# Examen Parcial de IA

**(20 de abril de 2009)**
**grupo 20**
**Duración: 1 hora**

1. (6 puntos) Una empresa chocolatera tiene planteado el siguiente problema, desea comercializar una nueva caja de bombones que cumpla cierto conjunto de características: el precio ha de ser el mínimo posible, el beneficio que se obtenga ha de ser el máximo posible, ha de tener un peso limitado (no mas de 500 g, ni menos de 400 g) y ha de estar compuesta de una combinación de los diferentes tipos de bombones que fabrica, estos pertenecen a $K$ tipos diferentes, cada uno de ellos con un precio, un peso y un beneficio por bombón diferentes. La caja ha de contener no más de $N$ y no menos de $M$ bombones de cada tipo.

   Una posible solución a éste problema se puede obtener mediante el uso de algoritmos de búsqueda local. En los siguientes apartados se proponen diferentes alternativas para algunos de los elementos necesarios para plantear la búsqueda (solución inicial, operadores, función heurística,...). Comenta muy brevemente la solución que se propone respecto a si es correcta y si es mejor/peor respecto a otras alternativas posibles. Justifica tus respuestas.

   *a*) Se plantea solucionarlo mediante Hill-climbing utilizando como solución inicial una caja vacía y como operadores añadir y quitar bombones en la caja sin pasar los límites máximo y mínimo de número de bombones. Como función de evaluación de las soluciones la suma de precios de los bombones menos las suma del beneficio de los bombones.

   *b*) Se plantea solucionarlo mediante Hill-climbing utilizando como solución inicial llenar la caja de bombones de un solo tipo hasta llegar al peso mínimo y como operadores añadir un bombón sin pasar del peso máximo y cambiar un bombón por otro de un tipo distinto. La función de evaluación es la suma del precio de los bombones multiplicada por la suma de los beneficios.

   *c*) Se plantea solucionarlo mediante Hill-climbing utilizando como solución inicial asignar un número aleatorio de bombones de cada tipo dentro del rango [M,N]. Como operadores tenemos cambiar un bombón de un tipo por otro de otro tipo siempre que se mantengan las cantidades máximas y mínimas por tipo y no se violen las restricciones de peso. Se plantea utilizar como función de evaluación de las soluciones la suma del beneficio de los bombones de cada tipo que pasa de $M$ unidades dividida por la suma de precios de los bombones de cada tipo que pasa de $M$ unidades.

   *d*) Se plantea utilizar algoritmos genéticos donde la representación de la solución es una tira de bits donde hay $K \cdot N$ bits donde el bit determina si el bombón está o no en la caja. Las soluciones iniciales se obtienen llenando la caja con $M$ bombones de cada tipo. Se usan los operadores habituales de cruce y mutación. La función de evaluación es la suma de los precios penalizando la solución con el valor $+\infty$ cuando el peso está por encima o por debajo de los límites de peso o se violan los límites de número de bombones por tipo.

2. (4 puntos) En la pasarela Gaudí han de organizar diez desfiles para diez diseñadores, dos por día, sabiendo que algunos diseñadores se niegan a desfilar el mismo día que algún otro. Se han de asignar las y los modelos a los desfiles ($m$ por desfile) sin que la misma persona desfile más de tres veces en total. Cada diseñador tiene también unas restricciones de altura y peso para sus modelos. Los organizadores de la pasarela han de pagar a los modelos y cada uno tiene su tarifa por participar en un desfile. Obviamente, los organizadores quieren pagar lo menos posible. La organización se plantea dos estrategias distintas para resolver el problema de la pasarela:

   *a*) Resolvemos primero el problema de asignar los diseñadores a los desfiles mediante un algoritmo de satisfacción de restricciones. A partir de esa asignación usamos el algoritmo de A* para solucionar la asignación de los modelos a los desfiles. El estado es una asignación parcial de modelos a desfiles. Utilizamos como operador `añadir_modelo` a un desfile concreto que comprueba que no haya más de $m$ modelos en el desfile y que se cumplan las restricciones del diseñador e `intercambiar_modelo` entre dos desfiles que tambien comprueba las restricciones del diseñador, el coste del operador asignar es la tarifa del modelo asignado, la del operador intercambiar es la del modelo mas caro de los dos. Como función heurística usamos el número de modelos que falta incluir en los desfiles multiplicado por la tarifa del modelo mas caro.

*b*) Usar un algoritmo de satisfacción de restricciones. Primero resolver el problema de asignar modelos a cada uno de los desfiles, teniendo en cuenta las restricciones de cada diseñador, y luego, una vez determinada la composición de cada desfile, montar el calendario (qué dos desfiles tocan cada día) teniendo en cuenta que no se emparejen dos diseñadores que no se pueden ver y que no haya modelos que desfilen más de tres veces.

Comenta cada una de las posibilidades indicando si resuelven o no el problema, qué errores te parece que tiene cada solución y cómo se podrían corregir, y qué ventajas e inconvenientes tienen cada una de ellas. Justifica la respuesta.

# Examen Parcial de IA

**(22 de abril de 2009)**
**grupo 10**
**Duración: 1 hora**

1. (6 puntos) Se ha de celebrar un congreso multitudinario en la ciudad y los organizadores del evento han de acomodar a los $A$ asistentes. Disponen de una lista de hoteles y de cada uno de ellos saben:

   - A qué distancia se encuentra del lugar de la celebración del congreso,
   - Cuantas habitaciones tiene disponibles y
   - El precio de la habitación.

   Su problema es encontrar una asignación de las $A$ personas a hoteles de modo que no se supere el número de habitaciones disponibles en el hotel, se minimice la suma de las distancias recorridas por todos los asistentes y se minimice la cantidad de dinero que la organización va a dedicar al pago de las habitaciones.

   Una posible solución a éste problema se puede obtener mediante el uso de algoritmos de búsqueda local. En los siguientes apartados se proponen diferentes alternativas para algunos de los elementos necesarios para plantear la búsqueda (solución inicial, operadores, función heurística,...). Comenta muy brevemente la solución que se propone respecto a si es correcta y si es mejor/peor respecto a otras alternativas posibles. Justifica tus respuestas.

   *a*) Usar Hill Climbing. Como solución inicial se utiliza una estrategia avariciosa poniendo a las personas en los hoteles más baratos hasta colocarlas a todas. Como operador se utiliza mover una persona de un hotel a otro. La función heurística es el producto entre la suma de las distancias recorridas por todas las personas y la suma de los precios de sus habitaciones.

   *b*) Usar Hill Climbing. Como solución inicial se distribuyen aleatoriamente todas las personas entre los hoteles. Como operador se utiliza mover un grupo de personas de un hotel a otro, moviendo tantas como quepan en el hotel destino. La función heurística es la suma de las distancias recorridas por todas las personas multiplicada por una constante $D$ más la suma de los precios de sus habitaciones multiplicada por una constante $P$.

   *c*) Usar Hill climbing. Como solución inicial se utiliza una estrategia avariciosa poniendo a las personas en los hoteles con el cociente precio/distancia más bajo. Como operador se utiliza mover un grupo de personas de un hotel a otro, moviendo tantas como quepan en el hotel destino. La función heurística es la suma para todas las personas del producto entre la distancia recorrida por una persona y el precio de su habitación.

   *d*) Usar algoritmos genéticos. Representaremos la solución como una tira de bits donde asignamos a cada hotel tantos bits como se necesiten para representar el número máximo de personas que caben. La solución inicial se calcula de la misma manera que en el apartado a). Utilizamos los operadores habituales de cruce y mutación. La función heurística es la suma de las distancias de todos los hoteles que tienen alguna persona, dividida por la suma de los precios de las habitaciones de todas las personas.

2. (4 puntos) Una fábrica de coches quiere optimizar su proceso de fabricación integrando el montaje de los $k$ diferentes modelos que fabrica en un solo proceso. Cada modelo de coche necesita utilizar un conjunto de máquinas en un orden específico que se puede describir mediante una secuencia de proceso. Para simplificar supondremos que cada paso del proceso tiene la misma duración. Hay que tener en cuenta que un coche puede necesitar un mismo tipo de maquina en diferentes pasos de su proceso de fabricación.

   Disponemos de un número fijo de máquinas de cada tipo ($M$). Supondremos que el proceso lo diseñamos para fabricar una unidad de cada modelo de coche y solo podemos volver a iniciar el proceso una vez que hemos acabado, es decir, en el proceso integrado solo fabricará una unidad de cada modelo a la vez.

Se nos plantean las siguientes alternativas:

*a*) Queremos utilizar A* para minimizar el número total de pasos del proceso integrado. El estado lo forman los pasos de los procesos individuales que hemos encajado en cada paso del proceso integrado. El operador de cambio de estado consiste en colocar el máximo número de pasos de cada proceso en el paso actual del proceso integrado, el coste del operador es el número de pasos integrados. La función heurística es la suma de pasos de los procesos individuales que nos quedan por integrar.

*b*) Queremos utilizar satisfacción de restricciones para minimizar el número total de pasos. Para ello elegimos tener como variables la asignación de un paso de fabricación de un modelo a una máquina en un paso del proceso integrado. Suponemos que el proceso integrado tendrá como máximo tantos pasos como la suma de pasos de todos los procesos individuales, por lo que necesitaremos como variables ese valor multiplicado por el número de máquinas. El dominio de las variables son los modelos de coche. Con esto estamos representando para cada paso del proceso integrado qué máquinas tienen asignado un paso de los procesos individuales. La restriccion que mantenemos es que ninguna asignación viole el orden de las secuencias individuales, es decir, que para un paso concreto no puede haber un modelo asignado a mas de una máquina y que entre pasos se cumpla el orden de las secuencias.

Comenta cada una de las posibilidades indicando si resuelven o no el problema, qué errores te parece que tiene cada solución y cómo se podrían corregir, y qué ventajas e inconvenientes tienen cada una de ellas. Justifica la respuesta.

# Examen Parcial de IA

**(22 de abril de 2009)**
**grupo 30**
**Duración: 1 hora**

1. (6 puntos) Parques y Jardines ha de decidir qué plantas va a colocar en un parque de $X$ Ha. En el almacén tienen $K$ plantas distintas y de cada planta se sabe la superficie que ocupa ($s$) su coste ($c$) la duración esperada de vida ($v$) y su temporada de auge ($t$). Los valores para la temporada de auge son: TT, PV, OI. Se trata de escoger un subconjunto $C$ (con $C \lll K$) de plantas diferentes que quepa en el parque de modo que se maximice la superficie que ocupan, se minimice el coste, la vida esperada del conjunto sea mayor de $T$ años y se garantice la siguiente composición:

   - de las de temporada=PV ha de haber un 30 % de las $X$ Ha como mínimo,
   - de las de temporada=OI ha de haber un 20 % como mínimo y
   - de las de temporada=TT ha de haber un 10 % como mínimo.

   La duración esperada de la vida del conjunto ($vec$) se calcula sumando la vida esperada de cada planta y dividiendo el resultado entre el número de plantas involucradas $j$, con $j = |C|$, es decir:

   $$vec = \frac{\sum_{i=1}^{j} v_i}{j}$$

   Una posible solución a este problema se puede obtener mediante el uso de algoritmos de búsqueda local. En los siguientes apartados se proponen diferentes alternativas para algunos de los elementos necesarios para plantear la búsqueda (solución inicial, operadores, ...). Recuerda que el objetivo es minimizar el valor de la función heurística. Comenta muy brevemente la solución propuesta respecto a si es correcta y si es mejor/peor respecto a otras alternativas posibles. Justifica tus respuestas.

   *a*) Se propone usar Hill-Climbing. La solución de partida contiene la planta que cubre la mayor superficie de todas. El operador disponible es `poner_planta` y la función heurística es

   $$h'(n) = \sum_{i=1}^{j} s_i - \sum_{i=1}^{j} c_i$$

   *b*) Se propone usar Hill-Climbing. La solución inicial se obtiene escogiendo plantas secuencialmente hasta que no quepan más. Los operadores de cambio de estado son: `quitar_planta`, siempre y cuando la planta a quitar haga que se cubra el porcentaje mínimo para la temporada de esa planta, y `añadir_planta`, siempre y cuando la superficie total ocupada no supere $X$ Ha. La función heurística es

   $$h'(n) = (\sum_{i=1}^{j} \frac{c_i}{s_i}) - (T - vec)$$

   *c*) Se propone usar Hill-Climbing. La solución inicial se construye de la siguiente manera: se forman 3 grupos de plantas, uno por temporada de auge, y se ordena cada grupo en orden creciente de precio. Sobre cada grupo ordenado, se van tomando plantas hasta que se cubre el porcentaje mínimo de superficie para cada una de las temporadas. Finalmente, se agrupan las plantas no colocadas de los 3 grupos y se van poniendo en la solución las más baratas hasta que no quepan más. El operador de transformación es `sustituir_una_por_otra` que quita una planta de la solución y coloca en su lugar otra que no estaba. Para aplicarlo se ha de cumplir que la solución resultante siempre cubra los mínimos de temporada y la superficie total que ocupe no supere $X$ Ha.

La función heurística es:

$$h'(n) = (\sum_{i=1}^{j} \frac{s_i}{c_i}) + (T - vec)$$

*d*) Se propone usar un algoritmo genético. Un individuo es una tira de $K$ bits, y cada bit identifica a una planta. Cada uno de los individuos de la población de partida es una codificación binaria de las diferentes soluciones iniciales que se pueden obtener aplicando el algoritmo propuesto en el apartado c). Como operadores utilizamos los habituales de cruce y mutación. La función de fitness vale $\infty$ si la superficie ocupada es mayor que $X$ y en el resto de casos es

$$h'(n) = (T - vec) \cdot (X - \sum_{i=1}^{j} \frac{c_i}{s_i})$$

2. (4 puntos) Queremos generar los horarios para una compañía aérea desde distintas ciudades europeas ($c$ ciudades) a Nueva York. Se conoce el número estimado de pasajeros por día de la semana y ciudad. Cada ciudad ha de tener un número mínimo de vuelos por semana ($vs$), preestablecido por la compañía. Además se fija también un número máximo $M$ total vuelos que la compañía puede hacer llegar al día a Nueva York. En cada avión caben $p$ pasajeros. Con los $M$ vuelos diarios deberíamos tener suficiente para poder cubrir los $vs$ vuelos semanales desde cada ciudad. Se plantean las siguientes alternativas para maximizar el número de usuarios transportados por semana:

   *a*) Queremos usar satisfacción de restricciones, donde hay una variable por cada combinación de ciudad y días de la semana, los valores son el número de vuelos de esa combinación. Las restricciones son: La suma del número de vuelos para una ciudad en una semana no puede ser menor que $vs$, la suma del número de vuelos para un día no ha de ser superior $M$ y el número total de plazas que se quedan vacías tiene que ser menor que un cierto valor $u \times c$.

   *b*) Queremos usar A$^*$ para maximizar el número de personas que transportamos. El estado es una asignación parcial de aviones a ciudades y dias de la semana. Tenemos un operador que consiste en incrementar los vuelos de un día y ciudad concretos si quedan pasajeros por transportar comprobando no salgan mas de $M$ vuelos diarios, el coste del operador es uno o infinito si superamos los vuelos por semana de la ciudad. Como función heurística usamos la parte entera entre el cociente entre $p$ y el número de pasajeros que quedan por transportar.

   Comenta cada una de las posibilidades indicando si resuelven o no el problema, qué errores te parece que tiene cada solución y cómo se podrían corregir, y qué ventajas e inconvenientes tienen cada una de ellas. Justifica la respuesta.