OptimEx2: Mejorando un Sistema para la Experimentación con Algoritmos de Optimización

J. Ángel Velázquez-Iturbide
Departamento de Informática y Estadística
Universidad Rey Juan Carlos
Móstoles, Madrid
angel.velazquez@urjc.es

***Abstract*—OptimEx es un sistema de experimentación con algoritmos para comprobar si producen resultados óptimos. Dada su novedad, no existen sistemas similares que sirvan de referencia. Por tanto, es necesario evaluar su usabilidad para ir mejorando su diseño e implementación. Una evaluación previa midió una alta usabilidad pero también permitió identificar aspectos del sistema que debían mejorarse. En esta comunicación se presenta una versión mejorada de OptimEx así como los resultados de una segunda evaluación de usabilidad. El resultado es una mayor usabilidad de OptimEx.**

Keywords— Algoritmos de optimización, experimentación, usabilidad

I. INTRODUCTION

Los algoritmos forman parte del núcleo de la informática, como recogen las principales recomendaciones internacionales de planes de estudio [1]. Su conocimiento conlleva aspectos teóricos y prácticos [2], por lo que es importante que el alumno experimente con las diversas propiedades de los algoritmos: corrección, eficiencia y optimalidad.

La optimalidad [3] puede considerarse una variante de la propiedad de corrección. Parte de la especificación de un problema de optimización declara que la solución no sólo debe ser válida, sino también óptima con respecto a alguna medida. Normalmente, dicha optimalidad se concreta en obtener beneficio máximo o tener coste mínimo. La optimalidad no es una propiedad tan general como la corrección o la eficiencia, pero está omnipresente en muchos de los algoritmos y técnicas de diseño más usados.

La experimentación es una actividad adecuada para el aprendizaje activo de la optimalidad, aunque no se encuentran experiencias en la documentación científica. En nuestra experiencia docente, tuvimos un primer acercamiento didáctico a la optimalidad al experimentar con funciones de selección de algoritmos voraces [4]. En esencia, se planteaba al alumno un problema de optimización y se proporcionaban varias funciones de selección como base para diseñar algoritmos voraces. El alumno debía experimentar para comparar sus resultados y deducir qué funciones de selección podían ser óptimas. La experimentación se realizaba mediante un sistema interactivo denominado GreedEx [5].

Posteriormente se desarrolló el sistema OptimEx (acrónimo de "OPTIMization EXperimentation") [6], similar a GreedEx pero de uso más general. El objetivo de OptimEx es permitir que el alumno experimente con algoritmos desarrollados por él mismo para cualquier problema de optimización, comparando sus resultados y determinando su posible optimalidad o suboptimalidad. Los sistemas GreedEx y OptimEx pueden usarse de forma combinada [7].

Se evaluó la usabilidad de OptimEx [6], con resultados buenos. Los alumnos calificaron con una media superior a 4 su utilidad, facilidad de uso y satisfacción con OptimEx. La evaluación también permitió identificar aspectos del sistema susceptibles de mejora. En esta comunicación se presenta una segunda versión de OptimEx, mejorada a partir de los resultados de dicha evaluación de usabilidad, así como los resultados de una segunda evaluación de usabilidad.

La estructura de la comunicación es la siguiente. La sección II describe OptimEx brevemente y resumen los resultados de la primera evaluación de usabilidad que se utilizaron para su mejora. La sección III muestra las mejoras introducidas en OptimEx y la sección IV, su evaluación de usabilidad. Finalmente, la sección V resume nuestras conclusiones.

II. ANTECEDENTES

En esta sección presentamos las características principales de OptimEx, así como los resultados principales de la evaluación de usabilidad antes mencionada.

A. OptimEx

OptimEx es una aplicación .JAR de Java. Permite experimentar con cualquier algoritmo codificado en Java, siempre que los algoritmos a comparar estén codificados en una sola clase y que su cabecera sólo contenga tipos de datos predefinidos. En este apartado solamente mostramos las características comunes de las dos versiones de OptimEx, dejando para la sección III sus diferencias.

Este trabajo se ha financiado con los proyectos de investigación TIN2015-66731-C2-1-R del Ministerio de Economía y Competitividad, S2013/ICE-2715 de la Comunidad Autónoma de Madrid, y 30VCPIGI15 de la Universidad Rey Juan Carlos.

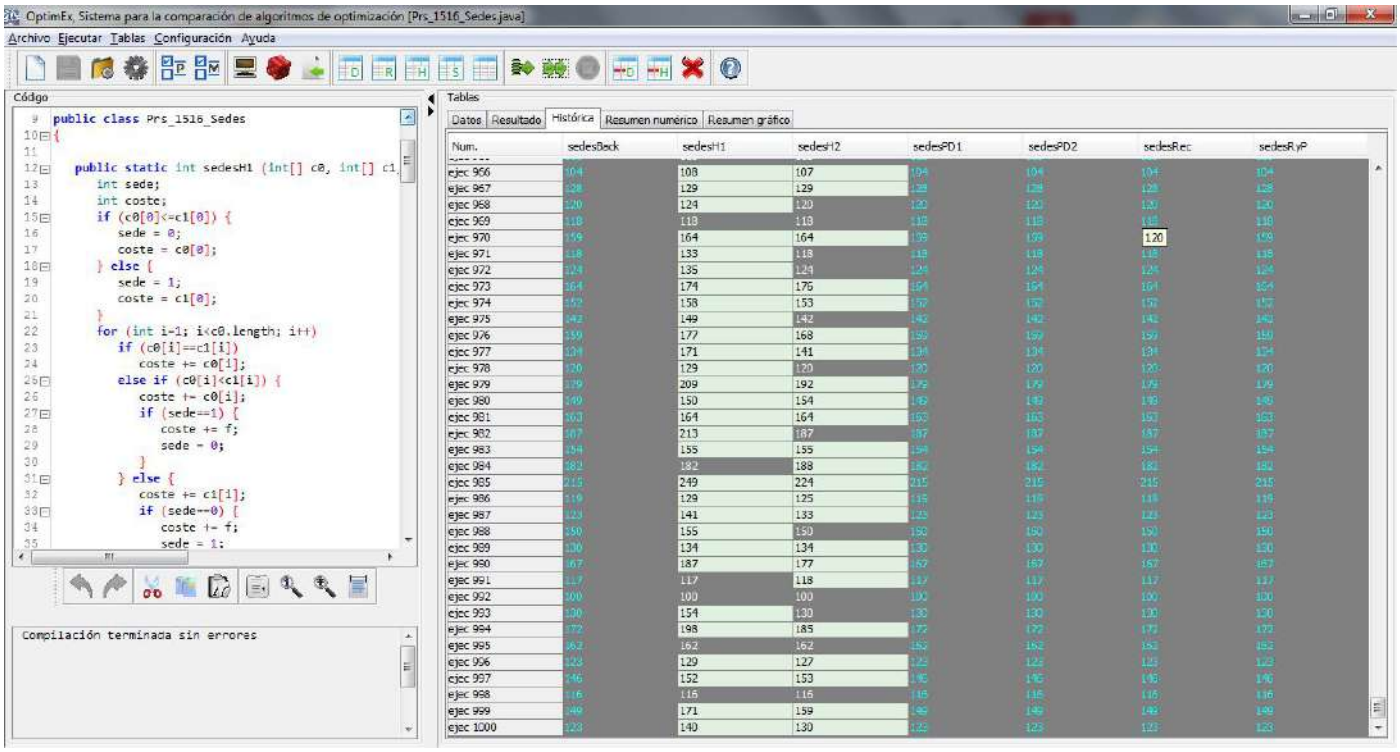


Fig. 1. Interfaz de usuario de OptimEx tras experimentar con siete algoritmos para el problema del plan de sedes de coste mínimo

La Fig. 1 muestra una captura de pantalla de OptimEx. Su estructura es muy sencilla, con dos paneles: código y tablas. El panel de código está situado a la izquierda e incluye un editor, algunos iconos de edición y un área para mensajes del compilador. En la Figura se ha cargado un fichero con siete algoritmos que resuelven el problema de minimización denominado problema del plan de sedes de coste mínimo [8]. Dos algoritmos son heurísticos, y por tanto inexactos, mientras que los otros cinco son exactos (un algoritmo de vuelta atrás, otro de ramificación y poda, y tres algoritmos obtenidos en sucesivas fases de la técnica de programación dinámica).

El panel de tablas está situado a la derecha y permite comparar los resultados de los algoritmos ejecutados, con distintos niveles de detalle. La Figura muestra la tabla histórica, donde cada columna corresponde a un algoritmo y cada fila, a un juego de datos de entrada. En cada fila, se resaltan en gris los resultados menores. Los algoritmos exactos tienen todas las celdas de su columna resaltadas en gris.

El usuario debe seguir los siguientes pasos para comparar los resultados de varios algoritmos:

1. Editar una clase nueva de Java o cargarla de fichero, así como compilarla.
2. Seleccionar el problema al que corresponden los algoritmos a comparar. OptimEx impone una restricción para comparar varios algoritmos entre sí: dado que resuelven el mismo problema, debe ser igual la signatura (tipos de datos de sus parámetros y de su resultado) del método principal de dichos algoritmos. Además, debe indicarse si es un problema de maximización o de minimización.
3. Introducir los juegos de datos de entrada a utilizar para la experimentación.
4. Ejecutar los algoritmos sobre los juegos de datos.
5. Consultar los resultados presentados en varias tablas, sobre todo la tabla histórica y las tablas de resumen. La primera muestra el resultado obtenido por cada algoritmo para cada juego de datos de entrada. Por otro lado, la tabla de resumen numérico muestra mediante estadísticos descriptivos el resultado obtenido por cada algoritmo. Otra pestaña de resumen presenta estos mismos resultados mediante diagramas.

El sistema OptimEx combina una gran sencillez y versatilidad. Es sencillo porque tiene un número reducido de funciones y su diseño se inspira en el de GreedEx (que fue objeto de un alto número de evaluaciones de usabilidad [5]).

La versatilidad de OptimEx permite realizar operaciones de varias formas. Esta versatilidad se encuentra en las funciones de generación de datos de entrada, de ejecución de los algoritmos y de exportación de resultados. Por ejemplo, el usuario puede marcar o no un algoritmo como óptimo. En caso de marcarlo, se tomará como referencia y los resultados de los demás algoritmos se compararán con los de éste. Si no se marca ningún algoritmo como óptimo, el sistema muestra os resultados de las ejecuciones de forma que el usuario pueda deducir qué algoritmo parece óptimo.

B. Evaluación

Se realizó una evaluación de usabilidad en la asignatura optativa “Algoritmos Avanzados”, de cuarto curso del Grado en Ingeniería Informática durante el primer cuatrimestre del

curso académico 2013-14. En la asignatura se habían estudiado varias técnicas de diseño de algoritmos de optimización: técnica voraz, vuelta atrás, ramificación y poda, programación dinámica, y algoritmos heurísticos y aproximados.

La evaluación de usabilidad se realizó durante la sesión de laboratorio de la quinta y última práctica. El objetivo de la práctica era que los alumnos experimentaran con algoritmos exactos e inexactos, la mayor parte de ellos desarrollados en prácticas anteriores.

Veamos con cierto detalle el instrumento de evaluación de usabilidad porque ha sido el mismo en ambas evaluaciones. Al final de la sesión se entregó un cuestionario a los alumnos, cuya cumplimentación era voluntaria. El cuestionario constaba de tres partes:

1. Cinco preguntas con múltiples opciones sobre características generales de OptimEx.
2. Diez preguntas con múltiples opciones sobre elementos concretos de OptimEx.
3. Cinco preguntas abiertas sobre características positivas y negativas de OptimEx.

Las preguntas con múltiples opciones permitían contestar en una escala de Likert, con valores comprendidos entre 1 (muy mal) a 5 (muy bien). Se analizaron con métodos de estadística descriptiva. Las preguntas abiertas se analizaron con métodos cualitativos, identificando categorías que caracterizaran las respuestas.

Veamos un resumen de los resultados obtenidos [6]:

- Los alumnos dieron una puntuación alta a OptimEx, desde una media de 3’81 para su calidad general hasta 4’38 para su utilidad. La facilidad de uso y satisfacción fueron puntuadas con 4 o más puntos.
- Los elementos de OptimEx también recibieron puntuaciones altas, comprendidas entre 3’83 y 4’38. Las tablas fueron los elementos con mayor aceptación, seguidos de la interfaz de usuario y terminando en las funciones de manejo de datos y ejecución de algoritmos.
- Los alumnos identificaron muchos elementos positivos en OptimEx y realizaron numerosas sugerencias de mejora.
- Las características positivas de OptimEx pueden dividirse en genéricas (facilidad de uso y utilidad) y concretas (manejo de datos, interfaz de usuario, soporte a la comparación de resultados, y compilación).
- Los principales elementos cuya mejora se pidió son:
  - Mejorar el soporte de Java para la compilación y ejecución de algoritmos.
  - Incluir diagramas estadísticos.
  - Mejorar el proceso de carga de datos y preparación de la ejecución.
  - Mejorar la funcionalidad del editor.
  - Permitir que los resultados puedan exportarse como ficheros gráficos.
  - Revisar las funciones de compilación y ejecución.

III. OPTIMEX2

Se utilizaron los resultados de la evaluación de usabilidad para introducir mejoras en OptimEx entre junio y septiembre de 2016.

Veamos las principales mejoras introducidas:

- Soporte de Java para la compilación y ejecución. Se eliminaron problemas de instalación y se añadió una opción de configuración de la máquina virtual de Java. También se eliminaron problemas de manejo de directorios con caracteres en blanco.
- Inclusión de diagramas estadísticos. Se dejaron dos pestañas de resumen en el panel de tablas. La tabla de resumen numérico se mantuvo con el mismo formato, salvo en un detalle. Se consideró que eran poco claros para los alumnos los porcentajes de desviación de los resultados producidos por los algoritmos inexactos con respecto a los resultados de los algoritmos exactos (p.ej. el primer algoritmo heurístico H1 del ejemplo se desvía una media de 7’85% en la experimentación mostrada en la Fig. 1). En su lugar, se presenta el valor medio de los resultados producidos (en este ejemplo, al ser un problema de minimización, el valor medio es 107’85%). La Fig. 2 muestra las columnas de tres algoritmos (de vuelta atrás y los dos algoritmos heurísticos H1 y H2); las columnas de los otros cuatro algoritmos son iguales que la primera.

  Las cifras del resumen numérico también se presentan de forma gráfica en una nueva pestaña, de “resumen gráfico”. Los tres primeros porcentajes y los tres últimos se agrupan en gráficas distintas (véase Fig. 3):

Medida	sedesBack	sedesH1	sedesH2
Núm. ejecuciones	1000	1000	1000
% subóptimas	0,00 %	67,30 %	59,80 %
% óptimas	100,00 %	32,70 %	40,20 %
% superóptimas	0,00 %	0,00 %	0,00 %
% desviación media	0,00 %	7,85 %	4,08 %
% desviación máxima superóptima	0,00 %	0,00 %	0,00 %
% desviación máxima subóptima	0,00 %	43,94 %	18,47 %

(a)

Medida	sedesBack	sedesH1	sedesH2
Núm. ejecuciones	1000	1000	1000
% soluciones subóptimas	0,00 %	67,30 %	59,80 %
% soluciones óptimas	100,00 %	32,70 %	40,20 %
% soluciones sobreóptimas	0,00 %	0,00 %	0,00 %
% valor medio no óptimo	0,00 %	107,85 %	104,08 %
% valor subóptimo extremo	0,00 %	143,94 %	118,47 %
% valor sobreóptimo extremo	0,00 %	0,00 %	0,00 %

(b)

Fig. 2. Leyenda y contenido de las filas de la tabla de resumen numérico en (a) OptimEx y (b) OptimEx2

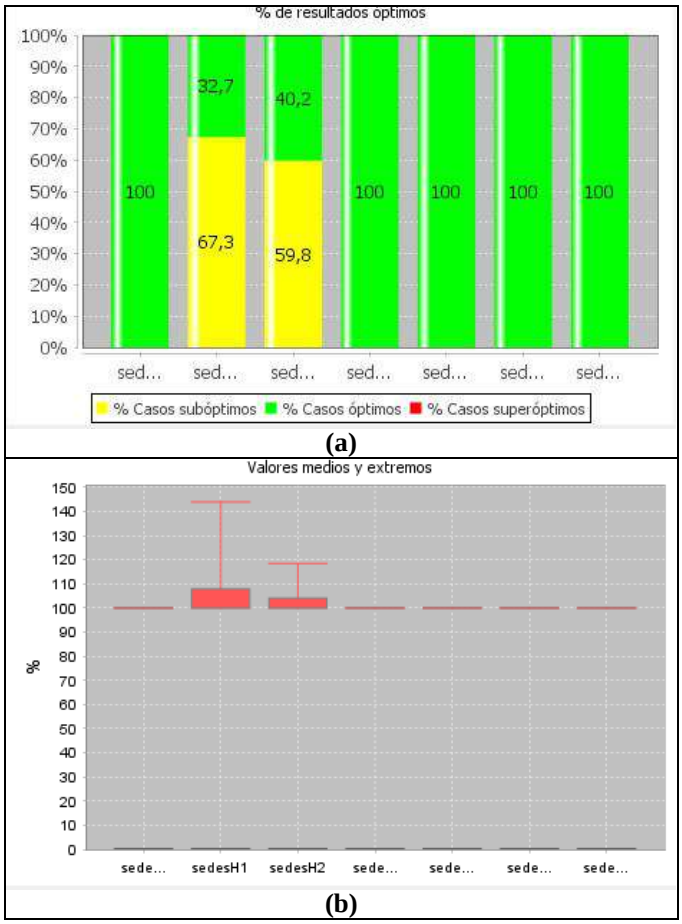


Fig. 3. Diagramas mostrados en la pestaña de resumen gráfico: (a) diagramas de barras apiladas y (b) diagramas de cajas

- Porcentajes de soluciones subóptimas, óptimas y "sobreóptimas". Dadas *n* ejecuciones de varios algoritmos, los algoritmos exactos deberían producir un 100% de soluciones óptimas. Los algoritmos inexactos deberían repartir sus ejecuciones entre óptimas y subóptimas. Las soluciones "sobreóptimas" son soluciones erróneas, ya que corresponden a soluciones "mejores que las óptimas". Esta situación solamente puede darse cuando el usuario marca un algoritmo como exacto y algún algoritmo produce resultados erróneos.

  Obsérvese que deben desglosarse el 100% de soluciones en tres clases. El diagrama estadístico de barras apiladas o seccionadas es probablemente el más sencillo y adecuado para este fin. En la Fig. 3(a) puede verse que hay cinco algoritmos exactos y dos algoritmos heurísticos, inexactos. De ambos, H2 produce un mayor porcentaje de soluciones óptimas (40'2%) que H1 (32'7%).

- Valores de las desviaciones. Dadas *n* ejecuciones, un algoritmo inexacto producirá soluciones no óptimas en un número *m* de ejecuciones ($m \leq n$). Las tres filas últimas de la tabla de resumen muestran, respectivamente, el valor medio de las *m* soluciones no óptimas, el valor "extremo" de las soluciones subóptimas y el valor "extremo" de las soluciones "sobreóptimas". El valor extremo de soluciones subóptimas en un problema de minimización es el mayor coste producido; el valor extremo de soluciones "sobreóptimas" es el menor coste producido.

  No hemos encontrado ningún diagrama que se ajuste claramente a nuestras necesidades, habiendo optado por adaptar los diagramas de cajas (*box plot*). En la Fig. 3(b) se muestra que los 5 algoritmos exactos tienen todos sus resultados concentrados en el 100%. Sin embargo, los algoritmos H1 y H2 presentan resultados subóptimos. Al ser un problema de minimización, estos resultados se sitúan por encima del resultado óptimo (es decir, tienen un coste mayor). La caja de cada algoritmo agrupa los resultados subóptimos producidos dentro de su desviación media. El diagrama extiende su extremo hasta el valor subóptimo que presenta mayor desviación.

  En general, la interpretación de los diagramas es sencilla. Sin embargo, puede complicarse si algún algoritmo produce valores erróneos o si el usuario marca equivocadamente que un algoritmo inexacto es exacto.

  La Fig. 4 muestra la situación que se da, en el ejemplo anterior, cuando el algoritmo H2 es marcado como exacto. La Fig. 4(a) muestra los nuevos valores mostrados en la tabla de resumen numérico, mientras que la Fig. 4(b) muestra el nuevo diagrama de barras apiladas. En la tabla se contabilizan valores "sobreóptimos". El algoritmo H1 incluso presenta casos subóptimos, óptimos y "sobreóptimos". El porcentaje de casos "sobreóptimos" se muestra en el diagrama de barras apiladas en color rojo, para resaltar que estos casos no deben darse.

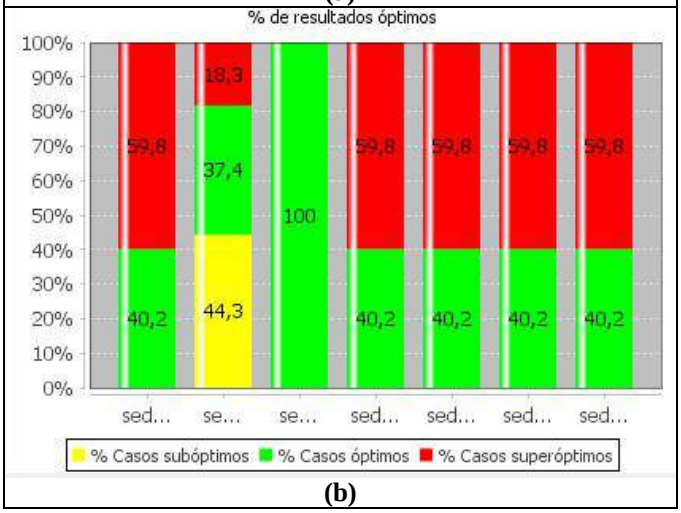
- Mejora del proceso de carga de datos y preparación de la ejecución. En OptimEx se puede disponer de un número alto de casos de prueba (bien generándolos aleatoriamente bien cargándolos de fichero). Un diálogo adicional permite seleccionar qué datos se usarán en la ejecución, ya que se podría desear la ejecución de sólo un subconjunto de los casos.

  Asimismo, la ejecución exige que se hayan especificado las condiciones del experimento, es decir: problema a resolver (signatura de los métodos y si el problema es de maximización o de minimización), métodos a ejecutar (entre los que satisfacen la signatura seleccionada) y datos a utilizar.

  La versión primera de OptimEx era algo confusa sobre el orden de realización de estas operaciones y a veces, engorrosa o propensa a errores. En la versión actual, se han separado completamente los tres pasos, obligando al usuario a especificar todos los detalles para evitar errores por omisión. Por defecto, se seleccionan todos los métodos y datos.

Núm. ejecuciones	1000	1000	1000
% soluciones subóptimas	0,00 %	44,30 %	0,00 %
% soluciones óptimas	40,20 %	37,40 %	100,00 %
% soluciones sobreóptimas	59,80 %	18,30 %	0,00 %
% valor medio no óptimo	96,17 %		0,00 %
% valor subóptimo extremo	0,00 %	136,49 %	0,00 %
% valor sobreóptimo extremo	15,59 %	12,32 %	0,00 %

(a)



(b)

Fig. 4. Presentación de una situación erronea en: (a) las tres primeras columnas de la tabla de resumen y (b) el diagrama de barras apiladas

- Editor de código. Se cambió el editor anterior por otro multicolor y con la posibilidad de ampliar/colapsar algunos elementos conforme a la sintaxis de Java (p.ej. una declaración de método). En su parte inferir se añadieron algunos iconos para funciones de edición corrientes (deshacer, copiar, pegar, etc.).
- Exportación de tablas como fichero gráfico. Se amplió la exportación de tablas (en formato Excel) para permitir formatos gráficos. También se añadió una función para exportar todas las tablas simultáneamente.
- Funciones de compilación. Anteriormente se capturaban los mensajes de error de compilación que generaba la máquina virtual de Java y se mostraban en una ventana. Se manipuló cada mensaje de error, descomponiéndolo de forma que resultaba más fácil de comprender. Además, se integró en la aplicación, en una zona situada debajo del editor (véase Fig. 1). La Fig. 5 muestra la diferencia entre los mensajes generados en ambas versiones de OptimEx.
- Interfaz de usuario. Se cambió el estilo de la barra de iconos y se cambiaron algunos. También se adaptaron los diálogos de algunas funciones para que recordaran las últimas opciones introducidas por el usuario.
- Ayuda interactiva. Se actualizó y completó.

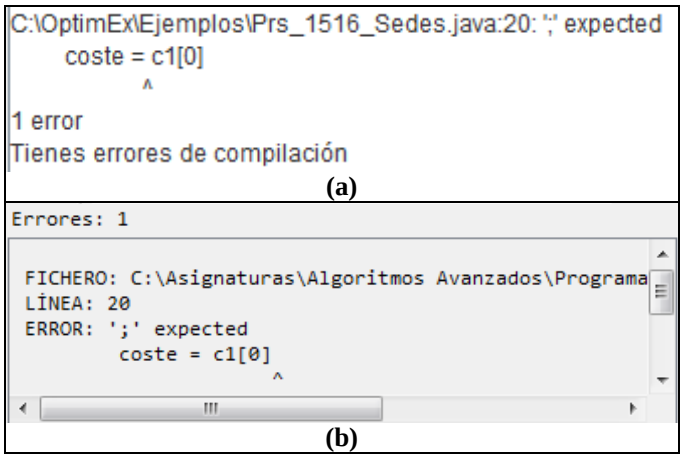


Fig. 5. Mensaje de error de compilación mostrado por (a) OptimEx y (b) OptimEx2

IV. Evaluación de Usabilidad

Se realizó una segunda evaluación de usabilidad durante el primer cuatrimestre del curso académico 2016-17. Las condiciones fueron similares, salvo que se realizó al final de la sesión de laboratorio de la segunda práctica, dedicada a algoritmos heurísticos. Se recogieron 21 cuestionarios.

Veamos primero las respuestas a las preguntas de múltiple opción sobre características generales de OptimEx. Los resultados se muestran en la Tabla I, ordenados de mayor a menor puntuación. Obsérvese que los resultados son muy altos para todas las propiedades. La utilidad y la facilidad de uso se puntúan por encima de 4’5, mientras que la satisfacción y calidad general obtienen más de 4 puntos.

TABLE I. Resultados de las preguntas generales

Pregunta	Media	Desviación típica
Utilidad para reforzar la seguridad en que realmente son exactos los algoritmos que, según la teoría, deberían serlo	4,81	0,40
Fácil de usar	4,71	0,46
Utilidad para analizar porcentaje de resultados óptimos de los algoritmos inexactos	4,67	0,48
Calidad general	4,24	0,54
En conjunto te ha gustado OptimEx	4,24	0,54

Otro conjunto de doce preguntas pedía valorar la calidad de elementos concretos de OptimEx. Los resultados se muestran en la Tabla II, de nuevo ordenados en orden decreciente de puntuación.

Los resultados son muy altos en general. Las tablas y la estructura del menú principal se valoran muy alto, alrededor de 4’5. Seis funciones relacionadas con la preparación de los datos de entrada y la ejecución de los algoritmos, más la exportación en ficheros, obtienen puntuaciones situadas alrededor de 4’25. El editor y los iconos reciben las puntuaciones más bajas (4 y 3’62, respectivamente).

TABLE II. RESULTADOS DE LAS PREGUNTAS SOBRE LA CALIDAD DE SUS ELEMENTOS

Elemento	Media	Desviación típica
Tabla histórica	4,57	0,60
Tabla de resumen gráfico	4,52	0,60
Tabla de resumen numérico	4,48	0,51
Estructura del menú principal	4,43	0,51
Funciones de manejo de datos de entrada	4,33	0,66
Funciones de ejecución del algoritmo	4,29	0,64
Diálogo de selección de signatura	4,24	0,94
Diálogo de selección de métodos	4,24	0,89
Exportación de imágenes y tablas	4,19	0,93
En general, el soporte al proceso de experimentación	4,19	0,51
Editor	4,00	0,45
Iconos	3,62	0,74

Por último, se recibieron un total de 69 respuestas a las cinco preguntas abiertas. Sin embargo, muchas respuestas no aportaban información útil, mientras que otras agrupaban varias opiniones o incluso deberían haberse escrito como respuesta a otras preguntas. Por tanto, se eliminaron las respuestas sin utilidad, se partieron las respuestas compuestas en respuestas simples y se recatalogaron en tres categorías:

- Aspectos positivos: 21 respuestas simples.
- Partes poco útiles, que podrían suprimirse de OptimEx: 7 respuestas simples.
- Aspectos a mejorar: 30 respuestas simples.

Los resultados sobre aspectos positivos se muestran en la Tabla III. Las hemos agrupado en 3 categorías genéricas y 2 correspondientes a elementos concretos de OptimEx. Lo más valorado fue la utilidad de OptimEx y las tablas.

TABLE III. ASPECTOS POSITIVOS

Aspecto	Número de respuestas simples
Utilidad para comparar algoritmos de optimización	9
Tablas	6
Facilidad de uso	2
Manejo de datos	2
Otros	2

Las respuestas sobre elementos a suprimir se muestran en la Tabla IV.

TABLE IV. ELEMENTOS A SUPRIMIR

Elemento	Número de respuestas simples
Algunas tablas	3
Opciones de exportar	2
Iconos de edición	2

Por último, las sugerencias de mejora se muestran en la Tabla V. El aspecto más comentado es la compilación, bien para pedir mayor flexibilidad (p.ej. admitir métodos de varias clases de Java), bien para informar de algunos errores detectados. Los comentarios sobre tablas señalan problemas de legibilidad, dificultad de interpretación de las tablas de resumen o algunos errores. Se comenta que el usuario no conoce *a priori* los pasos a seguir en un experimento, así como algunas sugerencias de mejora. También se señala que algunos iconos son poco descriptivos.

TABLE V. ASPECTOS A MEJORAR

Aspecto	Número de respuestas simples
Compilación	10
Tablas	5
Proceso de experimentación	5
Iconos	3
Ayuda interactiva	2
Funciones de exportación	2
Configuración	1
Funciones de importación	1
“Inteligencia”	1

V. CONCLUSIONES

Hemos presentado las mejoras introducidas en el sistema OptimEx, así como una evaluación de usabilidad. El resultado general es de alta usabilidad. Aun así, algunos elementos son susceptibles de más mejoras, sobre todo la compilación, las tablas, el proceso de experimentación y la interfaz de usuario.

A pesar de posibles mejoras futuras, el sistema es bastante estable y usable. Podrían encontrarse oportunidades más interesantes de mejora futura en incluir soporte para comparar tiempos de ejecución de los algoritmos. De esta forma, la comparación de algoritmos sería más completa. Por ejemplo, podría comprobarse que un algoritmo de vuelta atrás y uno de ramificación y poda son exactos, pero que el primero es menos eficiente en tiempo que el segundo.

REFERENCIAS

[1] ACM, and IEEE Computer Society, The Joint Task Force on Computing Curricula: Computer Science Curricula 2013. http://www.acm.org/education/CS2013-final-report.pdf, 2013.

[2] P.J. Denning, D.E. Comer, D. Gries, M.C. Mulder, A.B. Tucker, A.J. Turner y P.R. Young, "Computing as a discipline." Communications of the ACM, vol. 32, no. 1, pp. 9-23, January 1989.

[3] J.Á. Velázquez-Iturbide, O. Debdi y M. Paredes-Velasco, “A review of teaching and learning through practice of optimization algorithms,” en Innovative Teaching Strategies and New Learning Paradigms in Computer Programming, R. Queirós, Ed., IGI Gobal, 2015, pp. 65-87.

[4] J.Á. Velázquez-Iturbide, “An experimental method for the active learning of greedy algorithms,” ACM Transactions on Computing Education, vol. 13, no. 4, article 18, 2013.

[5] J.Á. Velázquez-Iturbide, O. Debdi, N. Esteban-Sánchez y C. Pizarro, "GreedEx: A visualization tool for experimentation and discovery learning of greedy algorithms," IEEE Transactions on Learning Technologies, vol. 6, no. 2, pp. 130-143, Abril-Junio 2013.

[6] J.Á. Velázquez-Iturbide, “Design and evaluation of OptimEx, an experimentation system for optimization algorithms,” en ICT in Education - Multiple and Inclusive Perspectives, M.J. Marcelino, A.J. Mendes y C. Azevedo Gomes, Eds., Springer, 2016, pp. 51-68.

[7] J.Á. Velázquez-Iturbide, “GreedEx and OptimEx: Two tools to experiment with optimization algorithms,” International Journal of Engineering Education, vol. 32, no. 3A, pp. 1.097-1.106, 2016.

[8] J. Kleinberg y É. Tardos, Algorithm Design. Cambridge, MA: Addison-Wesley, 2006.

[9] J.Á. Velázquez-Iturbide, Una segunda evaluación de usabilidad de OptimEx. Serie de Informes Técnicos DLSI1-URJC, no. 2016-02, Departamento de Lenguajes y Sistemas Informáticos I, Universidad Rey Juan Carlos, 2016.