Metaheurísticas

Seminario 1. Ejemplos de resolución de problemas con metaheurísticas: problemas clásicos y reales. Software de metaheurísticas

1. Introducción: Optimización
2. Problemas de Optimización Combinatoria Clásicos
3. Algunos Ejemplos de Problemas de Optimización Reales
4. Software de Metaheurísticas

1. Introducción: Optimización

✓ Lenguaje coloquial, **optimizar** significa más o menos **mejorar**

✓ Contexto científico: la **optimización** es el proceso de tratar de encontrar la mejor solución posible para un determinado problema

- Problema de optimización:

✓ Diferentes **soluciones**, un criterio para **discriminar** entre ellas y el **objetivo** es encontrar la mejor

> *Encontrar el valor de unas variables de decisión (sujeto a restricciones) para los que una determinada función objetivo alcanza su valor máximo o mínimo*

1. Introducción: Optimización

- Problema de optimización (<u>maximización</u>):

✓ Dado un dominio X y una función objetivo

$$f(x): x \in X \rightarrow R$$

✓ El objetivo es encontrar x^* que verifique

$$x^* \in X: f(x^*) \geq f(x), \forall x \in X$$

- Optimización combinatoria $\rightarrow$ Variable discreta

"Un Problema de Optimización Combinatoria consiste en encontrar un objeto entre un conjunto finito (o al menos contable) de posibilidades"

1. Introducción: Optimización

- Tipos de problemas de optimización (representación de una solución):

✓ Permutaciones (Problemas de ordenación)

✓ Binarios (Problemas de pertenencia)

✓ Enteros (Problemas de cardinalidad, asignación, selección)

✓ De optimización numérica (Optimización de funciones no lineales)

1. Introducción: Optimización

- Problemas de optimización fáciles de resolver:

✓ *Lineales:* función objetivo y restricciones lineales (método *Simplex*)

- Problemas de optimización difíciles de resolver (NP-duros):

✓ No se puede garantizar el encontrar la **mejor** solución posible en un **tiempo** razonable

✓ Mayoría de los problemas con **aplicación** práctica, científica o industrial

✓ Desarrollo de procedimientos **eficientes** para encontrar buenas soluciones aunque sean **no óptimas**

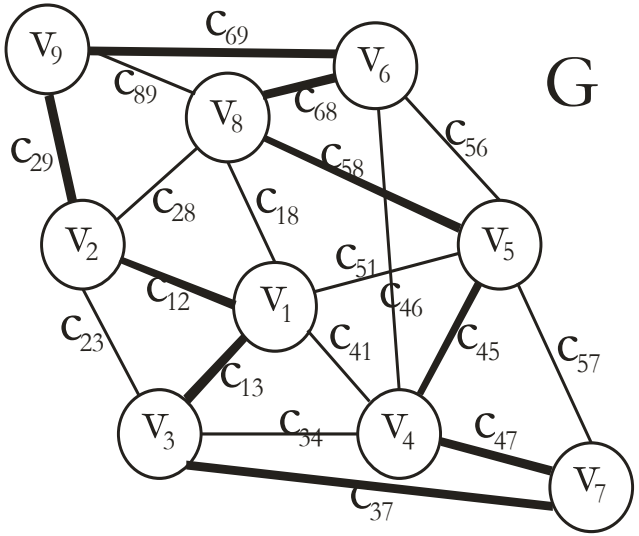
2. Problemas de Optimización Combinatoria Clásicos

- Viajante
- Mochila
- Coloreado de Grafos
- Partición de Conjuntos
- Asignación Cuadrática
- Asignación Generalizada
- Ordenación Lineal
- Diversidad
- Enrutamiento de vehículos
- …

Viajante de Comercio

- **Problema del Viajante de comercio, TSP**:

  Un viajante de comercio ha de visitar n ciudades, comenzando y finalizando en su propia ciudad. Conociendo el coste de ir de cada ciudad a otra, determinar el recorrido de coste mínimo



$$TSP = \left\{ \begin{array}{ll} min: & c_{\pi(1)\pi(n)} + \sum_{i=1}^{n-1} c_{\pi(i)\pi(i+1)} \end{array} \right.$$

Problema de la Mochila

- **Problema de la mochila, *Knapsack Problem***:

  Dados n objetos, cada uno con un peso wj y un valor vj, se debe seleccionar el conjunto de objetos cuyo valor total sea máximo, sin exceder un peso máximo W

$$KP = \begin{cases} max: & \sum_{i=1}^{n} v_i x_i \\ s.a., & \\ & \sum_{i=1}^{n} w_i x_i \leq W \end{cases}$$

Problema de la Asignación Cuadrática

- **Problema de la asignación cuadrática, *QAP***:

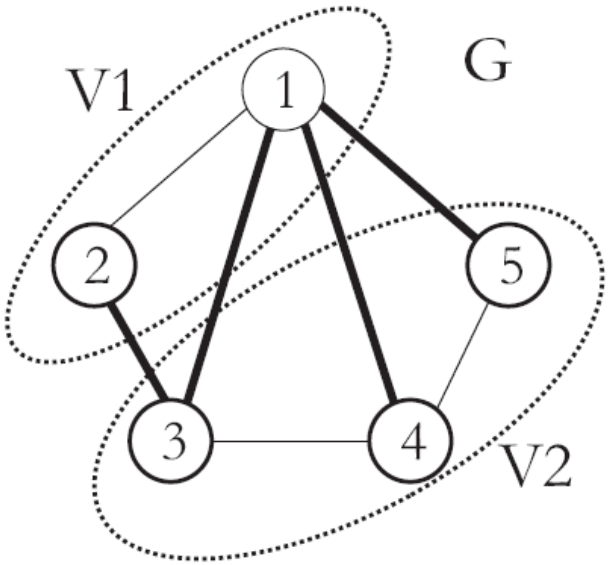
  Dadas n unidades y n localizaciones posibles, el problema consiste en determinar la asignación óptima de las unidades en las localizaciones conociendo el flujo existente entre las primeras y la distancia entre las segundas

$$QAP = \min_{S \in \Pi_N} \left(\sum_{i=1}^{n} \sum_{j=1}^{n} f_{ij} \cdot d_{S(i)S(j)} \right)$$

Problema del Corte Máximo

- **Problema del corte máximo, *Max-Cut Problem***:

  Dado un grafo ponderado y no dirigido y un conjunto de pesos asociados a cada arista, encontrar una partición del conjunto de vértices en dos subconjuntos disjuntos, tal que maximice la suma de los pesos en aristas cuyos extremos se encuentran en subconjuntos distintos

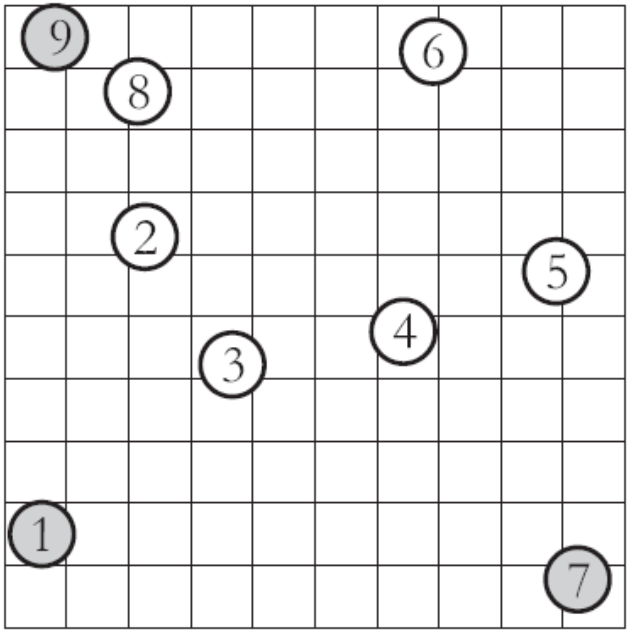


$$MCP = \begin{cases} max: & \frac{1}{2} \sum_{1 \le i \le j \le n} w_{ij}(1 - y_i y_j) \\ s.a., & \\ & y_i \in \{-1, 1\}, \quad \forall i \in V \end{cases}$$

Problema de la Diversidad Máxima

- **Problema de la diversidad máxima, MDP**:

  Seleccionar un conjunto de elementos m de una colección más grande n de tal forma que los elementos seleccionados tengan las características más variadas entre sí

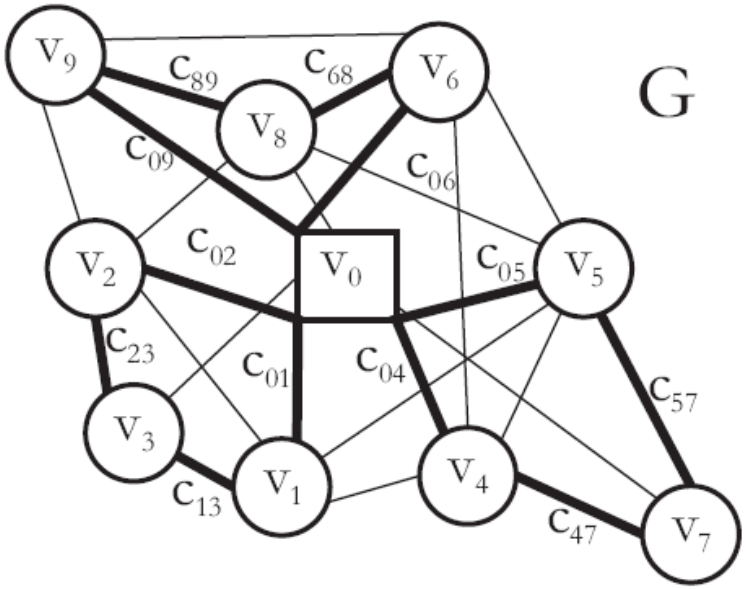


$$MDP = \begin{cases} max: & \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} d_{ij} x_i x_j \\ s.a., & \\ & \sum_{i=1}^{n} x_i = m \\ & x_i = \{0,1\} \mid 1 \leq i \leq n \end{cases}$$

Enrutamiento de Vehículos

- **Problema del enrutamiento de vehículos, VRP**:

  Obtener el conjunto de rutas más cortas posibles utilizando un conjunto de vehículos (con capacidad limitada) lo más pequeño posible tal que, partiendo de un almacén y regresando sucesivamente a él, abastecen a una serie de clientes (con demanda diferente) distribuidos geográficamente

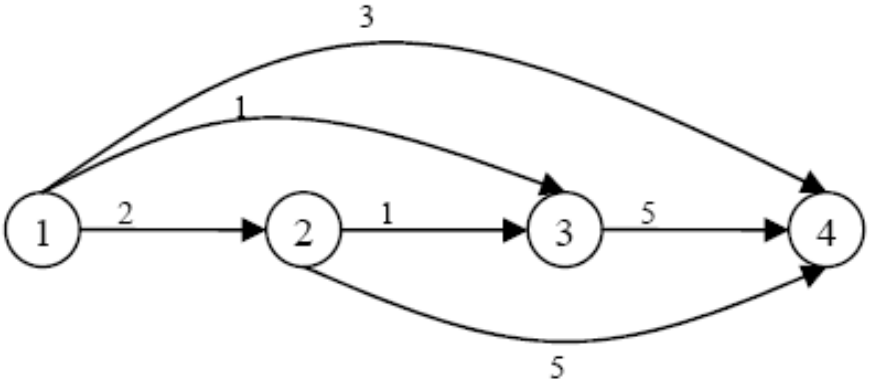


$$VRP = \begin{cases} min: & \sum_{k=1}^{m} \sum_{i=1}^{n_k} c_{\pi^k(i)\pi^k(i+1)} \\ s.a., & \\ & \sum_{i=1}^{n_k} q_{\pi^k(i)} \leq Q_k \ \forall k \in [1,m] \\ & \sum_{k=1}^{m} n_k \leq n \end{cases}$$

Ordenación Lineal

- **Problema de la ordenación lineal, LOP**:

  Dada una matriz de pesos de tamaño n×n, encontrar una permutación de las columnas y filas (simultáneamente) que maximice la suma de los pesos en el triángulo superior



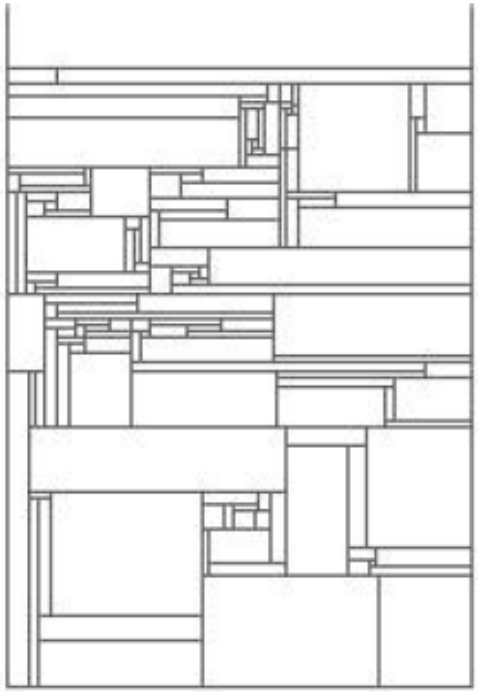
$$C = \begin{pmatrix} 0 & 2 & 1 & 3 \\ 4 & 0 & 1 & 5 \\ 2 & 3 & 0 & 5 \\ 1 & 2 & 1 & 0 \end{pmatrix}$$

$$LOP = \left\{ \; max: \quad \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} e_{p(i)p(j)} \right.$$

Empaquetado en Cinta

- **Problema del empaquetado en cinta, SPP:**

  Dado un conjunto de rectángulos de diferentes dimensiones se colocan sobre una cinta de anchura fija y altura indefinida de tal forma que se minimice la altura alcanzada por dicha colocación



Otros problemas

- **Problema del cliqué máximo**: Encontrar la mayor componente conexa de un grafo dado
- **Problema del coloreado de grafos**: Encontrar la mínima cantidad de colores tal que dos vértices adyacentes no pueden tener el mismo color
- **Problema del árbol de Steiner**: Encontrar un árbol de coste mínimo que conecte un conjunto de vértices dado
- **Problemas de asignación**: Dada una tabla de tareas y personas que pueden realizarlas (coste distinto), encontrar la asignación de coste mínimo

3. Algunos Ejemplos de Problemas de Optimización Reales

- Diseño Aeronáutico
- Planificación de Rutas para Transporte de Mercancías
- Equilibrado de Líneas de Montaje en Nissan
- Identificación Forense de Personas Desaparecidas

Diseño Aeronáutico

- En los últimos años, se ha avanzado mucho en el **diseño óptimo aerodinámico**

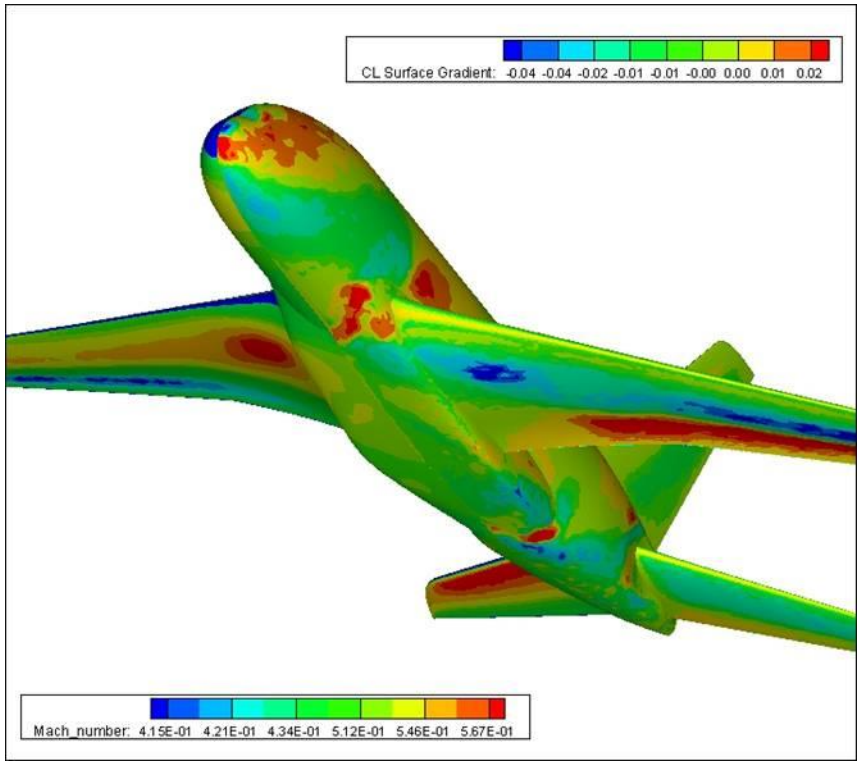
- Esta área abarca desde elementos pequeños del campo del deporte, como los cascos de los ciclistas o los monos de los motoristas, a otros de un tamaño mucho mayor como **automóviles**, **aviones** o **barcos**

- En todos los casos no se trata sólo de una cuestión estética, sino de aerodinámica, con objeto de **optimizar el rendimiento** de los vehículos o los deportistas

- Hoy en día, **casi todo el proceso de optimización se realiza mediante el empleo del ordenador**

Diseño Aeronáutico

- En diseño aeronáutico, la **optimización** o **diseño óptimo** se refiere a la investigación para determinar la forma externa de la aeronave que resulte aerodinámicamente más eficiente, dentro de unas determinadas restricciones estructurales



Diseño Aeronáutico

- La **disminución de la resistencia al avance de un avión** en el aire es el factor clave

- Se calcula resolviendo unas ecuaciones que simulan el comportamiento de un objeto sólido (el avión) en interacción con un fluido (el aire), según la **Dinámica Computacional de Fluidos**

- Después se usan **métodos de optimización** para obtener la forma óptima del avión que minimiza la resistencia verificando los requisitos geométricos y físicos

- Los diseños prometedores mediante la simulación computacional son validados en el **túnel de viento**

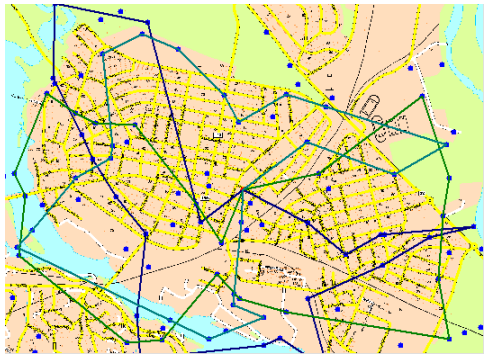
Diseño Aeronáutico

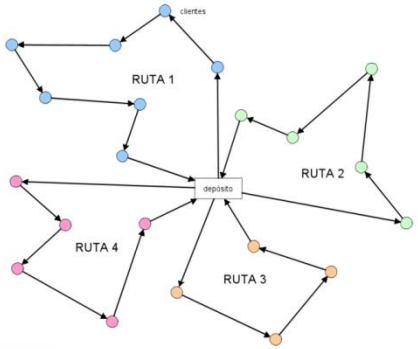
- El problema de los **algoritmos de optimización numérica clásicos** es su **lentitud**
- En casos reales, cada ejecución puede requerir meses de cómputo en máquinas de alto rendimiento al tener que realizar una gran cantidad de simulaciones
- Por ello, se han empleado los **algoritmos evolutivos** para esta tarea, que son capaces de proporcionar **diseños de buena calidad en un tiempo mucho más reducido**
- Además, como optimizadores multiobjetivo, pueden optimizar varios criterios a la vez (velocidad, estabilidad o gasto de combustible, por ejemplo)

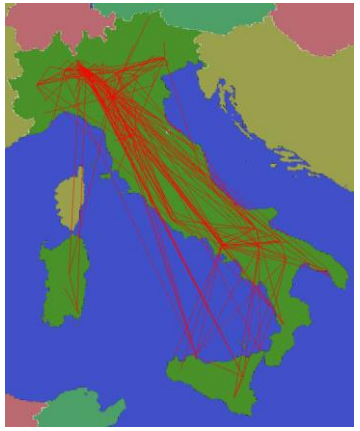
Planificación de Rutas para Transporte de Mercancías

- Hoy en día es difícil encontrar empresas que gestionen las **operaciones de logística** sin la ayuda del ordenador

- El problema típico es **diseñar las rutas más adecuadas de transporte/recogida de productos** entre un almacén central y unos destinos dispersos geográficamente







- Su resolución de forma adecuada puede suponer **ahorros muy significativos para la empresa**

Planificación de Rutas para Transporte de Mercancías

- Esta tarea se lleva a cabo empleando una **flota de vehículos** pertenecientes o no a la empresa

- Un **sistema de planificación de vehículos** debe proporcionar un conjunto de **rutas de reparto** a los conductores

- Las mercancías deben ser entregadas cuándo y donde se requieran, con el mínimo coste posible y verificando todas las restricciones legales y políticas de la empresa

- **Los algoritmos de hormigas (AntRoute) son una herramienta muy potente para la planificación de rutas**

Planificación de Rutas para Transporte de Mercancías

- **AntRoute** planifica diariamente las rutas de reparto desde el almacén central de **Migros**, una gran cadena suiza con 600 supermercados, localizado en **Suhr** (AG), **a toda Suiza**

- Migros dispone de una **flota de entre 150 y 200 vehículos** con tres tamaños: camiones (capacidad de 17 palés), trailers (35 palés) y unidades tractoras (33 palés)

- Esto provoca restricciones de acceso a los almacenes de los supermercados, restricciones de uso de ciertas carreteras, …

- Los repartos tienen de realizarse a horas específicas, todos ellos en un solo día (**productos perecederos**) y el último tiene que hacerse lo más lejos posible del almacén (**servicios extra**)

Planificación de Rutas para Transporte de Mercancías

- Por ejemplo, en un reparto de 52000 palés a 6800 clientes en un periodo de 20 días, AntRoute obtuvo el diseño diario de rutas en menos de 5 minutos en un PC estándar

- Los expertos de la empresa necesitaron tres horas…

- Las soluciones de AntRoute fueron de mucha mejor calidad en cuanto al número de rutas necesario, la distancia total recorrida y al aprovechamiento de los vehículos:

	Human Planner	AR-RegTW	AR-Free
Total number of tours	2056	1807	1614
Total km	147271	143983	126258
Average truck loading	76.91%	87.35%	97.81%

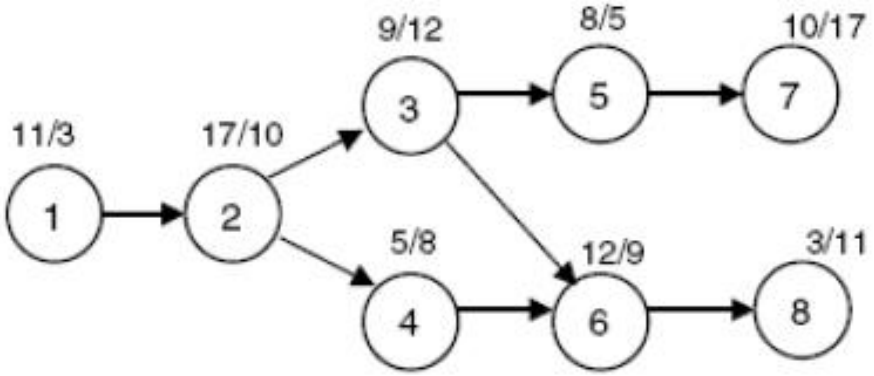
Equilibrado de Líneas de Montaje en Nissan

- La mayoría de los sistemas productivos actuales se basan en líneas de montaje
- La producción de un ítem se divide en un **conjunto de tareas** que tienen que llevarse a cabo según un orden concreto y respetando una serie de precedencias
- Cada tarea necesita un tiempo dado (más un área de trabajo) y tiene asociada un conjunto de predecesores directos
- El diseño (**equilibrado**) de la línea requiere **agrupar de forma eficiente las tareas necesarias en estaciones de trabajo** para maximizar la producción y reducir tiempos muertos

Equilibrado de Líneas de Montaje en Nissan

- El parámetro clave es el tiempo de ciclo que indica el máximo tiempo permitido para que una estación procese sus tareas. **A menor tiempo de ciclo, mayor capacidad productiva de la línea**

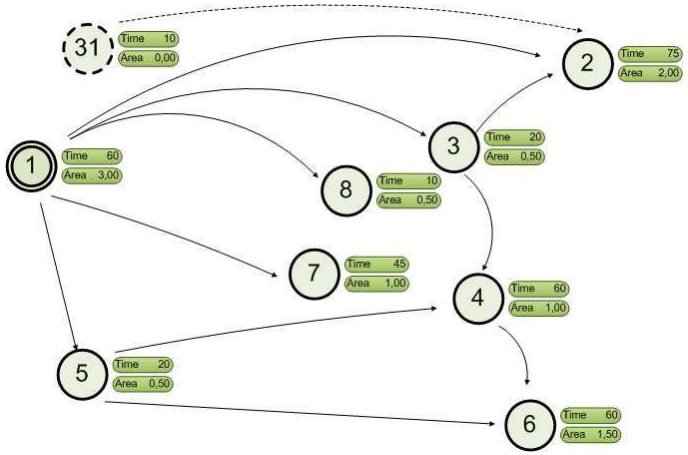
- Los objetivos del equilibrado son:
  - agrupar las tareas en el menor número posible de estaciones de trabajo satisfaciendo un tiempo de ciclo, u
  - obtener la agrupación que minimiza el tiempo de ciclo



Equilibrado de Líneas de Montaje en Nissan

- Los algoritmos de OCH se han aplicado con gran éxito al equilibrado de líneas de montaje, resolviendo problemas cada vez más complejos y realistas

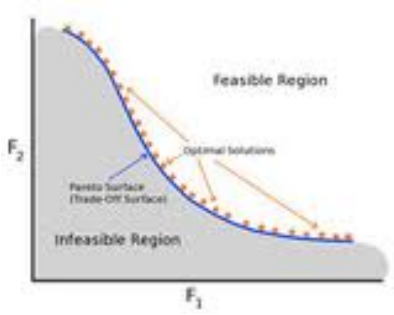
- Trabajamos con la **Cátedra Nissan de la UPC** para resolver el problema **multiobjetivo** de minimizar el número de estaciones y su área para un tiempo de ciclo dado en la línea de montaje del **motor del Nissan Pathfinder**:

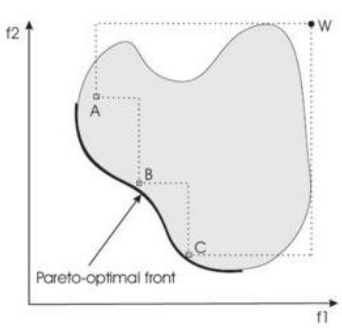


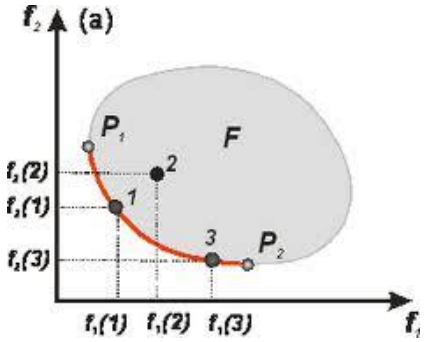
Equilibrado de Líneas de Montaje en Nissan

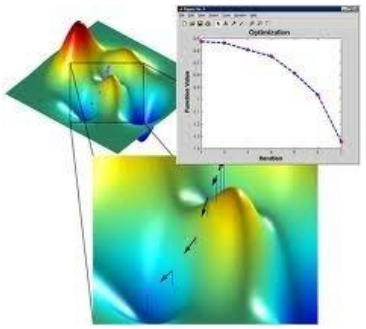
- Es un **problema multiobjetivo** con **muchas restricciones fuertes y un espacio de búsqueda de gran dimensión**: oportunidad para metaheurísticas como los algoritmos de OCH

- El objetivo es proporcionar al ingeniero de planta con diversas opciones de diseño optimales con distinto equilibrio entre ambos objetivos en una sola ejecución del algoritmo



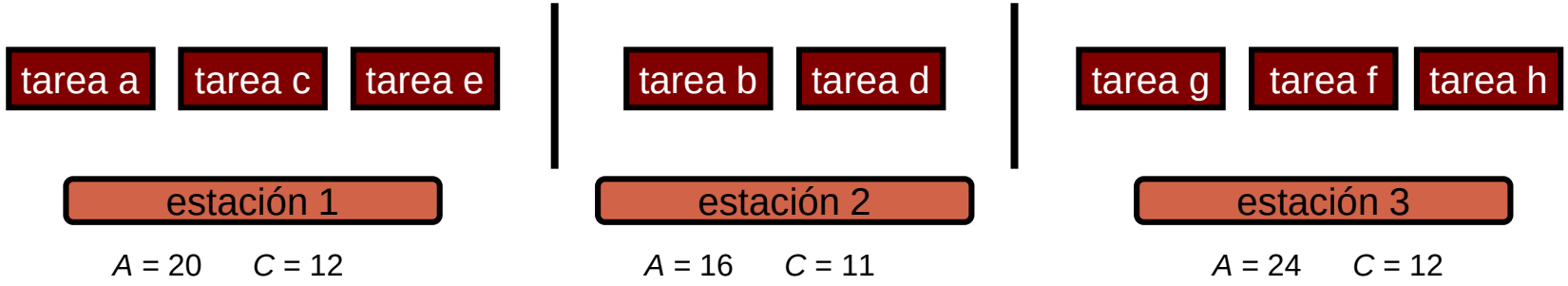






Equilibrado de Líneas de Montaje en Nissan

- Una solución a este problema (**TSALBP**) es una asociación de tareas a las distintas estaciones que cumpla las restricciones

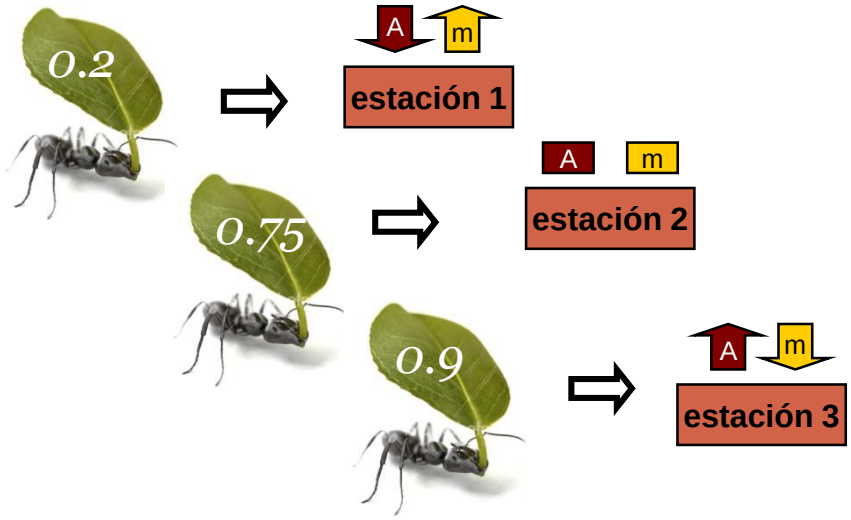


- Hemos diseñado un **algoritmo de Optimización de Colonias de Hormigas multiobjetivo** que proporciona varias soluciones con un equilibrio distinto entre el área y el número de estaciones al ingeniero de la planta

- El rastro de feromona se asocia al par (tarea, estación)

Equilibrado de Líneas de Montaje en Nissan

- Introducimos una filosofía multicolonia para obtener un mayor abanico de soluciones posibles: **cada hormiga utiliza distintos umbrales de llenado de estación**

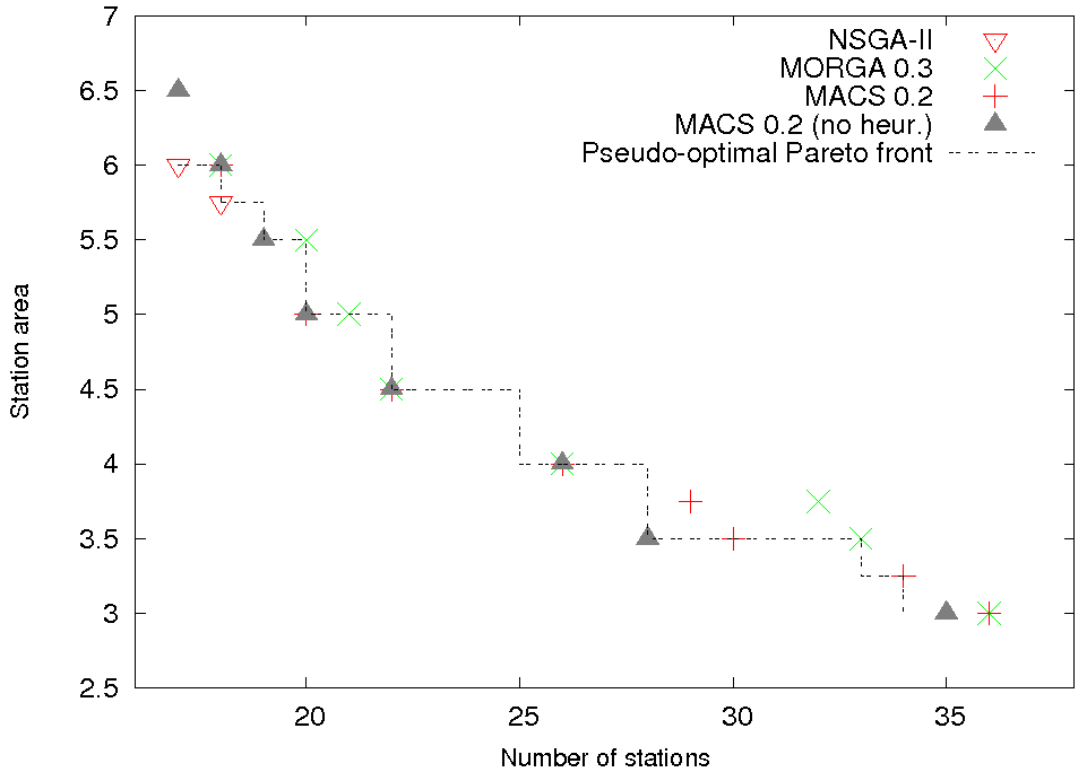


- Nuestra propuesta obtiene muy buenos resultados. El algoritmo de hormigas mejora a otras técnicas de búsqueda

Equilibrado de Líneas de Montaje en Nissan

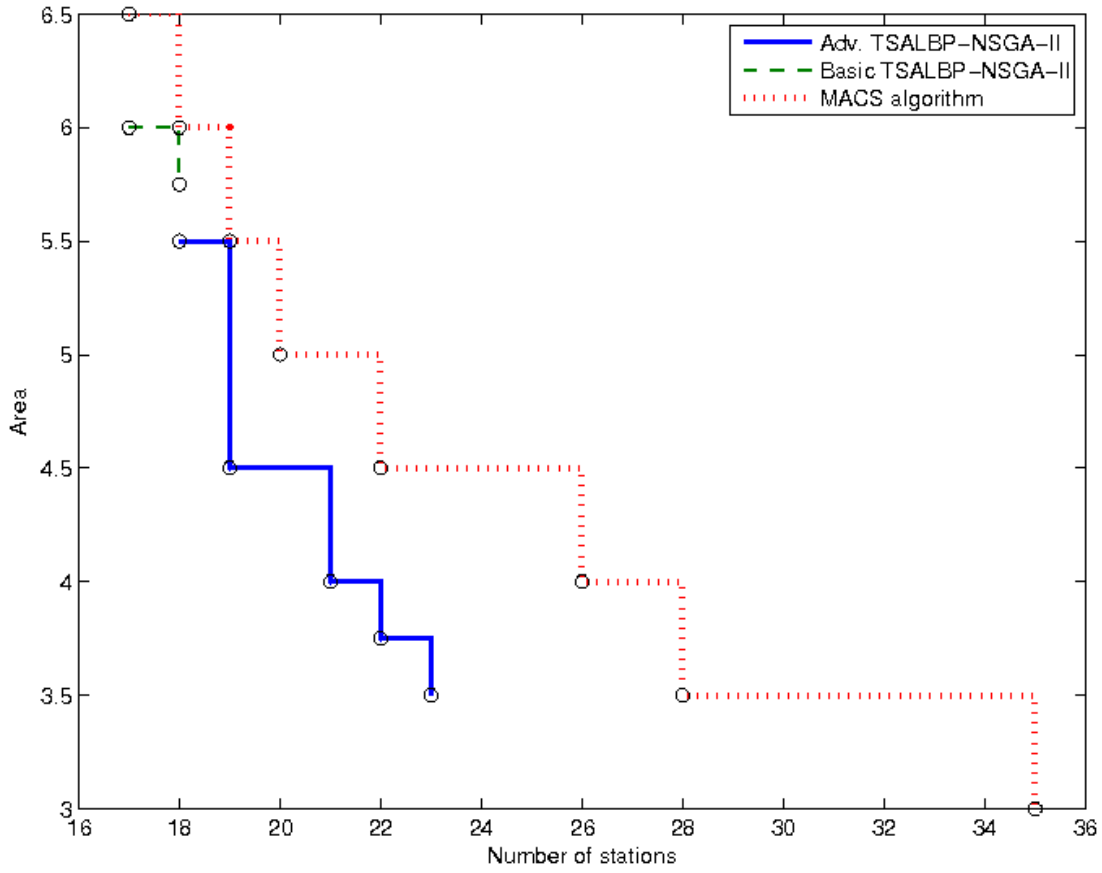
Motor del Nissan Pathfinder:

- 747 piezas y 330 referencias en 6 versiones del motor diesel
- 378 operaciones de montaje (prueba rápida incluida) agrupadas en 140
- 79 operarios para un turno de 301 motores



Equilibrado de Líneas de Montaje en Nissan

Recientemente, hemos diseñado un algoritmo genético multiobjetivo específico para el TSALBP que mejora los resultados del algoritmo de OCH:



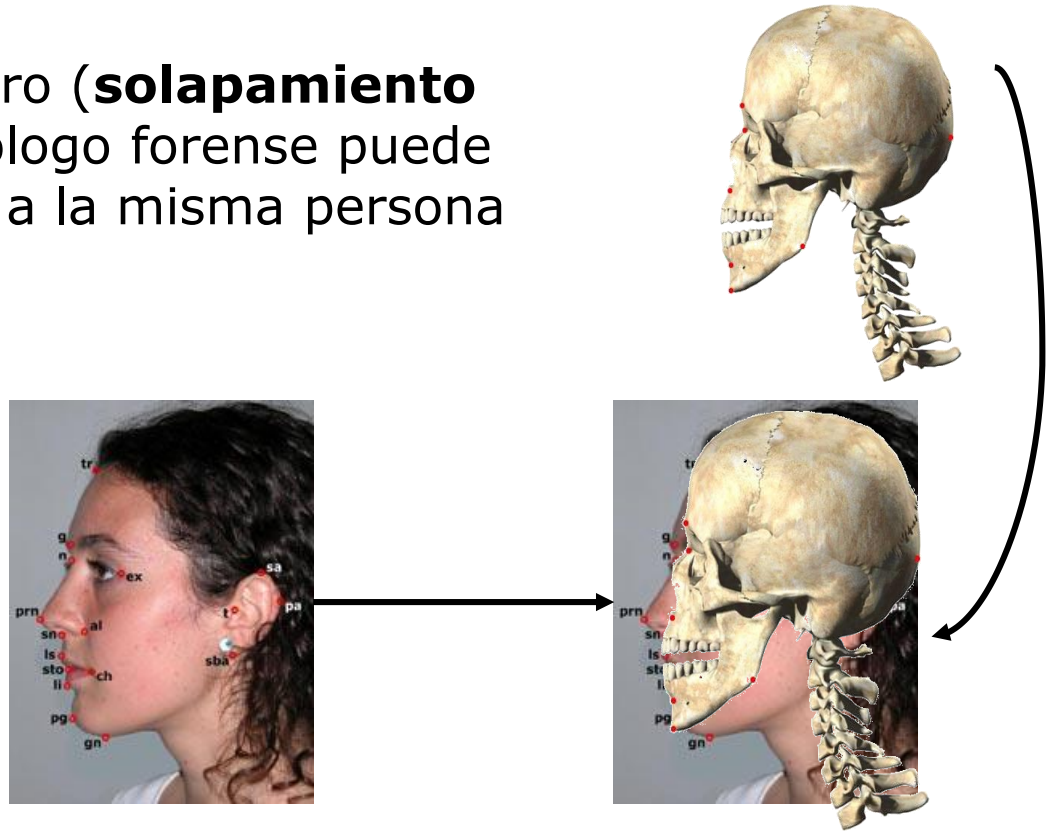
Identificación Forense de Personas Desaparecidas

CSI: crime scene investigation

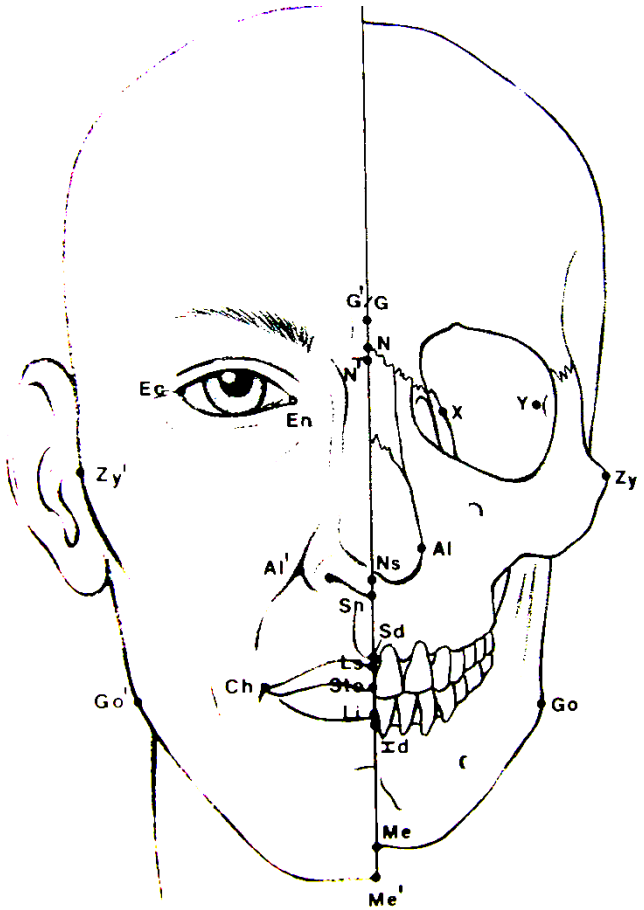
Identificación Forense de Personas Desaparecidas

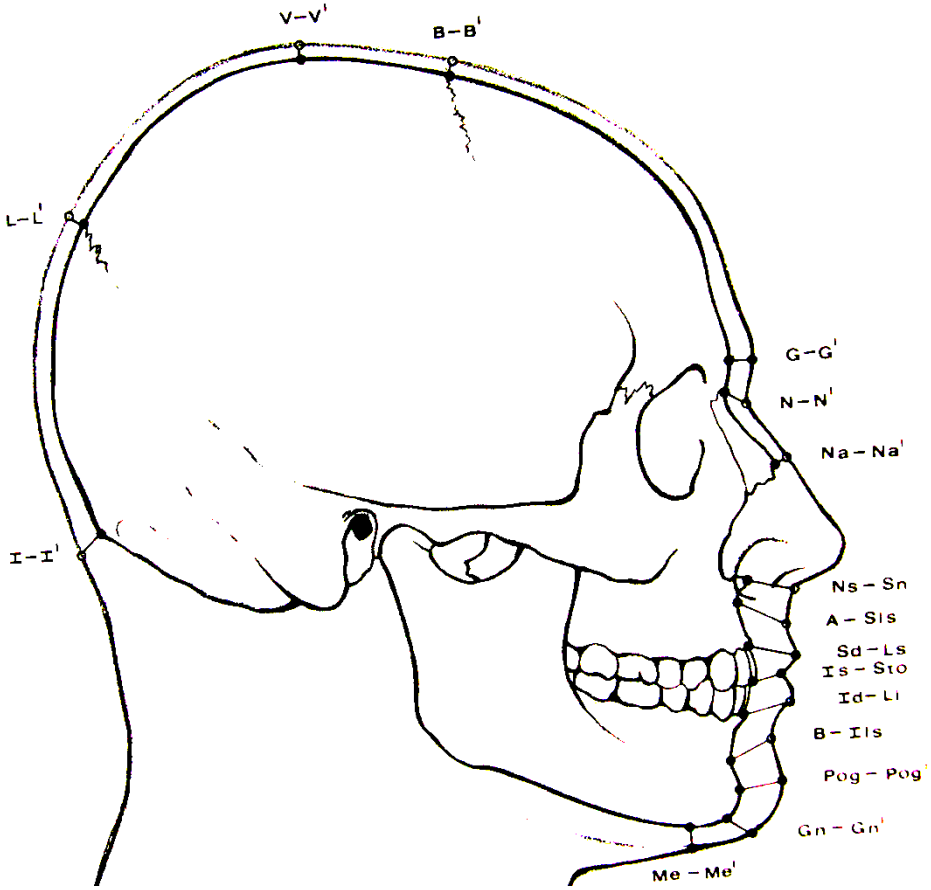
- **La superposición craniofacial** es una técnica de identificación forense basada en la comparación de un "modelo" del cráneo encontrado y una foto de una persona desaparecida

- Proyectando uno sobre otro (**solapamiento cráneo-cara**), el antropólogo forense puede determinar si pertenecen a la misma persona



Identificación Forense de Personas Desaparecidas





Correlación entre los puntos craneométricos y cefalométricos

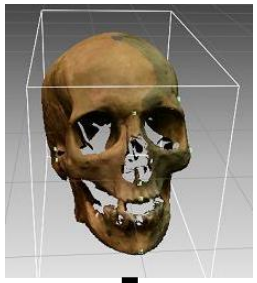
Identificación Forense de Personas Desaparecidas

- Diseño de un procedimiento automático basado en el ordenador para automatizar el proceso de identificación forense por superposición craneofacial:
  - Diseño de métodos automáticos de registrado de imágenes de rango para obtener **modelos 3D** de cráneos en cualquier condición (mediante algoritmos evolutivos, AEs)
  - Diseño de métodos de registrado 3D/2D automáticos para el **solapamiento cráneo-cara** (mediante AEs y conjuntos fuzzy)
- Proyectos Plan Nacional I+D+I (2006-09, 2009-12, 2013-15, 2016-18) y Excelencia Junta de Andalucía (2007-10, 2013-18). **Patente internacional Febrero 11. Proyecto Europeo FP7-Security MEPROCS (2012-14). Premios Internacionales**

Identificación Forense de Personas Desaparecidas

1. Modelado del cráneo y mejora facial

Identificación {Positiva/negativa/ probable positiva/probable negativa/ indeterminada}



Reconstrucción del modelo 3D

Procesado de imagen y localización de landmarks

2. Solapamiento cráneo-cara

RI 3D-2D: traslación, rotación, escalado, y proyección 2D

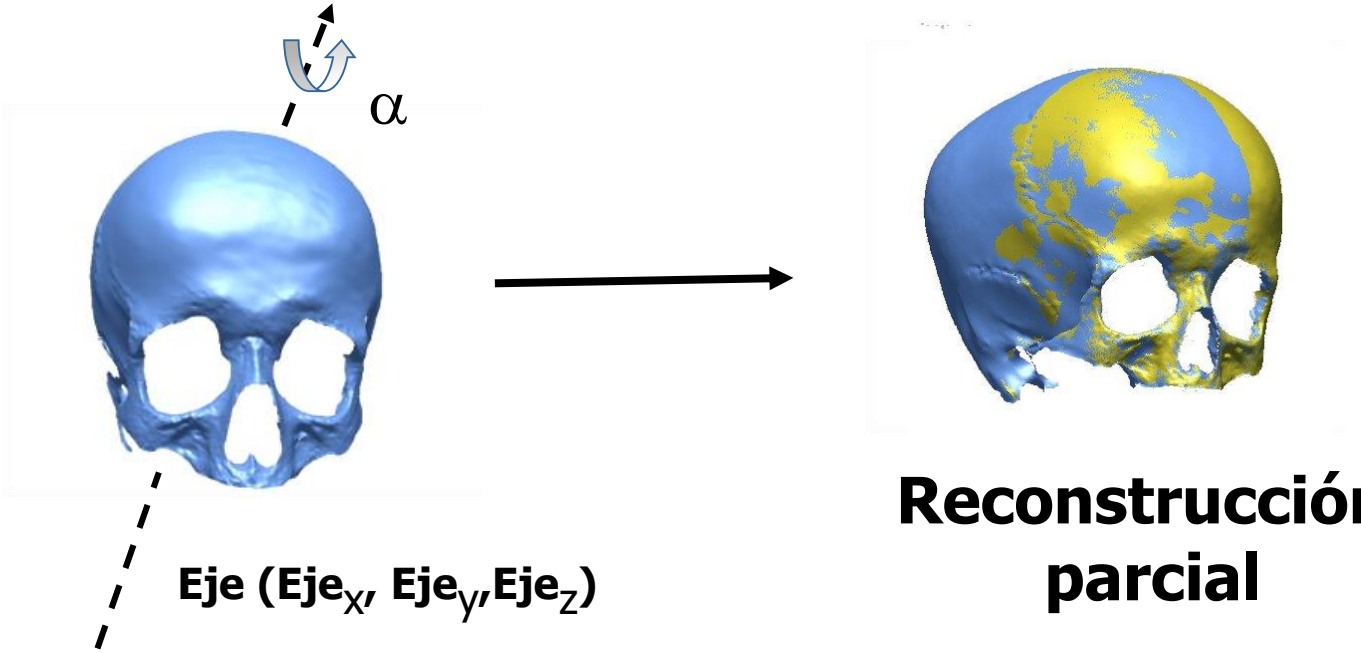
3. Toma de decisiones

Identificación Forense de Personas Desaparecidas

Escáner de rango

- Algoritmos Meméticos con codificación real para el modelado 3D de cráneos. Representación de una solución a este problema:

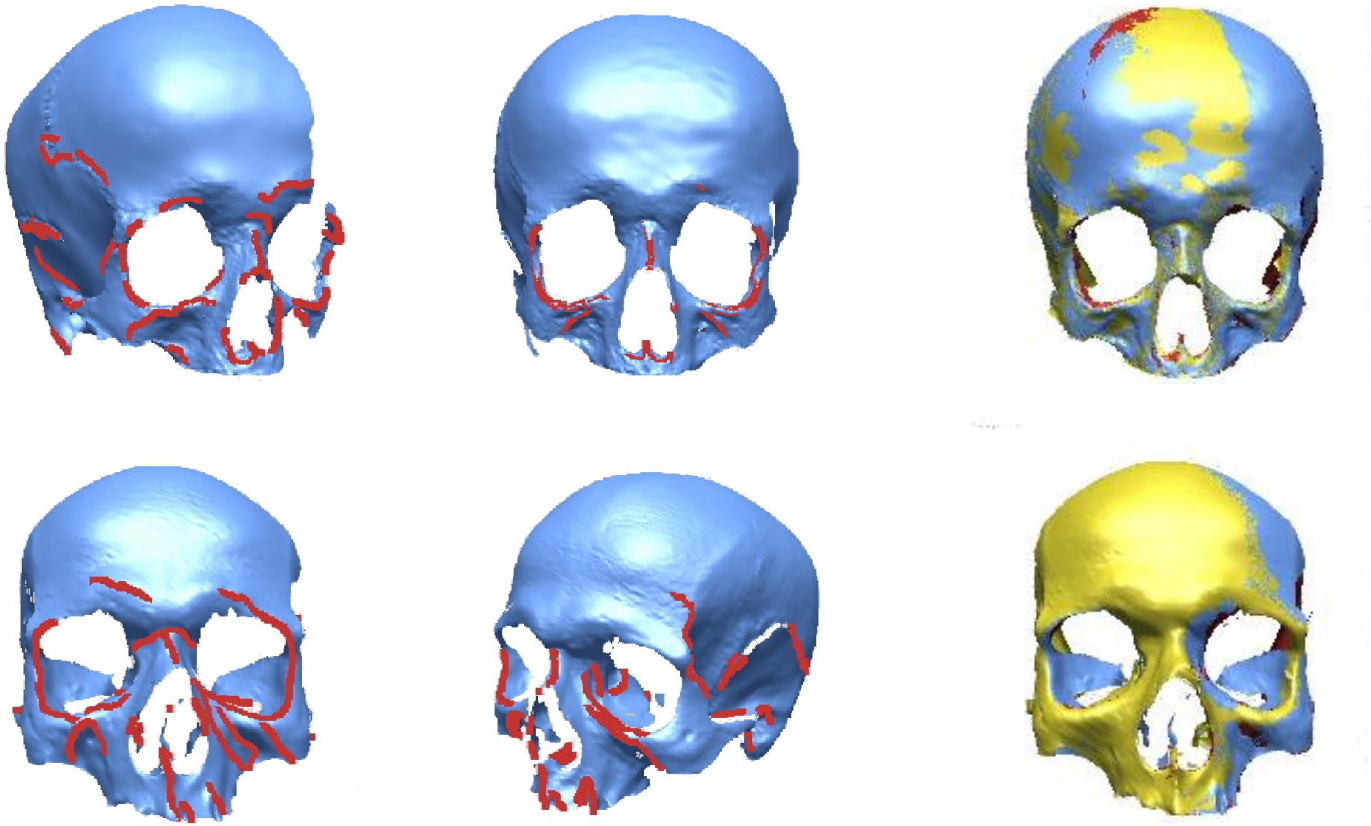
Rotación **Traslación**



Identificación Forense de Personas Desaparecidas

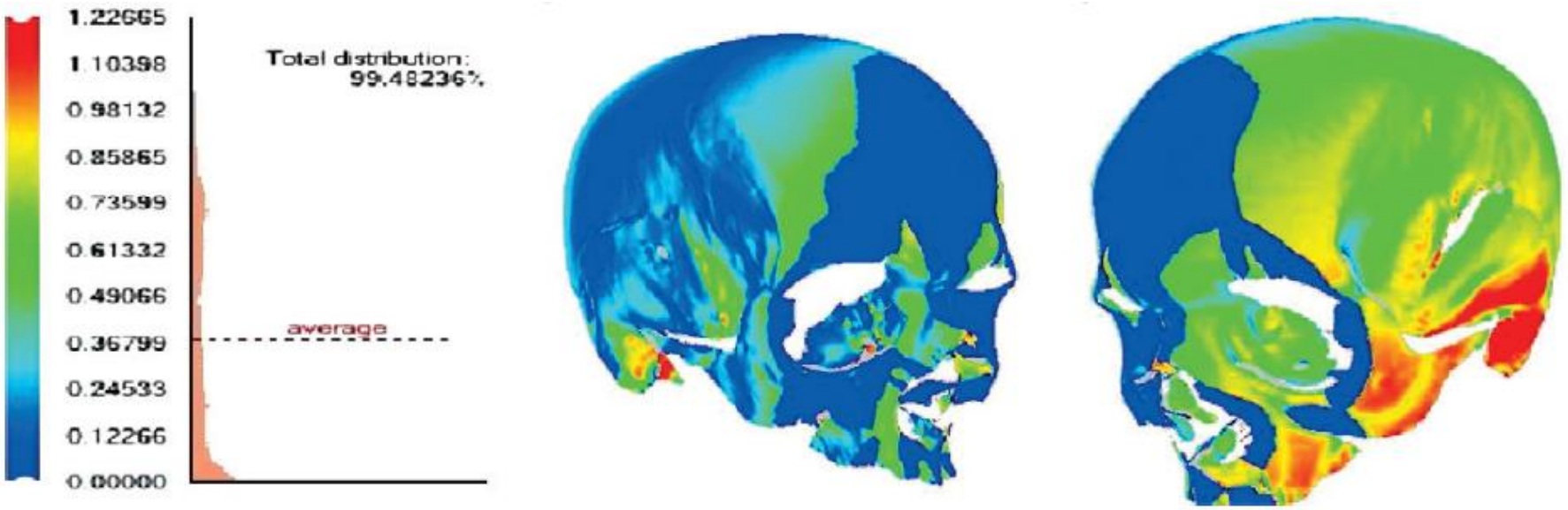
Entrada: vistas 3D

Reconstrucción



Identificación Forense de Personas Desaparecidas

- Error global del modelo 3D: menor de 1 milímetro
- Tiempo de reconstrucción 3D: 2 minutos
- Robustez del método: baja desviación típica en 30 ejecuciones distintas



Búsqueda de la mejor superposición 3D-2D (Algoritmo Evolutivo con Codificación Real)

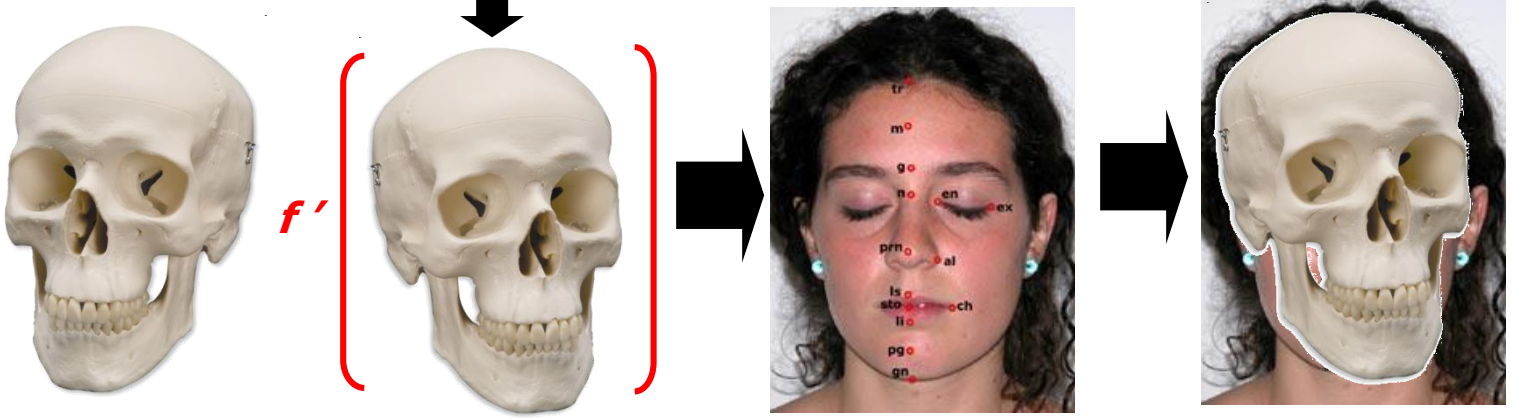
Error de Registrado

$f' \cong f^*$

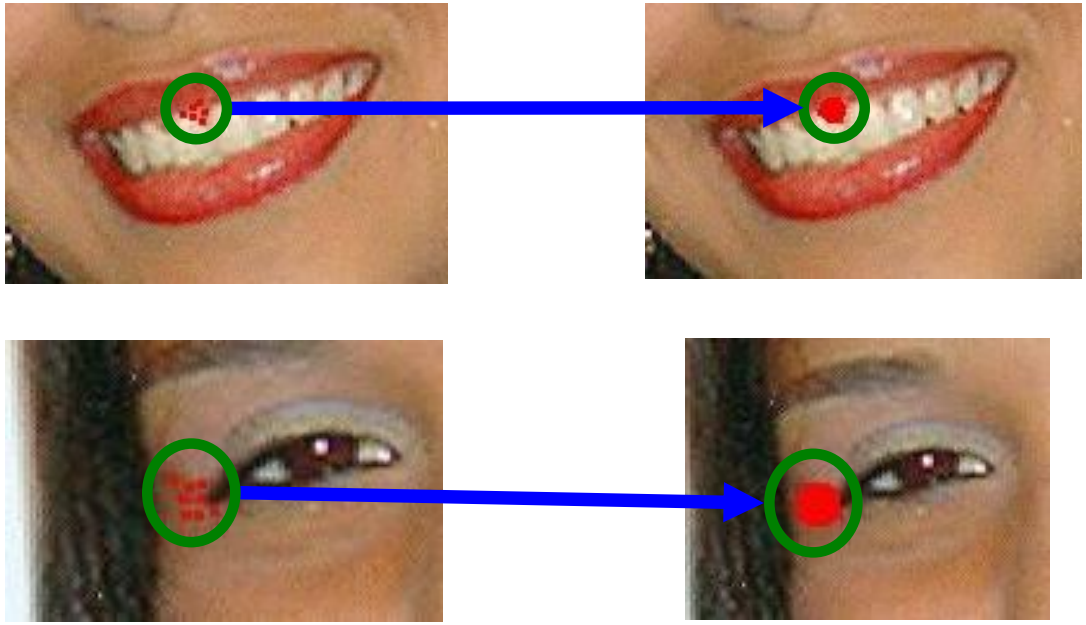
Evaluación f '

Rotación = {60°,(0,1,0)}
Traslación = {2, 0, 1}…

Medir la distancia entre cada par de puntos de referencia



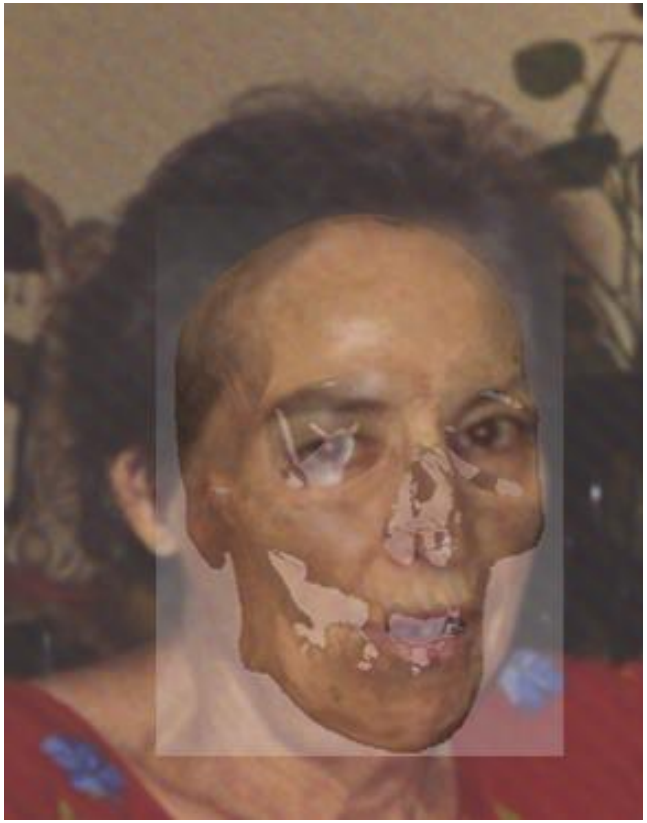
- **Landmarks imprecisos**. Cada punto cefalométrico es un área (elíptica o circular). Mayor incertidumbre asociada al landmark → mayor área



- Cada landmark cefalométrico es un **punto fuzzy definido por un conjunto fuzzy bidimensional**. A mayor incertidumbre sobre la posición del landmark → mayor será la región fuzzy

Identificación Forense de Personas Desaparecidas

Manual



Area deviation error: 34.70%

varias horas

Fuzzy AE

Area deviation error: 13.23%

2-4 minutos

4. Software de Metaheurísticas

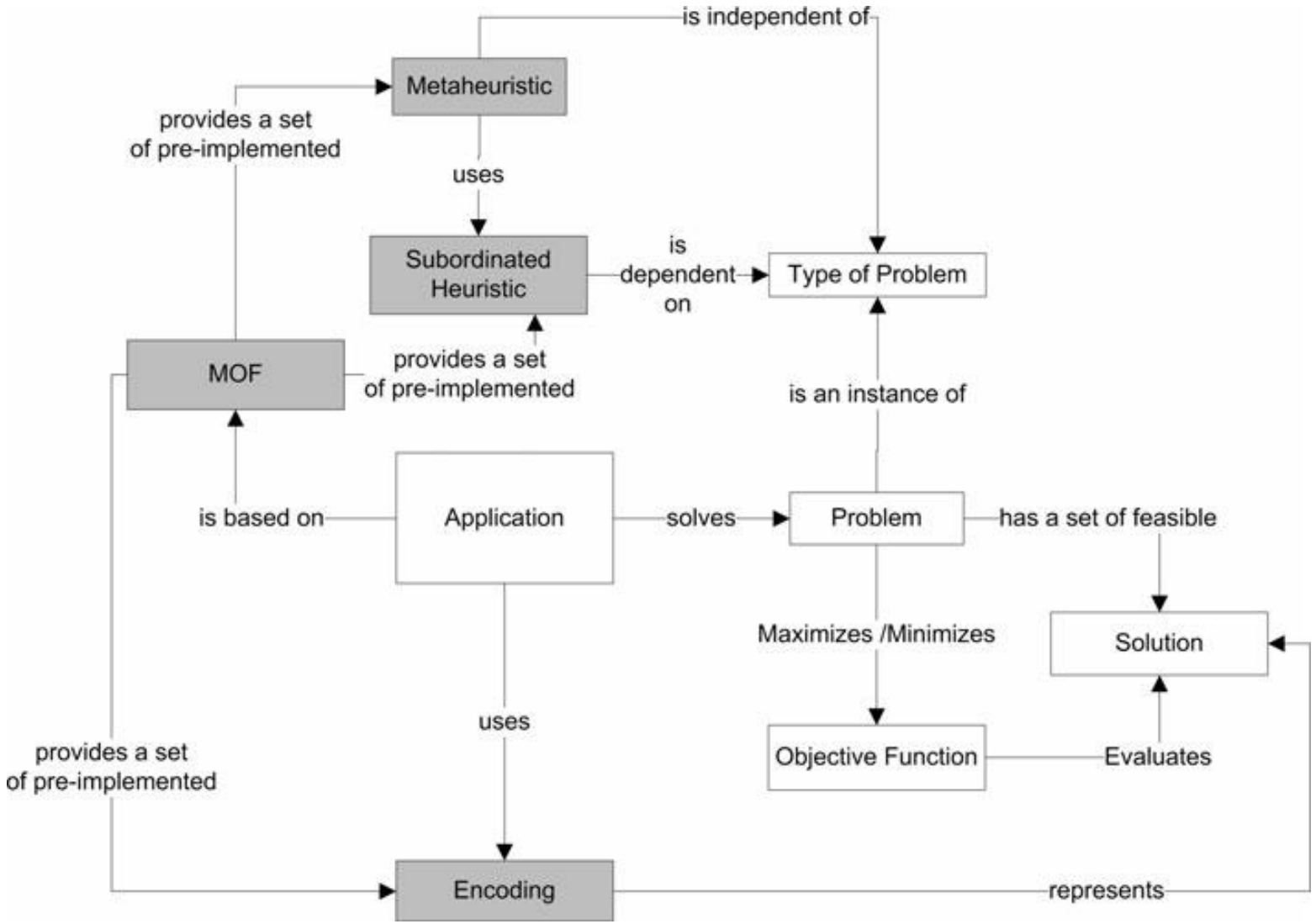
Metaheuristic optimization frameworks: a survey and benchmarking

José Antonio Parejo · Antonio Ruiz-Cortés · Sebastián Lozano · Pablo Fernandez

Soft Comput (2012) 16:527–561
DOI 10.1007/s00500-011-0754-8

Abstract This paper performs an unprecedented comparative study of Metaheuristic optimization frameworks. As criteria for comparison a set of 271 features grouped in 30 characteristics and 6 areas has been selected. These features include the different metaheuristic techniques covered, mechanisms for solution encoding, constraint handling, neighborhood specification, hybridization, parallel and distributed computation, software engineering best practices, documentation and user interface, etc. A metric has been defined for each feature so that the scores obtained by a framework are averaged within each group of features, leading to a final average score for each framework. Out of 33 frameworks ten have been selected from the literature using well-defined filtering criteria, and the results of the comparison are analyzed with the aim of identifying improvement areas and gaps in specific frameworks and the whole set. Generally speaking, a significant lack of support has been found for hyper-heuristics, and parallel and distributed computing capabilities. It is also desirable to have a wider implementation of some Software Engineering best practices. Finally, a wider support for some metaheuristics and hybridization capabilities is needed.

4. Software de Metaheurísticas



4. Software de Metaheurísticas

Table 2 Selected MOFs

Name	Web
EasyLocal (Di Gaspero and Schaerf 2003)	http://satt.diegm.uniud.it/EasyLocal++/
ECJ (Luke et al. 2009)	http://cs.gmu.edu/~eclab/projects/ecj/
EO/ ParadisEO/ MOEO/ PEO (Cahon et al. 2004)	http://paradiseo.gforge.inria.fr http://eodev.sourceforge.net/
EvA2 (Kronfeld et al. 2010)	http://www.ra.cs.uni-tuebingen.de/software/EvA2/
FOM (Parejo et al. 2003)	http://www.isa.us.es/fom
HeuristicLab (Wagner 2009)	http://dev.heuristiclab.com
JCLEC (and KEEL) (Ventura et al. 2008)	http://JCLEC.sourceforge.net http://sci2s.ugr.es/keel/
MALLBA (Alba et al. 2007)	http://neo.lcc.uma.es/mallba/easy-mallba/index.html
Optimization Algorithm Toolkit (Brownlee 2007)	http://optalgtoolkit.sourceforge.net
Opt4j (Martin Lukasiewycz and Helwig 2009)	http://opt4j.sourceforge.net

- Todos están implementados en Java o C++, con código documentado, permiten la generación de código ejecutable fuera del framework e incluyen al menos dos tipos de metaheurísticas distintas

4. Software de Metaheurísticas

Table 8 MOFs Programming languages, platforms and licenses

MOF	Prog. Lang.	Platforms	License
EasyLocal	C++	Unix	GPL
ECJ	Java	All	Open Source (Academic free license)
ParadisEO	C++	All (Except for windows if using PEO)	CECILL (ParadisEO) and LGPL (EO)
EvA2	Java	All	LGPL
FOM	Java	All	GPL
HeuristicLab	C#	Windows	GPL
JCLEC (and KEEL)	Java	All	LGPL
MALLBA	C++	Unix	Open source
Optimization Algorithm Toolkit	Java	All	LGPL
Opt4j Martin Lukasiewycz and Helwig (2009)	Java	All	LGPL

Characteristic	Feature	Weight	ECJ	ParadisEO	EvA2	FOM	JCLEC	OAT	Opt4j	EasyLocal	HeuristicLab	MALLBA	Sum
SD/HC	Basic Implementation	0.5	✔	✔	✔	✔		✔		✔	✔	✔	8
	Multi-Start	0.5	✔	✔	✔	✔		✔		~	✔	✔	7.5
SA	Basic Impl.	0.5		✔	✔	✔			✔	✔	✔	✔	7
	Lineal Annealing	0.1		✔		✔					✔		3
	Exponential/Geometric Annealing	0.1		✔		✔				✔	✔	✔	5
	Logaritmic Annealing	0.1				✔							1
	Metropolic Acceptance	0.1		✔	✔	✔			✔	✔	✔	✔	7
	Logistic Acceptance	0.1											0
TS	Basic Impl.	0.3		✔		✔				✔	✔		4
	Recent Features/Moves Based Tabu Memory	0.2		✔		~				✔	✔		3.5
	Frecuency Based Tabu Memory	0.3				✔				✔			2
	Basic Aspiration Criteria	0.2		✔		✔				✔	✔		4
GRASP		1				✔							1
VNS	Basic VNS (VNS)	0.2		✔		✔							2
	Variable Neighborhood Descent (VND)	0.2								✔			1
	Reduced VNS (RVNS)	0.2											0
	VNS with Decomposition	0.2											0
	Skewed VNS (SVNS)	0.2											0
EA	Basic EA Implementation of GA	0.2	✔	✔	✔	✔	✔	✔	✔		✔	✔	9
	Basic EA Implementation of ES	0.2	✔	✔	✔		✔	✔	✔		✔	✔	8
	Basic EA Implementation of GP	0.2	✔	✔	✔		✔	✔	✔		✔		7
	GAVaPS	0.05											0
	Diploid Individuals support	0.05				✔							1
	Coevolution support	0.1	✔										1
	Differential evolution	0.1	✔		✔			✔	✔				4
	Niching Methods	0.1	~		✔		✔				✔		3.5
PSO	Basic Implementation	0.3	✔	✔	✔				✔		✔	✔	6
	Discrete Variable Support	0.2		✔									1
	Customizable Dynamic Equations	0.2		✔	✔				✔				3
	Topologies	0.2		✔	✔						✔		3
	Lifetime support	0.1											0
AIS	CLONAG	0.25						✔					1
	optIA	0.25											0
	Immune Networks	0.25											0
	Detritic Cell Algorithms	0.25											0
ACS	AS	0.1				✔		✔				✔	3
	ACS	0.2				✔		✔				✔	3
	MMAS	0.4				✔		✔					2
	ASrank	0.2						✔					1
	API	0.1											0
Scatter Search	Basic. Impl.	1			✔								1

4. Software de Metaheurísticas

✓ **LOCAL++** (en C++) compuesta por una jerarquía de clases conteniendo *templates* de metaheurísticas de búsqueda, y que permiten especialización para abordar problemas específicos, combinaciones entre ellas o también la creación de nuevas estrategias

✓ **EasyLocal++** (http://tabu.diegm.uniud.it/EasyLocal++/) es una herramienta orientada a objetos (sucesora de LOCAL++) para desarrollar metaheurísticas de búsqueda local compuesta de clases, que implementa partes invariantes de las estrategias que se especializan mediante clases concretas con la parte dependiente del problema específico

✓ **ParadisEO** (http://paradiseo.gforge.inria.fr/) es otro framework orientado a objetos (C++) que genera código portable a Windows, Linux, Unix y MacOSX. Incorpora técnicas muy diversas como búsquedas locales, algoritmos evolutivos, particle swarm optimization, metaheurísticas paralelas, etc. Separa claramente los métodos de los problemas, lo que proporciona una gran potencia de reutilización de código y diseño

http://paradiseo.gforge.inria.fr/index.php?n=Doc.Tutorials

4. Software de Metaheurísticas

✓ **HeuristicLab** (http://dev.heuristiclab.com) es una herramienta de código abierto para el desarrollo de metaheurísticas realizada en Microsoft .NET y C#. Presenta un interfaz gráfico que permite ajustar y extender los algoritmos para un problema completo sin necesidad de escribir código. Incorpora las búsquedas por trayectorias más conocidas así como particle swarm optimization y distintos algoritmos evolutivos

✓ **FOM** (Framework for Optimization using Metaheuristics, http://www.isa.us.es/fom) es un entorno dirigido a objetos (Java) desarrollado por los autores del artículo comparativo. Incorpora los métodos habituales de trayectorias simples, GRASP y VNS, algoritmos evolutivos y de optimización mediante colonias de hormigas.

4. Software de Metaheurísticas

- Otros:

✓ **HotFrame** (Heuristic Optimization Framework, http://www1.uni-hamburg.de/IWI/hotframe/hotframe.html) proporciona componentes adaptables en C++, incluyendo diversas metaheurísticas y una arquitectura de colaboración entre las distintas componentes y clases específicas de aplicaciones y permitiendo la hibridación y la incorporación de nuevas metaheurísticas. ¡¡Sin versiones nuevas en los últimos 5 años!!

✓ **CPLEX** es una biblioteca comercial estándar de resolución de problemas de optimización desarrollada por IBM ILOG. Ganó el primer concurso internacional de la Sociedad de Investigación Operativa estadounidenses INFORMS en 2004. Incorpora técnicas clásicas (Simplex). Incluye interfaces para C++, C#, Java, Python, Excel y Matlab.

✓ **GPLK** (http://www.gnu.org/software/glpk/) es un software libre en ANSI C que implementa el método de Simplex revisado

4. Software de Metaheurísticas

- **Para desarrollar las prácticas de la asignatura se podrá emplear el software que se desee**, bien sea cualquier framework existente o bien código desarrollado por el propio alumno o bajado de Internet

- El profesor de prácticas proporcionará distintos códigos básicos de metaheurísticas desarrollados en C

- El alumno deberá indicar el software considerado en su documentación de prácticas y proporcionar las fuentes y los ejecutables realizados