Comparativa de algoritmos bioinspirados aplicados al problema de calendarización de horarios

Lucero de Montserrat Ortiz Aguilar[1], Juan Martín Carpio Valadez[1], Héctor José Puga Soberanes[1], Claudia Leticia Díaz González[1], Carlos Lino Ramírez[1] y Jorge AlbertoSoria-Alcaraz[2]

[1] Instituto Tecnológico de León, León, Guanajuato, México

[2] Universidad de Guanajuato, Guanajuato, Guanajuato, México

{ldm_oa,jmcarpio61}@hotmail.com, pugahector@yahoo.com,{posgrado,carloslino}@itleon.edu.mx, jorge.soria@ugto.mx

Resumen. El problema de calendarización de eventos está presente en diversas organizaciones como lo son escuelas, hospitales, centros de transporte, etc. La calendarización de actividades en una universidad tiene como propósito el garantizar que todos los estudiantes tomen sus asignaturas requeridas apegándose a los recursos que están disponibles. El conjunto de restricciones que debe contemplarse en el diseño de horarios involucra a los alumnos, maestros e infraestructura. En este trabajo se muestra que mediante la aplicación de algoritmos Genéticos, Memético y Sistema Inmune se generan soluciones aceptables, para el problema de calendarización de tareas, Los algoritmos son aplicados a instancias reales del Instituto Tecnológico de León y sus resultados son comparables con los de un experto humano.

Palabras clave: algoritmo genético, algoritmo memético, sistema inmune, faculty timetabling, course timetabling.

1. Introducción

La calendarización de tareas dentro de las organizaciones es uno de los problemas más comunes y difíciles de tratar, debido a que se busca asignar diversas actividades y recursos en un espacio de tiempo [1]. En las Universidades, se busca generar un diseño de horarios que cumpla con las restricciones de los alumnos, profesores, plan de estudios e inmuebles de la institución. Además de que el problema de calendarización depende del tipo de escuela, universidad y sistema de educación, por lo cual no existe un diseño de horarios que pueda ser aplicado de manera generalizada a todos los casos [2].

En general el problema de calendarización de horarios se define a partir de un conjunto de eventos (clases, cursos, exámenes), los cuales deben ser asignados en un conjunto de espacios de tiempo y que están sujetos a un conjunto de restricciones [3].

La calendarización Universitaria de acuerdo con Adriaen et.al [4] se clasifica en 5 grupos:

1. **Faculty Timetabling (FTT)**. Es la asignación de maestros a materias.
2. **Class-Teacher Timetabling (CTTT)**. Es asignación de materias con el menor conflicto temporal posible entre grupos de alumnos.
3. **Course Timetabling (CTT)**. Es la asignación de materias con el menor conflicto temporal posible entre alumnos individuales.
4. **Examination Timetabling (ETT)**. Es la asignación de exámenes a los alumnos, de tal forma que el alumno no aplique dos pruebas al mismo tiempo.
5. **Classroom assignment (CATT)**. Después de asignar las clases a los maestros, se asignan *class-teacher* a los salones.

En este trabajo se enfoca a generar soluciones aceptables al problema de calendarización de horarios, mediante el uso de algoritmos Metaheurísticos. Existen una diversa cantidad de enfoques que han sido utilizados para resolver el problema de calendarización de horarios como el coloreo de grafos [5], programación de Satisfacción de Restricciones(CSP) Métodos Basados[7], IP/LP (programación entera/programación lineal) [6], Algoritmos Genéticos [2, 8, 9], Algoritmos Meméticos [10, 11], Búsqueda Tabú [12, 13], Recosido Simulado [14], Búsqueda Local [15], Mejor-Peor sistema de Hormigas (BWAS) y las optimización por colonia de hormigas[16] y enfoque hiperheurístico [17].

Existen una gran cantidad diferentes problemas al menos de clase NP, los cuales pueden ser resueltos con diversos algoritmos Metaheurísticos, pero como nos indica el Teorema de No-Free Luch [18] no existe una Metaheurística que supere en rendimiento a todas las demás para todos los problemas conocidos de la clase NP. Debido a lo anterior, en este trabajo se hace una comparativa entre algoritmo Genético, Memético y Sistema Inmune, aplicando pruebas estadísticas no paramétricas.

Las instancias utilizadas pertenecen a datos reales del Instituto Tecnológico de León (ITL), donde la calendarización de horarios se elabora por un experto y se busca mediante los algoritmos Metaheurísticos generar soluciones que compitan con las propuestas el experto humano.

2. El problema de calendarización de tareas

El calendarización de tareas puede ser definido como el proceso de asignar clases a recursos como lo son de tiempo, espacio (salones) y maestros (personal), mientras se satisfaga un conjunto restricciones [19].

Existen dos tipos de restricción [20]:

- **Duras.** Es la restricción que absolutamente no puede ser quebrantada. Algunos ejemplos de restricciones duras son [12]: disponibilidad del salón, conflictos entre alumnos, disponibilidad de recursos (maestros, aulas).
- **Blandas.** El conjunto de restricciones que se prefieren satisfacer pero no se supone satisfacerlas todas. Algunos ejemplos de restricciones blandas son [12]: capacidad del Salón, mínimo de días ocupados, Etc.

Una definición del problema de calendarización de tareas está dada por Lewis Rhydian [21]: Dada una 4-tupla (e, t, p, S) la cual es la representación de una posible solución, en donde $E = \{e_1, e_2, \dots, e_n\}$ es un conjunto de eventos (clases o materias), $T = \{t_1, t_2, \dots, t_s\}$ es un conjunto de periodos de tiempo, $P = \{p_1, p_2, \dots, p_m\}$ es un conjunto de lugares (salones), $A = \{a_1, a_2, \dots, a_o\}$ un conjunto de usuarios (estudiantes registrados en cursos) y $S \subseteq A$ un subconjunto de estudiantes; la 4-tupla tiene asociada una función de costo $f(t)$. El problema es buscar las 4-tuplas (e, t, p, S) o soluciones tales que minimicen la función de costo $f(t)$ asociada.

2.1. Metodología API-Carpio

La metodología API-Carpio [22] describe el proceso de calendarización de horarios educativos como:

$$f(x) = FA(x) + FP(x) + FI(x) \tag{1}$$

Dónde:

$FA(x) =$ Número de estudiantes en conflicto dentro del horario x, (CTT).

$FP(x) =$ Número de profesores en conflicto dentro del horario x, (FTT).

$FI(x) =$ Número de aulas y laboratorios en conflicto dentro del horario x,(CATT).

En este trabajo se restringe a tomar solamente hasta $FA(x)$ la cual está definida como:

$$FA = \sum_{j=1}^{k} FA_{V_j} \tag{2}$$

dónde:

$$FA_{V_j} = \sum_{s=1}^{\left(M_{V_j}\right)-1} \sum_{l=1}^{\left(M_{V_i}\right)-s} (A_{j,s} \wedge A_{j,s+l}) \tag{3}$$

Teniendo:

$FA_{V_j} =$ Número de estudiantes en conflicto dentro del vector V_j.

$V_j =$ Es un vector de tiempo que contiene diferentes materias.

$A_{j,s} \wedge A_{j,s+l} =$ Número de alumnos que demandan la inscripción simultánea de las materias $M_{j,s} \wedge M_{j,s+1}$.

2.2. Metodología del diseño

La metodología del diseño propuesta por Soria et al. [23] permite que diferentes políticas de la calendarización de tareas y listas de restricciones sean modelados mediante la conversión de todas las restricciones de tiempo y espacio en un simple tipo de restricción: conflictos de estudiantes. En esta se proponen estructuras como lo son la matriz MMA, lista LPH, lista LPA y lista LPS. Las primeras tres representan las restricciones duras y la última representa las restricciones blandas. En este trabajo se

utilizaran dos de las cuatro estructuras las cuales son: matriz MMA y lista LPH. En [23] nos definen que sus estructuras son las siguientes:

Matriz MMA: Contiene el número de conflictos (entre alumnos) posibles si dos lecciones son asignadas en el mismo espacio de tiempo. La figura 1a muestra un ejemplo de matriz MMA.

Lista LPH: Esta lista nos da información acerca de cada lección (clase, evento o materia) en que posible espacio de tiempo puede ser asignada. La figura 1b muestra un ejemplo de LPH.

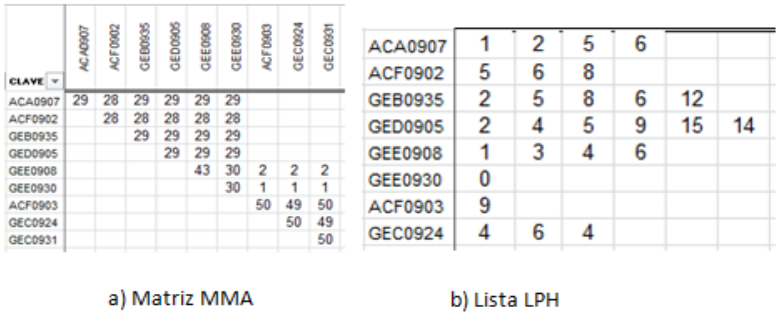

Fig. 1. Ejemplo de MMA y LPH

3. Algoritmo genético

Los algoritmos Genéticos fueron desarrollados por J. Hollan en los 70s, con el fin de entender el proceso adaptativo del sistema natural, para luego aplicarlo en la optimización y Machine Learning en los 1980s [25]. Los algoritmos Genéticos son métodos adaptativos, generalmente usados en problemas de búsqueda y optimización de parámetros, basados en la reproducción sexual y en el principio de supervivencia del más apto. En [26] define el algoritmo 1 que corresponde a un Genético Simple.

La representación de las soluciones candidatas se muestra en la figura 2, donde en cada posición representan un evento o materia y la columna representa el espacio de tiempo asignado de la LPH.

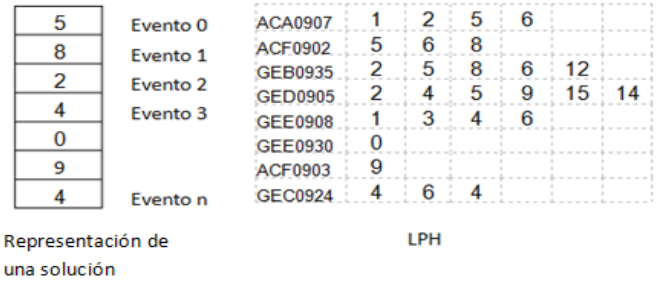

Fig. 2. Representación de las soluciones candidatas

Algoritmo 1 Genético Simple

Requiere: Función Objetivo $F(x), x = (x_1, \dots, x_n)^T$

```
Generar una población Inicial
Mientras t < Max número de generaciones hacer
    Generar una nueva solución por la cruza y la muta
        Cruza
        Muta
    Seleccionar la mejor solución para la siguiente generación (elitismo)
Fin del lazo
Regresar La mejor solución de la población
```

En [24] menciona que el Genético es una de las estrategias más usadas y en este trabajo se empleó la selección por Ruleta pesada. Esta consiste en asignar a cada individuo una parte proporcional de probabilidad en relación a la función de aptitud. Teniendo f_i el fitness del individuo p_i en la población P y La fórmula de probabilidad se puede consultar en [27]. La cruza es el proceso probabilístico que cambia la información entre dos cromosomas (padres) para generar dos cromosomas hijos [30] y la empleada en este trabajo es la cruza a un punto descrita en [24], donde el sitio de cruza k es seleccionado aleatoriamente y los dos hijos son creados intercambiando los segmentos de los padres. Y para el operador de muta tenemos que será a un punto donde seleccionaremos aleatoriamente una posición k y cambiaremos el valor de esa posición por otro que se encuentre en la LPH [24].

4. Algoritmos meméticos

En 1976 Dawkins diseñó el concepto de meme, el cual a diferencia del gen puede ser modificado por su portador. Supuso que existe un progreso como un gen del algoritmo genético que es transferido a la próxima generación, es decir, las características obtenidas se transfieren de una generación anterior a una siguiente, junto con los cambios de población. Moscato en [32] donde describe lo que es el algoritmo Memético.

Algoritmo 2 Memético

Requiere: Función Objetivo $F(x), x = (x_1, \dots, x_n)^T$

```
Generar una población Inicial
Mientras t < Max número de generaciones hacer
    Generar una nueva solución por la cruza y la muta
    Hacer Cruza
    Hacer búsqueda Local
    Hacer Muta
    Seleccionar la mejor solución para la siguiente generación (elitismo)
Fin del lazo
Regresar La mejor solución de la población
```

Un meme es una unidad de datos que puede recrearse a sí misma. Estas unidades se transmiten entre las personas y cualquier otra, que se pueden ajustar con ella, y es capaz de salvar la unidad de datos; mientras que un gen se mantiene sin cambios durante la transmisión [33]. Los componentes de un Algoritmo Memético son [24]: Algoritmo Genético y Búsqueda Local.

La búsqueda local es una modificación que se puede llegar a hacer toda la población de individuos con la que trabaja el algoritmo. En este se realiza una copia de cada individuo, donde esta copia es alterada de alguna forma; si la copia de un individuo en específico es mejor que la original, la copia remplaza al individuo original. En el algoritmo 2 se describen los pasos de un Memético.

5. Sistema inmune

Estos se basan en imitar el comportamiento del sistema inmunológico humano, el cual se encarga de proteger al cuerpo de los patógenos externos e internos y su tarea principal es reconocer las células en el cuerpo clasificarlas como propias y no propias. Los algoritmos de Sistema Inmune artificial han sido aplicados con éxito en diversos problemas de optimización [29].

Algoritmo 3 Sistema Inmune Artificial

Requiere: tamaño máximo de población, número de clones

1: Generar una población Inicial
2: **Mientras** $t < Max$ número de generaciones **hacer**
3: Selección.
4: Clonar.
5: Hipermuta
6: **Si tamaño de la población > max_tamaño entonces**
7: Poda (autorregulación)
8: **Fin**
9: **Fin del lazo**
10: Regresar La mejor solución de la población

Básicamente el proceso del algoritmo de Sistema Inmune Artificial consiste en generar aleatoriamente una población de soluciones candidatas; después seleccionamos un porcentaje de los mejores individuos, los cuales son clonados, luego a estos individuos se les aplica una hipermuta y finalmente continuamos hasta llegar a nuestra solución objetivo, pero para evitar que nuestra población crezca sin medida se pone una poda la cual nos permitirá regresar al tamaño inicial de la población [34].

El algoritmo 3 propuesto por [31], corresponde al Sistema Inmune Artificial.

6. Experimentación y resultados

Las instancias usadas para las pruebas con las Metaheurísticas antes mencionadas pertenecen a ITL, estas corresponden a dos planes educativos distintos, pertenecientes al año del 2009 y 2014, cuentan con aproximadamente de 46 a 58 clases (eventos) y una cantidad de 9 a 11 espacios de tiempo respectivamente. La configuración utilizada en los algoritmos Genético, Memético y SI se muestran en la tabla 1, donde tenemos que las llamadas a función fueron 300,000, la población inicial para cada uno fue la misma. El criterio de paro de los algoritmos fue el de llamadas a función.

Tabla 1. Datos para la configuración Inicial del AG, AM y SI

Parámetro	Genético	Memético	Sistema Inmune
Población	20	20	20
Llamadas a función	300,000	300,000	300,000
Elitismo	0.1	0.1	NA
Cruza	0.9	0.9	NA
Muta	0.15	0.15	NA
Poda	NA	NA	100

La tabla 2 muestra los resultados obtenidos (conflictos) de acuerdo a la función objetivo mostrada en 1, donde se puede observar la media, la mediana y la deviación estándar de las ocho instancias evaluadas con el AG, AM, SI, y los conflictos del experto.

Tabla 2. Resultados las diferentes Metaheurísticas y experto aplicados a las instancias.

Instancia	Media			Mediana			Desviación estándar			Experto
-	AG	AM	SI	AG	AM	SI	AG	AM	SI	
1	281.3	282.6	295.7	279	282	304	16.24	16.10	18.7	577
2	161.6	162.5	194.2	162	160	209	14.73	14.00	13.54	308
3	247.6	265.0	293.9	250	262	325	19.71	19.84	20.58	444
4	160.6	150.9	177.3	147	152	189	11.03	10.61	9.07	249
5	129.1	135.7	164.7	131	137	178	10.34	8.02	9.59	300
6	73.9	77.9	89.0	74	77	99	6.81	7.19	7.73	157
7	75.0	79.5	180.9	75	81	190	7.69	9.47	9.92	138
8	91.1	97.0	104.7	89	95	109	12.8	16.99	11.55	258

Se puede observar en la tabla 2 que las desviaciones estándar de algunas instancias no hay gran diferencia entre algoritmos, como son el caso de la instancia 2, 3 y 6, ya que lo que se busca es encontrar algoritmos que tengan soluciones aceptables con una baja desviación estándar, lo que quiere decir que los datos están muy cercanos a la media, lo cual es indicativo de reproducibilidad de resultados por parte de los algoritmos Metaheurísticos utilizados. Para aplicar las pruebas estadísticas no paramétricas se aplicó el test de Friedman [28], de donde se tomarán las medianas de los algoritmos genético, Memético y SI y después aplicar el test y conocer cuál fue el algo-

ritmo con el mejor rendimiento se aplicó la prueba de suma de rangos con signo de Wilcoxon, para contrastar los resultados del algoritmo con mejor rendimiento y los resultados del experto humano.

En la tabla 3 se muestran los rangos y resultados del test ómnibus de Friedman donde $h_0 =$ No existen diferencias en el desempeño de los algoritmos y $h_a =$ Existen diferencias entre Algoritmos, de donde el valor *P* es menor en los tres casos, que el valor de $\alpha = 0.05$, por lo que no tenemos suficiente evidencia para aceptar h_0. Tomando como algoritmo de control al Genético, debido a que es el que tiene el menor rango hacemos las pruebas post-hoc, con un valor de $\alpha = 0.05$.

Tabla 3. Rangos, estadísticos y valor p para AG, AM y SI.

Algoritmos	**Friedman**	**Friedman Alineado**	**Quade**
Genético	**1.125**	**59**	**1.083**
Memético	1.875	77.1	1.916
SI	3.000	164	3.000
Estadístico	14.25	12.19	19.26
Valor *P*	0.0008	0.0079	6.565E-05

En la tabla 4 se muestran los valores z y los valores p con ajuste Bonferroni [28]. Como podemos observar en las pruebas a pares para el caso del AG vs el AM en los tres test el valor p es menor que el $\propto$ por lo cual no tenemos suficiente evidencia para rechazar h_0, mientras que para el caso de AG vs SI nuevos valores p son menores que el α, por lo cual nos dice que existe diferencia en el comportamiento de los algoritmos

Tabla 4. Pruebas Post-Hoc. Tomando como algoritmo de control al Algoritmo Genético.

Algoritmo	**Friedman**		**Friedman Alineado**		**Quade**	
-	z	Bonferroni	z	Bonferroni	z	Bonferroni
AG vs AM	1.50	0.2672	1.06	0.5723	1.48	0.2749
AG vs SI	3.75	0.0003	6.18	1.227E-09	3.41	0.0012

Como no existe diferencia en el comportamiento del Genético y el Memético utilizaremos al que tuvo el rango más pequeño en las pruebas y en este caso fue el Genético (ver tabla 3) para compararlo en la prueba de suma de rangos de con signo Wilcoxon y ver si los resultados del genético con los resultados del experto provienen de poblaciones con medianas iguales. Los resultados se muestran en la tabla 5.

Al Aplicar la prueba de suma de rangos con signo de Wilcoxon (ver tabla 5), bajo las hipótesis: $h_0 =$ No existen diferencias en las medianas vs. $h_a =$ Existen diferencias entre las medianas, donde se tomó un valor de significancia $\alpha = 0.05$ y 8 grados de libertad, encontramos que existe evidencia suficiente para rechazar h_0 . Por lo tanto, tenemos que efectivamente los datos provienen de poblaciones con medianas diferentes, es decir, el algoritmo Genético mejora los resultados obtenidos por el experto humano.

Tabla 5. Prueba de suma de rangos con signo de Wilcoxon entre Genético y resultados del Experto humano

Instancia	Genético	Experto	Rankeo	Signo
1	279	577	8	-
2	162	308	4	-
3	250	444	7	-
4	147	249	3	-
5	131	300	5.5	-
6	74	157	2	-
7	75	138	1	-
8	89	258	5.5	-
W^- =	36	W =	0	
W^+=	0	W_0=	4	

7. Conclusiones

El uso de algoritmos de optimización en la calendarización de horarios nos permiten minimizar el número de conflictos entre los diferentes recursos de la institución como lo son los docentes, el inmueble; permitiendo que los estudiantes puedan tener un calendario que satisfaga sus necesidades. En este trabajo se muestra primero una comparación entre diferentes Metaheurísticas, que nos permiten encontrar soluciones que resuelven el problema de la calendarización de tareas; posteriormente se determina cual es el que obtuvo mejor desempeño y se hace una comparación con los resultados del experto mediante una prueba estadística no paramétrica.

Las instancias utilizadas pertenecen a insumos reales del ITL, de donde se obtuvieron el número de conflictos de la solución propuesta por el experto encargado del diseño de los horarios; permitiendo hacer una comparación entre los resultados generados por las Metaheurísticas vs. los del experto humano. Los resultados muestran que el Algoritmo Genético fue él tuvo mejor desempeño, sin embargo, la prueba estadística de Friedman nos señala que no existe suficiente evidencia discernible entre el comportamiento de este y el Memético, pero debido al valor de los rangos fue el que se tomó para comparación con los resultados del experto.

Por último al hacer la prueba de rangos con signo de Wilcoxon entre el Genético y los resultados del experto, nos indica que existe diferencia de posición entre las distribuciones de resultados del Genético y el experto humano, por lo que, basados en los resultados podemos afirmar que el algoritmo Genético mejoró los resultados para este conjunto de instancias.

Como trabajo futuro se propone el integrar otros algoritmos Metaheurísticos empleando otro tipo de selección, cruza y muta para el caso del genético y Memético e implementar otras versiones de Sistema Inmune que nos generen mejores soluciones y aplicar la prueba ómnibus incorporando los resultados del experto humano junto a los Metaheurísticos. Además de tomar un enfoque que involucre las variables restan-

tes de la Metodología API-Carpio, como lo es la suma de los conflictos del maestro, junto con los de las aulas.

Agradecimientos. Agradecimientos al Consejo Nacional de Ciencia y Tecnología (CONACYT) México por el apoyo brindado en esta investigación con el número de beca 308646 y a la División de Estudios de Posgrado del Instituto Tecnológico de León.

Referencias

1. Jorge A, S., Martin, C. J., & Hugo, T.: Academic timetabling design using hyper-heuristics. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 43–56 doi:10.1007/978-3-642-15534-5_3 (2011; 2010)
2. Asratian, A. S., de Werra, D.: A generalized class–teacher model for some timetabling problems, University of Technology, Department of Engineering Sciences and Mathematics, Mathematical Science, & Mathematics, European Journal of Operational Research, pp. 531–542 (2002) doi:10.1016/S0377-2217(01)00342-3.
3. Soria-Alcaraz Jorge, A., Martín, C., Héctor, P., & Sotelo-Figueroa, M. A.: Comparison of metaheuristic algorithms with a methodology of design for the evaluation of hard constraints over the course timetabling problem. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 289–302, doi:10.1007/978-3-642-33021-6_23 (2013)
4. Adriaen, M., Causmaecker, P., Demeester, P.: Tackling the university course timetabling problem with an aggregation approach. In: Burke, K., Rudova, H. (eds.) Proceedings PATAT 2006, pp. 330–335 (2006)
5. De Werra, D.: An introduction to timetabling. European Journal of Operational Research, vol. 19, no. 2, pp. 151–162 (1985).
6. Obit, J. H., Ouelhadj, D., Landa-Silva, D., Vun, T. K., & Alfred, R.: Designing a multi-agent approach system for distributed course timetabling, pp. 103–108, doi:10.1109/HIS.2011.6122088 (2011)
7. Lewis, M. R. R.: Metaheuristics for university course timetabling. Ph.D. Thesis, Napier University (2006)
8. Deng, X., Zhang, Y., Kang, B., Wu, J., Sun, X., & Deng, Y.: An application of genetic algorithm for university course timetabling problem, pp. 2119–2122, doi:10.1109/CCDC.2011.5968555 (2011)
9. Mahiba, A.A. & Durai, C.A.D.: Genetic algorithm with search bank strategies for university course timetabling problem. Procedia Engineering, vol. 38, pp. 253–263 (2012)
10. Soria-Alcaraz, J. A.; Carpio, J. M.; Puga, Hé.; Melin, P.; Terashima-Marn, H.; Reyes, L. C. & Sotelo-Figueroa, M. A. Castillo, O.; Melin, P.; Pedrycz, W. & Kacprzyk, J.: Generic Memetic Algorithm for Course Timetabling. In: ITC2007 Recent Advances on Hybrid Approaches for Designing Intelligent Systems, Springer, vol. 547, pp. 481–492 (2014)
11. Nguyen, K., Lu, T., Le, T., & Tran, N.: Memetic algorithm for a university course timetabling problem. pp. 67–71, doi:10.1007/978-3-642-25899-2_10 (2011)
12. Aladag, C., & Hocaoglu, G.: A tabu search algorithm to solve a course timetabling problem. Hacettepe journal of mathematics and statistics, pp. 53–64 (2007)
13. Moscato, P.: On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Towards Memetic Algorithms. Caltech Concurrent Computation Program (report 826) (1989)

14. Frausto-Solís, J., Alonso-Pecina, F., & Mora-Vargas, J.: An efficient simulated annealing algorithm for feasible solutions of course timetabling. Springer, pp. 675–685 (2008)
15. Joudaki, M., Imani, M., & Mazhari, N.: Using improved Memetic algorithm and local search to solve University Course Timetabling Problem (UCTTP). Doroud, Iran: Islamic Azad University (2010)
16. Thepphakorn, T., Pongcharoen, P., & Hicks, C.: An ant colony based timetabling tool. International Journal of Production Economics, vol. 149, pp. 131–144 doi:10.1016/j.ijpe.2013.04.026 (2014)
17. Soria-Alcaraz, J., Ochoa, G., Swan, J., Carpio, M., Puga, H., & Burke, E.: Effective learning hyper-heuristics for the course timetabling problem. European Journal of Operational Research, pp. 77–86, doi:10.1016/j.ejor.2014.03.046 (2014)
18. Wolpert, H., Macready, G.: No free lunch Theorems for Search. Technical report, The Santa Fe Institute, vol. 1 (1996)
19. Lai, L. F., Wu, C., Hsueh, N., Huang, L., & Hwang, S.: An artificial intelligence approach to course timetabling. International Journal on Artificial Intelligence Tools, pp. 223–240, doi:10.1007/s10479-011-0997-x (2008)
20. McCollum, B., McMullan, P., Parkes, A. J., Burke, E. K., & Qu, R.: A new model for automated examination timetabling. Annals of Operations Research, pp. 291–315 (2012; 2011)
21. Conant-Pablos, S.E., et al.: Pipelining Memetic algorithms, constraint satisfaction, and local search for course timetabling. In: MICAI Mexican International Conference on Artificial Intelligence, vol. 1, pp 408–419 (2009)
22. Carpio-Valadez, J.M.: Integral Model for optimal assignation of academic tasks. In: Encuentro de investigacion en ingenieria electrica, ENVIE, Zacatecas, pp. 78–83 (2006)
23. Soria-Alcaraz, J. A., Martin, C., Héctor, P., Hugo, T., Laura, C. R., & Sotelo-Figueroa, M. A.: Methodology of design: A novel generic approach applied to the course timetabling problem. pp. 287–319, doi:10.1007/978-3-642-35323-9-12 (2013)
24. Talbi, E. Metaheuristics: From design to implementation. US: Wiley (2009)
25. Goldberg, D. E.: Genetic algorithms in search, optimization, and machine learning. Reading, Mass: Addison-Wesley Pub. Co. (1989)
26. Yang, X.-S. Nature-inspired metaheuristic algorithms. Luniver press (2010)
27. Abdoun O., Abouchabaka J.: A Comparative Study of Adaptive Crossover Operators for Genetic Algorithms to Resolve the Traveling Salesman Problem. International Journal of Computer Applications (2011)
28. Derrac, J., García, S.: A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence. Swarm and Evolutionary Computation (2011)
29. Azuaje, F.: Review of "Artificial immune systems: A new computational intelligence approach." Journal Neural Networks, vol. 16, no.8, Elsevier, pp. 1229–1229 (2003)
30. Maulik, U., Bandyopadhyay, S.: Genetic algorithm-based clustering technique. Pattern Recognition, Vol. 33, pp. 1455–1465 (2000)
31. Herrera Lozada, J. C., Calvo, H., & Taud, H.: A micro artificial immune system. (2011)
32. Lü, Z., & Hao, J.: Adaptive tabu search for course timetabling. European Journal of Operational Research, pp. 235–244, doi:10.1016/j.ejor.2008.12.007 (2010)
33. Araujo, L., Cervigón, C.: Algoritmos Evolutivos, Un enfoque práctico. Alfaomega (2009)
34. Villalobos Arias, M., Coello Coello, C. A., & Hernández Lerma, O.: Convergence analysis of a multiobjective artificial immune system algorithm. (2004)