Departamento de Informática
Universidad Carlos III de Madrid

Grado en Informática

Heurística y Optimización

Junio 2013

Normas generales del examen

- El tiempo para realizar el examen es de **4 horas**
- Cada pregunta debe responderse en páginas separadas
- No se responderá a ninguna pregunta sobre el examen
- Si se sale del aula, no se podrá volver a entrar durante el examen
- No se puede presentar el examen escrito a lápiz

Problema 1. (2 puntos)

Un *portfolio* consiste en una combinación de algoritmos A, B, ... que se ejecutan en secuencia para resolver un problema en particular. Si un algoritmo no encuentra solución al problema, entonces se invoca el siguiente y así sucesivamente hasta que, o bien uno de los algoritmos del portfolio resuelve el problema (en cuyo caso el portfolio sale con éxito), o se han agotado todos los algoritmos en el portfolio sin encontrar una solución —en cuyo caso, el portfolio termina con fallo.

Se dispone de tres algoritmos A, B y C. La Tabla siguiente muestra qué problemas resuelve cada algoritmo sobre un conjunto de cinco problemas donde ✓ representa que el problema de la columna correspondiente es resoluble por el algoritmo indicado en la fila y ✗ que no lo es.

Algoritmo	1	2	3	4	5
A	✓	✗	✓	✓	✗
B	✗	✓	✓	✗	✓
C	✓	✓	✗	✗	✗

Se pide configurar automáticamente un portfolio que *resuelva los cinco problemas con el menor número de algoritmos* usando técnicas de Programación Lineal. Responde razonadamente las siguientes preguntas:

1. **(0,5 puntos)** Determina las variables de decisión indicando claramente su significado
2. **(0,5 puntos)** Determina la función objetivo del problema de optimización
3. **(0,5 puntos)** Determina las restricciones del problema de optimización
4. **(0,5 puntos)** Resuelve el problema anterior. ¿Cuál es la solución óptima?

Problema 2. (3 puntos)

Considera el siguiente problema de Programación Lineal:

$$\max z = x_1 + 2x_2 - x_3 + x_4$$

$$\begin{array}{rrrrrcr}
-\,x_1 & -\,x_2 & -3x_3 & -x_4 & & = & -8 \\
-\,x_1 & +2x_2 & +\,x_3 & & & = & 6 \\
2x_1 & & +2x_3 & & +x_5 & = & 5
\end{array}$$

$$\mathbf{x} \geq \mathbf{0}$$

y responde razonadamente las siguientes preguntas:

1. **(0,5 puntos)** Expresa el problema de Programación Lineal anterior en forma *estándar* de maximización.
2. **(0,5 puntos)** Prepara el problema de Programación Lineal que ha resultado del apartado anterior para su aplicación con el algoritmo Simplex para garantizar que pueda comenzarse su aplicación con la matriz identidad.
3. **(1 punto)** Resuelve el problema de Programación Lineal obtenido en el apartado anterior con el algoritmo Simplex. Indica claramente, en cada iteración: las variables escogidas en la base, su valor, y el valor de la función objetivo.
4. **(0,5 puntos)** Interpreta el resultado y explica qué conclusiones pueden extraerse de él.
5. **(0,5 puntos)** ¿Qué debería hacerse para calcular la contribución al incremento de la función objetivo por unidad de recurso?

Problema 3. (2 puntos)

Considérense las siguientes etapas en la construcción de una vivienda, identificadas por una letra y para cada una de las cuales se indica su duración:

	Etapa	Duración
A	Excavación y cimentación	8
B	Cerramientos y fachadas	7
C	Tabiques interiores	5
D	Carpintería en puertas y ventanas	3
E	Pintura	4

Se sabe, además, que la construcción de una vivienda está sujeta a las siguientes *restricciones*:

R1 *Cerramientos y fachadas* no pueden iniciarse antes de concluir la *Excavación y cimentación*

R2 La *Carpintería en puertas y ventanas* debe empezarse como muy tarde antes de acabar *Tabiques interiores*

R3 La *Pintura* debe iniciarse como muy tarde dos días antes de acabar *Cerramientos y fachadas*

Se desea construir una nueva vivienda de modo que ninguna de estas tareas tarde más de 20 días en iniciarse. Se pide resolver razonadamente las siguientes cuestiones:

1. **(1 punto)** Representar el problema formalmente como un problema de *satisfacción de restricciones*
2. **(0,5 puntos)** Aplicar *arco-consistencia* a las variables involucradas en la restricción R1 y determinar los dominios resultantes
3. **(0,5 puntos)** Aplicar *camino-consistencia* a las variables involucradas en las restricciones R1 y R3 y determinar los dominios resultantes

Problema 4. (3 puntos)

En las investigaciones sobre la historia evolutiva del genoma, cada vez está cobrando más importancia el estudio de las permutaciones con inversiones. Una permutación es una secuencia de símbolos que, en este caso, consistirán en letras del alfabeto como se muestra a continuación:

$$\mathcal{ABCDEFGH}$$

Se ha observado que los cromosomas mutan con frecuencia un intervalo de genes (representados por letras), invirtiendo su orden. Estas inversiones ocurren siempre a partir de la primera posición y afectan a los primeros k símbolos. Por ejemplo, las siguientes permutaciones resultarían de inversiones con $k = 2$, $k = 3$ y $k = 4$ a partir de la permutación anterior:

$$\mathcal{BACDEFGH} \qquad \mathcal{CBADEFGH} \qquad \mathcal{DCBAEFGH}$$

Con el objeto de que las inversiones puedan ocurrir en cualquier posición, es posible desplazar circularmente las permutaciones hacia la derecha o la izquierda. Por ejemplo, las siguientes permutaciones resultan de desplazar circularmente la primera permutación de las anteriores ($\mathcal{BACDEFGH}$) hacia la derecha una, dos y tres posiciones:

$$\mathcal{HBACDEFG} \qquad \mathcal{GHBACDEF} \qquad \mathcal{FGHBACDE}$$

El cálculo del número mínimo de movimientos (desplazamientos e inversiones para un mismo valor de k) para convertir una permutación de cualquier longitud en otra —como la primera de todas, esto es, con todos los símbolos ordenados— es de vital importancia para explicar la evolución de algunos organismos como la drosófila, o algunos virus y, en el futuro, de otros seres vivos de mayor complejidad.

Se pide:

1. **(0,5 puntos)** Formalizar el espacio de estados. En particular, modelizar los operadores para permutaciones de cualquier longitud con cualquier valor de k.

2. **(0,5 puntos)** ¿Cuál es el tamaño del espacio de estados para el caso $k = 2$? Razona tu respuesta.

3. **(1 punto)** Definir una función heurística $h(n)$ que sea *admisible* e *informada* para $k = 2$ explicando claramente su construcción.

En algunos casos, se considera que los desplazamientos circulares a derecha e izquierda ocurren sin ningún esfuerzo adicional, por lo que su coste es nulo. En este supuesto y sabiendo que se ha decidido usar un algoritmo de el mejor primero para resolver el problema:

4. **(0,5 puntos)** ¿Es preciso reformular la modelización de operadores del apartado 1? Si es así, ¿cómo? Razona tu respuesta

5. **(0,5 puntos)** ¿Sigue siendo *admisible* la función heurística del apartado 3? Si o no, y por qué.

Soluciones del examen de Heurística y Optimización
Junio 2013

Solución al problema 1

1. Para determinar el menor número de algoritmos que formarán parte del portfolio es preciso saber cuáles se incorporarán y cuáles no. Por lo tanto, las variables de decisión son:

   $$\begin{array}{lcl} x_A & \rightleftharpoons & \text{El algoritmo } A \text{ es parte del portfolio} \\ x_B & \rightleftharpoons & \text{El algoritmo } B \text{ es parte del portfolio} \\ x_C & \rightleftharpoons & \text{El algoritmo } C \text{ es parte del portfolio} \end{array}$$

   de modo que si x_i toma el valor 1 entonces formará parte del portfolio. En otro caso, x_i tomará el valor 0.

   En particular, se trata de un problema de Programación Entera 0-1

2. El objetivo del problema de optimización consiste en minimizar el número de algoritmos a usar por el portfolio o, equivalentemente a partir de las definiciones del apartado anterior, en minimizar la suma de x_i:

   $$\text{mín } z = \sum_{i\in\{A,B,C\}} x_i$$

3. Las restricciones del problema se dividen en dos grupos:

   - Primero, las restricciones identificadas en el apartado 1 que indican que se trata de un problema de Programación Entera 0-1:

     $$x_i \in \{0,1\} \quad \forall i \in \{A,B,C\}$$

   - Segundo, las que resultan de asegurar que la selección de algoritmos resolverá los cinco problemas indicados en el enunciado.

     Esta modelización es muy sencilla y se hace en base a reglas como la siguiente: para resolver el primer problema es preciso escoger el algoritmo A o C o, en otras palabras, $x_A + x_C \geq 1$. Equivalentemente, para resolver el segundo problema es preciso escoger el segundo o tercer algoritmo, esto es, $x_B + x_c \geq 1$.

     A continuación se muestran las restricciones que resultan de considerar la resolución de todos los problemas:

     $$\begin{array}{ccccccc} x_A & & & + & x_C & \geq & 1 \\ & & x_B & + & x_C & \geq & 1 \\ x_A & + & x_B & & & \geq & 1 \\ x_A & & & & & \geq & 1 \\ & & x_B & & & \geq & 1 \end{array}$$

4. A estas alturas, el problema de Programación Lineal Entera 0-1 a resolver está completamente caracterizado:

   $$\text{mín } z = \sum_{i\in\{A,B,C\}} x_i$$

   $$\begin{array}{ccccccc} x_A & & & + & x_C & \geq & 1 \\ & & x_B & + & x_C & \geq & 1 \\ x_A & + & x_B & & & \geq & 1 \\ x_A & & & & & \geq & 1 \\ & & x_B & & & \geq & 1 \end{array}$$

   $$x_i \in \{0,1\} \quad \forall i \in \{A,B,C\}$$

   Para resolver este problema no es preciso aplicar ninguna de las técnicas de ramificación y acotación que se usan en estos casos. En su lugar basta con observar que la cuarta restricción *domina* la primera y tercera de modo que éstas pueden eliminarse. Asimismo, la última restricción domina las restricciones segunda y tercera. Cancelanado también estas restricciones, el problema de Programación Lineal resultante es:

$$\min z = \sum_{i\in\{A,B,C\}} x_i$$

$$\begin{array}{lcl} x_A & \geq & 1 \\ \quad x_B & \geq & 1 \end{array}$$

$$x_i \in \{0,1\} \quad \forall i \in \{A,B,C\}$$

cuya única solución es $x_A = x_B = 1$.

De hecho, una observación detallada de la tabla del enunciado revela que: uno, no existe ningún algoritmo que resuelva todos los problemas; dos, que la única combinación de dos algoritmos que resuelve todos los problemas son precisamente A y el B como sugiere la resolución mostrada.

Solución al problema 2

Un problema parecido (donde desigualdades de distinto tipo han sido sustituidas todas por igualdades) puede encontrarse en Sixto Ríos Insua. *Investigación Operativa – Optimización*. Editorial Centro de Estudios Ramón Areces, página 102.

1. En el primer apartado se pedía expresar el problema de Programación Lineal del enunciado en *forma estándar* (de maximización), precisamente en preparación a su resolución con el algoritmo SIMPLEX.

   Un problema de programación lineal está en forma *estándar* si todas las restricciones son de igualdad, las variables de decisión son no negativas y, por último, el vector de constantes o recursos $\mathbf{b}$ no contiene términos negativos. Estará, además, en forma de maximización si la función objetivo maximiza y de minimización en otro caso. El problema, tal y como estaba enunciado, ya verifica todas estas condiciones salvo la tercera puesto que el recurso de la primera restricción es -8.

   Por lo tanto, el problema de Programación Lineal queda, como sigue, en forma estándar de maximización:

$$\max z = x_1 + 2x_2 - x_3 + x_4$$

$$\begin{array}{rrrrrcl} x_1 & +x_2 & +3x_3 & +x_4 & & = & 8 \\ -x_1 & +2x_2 & +x_3 & & & = & 6 \\ 2x_1 & & +2x_3 & & +x_5 & = & 5 \end{array}$$

$$\mathbf{x} \geq \mathbf{0}$$

   donde simplemente se ha multiplicado la primera restricción por -1

2. Para poder empezar a aplicar el algoritmo SIMPLEX es preciso disponer de una *base factible inicial* que, por lo tanto, debe verificar:

   a) La base elegida B debe ser invertible. Esto es $|B| \neq 0$

   b) El producto de la inversa de la base, B^{-1}, por el vector de recursos $\mathbf{b}$ debe dar valores no negativos

   Normalmente, una buena elección para iniciar la aplicación del algoritmo SIMPLEX consiste en elegir la matriz identidad (de la dimensión adecuada) puesto que: primero, es evidentemente invertible; segundo, su inversa, siendo igual a sí misma y multiplicada por el vector de recursos $\mathbf{b}$ sólo puede dar términos no negativos si el problema está en forma estándar porque, como se explicó en el primer apartado, el vector de recursos debe consistir únicamente de términos no negativos.

   Ahora bien, no existe ninguna selección de vectores columna a_i en la matriz de coeficientes del problema de Programación Lineal obtenido en el apartado anterior que sea igual a la matriz identidad de dimension 3, I_3. Sin embargo, añadiendo una variable artificial, x_6, a la segunda restricción resulta el problema de Programación Lineal:

$$\text{máx}\, z = x_1 + 2x_2 - x_3 + x_4 - \infty x_6$$

$$\begin{array}{rrrrrrcl}
x_1 & +x_2 & +3x_3 & +x_4 & & & = & 8 \\
-x_1 & +2x_2 & +x_3 & & & +x_6 & = & 6 \\
2x_1 & & +2x_3 & & +x_5 & & = & 5
\end{array}$$

$$\mathbf{x} \geq \mathbf{0}$$

donde, como puede verse, la variable artificial está infinitamente penalizada en la función objetivo.

Ahora es posible elegir la base formada por las variables $\{x_4, x_6, x_5\}$ que forman la matriz identidad de dimension 3. Si trabajar con las variables en un orden diferente del natural no fuera cómodo, también pueden permutarse las restricciones segunda y tercera y entonces la base elegida sería la base $\{x_4, x_5, x_6\}$ con idéntico resultado.

3. El algoritmo del SIMPLEX consiste en la aplicación iterativa de tres pasos: cálculo de las variables básicas, selección de la variable de entrada y selección de la variable de salida hasta que se detecte alguna de las siguientes condiciones:

   - El problema puede mejorar el valor de la función objetivo indefinidamente. Se dice entonces que el problema está *no acotado.* Este caso se detecta cuando todas las componentes y_i de la variable de decisión x_i elegida para entrar en la base son todos no positivos.
   - El problema es irresoluble. Esto ocurre cuando en el segundo paso, todos los costes reducidos son positivos y el primer paso asignó un valor no negativo a alguna variable artificial.
   - Se alcanza una solución factible y puede demostrarse que no es posible mejorarla. Esta condición se detecta como en el segundo caso pero cuando las variables artificiales (si las hubiera) tienen valores nulos.

Paso 0 Cálculo de una solución factible inicial

a) Cálculo de las variables básicas
Inicialmente, la base está formada por las variables x_4, x_5 y x_6 pero en el orden adecuado $B_0 = \{x_4, x_6, x_5\}$ para que $B_0 = I_3$ como se requería originalmente:

$$B_0 = I_3 \qquad\qquad B_0^{-1} = I_3$$

$$x_0^* = B_0^{-1} b = b = \begin{pmatrix} 8 \\ 6 \\ 5 \end{pmatrix} \qquad z_0^* = c_{B_0}^T x_0^* = \begin{pmatrix} 1 & -\infty & 0 \end{pmatrix} \begin{pmatrix} 8 \\ 6 \\ 5 \end{pmatrix} = -\infty$$

b) Selección de la variable de entrada
En las expresiones siguientes el cálculo de los vectores y_i se ha embebido en el cálculo de los *costes reducidos* directamente:

$$z_1 - c_1 = c_{B_0}^T B_0^{-1} a_1 - c_1 = \begin{pmatrix} 1 & -\infty & 0 \end{pmatrix} I_3 \begin{pmatrix} 1 \\ -1 \\ 2 \end{pmatrix} - 1 = +\infty$$

$$z_2 - c_2 = c_{B_0}^T B_0^{-1} a_2 - c_2 = \begin{pmatrix} 1 & -\infty & 0 \end{pmatrix} I_3 \begin{pmatrix} 1 \\ 2 \\ 0 \end{pmatrix} - 2 = -2\infty - 1$$

$$z_3 - c_3 = c_{B_0}^T B_0^{-1} a_3 - c_3 = \begin{pmatrix} 1 & -\infty & 0 \end{pmatrix} I_3 \begin{pmatrix} 3 \\ 1 \\ 2 \end{pmatrix} + 1 = -\infty + 4$$

c) Selección de la variable de salida
La regla de salida establece que debe salir aquella variable con el menor cociente x_i/y_{ij} donde x_i es la variable elegida en el paso anterior para añadirse a la base y $0 \leq j < 3$ puesto que la base tiene dimensión 3:

$$\min\left\{\frac{8}{1},\frac{6}{2},\cancel{\frac{5}{0}}\right\}$$

y sale la variable x_6 (esto es, la que ocupa el segundo lugar en la base) como cabía esperar puesto que al ser una variable artificial tiene una penalización arbitrariamente grande en la función de coste.

Paso 1 Mejora de la solución actual (iteración #1)

a) Cálculo de las variables básicas
En el segundo paso del SIMPLEX, la base es $B_2 = \{x_2, x_4, x_5\}$

$$B_1 = \begin{pmatrix} 1 & 1 & 0 \\ 2 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad B_1^{-1} = \begin{pmatrix} 0 & \frac{1}{2} & 0 \\ 1 & -\frac{1}{2} & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$x_1^* = B_1^{-1}b = \begin{pmatrix} 3 \\ 5 \\ 5 \end{pmatrix} \qquad z_1^* = c_{B_1}^T x_1^* = \begin{pmatrix} 2 & 1 & 0 \end{pmatrix} \begin{pmatrix} 3 \\ 5 \\ 5 \end{pmatrix} = 11$$

b) Selección de la variable de entrada
Como antes, el cálculo de los vectores $y_i = B_1^{-1}a_i$ se ha embebido directamente en el cálculo de los costes reducidos como se indica a continuación:

$$z_1 - c_1 = c_{B_1}^T B_1^{-1} a_1 - c_1 = \begin{pmatrix} 2 & 1 & 0 \end{pmatrix} \begin{pmatrix} 0 & \frac{1}{2} & 0 \\ 1 & -\frac{1}{2} & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ -1 \\ 2 \end{pmatrix} - 1 = -\frac{1}{2}$$

$$z_3 - c_3 = c_{B_1}^T B_1^{-1} a_3 - c_3 = \begin{pmatrix} 2 & 1 & 0 \end{pmatrix} \begin{pmatrix} 0 & \frac{1}{2} & 0 \\ 1 & -\frac{1}{2} & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 3 \\ 1 \\ 2 \end{pmatrix} + 1 = \frac{9}{2}$$

c) Selección de la variable de salida
La regla de salida establece que debe salir aquella variable con el menor cociente x_i/y_{ij} donde x_i es la variable elegida en el paso anterior para añadirse a la base y $0 \leq j < 3$ puesto que la base tiene dimensión 3:

$$\min\left\{\cancel{\frac{3}{-\frac{1}{2}}},\frac{5}{\frac{3}{2}},\frac{5}{2}\right\}$$

y sale la variable x_5, la última en la base de esta iteración

Paso 2 Mejora de la solución actual (iteración #2)

a) Cálculo de las variables básicas
En el tercer (y último) paso del SIMPLEX, la base es $B_2 = \{x_1, x_2, x_4\}$, de modo que:

$$B_2 = \begin{pmatrix} 1 & 1 & 1 \\ -1 & 2 & 0 \\ 2 & 0 & 0 \end{pmatrix} \qquad B_2^{-1} = \begin{pmatrix} 0 & 0 & \frac{1}{2} \\ 0 & \frac{1}{2} & \frac{1}{4} \\ 1 & -\frac{1}{2} & -\frac{3}{4} \end{pmatrix}$$

$$x_2^* = B_2^{-1}b = \begin{pmatrix} \frac{5}{2} \\ \frac{17}{4} \\ \frac{5}{4} \end{pmatrix} \qquad z_2^* = c_{B_2}^T x_2^* = \begin{pmatrix} 1 & 2 & 1 \end{pmatrix} \begin{pmatrix} \frac{5}{2} \\ \frac{17}{4} \\ \frac{5}{4} \end{pmatrix} = \frac{49}{4}$$

b) Selección de la variable de entrada
Como antes, el cálculo de los vectores $y_i = B_2^{-1}a_i$ se ha embebido directamente en el cálculo de los costes reducidos como se indica a continuación:

$$z_3 - c_3 = c_{B_2}^T B_1^{-1} a_3 - c_3 = \begin{pmatrix} 1 & 2 & 1 \end{pmatrix} \begin{pmatrix} 0 & 0 & \frac{1}{2} \\ 0 & \frac{1}{2} & \frac{1}{4} \\ 1 & -\frac{1}{2} & -\frac{3}{4} \end{pmatrix} \begin{pmatrix} 3 \\ 1 \\ 2 \end{pmatrix} + 1 = 5$$

$$z_5 - c_5 = c_{B_2}^T B_2^{-1} a_5 - c_5 = \begin{pmatrix} 1 & 2 & 1 \end{pmatrix} \begin{pmatrix} 0 & 0 & \frac{1}{2} \\ 0 & \frac{1}{2} & \frac{1}{4} \\ 1 & -\frac{1}{2} & -\frac{3}{4} \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} - 0 = \frac{1}{4}$$

Como quiera que no hay costes reducidos negativos, el SIMPLEX concluye en este paso con:

$$x^* = \begin{pmatrix} \frac{5}{2} \\ \frac{17}{4} \\ 0 \\ \frac{5}{4} \\ 0 \end{pmatrix}$$

con un valor óptimo de la función objetivo $z^* = \frac{49}{4}$

4. La única variable artificial que se había considerado en el problema anterior, x_6 toma el valor 0 en el punto óptimo y, por lo tanto, el problema es factible y puede concluirse que la solución anterior es óptima.

   Además, no existen variables de holgura en el problema, lo que significa que los recursos se utilizan exactamente —como, de hecho, demanda el uso de las igualdades.

5. Para ello, basta con resolver el problema dual de éste y observar el valor óptimo de las variables duales.

Solución al problema 3

Una versión más elaborada de este problema puede encontrarse en: José Cuena. *Notas sobre Modelos de Razonamiento*. Facultad de Informática, Universidad Politécnica de Madrid. 1996, pp. 55–59.

1. Un problema de satisfacción de restricciones se define como una terna (X, D, C) donde $X = \{x_i\}_{i=1}^n$ es el conjunto de variables; $D = \{D_i\}_{i=1}^n$ representa los dominios de cada variable respectivamente y $C = \{C_i\}_{i=1}^m$ es el conjunto de restricciones del problema.

   En este caso, las variables de decisión son el día en el que se inicie cada actividad, en particular:

   x_A $\rightleftharpoons$ Día en el que se inicia *Excavación y cimentación*
   x_B $\rightleftharpoons$ Día en el que se inicia *Cerramientos y fachadas*
   x_C $\rightleftharpoons$ Día en el que se inicia *Tabiques interiores*
   x_D $\rightleftharpoons$ Día en el que se inicia *Carpintería en puertas y ventanas*
   x_E $\rightleftharpoons$ Día en el que se inicia *Pintura*

   Como se sabe que ninguna actividad puede iniciarse más allá del vigésimo día desde el inicio de la construcción de la vivienda, $D_i \in [1, 20]$ para cualquier actividad

   Por último, las restricciones entre dos variables se representan normalmente como pares que representan los valores legales para cada variable. Estos valores pueden representarse, o bien explícitamente (como tuplas) o con relaciones que determinen automáticamente esos valores. Como la segunda alternativa es normalmente más eficiente que la primera debe elegirse siempre que sea posible. De hecho, las restricciones del enunciado pueden representarse de esta manera como se indica a continuación:

   R1 *Cerramientos y fachadas* (x_B) no pueden iniciarse antes de concluir la *Excavación y cimentación* (x_A)
   Si la tarea A comienza en x_A, entonces concluirá en $x_A + 8$ puesto que tiene una duración de 8 días y, por lo tanto:

$$x_B \geq x_A + 8$$

R2 La *Carpintería en puertas y ventanas* (x_D) debe empezarse como muy tarde antes de acabar *Tabiques interiores* (x_C)
Razonando como antes, si la tarea C comienza en x_C, y se sabe que dura exactamente 5 días, entonces concluirá en $x_C + 5$ de donde se sigue que:

$$x_D < x_C + 5$$

R3 La *Pintura* (x_E) debe iniciarse como muy tarde dos días antes de acabar *Cerramientos y fachadas* (x_B)
Esta restricción se representa también análogamente a las anteriores: si la tarea B comienza el día x_B, y se sabe que dura 7 días, entonces concluirá el día $x_B + 7$ desde el inicio de la construcción de la vivienda. Por lo tanto:

$$x_E \leq x_B + 7 - 2 = x_B + 5$$

Es una práctica habitual representar las restricciones con los mismos subíndices que las variables que relacionan. Por lo tanto, el conjunto final de restricciones de este problema es:

$$\begin{array}{lcl} R_{AB} & : & x_A + 8 \leq x_B \\ R_{CD} & : & x_C + 5 > x_D \\ R_{BE} & : & x_B + 5 \geq x_E \end{array}$$

2. La arco consistencia entre dos variables x_i y x_j sirve para determinar si para cada valor $a_i \in D_i$ existe otro $a_j \in D_j$ que satisfaga la restricción que relaciona las variables i y j, R_{ij}. Si no fuera así, entonces a_i puede eliminarse del dominio de la primera variable, D_i. Como quiera que la arco-consistencia es direccional, su aplicación se debe hacer también considerando la afectación al dominio de la primera variable, D_i, a partir de las asignación de valores legales a la segunda variable, x_j.

   Las variables involucradas en la primera restricción son x_A y x_B cuyos dominios, originalmente son, como se indicó en el apartado anterior, $[1, 20]$. De la observación de $R_{AB} : x_A + 8 \leq x_B$ resulta obvio que x_A no puede tomar valores mayores que 12 (o, de lo contrario, x_B tomaría valores mayores que el máximo de su dominio) y, de la misma manera, x_B no puede tomar valores menores que 9 o, de lo contrario, x_A tomaría valores menores que el mínimo de su dominio.

   Por lo tanto, x_A y x_B son arco-consistentes y de la aplicación de la arco-consistencia entre éllas resultan los dominios $D_A = [1, 12]$ y $D_B = [9, 20]$

3. La camino consistencia (de longitud 2) entre dos variables x_i y x_j respecto de una tercera x_k, consiste en determinar si para cada asignación de valores $a_i \in D_i$ y $a_j \in D_j$ a las variables x_i y x_j respectivamente compatible con la restricción que las une, R_{ij}, existe un valor $a_k \in D_k$ que sea consistente con las relaciones R_{ik} y R_{jk}. Si no fuera así, la asignación (a_i, a_j) puede eliminarse de la relación R_{ij}

   Como las restricciones R1 y R3 del enunciado dan lugar a R_{AB} y R_{BE}, la camino consistencia sólo puede hacerse entre las variables x_A y x_E respecto de x_B. Se trata, por lo tanto, de camino-consistencia de longitud 2. Nótese que no existe una restricción entre x_A y x_E definida explícitamente o, en otras palabras, es posible cualquier asignación simultánea de valores a esas dos variables.

   Puede comprobarse fácilmente que para cualquier par de valores asignados simultaneamente a x_A y x_E siempre existe un valor de x_B que satisface todas las restricciones. Por lo tanto, x_A y x_E son camino consistentes con respecto a x_B y el dominio de la restricción R_{AE} no se afecta en absoluto.

Solución al problema 4

1. La representación del espacio de estados de este problema es muy sencilla. Básicamente, consiste en la manipulación de secuencias o permutaciones (que representan *genomas*) que resultan de la **agregación** de posiciones (o *genes*) con un valor determinado. Por lo tanto, son marcos de la representación del espacio de estados los siguientes:

Secuencia	
`es-un:`	
Atributo	**Posibles valores/Valor**
n	entero
k	entero
posición	#POSICIÓN

Posición	
`es-un:`	
Atributo	**Posibles valores/Valor**
casilla	entero
valor	símbolo

En el estado inicial habrá una única instancia de `Secuencia` que, además, contendrá información necesaria para la generación de los operadores en el apartado siguiente, como son la longitud de la secuencia (`n`) y el número de posiciones que se invierten en cada paso —atributo `k`. Además, habrá hasta `n` instancias de `Posición`, cada una de las cuales almacena el `valor` de cada `posición` de la `secuencia`.

Existen claramente dos tipos diferentes de operadores: de una parte, los de *desplazamiento* y, de la otra, los de *inversión*. Para la formalización de la representación de los operadores sólo es preciso ahora resolver algunos detalles adicionales. Por ejemplo, pueden usarse operadores diferentes para desplazar a derechas e izquierdas; pero también podría hacerse con un único operador que dispone de un argumento en función del cual se hace el desplazamiento en un sentido u otro. De la misma manera, podría usarse el mismo argumento para decidir el número de posiciones que se desplazan.

La siguiente formalización representa, en un único operador, los desplazamientos de cualquier tipo (distinguidos por un valor positivo o negativo de `desplazamiento`) con un coste siempre igual a la unidad:

`Desplazar (desplazamiento)`:

SI $\exists s \in$ `Secuencia` $\wedge$
$\exists p_1 \in$ `Posición` $\wedge$
$\exists p_2 \in$ `Posición` $\wedge$
p_2.`casilla` $= (p_1$.`casilla` $+$ desplazamiento$)$ mod s.`n`
ENTONCES p_2.`valor` $= p_1$.`valor`
k = 1

Ahora bien, el mismo operador puede utilizar el valor de desplazamiento (positivo o negativo) para permitir desplazamientos de cualquier longitud. En este caso, sólo es preciso alterar el coste del nuevo operador, más general, que será igual al valor absoluto del desplazamiento solicitado:

`Desplazar (desplazamiento)`:

SI $\exists s \in$ `Secuencia` $\wedge$
$\exists p_1 \in$ `Posición` $\wedge$
$\exists p_2 \in$ `Posición` $\wedge$
p_2.`casilla` $= (p_1$.`casilla` $+$ desplazamiento$)$ mod s.`n`
ENTONCES p_2.`valor` $= p_1$.`valor`
k = abs(desplazamiento)

Por otra parte, el operador de inversión, con un coste igual a la unidad, se formula simplemente como sigue:

`Invertir`:

SI $\exists s \in$ `Secuencia` $\wedge$
$\exists p_1 \in$ `Posición` $\wedge$
$\exists p_2 \in$ `Posición` $\wedge$
p_2.`casilla` $= 1 + s$.`k` $- p_1$.`casilla` $+$ desplazamiento
ENTONCES p_2.`valor` $= p_1$.`valor`
k = 1

donde se ha supuesto, implícitamente, que la primera posición tiene el índice 1, en vez de 0. Asimismo, se asume que $k < n$ (lo que es más que razonable), de modo que no es preciso hacer operaciones módulo (*mod*) como ocurría en el caso del operador de desplazamiento.

2. Para el caso de `k` igual a 2, es fácil observar que cualquier permutación de los `n` símbolos en el estado final es alcanzable desde cualquier permutación en el estado inicial, de modo que el tamaño del espacio de estados será simplemente, el número de permutaciones que pueden hacerse con `n` símbolos: $n!$.

   ¡Nótese que esto no es cierto para otros valores de `k`! Por ejemplo, con $k = 3$ es claramente imposible generar algunas permutaciones: mientras que los desplazamientos a derechas e izquierdas no alteran la posición relativa de unos contenidos respecto de otros (de modo que después de desplazar $\mathcal{ABCDEFGH}$ a la derecha, la $\mathcal{B}$ seguirá junto a la $\mathcal{A}$ ¡o la $\mathcal{H}$ junto a la $\mathcal{A}$!), las inversiones sí las pueden alterar, de modo que a partir de $\mathcal{ABCDEFGH}$ se pasaría a $\mathcal{CBADEFGH}$. Sin embargo, para $k = 3$, cuando una posición está inmediatamente a continuación de otra, una inversión la pone a dos posiciones de ella (originalmente, la $\mathcal{A}$ está junto a la $\mathcal{H}$ y, después de la inversión, está a dos posiciones de ella). Por lo tanto, si el estado inicial fuera $\mathcal{ACBDEFGH}$, resulta claro que nunca puede alcanzarse $\mathcal{ABCDEFGH}$.

3. Como de costumbre, se sugiere aplicar la técnica de *relajación de restricciones* para obtener una función heurística *admisible* (esto es, que no sobreestime el esfuerzo para llegar hasta el estado final) e *informada* —que no sea igual a una constante.

   En este caso, la restricción más aparente es que después de cualquier inversión con $k = 2$ (esto es, invirtiendo únicamente las dos primeras posiciones de la cadena), no sólo puede moverse una casilla hacia su posición deseada sino que, además, se altera la posición de la otra. Lo mismo es cierto si en vez de aplicar el operador de inversión se aplica el de desplazamiento.

   Relajando esta restricción, resulta muy fácil observar que el número mínimo de movimientos para llevar cualquier contenido hasta su posición deseada es exactamente igual al número de posiciones que la separan —tanto si para moverla se usan desplazamientos como si se usan inversiones. Esto es, su distancia de Manhattan.

   Ahora bien, como en el problema real ocurre que los operadores no sólo modifican la posición de una casilla, sino también otras, la estimación admisible no puede resultar de sumar la distancia de Manhattan de cada casilla. En su lugar, sólo podemos calcular el máximo.

   La función heurística sugerida es, por lo tanto:

$$h(n) = \max_{\forall i} \{\mathrm{Manhattan(p_i)}\}$$

4. ¡Por supuesto que es preciso! Si no se hiciera, habría operadores con coste nulo. En concreto, los de desplazamiento. Por otra parte, los algoritmos de *el mejor primero* pueden caer en bucles infinitos (expandiendo una y otra vez nodos en un mismo camino) si existen operadores de este tipo. Por este motivo, se exige que los operadores tengan coste siempre estrictamente positivos.

   Ahora bien, si se eliminan los operadores de desplazamiento, entonces la formulación del apartado 1 sólo permitiría que se pudieran invertir las primeras posiciones, sin habilitar medios para invertir otras posiciones de la misma cadena. Por ello, es preciso reformular el operador de inversión que ahora debe poder visitar cualquier posición `p` de la cadena:

   `Invertir (p)`:

   SI $\exists s \in$ `Secuencia` $\wedge$
   $\exists p_1 \in$ `Posición` $\wedge$
   $\exists p_2 \in$ `Posición` $\wedge$
   p_2.`casilla` $= p + s.$`k` $- p_1$.`casilla` $+$ desplazamiento
   ENTONCES p_2.`valor` $= p_1$.`valor`
   k = 1

   Nótese que el único cambio ha sido sustituir el valor 1 por `p` en la última comprobación del antecedente de esta regla.

5. Puesto ahora es posible alcanzar los mismos estados que antes con un coste inferior, efectivamente podría darse el caso de que nuestra función heurística fuera no admisible. Sin embargo, la función heurística de la sección 3, no consideraba desplazamientos, por lo que aquella sigue siendo admisible.