TECNICAS INTELIGENCIA ARTIFICIAL

TRABAJOS PROPUESTOS (2020-21)

1) **Propuestas de Aplicaciones: Alg. Genéticos, Enfriamiento Simulado. Diseño, Desarrollo y Evaluación. Comparativa.**

2) **Propuestas de Aplicaciones: Metaheurísticas Opcionales.**

Metaheurísticos de Mejora (Búsqueda Local). ***Requiere solución previa.***

- **Escalada Estocástica**
- **Búsqueda Tabú,**
- **Enfriamiento Simulado,**
- **Búsqueda en Haz**

Inteligencia de Enjambre

- **Alg. de las Hormigas**
- **Colonia de Abejas**
- **Cúmulo de Partículas**

Metaheurísticos Híbridos: Constructivos + Mejora:

- **GRASP: Básico, Semi-greedy, reactivo, clustering.**

Metaheurísticos Evolutivos. ***(requiere soluciones previas)***

- **Algoritmos Genéticos**
- **Búsqueda dispersa,**
- **Alg. Meméticos**

3) **Propuestas Abiertas (TIA-G)**

a) **Desarrollo de SBC (Sistemas Expertos). Diseño de Ontologías. Protégé / Jess**
b) **Razonamiento Incierto. Aplicación Redes Bayesianas.**
c) **Razonamiento Impreciso / Lógica Difusa. Aplicación en Fuzzy-Clips.**
d) **Técnicas de Planificación. Diseño PDDL. Aplicación Planificadores.**
e) **Sistemas de Decisión.**
f) **Teoría de Juegos. Juegos complejos (azar), Juegos Simples (Otello, Oware, damas, etc.): Minimax / Alfa-Beta**
g) **Satisfacción de Restricciones (MiniZinc, Choco, Gams, etc.)**

Propuestas Metaheurísticos

1. **Problemas de recubrimiento. Bloques rectangulares** ***(4 variantes)***
2. **Asignación optimizada de turnos 24h**
3. **Placas aislantes**
4. **Secuencias de números**
5. **Problema del Viajante de Comercio. Variantes.**
6. **Problema del Cartero Chino**
7. **Planificación Atraque Barcos**
8. **Generación Horarios Ferroviarios**
9. **Asignación Lectores y Periódicos**
10. **Asignación rutas y frecuencias en comunicaciones** ***(complejo)***
11. **Rutas de Vehículos** ***(complejo)***
12. **Optimización del corte de paneles.**
13. **Problema propuesto por el alumno**

- **Cubo de Rubik**
- **K-coloreado de mapas**
- **Problema del 8-puzzle**
- **Conecta-4**
- **Problemas de Laberintos**
- **El problema del Caballo de Ajedrez (sencillo)**
- **Problema de Cartas** ***(sencillo)***

Notas:

- **Los trabajos se realizan de forma individual**
- **La asignación se realizará previa petición del alumno**
- **Se puede elegir la utilización de un entorno generalista, librerías y herramientas de apoyo o implementación desde cero utilizando el lenguaje de programación deseado**
- **En la bibliografía recomendada e Internet se pueden encontrar ampliaciones sobre las propuestas y datos para casos de prueba**

Para la evaluación de los trabajos se tendrá principalmente en cuenta:

1. ***Elección de la metaheurística. Diseño del algoritmo. Adecuada parametrización.***
2. ***Resultados obtenidos (métricas). Fundamentalmente, calidad de la respuesta vs. tiempo de cómputo***
3. ***Diversidad de tallas/casos de prueba (benchmarks). Variación de parámetros.***
4. ***Conclusiones obtenidas***
5. ***Memoria entregada. Presentación de los resultados***

Esquema de memoria (y presentación: < 10 minutos)

a) ***Descripción del problema (1 transparencia) y del método aplicado (2 transparencias)***

b) ***Diseño del algoritmo para la resolución del problema en sus dos versiones. Parámetros (3 transparencias)***

c) ***Implementación de la solución o utilización de entorno generalista (2 transparencias)***

d) ***Evaluación. Casos de prueba y métricas comparativas. Discusión (3 transparencias)***

e) ***Conclusiones (1 transparencia)***

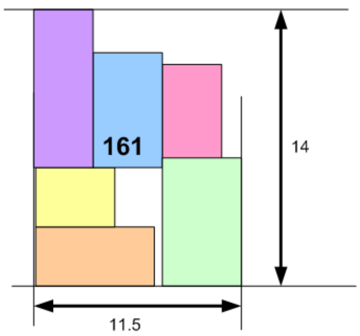
ENTREGA (poliformat):

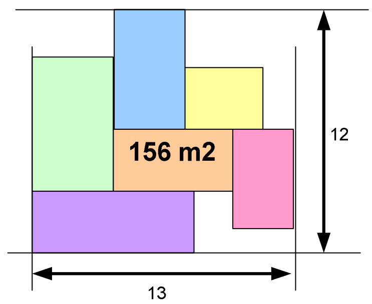
- **Memoria, presentación, código ejecutable y/o fuente.**
- **Subir a PoliformaT un fichero comprimido zip/rar cuyo nombre sea el del alumno.**
- **Habrá una presentación del trabajo en la última sesión de prácticas de la asignatura**

Problemas de recubrimiento (2-D)

Dado un conjunto de n bloques rectangulares, de distintas alturas y anchuras, se debe obtener la posición de todos los bloques en un espacio cartesiano bidimensional, tal que no haya solapamientos entre los bloques y se minimice la superficie (anchura x altura) del espacio contenedor.

Dos ejemplos:





Casos alternativos:

Caso a) Los bloques no pueden girarse (tienen una orientación determinada).

Caso b) Los bloques pueden girar 90º, a izquierda o a derecha.

Caso-c) La superficie contenedora está limitada, tal que no caben todos los bloques y se quiere maximizar el número de bloques colocados.

Caso-d) La anchura es fija (90) y se desea minimizar la altura de la superficie requerida para colocar todos los bloques (bobina continua).

Como instancias de ejemplo, asumid los casos siguientes (óptimo a alcanzar, caso b, 20 x 20):

Alt	**2**	**7**	**8**	**3**	**3**	**5**	**3**	**3**	**5**	**2**	**3**	**4**	**3**	**4**	**9**	**11**
Anc	**12**	**12**	**6**	**6**	**5**	**5**	**12**	**7**	**7**	**6**	**2**	**2**	**4**	**4**	**2**	**2**

Alt	**4**	**5**	**2**	**9**	**5**	**2**	**7**	**3**	**6**	**3**	**6**	**4**	**6**	**10**	**6**	**10**
Anc	**14**	**2**	**2**	**7**	**5**	**5**	**7**	**5**	**5**	**2**	**2**	**6**	**3**	**3**	**3**	**3**

Probad el método desarrollado con otros bloques ejemplo.

Métrica de evaluación: Superficie ocupada vs tiempo de cómputo.

Notas:

- Se sugiere determinar la posición de cada bloque mediante la asignación de su centro geométrico al espacio bidimensional, o bien, mediante relación de orden con bloques contiguos.
- Debe notarse que la colocación de cada bloque tiene un grado de libertad bastante limitado (ya que no tiene sentido colocar un bloque seleccionado de forma no adyacente a alguno de los ya colocados).

La asignación de bloques es un problema clásico de optimización combinatoria NP-hard. El problema concreto suele llamarse **"Strip packing"**, intentando minimizar la superficie (o volumen, en caso 3-D).

El problema tiene relación con el ***"cutting stock problem"*** que, contrariamente, intenta minimizar la merma (o superficie/volumen) que se desperdicia. Son 'complementarios' y se sitúan en lo que podría llamarse ***"Cutting and Packing Problems"***

Extensiones reales: corte unidimensional, bidimensional. Corte no recto (ropa, piel, etc.)

Asignación Optimizada de Turnos en empresa 24 horas

El problema es asignar los turnos de Mañana/Tarde/Noche al personal de una empresa, para cada día de trabajo:

		JUNIO																																JUNIO			
Cod.	Nombre	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	1	2	M	T	N	Tot
		L	M	X	J	V	S	D	L	M	X	J	V	S	D	L	M	X	J	V	S	D	L	M	X	J	V	S	D	L	M	X	J				
1		N	N	-	-	T	T	T	T	T	-	-	M	M	M	M	M	-	-	-	-	-	N	N	N	N	-	-	-	-	-	T	T	5	5	6	16
2		-	M	M	M	M	-	-	-	-	-	-	N	N	N	N	-	-	-	-	T	T	T	T	-	-	N	N	N	N	N	V	V	4	4	9	17
3		M	M	M	M	M	-	-	M	M	M	M	M	-	-	-	-	-	-	M	M	M	M	M	-	-	M	M	M	M	M	V	V	20	0	0	20
4		M	M	M	M	M	-	-	-	-	N	N	-	-	-	T	T	T	T	T	-	-	-	-	N	N	N	N	N	-	-	-	M	5	5	7	17
5		T	T	T	T	T	-	-	-	T	T	T	T	-	-	-	-	-	-	M	M	M	M	M	-	-	-	M	M	M	M	M	-	9	9	0	18
6		N	N	-	-	-	-	-	T	T	T	T	T	-	-	N	N	N	N	-	-	-	-	-	M	M	M	M	M	-	-	T	T	5	5	6	16
7		-	-	-	-	N	N	N	N	-	-	-	M	M	M	M	M	-	-	T	T	T	T	T	-	-	-	-	-	T	T	V	V	5	7	4	16
8		-	-	-	-	N	N	N	N	N	-	-	T	T	T	T	T	-	-	N	N	N	N	N	-	-	-	-	-	T	T	T	T	0	7	10	17
9		-	-	-	-	T	T	T	T	T	-	-	M	M	M	M	M	-	-	N	N	N	N	N	-	-	-	-	-	M	M	M	M	7	5	5	17

Con las siguientes **RESTRICCIONES**:

- Existen 50 trabajadores y un horizonte de asignación de 4 semanas.
- Se asignarán secuencias de turnos de M/T/N, con un mínimo de DOS turnos de descanso entre cada secuencia de turnos. Como excepciones:
    a) No pueden haber dos agrupaciones de turnos N seguidas.
    b) El cambio de turnos de N a M solo se podrá hacer tras 3 días de descanso.
    c) Un cambio de agrupación de M a N se podrá hacer tras tan solo un día de descanso.
- Los máximos turnos de trabajo que puede hacer cada trabajador en las 4 semanas es de 20 turnos.
- Las agrupaciones mínimas en una secuencia de turnos son de DOS turnos. Las máximas de 5 turnos seguidos, antes de un periodo de descanso.
- Ampliaciones:
    a) En las cuatro semanas, todos los trabajadores deben tener al menos un fin de semana libre (no tendrán asignación el sábado, ni el domingo, ni el viernes por la noche).
    b) No se podrá iniciar una secuencia de turnos en domingo.

Intentando satisfacer al máximo los siguientes **CRITERIOS DE OPTIMALIDAD**:

1. Se intentará cubrir una demanda concreta de aproximadamente 33 turnos (M, T o N) cada día laborable. El porcentaje deseado de asignación diaria debe tender a: 37%, 37% y 26% de turnos en M/T/N, respectivamente.
2. Ello implica cubrir 12 turnos de mañana, 12 de tarde y 9 de noche, aproximadamente. En domingo, se puede asumir una reducción a 9 turnos de cada tipo.
3. Nótese que el objetivo es asignar 900 turnos de un total de 50x20 = 1.000 turnos disponibles.
4. Se intentará equilibrar la asignación de turnos M/T entre todos los trabajadores y, especialmente, la de turnos Nocturnos.

NOTAS:

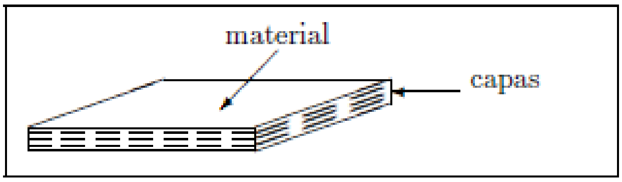
- Se recomienda utilizar un método GRASP o Algoritmos Genéticos.
- ***Métrica de evaluación: (i) Turnos asignados vs tiempo de cómputo, (ii) Equilibrios***

Competición (International Nurse Rostering Competition): http://mobiz.vives.be/inrc2/

Capas de Aislante

Resolver el problema de diseño de un material formado por un número de capas aislantes (resultante de la imprimación realizada sobre dicha capa). Este problema-tipo es de aplicación en diferentes escenarios de aislantes térmicos, eléctricos, acústicos, hidrófugos, ópticos, etc.

El orden según el cual se colocan estas capas determina el valor de aislamiento total del material resultante



El problema consiste en encontrar el **orden de las capas** que maximiza el valor de aislamiento total del material compuesto.

Supongamos que se consideran, inicialmente, 12 capas de material, con un determinado asilamiento entre cada par de materiales. El aislamiento total es la suma del aislamiento entre cada par de capas.

	A	B	C	D	E	F	G	H	I	J	K	L
A		10	15	25	32	25	21	21	15	22	12	54
B	41		57	24	52	2	66	55	61	15	6	7
C	21	31		21	21	44	21	22	22	61	47	61
D	66	22	15		47	21	41	15	21	22	32	34
E	21	44	61	47		32	26	61	55	34	18	12
F	22	18	22	23	41		21	22	44	55	54	54
G	15	25	34	21	26	27		34	25	41	7	22
H	61	34	12	54	21	23	15		21	21	55	55
I	22	54	54	65	3	25	61	77		47	22	22
J	34	7	22	23	54	42	22	54	21		12	15
K	26	61	55	22	18	18	22	18	34	21		12
L	22	18	25	34	21	22	18	61	55	2	22	

Caso a) Se deben utilizar todos los materiales. Las combinaciones posibles son: 12!= 479.001.600

Caso b) Se pueden repetir el material en las capas (pero no juntar dos capas del mismo material). Por lo tanto, no es preciso utilizar todos los materiales. Este caso permitiría simular capas de material de distinto espesor. Las combinaciones posibles son: $12 * 11^{11}$= 3.423.740.047.332

Caso c) Aumentad el nº de capas (rellenando los blancos con 0)

	10	15	25	32	25	21	21	15	22	12	54		10	15	25	32	25	21	21	15	22	12	54
41		57	24	52	2	66	55	61	15	6	7	41		57	24	52	2	66	55	61	15	6	7
21	31		21	21	44	21	22	22	61	47	61	21	31		21	21	44	21	22	22	61	47	61
66	22	15		47	21	41	15	21	22	32	34	66	22	15		47	21	41	15	21	22	32	34
21	44	61	47		32	26	61	55	34	18	12	21	44	61	47		32	26	61	55	34	18	12
22	18	22	23	41		21	22	44	55	54	54	22	18	22	23	41		21	22	44	55	54	54
15	25	34	21	26	27		34	25	41	7	22	15	25	34	21	26	27		34	25	41	7	22
61	34	12	54	21	23	15		21	21	55	55	61	34	12	54	21	23	15		21	21	55	55
22	54	54	65	3	25	61	77		47	22	22	22	54	54	65	3	25	61	77		47	22	22
34	7	22	23	54	42	22	54	21		12	15	34	7	22	23	54	42	22	54	21		12	15
26	61	55	22	18	18	22	18	34	21		12	26	61	55	22	18	18	22	18	34	21		12
22	18	25	34	21	22	18	61	55	2	22		22	18	25	34	21	22	18	61	55	2	22	
	10	15	25	32	25	21	21	15	22	12	54		10	15	25	32	25	21	21	15	22	12	54
41		57	24	52	2	66	55	61	15	6	7	41		57	24	52	2	66	55	61	15	6	7
21	31		21	21	44	21	22	22	61	47	61	21	31		21	21	44	21	22	22	61	47	61
66	22	15		47	21	41	15	21	22	32	34	66	22	15		47	21	41	15	21	22	32	34
21	44	61	47		32	26	61	55	34	18	12	21	44	61	47		32	26	61	55	34	18	12
22	18	22	23	41		21	22	44	55	54	54	22	18	22	23	41		21	22	44	55	54	54
15	25	34	21	26	27		34	25	41	7	22	15	25	34	21	26	27		34	25	41	7	22
61	34	12	54	21	23	15		21	21	55	55	61	34	12	54	21	23	15		21	21	55	55
22	54	54	65	3	25	61	77		47	22	22	22	54	54	65	3	25	61	77		47	22	22
34	7	22	23	54	42	22	54	21		12	15	34	7	22	23	54	42	22	54	21		12	15
26	61	55	22	18	18	22	18	34	21		12	26	61	55	22	18	18	22	18	34	21		12
22	18	25	34	21	22	18	61	55	2	22		22	18	25	34	21	22	18	61	55	2	22	

Métrica de evaluación: Orden idóneo para las capas vs tiempo de cómputo.

Secuencia de Números

El problema es obtener una combinación de operaciones aritméticas que aplicadas sobre un conjunto de 6 números enteros obtenga un número objetivo también entero.

Las reglas son:

- Los seis números de partida se eligen al azar entre ***1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 25, 50, 75 y 100*** teniendo en cuenta que la probabilidad de elección de los números del 1 al 10 ha de ser el doble que la del resto de números (P(1..10) = 1/14 y P(25..100) = 1/28).
- El número objetivo se elige al azar entre 101 y 999.
- Hay que hacer operaciones con los números de partida o con los obtenidos con ellos. Las operaciones permitidas son la suma, la resta, la multiplicación y la división. Los resultados de estas operaciones deben ser números enteros y positivos. Sólo se puede hacer una división si el resultado es entero. Se recomienda no tener prioridad en las operaciones.
- No es obligatorio usar todos los números, pero un número no se puede utilizar dos veces.

Si no es posible conseguir el número exacto, la mejor solución será aquella que más se aproxime al número objetivo. La búsqueda terminará cuando encuentre la combinación de operadores que obtienen el valor objetivo o cuando se ha superado un máximo de iteraciones/generaciones. En este caso se devolverá la mejor solución.

Casos concretos a probar:

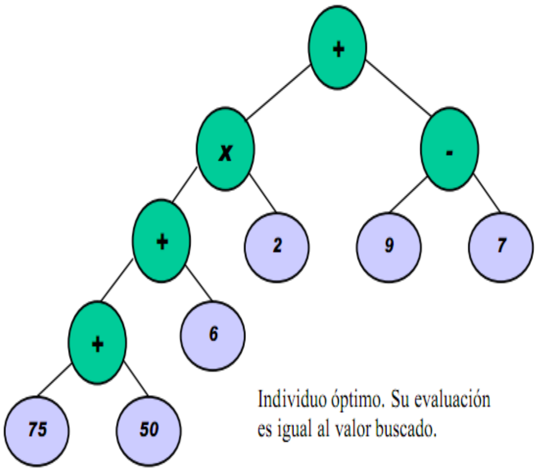
Casos Fáciles							OBJETIVO
F1	4	10	7	9	2	25	232
F2	10	2	9	5	7	100	298
F3	2	75	9	6	100	8	474
F4	3	10	7	6	75	10	381
F5	7	3	50	25	6	100	741
F6	2	4	75	4	50	2	502
F7	8	25	10	2	9	4	549
F8	7	10	3	2	1	25	223
F9	7	4	100	5	9	10	156
F10	2	10	7	50	100	10	259

Casos Difíciles							OBJETIVO
D1	***9***	***6***	***75***	***7***	***2***	***50***	***264***
D2	5	3	9	50	4	5	458
D3	1	10	3	3	75	4	322
D4	100	2	6	4	25	6	305
D5	2	1	10	10	7	100	274
D6	1	100	4	50	3	4	661
D7	10	8	75	2	100	4	431
D8	100	75	1	8	3	75	511
D9	100	3	2	25	3	9	407
D10	5	50	1	8	75	8	713

Notas:

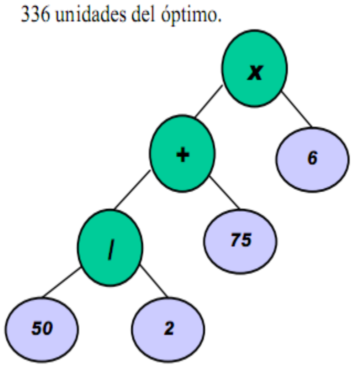
- Se recomienda usar notación polaca.
- ***Métrica: Tiempo de cómputo (vs. aproximación al objetivo)***
- ***Métodos Recomendados: A.*** Genético.

Ejemplos (para el Caso D1):

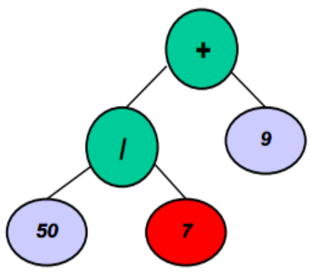


Individuo óptimo. Su evaluación es igual al valor buscado.

Individuo válido. Su evaluación es igual a 600 por lo que difiere en 336 unidades del óptimo.

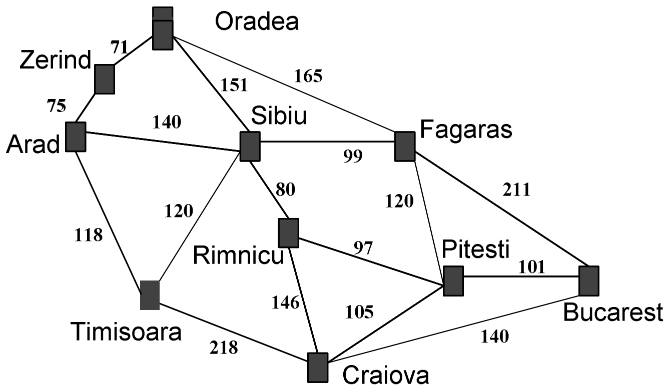


Individuo erróneo. La división no se puede utilizar si no tiene un resultado entero.



Problema del Viajante de Comercio

Resolver el para un conjunto de N ciudades. Como ejemplo de caso *muy trivial*:



Aplicarlo a casos con más nodos y más densamente conectados (80-500 ciudades). O bien, recorrer todas las capitales del mundo. Obtener otros ejemplos (que incluyen resultados para poder contrastar) de:

- http://www.tsp.gatech.edu/world/countries.html.
- http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/.
- http://www.math.uwaterloo.ca/tsp/world/countries.html
- http://elib.zib.de/pub/mp-testdata/tsp/

Particularmente, para una evaluación comparativa sobre mismos conjuntos de prueba, se puede utilizar la TSPLIB incluida en http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/.

Particularmente, se sugiere utilizar los conjuntos: ***brazil58.xml.zip, a280.xml.zip y att532.xml.zip*** indicados en http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/XML-TSPLIB/instances/.

Efectuar diversas evaluaciones variando los parámetros correspondientes.

Métrica de evaluación: ***longitud del camino vs tiempo de cómputo (generaciones / iteraciones)***

Se pueden plantear otras posibles variaciones del TSP, por ejemplo:

- Versión del viajante de comercio multi-objetivo, donde no solo se desea minimizar la distancia recorrida si no también minimizar el coste económico de dicho recorrido. Para ello, entre cada par de nodos habrá una distancia y un coste.

- Versión donde no sea necesario recorrer todas las ciudades y se pueda recoger viajeros, maximizando (coste-recompensas)

Problema del Cartero Chino

Resolver el problema del cartero chino para un conjunto de N aristas.

Un cartero debe repartir la correspondencia a cada una de las casas de su distrito, siendo la oficina de correos su punto de partida y llegada. Deberemos encontrar una ruta óptima para que el cartero camine la menor distancia posible.

Este problema consiste en encontrar el camino más corto que pase a**l menos una vez por cada arista del grafo**, volviendo a la posición (nodo) de partida. Es decir, se trata de que el cartero visite todas las calles para poder realizar el reparto de correo.

Nota. En este problema, la solución es desconocida pero el valor de la métrica (distancia recorrida) de la solución óptima es conocido. ¿Cuál es?

Como caso general de evaluación se pueden usar las mismas tablas que las empleadas para el problema del viajante de comercio.

Variantes: dentro del problema del cartero chino hay diversas variantes, el grafo puede ser dirigido, no dirigido, o mixto con algunos arcos dirigidos y otros no dirigidos.

Métrica de evaluación: Distancia Recorrida.

Asignación de recursos: atraque de barcos

En una Terminal de Contenedores, existe una ***lista de llegadas esperadas de barcos para el atraque***.

Como ejemplo:

Cada Barco se caracteriza por:

- Hora prevista de llegada
- Eslora (longitud de muelle a asignar).
- Movimientos (import/export) de contenedores. El tiempo de atraque para carga/descarga es M/G, (en minutos), siendo M el número de movimientos y G el número de grúas asignadas.
- Prioridad

El muelle de atraque tiene una longitud total de 700 m. y el número de grúas asignables es 7.

Ante una lista de llegada, obtener la mejor secuencia de asignación de atraque (tiempo de atraque para cada barco) y número de grúas asignadas a cada barco en toda su operación en el muelle.

El criterio de optimalidad, a minimizar, es:

$$\sum_{barcos} \textbf{Prioridad [(k*Tiempo-espera) + ((1-k)*Tiempo-atraque)],} \quad \textbf{asumiendo k=0,8}$$

Generar instancias aleatorias de prueba, con colas de 20 barcos, con densidades de llegada de [60' – 180'], nº de movimientos [100 – 1000], y esloras e [100 – 500].

Nota: No se trata de obtener una pìla de asignación de altura mínima, sino que cada bloque tenga la mínima distancia posible de su origen (tiempo de llegada)

Métrica de evaluación: Coste de la secuencia de asignación de barcos.

Metaheurísticas recomendadas: Enfriamiento Simulado, GRASP, Algoritmos Genéticos.

Generación de horarios ferroviarios

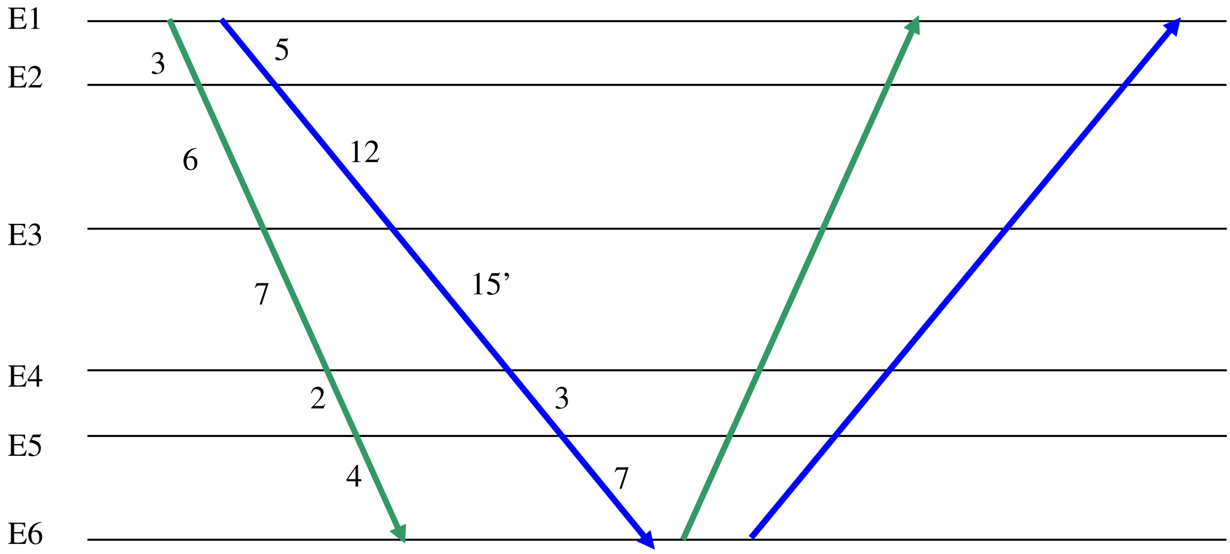
Dada una demanda de tráfico de **dos tipos de trenes**, cada uno de ellos a una determinada cadencia (cad-1 y cad-2) y tiempos de recorrido, que recorren el trayecto en **vía única** entre dos estaciones 'inicial' y 'final' en ambos sentidos de ida y vuelta, se quiere obtener, para un horizonte de 8 horas de grafiado:

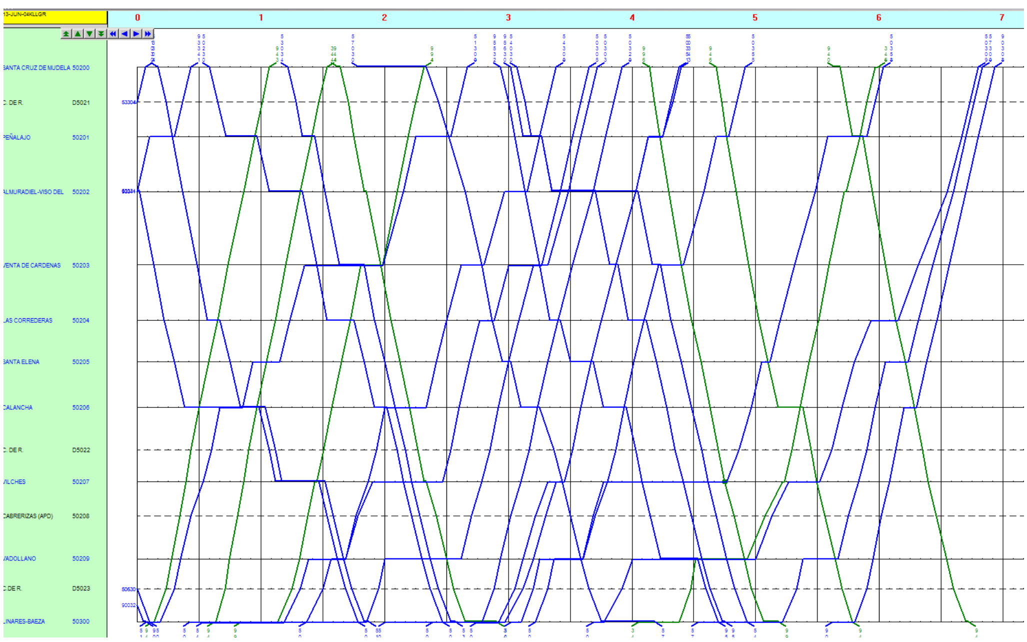
a) Un horario compatible si cad-1=25' y cad-2=30', a partir de las 8:00 y 8:10 horas, respectivamente, y desde las estaciones iniciales (E1) / finales (E6) del recorrido:
Por ejemplo (desde E1): 8:00, 8:10, 8:25, 8:40, 8:50, 9:10, 9:15, etc.
(desde E6): 8:05, 8:13, 8:30, 8:43, 8:55, 9:13, 9:20, etc.

b) La mejor hora de salida del tren inicial de ambos trenes (desde E1 y E6) en el intervalo de 8:00 a 9:00 horas, para minimizar el tiempo de recorrido.

Incremento complejidad: T-sucesión=3', T-recepción=3', T-expedición=1'

La métrica de evaluación será el tiempo total de los trenes vs tiempo de cómputo





Recomendación: Algoritmos genéticos, Grasp

El problema de los lectores y periódicos (en general, de tareas a recursos)

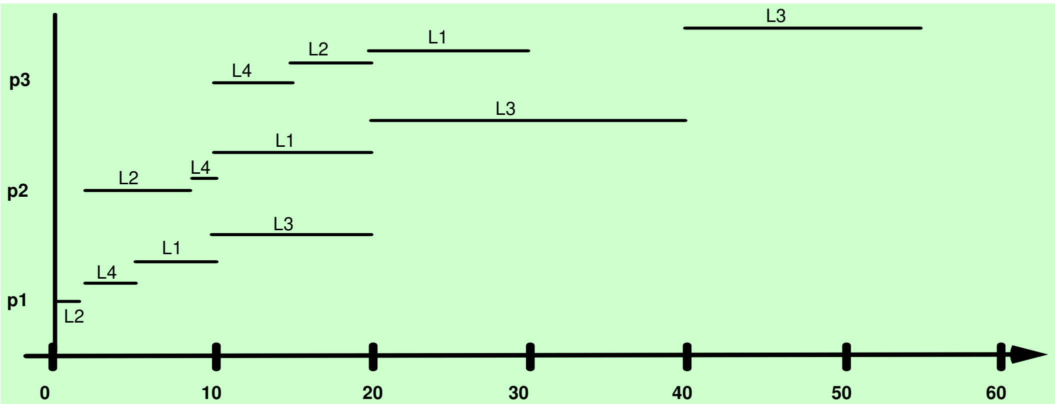
Existen 10 periódicos (P1..P10) y 10 lectores (L1..L10), que desean leer los periódicos en el mismo orden. Todos deben empezar a partir de un mismo tiempo de inicio y no pueden compartir los periódicos mientras lo leen.

El objetivo es obtener la asignación de periódicos a lectores, tal que se minimice el tiempo total en el que todos los lectores han leído todos los periódicos

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10
L1	10	15	25	12	22	66	25	35	50	12
L2	18	20	15	30	15	10	14	50	30	15
L3	20	30	10	30	55	12	40	15	12	18
L4	45	45	44	15	18	15	25	10	25	20
L5	15	12	15	22	20	20	28	18	22	20
L6	20	10	8	11	20	12	22	20	14	45
L7	12	12	6	12	10	18	40	20	12	70
L8	15	18	22	55	10	20	20	22	30	22
L9	70	30	12	18	12	20	21	12	40	30
L10	15	15	40	20	15	15	30	6	20	15

Métrica: Tiempo total requerido vs tiempo de cómputo.

Un ejemplo de resultado gráfico parcial:

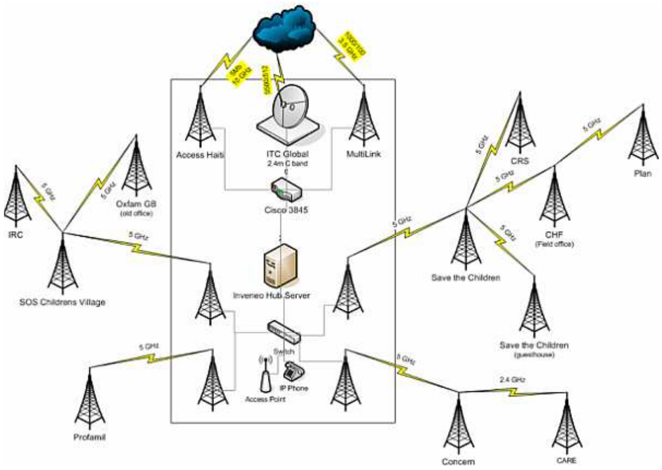


Metaheurísticas recomendadas: Enfriamiento Simulado, GRASP, Algoritmos Genéticos.

Asignación de rutas y frecuencias

El problema de asignación de Lectores/Periódicos se puede modificar para su aplicación en escenarios idealizados de ruteo de servicios de comunicaciones

En una red de comunicaciones (nodos y arcos), cada comunicación se genera en un Nodo-Origen y debe alcanzar un Nodo-Destino en la red. Para pasar de nodo a nodo se requiere una asignación de frecuencias (recurso compartido de forma no simultánea entre dos nodos).



La conectividad (bidireccional) entre nodos viene dada por una matriz, indicando número de canales (frecuencias) disponibles entre los nodos:

	Nodo-1	Nodo-2	Nodo-3	Nodo-4	Nodo-5	Nodo-6	Nodo-7	Nodo-8
Nodo-1	-	X	3	3	6	3	4	2
Nodo-2		-	4	X	3	5	3	4
Nodo-3			-	X	X	2	X	3
Nodo-4				-	5	X	4	5
Nodo-5					-	3	5	3
Nodo-6						-	3	X
Nodo-7							-	3
Nodo-8								-

Asumamos la siguiente demanda instantánea, en un tiempo inicial, con una duración de 10' cada demanda:

Nodo -Origen	Nodo-Destino	Solicitudes
Nodo-3	Nodo-7	7
Nodo-2	Nodo-4	5
Nodo-4	Nodo-6	6
Nodo-5	Nodo-3	8
Nodo-4	Nodo-3	5
Nodo-2	Nodo-1	4
Nodo-6	Nodo-8	6

Considerando que cada demanda requiere 10'' del canal, el sistema debe obtener la mejor ruta para las comunicaciones tal que su demora sea mínima y se minimice en lo posible el número de nodos intermedios).

Por ejemplo, puede plantearse las siguientes rutas:

1. Nodo-3 / Nodo-6 / Nodo-7
2. Nodo-3 / Nodo-8 / Nodo-1 / Nodo-7

   etc.

Métrica: *A) Demora en la asignación (similar al problema de atraque de barcos).*
B) Minimización de nodos intermedios.

Rutas de vehículos (Vehicle Routing Problem)

El Vehicle Routing Problem (VRP) consiste en uno o más depósitos, un conjunto de clientes y un conjunto de vehículos (que hacen reparto o recogida).

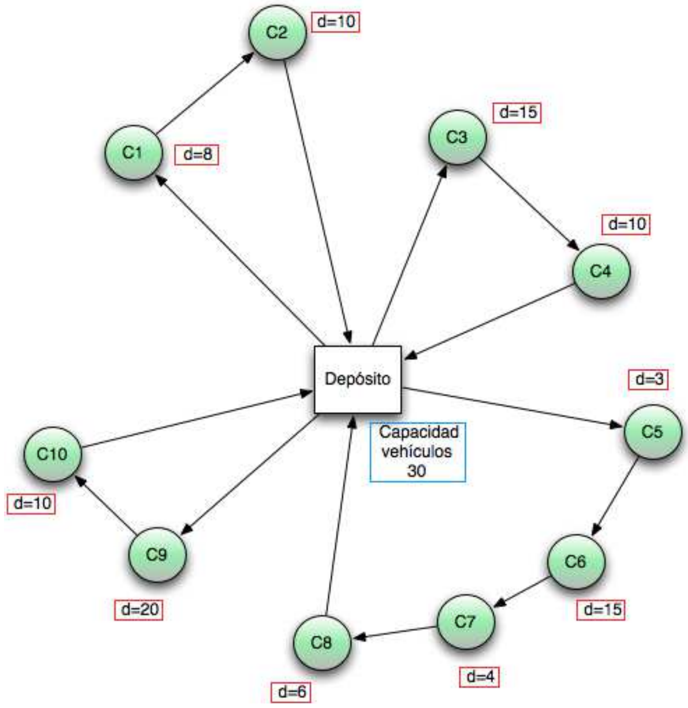
Hay que asignar a cada vehículo una ruta (es decir, una lista ordenada de clientes) de manera que todos los clientes son visitados por un (y solo un) vehículo. Cada cliente necesita una cierta cantidad del producto que se entrega, y cada vehículo puede transportar una cantidad máxima de dicho producto. Todos los clientes pueden conectarse entre sí y con los depósitos.

El problema consiste en encontrar un conjunto de rutas factibles y de kilometraje mínimo. Es un problema típico en empresas de distribución/recogida comercial.

La figura 1 muestra una posible solución a una instancia del problema con:

1 depósito, 4 vehículos con capacidad 30, y 10 clientes (cada uno con su demanda d):

CLIENTE	1	2	3	4	5	6	7	8	9	10
DEMANDA	8	10	15	10	3	15	4	6	20	10



Como ejemplo, puede asumirse el siguiente caso concreto:

Un depósito, 6 vehículos (3 con capacidad 12, 3 con capacidad 20), 20 clientes.

Cantidades requeridas por los clientes:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
8	5	9	12	7	6	10	5	2	3	7	5	6	7	2	9	7	6	8	4

Puede asumirse que la distancia de cada cliente n al deposito es 50 + 5*(n-10) y la distancia entre dos clientes (n1, n2) es [|n1 – n2| x |d1 – d2|]. Por ejemplo, la distancia entre los clientes 5 y 15 es: 10 * 5 = 50 km.

Opcionalmente, podría asumirse un coste diferente para el kilómetro recorrido por cada tipo de vehículo.

Métrica de evaluación: Longitud los recorridos.

Optimizacion del corte de paneles.

Se asume un proceso de fabricación de paneles a partir del corte de los mismos sobre una chapa de bobina continua. El material continuo se dispone sobre una superficie plana y fija (que llamamos 'mesa'), de WM mm de ancho, y LM mm de largo. En dicha mesa, se cortan los paneles en las medidas y cantidad que solicita en los pedidos.

Los datos de cada pedido son:

a) Cantidad de cada tipo de panel a cortar. Cada tipo viene caracterizado por su longitud L, siendo su ancho fijo.
b) Longitud máxima de la mesa. Un caso típico es que sea de LM=12000 mm, pero puede ser inferior (incluso 6000 mm), dependiendo del tipo de chapa o material que se utiliza.

Los paneles pueden tener cualquier longitud (ancho fijo de W mm), sin superar el máximo indicado para la mesa, (LM) y siempre se cortan en múltiplos de 300 mm, excepto el último panel. La diferencia entre la medida del panel y un múltiplo de 300 es desperdicio. No se considera desperdicio, el sobrante del último panel que cortamos de cada mesa, al asumirse que entra en el siguiente proceso de corte. Por ejemplo, tres paneles de medidas 2820, 2810 y 2200, darían lugar a una mesa de: (2820+180)mm + (2810+190)mm + 2200mm = 8200mm (total chapa utilizada, desperdicio, (180+190)=370mm

El sistema a desarrollar debe planificar el corte de los paneles de un pedido (compuesto de paneles de diversas longitudes y cantidades), de forma que se optimice la utilización de materiales para fabricar dicho pedido. Como ejemplo:

Parámetros:		Longitud máxima: 12000 Criterio optimización: Mesas | Desperdicio | Sierras | Equilibrado		
Paneles:	N	ID. PANEL	CANTIDAD	LONGITUD
	1	P3	28	4.930mm
	2	P4	14	4.270mm
	3	T9	6	5.900mm
	4	CUC-1	1	5.900mm
	5	T10	9	4.385mm

Se podrán establecer distintos 'Criterios de Optimización', lo que permitirá tener diversas soluciones alternativas, a fin de elegir la más conveniente en cada momento. Los criterios pueden ser:

a) Minimizar nº de mesas, manteniendo desperdicio y nº sierras en valores razonablemente bajos.
b) Minimizar desperdicio total, manteniendo nº de mesas y nº sierras en valores razonablemente bajos.
c) Minimizar nº sierras, manteniendo el nº de mesas y nº sierras en valores razonablemente bajos.

O bien, una solución equilibrada, en las que se optimicen de forma ponderada los tres criterios. Un posible resultado, seria:

ID	Sierra	Long. Mesa	Merma Mesa	Mesas Requeridas	Long Paneles Fabr.	Merma	Long Total Requerida
SIR1-P1-P10-P4	1-P2720-1-P2390-1-P2470	7820	240	4	31280	960	30320
SIR2-P2-P10-P4-P4	1-P2150-1-P2390-1-P2470-1-P2470	9950	470	1	9950	470	9480
SIR3-P2-P10-P5-P5	1-P2150-1-P2390-1-P2500-1-P2500	9950	410	3	29850	1230	28620
SIR4-P3-P10-P5	1-P2760-1-P2390-1-P2500	7860	210	4	31440	840	30600
SIR5-P4-P9	1-P2470-1-P5085	7570	15	9	68130	135	67995
SIR6-P4-P8-P5	1-P2470-1-P2355-1-P2500	7570	245	4	30280	980	29300
SIR7-P4-P7-P5	1-P2470-1-P2920-1-P2500	8170	280	2	16340	560	15780
SIR8-P6-P6-P6	1-P2905-1-P2905-1-P2905	9000	190	4	36000	760	35240
SIR9-P7-P7	1-P2920-1-P2920	5920	80	1	5920	80	5840
Nº de sierras medidas diferentes 9				32	259190	6015	253175

Para su contraste, se disponen der diversos resultados reales.

OTRAS

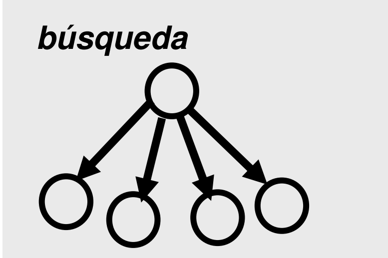
PROPUESTAS

Incluyendo:

Problemas propuestos por el alumnado

Se puede trabajar con cualquier otro problema propuesto por el alumnado. Para ello primero habrá que proponerlo y comentarlo con el profesor de la asignatura, quien analizará la viabilidad de la propuesta.

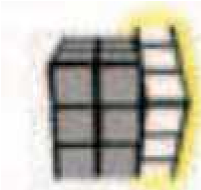
Cubo de Rubick



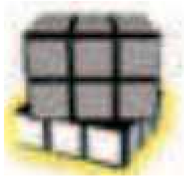
Recomendación de resolución:
Algoritmo IDA. Algoritmo A.

Movimientos (6):

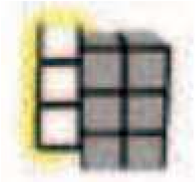
1. Mover Cara Derecha (1/4 de vuelta)



2. Mover Cara Inferior(1/4)



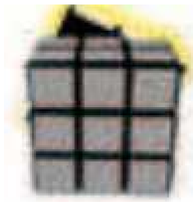
3. Mover Cara Izquierda (1/4)



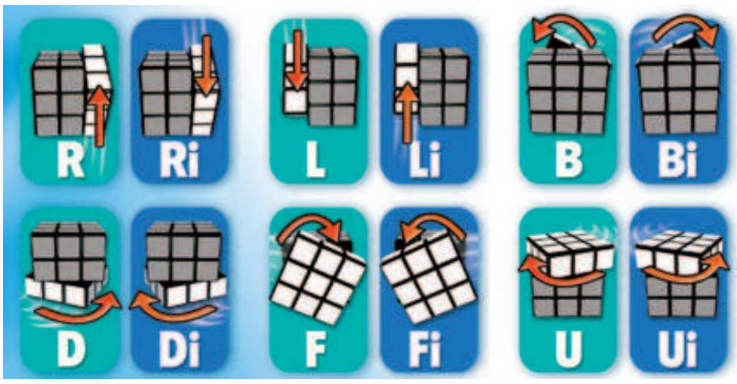
4. Mover Cara Frontal (1/4)

5. Mover Cara Superior (1/4)

6. Mover Cara Posterior (1/4)



Más los 6 Movimientos Inversos

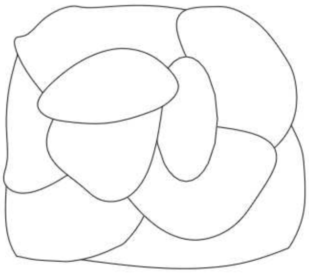


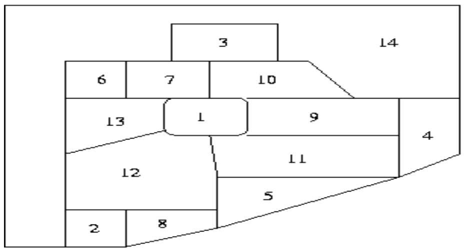
Notas:

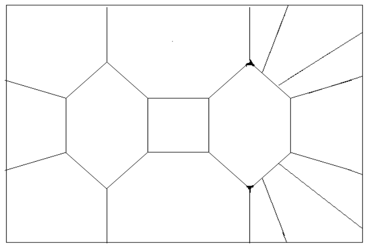
- El problema puede simplificarse considerando un cubo 2x2.
- Métrica de evaluación: Tiempo de cómputo, Memoria requerida, en diversos casos de prueba
- Diversos estudios han ido aproximando el número mínimo de movimientos requeridos para resolver cualquier estado inicial *(por algunos llamado el "número de Dios")*. Actualmente, el número mínimo es 20 movimientos (www.cube20.org).
  - 20 niveles, 12 sucesores: 10^{20} estados finales. Realmente, $43*10^{18}$ posibles estados. Calculando un estado cada 10^{-9} seg (1.000.000.000 estados/segundo), 1.400 años.
- Reto: http://www.recordholders.org/en/list/rubik.html

Coloreado de mapas (K-Coloring Graph)

El problema base trata de colorear las distintas regiones de un mapa de forma que dos regiones adyacentes no tengan el mismo color, con un mínimo número (k) de colores posible.







El mapa puede ser visto como un grafo no dirigido G=(V, E), donde los arcos (E) representan los bordes entre las regiones o vértices del grafo (V).

$G(V, E), \quad V=\{v_1,v_2,\ldots,v_n\}\ , \ E \subset V \times V$

La solución es una partición de los vértices V en k conjuntos disjuntos **(C_1, C_2, … , C_k)**, también llamados colores, tal que ninguno de los vértices incluidos en un C_j están unidos por un arco en G.

Un **k-coloración equitativa de G** es cuando los colores están equitativamente repartidos, tal que el número de regiones coloreadas por uno u otro color difiere como máximo en uno. Es decir,

$|C_i| - |C_j| \leq 1, \ \forall\ i \neq j.$

El problema es determinar el número mínimo de colores (k) para un grafo G determinado.

Este problema tiene una amplia variedad de aplicaciones: recogida de basura, separación y equilibrio de cargas, scheduling, etc.

> *Ver:* ***'Backtracking based iterated tabu search for equitable coloring'. Lai, Hao, Glover. Engineering Applications of Artificial Intelligence v.46, November 2015.***

Ver instancias de problemas en:

- **DIMACS Graphs: Benchmark Instances and Best Upper Bounds (http://www.info.univ-angers.fr/pub/porumbel/graphs/)**
- **Graph Coloring Benchmark Instances (http://www.cs.hbg.psu.edu/txn131/graphcoloring.html)**

Metaheurísticas recomendadas: Tabú Search, Enfriamiendo Simulado, Algoritmos Genéticos.

Métrica de evaluación: *Para un problema dado, mínimo número (k) obtenido vs tiempo de cómputo.*

Nota: K-Coloring Graph es un problema NP-hard. Ha sido probado que un grafo G=(V, E) tiene un (a+1)-coloración equitativa (nº de colores requerido), donde:

$a = \max_{v \in V} \{d(v)\}$, donde d(v) es el grado (arcos) del vértice v

Por otra parte, de acuerdo al Teorema de 4-colores, "Cualquier mapa, con cualquier numero de zonas, se puede colorear con 4 colores tal que dos zonas adyacentes no tengan el mismo color"

Obtención de sendas optimizadas en el problema del 8-puzzle

EL problema del puzzle es un típico problema de IA en el que el objetivo es obtener la secuencia de pasos necesaria para alcanzar un estado final ***meta*** desde un dado estado inicial.

3	8	4
1		5
7	2	6

→

1	2	3
8		4
7	6	5

Una búsqueda A* es muy costosa para estados iniciales complejos, con longitudes de senda largas. Entonces, se requiere sobreponderar la heurística, realizar una búsqueda local, o utilizar procedimientos metaheurísticos que mejoran los métodos heurísticos locales.

Aplicar uno de los procedimientos metaheurísticos vistos para obtener sendas optimizadas en el problema del 8-puzzle.

Caso de utilizar un método de mejora de soluciones, las soluciones iniciales de partida pueden obtenerse mediante ‘alargamientos’ de soluciones optimas, obtenidas mediante ‘pasos_inversos’ desde el estado final.

1. ***Diseñar y aplicar un método metaheurístico. Analizar las soluciones obtenidas.***

2. ***Contrastarlo con soluciones obtenidas mediante búsquedas A* y A.***

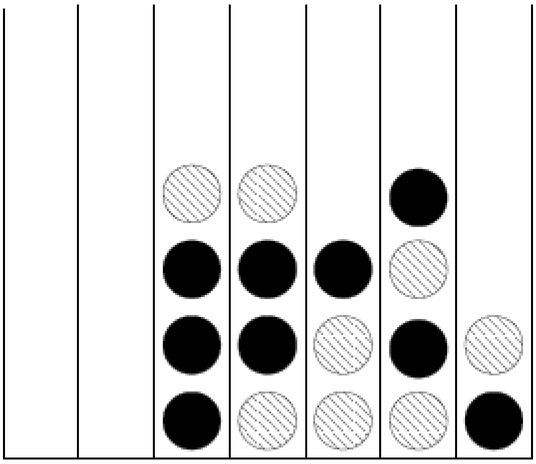
Conecta-4: juegos con adversario (minimax, alfa-beta)

Conecta-4 es un juego de dos jugadores que utiliza una tabla vertical con 6 filas y 7 columnas.

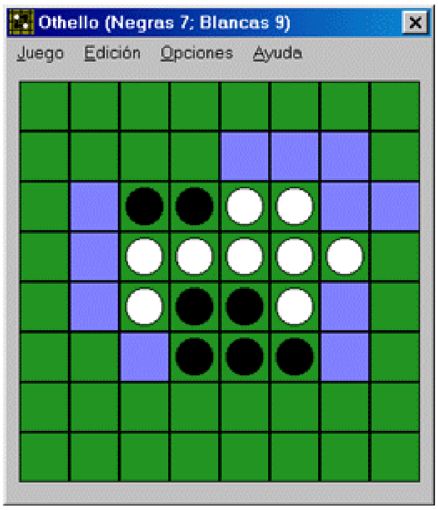
Cada jugador tiene 21 discos, cada jugador de un color (blanco, negro). En cada jugada, un jugador coloca un disco en una de las columnas que no esté llena, tal que el disco ocupará la posición más baja de la columna.

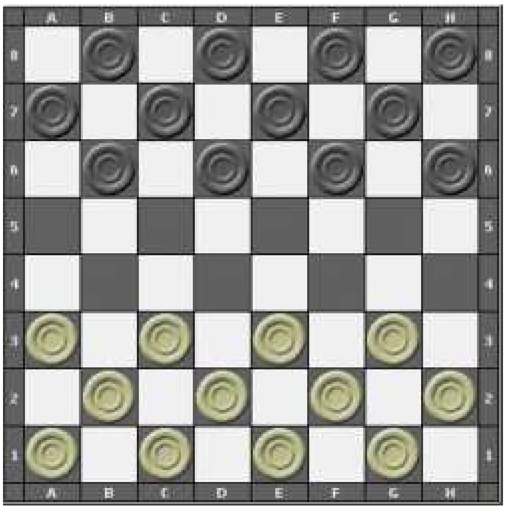
Un jugador gana cuando consiga colocar 4 de sus discos consecutivamente en una línea (fila, columna o diagonal). El juego acabaría en tablas, si se llenan las columnas sin que gane ningún jugador.

Ejemplo de estado en el que las fichas negras acaban de ganar:



Otras alternativas de juegos: Otello, Oware, damas…

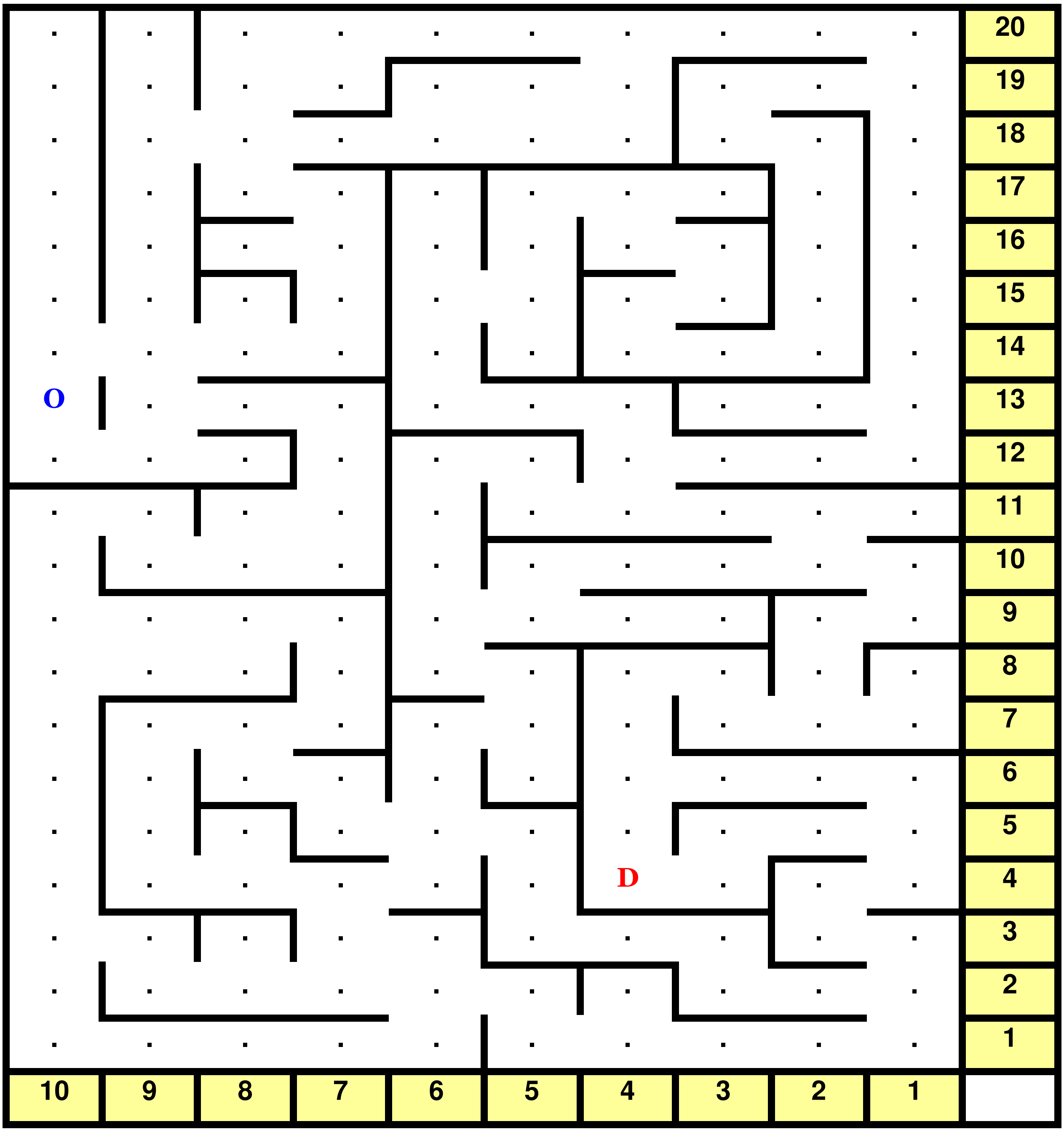




Senda óptima en un laberinto

Obtener una senda (optimizada) en un laberinto:

Aplicarlo a este mapa (10 x 20), obtener una senda optimizada entre O y D:



Puede aplicarse sobre otros mapas, o bien pueden obtenerse mapas típicos en:

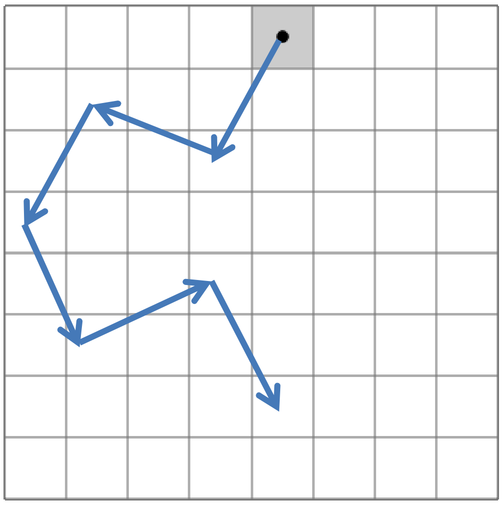
https://code.google.com/p/straightedge/ o en http://www.movingai.com/benchmarks/

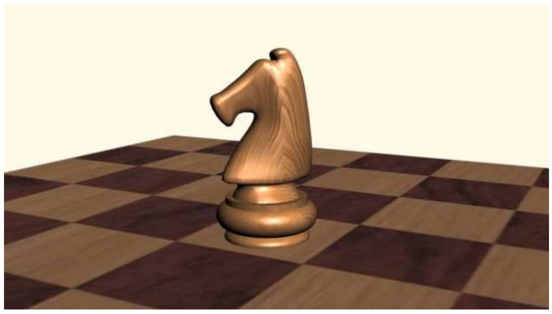
Métrica de evaluación: Coste de la senda vs tiempo de cómputo

Recorrido de un tablero mediante movimientos del caballo de ajedrez

Este problema puede verse como una modificación del problema de Viajante.

El problema del caballo de ajedrez consiste en encontrar una secuencia de 64 movimientos que pase por cada una de las casillas de un tablero de ajedrez (solo una vez) saltando según el movimiento de un caballo de ajedrez.





Método de Resolución: No emplear algoritmos matemáticos, sino metaheurísticas para la obtención de sendas.

Métrica: Nº de soluciones encontradas vs Tiempo de Cómputo.

Notas:

El problema no es difícil a la hora de encontrar una solución, pero destaca el enorme número de soluciones posibles y que van desde la primera conocida, que data del siglo IX, hasta las últimas en las que se emplearon potentes ordenadores.

Las soluciones se disparan si uno puede empezar y terminar en cualquier casilla del tablero y se acotan cuando el recorrido es cerrado.

Con la ayuda de 20 ordenadores y más de cuatro meses de trabajo, Martin Löbbing e Ingo Wegener encontraron una cifra de soluciones que superaba los 33 billones. Dos años más tarde, en 1997, el profesor australiano Brendan McKay estudió el tablero dividiéndolo en dos mitades y obtuvo otra cifra impresionante pero algo menor, 13 billones de soluciones.

Lo que nadie ha hecho hasta el momento es encontrar un algoritmo que encierre una pauta de movimientos válidos, ni por supuesto, presentar la lista completa de resultados.

Problema de cartas

El problema de las cartas consiste en un conjunto de 10 cartas, numeradas de 1 a 10.

Hay que dividir las cartas en dos pilas, tal que la suma de las cartas en la pila 0 sea lo más cerca posible de 36, y el producto de las cartas en la pila 1 sea lo más cerca posible de 360.

$$\sum Pila_0 \rightarrow 36 \qquad \prod Pila_1 \rightarrow 360$$

Variantes: se pueden establecer otras restricciones en las pilas. ¿Qué pasaría si añadimos demasiadas restricciones al problema?