : Alfabeto de entrada

Σ_C : Alfabeto de cinta, $\Sigma_C = \Sigma_E \cup \{ >, < \}$

Γ : Alfabeto de la pila/cola

Q : Conjunto de estados

q_o : Estado inicial, $q_o \in Q$

\# : Referencia de pila / cola vacía, $\# \in \Gamma$

A : Estados de aceptación, $A \subset Q$

f : Función de transición,
$\Omega \times \Sigma_C \times \Gamma \rightarrow Q \times \Gamma^* \times m$

Siendo:

Ω : Conjunto de estados con identificadores del tipo de acceso que es operado por cada uno (Lifo/pila o Fifo/cola):
$\Omega = \{ \omega / \omega = \mu q, \mu \in \{+, -\}, q \in Q \}$
"+" = acceso Lifo,
"-" = acceso Fifo

m : Sentidos del próximo movimiento del cabezal sobre la cinta de entrada, m = {I, N, P, D}, Izquierda, Neutro, Parada y Derecha

Representando a la descripción instantánea del autómata por la cuaterna (q, <α>, *k*, δ) que define: *i*) el estado actual, *ii*) contenido de la cinta, *iii*) posición del cabezal y *iv*) contenido de la pila, el autómata puede realizar como ejemplo los siguientes movimientos:

a) **acceso LIFO**, $f(+q, a, c) = (s, ca, D)$

$(+q, >a\beta<, 1, \#bc) \vdash (s, >a\beta<, 2, \#bca)$

El símbolo de tope de pila "c" es leído y repuesto. A continuación se apila el símbolo "a" leído de la cinta de entrada.

b) **acceso FIFO**, $f(-q, a, b) = (s, ba, D)$

$(-q, >a\beta<, 1, \#bc) \vdash (s, >a\beta<, 2, \#cba)$

El símbolo de fondo de pila "b" es leído y apilado. Luego, a continuación se apila el símbolo "a" leído de la cinta de entrada.

En los citados ejemplos de accesos, son: $a \in \Sigma_E$, $\beta \in \Sigma_E^*$, $s \in Q$, $D \in m$, c, ca, ba, #bc, #bca, #cba $\in \Gamma^*$ y $+q, -q \in \Omega$.

Aquí debe observarse que la condición de acceso a la memoria está asociada a cada estado, es decir que hay estados desde los cuales el acceso es Lifo y desde otros es Fifo, lo que lo distingue de las referencias [13] y [14] citadas. En caso de admitirse ambos accesos en un mismo estado, quedaría planteada una nueva forma de no determinismo que no es aquí objeto de estudio.

Nótese que los dos tipos de acceso disponibles permiten leer desde el tope (Lifo) o desde el fondo (Fifo). En este último caso el símbolo leído es el penúltimo, salvo que la cola esté vacía, en cuyo caso se lee el último (#). Es decir, al estar vacía la pila o cola la máquina recibe como entrada la marca específica que identifica esta condición (#).

Por su parte las grabaciones (Γ^*) tienen siempre como destino el tope de la pila, pudiendo tratarse de una cadena vacía (λ).

Como consecuencia de lo expuesto, es

importante observar que, con un solo acceso Fifo y salvo que esté vacía, la memoria de cola nunca puede restablecerse a su condición anterior a un movimiento, ya que se descarga por abajo y carga por arriba. Por el contrario, esto si es posible con un acceso Lifo, que descarga y carga por el mismo extremo. Las ventajas y limitaciones de esta diferencia es uno de los temas a comprobar a través del simulador.

Como ocurre con los autómatas con pila convencionales, para poder representar la función de transición con una tabla debe adoptarse el criterio de que las columnas corresponden a los elementos de Σ_C y las filas a pares Ω x Γ (estados con sus formas de acceso y símbolos leídos de la memoria Lifo/Fifo). Esta propuesta de mostrar funciones tridimensionales en tablas es atribuida a Alfonseca [15] y posibilita una representación muy compacta y conveniente. Asimismo, como ocurre con todas las máquinas abstractas, los *AFDB-LF* también pueden ser representados con un grafo.

CASO DE ESTUDIO

El objetivo planteado es comparar la complejidad temporal obtenida a partir de la solución de un caso de estudio; utilizando una Máquina de Turing Determinista (*MTD*) y un Autómata Finito Determinista Bidireccional Lifo/Fifo (*AFDB-LF*). Para ello se utilizaron dos simuladores, el primero disponible en [16] y el otro desarrollado especialmente según las características de la nueva máquina.

Como caso de estudio se optó por un lenguaje que no es independiente de contexto, tal como: $L = \{\alpha / \alpha = \delta c \beta \delta \gamma, \delta \in \{a,b\}^+, \beta, \gamma \in \{a,b\}^*$ y $\Sigma_E = \{a,b,c\}\}$. Cada sentencia contiene un prefijo δ seguido de un separador "c", luego del cual se repite la cadena δ en un contexto $\beta\delta\gamma$ (Rytter, caso 49 [17]).

Tal como fue anticipado, el problema es resuelto mediante la *MTD* y el *AFDB-LF*, y sus grafos son representados a continuación.

Como puede comprobarse en el grafo de la Figura 1, por cada símbolo del prefijo la *MTD* busca encontrar en la secuencia correcta el mismo símbolo en el sufijo. De no ser así, la máquina regresa su cabezal al punto de partida y repite el procedimiento, ignorando en cada reintento un símbolo adicional en el sufijo a partir del símbolo separador "c".

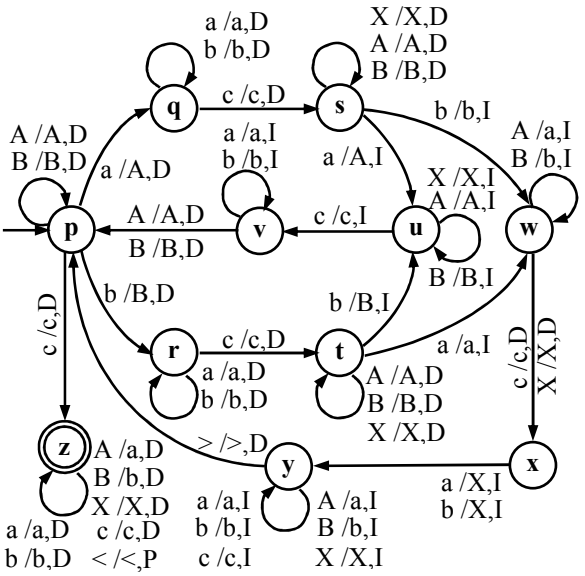

Figura 1: Grafo de la *MTD*

En la Figura 2 se representa el grafo del *AFDB-LF*, que prevé accesos Lifo a la memoria auxiliar desde los estados "p" y "u", mientras que el acceso es Fifo a la misma memoria desde los estados "q", "r" y "s".

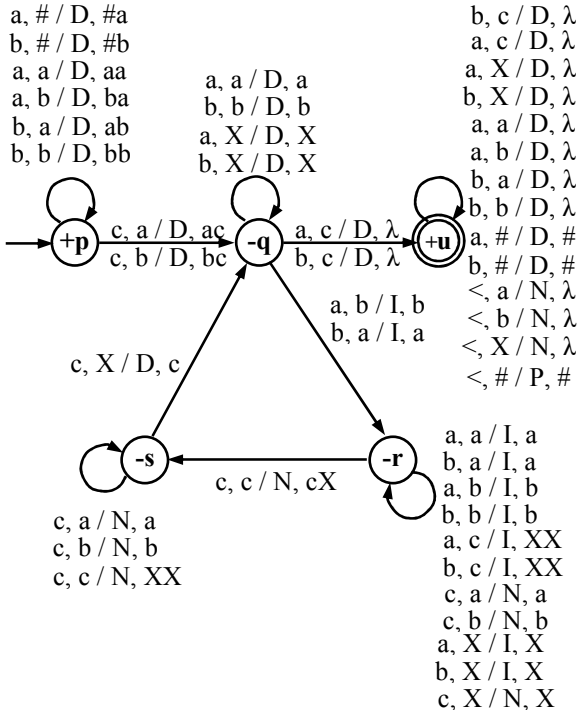

Figura 2: Grafo del *AFDB-LF*

De esta forma puede cargar una sola vez el prefijo y lo va haciendo rotar en la memoria (Fifo) a medida que recorre el sufijo. Cada vez que falla, vuelve al separador y sincroniza el contenido de la memoria de manera de saltear un símbolo adicional, idea equivalente a la utilizada por la *MTD*.

DISCUSIÓN DE RESULTADOS

La complejidad temporal es un indicador indirecto, cuyo valor representa tres factores: *i*) la complejidad intrínseca del propio problema, *ii*) la del procedimiento propuesto para resolverlo (en este caso la máquina adoptada) y *iii*) las particularidades de los propios datos. Y esta triple dependencia debe ser tenida en cuenta en este tipo de estudio.

Ante la presunción de que la complejidad temporal no solo depende del largo de la cadena $n = |\alpha|$, y que también influye la posición relativa de δ en el contexto βδγ, se adoptó un criterio común. Este es que $|\beta| = |\delta| = |\gamma|$, por lo que $\boldsymbol{n} = 4|\delta| + 1$, lo que equivale a decir que el prefijo buscado δ está en el centro del sufijo βδγ.

Además, para mostrar el impacto del ordenamiento de los datos se seleccionaron cuatro casos, a saber:

Caso 1: α_1 = **ababa**cbbbbb**ababa**aaaaa
Caso 2: α_2 = **baaaa**caaaaa**baaaa**bbbbb
Caso 3: α_3 = **abbbb**caaaaa**abbbb**bbbbb
Caso 4: α_4 = **aaaab**caaaaa**aaaab**bbbbb

Las cadenas son representadas con un tipo de caracteres de "ancho fijo" (courier) para facilitar su comparación y se ha subrayado el prefijo, que luego se repite en el sufijo.

Para obtener las expresiones de complejidad se planteó la hipótesis de que se trata de polinomios de 3er grado como máximo, lo que fue confirmado haciendo distintas determinaciones con diferentes largos de cadenas. Luego, para comparar el desempeño de los dos autómatas se utilizaron cadenas de largo $\boldsymbol{n}$ = 9, 13, 17 y 21. Conocidos los correspondientes valores de $\boldsymbol{T(n)}$, se plantearon los sistemas de ecuaciones (2) y obtuvieron los coeficientes de los polinomios:

$$\boldsymbol{An^3 + Bn^2 + Cn + D = T(n)} \qquad (1)$$

$$\begin{bmatrix} 9^3 & 9^2 & 9 & 1 \\ 13^3 & 13^2 & 13 & 1 \\ 17^3 & 17^2 & 17 & 1 \\ 21^3 & 21^2 & 21 & 1 \end{bmatrix} \begin{Bmatrix} \boldsymbol{A} \\ \boldsymbol{B} \\ \boldsymbol{C} \\ \boldsymbol{D} \end{Bmatrix} = \begin{Bmatrix} \boldsymbol{T}(9) \\ \boldsymbol{T}(13) \\ \boldsymbol{T}(17) \\ \boldsymbol{T}(21) \end{Bmatrix} \qquad (2)$$

En las Tablas 1 y 2 se muestran las expresiones de complejidad obtenidas para los cuatro casos con la *MTD* y el *AFDB-LF*.

Tabla 1: ***MTD*** - Expresiones de ***T***(n) y orden ***O***

Caso	Expresiones de complejidad	***O***
1	$0{,}4375\boldsymbol{n^2} + 1{,}8750\boldsymbol{n} - 1{,}3125$	n^2
2	$0{,}4375\boldsymbol{n^2} + 1{,}8750\boldsymbol{n} - 1{,}3125$	n^2
3	$0{,}6250\boldsymbol{n^2} + 2{,}2500\boldsymbol{n} - 1{,}8750$	n^2
4	$0{,}0469\boldsymbol{n^3} + 0{,}2969\boldsymbol{n^2} + 1{,}265\boldsymbol{n} - 0{,}609$	n^3

Tabla 2: ***AFDB-LF*** - Expresiones de ***T***(n) y orden ***O***

Caso	Expresiones de complejidad	***O***
1	$0{,}1875\boldsymbol{n^2} + 1{,}1250\boldsymbol{n} + 0{,}6875$	n^2
2	$0{,}1875\boldsymbol{n^2} + 1{,}1250\boldsymbol{n} + 0{,}6875$	n^2
3	$0{,}1875\boldsymbol{n^2} + 2{,}1250\boldsymbol{n} - 1{,}3125$	n^2
4	$0{,}2813\boldsymbol{n^2} + 1{,}0625\boldsymbol{n} - 0{,}6563$	n^2

Los resultados demuestran la fuerte dependencia del comportamiento de las máquinas con respecto al orden de los caracteres en las cadenas, lo que confirma la importancia del impacto de los datos en las evaluaciones de complejidad.

Con la finalidad de destacar las diferencias entre los valores obtenidos con los casos 1/2 y 4, se presentan en la Tabla 3 sus complejidades temporales:

Tabla 3: Valores de complejidad ***T***(n), Casos 1/2 y 4

Complejidad *T*(n) – Casos 1/2 y 4					
Máquina	**caso**	**n = 9**	**n = 13**	**n = 17**	**n = 21**
MTD	1/2	51	97	157	231
	4	69	169	337	591
AFDB-LF	1/2	26	47	74	107
	4	33	62	100	147

Poniendo ahora foco en el *AFDB-LF*, que es presentado en este trabajo, a través de mínimos cuadrados se determinó la expresión que mejor representa los resultados obtenidos, que es la siguiente:

$$\boldsymbol{T(n)} = 0{,}2188\boldsymbol{n^2} + 1{,}4375\boldsymbol{n} + 0{,}0104 \qquad (3)$$

Los valores de complejidad de los cuatro casos y su complejidad media (Ecuación 3, línea punteada) son representados en el gráfico de la Figura 3.

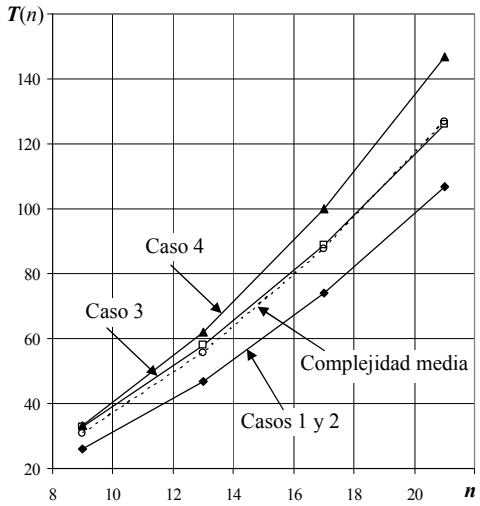

Figura 3: Representación de los polinomios de complejidad temporal $\boldsymbol{T}(n)$ obtenidos con el ***AFDB-LF***.

A partir de una inspección de los resultados presentados surgen los siguientes comentarios:

- El *AFDB-LF* fue más eficiente que la *MTD* en todos los casos estudiados.
- A pesar de que las cadenas de los casos 1 y 2 (α_1 y α_2) aparentan tener muy poca semejanza entre sí, brindaron los mismos resultados con cada máquina.
- Con el caso 4 se comprueba que, si bien la cadena no parece ser demasiado diferente de las anteriores, la *MTD* conduce a una expresión de complejidad de tercer grado. Sin embargo, con el *AFDB-LF* la complejidad sigue representada por un polinomio de segundo grado.
- Analizando los valores de la Tabla 3 se comprueba, con una cadena de largo $\boldsymbol{n} = 21$, que con los casos 1/2 la relación entre la complejidad de la *MTD* respecto del *AFDB-LF* es de 2,16 y con la cadena del caso 4 el valor de la relación sube a 4,0.
- Con la *MTD* la misma comparación entre los casos 1/2 y 4 arroja un valor de 2,56, mientras que con el *AFDB-LF* es de 1,37. Es decir, además de menor complejidad, el autómata propuesto muestra también menor dispersión entre los resultados.
- El gráfico de la Figura 3 muestra que la complejidad media es prácticamente coincidente con la obtenida con el caso 3.
- Todas las cadenas estudiadas con el *AFDB-LF*, de las cuales se seleccionaron los cuatro casos presentados, brindaron resultados dentro de los obtenidos para los casos 1/2 y 4**.** Hasta donde pudo comprobarse, representarían las cotas superior e inferior de la complejidad de este problema cuando es resuelto con ésta máquina.

Con relación a los objetivos de este trabajo, puede comentarse que:

- Si bien la *MTD* es la de mayor capacidad de cálculo, representando el límite de lo computable, una máquina más especializada dotada de memoria auxiliar permitió reducir la complejidad de las soluciones de manera notable.
- La posible dependencia del desempeño de una máquina de sus datos reviste singular importancia, luego este aspecto debe ser tenido especialmente en cuenta siempre que los lenguajes admitan variantes en sus sentencias.
- Cuando se compruebe tal dependencia, en casos en que se desea hacer predicciones a partir de los resultados disponibles, será necesario un trabajo minucioso para identificar las condiciones extremas que pueden presentarse.
- El *AFDB-LF* demostró ser una excelente opción por su desempeño relativamente regular en todos los casos. Además, la posibilidad de rotar el contenido de la memoria auxiliar ofrece un gran potencial que debe seguirse estudiando.
- Asimismo, la disponibilidad en una misma máquina de dos tipos de acceso a la misma memoria (Lifo y Fifo) asociados a sus estados, dan lugar a un muy interesante complemento que ofrece gran flexibilidad y un vasto campo de estudio que debe también ser explorado.
- Quedó confirmada una vez más la importancia de disponer de simuladores apropiados en este tipo de análisis y su gran valor didáctico. Sin la ayuda de simuladores, además de ser muy difícil

estudiar la complejidad de las máquinas, también lo sería la adecuada comprensión de sus comportamientos e identificación de las causas de sus fallas.

CONCLUSIONES

Este trabajo refleja la actividad en un proyecto de investigación de los mismos autores (*Evaluación del impacto de variantes no convencionales en el desempeño de autómatas finitos con memoria de pila*, UTN-3591, 2015), poniéndose el foco en la presentación de un nuevo autómata y la comparación de su desempeño con el de una *MTD*. Para ello, el primer paso fue una indagación histórica de antecedentes, ya que un buen conocimiento de las realizaciones anteriores es siempre el mejor punto de partida. Luego se definió el autómata y un caso de estudio. Los resultados obtenidos permitieron comprobar que: *i*) la especialización de máquinas, que dispongan de ciertos recursos mínimos, puede conducir a mayor eficiencia que las de máquinas generales de mayor capacidad de cálculo y *ii*) la máquina presentada, con memoria auxiliar de acceso Lifo/Fifo, ofrece grandes posibilidades. En efecto, la combinación de ambos tipos de accesos, relacionados a los estados de la máquina, otorga un potencial muy prometedor que debe continuarse estudiando y será motivo de la actividad a realizarse en el futuro inmediato.

REFERENCIAS

[1] Rabin M., Scott D.; *Finite automata and their decision problems*. IBM J. Res. Dev. 3(2), pp. 114–125, 1959.

[2] Sheperdson J.; *The reduction of two-way autómata to one-way autómata*, IBM J. Res., 3:2, 198-200, 1959.

[3] Oettinger A.; *Automatic syntactic analysis and the pushdown store*, Proc. of Symposia on Applied Mathematics, 12, American Mathem. Soc., Rhode Island, 1961.

[4] Schutzenberger M.; *On Context-Free Languages and Push-Down Automata*, Information and Control, 6, 246-264, 1963.

[5] Cherubini A., Citrini C., Crespi-Reghizzi S., Mandrioli D.; *Qrt fifo automata, breath-first grammars and their relations*. Theoret. Comput. Science, 85(1):171-203, 1991.

[6] Petersen H, Robson J.; *Efficient simulations by queue machines,* SIAM Journal on Computing, , Vol. 35, No. 5, pp. 1059-1069, 2006.

[7] Rosenberg B.; *Simulating a stack by queues*, Proceedings of the XIX Latinamerican Conference on Computer Science 1, 3-13, 1993.

[8] Koslowski J.; *Deterministic single-state 2PDAs are Turing complete*; Instituto de ciencias de la computación,TU Braunschweig, Alemania, 2013.

[9] Aho A., Hopcroft J. y Ullman J.; *Time and tape complexity of push-down automaton languages*, Information and Control, 13:3, pp 186-206,1968.

[10] Hopcroft J. y Ullman J.; *Introduction to Automata Theory, Languajes and Computation,* Cap. 14, Addison Wesley, 1979.

[11] Gluck R.; *Simulation of Two-Way Pushdown Automata Revisited,* Dept. of Computer Science, University of Copenhagen, pp. 250-258, 2013.

[12] Alonso M., Díaz V., Vilares M.; *Bidirectional Push Down Automata*, CIAA'02 Proceedings of the 7th int. conference on Implementation and application of automata, pp 35-46, 2003.

[13] Bhattacharjee, A., Uddin R., Debnath, B. DQA: *Automata with new memory, properties and applications*, Int. Journal of Electrical, Electronics and Computer Systems (IJEECS), 2011.

[14] Asha latha B., Vishnupriya T., Himabindu N.; *Deque Automata for all classes of Formal languages*, Inter. Journal of Computational Engineering Research, 2012.

[15] Alfonseca E., Alfonseca M., Morrión R.; *Teoría de Autómatas y Lenguajes Formales*, Editorial McGraw-Hill, 2007.

[16] Giró J., Vázquez J., Meloni B., Constable L.; *Lenguajes Formales y Teoría de Autómatas*, Alfaomega, 2015.

[17] Rytter, W.; *100 Excercises in the Theory of Automata and Formal Languages*, Research Report 99, Ins. of Informatics, Warsaw, 1987.

Agradecimientos

Se agradece a la Secretaría de Ciencia, Tecnología y Posgrado de la de la Universidad Tecnológica Nacional por el soporte financiero brindado al proyecto UTN-3591: *Evaluación del impacto de variantes no convencionales en el desempeño de autómatas finitos con memoria de pila.*

Datos de Contacto

Juan Francisco Giró	*juanfgiro@gmail.com*
Juan Carlos Vázquez	*jcjvazquez@gmail.com*
Brenda Elizabeth Meloni	*bmeloni@gmail.com*
Leticia Edith Constable	*leticiaconstable@gmail.com*

Los autores son miembros de la Cátedra de Sintaxis y Semántica de Lenguajes, Departamento de Ingeniería en Sistemas de Información, FRC, UTN.