TECNICAS INTELIGENCIA ARTIFICIAL

PROBLEMAS PROPUESTOS (2023-2024)

Propuestas de Aplicaciones:

Metaheurísticas: Alg. Genéticos y Enfriamiento Simulado.

Diseño, Desarrollo y Evaluación. Casos de Prueba. Comparativa.

Alternativamente,

a) **Pueden/deben proponerse otras aplicaciones alternativas.**

b) **Pueden desarrollar y aplicarse otras metaheurísticas opcionales:**

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

- **Búsqueda Tabú**
- **Enfriamiento Simulado**
- **Búsqueda en Haz**

Inteligencia de Enjambre

- **Alg. de las Hormigas**
- **Enjambre de Partículas**

Metaheurísticos Híbridos: Constructivos + Mejora:

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

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

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

Propuestas de Aplicaciones (para dar ideas)

1. **Problemas de recubrimiento. Bloques rectangulares *(4 variantes)***
2. **Asignación optimizada de turnos 24h *(complejo)***
3. **Capas de 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. **Coloreado de mapas**
12. **Asignación de carga**
13. **Rutas de Vehículos *(complejo)***
14. **SUDOKU (como contraste a CSP)**
15. **Otras alternativas. Problema propuesto por alumnado**

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 <u>implementación desde cero</u> 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. ***Adaptación de la metaheurística. Diseño del algoritmo. Modelización del problema y evaluación de soluciones***
2. ***Adecuada parametrización de la metaheurística.***
3. ***Resultados obtenidos (métricas). Fundamentalmente, calidad de la respuesta vs. Soluciones generadas/tiempo de cómputo***
4. ***Diversidad de tallas/casos de prueba (benchmarks). Variación de parámetros.***
5. ***Conclusiones obtenidas***
6. ***Memoria entregada. Presentación de los resultados***

Esquema de memoria (y presentación: < 10 minutos)

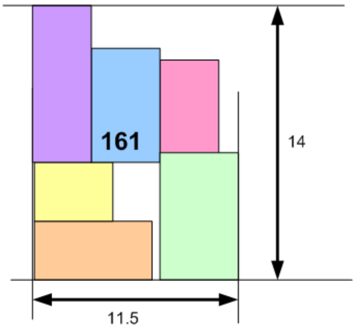
a) ***Descripción del problema (1 transparencia).***
b) ***Descripción del método aplicado. Diseño del algoritmo y Parámetros (2-4 transparencias)***
c) ***Implementación de la solución o utilización de entorno generalista (1-2 transparencias)***
d) ***Evaluación. Casos de prueba y métricas comparativas. Discusión (3-4 transparencias)***
e) ***Conclusiones (1 transparencia)***

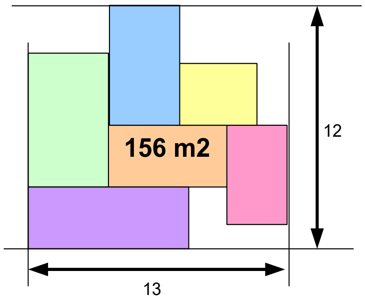
<u>ENTREGA (poliformat)</u>:

- **Memoria, presentación, código ejecutable y/o fuente.**
- **Habrá una presentación del trabajo en la última sesión de prácticas de la asignatura (o videopresentación)**

1. 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 (Caso c, 60 x 60), se pueden asumir los casos siguientes para bloques Alt x Anc:

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 más bloques ejemplo

Métrica de evaluación: Superficie ocupada vs tiempo de cómputo.

Notas:

- Se sugiere determinar la posición de cada bloque mediante la asignación de alguna de sus esquinas, o mediante la 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 tipo suele llamarse **Empaquetamiento** (**Strip packing**), que intenta minimizar la superficie (o volumen, en caso 3-D) o número de contenedores requerido.

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

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

2. 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, por ejemplo:

JUNIO																	
Cod.	Nombre	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
		L	M	X	J	V	S	D	L	M	X	J	V	S	D	L	M
1		N	N	-	-	T	T	T	T	T	-	-	M	M	M	M	M
2		-	M	M	M	M	-	-	-	-	-	-	N	N	N	N	-
3		M	M	M	M	M	-	-	M	M	M	M	M	-	-	-	-
4		M	M	M	M	M	-	-	-	-	N	N	-	-	-	T	T
5		T	T	T	T	T	-	-	-	T	T	T	T	-	-	-	-
6		N	N	-	-	-	-	-	T	T	T	T	T	-	-	N	N
7		-	-	-	-	N	N	N	N	-	-	-	M	M	M	M	M
8		-	-	-	-	N	N	N	N	N	-	-	T	T	T	T	T
9		-	-	-	-	T	T	T	T	T	-	-	M	M	M	M	M

JUNIO																		JUNIO			
Cod.	17	18	19	20	21	22	23	24	25	26	27	28	29	30	1	2		M	T	N	Tot
	X	J	V	S	D	L	M	X	J	V	S	D	L	M	X	J					
1	-	-	-	-	-	N	N	N	N	-	-	-	-	-	T	T		5	5	6	16
2	-	-	-	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	V	V		20	0	0	20
4	T	T	T	-	-	-	-	N	N	N	N	N	-	-	-	M		5	5	7	17
5	-	-	M	M	M	M	M	-	-	-	M	M	M	M	M	-		9	9	0	18
6	N	N	-	-	-	-	-	M	M	M	M	M	-	-	T	T		5	5	6	16
7	-	-	T	T	T	T	T	-	-	-	-	-	T	T	V	V		5	7	4	16
8	-	-	N	N	N	N	N	-	-	-	-	-	T	T	T	T		0	7	10	17
9	-	-	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 2 a 5 turnos de M/T/N, con un mínimo de DOS turnos de descanso entre cada secuencia de turnos.
- Los máximos turnos de trabajo que puede hacer cada trabajador en las 4 semanas es de 20 turnos.
- Como restricciones adicionales (optativas):
    a) No puede 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.

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

1. Se intentará cubrir una demanda 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.

<u>NOTAS:</u>

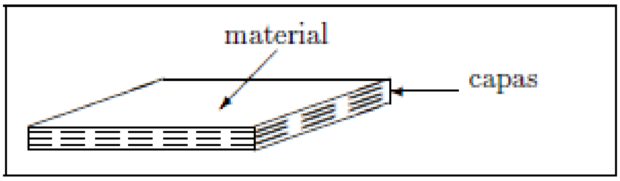
- Se recomienda ***inicializar soluciones con GRASP*** y mejorarlas con el algoritmo metaheurístico.
- ***Métrica de evaluación: (i) Turnos asignados vs tiempo de cómputo, (ii) Equilibrios***
- ***Hay diversas variantes: M/T/N, Entornos sanitarios, Localización Física, etc.***

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

3. 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., en muy diversos objetos (aviones, ropa, material ignifugo, militar, 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 aislamiento 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. Es una caso muy simple: las combinaciones posibles son 12!= 479.001.600

Caso b) Se puede 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 de las capas vs Tiempo de cómputo.

4. 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. Solo se puede hacer una división si el resultado es entero. En función de la codificación utilizada se puede optar por tener, o no, 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. **Nota**: este problema es un caso simplificado del problema de *symbolic regression* para encontrar expresiones matemáticas que se adapten a un determinado valor.

Casos concretos a probar:

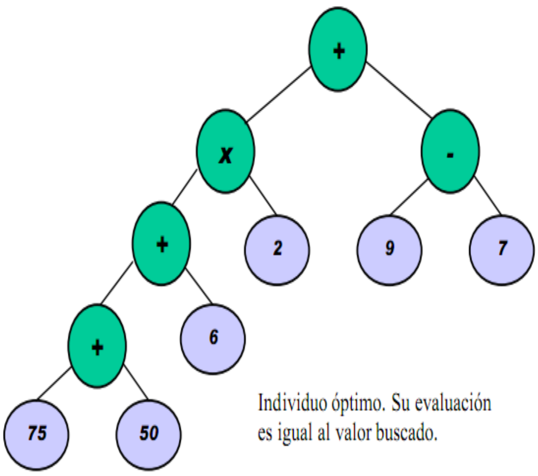
Casos Fáciles							OBJETIVO
F1	4	10	7	9	2	25	232
F2	10	2	9	5	7	100	298
F3	2	75	9	6	100	8	474
F4	3	10	7	6	75	10	381
F5	7	3	50	25	6	100	741
F6	2	4	75	4	50	2	502
F7	8	25	10	2	9	4	549
F8	7	10	3	2	1	25	223
F9	7	4	100	5	9	10	156
F10	2	10	7	50	100	10	259

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:

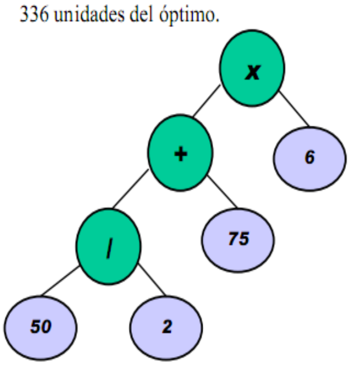
- Se recomienda usar notación polaca.
- ***Métrica: Tiempo de cómputo (vs. aproximación al objetivo)***

Ejemplos (para el Caso D1):

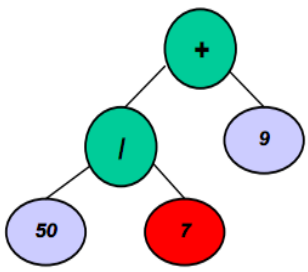


Individuo óptimo. Su evaluación es igual al valor buscado.

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

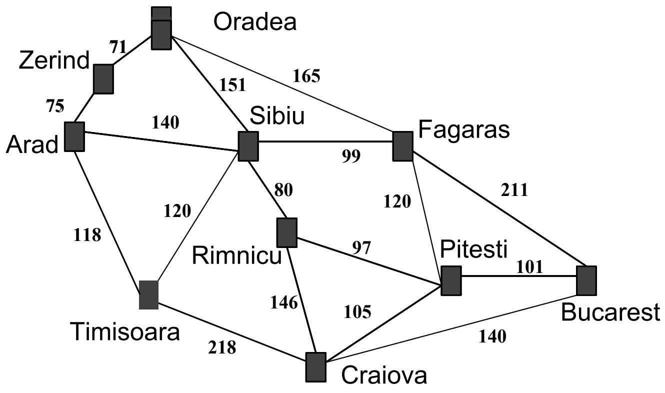


Individuo erróneo. La división no se puede utilizar si no tiene un resultado entero.



5. Problema del Viajante de Comercio (TSP, camino hamiltoniano)

Resolver el TSP 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, etc.

Obtener otros ejemplos (que incluyen resultados para poder contrastar) de:

- http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95
- http://www.math.uwaterloo.ca/tsp/world/countries.html

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/tsp/

Particularmente, se sugiere utilizar los conjuntos: ***berlin52.tsp.gz***, ***ch130.tsp.gz*** ***y*** ***in318.tsp.gz***. Los mejores resultados se pueden encontrar en http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/STSP.html

Efectuar diversas evaluaciones variando los parámetros correspondientes.

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

Importante. Si se elige este problema, hay que plantear alguna variación al problema original 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): **Ride-sharing**
- Versión donde no todas las ciudades están conectadas entre sí, etc.

6. Problema del Cartero Chino (circuito euleriano)

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 óptima es desconocida pero el ***valor de la métrica (distancia recorrida) de la solución óptima es conocido***. Si cada nodo del grafo es de grado par, esta solución es llamada *camino euleriano*: camino no cerrado y que pasa por todas las aristas del grafo una y solamente una vez.

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

Otras 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, que se desea minimizar.

7. Planificación atraque barcos (asignación de recursos)

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/2, (en minutos), siendo M el número de movimientos.
- Prioridad

El muelle de atraque tiene una longitud total de 700 m.

Ante una lista de llegada, obtener la mejor secuencia de asignación de atraque (tiempo del atraque para cada barco).

El criterio de optimalidad, a minimizar, es:

$\sum_{\text{barcos}}$ **[Prioridad * (Tiempo-Atraque - Tiempo-llegada)]**

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 asignación con un tiempo total de proceso óptimo (pila 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

8. Generación de horarios ferroviarios

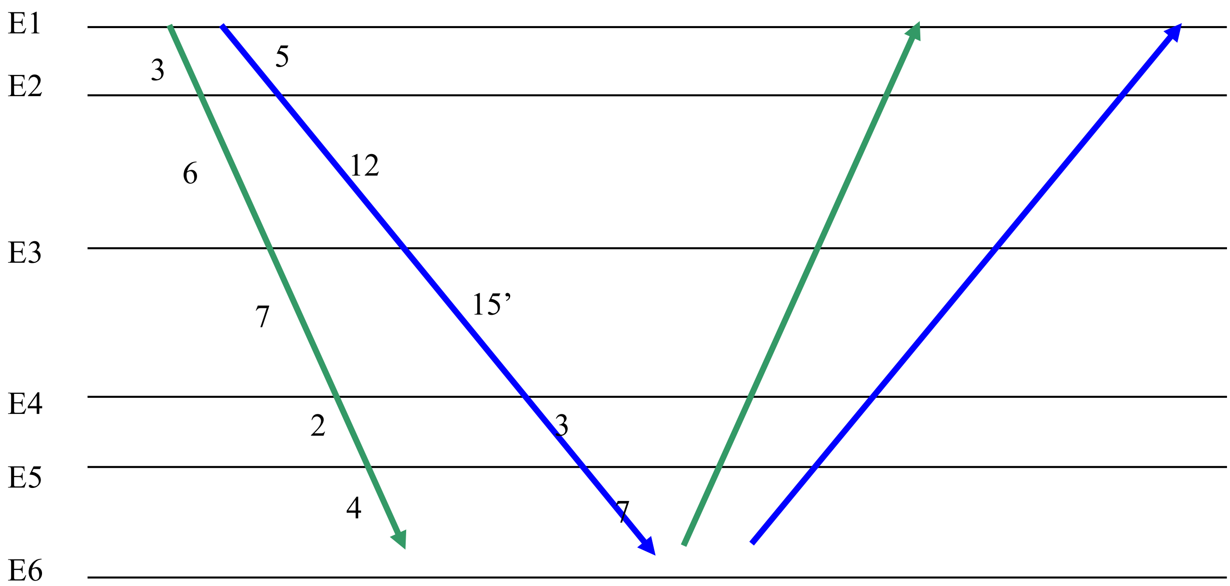
Disponemos de una demanda de tráfico de **dos tipos de trenes**, cada uno de ellos con una determinada cadencia (cad-1 y cad-2) y tiempos de recorrido, que recorren el trayecto en **vía única** entre dos estaciones 'inicial' y 'final' en ambos sentidos de ida y vuelta. Se quiere obtener, para un horizonte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

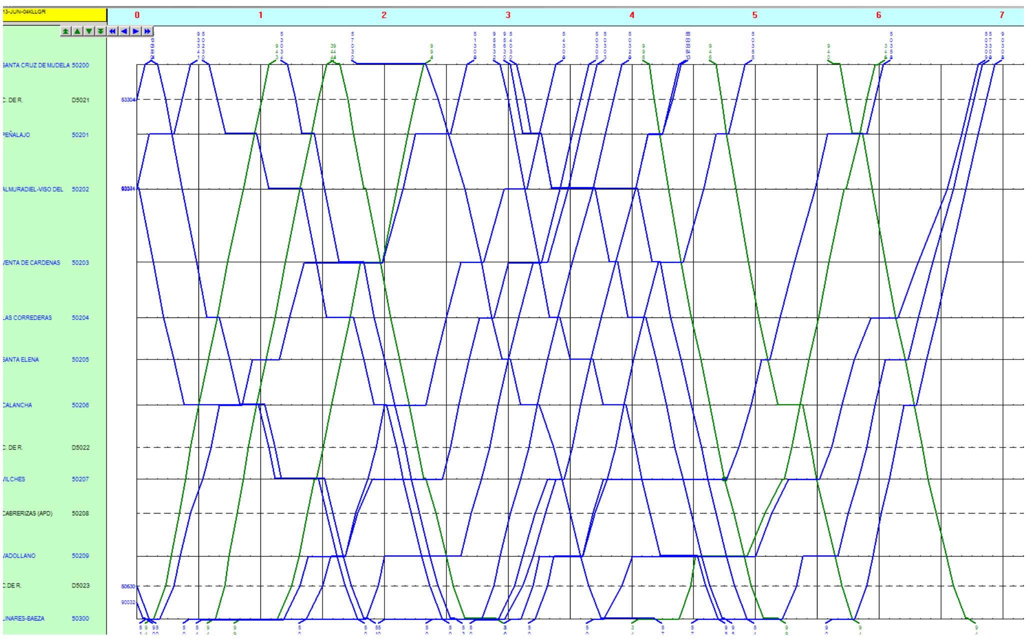
a) Un horario compatible si cad-1=25' y cad-2=30', a partir de las 8:00 y 8:10 horas, respectivamente, y desde las estaciones iniciales (E1) / finales (E6) del recorrido:
   Por ejemplo (desde E1): 8:00, 8:10, 8:25, 8:40, 8:50, 9:10, 9:15, etc.
   (desde E6): 8:05, 8:13, 8:30, 8:43, 8:55, 9:13, 9:20,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





9. Asignación de lectores y periódicos (en general, asignar tareas a recursos)

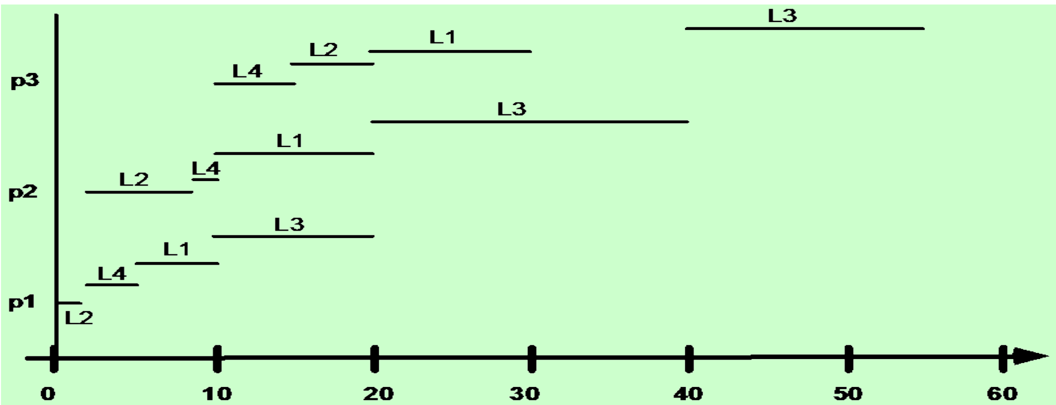
Existen 10 periódicos (P1..P10) y 10 lectores (L1..L10), que desean leer los periódicos en el mismo orden. Todos deben empezar a partir de un mismo tiempo de inicio y no pueden compartir los periódicos mientras lo leen.

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.

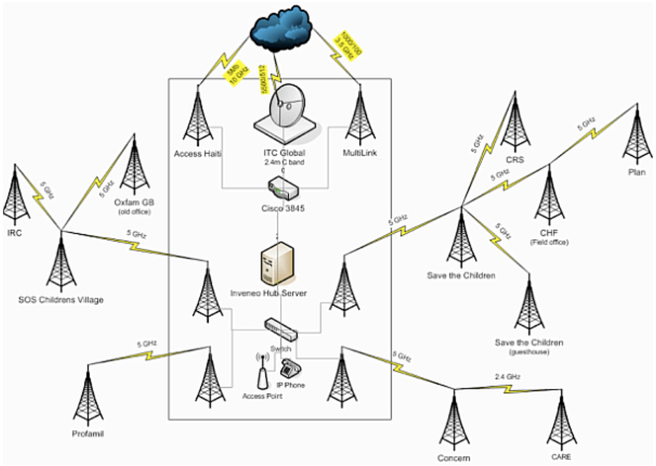
Un ejemplo de resultado gráfico parcial:



10. Asignación de rutas y frecuencias en comunicaciones

El problema de asignación de Lectores/Periódicos *(asignación de tareas a recursos limitados)* 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:

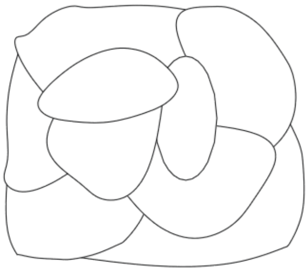
1. Nodo-3 / Nodo-6 / Nodo-7
2. Nodo-3 / Nodo-8 / Nodo-1 / Nodo-7

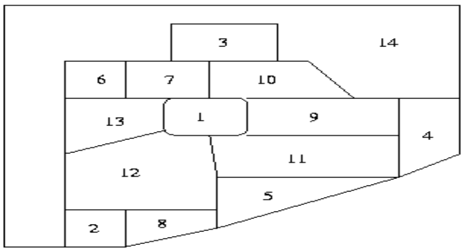
etc.

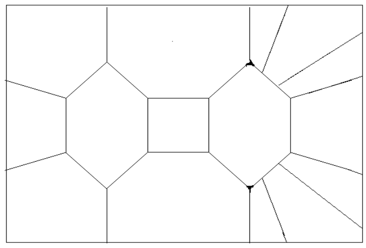
Métrica: A) Demora en la asignación (similar al problema de atraque de barcos).
B) Minimización de nodos intermedios.

11. 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 4 colores posibles.







De acuerdo al Teorema de 4-colores, “Cualquier mapa plano, con cualquier número de zonas, se puede colorear con 4 colores tal que dos zonas adyacentes no tengan el mismo color (siempre y cuando cualquier frontera tiene una determinada longitud, no un punto)”.

Básicamente se trata de un problema de ***satisfactibilidad*** y el objetivo es encontrar una solución factible en el menor tiempo posible. No obstante, este problema se puede convertir en un problema de optimalidad y así utilizar el mínimo número posible de colores. Durante el proceso de obtención de la solución, la optimalidad puede relacionarse con el número de zonas adyacentes con el mismo color (número de restricciones violadas)

Notas:

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).

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

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,

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

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

Ver instancias de problemas en:

- ***‘Backtracking based iterated tabu search for equitable coloring’. Lai, Hao, Glover. Engineering Applications of Artificial Intelligence v.46, November 2015.***

12. Asignación de carga (asignación lineal)

El problema trata de asignar un conjunto de objetos, de diferentes pesos, a un conjunto de vehículos, cada uno de ellos de diferente capacidad de carga.

- Un conjunto de objetos {Oi} de diferentes pesos deben ser cargados en un conjunto de camiones.
- Cada camión tiene un máximo peso que puede cargar {Ci}
- Se quiere obtener la distribución de los objetos en los camiones, asumiendo que no caben todos los objetos y se quiere maximizar el número de objetos cargados.

{O1, O2, O3, …., On} ⇒ {C1, C2, C3, ….. Cm}

¿?

Casos de prueba

Peso Objetos 1..50: [34, 34, 81, 72, 42, 51, 19, 61, 63, 43, 53, 87, 48, 61, 33, 41, 44, 58, 56, 67, 24, 34, 21, 32, 12, 31, 19, 61, 23, 43, 13, 27, 38, 21, 33, 41, 24, 28, 16, 47, 10, 15, 17, 19, 21, 23, 37, 18, 29, 31]

Capacidad Camiones 1..5: [220, 260, 280, 330. 180]

(*) Duplicar los objetos y disponer de Camiones 1..5: [420, 360, 380, 530. 380]

Ampliación-1: Priorizar objetos

Ampliación-2: Considerar volúmenes

Ampliacion-3: Caben todos los objetos, Exceso de camiones, pero con coste de uso.

Es un típico problema de logística, de gran interés y con múltiples variantes.

13. Rutas de vehículos (Vehicle Routing Problem)

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

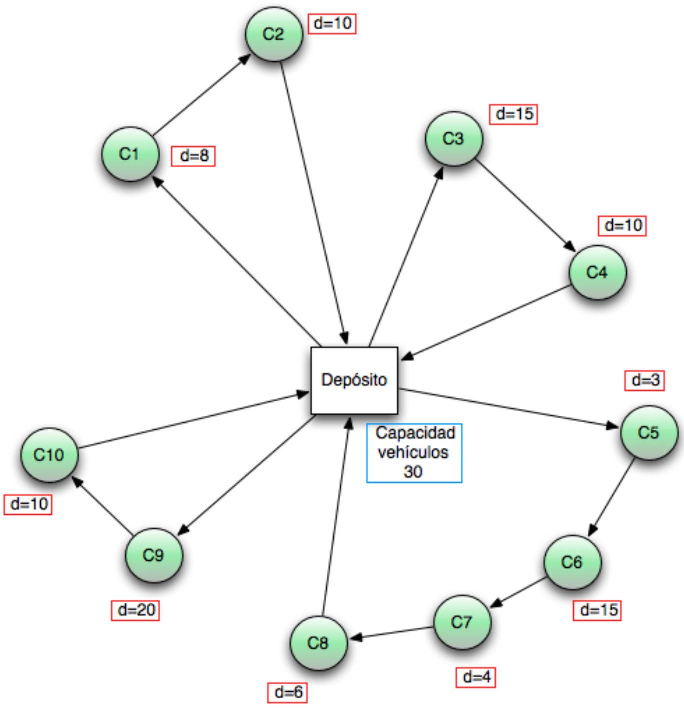
Hay que asignar a cada vehículo una ruta (es decir, una lista ordenada de clientes) de manera que todos los clientes son visitados por un (y solo un) vehículo. Cada cliente necesita una cierta cantidad del producto que se entrega, y cada vehículo puede transportar una cantidad máxima de dicho producto. 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 siguiente figura 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 y 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 *d* 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|] o cualquier otra expresión.

Opcionalmente, podría asumirse un coste diferente para el kilómetro recorrido por cada tipo de vehículo.
Métrica de evaluación: longitud de los recorridos planificados.

14. SUDOKU: búsqueda de la asignación factible

¿Factibilidad? ¿Optimalidad?

```
+---------+---------+---------+
|         |         |       9*|
|    9*   |         |    3*   |
| 5* 1* 3*| 9*      |         |
+---------+---------+---------+
|       6*|    7*   | 4* 1* 5*|
|    4*   | 5* 6*   |         |
|       8*|         | 7* 6*   |
+---------+---------+---------+
|         |    3* 2*| 1*      |
|         | 7*      |    9* 6*|
| 7* 3*   | 1*    6*| 2*    4*|
+---------+---------+---------+
```

```
+---------+---------+---------+
| 8  7  2 | 6  1  3 | 5  4  9*|
| 6  9* 4 | 2  5  7 | 8  3* 1 |
| 5* 1* 3*| 9* 8  4 | 6  7  2 |
+---------+---------+---------+
| 9  2  6*| 3  7* 8 | 4* 1* 5*|
| 3  4* 7 | 5* 6* 1 | 9  2  8 |
| 1  5  8*| 4  2  9 | 7* 6* 3 |
+---------+---------+---------+
| 4  6  9 | 8  3* 2*| 1* 5  7 |
| 2  8  1 | 7* 4  5 | 3  9* 6*|
| 7* 3* 5 | 1* 9  6*| 2* 8  4*|
+---------+---------+---------+
```

El problema es obtener la asignación solución, que cumpla la condición de satisfactibilidad del juego, mediante la técnica metaheurísitica utilizada.

El Sudoku no es un problema idóneo de optimización sino más bien de factibilidad, pero puede utilizarse para aplicar una metaheurística de optimización (enfriamiento simulado, algoritmo genético, etc.) minimizando el número de restricciones violadas. Es más, si tratamos de resolver Sudokus con tallas mayores a 9x9 llegará el momento en el que habrá que relajar restricciones porque no podremos encontrar una solución factible en un tiempo razonable.

Referencias:

- **Solving Sudoku With Genetic Algorithms. http://fendrich.se/blog/2010/05/05/solving-sudoku-with-genetic-algorithms**

- **[Metaheuristics can solve Sudoku puzzles](). Lewis, Rhydian. (2007) J. Heuristics. 13. 387-401. 10.1007/s10732-007-9012-8**