**J. Ángel Velázquez Iturbide**

# Interacción de la Experimentación con Teoría y Desarrollo en una Asignatura de Algoritmos

Número 2020-01

**Serie de Informes Técnicos DLSI1-URJC**
**ISSN 1988-8074**
**Departamento de Lenguajes y Sistemas Informáticos I**
**Universidad Rey Juan Carlos**

# Índice

# Interacción de la Experimentación con Teoría y Desarrollo en una Asignatura de Algoritmos

J. Ángel Velázquez Iturbide

Departamento de Informática y Estadística, Universidad Rey Juan Carlos,
C/ Tulipán s/n, 28933, Móstoles, Madrid
angel.velazquez@urjc.es

**Resumen.** El diseño de algoritmos se basa en métodos ingenieriles y su análisis, en la mayor parte de los casos, en métodos matemáticos y demostraciones formales. La experimentación con los propios algoritmos suele limitarse a comprobar que sus tiempos de ejecución cumplen las predicciones de la teoría. En este informe técnico analizamos los resultados de intentar una mayor integración de la experimentación con la teoría y la ingeniería. Como material usamos un método experimental basado en comparar la optimalidad de distintos algoritmos desarrollados que resuelven un mismo problema de optimización (en general desarrollados con técnicas de diseño distintas). Se analizan en cuatro prácticas planteadas sobre dos problemas de optimización las siguientes cuestiones: relación entre precondición y generación de aleatoria de juegos de datos, uso de la experimentación para obtener contraejemplos "sencillos" de optimalidad, uso de la experimentación como método de prueba de algoritmos y comprobación de las predicciones teóricas sobre la exactitud de las distintas técnicas de diseño. El informe incluye, como apéndices, los enunciados de las prácticas.

**Palabras clave:** Algoritmos, teoría, experimentación, precondición, generación aleatoria de datos, contraejemplos, pruebas.

## 1 Introducción

El diseño de algoritmos se basa principales en unos métodos ingenieriles denominados técnicas de diseño de algoritmos. Asimismo, su análisis se basa en la mayor parte de los casos en métodos matemáticos y demostraciones formales. La experimentación con los propios algoritmos suele limitarse a comprobar que sus tiempos de ejecución cumplen las predicciones de la teoría.

En este informe técnico analizamos los resultados de intentar una mayor integración de la experimentación con la teoría y la ingeniería. Como material usamos un método experimental basado en comparar la optimalidad de distintos algoritmos desarrollados que resuelven un mismo problema de optimización (en general desarrollados con técnicas de diseño distintas). Aunque el método experimental original se limitaba a distintos criterios voraces [1][2], puede extenderse a algoritmos de optimización desarrollados con otras técnicas de diseño. La experimentación es

soportado en sus detalles por el sistema OptimEx, tanto para el criterio de optimalidad [3] como de eficiencia en tiempo [4]. El estudio complementa otros estudios anteriores, aunque centrados en dificultades y malas concepciones [5][6][7], así como actitudes [6].

La estructura del informe es la siguiente. El apartado 2 describe el contexto docente y los materiales usados para la evaluación. Los apartados 3, 4 y 5 presentan, respectivamente, los resultados de analizar la interacción entre teoría y experimentación en cuatro cuestiones diferentes: relación entre problema y algoritmo, contraejemplos de optimalidad, e interacción entre optimalidad y corrección (con el objetivo doble de pruebas de algoritmos y comprobación de las predicciones teóricas sobre la exactitud de las distintas técnicas de diseño). En el sexto apartado se presentan las conclusiones de los análisis realizados. Finalmente, un apéndice recoge información detallada de la evaluación, tanto el cuestionario utilizado como las respuestas de los alumnos.

## 2 La Asignatura

Esta evaluación se realiza en el contexto de la asignatura optativa "Algoritmos Avanzados", de cuarto curso del Grado en Ingeniería Informática (GII) de la Universidad Rey Juan Carlos. La asignatura también se ofrece a los alumnos del Grado en Ingeniería de Computadores (GIC) y a los alumnos del grado doble GII-GIC. Todos estos alumnos asisten al mismo grupo presencial, por lo que no los desglosamos en el análisis presentado en este informe. Asimismo, se ofrece a los alumnos del grupo de GII online, en extinción. La evaluación se ha realizado durante el periodo septiembre-diciembre del curso académico 2019-20.

### 2.1 Prácticas de la Asignatura

El temario de la asignatura incluye varias técnicas algorítmicas para la resolución de problemas de optimización (técnica voraz, algoritmos heurísticos y aproximados, vuelta atrás, ramificación y poda, y programación dinámica), así como algoritmos probabilistas (de ámbito más general).

Se propuso la realización de 6 prácticas al grupo presencial y sólo las 5 primeras al grupo online:

1. Técnica voraz. Dado un problema de planificación de tareas [8, págs. 392-393] y una descripción de un criterio voraz, se pide desarrollar un algoritmo voraz que no asuma que las actividades están inicialmente ordenadas.
2. Algoritmos heurísticos. Se plantea un problema de planificación de trabajos con beneficio máximo [9, pág. 313] y se esboza un algoritmo heurístico. Se pide especificar el problema, programar al menos dos algoritmos heurísticos (el esbozado y otros a discreción del alumno) y comprobar su optimalidad con OptimEx.
3. Técnicas de búsqueda. La práctica se realizó con dos entregas:

   a. Dado el mismo problema de planificación de tareas, se pide desarrollar un algoritmo de vuelta atrás y comparar la optimalidad de este algoritmo y de los algoritmos heurísticos desarrollados en la práctica 2.
   b. Se pide desarrollar un algoritmo de ramificación y poda, y comparar la optimalidad y la eficiencia de este algoritmo y de los anteriormente desarrollados.
4. Eliminación de recursividad múltiple redundante. Se da una función recursiva $f$ y se pide analizar su redundancia, eliminarla mediante memorización y tabulación, y analizar la complejidad en tiempo y memoria de ambos algoritmos.
5. Técnica de programación dinámica. Dado el mismo problema de planificación de tareas de las prácticas 2 y 3, se pide desarrollar un algoritmo recursivo, convertirlo en un algoritmo de programación dinámica (tabulador) y comparar la optimalidad de ambos algoritmos y de los anteriormente desarrollados.
6. Algoritmos probabilistas. Dada la versión de minimización del problema de llenado de cajas [10, págs. 534-535], se esboza un algoritmo probabilista y se pide comparar su optimalidad con la de otros algoritmos disponibles en el Aula Virtual para el mismo problema.

Las prácticas podían realizarse individualmente o en pareja, aunque no distinguimos este aspecto en el informe ya que no tiene relevancia para el estudio. Cada grupo de prácticas debía entregar un informe, siguiendo un índice especificado en el enunciado de cada práctica.

Una vez que una práctica estaba corregida, en el plazo de varios días se publicaba su calificación junto con comentarios a los alumnos sobre su entrega. Los alumnos podían aprovechar estos comentarios para volver a entregarla en un breve plazo, estando avisados de que podían subir su calificación hasta un 8 cómo máximo. Por esta razón, algunos grupos han realizado dos entregas de algunas prácticas.

### 2.2 Experimentación, Teoría e Ingeniería en las Prácticas

Las prácticas 1 y 4 tienen como objetivo ejercitarse en el uso correcto de alguna técnica de codificación, complementado con el análisis de complejidad de los nuevos algoritmos desarrollados. Son prácticas de la categoría Aplicar de la taxonomía revisada de Bloom, sin apenas margen para ninguna innovación. Sin embargo, las prácticas 2, 3, 5 y 6 tienen como objetivo ejercitarse en el uso de alguna técnica de diseño de algoritmos. Son prácticas de la categoría Crear, que permiten cierto margen a la innovación y la experimentación para comparar los resultados calculados por los distintos algoritmos y determinar su exactitud. Por tanto, utilizamos estas prácticas como material para nuestro análisis. Incluimos sus enunciados en el Apéndice A.

También analizaremos algunos usos "convencionales" de la teoría o la experimentación. La práctica 5 pide el análisis de complejidad en espacio y en tiempo del algoritmo de programación dinámica desarrollado, así como que se midan tiempos de ejecución de los algoritmos para comprobar las predicciones teóricas. Las prácticas 2, 3 y 6 piden comparar la optimalidad de los algoritmos heurísticos, de búsqueda y probabilistas desarrollados, respectivamente, que no pueden determinarse teóricamente y son desconocidos.

Si analizamos los apartados pedidos en las cuatro prácticas, podemos distinguir varias cuestiones teóricas o ingenieriles relacionadas con la experimentación:

- Problema y algoritmo. En la práctica 2 se pide una especificación del problema, diferenciando sus partes. La precondición se utiliza en las cuatro prácticas para fijar los rangos de valores en la generación aleatoria de datos de entrada.
- Contraejemplos de optimalidad. En la práctica 2 se pide que, aprovechando los contraejemplos obtenidos experimentalmente de la optimalidad de los algoritmos desarrollados, se analicen y se presenten contraejemplos más sencillos que ilustren casos en los que los algoritmos calculan valores subóptimos.
- Corrección de los algoritmos. Los experimentos de las prácticas 3a, 3b, 5 y 6 implican algoritmos (teóricamente) exactos. Por tanto, los resultados experimentales deben coincidir con la predicción de la teoría. En caso contrario, permiten detectar errores en algunos de los algoritmos comparados, que habrá que depurar.
- Predicciones de optimalidad. Es el uso más convencional y común de la experimentación con la eficiencia. Los experimentos permiten comprobar que las predicciones de la teoría quedan corroboradas experimentalmente. Esta cuestión y la anterior son la misma pero con distintos énfasis. En el punto anterior analizamos el uso por los alumnos de los datos experimentales para una fase ingenieril, la prueba. En el punto actual, analizamos las discrepancias presentadas entre los datos experimentales y la teoría.

La corrección o la optimalidad son criterios que no admiten el grado comparativo: su propiedad se cumple o no se cumple. Sin embargo, la eficiencia admite la comparación, por lo que su análisis es más complejo. Por esta razón, dejamos fuera del ámbito de este informe el análisis de las predicciones de eficiencia, que será objeto de análisis en un informe posterior.

## 3 Problema y Algoritmo

En la práctica 2 hay dos detalles que implican la teoría y la experimentación:

- Para comparar la optimalidad de distintos algoritmos que resuelven el mismo problema mediante los mismos datos de entrada, es necesario que todos tengan la misma cabecera. La cabecera se deduce de la especificación del problema.
- Para generar aleatoriamente los datos de entrada a usar en la comparación experimental, deben especificarse rangos de valores. Estos valores deben ser compatibles con la precondición del problema. Recordemos que el problema de planificación de trabajos con beneficio máximo [9, pág. 313] se refiere a beneficios por trabajos realizados, donde sólo tiene sentido que sean mayores que cero.

El primer punto no debía presentar problemas en la práctica 2 porque se exigía que los algoritmos tuvieran una cabecera concreta, que s proporcionaba.

En esta sección abordamos el segundo punto. De hecho, dos malas concepciones comunes en los alumnos [7] son la distinción entre especificación y algoritmo y entre las distintas partes de una especificación. Retomamos la segunda cuestión y su relación con la generación de datos aleatorios para realizar pruebas. El análisis surge porque se detectó anecdóticamente que algunos grupos utilizaban rangos para la generación de datos que incluían la posibilidad de un beneficio igual a cero.

La precondición del problema de planificación de trabajos con beneficio máximo [9, pág. 313] es que los vectores de ingresos por tareas de alta y de baja tensión deben ser iguales, y que sus ingresos deben ser mayores que cero. La matización de que los ingresos por las tareas de alta tensión suelen ser mayores que por tareas de baja tensión, pero no siempre, es difícil de recoger en la precondición. Sin embargo, resulta más sencillo especificarlo al generar datos de entrada: los rangos de valores de las tareas de alta tensión deben ser mayores que los de baja tensión pero solapados, de forma que pueda darse la situación inversa. Por ejemplo, podría especificarse que el rango de valores para tareas de alta tensión sea 20..40 y para las tareas de baja tensión, 10..30.

De la práctica 2, se recogieron 34 informes de prácticas en la primera entrega, 2 del grupo online y 32 del grupo presencial. Asimismo, se recogieron 9 informes en la segunda entrega, 1 del grupo online y 8 del grupo presencial.

En la Tabla 5 incluimos las características de las entregas recibidas. De izquierda a derecha, se presenta el número de cada grupo de prácticas, el número de entrega de la práctica (primera o segunda), una breve descripción de las partes de la precondición propuestas por el grupo, el rango de valores utilizado para la generación aleatoria de valores de entrada, y una codificación de la entrega del grupo.

Esta codificación de cada entrega consta de tres códigos:

- Precondición: C = completa, P = parcial, – = vacía (no ha incluido especificación).
- Coherencia entre la precondición y el rango de valores generados: C = coherente, P = parcialmente coherente (permite la generación de beneficios nulos, mientras que la precondición no lo admite), , – = desconocida (no ha incluido especificación o ésta solamente especifica que la longitud de los dos vectores debe ser igual).
- Coherencia entre el enunciado y el rango de valores generados: C = coherente, P = parcialmente coherente (los rangos de los beneficios de los trabajos de alta tensión no son mayores que los rangos para trabajos de baja tensión).

**Tabla 1.** Informes de prácticas presentados para la práctica 2 – precondición y valores generados

| Número de grupo | Número de entrega | Precondición | Rango de valores generados | Categoría |
|---|---|---|---|---|
| 1. | 1 | Longitudes iguales<br>Beneficios positivos | 20-40, 10-30 | C, C, D |
| 2. | 1 | Longitudes iguales<br>Beneficios positivos | 1-30, 10-15 | C, C, d |
| | 2 | Longitudes iguales<br>Beneficios positivos | 15-30, 10-15 | C, C, D |

| | | | | |
|---|---|---|---|---|
| 3. | 1 | Beneficios positivos | 0-30, 0-30 | P, P, G |
| 4. | 1 | Beneficios positivos | 10-20, 1-10 | P, C, D |
| 5. | 1 | Longitudes iguales | 0-100, 0-100 | P, –, G |
| 6. | 1 | Beneficios positivos | 10-15, 10-15 | P, C, G |
| | 2 | Beneficios positivos | 10-20, 5-10 | P, C, D |
| 7. | 1 | Longitudes iguales<br>Beneficios positivos | 0-20, 0-20 | C, P, G |
| 8. | 1 | Longitudes iguales<br>Beneficios positivos[1] | 10-30, 10-15 | C, C, D |
| 9. | 1 | Longitudes iguales | 20-50, 0-30 | P, –, D |
| 10. | 1 | Longitudes iguales[2]<br>Beneficios positivos | 10-15, 5-10<br>25-40, 15-25<br>35-60, 10-25 | C, C, D |
| 11. | 1 | Longitudes iguales | 0-10.000, 0-10.000 | P, –, G |
| 12. | 1 | Longitudes iguales | 0-40, 0-40 | P, –, G |
| | 2 | Longitudes iguales<br>Beneficios “positivos” | 0-40, 0-40 | C, P, G |
| 13. | 1 | – | 10-30, 10-30 | –, –, G |
| 14. | 1 | Longitudes iguales<br>Beneficios no negativos | 0-30, 0-30 | C, C, G |
| 15. | 1 | Longitudes iguales | 20-50, 0-20 | P, –, D |
| 16. | 1 | Longitudes iguales | 0-30, 0-30 | P, –, G |
| 17. | 1 | – | 0-100, 0-100<br>0-100, 0-100 | –, –, G |
| 18. | 1 | Longitudes iguales<br>Beneficios positivos | 10-30, 5-20 | C, C, D |
| 19. | 1 | Longitudes iguales | 1-30, 1-20 | P, –, D |
| 20. | 1 | Longitudes iguales<br>Beneficios “positivos” | 0-70, 0-70 | C, P, G |
| 21. | 1 | Longitudes iguales<br>Beneficios positivos | 1-30, 1-30 | C, C, G |
| 22. | 1 | – | – | – |
| 23. | 1 | Longitudes iguales<br>Beneficios no negativos | 1-10.000 – 1-10.000 | C, C, G |
| 24. | 1 | Beneficios positivos | 1-30, 1-30 | P, C, G |
| 25. | 1 | Longitudes iguales<br>Beneficios positivos | 1-15, 1-15 | C, C, G |
| 26. | 1 | Longitudes iguales | 1-30, 1-30 | P, –, G |
| 27. | 1 | – | 20-30, 0-30 | –, –, D |
| | 2 | – | 20-30, 0-30 | –, –, D |
| 28. | 1 | – | 12-30, 7-20 | –, –, D |
| | 2 | Longitudes iguales<br>Beneficios positivos | 12-30, 7-20 | C, C, D |
| 29. | 1 | – | 0-30, 0-30 | –, –, G |
| | 2 | Beneficios positivos | 1-30, 1-30 | P, C, G |

[1] Incoherencia entre dos afirmaciones: “el entero de cada posición del array es mayor que 0” y “beneficio semanal >= 0”. Dados los rangos de generación de datos, podríamos suponer que es un despiste y tomar como válida su primera afirmación.

[2] Expresan la restricción de que los dos vectores deben tener la misma longitud pero no la incluyen en la precondición.

| | | | | |
|---|---|---|---|---|
| 30. | 1 | Longitudes iguales | 0-50, 0-50 | P, –, G |
| | 2 | Beneficios positivos | 20-50, 20-50 | P, C, G |
| 31. | 1 | – | 15-30, 1-15 | –, –, D |
| 32. | 1 | Longitudes iguales<br>Beneficios enteros[3] | 1-30, 1-30 | P, C, G[4] |
| | 2 | – | 1-30, 1-30 | –, –, G |
| 33. | 1 | – | 1-30, 1-30<br>1-20. 1-20 | –, –, G |
| | 2 | – | 1-30, 1-30<br>1-20. 1-20 | –, –, G |
| 34. | 1 | Longitudes iguales | 10-20, 5-15 | P, –, D |

Podemos plantear tres cuestiones relacionadas con la especificación del problema y la generación aleatoria de datos de entrada.

En primer lugar, podemos comprobar la calidad de las precondiciones especificadas por los alumnos. La Tabla 2 contiene un resumen de la información sobre la precondición recogida en la Tabla 1 (última columna, primer código). Puede observarse que se distinguen tres categorías:

- Completa. Incluyen las dos partes de la precondición.
- Parcial. Solamente incluyen una de las dos partes de la precondición.
- Vacía. No incluyen precondición o ésta contiene elementos que no añaden información (p.ej. los tipos de datos) o que no corresponden a una precondición.

**Tabla 2.** Resultados de calidad de la precondición (N=34)

| **Precondición** | **Grupos** | **# (%) grupos** |
|---|---|---|
| Completa | G01, G02, G07, G08, G10, G12, G14, G18, G20, G21, G23, G25, G28 | 13 (38%) |
| Parcial | G03, G04, G05, G06, G09, G11, , G15, G16, G19, G24, G26, G29, G30, G32, G34 | 15 (44%) |
| Vacía | G13, G17, G22, G27, G31, G33 | 6 (18%) |

Recuérdese que hubo 9 segundas entregas. De éstas, cuatro grupos no variaron sus precondiciones (G02, G06, G027, G32). De las cinco entregas restantes, tres dieron una respuesta mejor (G12, G28, G29), un grupo cambió una parte de la precondición por otra (G30) y otro la empeoró (G32). En todos los casos, dejamos su mejor contribución.

En todo caso, obsérvese que los alumnos que no especifican la precondición correctamente asciende al 62%, de los cuales el 18% no especifican nada de la precondición.

[3] A pesar de incluir dos partes en su respuesta, consideramos que ésta sólo es parcial porque no restringe el tipo *int* de los beneficios.

[4] Sin embargo, mencionan que los beneficios de las tareas de alta tensión deberían ser mayores que los de las tareas de baja tensión.

En segundo lugar, podemos preguntarnos si los rangos de generación de datos son coherentes con la parte de la precondición que declara los valores admitidos como beneficios esperables por las tareas.

Los resultados se muestran en la Tabla 3. En ella no aparece el grupo 22 ya que no realizó la experimentación. Puede observarse que se distinguen tres categorías:

- Coherente. Grupos que han sido coherentes entre su precondición (sea completa o se refiera únicamente a los beneficios) y los rangos de datos. Casi todos los grupos declararon que los beneficios esperados debían ser mayores que cero. Hay dos casos especiales. Así, el grupo 8 contenía en dos sitios las dos posibilidades para los beneficios nulos, pero podemos interpretar que es un error y tomamos la que es coherente con sus rangos (beneficios mayores que cero). Asimismo, el grupo 14 claramente especifica números no negativos pero su generación de datos es coherente con la precondición.
- (Parcialmente) coherente. Grupos que especifican correctamente la parte de la precondición sobre los beneficios, pero los rangos de generación incluyen el cero. Los grupos 12 y 20 declaran valores "positivos" y permiten generar beneficios nulos, pero no sabemos si su comprensión de los números "positivos" incluye el cero, por lo que al menos son parcialmente coherentes.
- Desconocida. En general, estos grupos omiten en su precondición toda mención a los beneficios esperables, por lo cual no podemos determinar su grado de coherencia. El resto de grupos se dividen entre quienes contemplan generar el valor cero (6 grupos: 5, 11, 13, 16 y 17), quienes no generan el cero (5 grupos: 19, 26, 31, 33 y 34) y quienes admiten generar el valor cero para las tareas de baja tensión (3 grupos: 9, 15 y 27).

**Tabla 3.** Resultados de coherencia entre precondición y valores generados (N=33)

| Categoría | Códigos | Grupos | # grupos | % grupos |
|---|---|---|---|---|
| Coherente | C → C | G01, G02, G08, G10, G14, G18, G21, G23, G25, G28, G32 | 16 | 49% |
| | P (bs) → C | G04, G06, G24, G29, G30 | | |
| (Parcialmente) coherente | C → P | G07, G12, G20 | 4 | 12% |
| | P (bs) → P | G03 | | |
| Desconocido | P (ls) → – | G05, G09, G11, G15, G16, G19, G26, G34 | 13 | 39% |
| | – → – | G13, G17, G27, G31, G33 | | |

Recuérdese que nueve grupos realizaron dos entregas. Cuatro grupos mantuvieron la coherencia de su contribución (G02, G06, G27, G33). Sin embargo, dado que el profesor les hacía comentarios sobre los aspectos de la práctica que estaba mal, otros grupos mejoraron en su segunda entrega la especificación de los beneficios esperados, de forma que podemos apreciar mejor su coherencia (G12, G28, G29, G30). El grupo 32 empeoró su precondición, lo cual podemos atribuirlo a una incomprensión de qué debía mejorar, por lo que mantenemos su primera entrega.

En definitiva, encontramos que la mitad de los alumnos son coherentes, pero los grupos cuya coherencia es desconocida es muy alto, aproximadamente 40%. Solamente hay dos grupos que han sido incoherentes (3 y 7)), al permitir que se generen beneficios nulos cuando su precondición establecía que debían ser mayores que cero. Además, hemos incluido en la categoría de parcialmente coherente a los grupos 12 y 20, ya que no sabemos qué quieren decir con "valores positivos".

Una tercera cuestión es si los rangos de generación aleatoria de datos son mayores para las actividades de alta tensión que para las actividades de baja tensión. El enunciado afirmaba que lo normal es que así fuera, pero dado solamente su carácter "probable", no forma parte de la precondición.

Los resultados se muestran en la Tabla 4. De nuevo, no se incluye el grupo 22. Sólo distinguimos dos categorías:

- Rangos distintos para las tareas de alta y baja tensión. Podría distinguirse entre quienes el solapamiento entre ambos rangos es otro rango o es solamente un valor.
- Rangos distintos para las tareas de alta y baja tensión.

**Tabla 4.** Resultados de distinción entre rangos de valores generados (N=33)

| Categoría | Subcategoría | Grupos | # grupos | % grupos |
|---|---|---|---|---|
| Rangos distintos | Solapamiento en rango | G01, G08, G09, G10, G18, G19, G27, G28, G34 | 9 | 27% |
| | Solapamiento en punto | G02, G04, G06, G15, G31 | 5 | 15% |
| Rangos iguales | | G03, G05, G07, G11, G12, G13, G14, G16, G17, G20, G21, G2, G24, G25, G26, G29, G30, G32, G33 | 19 | 58% |

El grupo 10 experimentó generando datos en tres ocasiones, todas ellas con rangos distintos para las dos clases de tareas. En dos casos, los datos generados se solapaban en un punto mientras que en otra se solapaban en rango. Recogemos esta última categoría de rangos.

Recuérdese que nueve grupos realizaron dos entregas. Un grupo mejoró los rangos de generación de datos aleatorios (G06), mientras que cinco grupos mantuvieron rangos iguales para las dos clases de beneficios (G12, G29, G30, G32, G33). Otros tres grupos ya proponían rangos distintos, aunque cabe destacar que un grupo cambió rangos solapados en rango por rangos solapados en punto (G02)

El porcentaje de grupos que especificaron el mismo rango de valores para ambas clases de trabajos es 60-40 aproximadamente, siendo mayor el de quienes no distinguieron.

# 4 Contraejemplos

En la práctica 2 también se proponía aprovechar los contraejemplos obtenidos experimentalmente de la optimalidad de los algoritmos desarrollados para analizar sus características y que presentaran contraejemplos más sencillos.

La Tabla 5 resume las características de los algoritmos heurísticos entregados por los alumnos y los contraejemplos encontrados o propuestos. De izquierda a derecha, las sucesivas columnas contienen el número de grupo, número de entrega, las condiciones del experimento realizado (número de casos y longitud de los vectores), un resumen de las características de los algoritmos entregados, el porcentaje de optimalidad de cada algoritmo, un resumen de las características de los contraejemplos encontrados y de los diseñados, y si el informe contiene algún razonamiento sobre la causa de su suboptimalidad.

**Tabla 5.** Informes de prácticas presentados para la práctica 2 – contraejemplos

| Nº grupo | Nº entrega | Condiciones experimento | Algoritmos | Resultados experimentales | Contraejs. experimentales | Contraejs. diseñados | Contraejs. razonados |
|---|---|---|---|---|---|---|---|
| 1. | 1 | N=1.000, L=12 | 1. Similar a H1. No contempla la primera semana. Incrementos de 2<br>2. Alterna trabajos de baja tensión, descanso y alta tensión desde la primera semana con alta tensión. Incrementos de 3<br>3. Sólo toma trabajos de alta tensión en semanas pares (desde la primera), salvo quizá en la última semana impar. Incrementos de 2<br>4. Elige entre todos los trabajos pares de baja o de alta tensión, salvo quizá en la última semana impar<br>5. (H1) Contempla la primera semana. Incrementos de 2 | 1: 24'6%<br>2: 0'01%<br>**3: 0%**<br>4: 5'9%<br>5: 90'9% | **1**-5<br>1-**5** | L=6: 1-**5** | Sí |
| 2. | 1 | N=1.000, L=5 | 1. Similar a H1. No contempla la primera semana. Incrementos de 1. Incumple restricciones: puede sumarse b[i] 2 veces o b[i]+a[i+1]<br>2. Similar a H1. No contempla la primera semana. Incrementos de 1. Incumple restricciones: puede sumarse a[i+1] sucesivas veces | 1: 60'5%<br>**2: 100%** | – | – | – |
| | 2 | N=1.000, L=5 (corregido el rango de valores del vector *as*) | Los mismos | 1: 29'4%<br>2: 71'8% | **1**-2, sin valores<br>1-**2**, sin valores | – | – |
| 3. | 1 | N=100, L=12 | 1. Similar a H1. No contempla la primera semana. Incrementos de 2<br>2. Elige entre todos los trabajos de baja o de alta tensión. Contempla elegir un trabajo de alta tensión en la primera semana pero no coger ninguna en la última semana si hay un número impar | **1: 100%**<br>2: 28% | 1-**2**, sin valores | L=5, sin valores | – |
| 4. | 1 | N=100, L=10 | 1. Similar a H1. No contempla la primera semana. Incrementos de 2<br>2. Similar a H1. Contempla la primera semana. Incrementos de 3<br>3. Elige entre todos los trabajos de baja tensión, pares de alta tensión o impares de alta tensión<br>4. (H1) Contempla la primera semana. Incrementos de 2 | 1: 13%<br>2: 16%<br>3: 24%<br>4: 72% | – | L=4: "valor óptimo"<br>L=6: "valor óptimo"<br>L=4: "valor óptimo"<br>L=5: "valor óptimo" | Sí<br>Sí<br>Sí<br>Sí |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 5. | 1 | N=100, L=5 | 1. Similar a H1. Contempla la primera semana. Incrementos de 1. Incumple restricciones: puede sumarse a[i+1] anticipadamente<br>2. Sólo toma trabajos de baja tensión<br>3. Elige entre los trabajos pares de alta tensión o impares de alta tensión | 1: 96%<br>2: 47%<br>3: 4% | –<br>2, sin valores<br>3, sin valores | –<br>L=5, sin valores<br>L=5, sin valores | –<br>Sí<br>Sí |
| 6. | 1 | N=100, L=15 | 1. Similar a H1. No contempla la primera semana. Incrementos de 1<br>2. Alterna trabajos de baja tensión, descanso y alta tensión desde la primera semana con alta tensión. Incrementos de 1, con un contador auxiliar<br>3. (H1) Contempla la primera semana. Incrementos de 2 | 1: 30%<br>**2: 0%**<br>3: 75% | – | – | – |
| | 2 | N=100, L=10 | Los mismos | 1: 17%<br>2: 15%<br>3: 74% | 1-**2**-3 | L=5 | – |
| 7. | 1 | N=100, L=6 | 1. Similar a H1. No contempla la primera semana. Incrementos de 2<br>2. Similar a H1. Contempla la primera semana. Incrementos de 1, con una variable auxiliar | 1: 46%<br>2: 95% | **1**-2<br>1-**2** | – | Sí<br>Sí |
| 8. | 1 | N=100, L=5 | 1. Similar a H1. No contempla la primera semana. Incrementos de 2<br>2: Similar a H1. No contempla la primera semana. Incrementos de 1. Permite varios descansos sucesivos para ganar un beneficio alto | 1: 73<br>2: 99 | – | – | – |
| 9. | 1 | N=200, L=13 | 1. Similar a H1. No contempla la primera semana. Incrementos de 2<br>2. (H1) Contempla la primera semana. Incrementos de 2<br>3. Elige entre trabajar sólo días pares o sólo días impares, eligiendo siempre el trabajo de mayor beneficio en la semana | 1: 26%<br>2: 77%<br>3: 0’5% | **1**-2-3<br>1-**2**-3<br>1-2-**3** | L=4: “valor óptimo”<br>L=4: “valor óptimo”<br>L=4: “valor óptimo” | Sí<br>Sí<br>Sí |
| 10. | 1 | N=100, L=5<br>N=100, L=20<br>N=100, L=20 | 1. (H1) Contempla la primera semana. Incrementos de 2<br>2. Elige entre todos los trabajos de baja tensión, pares de alta tensión o impares de alta tensión | 1: 53%<br>2: 76%<br>**1: 100%**<br>2: 32%<br>1: 66%<br>2:53% | L=10:<br>1, “valor óptimo”<br>2, “valor óptimo” | L=5: “valor óptimo”<br>L=6: “valor óptimo” | Sí<br>Sí |
| 11. | 1 | N=100, L=20 | 1. Similar a H1. No contempla la primera semana. Incrementos de 2<br>2. (H1) Contempla la primera semana. Compara sucesivamente un trabajo de baja tensión y uno de alta, descanso y uno de alta, y dos trabajos de baja tensión. Incrementos de 1 | 1: 20%<br>2: 97% | **1**-2<br>1-**2** | L=5, valores mayores<br>L=6, valores mayores | Sí<br>Sí |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 12. | 1 | N=100, L=7 | 1. Similar a H1. No contempla la primera semana. Incrementos variables de 1 (trabajos de baja tensión) ó 2 (de alta)<br>2. Se ordenan los beneficios altos y se van eligiendo todos los posibles, manteniendo las restricciones del problema | 1: 77%<br>2: 25% | – | – | – |
| | 2 | N=100, L=7 | 1. Similar a H1. No contempla la primera semana. Incrementos variables de 1 (trabajos de baja tensión) ó 2 (de alta)<br>2. Se ordenan los beneficios altos y se van eligiendo todos los posibles, manteniendo las restricciones del problema<br>3. Similar a H1. Contempla la primera semana. Incrementos variables de 1 (trabajos de baja tensión) ó 2 (de alta) | 1: 49%<br>2: 11%<br>3: 48% | **1-2**-3<br>1-**2-3** | – | Sí<br>Sí |
| 13. | 1 | N=1.000, L=10 | 1. Similar a H1. No contempla la primera semana. Incrementos de 1+1<br>2. Elige entre todos los trabajos de baja o de alta tensión, contando sólo los trabajos de alta tensión mayores que los respectivos de baja tensión | **1: 100%**<br>2: 90'1% | 1-**2** | – | – |
| 14. | 1 | N=200, L=11,12 | 1. Similar a H1. No contempla la primera semana. Incrementos variables de 1 (trabajos de baja tensión) ó 2 (de alta)<br>2. Variante de la anterior, donde sólo se comparan a[i+1] y b[i+1] | 1: 92%<br>2: 9'5% | 1: L=11, "valor óptimo"<br>2: L=12, "valor óptimo" | L=3: "valor óptimo"<br>L=3: "valor óptimo" | Sí<br>Sí |
| 15. | 1 | N=1.000, L=12 | 1: Similar a H1. No contempla la primera semana. Incrementos de 2<br>2: Similar a H1. Contempla la primera semana (de forma algo diferente). Incrementos variables de 1 (trabajos de baja tensión) ó 2 (de alta)<br>3: Variante de la anterior, donde también se comparan trabajos de tres semanas consecutivas. Incrementos variables de 1 (trabajos de baja tensión), 2 (de alta) ó 3 (trabajo de baja tensión en semana 1 y alta tensión en semana 3) | 1: 43'2%<br>2: 72'2%<br>3: 16'8% | **1**-2-3<br>1-**2**-3<br>1-2-**3** | L=4<br>L=4<br>L=5 | Sí<br>Sí<br>Sí |
| 16. | 1 | N=100, L=5 | 1: Similar a H1. No contempla la primera semana. Incrementos de 2<br>2: Elige entre todos los trabajos de baja tensión, pares de alta tensión o impares de alta tensión | 1: 88%<br>2: 73% | **1**-2<br>1-**2** | L=5<br>L=5 | – |
| 17. | 1 | N=10, L=10<br><br>N=10, L=9 | 1: Similar a H1. No contempla la primera semana. Incrementos de 2<br>2: (H1) Contempla la primera semana. Incrementos de 2<br>3: Tomar los trabajos de alta tensión de las semanas pares | 1: 40%<br>**2: 100%**<br>**3: 0%**<br><br>1: 40%<br>**2: 100%**<br>3: 10% | L=10:<br>1, "valor óptimo"<br>3, "valor óptimo" | L=5<br>L=5 | Sí<br>Sí |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 18. | 1 | N=100, L=8 | 1. Similar a H1. No contempla la primera semana. Incrementos de 2<br>2. Sólo toma trabajos de baja tensión, salvo quizá en la primera semana<br>3. (H1) Contempla la primera semana. Incrementos de 2<br>4. Sólo toma trabajos de alta tensión en semanas pares, salvo quizá en la última semana impar | 1: 28%<br>2: 24%<br>3: 78%<br>4: 3% | 1, “valor mejor”<br>2, “valor mejor”<br>3, “valor mejor”<br>4, “valor mejor” | L=4<br>L=4<br>L=5<br>L=5 | Sí<br>Sí<br>Sí<br>Sí |
| 19. | 1 | N=2.000, L=55 | 1. Similar a H1. No contempla la primera semana. Incrementos de 2<br>2. (H1) Contempla la primera semana. Incrementos de 2<br>3. Recorrido en orden inverso comparando a[i] y b[i]. Decrementos variables de 1 (trabajos de baja tensión) ó 2 (de alta)<br>4. Similar a H1 y al anterior. Recorrido en orden inverso. Decrementos variables de 1 (trabajos de baja tensión) ó 2 (de alta) | 1: 6’35%<br>2: 11’3%<br>**3: 0%**<br>4: 84’3% | 1-**2**-**3**-**4**<br>**1**-2-**3**-**4**<br>**1**-**2**-**3**-4 | L=5<br>L=5<br>L=5<br>L=5 | Sí<br>Sí<br>Sí<br>Sí |
| 20. | 1 | N=1.000, L=10 | 1. Similar a H1. No contempla la primera semana. Incrementos de 2<br>2. Similar a H1. Contempla la primera semana. Incrementos de 1. Incumple restricciones: puede sumarse a[i+1] sucesivas veces<br>3. Sólo toma trabajos de baja tensión, salvo quizá en la última semana | 1: 65’6%<br>2: 94%<br>3: 0’1% | **1**-2-3<br>1-**2**-3<br>1-2-**3** | L=4<br>L=4<br>L=4 | Sí<br>Sí<br>Sí |
| 21. | 1 | N=1.000, L=10 | 1. Similar a H1. No contempla la primera semana. Incrementos de 2<br>2. (H1) Contempla la primera semana. Incrementos de 2 | 1: 54’5%<br>**2: 100%** | **1**-2<br>2, “valor óptimo” | L=3<br>L=3 | Sí<br>Sí |
| 22. | 1 | – | 1. Similar a H1. No contempla la primera semana. Incrementos variables de 1 (trabajos de baja tensión) ó 2 (de alta)<br>2. Similar a H1. Contempla la primera semana. Incrementos variables de 1 (trabajos de baja tensión) ó 2 (de alta)<br>3. Parece igual al anterior | – | – | – | – |
| 23. | 1 | N=100, L=12 | 1. Similar a H1. No contempla la primera semana. Incrementos de 2<br>2. Variante de la anterior donde se comparan a[i]+b[i+1], a[i+1] y b[i]+b[i+1], teniendo en cuenta los descansos. Incrementos de 1 | 1: 30%<br>2: 96% | 1-**2** | L=6 | Sí |
| 24. | 1 | N=100, L=12 | 1. Similar a H1. No contempla la primera semana. Incrementos de 2<br>2. Variante de la anterior donde se comparan a[i], a[i+1] y b[i]+b[i+1], teniendo en cuenta los descansos. Incrementos de 2<br>3. Variante de la anterior donde se comparan a[i]+b[i+1], a[i]+a[i+2], b[i]+a[i+2] y b[i]+b[i+1]+b[i+2], teniendo en cuenta los descansos. Incrementos de 3 | 1: 49%<br>2: 54%<br>3: 75% | **1**-2-3<br>1-**2**-3<br>1-2-**3** | L=2<br>L=3<br>L=4 | Sí<br>Sí<br>Sí |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 25. | 1 | N=1.000, L=5 | 1. Similar a H1. No contempla la primera semana. Incrementos de 2<br>2. Variante de la anterior, implementado de forma complicada (se comprueba si merece la pena elegir un trabajo de alta tensión aunque suponga deshacer la elección del trabajo anterior). Contempla la primera semana. Incrementos de 1 | 1: 47'9%<br>2: 99'1% | **1**-2<br>1-**2** | L=3<br>L=4 | Sí<br>Sí |
| 26. | 1 | N=100, L=12 | 1. Similar a H1. No contempla la primera semana. Incrementos de 2<br>2. Variación del anterior. Compara sucesivamente b[i]+b[i+1], a[i]+b[i+1] y a[i+1]. Incrementos de 2<br>3. (H1) Contempla la primera semana. Incrementos de 2 | 1: 27%<br>2: 53%<br>3: 64% | **1**-2-3<br>1-**2**-3<br>1-2-**3** | L=5<br>L=5<br>L=5 | Sí<br>Sí<br>Sí |
| 27. | 1 | N=100, L=12 | 1. (H1) Contempla la primera semana. Incrementos de 1. No gestiona bien los descansos.<br>2. Variación del anterior. Compara también a[i+1] con a[i+2] pero estructura mal las alternativas y sólo suma beneficios en el último mes. Incrementos de 1<br>3. (H1). Nunca toma el último mes. Incrementos de 1 | **1: 100%**<br>**2: 0%**<br>**3: 0%** | – | – | – |
| | 2 | N=100, L=12 | 1. (H1) Contempla la primera semana. Incrementos de 1.<br>2. Variación del anterior. Compara también a[i+1] con a[i+2]. Incrementos de 1<br>3. (H1). Siempre toma a[i] en la primera semana. Incrementos de 1 | 1: 96%<br>2: 8%<br>3: 18% | **1**-2-**3**<br>**1-2**-3 | – | – |
| 28. | 1 | N=100, L=6<br>N=300, L=6 | 1. Similar a H1. No contempla la primera semana. Incrementos de 2<br>2. Similar a H1. No contempla la primera semana. Incrementos de 1 | 1: 36%<br>2: 96%<br><br>1: 43%<br>2: 98'33% | 1, "valor óptimo"<br>2, "valor óptimo" | – | – |
| | 2 | N=100, L=6<br>N=300, L=6 | Los mismos | 1: 36%<br>2: 96%<br><br>1: 43%<br>2: 98'33% | 1, "valor óptimo"<br>2, "valor óptimo" | L=3<br>L=3 | – |
| 29. | 1 | N=100, L=6 | 1. (H1) Contempla la primera semana, pero teniendo en cuenta también la segunda. Incrementos de 1, con variables auxiliares<br>2. (H1) Contempla la primera semana. Incrementos de 1, con variables auxiliares | 1: 98%<br>2: 96% | **1**-2 | L=3 | Sí |
| | 2 | Lo mismo | Los mismos | Los mismos | El mismo | El mismo | Lo mismo |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 30. | 1 | N=5, L=12 | 1. Similar a H1. Contempla la primera semana, al comparar a[i], a[i+1] y b[i]+b[i+1]. Incrementos de 1+1<br>2. Similar a H1. Contempla la primera semana. A veces, quizá por error, sólo compara a[i+1] y b[i+1]. Incrementos de 1 y variables auxiliares | 1: 80%<br>2: 20% | **1**-2<br>1-**2** | – | – |
| | 2 | N=50, L=12 | Los mismos | 1: 98%<br>2: 2% | **1**-2<br>1-**2** | –<br>– | Sí<br>Sí |
| 31. | 1 | N=100, L=7 | 1. Similar a H1. No contempla la primera semana. Incrementos de 2<br>2. Elige entre todos los pares o impares de alta tensión<br>3. (H1) Contempla la primera semana. Incrementos de 1 | 1: 12%<br>2: 63%<br>3: 80% | 1, "valor óptimo"<br>2, "valor óptimo"<br>3, "valor óptimo" | L=7<br>L=7<br>L=7 | Sí<br>Sí<br>Sí |
| 32. | 1 | N=100, L=12 | 1. (H1) Contempla la primera semana, manipulando el índice i. Incrementos de 2<br>2. Ordenación de los distintos trabajos y sucesivos recorridos de los vectores | 1: 83%<br>2: 56% | **1**-2<br>1-**2** | – | – |
| | 2 | N=100, L=12 | Los mismos | 1: 83%<br>2: 56% | – | L=5<br>L=5 | Sí<br>Sí |
| 33. | 1 | N=100, L=10<br>N=500, L=10 | 1. Similar a H1. No contempla la primera semana. Incrementos de 1+1<br>2. Elige entre todos los trabajos de baja tensión, pares de alta tensión o impares de alta tensión | **1: 100%**<br>2: 34%<br>**1: 100%**<br>2: 44'4% | – | – | – |
| | 2 | N=100, L=10<br>N=500, L=10 | Los mismos | **1: 100%**<br>2: 34%<br>**1: 100%**<br>2: 44'4% | 1-**2** | L=5 | Sí |
| 34. | 1 | N=1.000, L=15 | 1. Similar a H1. No contempla la primera semana. Incrementos de 2<br>2. Similar a H1. Contempla la primera semana mediante un recorrido inverso. Decrementos de 2 | 1: 21'9%<br>2: 85'1% | 1, "valor óptimo"<br>2, "valor óptimo" | L=5<br>L=5 | Sí<br>Sí |

De nuevo, excluimos de este análisis el grupo 22, que no experimentó con sus algoritmos.

Recuérdese que 9 de los 34 grupos realizaron una segunda entrega. En ésta, cinco grupos aportaron contraejemplos de forma completa o incompleta. Llama la atención el grupo 32 que, habiendo aportado dos contraejemplos de OptimEx en la primera entrega, no lo hizo en la segunda (quizá un despiste).

Veamos sucesivamente las contribuciones realizadas según las últimas tres columnas de la Tabla 5: contraejemplos mostrados por OptimEx, contraejemplos diseñados por el grupo de prácticas, y si contiene alguna justificación.

La Tabla 6 muestra los resultados para los contraejemplos generados por OptimEx. Se han distinguido tres categorías de grupos: que han presentado contraejemplos completos para todos los algoritmos, que han realizado una presentación parcial (bien de menos algoritmos o sin dar los valores correspondientes según OptimEx), y los que no presentan ningún contraejemplo

Puede comprobarse que las categorías más pobladas en la primera entrega son la primera y segunda, cada una con un número de grupos algo mayor que la tercera parte (39%) y después queda la categoría tercera con una quinta parte (21%).

En cuanto a su evolución, la primera categoría se mantiene constante en ambas entregas. (Si G32 no hubiera tenido un despiste, habría habido un ligero incremento.) Sin embargo, la segunda categoría crece desde el 39% al 51%, acompañado del mismo decremento en la tercera categoría (del 21% al 9%).

La Tabla 7 muestra los resultados para los contraejemplos diseñados por los propios alumnos. Se han distinguido las mismas categorías para la Tabla 6. Puede comprobarse que las tres categorías tienen el mismo porcentaje (33%).

En cuanto a su evolución, se produce un acusado decremento en la tercera categoría (del 32% al 21%). Sin embargo, la segunda categoría crece ligeramente (desde el 32% al 36%) y, sobre todo, la primera categoría (del 32% al 42%).

La Tabla 8 muestra los resultados tras analizar si los contraejemplos presentados por los alumnos están razonados. Realizamos el análisis para los grupos que presentaron algún contraejemplo, es decir 27 en la primera entrega y 32 en la segunda. Pueden distinguirse dos categorías: los que han razonado sus contraejemplos y los que no.

La categoría más numerosa ha sido la de razonar contraejemplos (78% en ambas entregas). No obstante, hay un incremento en el número de grupos que razonan de la primera entrega (21 grupos) a la segunda (25). Dentro de esta categoría, son mayoritarios los que presentan tanto contraejemplos experimentales como propios (70% y 63% en la entrega primera y segunda, respectivamente).

**Tabla 6**. Resultados de contraejemplos generados por OptimEx en las dos entregas de la práctica (N=33)

| Propuesta | # (%) grupos | Grupos | # (%) grupos | Grupos |
|---|---|---|---|---|
| Contraejemplos para todos los algoritmos | 13 (39%) | G03, G07, G09, G11, G15, G16, G19, G20, G24, G25, G26, G30, G32 | 13 (39%) | G03, G07, G09, G11, G15, G16, G19, G20, G24, G25, G26, G27, G30 |
| Contraejemplos para todos los algoritmos, incompletos | 7 (21%) | G10, G14, G18, G21, G28, G31, G34 | 8 (24%) | G02, G10, G14, G18, G21, G28, G31, G34 |
| Contraejemplos para algunos algoritmos | 4 (12%) | G01, G13, G23, G29 | 7 (21%) | G01, G06, G12, G13, G23, G29, G33 |
| Contraejemplos para algunos algoritmos, incompletos | 2 (6%) | G05, G17 | 2 (6%) | G05, G17 |
| Sin contraejemplos | 7 (21%) | G02, G04, G06, G08, G12, G27, G33 | 3 (9%) | G04, G08, G32 |

**Tabla 7**. Resultados de contraejemplos diseñados por los alumnos en las dos entregas de la práctica (N=33)

| Propuesta | # (%) grupos | Grupos | # (%) grupos | Grupos |
|---|---|---|---|---|
| Contraejemplos para todos los algoritmos | 11 (32%) | G15, G16, G18, G19, G20, G21, G24, G25, G26, G31, G34 | 14 (42%) | G15, G16, G18, G19, G20, G21, G24, G25, G26, G28, G31, G32, G33, G34 |
| Contraejemplos para todos los algoritmos, incompletos | 6 (18%) | G03, G04, G09, G10, G11, G14 | 6 (18%) | G03, G04, G09, G10, G11, G14 |
| Contraejemplos para algunos algoritmos | 4 (12%) | G01, G17, G23, G29 | 5 (15%) | G01, G06, G17, G23, G29 |
| Contraejemplos para algunos algoritmos, incompletos | 1 (3%) | G05 | 1 (3%) | G05 |
| Sin contraejemplos | 11 (32%) | G02, G06, G07, G08, G12, G13, G27, G28, G30, G32, G33 | 7 (21%) | G02, G07, G08, G12, G13, G27, G30 |

**Tabla 8**. Resultados de analizar el razonamiento sobre contraejemplos en las dos entregas de la práctica (N=33)

| Propuesta | # (%) grupos (N=27) | Grupos | # (%) grupos (N=32) | Grupos |
|---|---|---|---|---|
| Razonamiento, contraejemplos experimentales y propios | 19 (70%) | G01, G05, G09, G10, G11, G14, G15, G17, G18, G19, G20, G21, G23, G24, G25, G26, G29, G31, G34 | 20 (63%) | G01, G05, G09, G10, G11, G14, G15, G17, G18, G19, G20, G21, G23, G24, G25, G26, G29, G31, G33, G34 |
| Razonamiento, contraejemplos experimentales | 1 (4%) | G07 | 3 (9%) | G07, G12, G30 |
| Razonamiento, contraejemplos propios | 1 (4%) | G04 | 2 (6%) | G04, G32 |
| Sin razonamiento, contraejemplos experimentales y propios | 2 (7%) | G03, G16 | 4 (13%) | G03, G06, G16, G28 |
| Sin razonamiento, contraejemplos experimentales | 4 (15%) | G13, G28, G30, G32 | 3 (9%) | G02, G13, G27 |
| Sin contraejemplos | 6 | G02, G06, G08, G12, G27, G33 | 1 | G08 |

# 5 Corrección de Algoritmos

Las prácticas 3a, 3b y 5 pedían desarrollar algoritmos que resolvieran el problema de planificación de trabajos con beneficio máximo [9, pág. 313], pero utilizando diversas técnicas de diseño: vuelta atrás (práctica 3a), ramificación y poda (3b) y programación dinámica (5). Dado que estas técnicas son exactas, permiten detectar situaciones erróneas de varios tipos:

- Si se obtienen resultados subóptimos para los algoritmos exactos, hay algún error, bien en la lógica de algún algoritmo anterior (algoritmos heurísticos de la práctica 2), bien en la lógica de un algoritmo nuevo (como es el caso de un algoritmo de vuelta atrás, práctica 3a, o de un algoritmo recursivo de programación dinámica, práctica 5).
- Si se obtienen resultados distintos para algoritmos derivados de algoritmos exactos, hay algún error en el proceso de derivación de los nuevos algoritmos (como es el caso de un algoritmo de ramificación y poda a partir de uno de vuelta atrás, práctica 3b, o de un algoritmo tabulado de programación dinámica a partir de uno recursivo, práctica 5).

También se daría una situación similar al desarrollar algoritmos probabilistas y compararlos con algoritmos exactos ya existentes. Es el caso de la práctica 6, en la que se pedía desarrollar un algoritmo probabilista para la versión de minimización del problema de llenado de cajas [10, págs. 534-535]. En el Aula Virtual, se disponía de dos algoritmos aproximados, un algoritmo de vuelta atrás y un algoritmo de ramificación y poda. La situación errónea es:

- Si se obtienen algoritmos probabilistas óptimos o dejan de obtenerse resultados óptimos de algoritmos exactos ya existentes, hay algún error en la lógica de un algoritmo probabilista.

Se recogieron 11 prácticas. Sólo hubo una entrega. No informan de la modificación de ningún algoritmo.

Veamos a continuación los resultados obtenidos para el primer problema.

## 5.1 Depuración de Algoritmos

La Tabla 9 muestra aquellos casos en los que se han modificado algunos algoritmos gracias a los resultados obtenidos en la experimentación, según informan los propios alumnos en sus informes de prácticas. La primera columna contiene el número de grupo y las siguientes cinco columnas resumen las incidencias informadas en cinco entregas, a saber, de izquierda a derecha: vuelta atrás, ramificación y poda, segunda entrega de ramificación y poda, programación dinámica y segunda entrega de programación dinámica).

En cada celda se incluye información según la siguiente codificación:

- −: Entrega no realizada.
- SE: Sin experimentación.
- En blanco: Sin incidencias informadas.
- S: Sospechan que puede haber algún problema (pero lo atribuyen a la versión del sistema operativo).
- Uno o varios algoritmos seguidos de uno o varios algoritmos entre paréntesis: Según el informe entregado, los algoritmos incluidos en primer lugar se modificaron para subsanar los resultados del experimento. Los algoritmos incluidos entre paréntesis se usaron en el experimento pero no se modificaron.

En algunos casos, los informes documentan modificaciones realizadas en los algoritmos pero por causas distintas de la experimentación. Clasificamos estos casos con el código "blanco", pero explicamos la incidencia en una nota a pie de página.

**Tabla 9.** Modificaciones realizadas de prácticas debido a los resultados de experimentación

| Número de grupo | Pr.3a | Pr.3b | Pr.3b-2 | Pr.5 | Pr.5-2 |
|---|---|---|---|---|---|
| 1. | VA (H1, H2, H3, H4, H5) | RyP2, RyP3 (VA, RyP1) | – | SE | |
| 2. | | | | | – |
| 3. | | | – | | – |
| 4. | | VA (H1, H2, H3, RyP, RyP1) | – | | – |
| 5. | | VA (H2, H3, RyP) | – | H1 (VA, Rec, PD) | – |
| 6. | | VA, RyP (H1, H2, H3) | | | – |
| 7. | | .[5] | – | S | – |
| 8. | | | | RyP[6] (H1, H2, VA, PD1, PD2) | – |
| 9. | | | – | | – |
| 10. | | | – | | – |
| 11. | VA (H1, H2) | | – | PD (VA, RyP, Rec) | – |
| 12. | H2 (H1, VA) | | – | | – |
| 13. | | | RyP (H1, H2, VA) | | – |
| 14. | | | | – | – |
| 15. | | | – | | – |

[5] Aunque no se modificó ningún algoritmo por razones de optimalidad, tuvieron que modificar el algoritmo de vuelta atrás para poder desarrollar el algoritmo de ramificación y poda a partir de él.

[6] El informe explica la modificación realizada del algoritmo de ramificación y poda, pero no se debió a problemas de optimalidad porque en las dos entregas de la práctica 3b calculaba el 100% de resultados optimales. No obstante, los resultados de eficiencia de la práctica 5 muestran que el nuevo algoritmo de ramificación y poda tarda más tiempo que el de vuelta atrás (a diferencia de lo que sucedía en las dos entregas de la práctica 3b).

| | | | | | |
|---|---|---|---|---|---|
| 16. | | | – | – | – |
| 17. | | | | | – |
| 18. | VA<br>(H1, H2, H3, H4) | RyP<br>(H1, H2, H3, H4, VA) | | | – |
| 19. | H3<br>(H1, H2, H4, VA) | | – | | – |
| 20. | VA<br>(H1, H2) | RyP<br>(VA, H1, H2)[7] | – | | – |
| 21. | | RyP<br>(VA, H1, H2) | – | | – |
| 22. | | ¿RyP1, RyP2 o RyP3?<br>(VA, H1, H2) | – | Rec<br>(H1, H2, VA, RyP1,<br>RyP2, RyP3, PD) | – |
| 23. | H2<br>(H1, VA) | | – | | – |
| 24. | H3<br>(H1, H2, VA) | .[8] | – | | – |
| 25. | | RyP2<br>(H1, H2, VA, RyP1) | – | | – |
| 26. | | | – | | – |
| 27. | H2<br>(H1, VA) | | – | | – |
| 28. | | | | | – |
| 29. | H2<br>(H1, VA) | | .[9] | Rec<br>(H1, H2, VA, RyP, PD) | .[10] |
| 30. | | .[11] | – | | – |
| 31. | VA<br>(H1, H2) | VA<br>(H1, H2, RyP) | – | | – |
| 32. | VA<br>(H1, H2) | – | – | | – |

Podemos resumir los algoritmos que se revisaron en cada entrega, agrupados por su técnica de diseño. La Tabla 10 muestra esta información. Hemos agrupado las entregas primera y segunda de cada práctica. Cada fila contabiliza el número de grupos que modificaron algún algoritmo de la técnica de diseño correspondiente, no el

[7] Ha modificado el algoritmo de vuelta atrás, pero debido a mis comentarios sobre la entrega de la práctica 3a, no a su propia experimentación en la práctica 3b. En cuanto al algoritmo de ramificación y poda, comenta diversos aspectos que ha debido corregir, probablemente para alcanzar el 100% de los casos optimales.

[8] Ha modificado el algoritmo de vuelta atrás, pero debido a mis comentarios, no a problemas con la optimalidad.

[9] Han modificado el algoritmo de ramificación y cota pero debido a mis comentarios, no a problemas con la optimalidad.

[10] Han modificado el algoritmo recursivo (y, por tanto, también el algoritmo tabulador) pero debido a mis comentarios, no a problemas con la optimalidad.

[11] Este grupo no es claro porque su informe es muy escueto pero da a entender que OptimEx les ha ayudado a depurar algún algoritmo de búsqueda: "(…) en este caso la experimentación en base a la eficiencia, que además es muy útil para corregir posibles errores surgidos durante la realización de la práctica".

número de algoritmos. Por ejemplo, contamos uno para la técnica de diseño de ramificación y poda en la práctica 3b del grupo 1, aunque fueron dos los algoritmos de ramificación y poda modificados. Sin embargo, para el grupo 6, contamos uno para la técnica de vuelta atrás y uno para ramificación y poda en la misma práctica. Obsérvese que es el único caso en el que se informa de haber modificado en la misma práctica algoritmos de técnicas distintas.

**Tabla 10.** Resultados de modificaciones realizadas de prácticas debido a los resultados de experimentación

| **Tipo de algoritmo** | **# (%) práctica 3a (N=32)** | **# (%) práctica 3b (N=31)** | **# (%) práctica 5 (N=30)** |
|---|---|---|---|
| Algoritmos heurísticos | 6 (19%) | 0 (0%) | 1 (3%) |
| Vuelta atrás | 6 (19%) | 4 (13%) | 0 (0%) |
| Ramificación y poda | – | 8 (26%) | 1 (3%) |
| Programación dinámica – recursivo | – | – | 2 (6%) |
| Programación dinámica – tabulador | – | – | 1 (3%) |
| Total grupos | 12 (38%) | 11 (35%) | 5 (17%) |

## 5.2 Predicciones de Optimalidad

Los algoritmos heurísticos, aproximados y probabilísticos (de la práctica 6) son inexactos, mientras que los algoritmos de vuelta atrás, de ramificación y poda, y de programación dinámica son exactos. Es decir, la teoría hace las siguientes predicciones:

- Los algoritmos heurísticos, aproximados y probabilísticos producirán resultados optimales inferiores al 100%.
- Los algoritmos de vuelta atrás, de ramificación y poda y de programación dinámica producirán resultados optimales en el 100% los casos.

En el apartado anterior se identificaron los grupos que realizaron modificaciones en los algoritmos de las prácticas 3a, 3b y 5 porque los resultados experimentales no coincidían con las predicciones teóricas. La Tabla 11 presenta los casos en los que se producen resultados "imposibles" pero los alumnos no los han identificado. También incluimos los resultados improbables de 0 ó 100% de casos optimales para algoritmos heurísticos en las prácticas 2 y 6.

**Tabla 11.** Resultados experimentales inaceptables o improbables según las predicciones de la teoría

| Número de grupo | Pr.2 | Pr.3a | Pr.3b | Pr.3b-2 | Pr.5 | Pr.5-2 | Pr. 6 |
|---|---|---|---|---|---|---|---|
| 1. | H3: 0% | | | – | SE | | – |
| 2. | H2: 100% | | | | | – | – |
| 3. | H1: 100% | | | – | | – | – |
| 4. | | | | – | | – | |
| 5. | | | | – | | – | – |
| 6. | H2: 0% | | | | | – | – |
| 7. | | | | – | | – | – |
| 8. | | | | | | – | VA: 6’32% |
| 9. | | | | – | .[12] | – | – |
| 10. | H1: 100%[13] | | | – | | – | – |
| 11. | | | | – | | – | – |
| 12. | | | | – | | – | H2: 100% |
| 13. | | | | | | – | – |
| 14. | | | | | – | – | – |
| 15. | | | | – | | – | H2: 100% |
| 16. | | | | – | – | – | H2: 100% |
| 17. | H2: 100%[14]<br>H3: 0% | | | | Rec: 54%; sin Tab | – | – |
| 18. | | | | | | – | |

[12] Aunque los resultados parecen coherentes, no coinciden los nombres del algoritmo recursivo entre el resumen numérico y gráfico (*trabajosH*1 y *trabajosRecursivo*).

[13] Este resultado no es tan extraño como pudiera parecer porque el alumno realizó tres experimentos, con distintas longitudes y valores, obteniéndose este resultado solamente en el segundo caso.

[14] En esta práctica consideran que H2 es un algoritmo exacto pero en la práctica 3a reconocen su error y rectifican ante la evidencia experimental.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 19. | H3: 0% | | | – | | – | |
| 20. | | VA: 93’25% | H1, H2: 16’60%[15] | – | .[16] | – | – |
| 21. | H2: 100%[17] | | | – | | – | – |
| 22. | | | | – | | – | – |
| 23. | | | | – | | – | H2: 100%[18] |
| 24. | | | | – | | – | |
| 25. | | | | – | | – | – |
| 26. | | | | – | | – | – |
| 27. | H1: 100%[19]<br>H2, H3: 0% | | | – | | – | – |
| 28. | | | | | | – | – |
| 29. | | | | | | | |
| 30. | | | | – | sin Tab[20] | – | H2: 100% |
| 31. | | | | – | | – | – |
| 32. | | | – | – | .[21] | – | – |
| 33. | H1: 100% | – | – | – | – | – | – |
| 34. | | – | – | – | – | – | – |

[15] Son algoritmos distintos y con tiempos de ejecución distintos. N=500, L=8.

[16] Aunque los resultados parecen coherentes, no coinciden los nombres del algoritmo recursivo (*trabajosH*1) ni del tabulado (*trabajos_matriz*) con el incluido en el resumen numérico y gráfico (*trabajosHNUEVO*).

[17] Curiosamente, aporta contraejemplos, tanto experimentales como propios.

[18] Comenta que experimentando más debería producir algún resultado subóptimo.

[19] Detecta que los algoritmos están calculando resultados erróneos pero no los puede arreglar.

[20] Incluyen el código del algoritmo tabulado pero no lo comparan en la experimentación. Advierten de que está mal.

[21] Compara los dos algoritmos entre sí, pero no incluyen a ninguno de las prácticas anteriores.

Los algoritmos heurísticos con 0% o 100% de resultados óptimos podrían despertar sospechas de que están mal implementados. Sin embargo, no es extraño que los alumnos acepten sus resultados. En algunos casos, se proponían algoritmos heurísticos que eran variantes de otros. En otros casos, se utilizaba un criterio de selección poco razonable.

No obstante, el resultado del 100% está justificado en un caso (G10). Asimismo, otros dos grupos aportan un contraejemplo (¡aunque su experimento produce 100% de casos óptimos!, G21) o advierten de que los algoritmos están mal (G27).

Los resultados informados de la práctica 6 son extraños, ya que hay 5 prácticas de 11 en las que un algoritmo no exacto produce un 100% de resultados optimales. Sin embargo, puede deberse a que no es un algoritmo heurístico sino aproximado.

Analizando el resto de prácticas, solamente encontramos 3 casos en los que se han presentado algoritmos exactos con menos del 100% de resultados subóptimos: dos algoritmos de vuelta atrás (prácticas 3a y 6) y un algoritmo recursivo de programación dinámica (práctica 5). En otro caso, advierten que el algoritmo tabulado está mal y no lo incluyen en el experimento.

## 6 Conclusiones

Se han analizado los resultados de intentar una mayor integración de la experimentación con la teoría y la ingeniería en una asignatura de algoritmos. Se han analizado tres cuestiones en prácticas planteadas sobre un problema de optimización: relación entre precondición y generación de aleatoria de juegos de datos, uso de la experimentación para obtener contraejemplos “sencillos” de optimalidad, y uso de la experimentación como método de prueba de algoritmos y de comprobación de las predicciones teóricas sobre la exactitud de las distintas técnicas de diseño.

Podemos resumir los resultados obtenidos en:

- Los alumnos tienen mayores dificultades de las deseables para especificar precondiciones. Casi la mitad de los alumnos (44%) la especifican parcialmente y casi una quinta parte no aportan nada útil de ella (18%).
- Los alumnos especifican coherentemente la precondición y la generación de datos aleatorios, pero sin cuidar siempre los detalles, p.ej. permitiendo la generación del valor cero (25% de los grupos con especificación de esta parte de la precondición).
- Aunque no forma parte de la precondición, son menos los grupos que generan datos de entrada adecuados con la indicación adicional sobre rangos relativos de valores de trabajos de baja y alta tensión (42% frente al 58%).
- Los alumnos tienen mayores dificultades de las deseables para encontrar contraejemplos generados automáticamente. Un 39% sólo aportan algunos contraejemplos, mientras que una quinta parte (21%) no aporta ninguno. Con la oportunidad de realizar una segunda entrega, aumenta el porcentaje de algunos contraejemplos al 51%, disminuyendo el porcentaje de quienes no aportan contraejemplos (del 21 al 9%).
- Los alumnos tienen más dificultades para aportar contraejemplos propios que para encontrarlos generados automáticamente. No obstante, casi dos tercios

aportan algún contraejemplo propio (33% todos y 30% algunos). Con la oportunidad de realizar una segunda entrega, aumenta el porcentaje de grupos con todos los contraejemplos (del 33 al 42%), disminuyendo el porcentaje de quienes no aportan contraejemplos (del 36 al 21%).

- El número de grupos que presentan contraejemplos y los razonan es muy alto (78%), siendo mayoritarios los que presentan contraejemplos experimentales y propios (70% y 63% en las dos entregas, respectivamente). El número de los que no razonan prácticamente desaparece en la segunda entrega (de seis entregas −18% del total− a una −3%−).
- La tercera parte aproximadamente de los grupos (38% en la práctica 3a, 32% en la 3b y 17% en la 5) han detectado inconsistencias entre los resultados de exactitud esperables y los experimentales, depurando sus algoritmos, sean nuevos o de prácticas anteriores. Solamente en tres entregas no han detectado dichas inconsistencias (prácticas 3a, 5 y 6).

**Agradecimientos.** Este trabajo se ha financiado con los proyectos de investigación TIN2015-66731-C2-1-R del Ministerio de Economía y Competitividad y S2018/TCS-4307 de la Comunidad Autónoma de Madrid. Este último también está financiado con fondos estructurales (FSE y FEDER).

## Referencias

1. Velázquez Iturbide, J.Á.: An experimental method for the active learning of greedy algorithms. ACM Transactions on Computing Education, 13, 4 (octubre 2013) artículo 18, DOI 10.1145/2534972
2. Esteban Sánchez, N., Pizarro, C., y Velázquez Iturbide, J.Á.: Evaluation of a didactic method for the active learning of greedy algorithms. IEEE Transactions on Education, 57, 2 (2014) 83-91, DOI 10.1109/TE.2013.2275154
3. Velázquez-Iturbide, J.Á.: GreedEx and OptimEx: Two tools to experiment with optimization algorithms. International Journal of Engineering Education, 32, 3A (2016) 1.097-1.106
4. Velázquez Iturbide, J.Á., y Escudero Angulo, J.J.: Extending instructional support in an algorithm benchmarking system. En: Proceedings of the 21st International Symposium on Computers in Education (SIIE 2019), Gonçalo Marques, C., Pereira, I., y Pérez, D. (eds.). IEEE Xplore, 2019, 6 págs., DOI 10.1109/SIIE48397.2019.8970134
5. Velázquez Iturbide, J.Á.: Identification and removal of misconceptions on optimization concepts underlying greedy algorithms. Journal of Research and Practice in Information Technology, 45, 3/4 (agosto 2013) 203-217
6. Velázquez Iturbide, J.Á.: Difficulties, attitudes and misconceptions on experimenting with optimization algorithms. En: Proceedings of the 2014 International Symposium on Computers in Education (SIIE'14), IEEE Xplore (2014) 17-22, DOI 10.1109/SIIE.2014.7017698
7. Velázquez Iturbide, J.Á.: Students' misconceptions of optimization problems. En: Proceedings of the 24th Annual Conference on Innovation and Technology in Computer Science Education, ITiCSE 2019, ACM Press (2019) 464-470, DOI 10.1145/3304221.3319749

8. Martí Oliet, N., Ortega Mallén, Y., y Verdejo, A.: Estructuras de datos y métodos algorítmicos. Ibergarceta Publicaciones, 2ª ed., 2013
9. Kleinberg, J., Tardos, É.: Algorithm Design, Pearson Addison-Wesley, 2006
10. Brassard, G., Bratley, P.: Fundamentals of Algorithmics. Englewood Cliffs, NJ: Prentice-Hall, 1996

# Apéndice A: Enunciado de las Prácticas

## Grados en Ingeniería Informática e Ingeniería de Computadores
## Asignatura *Algoritmos Avanzados*
## Curso 2019/2020
## Práctica nº 2

### Objetivo

El objetivo de la práctica es que el alumno practique con los algoritmos heurísticos.

### Carácter

La realización de la práctica es voluntaria, pero su entrega es un requisito para poder entregar la práctica 3b. Puede hacerse individualmente o en pareja.

### Enunciado

Un equipo de informáticos va contratando y realizando trabajos semanalmente. Para organizarse mejor, dividen los trabajos entre los que conllevan poca tensión y los que suponen gran tensión. Si realizan un trabajo de baja tensión en la semana *i*-ésima, cobran una cantidad $b_i>0$ euros mientras que por un trabajo de alta tensión cobran $a_i>0$ euros. Un trabajo de baja tensión puede contratarse en cualquier semana pero, para poder aguantar la tensión, un trabajo de alta tensión solamente se contrata para la semana *i* si han descansado en la semana *i*-1 (con la única excepción de la semana 0, en la cual puede escogerse un trabajo de alta tensión). Por tanto, en una semana cualquiera puede realizarse un trabajo de baja tensión, un trabajo de alta tensión o descansar.

El *problema del plan de trabajos estresantes de beneficio máximo* consiste en decidir, dada una secuencia de beneficios $b_i$ y $a_i$, qué secuencia de trabajos debe contratarse en *n* semanas para obtener un beneficio máximo. Aunque los trabajos de alta tensión suelen dar un beneficio mayor que los de baja tensión, no tiene por qué ser necesariamente así.

Por ejemplo, sean los siguientes beneficios para 5 semanas:

| | Semana 0 | Semana 1 | Semana 2 | Semana 3 | Semana 4 |
|---|---|---|---|---|---|
| *a* | 10 | 15 | 15 | 15 | 15 |
| *b* | 10 | 10 | 10 | 10 | 10 |

Un plan de trabajos con beneficio máximo sería tomar los cinco trabajos de baja tensión, con un beneficio igual a 10+10+10+10+10=50. Puede comprobarse que no hay ningún otro plan que proporcione un beneficio mayor.

Un algoritmo heurístico H1 elige trabajos mediante la comparación sucesiva, desde $i$=0 y cada dos semanas $i$ e $i$+1, entre los valores $b_i$+$b_{i+1}$ y $a_{i+1}$.

Sea el ejemplo anterior de nuevo. El algoritmo H1 elige los dos trabajos de baja tensión en las semanas 0 y 1 (con beneficio $b_0$+$b_1$=10+10=20, superior a $a_1$=15), vuelve a hacer lo mismo en las semanas 2 y 3 (de nuevo, con beneficio 10+10=20) y finalmente elegirá el trabajo de baja tensión $b_4$=10. El beneficio total es 50, que es un valor maximal.

El objetivo de la práctica es implementar al menos dos algoritmos heurísticos y usar OptimEx para comparar su optimalidad, medida como el porcentaje de casos en los que cada algoritmo calcula una solución optimal con respecto al otro algoritmo.

**Informe**

El alumno debe entregar un informe con la estructura que se detalla a continuación. El código de los algoritmos no debe enviarse en ficheros separados, sino integrarse en el texto del informe. Sólo se debe incluir el código de los métodos pedidos y, si es el caso, de los métodos auxiliares que aquéllos utilicen.

1. **Especificación del problema.** Debe desarrollarse una especificación del problema, identificando sus distintas partes: precondición y poscondición (formada a su vez por condición de validez y función objetivo). No hace falta utilizar notación matemática, pero la respuesta debe ser precisa y completa.

2. **Implementación de los algoritmos.** Implementar el algoritmo heurístico antes esbozado con la siguiente cabecera:

   ```
   public static int trabajosH1 (int[] a, int[] b)
   ```

   donde $a[i]$ es el precio del trabajo de alta tensión de la semana $i$-ésima, $0 \le i \le n-1$, y $b[i]$ es el precio del trabajo de baja tensión de la misma semana.

   También se pide idear e implementar uno o más algoritmos heurísticos adicionales, que sean "razonables". Asimismo, para cada nuevo algoritmo, se explicará claramente el criterio voraz usado y una justificación intuitiva de por qué dicho criterio puede conducir a soluciones optimales. La cabecera de estos nuevos algoritmos debe ser igual a la anterior (aunque, obviamente, con distintos identificadores de método).

3. **Experimentación con la optimidad de los algoritmos.** Se comparará con OptimEx la optimalidad de los algoritmos desarrollados. Se aconseja leer detalladamente la guía de usuario (incompleta) disponible en el campus virtual. Debe aportarse la siguiente información:

   - Rangos de valores usados para la generación aleatoria de los datos de entrada.
   - Decidir si algún algoritmo es exacto.

   - Evidencias en las que se basa la conclusión anterior: tabla de resumen numérico y diagramas del resumen gráfico.
   - Si algún algoritmo es inexacto, aportar dos contraejemplos, uno encontrado en el experimento y otro más sencillo deducido tras analizar las características de los contraejemplos encontrados experimentalmente. Explica las características del segundo contrajemplo que producen su suboptimalidad.

4. **Conclusiones.** Se presentan las conclusiones obtenidas tras realizar la práctica. Estas conclusiones pueden consistir en una valoración de los algoritmos heurísticos o cualquier comentario sobre la práctica. Por ejemplo, pueden describirse las incidencias que han dificultado la realización de la práctica, sus aspectos más atractivos o más difíciles, sugerencias sobre cómo mejorar la práctica, etc.

**Entrega**

El alumno debe entregar el informe por medio del apartado de Evaluación del campus virtual. Si se tienen dificultades, puede enviarse por el correo del campus virtual con el asunto “Práctica 2”. El plazo de entrega es el domingo 6 de octubre de 2019, incluido.

**Evaluación**

Se evaluará la calidad y claridad de todos los apartados del informe.

# Grados en Ingeniería Informática e Ingeniería de Computadores
# Asignatura *Algoritmos Avanzados*
# Curso 2019/2020
# Práctica nº 3a

**Objetivo**

El objetivo de la práctica es que el alumno profundice en su conocimiento de las técnicas de búsqueda en espacios de estados.

**Carácter**

La realización de la práctica es voluntaria. Puede hacerse individualmente o en pareja.

**Enunciado**

En la práctica 2 se planteó el *problema del plan de trabajos estresantes de beneficio máximo*. Se dispone de ofertas de trabajos para $n$ semanas, unos de alta tensión y beneficio $b_i$ y otros de baja tensión y beneficio $a_i$, $0 \le i \le n-1$. Recuérdese que el enunciado del problema imponía restricciones para la contratación de un trabajo de alta tensión. El objetivo consiste en contratar una secuencia de trabajos a contratar para las $n$ semanas de forma que el beneficio obtenido sea máximo.

Por ejemplo, dados los siguientes beneficios para 5 semanas:

| | Semana 0 | Semana 1 | Semana 2 | Semana 3 | Semana 4 |
|---|---|---|---|---|---|
| *a* | 10 | 15 | 15 | 15 | 15 |
| *b* | 10 | 10 | 10 | 10 | 10 |

un plan de trabajos con beneficio máximo consiste en contratar los cinco trabajos de baja tensión, con un beneficio igual a 10+10+10+10+10=50.

El objetivo de la práctica es desarrollar *de forma sistemática* un algoritmo de vuelta atrás que resuelva el problema planteado y comprobar experimentalmente su optimalidad.

**Informe**

El alumno debe entregar un informe con la estructura que se detalla a continuación. El código de los algoritmos no debe enviarse en ficheros separados, sino integrarse en el texto del informe. Sólo se debe incluir el código de los métodos pedidos y, si es el caso, de los métodos auxiliares que aquéllos utilicen.

1. **Técnica de vuelta atrás.** Constará de varios apartados, que mostrarán el desarrollo sistemático de un algoritmo de vuelta atrás:

a) Diseño de un árbol de búsqueda adecuado para resolver el problema. Debe incluirse una figura del mismo, acompañada de una explicación breve y clara, en función de los parámetros del problema, de: (1) el número de niveles del árbol, y (2) los candidatos existentes en cada nodo del árbol. Si el árbol es muy grande, basta con mostrar una parte, siempre que sea fácil deducir la parte omitida.
b) Comprobación de validez. Debe explicarse breve y claramente la comprobación de validez de la solución parcial realizada en cada nodo del árbol de búsqueda.
c) Código de un algoritmo de vuelta atrás, que incorpore el diseño anterior. Al igual que en la práctica 2, la cabecera del método principal del algoritmo debe ser:

```
public static int trabajosVA (int[] a, int[] b)
```

donde $a[i]$ es el precio del trabajo de alta tensión de la semana $i$-ésima, $0 \le i \le n-1$, y $b[i]$ es el precio del trabajo de baja tensión de la misma semana.

Para el ejemplo anterior, la llamada del método principal será *trabajosVA*({10,15,15,15,15},{10,10,10,10,10}). El algoritmo debe devolver el beneficio de un plan óptimo de trabajos; opcionalmente, puede imprimir dicho plan óptimo.

2. **Comparación de optimalidad.** Se añadirá el algoritmo de vuelta atrás a la clase utilizada en la práctica 2 (con dos algoritmos heurísticos, al menos) y se comparará la optimalidad de todos los algoritmos. Obsérvese que el algoritmo de vuelta atrás debe ser exacto, es decir, debería calcular resultados óptimos en el 100% de los casos. Si no fuera así, deberían revisarse todos los algoritmos para identificar y corregir los errores, y repetir el experimento hasta que se obtengan los resultados predichos por la teoría. (En el manual de OptimEx puede consultarse un catálogo de situaciones conflictivas y su resolución.) Se pide:

   a) Material del experimento. Se debe:
      – Identificar cada algoritmo, indicando la técnica de diseño con la que se ha desarrollado y, si es necesario, algún elemento diferenciador (p.ej. la función de selección de cada algoritmo heurístico).
      – Indicar los rangos de valores usados para la generación aleatoria de los datos de entrada.
   b) Conclusión. Se dirá qué algoritmos son exactos según los resultados de la experimentación.
   c) Evidencias. Deben aportarse los resultados recogidos en la tabla de resumen numérico y en los diagramas de resumen gráfico, explicando su significado.
   d) Incidencias (opcional). Si durante la realización de esta práctica ha sido necesario revisar y modificar algún algoritmo de prácticas anteriores, se explicará por qué se realizó dicha modificación y en qué consistió.

3. **Conclusiones.** Se explican las conclusiones obtenidas tras realizar la práctica. Estas conclusiones pueden consistir en una valoración de las técnicas de búsqueda o cualquier otro comentario sobre la práctica (incidencias que han dificultado la realización de la práctica, sus aspectos más atractivos o más difíciles, sugerencias sobre cómo mejorar la práctica, etc.)

**Entrega**

El alumno debe entregar el informe por medio del apartado de Evaluación del campus virtual. Si se tienen dificultades, puede enviarse por el correo del campus virtual con el asunto “Práctica 3a”. El plazo de entrega del informe es el domingo 20 de octubre de 2019, incluido.

**Evaluación**

Se evaluará la calidad y claridad de todos los apartados del informe.

# Grados en Ingeniería Informática e Ingeniería de Computadores
# Asignatura *Algoritmos Avanzados*
# Curso 2019/2020
# Práctica nº 3b

**Objetivo**

El objetivo de la práctica es que el alumno profundice en su conocimiento de las técnicas de búsqueda en espacios de estados.

**Carácter**

La realización de la práctica es voluntaria. Debe hacerse en el mismo grupo que la práctica 3a (es decir, individualmente o con el mismo compañero).

**Enunciado**

En la parte (a) de la práctica 3 se pidió desarrollar de forma sistemática un algoritmo de vuelva atrás que resolviera el *problema del plan de trabajos estresantes de beneficio máximo* y comprobar su optimalidad. El objetivo de esta segunda parte es similar, pero referido a un algoritmo de ramificación y poda. Obsérvese que ambos algoritmos calcularán el mismo resultado, aunque previsiblemente de forma más eficiente en el caso de ramificación y poda.

**Informe**

El alumno debe entregar un informe con la estructura que se detalla a continuación. El código de los algoritmos no debe enviarse en ficheros separados, sino integrarse en el texto del informe. Sólo se debe incluir el código de los métodos pedidos y, si es el caso, de los métodos auxiliares que aquéllos utilicen.

1. **Técnica de vuelta atrás (opcional).** Este apartado sólo se rellenará si se realiza alguna modificación en el algoritmo presentado para la práctica 3a. En este caso, deben explicarse breve y claramente los cambios realizados usando los apartados previstos:
   a) Árbol de búsqueda.
   b) Condición de validez.
   c) Código del algoritmo de vuelta atrás.
2. **Técnica de ramificación y poda.** Partiendo del algoritmo de vuelta atrás anterior, se pide:

   a) Función de cota para este problema. Debe proponerse al menos una función de cota. Para cada una, debe proporcionarse: (1) definición de la función de cota en términos de los parámetros del problema, (2) valor inicial de la cota, y (3) cómo se actualiza su valor en cada nodo del árbol de búsqueda.
   b) Código de un algoritmo basado en la técnica de ramificación y poda, que incorpore una de las cotas anteriores. El algoritmo debe tener la misma cabecera para el método principal que el algoritmo de vuelta atrás.

3. **Comparación de optimalidad.** Se añadirá el algoritmo de ramificación y poda a los algoritmos de la clase utilizada en la práctica 3a y se comparará la optimalidad de todos los algoritmos. Obsérvese que todos los algoritmos de búsqueda deben ser exactos, es decir, deberían calcular resultados óptimos en el 100% de los casos. Si no fuera así, deberían revisarse todos los algoritmos para identificar y corregir los errores, y repetir el experimento hasta que se obtengan los resultados predichos por la teoría. (En el manual de OptimEx puede consultarse un catálogo de situaciones conflictivas y su resolución.) Se pide:

   a) Material del experimento. Se debe:
      – Identificar cada algoritmo, indicando la técnica de diseño con la que se ha desarrollado y, si es necesario, algún elemento diferenciador (p.ej. la función de selección de cada algoritmo heurístico).
      – Indicar los rangos de valores usados para la generación aleatoria de los datos de entrada.
   b) Conclusión. Se dirá qué algoritmos son exactos según los resultados de la experimentación.
   c) Evidencias. Deben aportarse los resultados recogidos en la tabla de resumen numérico y en los diagramas de resumen gráfico, explicando su significado.
   d) Incidencias (opcional). Si durante la realización de esta práctica ha sido necesario revisar y modificar algún algoritmo de prácticas anteriores, se explicará por qué se realizó dicha modificación y en qué consistió.

4. **Comparación de eficiencia en tiempo.** Se repetirá el experimento con los mismos algoritmos y datos que en el apartado anterior, pero seleccionando en OptimEx el criterio de eficiencia en tiempo (en lugar de optimalidad), es decir, se compararán los tiempos de ejecución de los algoritmos. Se pide:

   a) Conclusión. Se comentarán breve y claramente las diferencias en tiempo de ejecución de los algoritmos, según los resultados de la experimentación.
   b) Evidencias. Deben aportarse los tiempos medidos en la tabla de resumen numérico y en los diagramas de resumen gráfico, explicando su significado.

5. **Conclusiones.** Se explican las conclusiones obtenidas tras realizar la práctica. Estas conclusiones pueden consistir en una valoración de las técnicas de búsqueda o cualquier otro comentario sobre la práctica (incidencias que han dificultado la realización de la práctica, sus aspectos más atractivos o más difíciles, sugerencias sobre cómo mejorar la práctica, etc.)

**Entrega**

El alumno debe entregar el informe por medio del apartado de Evaluación del campus virtual. Si se tienen dificultades, puede enviarse por el correo del campus virtual con el asunto “Práctica 3b”. El plazo de entrega del informe es el lunes 28 de octubre de 2019, incluido.

**Evaluación**

Se evaluará la calidad y claridad de todos los apartados del informe.

# Grados en Ingeniería Informática e Ingeniería de Computadores
# Asignatura *Algoritmos Avanzados*
# Curso 2019/2020
# Práctica nº 5

**Objetivo**

El objetivo de la práctica es que el alumno profundice en su conocimiento de la técnica de programación dinámica.

**Carácter**

La realización de la práctica es voluntaria. Puede hacerse individualmente o en pareja.

**Enunciado**

En la práctica 2 se planteó el *problema del plan de trabajos estresantes de beneficio máximo*. Se dispone de ofertas de trabajos para $n$ semanas, unos de alta tensión y beneficio $b_i$ y otros de baja tensión y beneficio $a_i$, $0 \leq i \leq n-1$. Recuérdese que el enunciado del problema imponía restricciones para la contratación de un trabajo de alta tensión. El objetivo consiste en contratar una secuencia de trabajos a contratar para las $n$ semanas de forma que el beneficio obtenido sea máximo.

Por ejemplo, dados los siguientes beneficios para 5 semanas:

| | Semana 0 | Semana 1 | Semana 2 | Semana 3 | Semana 4 |
|---|---|---|---|---|---|
| *a* | 10 | 15 | 15 | 15 | 15 |
| *b* | 10 | 10 | 10 | 10 | 10 |

un plan de trabajos con beneficio máximo consiste en contratar los cinco trabajos de baja tensión, con un beneficio igual a 10+10+10+10+10=50.

El objetivo de la práctica es desarrollar *de forma sistemática* un algoritmo de programación dinámica que resuelva el problema planteado y comprobar experimentalmente su optimalidad y su eficiencia en tiempo.

**Informe**

El alumno debe entregar un informe con la estructura que se detalla a continuación. El código de los algoritmos no debe enviarse en ficheros separados, sino integrarse en el texto del informe. Sólo se debe incluir el código de los métodos pedidos y, si es el caso, de los métodos auxiliares que aquéllos utilicen.

1. **Algoritmo recursivo.** Se incluirá un algoritmo recursivo codificado en Java que resuelva el problema. Debe ir acompañado de una breve explicación de cómo

organiza la construcción de la solución. El algoritmo debe tener la misma cabecera para el método principal que en prácticas anteriores.

Opcionalmente pueden presentarse varios algoritmos recursivos (por ejemplo, uno con una formulación hacia adelante y otro hacia atrás), en cuyo caso debe identificarse el que se usa para el resto de la práctica.

2. **Tabulación.** Se incluirá:
   a) Un árbol de recursión y su grafo de dependencia correspondiente. Ambos gráficos deben ser representativos del comportamiento del algoritmo.
   b) Decisiones de diseño para la tabulación. Debe aportarse:
      – Representación gráfica de una tabla adecuada para tabular el algoritmo recursivo, de forma que se aprecie claramente: el número de celdas en cada dimensión y el subproblema que se almacenará en cada celda de la tabla. Si se considera conveniente, puede incluirse la declaración en Java de la tabla, expresada en función de los parámetros.
      – Explicación de un orden secuencial a seguir para rellenar las celdas de la tabla.
   c) Código del algoritmo de programación dinámica resultante de utilizar las anteriores decisiones de diseño de la tabulación.
   d) Análisis de complejidad en tiempo y en espacio del algoritmo.
3. **Determinación de decisiones.** Se ampliará el algoritmo del apartado anterior de forma que, junto al valor del beneficio óptimo, imprima los trabajos correspondientes.
4. **Comparación de optimalidad.** Se integrarán los dos algoritmos de los apartados 1 y 2 en la clase de Java utilizada en la práctica 3 (o en la práctica 2 si la práctica 3 no se realizó). Obsérvese que ambos algoritmos deben ser exactos, es decir, deben dar resultados óptimos en el 100% de los casos. Si no fuera así, deberían revisarse todos los algoritmos para identificar y corregir los errores, y repetir el experimento hasta que se obtengan los resultados predichos por la teoría. (En el manual de OptimEx puede consultarse un catálogo de situaciones conflictivas y su resolución.) Se pide:
   a) Material del experimento. Se debe:
      – Identificar cada algoritmo, indicando la técnica de diseño con la que se ha desarrollado y, si es necesario, algún elemento diferenciador (p.ej. la función de selección de cada algoritmo heurístico).
      – Indicar los rangos de valores usados para la generación aleatoria de los datos de entrada.
   b) Conclusión. Se dirá qué algoritmos son exactos según los resultados de la experimentación.
   c) Evidencias. Deben aportarse los resultados recogidos en la tabla de resumen numérico y en los diagramas de resumen gráfico, explicando su significado.
   d) Incidencias (opcional). Si durante la realización de esta práctica ha sido necesario revisar y modificar algún algoritmo de prácticas anteriores, se explicará por qué se realizó dicha modificación y en qué consistió.

5. **Comparación de eficiencia.** Se repetirá el experimento con los mismos datos que en el apartado anterior, pero según el criterio de eficiencia (en lugar de optimalidad), es decir, se compararán los tiempos de ejecución de los algoritmos. Se pide:
    a) Conclusión. Se comentarán breve y claramente las diferencias en tiempo de ejecución de los algoritmos, según los resultados de la experimentación.
    b) Evidencias. Deben aportarse los tiempos medidos en la tabla de resumen numérico y en los diagramas de resumen gráfico, explicando su significado.
6. **Conclusiones.** Se explican las conclusiones obtenidas tras realizar la práctica. Estas conclusiones pueden consistir en una valoración de la técnica de programación dinámica o cualquier otro comentario sobre la práctica (incidencias que han dificultado la realización de la práctica, sus aspectos más atractivos o más difíciles, sugerencias sobre cómo mejorar la práctica, etc.)

**Entrega**

El alumno debe entregar el informe por medio del apartado de Evaluación del campus virtual. Si se tienen dificultades, puede enviarse por el correo del campus virtual con el asunto “Práctica 5”. El plazo de entrega informe es el martes 3 de diciembre de 2019, incluido.

**Evaluación**

Se evaluará la calidad y claridad de todos los apartados del informe.

# Grados en Ingeniería Informática e Ingeniería de Computadores
# Asignatura *Algoritmos Avanzados*
# Curso 2019/2020
# Práctica nº 6

**Objetivo**

El objetivo de la práctica es que el alumno practique con los algoritmos probabilistas.

**Carácter**

La realización de la práctica es voluntaria. Puede hacerse individualmente o en pareja.

**Enunciado**

Sea la versión de minimización del *problema del llenado de cajas*. Se dispone de varios objetos, cada uno con un peso $p_i$, y un suministro ilimitado de cajas de capacidad $c$. El objetivo consiste en introducir los objetos en un número mínimo de cajas.

Por ejemplo, sean cinco objetos de pesos {5,2,4,1,8} y una capacidad $c$=10 de las cajas. Pueden introducirse los objetos en 2 cajas si en la primera se meten los objetos de pesos {5,4,1} y en la segunda, los objetos de pesos {2,8}.

Pueden encontrarse en el Aula Virtual varios algoritmos que resuelven este problema, tanto en el paquete de algoritmos aproximados como en el paquete de algoritmos de búsqueda. La cabecera de estos métodos es:

```
public static int menosCajas (int[] ps, int c)
```

Dado que es un problema de minimización, una forma de abordar el problema consiste en comenzar sin ninguna caja y que se vaya usando una caja nueva cada vez que sea necesario. Cada objeto nuevo podría introducirse en una caja ya disponible que tenga suficiente espacio libre o en una caja nueva. El problema puede resolverse de forma probabilista si la decisión de dónde introducir cada objeto se toma al azar entre una caja nueva o cualquiera de las cajas ya disponibles y con capacidad libre suficiente.

El objetivo de la práctica es implementar al menos un algoritmo probabilista que resuelva el problema y usar OptimEx para comparar su optimalidad.

**Informe**

El alumno debe entregar un informe con la estructura que se detalla a continuación. El código de los algoritmos no debe enviarse en ficheros separados, sino integrarse en el

texto del informe. Sólo se debe incluir el código de los métodos pedidos y, si es el caso, de los métodos auxiliares que aquéllos utilicen.

1. **Desarrollo del algoritmo.** Explicar e implementar el algoritmo probabilista antes esbozado, respetando la cabecera anterior.
2. **Experimentación con la optimidad del algoritmo.** Se comparará con OptimEx la optimalidad del algoritmo probabilista y algunos algoritmos disponibles en el Aula Virtual para este problema (al menos, los algoritmos heurísticos y el algoritmo de vuelta atrás). Debe aportarse la siguiente información:
   a) Material del experimento. Se debe:
      – Identificar cada algoritmo, indicando la técnica de diseño con la que se ha desarrollado y, si es necesario, algún elemento diferenciador (p.ej. la función de selección de cada algoritmo heurístico).
      – Indicar los rangos de valores usados para la generación aleatoria de los datos de entrada.
   b) Conclusión. Se dirá qué algoritmos son exactos según los resultados de la experimentación.
   c) Evidencias. Deben aportarse los resultados recogidos en la tabla de resumen numérico y en los diagramas de resumen gráfico, explicando su significado.
   d) Incidencias (opcional). Si durante la realización de esta práctica ha sido necesario revisar y modificar algún algoritmo, se explicará por qué se realizó dicha modificación y en qué consistió.
3. **Mejora (opcional).** Pueden aportarse ideas de cómo mejorar la optimalidad del algoritmo probabilista anterior o de algoritmos probabilistas basados en un diseño distinto, e incluso programarlos y medir su optimalidad.
4. **Conclusiones.** Se presentan las conclusiones obtenidas tras realizar la práctica. Estas conclusiones pueden consistir en una valoración de los algoritmos heurísticos o cualquier comentario sobre la práctica. Por ejemplo, pueden describirse las incidencias que han dificultado la realización de la práctica, sus aspectos más atractivos o más difíciles, sugerencias sobre cómo mejorar la práctica, etc.

**Entrega**

El alumno debe entregar el informe por medio del apartado de Evaluación del campus virtual. Si se tienen dificultades, puede enviarse por el correo del campus virtual con el asunto “Práctica 6”. El plazo de entrega es el domingo 15 de diciembre de 2019, incluido. No habrá segunda entrega.

**Evaluación**

Se evaluará la calidad y claridad de todos los apartados del informe.