Sistema de generación de horarios para el *student self-scheduling*

Leo Wong
20141479@aloe.ulima.edu.pe / Universidad de Lima. Lima, Perú

Recepción: 28-6-2018 / Aceptación: 20-8-2018

Resumen. En el estado de la literatura, pocos autores han investigado sobre un sistema de recomendación de horarios para estudiantes universitarios; es decir, un sistema que resuelva el problema del *student self-scheduling*. Este problema es una variante del *student sectioning* para la asignación de horarios de tipo *master timetabling*. En esta variante se les brinda la libertad a los estudiantes de confeccionar sus propios horarios; ellos deben realizar varias iteraciones para combinar cursos-sección en búsqueda del mejor horario factible dadas sus preferencias. Se propuso un sistema de generación de horarios para ayudar a los alumnos mediante la sugerencia de varios horarios que consideren sus preferencias. Para ello se propuso el Wong Evolutionary Algorithm (WEA), un algoritmo evolutivo que logró generar varios resultados de calidad en una sola ejecución. Además, el prototipo del sistema fue altamente aceptado por los estudiantes evaluados gracias a la calidad de las soluciones generadas.

Palabras clave: confección de horarios para estudiantes, horarios, sistema de recomendación, selección de cursos

Timetabling system for student self-scheduling

Abstract. In the state of literature, few authors have investigated a schedule recommendation system for university students; that is, a system that solves the student self-scheduling problem. This problem is a variant of the student sectioning for the master timetabling classroom assignment. In this variant, students are offered the freedom to make their own schedules; they must perform several iterations to combine course-sections in the search for the best feasible schedule given their preferences. A system for the generation of schedules was proposed to help the students by presenting several schedules that consider their preferences. For this, the Wong Evolutionary Algorithm (WEA) was proposed, this is an evolutionary algorithm that achieved to produce several quality results in a single run. Besides, the prototype of the system was highly accepted by the evaluated students due to the quality of the generated solutions.

Keywords: student self-scheduling, timetabling, recommender system, course selection

1. Introducción

En muchas universidades con asignación de horarios de tipo *master timetabling* se posee un sistema de soporte a las decisiones que ayuda al personal administrativo a confeccionar horarios de clase de acuerdo a ciertas restricciones e incluso a asignar determinados alumnos a una sección de un determinado curso; en otras palabras, resuelve el *university course timetabling problem* (UCTP) (Carter y Laporte, 1998). Sin embargo, en el estado de la literatura revisado, pocos han investigado respecto a un sistema de generación automática de horarios para ayudar a los alumnos a seleccionar sus horarios en universidades o escuelas que permitan el *student self-scheduling*. El *student self-scheduling*, una variante del proceso de *student scheduling* en la asignación de horarios de tipo *master timetabling*, se basa en permitir a los alumnos confeccionar sus propios horarios que satisfagan sus necesidades mientras cumplen ciertas restricciones y poseen información de los cursos-sección, como el docente y el horario de clase (Kelly, 1979).

Los autores encontrados sobre el tema son Uslu, Ozturan y Uslu (2016), y Fatt *et al*. (2000). Los primeros desarrollaron un sistema que utiliza la programación lineal entera y el filtrado colaborativo basado en datos históricos de las selecciones y calificaciones de los alumnos. Por otro lado, Fatt *et al*. desarrollaron una aplicación que genera todas las posibles combinaciones de los cursos-sección con la implementación de un algoritmo que detecta solapamientos de horarios en un determinado día de la semana. Las aplicaciones mencionadas básicamente se diferencian en la calidad y cantidad de recomendaciones. Mientras que el primero solo se enfoca en la generación de recomendaciones de calidad, el segundo solo se enfoca en la generación de varios resultados. Debido a ello, el primero obtiene solo un resultado y el segundo obtiene varias recomendaciones sin calidad.

Por otro lado, existen temas cercanos que pueden servir de guía para la resolución del problema tratado, de los cuales destacan dos: el *educational timetabling problem* (ETP) y la recomendación de selección de cursos.

La primera cuestión relacionada es el ETP, el cual consiste en la programación de una secuencia de eventos que involucran a profesores y estudiantes en un lapso definido mientras se satisface una serie de restricciones (Schaerf y Di Gaspero, 2001). Uslu *et al*. (2016) se enfocaron en el uso de la programación lineal entera, pudiendo haber sido desarrollado con otros métodos que resuelvan el ETP. Este problema tiene diferentes clasificaciones (Babaei, Karimpour y Hadidi, 2015; Petrovic y Burke, 2004; Schaerf y Di Gaspero, 2001; Qu, Burke, McCollum, Merlot y Lee, 2009; Kristiansen y Stidsen, 2013) en las cuales no contempla explícitamente al *student self-scheduling* como una parte o variante de él, aun pudiendo cumplir con su definición. Por otro lado, existen investigaciones relacionadas al ETP que contemplan las preferencias de profesores y/o de estudiantes como una restricción blanda; al contemplarlas, elevan el nivel de dificultad del problema y, en algunos casos, las convierten en *NP-Complete* (Dostert, Politz y Schmitz, 2016).

Un problema del ETP que no ha tenido muchas investigaciones es el *student sectioning*, el cual también es llamado *student scheduling* por Sahin, Kellegoz y Kokhan (2016). Estos autores realizaron un estudio de caso del individual *student scheduling problem*, un subproblema del *student sectioning*. La diferencia entre este subproblema y el *student self-scheduling* consiste en que el primero toma en cuenta las preferencias de la institución; mientras que el segundo no. Esta diferencia es basada principalmente en la preferencia de distribuir equitativamente los estudiantes a secciones. Los autores propusieron una solución que permite el solapamiento de cursos-sección bajo ciertas condiciones usando la programación lineal entera multiobjetivo y considerando tanto las preferencias del área administrativa como las de los estudiantes.

Los métodos y algoritmos para resolver el ETP normalmente están segmentados bajo las categorías de este. Una clasificación de los métodos y algoritmos para resolver el UCTP, uno de los problemas principales del ETP, es la de Babaei *et al*. (2015). Esta clasificación segmenta a los algoritmos en los siguientes grupos: métodos de investigación de operaciones, métodos metaheurísticos, enfoques multiobjetivos, métodos de *intelligent novel* y el enfoque de *multi-agent systems*. Los principales enfoques para resolver este problema son los métodos de investigación de operaciones y métodos heurísticos o metaheurísticos (Tamayo, Campaña y Expósito, 2007). En los métodos de investigación de operaciones se poseen los métodos de programación lineal entera y programación de restricciones. En los métodos metaheurísticos se posee los métodos basados en una población inicial, como los algoritmos evolutivos (EA) y algoritmos genéticos (GA), colonia de hormigas y algoritmos meméticos, y los métodos basados en una sola solución, como la búsqueda tabú (TS), recocido simulado (SA), búsqueda local, *variable neighborhood search algorithm* y *randomized iterative improvement with composite neighborhood algorithm*.

Además, Babaei *et al*. (2015) realizaron una comparación con 66 diferentes algoritmos. En su estudio se concluye que, generalmente, el algoritmo de colonias de hormigas posee mejor rendimiento que otros algoritmos. Por otro lado, se concluye que, generalmente, el algoritmo genético posee peor rendimiento en todo tipo de base de datos. Sin embargo, actualmente, los algoritmos genéticos y otros relacionados a los EA son ampliamente utilizados en la optimización multiobjetivo (Zhang, Tian y Jin, 2005). Además, existen variantes del algoritmo genético que, si bien pueden convertir al algoritmo a memético o híbrido, resultan ser altamente competitivas frente a otros algoritmos. Un ejemplo de una variante es el algoritmo propuesto por Yang y Jat (2011), el cual un método híbrido basado en un s*teady-state genetic algorithm* guiado por la búsqueda local (LS), que consiguió ser competitivo frente a los métodos de SA, TS y otros basados en LS.

La segunda cuestión es el sistema de recomendación de selección de cursos. La selección de cursos (*course selection*) es un proceso de toma de decisiones de carácter secuencial, donde la selección de un curso modifica los criterios de selección del próximo. Este proceso tiene la

finalidad de escoger un grupo de cursos que conforma el horario para un ciclo (Babad y Tayeb, 2003). Ante la difícil toma de decisiones y debido a que las decisiones son interdependientes e involucran las preferencias de los alumnos, se han diseñado sistemas de recomendación para cursos individuales. Es decir, en estos sistemas, la recomendación de un curso no influye en las otras recomendaciones. Para ello, se han investigado la utilización de sistemas de recomendación, en los cuales destacan los métodos basados en filtrado colaborativo y basados en contenido, y las técnicas de data mining y machine learning (Al-Badarenah y Alsakran, 2016; Unelsrød, 2011). Por ejemplo, Vialardi, Bravo, Shafti y Ortigosa (2009) propusieron un método de recomendación de cursos considerando la demografía, la matrícula de cursos, el número de cursos por semestre, el promedio de notas y el historial de notas de los estudiantes utilizando técnicas de minería de datos; con lo cual consiguieron 77,3 % de precisión al aplicarlo en la Universidad de Lima. Otro caso es el de Uslu *et al*. (2016), quienes propusieron un sistema de recomendación de cursos basado en el filtrado colaborativo.

2. Metodología

La Universidad de Lima fue considerada como universidad modelo para el análisis de las restricciones. Se estimaron como restricciones duras el cruce de horario y la posibilidad de tener solo una sección por curso, y como restricciones blandas, las principales preferencias de los alumnos, que fueron obtenidas mediante un análisis cualitativo en Facebook. Se implementó la generación de horarios basándose en el algoritmo básico de los EA, en este caso, el algoritmo utilizado se puede clasificar como memético. Sin embargo, se decidió mantener la clasificación genérica del algoritmo propuesto como EA y se etiquetó como Wong Evolutionary Algorithm (WEA). Por otro lado, el algoritmo puede clasificarse como un algoritmo de recomendación de tipo *knowledge-based* debido que utiliza el conocimiento puesto en la función objetivo.

Se diseñó el algoritmo de tal forma que intente mantener soluciones completas en cada generación; es decir, el algoritmo llena el cromosoma de genes que agreguen valor a la puntuación y sean compatibles con los otros genes. Para ello, se manejaron las restricciones duras mediante el mantenimiento de las soluciones factibles, lo cual fue implementado con la modificación de los operadores de mutación y de cruzamiento. Ante la posible pérdida de genes por los operadores mencionados, se adicionó el operador de *hill-climbing* (búsqueda local), el cual posee como principal función el llenado del cromosoma con genes compatibles (figura 1).

```
Inicio
Generar población inicial
Mientras condición de parada no se cumpla hacer
        Seleccionar individuos
        Cruzar individuos
        Multar individuos
        Aplicar Hill-Climbing
        Guardar a los mejores
        Reemplazar la población
Fin
```

Figura 1. Algoritmo WEA
Elaboración propia

La función de evaluación (función de *fitness*) fue elaborada a partir de métodos deductivos. Ante la posibilidad de que la puntuación no refleje las preferencias de un alumno en particular, la falta de calibración de pesos y la incertidumbre de distribución de sus soluciones preferidas; se optó por guardar a las soluciones con mayor puntaje de evaluación (incluyendo soluciones dominadas) en el "histórico de mejores". En las siguientes líneas se brindan algunos de los conceptos utilizados por el algoritmo WEA.

- Genotipo: Arreglo de cursos-sección de longitud fija, en el cual los índices corresponden a cada curso. El cromosoma puede tener espacios vacíos (figura 2).

Curso 1	Curso 2	Curso 3	Curso 4	Curso 5	Curso 6	Curso 7
Curso 1 Sección 4	Curso 2 Sección 3		Curso 4 Sección 1	Curso 5 Sección 8	Curso 6 Sección 8	

Figura 2. Ejemplo de la representación del genotipo
Elaboración propia

- **Cruzamiento:** *Uniform crossover* modificado para generar solo *offspring* factibles. Se implementó una alta probabilidad de sesgo para mitigar la pérdida de información (figura 3).
- **Mutación:** *Random resetting* modificado para mantener la factibilidad (figura 3).
- *Hill-climbing*: Búsqueda local estocástica que escala con una función de *roulette wheel* basada en el incremento de la puntuación mediante la inserción de genes en espacios vacíos hasta obtener un máximo local (figura 3).
- **Selección de padres:** *Tournament selection*.
- **Reemplazo de la población:** *Generational replacement model* con elitismo.

- **Histórico de mejores:** Arreglo de longitud fija donde se guardan las mejores soluciones. Al finalizar la ejecución se muestra este arreglo.
- **Condición de parada:** Se implementaron dos criterios para la condición de parada: 1) un límite de iteraciones y 2) un límite de no renovación del histórico de mejores.
- **Función de *fitness*:** La puntuación es calculada mediante el método de suma ponderada $\sum_{k}^{6} \omega_k o_k$. Se modeló cada objetivo O_k mediante la utilización de un indicador relacionado a una preferencia de los alumnos; las preferencias fueron obtenidas en el análisis cualitativo. Los objetivos O_2, O_3, O_4 y O_5 corresponden a objetivos de minimización. Por otro lado, los objetivos O_1 y O_6 corresponden a objetivos de maximización.
  - $\boldsymbol{O_1}$: Suma de *rating* de los profesores considerados en la solución. Corresponde a la preferencia de llevar cursos-secciones con profesores recomendados.
  - $\boldsymbol{O_2}$: Cantidad de horas de hueco presentes en el horario de la solución. Corresponde a la preferencia de tener la menor cantidad posible de horas de hueco.
  - $\boldsymbol{O_3}$: Suma del exceso de dificultad de cada día del horario de la solución. Corresponde a la preferencia de tener una dificultad balanceada en cada día.
  - $\boldsymbol{O_4}$: Penalización total por la asignación de horas no deseadas en el horario de la solución. Corresponde a la preferencia de tener la menor cantidad posible de horas no deseadas.
  - $\boldsymbol{O_5}$: Cantidad de exceso o déficit de cursos con respecto a la cantidad deseada. Corresponde a la preferencia de llevar un determinado número de cursos.
  - $\boldsymbol{O_6}$: Bonificación total por incluir los cursos prioritarios. Corresponde a la preferencia del alumno por determinados cursos

Para la validación se realizó una prueba de desempeño y una prueba de aceptación. En la prueba de desempeño se utilizaron los 20 cursos entre tercer y quinto ciclo de la carrera de ingeniería industrial del ciclo 2018-1 debido a su alta cantidad de secciones por curso. Se utilizó un muestreo estratificado de 30 ejecuciones por cada cantidad de cursos, el coeficiente de correlación lineal de Pearson (r) y un nivel de confianza de 95 %. Por otro lado, para el criterio de parada, se utilizó una máxima cantidad de generaciones en 10 000 y una máxima cantidad de generaciones de no cambio del histórico de mejores en 100, siendo su tamaño de 50. La prueba de aceptación consistió en un *focus group* y una prueba piloto con 4 alumnos de las carreras de ingeniería industrial y de sistemas entre quinto y octavo ciclos.

Cruzamiento

```
Inicio
Obtener padreA, padreB
Inicializar offspringA, offspringB
Obtener arreglo de índices aleatorios del
cromosoma
Para cada i en arreglo hacer
    Si random 0 > PROB_BIAS entonces
        genA ← padreA [i]
        genB ← padreB [i]
    Sino
        genA ← padreB [i]
        genB ← padreA [i]
    Fin Si
    Si offspringA es compatible con genA entonces
        offspringA [i] ← genA
    Fin Si
    Si offspringB es compatible con genB entonces
        offspringB [i] ← genB
    Fin Si
Fin Para
Devolver offspringA, offspringB
Fin
```

Mutación

```
Inicio
Obtener cromosoma
nuevoGen ← obtener gen aleatorio
Para cada gen en cromosomo hacer
    Si gen no es compatible con nuevoGen entonces
        Eliminar gen del cromosoma
    Fin Si
Fin Para
Insertar nuevoGen en el cromosoma
Devolver cromosoma
Fin
```

Hill-Climbing

```
Inicio
Obtener individuo
finLoop ← Falso
    mejoras ← obtener todas las posibles mejoras mediante
     la inserción de genes en espacios vacíos
    Si mejoras.tamaño > 0 entonces
        Mejora ← RouletteWheel(mejoras // basado en el
        incremento
        Individuo ← Mejora
    Sino
        finLoop ← Verdadero
    Fin Si
Fin Mientras
Devolver individuo
Fin
```

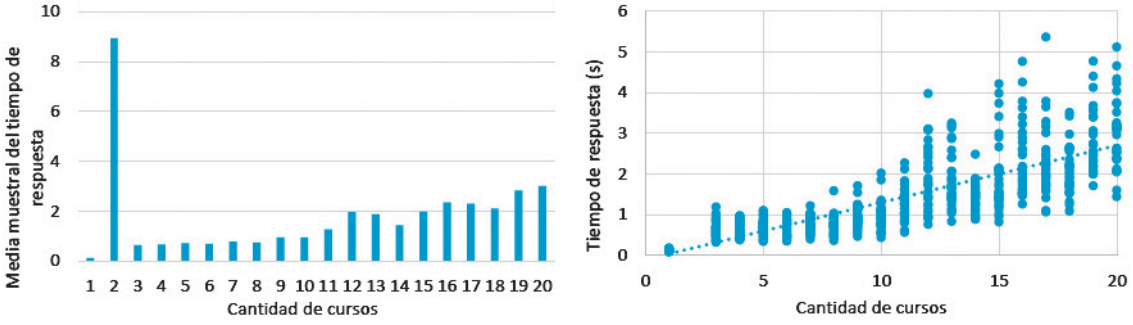
Figura 3. Operadores modificados del WEA
Elaboración propia

3. Resultados

En los resultados se observa un caso excepcional cuando la cantidad de cursos es igual a dos, donde se obtuvo la mayor media muestral del promedio de la cantidad de generaciones (siempre al máximo) y del tiempo de respuesta (8,93 s). Debido a ello, se exceptuará este caso en algunos análisis. Este fenómeno es causado por el comportamiento del criterio de parada con pocos cursos-sección, en el cual se posee la condición de detener la ejecución cuando el histórico de mejores no cambie en 100 generaciones consecutivas. Posiblemente el criterio mencionado es reiniciado principalmente debido al uso de la función de *roulette wheel* para la selección de genes, el cual normalmente selecciona a los mejores del individuo. Al seleccionar los mejores

genes y tener pocas combinaciones, normalmente se produce una solución de puntuación alta que ya está dentro del histórico de mejores, por lo que el histórico de mejores normalmente queda relativamente vacío. Entonces, a veces se produce una solución de baja puntuación que puede insertarse dentro del histórico de mejores, lo cual reinicia la condición de criterio de parada mencionado. De igual forma, esta condición puede reiniciarse debido a los operadores de mutación y de cruzamiento.

Exceptuando el caso mencionado, el coeficiente de correlación lineal entre el tiempo de respuesta y cantidad de cursos es de 0,79 ± 0,03. Debido a ello, se puede afirmar que, generalmente, a medida que se incrementa la cantidad de cursos, el tiempo de respuesta se incrementa (figura 4).



Nota: En el gráfico de la derecha se exceptúa el caso mencionado.

Figura 4. Correlación entre el tiempo de respuesta y la cantidad de cursos
Elaboración propia

La cantidad de cursos posee un coeficiente de correlación lineal de 0,79 con el coeficiente de variación muestral de puntuación; por lo que se puede afirmar que, generalmente, a mayor cantidad de cursos el sistema es menos fiable. Además se posee una cantidad de generaciones relativamente baja. Dada la creciente variación y la relativa poca cantidad de generaciones, posiblemente el algoritmo sea susceptible a una convergencia prematura dado al operador de *hill-climbing* que escala hasta obtener un máximo local (figura 5).

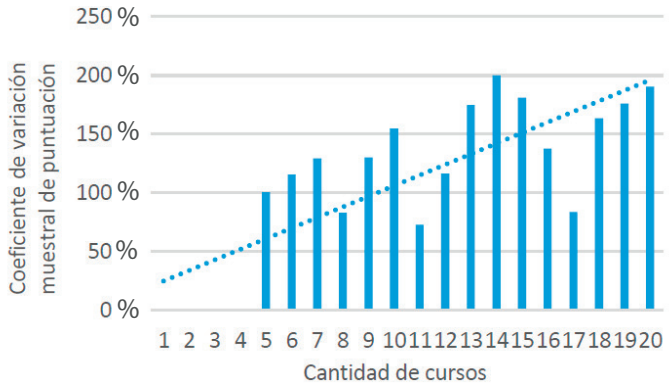


Figura 5. Línea de tendencia entre el coeficiente de variación de puntuación y la cantidad de cursos
Elaboración propia

En la prueba de aceptación, los participantes expresaron opiniones positivas respecto al prototipo en cuanto al desempeño en términos de calidad de soluciones y tiempo de respuesta. Esto es debido a que se consideraron las principales preferencias de ellos, el tiempo de respuesta fue menor a un segundo y ellos encontraron su horario dentro de las primeras recomendaciones o un horario mejor. Por otro lado, los participantes recomendaron una mejora respecto a la usabilidad del programa debido a las complicadas opciones para el manejo de los datos de entrada y los pesos (figura 6).

Nota: Los pesos correspondientes a objetivos de minimización son cambiados a valores negativos durante la ejecución.

Figura 6. Interfaz gráfica del prototipo
Elaboración propia

4. Conclusiones

El Wong Evolutionary Algorithm (WEA) logra producir varias soluciones de calidad con un desempeño aceptable bajo características correspondientes a un ciclo regular y con una cantidad de cursos de alrededor de cinco. No se realizó un análisis de dominancia ante la incertidumbre de la ubicación de la solución preferida de los alumnos. Debido a que el histórico de mejores no suele variar una vez halladas las soluciones que se tiende a generar en una ejecución con una cantidad de cursos-sección moderada o alta, se espera que la ejecución se termine cuando se cumple el segundo criterio de parada. Dada la creciente variación y la relativa poca cantidad de generaciones por cantidad de cursos, posiblemente el algoritmo es susceptible a una convergencia prematura debido al operador de *hill-climbing*, que escala hasta obtener un máximo local, o a que algún parámetro del WEA necesita un ajuste. Los resultados de la prueba de aceptación demostraron la viabilidad del sistema de generación de horarios. Sin embargo, la aceptación puede variar según el nivel, los cursos y secciones a llevar, la carrera del estudiante y sus preferencias respectivas. Debido a ello, se debe realizar un análisis más profundo, considerando un mayor rango y mejor distribución de participantes.

Para trabajos futuros se puede analizar el cambio de parámetros para potenciar el WEA e incorporar nuevos criterios de parada para evitar los casos que no terminan la ejecución en el segundo criterio. Además se puede realizar un análisis de dominancia para estudiar la distribución de los resultados del WEA y los resultados seleccionados por los alumnos. Para ello, posiblemente se puede adaptar el WEA mediante la utilización de técnicas del *multi-objective memetic optimization* (MOMO) u otras del *evolutionary multi-objective optimization* (EMO). Asimismo, se puede investigar la utilización de otra técnica o método de confección de horarios para el *student self-scheduling*. Incluso se pueden incorporar módulos o componentes anexos al sistema de generación de horarios; por ejemplo, un módulo de recomendación de profesores, de cursos-sección y/o de cursos, o un módulo de calibración de pesos, y un módulo de calibración de pesos o generación de la función de evaluación. Por otro lado, se puede investigar la aplicación del WEA en otros problemas de optimización combinatoria. Se cree que se puede aplicar a problemas de grafos no dirigidos con modificaciones mínimas; por ejemplo, el problema del máximo clique.

Referencias

Al-Badarenah, A., y Alsakran, J. (2016). An Automated Recommender System for Course Selection. *International Journal of Advanced Computer Science and Applications 7*(3), pp. 166-175. DOI:10.14569/ijacsa.2016.070323

Babad, E., y Tayeb, A. (2003). Experimental analysis of students' course selection. *British Journal of Educational Psychology 73*(3), 373-393. DOI:10.1348/000709903322275894

Babaei, H., Karimpour, J., y Hadidi, A. (2015). A survey of approaches for university course timetabling problem. *Computers & Industrial Engineering 86*, pp. 43-59. DOI: 10.1016/j.cie.2014.11.010

Carter, M. W., y Laporte, G. (1998). Recent developments in practical course timetabling. En: Burke, E., y Carter, M. (Eds.), *Lecture Notes in Computer Science: Vol. 1408. Practice and Theory of Automated Timetabling II* (pp. 3-19). Toronto: Springer. DOI:10.1007/bfb0055878

Dostert, M., Politz, A., y Schmitz, H. (2016). A complexity analysis and an algorithmic approach to student sectioning in existing timetables. *Journal of Scheduling 19*(3), pp. 285-293. DOI:10.1007/s10951-015-0424-2

Fatt, A. C. M., Kee, C. W., Heong, L. C., Seng, N. H., Har, K. N. S., Ni, P. S. ..., y Prakash, E. C. (2000). Software engineering approach for a timetable generator. En: T*ENCON 2000. Proceedings 3*, pp. 147-150. IEEE. DOI:10.1109/TENCON.2000.892240

Kelly, L. K. (1979). Student Self-Scheduling —Is It Worth the Risk? *NASSP Bulletin (BUL), 63*(424), pp. 84-91. DOI:10.1177/019263657906342414

Kristiansen, S., y Stidsen, T. R. (2013). *A Comprehensive Study of Educational Timetabling, a Survey* (Informe de DTU Management Engineering Report N.o 8.2013). Recuperado de http://orbit.dtu.dk/files/60366101/A_Comprehensive_Study.pdf

Petrovic, S., y Burke, E. K. (2004). University Timetabling. En: Leung, J. (Ed.), *Handbook of Scheduling: Algorithms, Models, and Performance Analysis*. Recuperado de https://pdfs.semanticscholar.org/acf8/6f8bf8bab064a34e9c1c0b258ec0896bf46c.pdf

Qu, R., Burke, E. K., McCollum, B., Merlot, L. T., y Lee, S. Y. (2009). A survey of search methodologies and automated system development for examination timetabling. *Journal of Scheduling, 12*(1), 55-89. DOI:10.1007/s10951-008-0077-5

Sahin, M., Kellegoz, T., y Kokhan, S. (2016). A multi-objective decision making model for class selection problem: a case study. *The Eurasia Proceedings of Educational & Social Sciences 5*, pp. 145-190. Recuperado de http://dergipark.gov.tr/epess/issue/30752/334511

Schaerf, A., y Di Gaspero, L. (2001, September). Local search techniques for educational timetabling problems. En: *Proceedings of the 6th International Symposium on Operational Research (SOR-01)* (pp. 13-23). Preddvor, Slovenia.

Tamayo, S., Campaña, C., y Expósito, C. (2007). Alternativa para el proceso de planificación de horarios docentes de una Universidad. *Ciencias Holguín 13*(4). Recuperado de http://www.ciencias.holguin.cu/index.php/cienciasholguin/article/view/411

Unelsrød, H. F. (2011). *Design and Evaluation of a Recommender System for Course Selection*. Tesis de maestría. Norwegian University of Science and Technology. Recuperado de https://brage.bibsys.no/xmlui/handle/11250/252564

Uslu, S., Ozturan, C., y Uslu, M. F. (2016). Course scheduler and recommendation system for students. En: *2016 IEEE 10th International Conference on Application of Information and Communication Technologies* (pp. 1-6). Bakú, Azerbaiyán: IEEE. DOI:10.1109/ICAICT. 2016.7991812

Vialardi, C., Bravo, J., Shafti, L., y Ortigosa, A. (2009). Recommendation in Higher Education Using Data Mining Techniques. *International Working Group on Educational Data Mining*, pp. 190-199. Córdoba, España. Recuperado de http://www.educationaldatamining.org/EDM2009/uploads/proceedings/vialardi.pdf

Yang, S., y Jat, S. N. (2011). Genetic algorithms with guided and local search strategies for university course timetabling. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 41*(1), pp. 93-106.

Zhang, X., Tian, Y., y Jin, Y. (2015). A knee point-driven evolutionary algorithm for many-objective optimization. *IEEE Transactions on Evolutionary Computation 19*(6), 761-776. DOI:10.1109/TEVC.2014.2378512