RESUMEN DE TESIS DOCTORAL
Desarrollo de Indicadores de Casos Aplicables a la Selección de Algoritmos en el Problema 2-Partition

Development of Instance Indicators Applicable to Algorithm Selection for the 2-Partition Problem

Graduated: Jorge A. Ruiz-Vanoye
Centro Nacional de Investigación y Desarrollo Tecnológico (CENIDET)
jruizvanoye@yahoo.com.mx, http://www.ruizvanoye.com
Graduated on Nov. 26, 2008.
Advisor: Joaquín Pérez Ortega
Centro Nacional de Investigación y Desarrollo Tecnológico (CENIDET)
jpo_cenidet@yahoo.com.mx
Advisor: Rodolfo A. Pazos Rangel
Centro Nacional de Investigación y Desarrollo Tecnológico (CENIDET)
r_pazos_r@yahoo.com.mx

Resumen. En este trabajo se abordó el problema de transformar instancias e indicadores de complejidad entre los problemas Bin-Packing y 2-Partition. Diversos investigadores han realizado reducciones y transformaciones polinomiales entre problemas NP-completos, los principales son Garey & Johnson, Karp y Cook. La transformación de 2-Partition a Bin-Packing existe en la literatura. Sin embargo no existe la transformación de Bin-Packing a 2-Partition, ni la transformación de indicadores con el fin de ser usados en la selección de algoritmos que mejor resuelven una instancia del problema 2-Partition. En esta tesis se propone un nuevo enfoque de solución para transformar instancias, desarrollar indicadores de complejidad y solución de los problemas Bin-Packing al problema 2-Partition, mediante una metodología y el desarrollo de lenguajes formales para expresar las instancias de ambos problemas.
Palabras Claves: Transformación polinomial, lenguajes formales de instancias, compiladores, NP-Completo, selección de algoritmos.

Abstract. In this work we approach the problem of transforming instances and complexity indicators between the problems discovered being Bin-Packing and 2-Partition. Several researchers have reductions and polynomial transformations between NP-complete problems, the main ones Garey & Johnson, Karp and Cook. The transformation of 2-Partition into Bin-Packing exists in literature. Nevertheless, does not exist the transformation of Bin-Packing into 2-Partition, nor the transformation of indicators with the purpose of to be used in the selection of algorithms that better solves an instance of the 2-Partition problem. In this thesis a new approach of solution is proposed to transform instances, to develop complexity indicators and to solve the Bin-Packing problem to the 2-Partition problem, by means of a methodology and formal languages to express the instances of both problems.
Keywords: Polynomial transformation, formal languages of instances, compilers, NP-Complete, algorithm selection.

1 Introducción

1.1 Motivaciones

En trabajos relacionados se han desarrollado transformaciones polinomiales entre diversos problemas NP-completos, pero no existe una manera formal de representar los lenguajes de los problemas ni métodos de verificación de las transformaciones polinomiales. Es interesante y motivante la nueva idea que propone esta investigación de redefinir de manera formal las instancias y las transformaciones polinomiales de instancias e indicadores (con el fin de ser usados para la selección de algoritmos) de un problema NP-completo a otro problema NP-completo usando la teoría de los lenguajes formales, ya que no existía una manera formal correcta de transformar y verificar la transformación de instancias e indicadores de complejidad de un problema NP-completo a otro.

En esta tesis se plantea la transformación de instancias e indicadores de un problema NP-completo a otro. Específicamente del problema Bin-Packing al problema 2-Partition, que permita la selección de algoritmos sobre instancias del problema 2-Partition.

1.2 Descripción del problema de investigación

La aportación principal consiste en transformar instancias del problema Bin-Packing al 2-Partition usando la teoría de los lenguajes formales, así como el transformar los indicadores de complejidad de un problema a otro con el fin de ser usados en la selección de algoritmos sobre el problema 2-Partition. La hipótesis de la investigación es la siguiente: La transformación de instancias e indicadores de complejidad del problema Bin-Packing a 2-Partition se realiza usando la teoría de los lenguajes formales.

1.3 Objetivos de la tesis

Los objetivos de la investigación son:

1. El desarrollo de manera formal de la transformación polinomial de instancias e indicadores del problema Bin-Packing al problema 2-Partition usando la teoría de los lenguajes formales.
2. Un método sistematizado de desarrollo de los lenguajes formales para definir instancias de problemas NP-completos (Bin-Packing y 2-Partition).
3. Resultados experimentales del proceso de transformación de instancias e indicadores del problema Bin-Packing al problema 2-Partition.

1.4 Problemas NP-completos e indicadores utilizados en esta tesis

El Problema de Distribución de Objetos en Contenedores [1] (Bin-Packing Problem ó BPP) consiste en dado un conjunto finito U de n elementos con tamaños $w \in Z^+$, $w = \{w_1,..,w_n\}$, un entero positivo c de la capacidad del contenedor y un entero positivo K (número de contenedores máximos), determinar si existe una partición de U_i ,($i \in \{1, \ldots, K\}$), tal que la suma de los tamaños de los elementos en cada U_i sea c o menos?

El problema 2-Partition consiste en dado un conjunto U de números enteros, determinar si existe una partición binaria de u tal que la suma de los valores de los elementos del subconjunto A es igual a la suma de los valores de A^c; es decir:

$$U = \{u_1, u_2, ..., u_n\}$$
$$A \subset U$$
$$A^c = U - A$$
$$A \cap A^c = \varnothing$$
$$\sum_{i \in A} u_i = \sum_{j \in A^c} u_j$$

Los índices de complejidad son creados para medir la influencia en el desempeño de los algoritmos en el problema de Bin-Packing. El índice del tamaño del caso, índice de capacidad ocupada por un objeto promedio e índice de dispersión del tamaño del objeto. Las definiciones de estos índices se presentan a continuación:

1. Índice del tamaño del caso. El índice p expresa la relación entre el tamaño de un caso y el tamaño del máximo caso resuelto. El tamaño de un caso es el número de objetos de un caso, n, y el tamaño del máximo caso resuelto es $nmax$. El valor de $nmax$ considerado es 1000, el cual corresponde con el número de objetos del caso más grande resuelto reportado en la literatura especializada:

$$p = \frac{n}{n\,max} \tag{1}$$

2. Índice de capacidad ocupada por un objeto. El índice *t* expresa una relación entre el tamaño de los objetos de un caso y el tamaño del contenedor. El tamaño del objeto *i* es S_i y el tamaño del contenedor es *c*. Esta métrica cuantifica la proporción del contenedor *c* que está ocupado por un objeto.

$$t = \frac{\sum_{i=1}^{n} \frac{S_i}{c}}{n} \; ; \; 1 \le i \le n \tag{2}$$

3. Índice de dispersión del tamaño del objeto. El índice *d* expresa el grado de dispersión de los valores de los tamaños de los objetos de un caso. Este índice se mide usando la desviación normal de *t*.

$$d = \sqrt{\frac{\sum_{i=1}^{n}\left[t - \left(\frac{S_i}{c}\right)\right]^2}{n}} \quad \text{ó} \quad d = (\sigma(t)) \tag{3}$$

2 Antecedentes

Las primeras transformaciones polinomiales de instancias conocidas como reducciones polinomiales se atribuyen a Karp y a Cook, aunque en un principio ellos se referían a la reducción de lenguajes de un problema a otro. Es necesario mencionar que a lo que ellos se referían como lenguaje en realidad eran cadenas de símbolos y no propiamente un lenguaje tal y cual lo conocemos actualmente.

El término reducibilidad polinómica se conocía anteriormente de diversas formas: P-reducibility fue la noción básica de reducibilidad entre lenguajes de Cook, la cual contenía reducciones de Turing en tiempo polinomial (polynomial time Turing reducibility), ocasionalmente conocida como Cook-reducibility [2]. Posteriormente Karp [3] cambio el término original de polynomial Turing reducibility por polynomial transformability y le dio como nombre Karp-reducibility. Karp-reducibility permite reducir los miembros de la clase de problemas entre ellos, y es ocasionalmente llamado to many-one reducibility [4, 5, 6]. Actualmente a la reducibilidad polinomial se le conoce como transformación polinomial [7, 8].

2.1. Reducciones polinomiales

La reducción polinomial menciona que un lenguaje L_1 de un problema de decisión es reducible en tiempo polinomial a un lenguaje L_2 de un problema de decisión ($L_1 \leq_p L_2$) si existe una función *f* computable en tiempo polinomial, para toda $x \in L_1$ si y sólo si $f(x) \in L_2$ [5, 6]. En otras palabras la reducción polinomial consiste en reducir un problema a otro a través de una subrutina que soluciona L_2 en tiempo polinomial. Se le llama función *f* a la función de reducción y algoritmo de reducción al algoritmo *F* de tiempo polinomial que computa *f* [5, 6].

2.2. Transformaciones Polinomiales

En la teoría de la NP-completitud existen 4 pasos para reducir problemas NP [5, 6]: 1. Demostrar que el problema *A* está en NP, es decir $A \in$ NP. 2. Seleccionar un problema *B* que se sabe está en la clase NP-completo. 3. Construir un algoritmo de transformación *f* del problema *B* al problema *A*. 4. Verificar que *f* es una función de transformación polinomial.

2.3. Trabajos relacionados

Existen diversos trabajos relacionados con esta investigación, como por ejemplo la investígación realizada por Garey & Johnson, Cook y Karp, pero ninguno de estos trabajos relacionados contiene lo que de manera innovadora se propone en este trabajo. Este trabajo contempla la transformación de instancias de BPP y 2-PAR y de los valores de

los indicadores de ambos problemas usando un compilador; la solución de las instancias transformadas de BPP a 2-PAR, así como la definición formal de lenguajes de instancias de ambos problemas.

3 Enfoque de Solución propuesto

En la teoría de la complejidad computacional existen las reducciones / transformaciones polinomiales de instancias de un problema A (NP-completo) a instancias de otro problema B (NP-completo). En esta sección se propone un nuevo enfoque de solución para transformar instancias y desarrollar indicadores de complejidad entre los problemas Bin-Packing y 2-Partition.

La propuesta consiste en: 1) Una metodología de transformación de instancias y desarrollo de indicadores de complejidad entre problemas NP-completos. 2) El desarrollo de lenguajes formales para expresar las instancias de los problemas Bin-Packing y 2-Partition. Además proponemos: a) un nuevo esquema de codificación de instancias de problemas NP-completos basados en lenguajes formales (haciendo una analogía con lo presentado por Johnson sería la función de reducción *f*), b) una nueva manera transformar instancias y desarrollar indicadores (metodología) mediante el compilador (haciendo una analogía con lo presentado por Johnson sería algoritmo de reducción F de tiempo polinomial que computa *f*) de lenguaje origen (L_1) que define instancias-sí ($Y_{\pi 1}$) del problema Bin-Packing al lenguaje destino (L_2) que define instancias-sí ($Y_{\pi 2}$) del problema 2-Partition.

Las diferencias claves y originales (Fig. 1) entre la metodología propuesta y la estándar propuesta por Johnson consisten: de una manera práctica mediante un compilador es posible transformar polinomialmente instancias del problema BPP a 2-PAR; en el trabajo de Johnson no existe definición de instancias de manera formal lo cual daba un sentido de ambigüedad de la transformación realizada, la metodología propuesta en este trabajo elimina todo tipo de ambigüedad de la transformación al definir formalmente lenguajes para las instancias origen y las instancias destino; en el trabajo de Johnson no se contempla el análisis semántico, la fase de generación de lenguajes (calculo de indicadores de complejidad y la solución del problema mediante un selector de algoritmos) de las transformaciones polinomiales.

La metodología propuesta para transformar instancias, desarrollar indicadores de complejidad y resolver problemas NP-completos usando la teoría de los lenguajes formales se muestra en los pasos a continuación:

1. Seleccionar un problema NP-completo *A* (problema origen) a transformar polinomialmente.
2. Definir el lenguaje formal L_1 (lenguaje origen) para el problema *A*.
3. Seleccionar un problema NP-completo *B* (problema destino).
4. Definir el lenguaje formal L_2 (lenguaje destino) para el problema *B*.
5. Elaborar un compilador que transforme L_1 a L_2 ($A \leq_p B$).
6. Realizar el cálculo de los indicadores, es decir, en la fase generación de lenguaje del compilador, ejecutar el algoritmo que calcula los indicadores implícitos en la instancia del problema *B*.
7. Realizar la transformación inversa de instancias del problema *B* a instancias del problema *A* ($B \leq_p A$).
8. En la misma fase de generación del lenguaje del compilador, ejecutar los algoritmos (aceptación por umbral ó TA, búsqueda tabú ó TS, colonia de hormigas ó ACO, primer ajuste decreciente ó FFD, mejor ajuste decreciente ó BFD, apareamiento al primer ajuste ó MFF, apareamiento al mejor ajuste ó MBF, ajuste modificado decreciente ó MBFD) del selector de algoritmos que resuelven instancias del problema *B*.

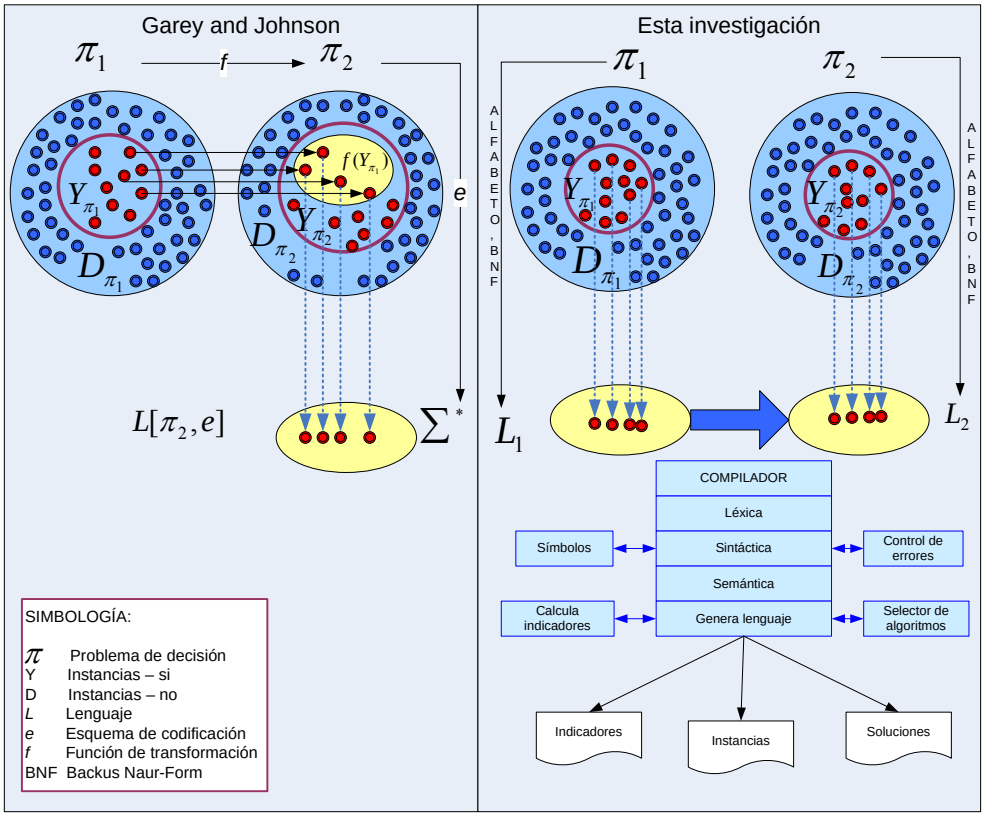

Fig. 1 Comparación de la metodología propuesta

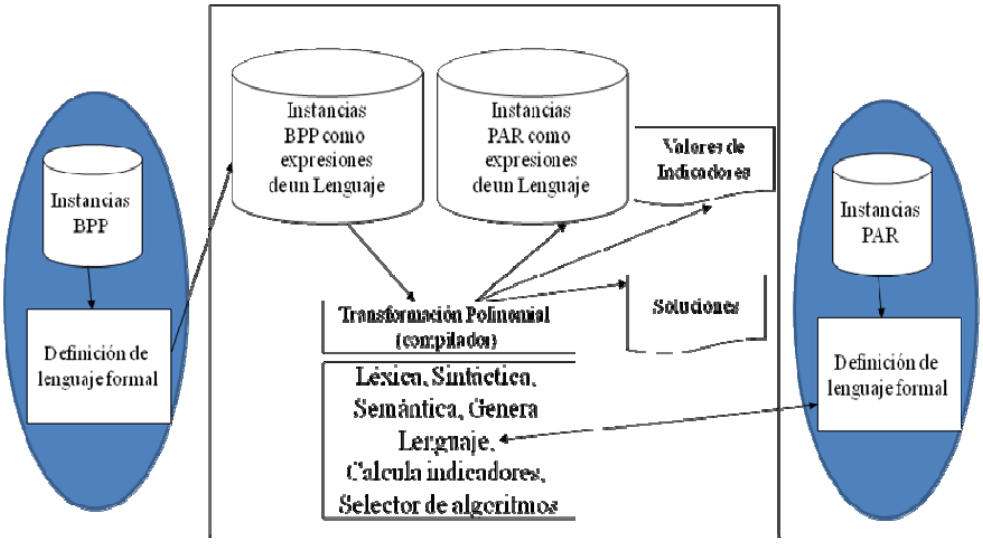

Fig. 2 Transformación de instancias, desarrollo de indicadores y solución de problema

4 Resultados experimentales

La figura 3 muestra el esquema de experimentación usado en esta tesis.

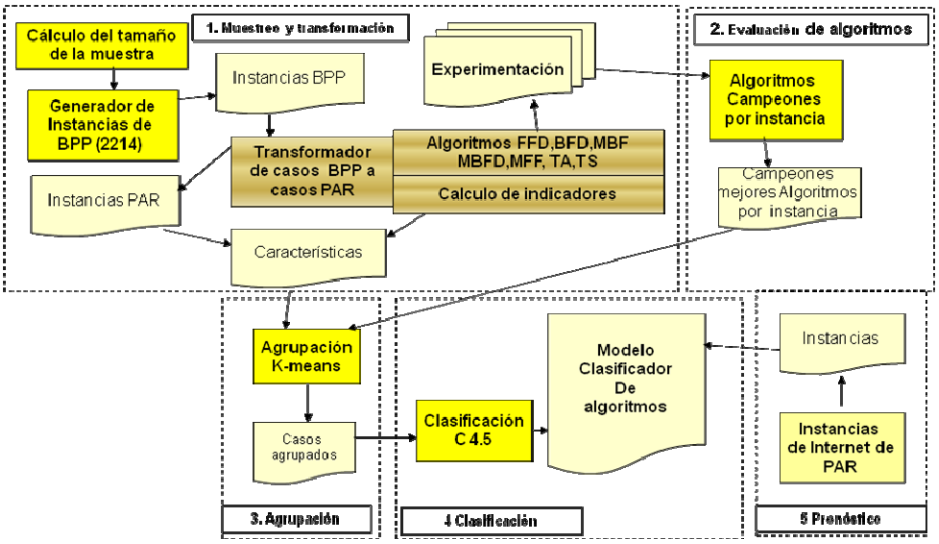

Fig. 3. Experimentación

Para verificar el enfoque de solución se propuso experimentar mediante un compilador ad hoc la metodología de transformación polinomial de instancias, de desarrollo de indicadores de complejidad y de solución de los problemas Bin-Packing al problema 2-Partition.

En la experimentación se generaron aleatoriamente 2214 instancias-sí del problema Bin-Packing. Mediante el compilador solamente se pudieron transformar y obtener soluciones factibles de 1161 instancias al problema 2-Partition y 1053 instancias no se pudieron transformar y en consecuencia no se obtuvieron soluciones factibles.

Con base en los resultados experimentales mencionamos que, a pesar de la creencia generalizada de que una instancia-sí de un problema NP-completo puede ser transformada a una instancia-sí de otro problema y viceversa, para algunas transformaciones de problemas NP-completos eso no es suficiente debido a que no se encuentra explícito que en la transformación inversa se obtenga la instancia original. Tampoco se especifica si para todas las instancias se obtienen instancias-sí en la transformación inversa. El concepto de instancia-sí no es adecuado para todos los casos de la transformación, por lo que proponemos nuevos conceptos: instancia-sí completa e instancia-sí incompleta.

Se define como instancia-sí incompleta a una instancia-sí L_1 ($A \in Y_{\pi 2}$) si es transformada a una instancia-sí L_2 ($B \in Y_{\pi 1}$), y si además al transformarla inversamente de L_2 a L_1 se obtiene una instancia-sí ($C \in Y_{\pi 1}$) diferente a la instancia-sí L_1, ver Fig. 4.

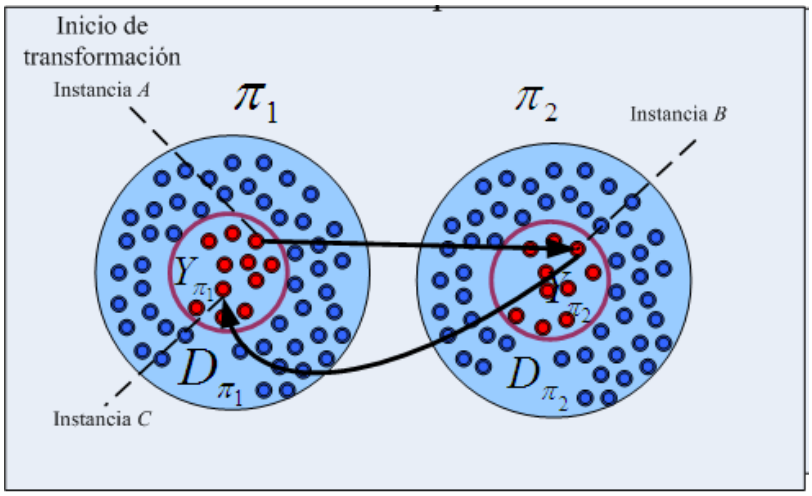

Fig. 4. Instancia-sí incompleta

Se define como instancia-sí completa a una instancia-sí L_2 ($A \in Y_{\pi 2}$) si es transformada a una instancia-sí L_1 ($B \in Y_{\pi 1}$), y si además al transformarla inversamente de L_1 a L_2 se obtiene la instancia-sí L_2 ($A \in Y_{\pi 2}$), ver Fig. 5.

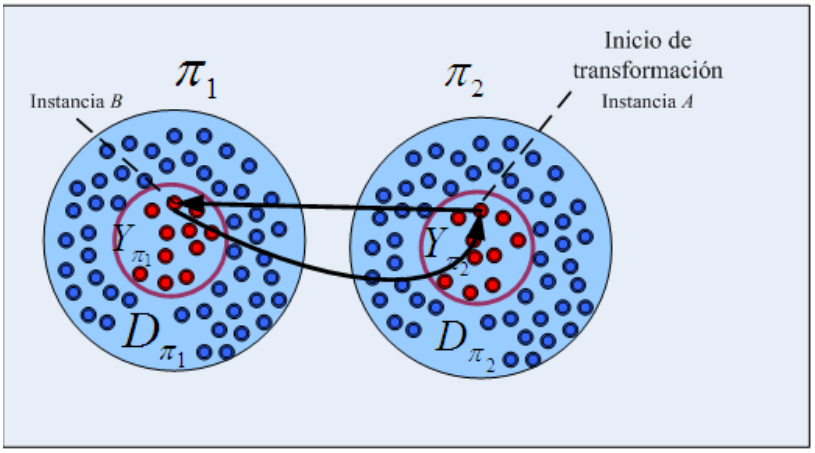

Fig. 5. Instancia-sí completa

5 Conclusiones

Las principales contribuciones de esta tesis fueron las siguientes: a) Se desarrolló una metodología de transformación de instancias entre problemas NP-completos (Bin-Packing y 2-Partition). b) Se definieron lenguajes formales para definir instancias de los problemas Bin-Packing y 2-Partition. c) Se desarrolló un compilador que permite transformar instancias expresadas como lenguajes del problema Bin-Packing al problema 2-Partition. d) Se desarrolló un algoritmo para el cálculo de los valores de los indicadores de las instancias transformadas para ser utilizado en el compilador de instancias. e) Se obtuvieron los siguientes resultados: para los lenguajes (instancias) transformados se generaron 2214 instancias-sí (con solución) del problema Bin-Packing que fueron correctamente transformadas a 1161 instancias-sí (con solución) y 1053 instancias-no (sin solución) del problema 2-Partition. f) Con base en las pruebas que se realizaron se encontró que en otras reducciones polinomiales (diferentes a la transformación de este

documento del problema Bin-Packing al problema Partition) era posible reducir instancias si a instancias-sí e instancias-sí a instancias-no, debido a que cuando se realizaron ese tipo de reducciones polinomiales aun no existía la teoría base para las transformaciones polinomiales propuesta por Garey & Johnson o no significaban lo mismo, la cual menciona que sólo es posible transformar instancias-sí con instancias-sí.

Con base en los resultados obtenidos en esta investigación (es posible la transformación de instancias e indicadores de complejidad del problema Bin-Packing a 2-Partition usando la teoría de los lenguajes formales o un compilador) creemos que es posible transformar polinomialmente de L_1 a L_2 mediante un compilador instancias de problemas NP-completos, con lo cual podemos sugerir como un trabajo futuro relacionado la verificación de la siguiente proposición: Un compilador puede transformar polinomialmente de L_1 a L_2 cualquier problema NP-completo a otro problema cualquiera NP-completo.

Referencias

1. Martello, S., Toth, P.: Knapsack Problems: Algorithms and Computer Implementations. John Wiley & Sons. England (1991) 221-239

2. Cook, S.A. The Complexity of Theorem Proving Procedures. Proc. 3rd ACM Symposium on Theory of Computing. Shaker Heights, Ohio (1971) 151-158

3. Karp, R.M.: Reducibility Among Combinatorial Problems. In: Complexity of Computer Computations. R.E. Miller and J.W. Thatcher, Eds. Plenum Press, New York (1972) 85-104

4. Hartmanis, J., Hopcroft, J.E.: An Overview of the Theory of Computational Complexity. Journal of the Association for Computing Machinery. Vol. 18, No. 3 (1971) 444-475

5. Garey, M.R., Johnson, D.S.: Computers and Intractability: A guide to the Theory of NP-Completeness. W.H. Freeman and Company. New York (1979)

6. Cormen, H.T., Charles, E.: Introduction to Algorithms. Second Edition. MIT Press (2001)

7. Papadimitriou, C.H.: Computational Complexity. Addison-Wesley (1994)

8. Sipser, M.: Introduction to the Theory of Computation. Thomson Course Technology (2006)

Jorge A. Ruiz Vanoye *was born in D.F. Mexico in 1975. He obtained his Ph.D. degree in Computer Science in 2008 from CENIDET, under supervision of Joaquín Pérez and Rodolfo Pazos. He has worked at Electric Research Institute of Mexico government (IIE) and in diverse companies more. He has given classes in diverse Mexican Universities since 1996. He is reviewer of Computers & Security (2008, 2009), Neural Computing & Applications journal (2009), Journal of Computer Information Systems (2009) and others journals and conferences; for publications and more information see www.ruizvanoye.com.*

Joaquín Pérez Ortega *He was born in México in 1958. He obtained his Ph. D. degree in Computer Sciences in 1999, with a thesis on design of distributed databases from the Tecnológico de Monterrey. Since 1989 he is professor at Centro Nacional de Investigación y Desarrollo Tecnológico. He is a National Researcher of Mexico (SNI), Senior Member of IEEE, and author of more than 100 publications on databases and metaheuristic optimization. He was director of 9 Ph. D. Thesis.*

Rodolfo A. Pazos Rangel *received the Ph. D. degree in computer science from the University of California at Los Angeles (USA) in 1983. His scientific interests include distributed databases design and natural language processing.*