Metaheurísticas y Optimización Combinatoria en redes de computadores

Fundamentos y Ejemplos

Pablo Rodríguez-Bocca

Diseño Topológico de Redes.

Facultad de Ingeniería, Universidad de la República.
INRIA/Université de Rennes 1, Rennes, France.

Noviembre 11, 2008

Contenidos

- Parte I: INTRODUCCIÓN
- Parte II: METAHEURÍSTICAS

Contenido

- Parte I: INTRODUCCIÓN
- Parte II: METAHEURÍSTICAS

Contenido de la Parte I

1 Introducción
- Sobre el Curso
- Motivación

Contenido

- Parte I: INTRODUCCIÓN
- Parte II: METAHEURÍSTICAS

Contenido de la Parte II

2 Heurísticas
- Definición
- Ejemplo

3 Metaheurísticas
- General
- Búsqueda local (LS-Local Search)
- Recocido simulado (SA-Simulated Annealing)
- Búsqueda Tabú (TS-Tabu search)
- GRASP-Greedy Randomized Adaptive Search Procedure
- Computación evolutiva (EC-Evolutionary Computation)
- Colonias de Hormigas (ANT-Ant colony optimization)

4 Resumen
- Esquema de clasificación
- Intensificación y Diversificación
- ¿Preguntas?

Part I

INTRODUCCIÓN

Contenido

1. Introducción
   - Sobre el Curso
   - Motivación

Contenido

1 Introducción
- Sobre el Curso
- Motivación

Contenido

1 Introducción
- Sobre el Curso
- Motivación

Bibliografía - libros

- Encyclopedia of Operations Research and Management Science. Gass, Saul I.; Harris, Carl M., eds. Kluwer, 1996. ISBN 0-7923-9590-5.
- Meta-heuristics: advances and trends in local search paradigms for optimization. Stefan Voss, Silvano Martello, Ibrahim H. Osman and Catherine Roucairol (eds.). Kluwer Academic Publishers, 1999. ISBN: 0-7923-8369-9.
- Essays and surveys in metaheuristics. C.C. Ribeiro, P. Hansen. Kluwer, 2001.
- Genetic Algorithms in search, optimization, and machine learning. David E. Goldberg. Addison-Wesley, 1989. ISBN 0201157675.
- Meta-heuristics : theory and applications. Osman, Ibrahim H.; Kelly, James P. eds.. Kluwer, 1996. ISBN: 0-792397-002.
- Facts, conjectures, and improvements for simulated annealing. Salamon, Peter; Sibani, Paolo; Frost, Richard. Siam, 2002. ISBN: 0898715083.

Bibliografía - artículos

- Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison. C. Blum and A. Roli. Technical report TR/IRIDIA/2001-13, IRIDIA, Université Libre de Bruxelles, Belgium
- Heuristics from nature for hard combinatorial optimization problems. A. Colorni and M. Dorigo and F. Maffioli and V. Maniezzo and G. Righini and M. Trubian. International Transactions in Operational Research 3(1):1-21.
- Testing Heuristics: We Have It All Wrong. Hooker, J. Journal of Heuristics 1:33-42. 1996.
- Designing and Reporting on Computational Experiments with Heuristic Methods. ichard S. Barr, Bruce L. Golden, James Kelly, William R. Stewart, Mauricio G.C. Resende. June 27,1995
- Metaheuristicas: Una visión global. Melián, B., Moreno Perez, J.A., Marcos Moreno-Vega, J. Revista Iberoamericana de Inteligencia Artificial. Numero 19, Volumen 2, páginas 7-28, 2003.

Links útiles

- Curso de Grado: Introducción a la Investigación de Operaciones
  http://www.fing.edu.uy/inco/cursos/io/
- Curso de Postgrado: Seminario - Elementos de Metaheurísticas.
  http://www.fing.edu.uy/inco/grupos/invop/mh/
- Curso de Postgrado: Metaheurísticas y Optimización sobre Redes
  http://www.fing.edu.uy/inco/grupos/invop/mor/
- Curso de Postgrado: Fundamentos para la investigación en Redes de Computadores
- Curso de Postgrado: Diseño Topológico de Redes.
  http://www.fing.edu.uy/inco/grupos/invop/dtr/

Contenido

1 Introducción
- Sobre el Curso
- Motivación

Investigación de Operaciones

Investigación de Operaciones

- Empleo de la metodología científica en la búsqueda de soluciones óptimas y como apoyo a la toma de decisiones a nivel operativo y gerencial.
- Área de investigación y de actividad profesional establecida a partir de la 2da Guerra Mundial.
- Etapa de crecimiento y divulgación explosiva a partir de mediados de los ’80, que continua actualmente.
- Importante sinergia con el desarrollo de las tecnologías de la información y la comunicación.

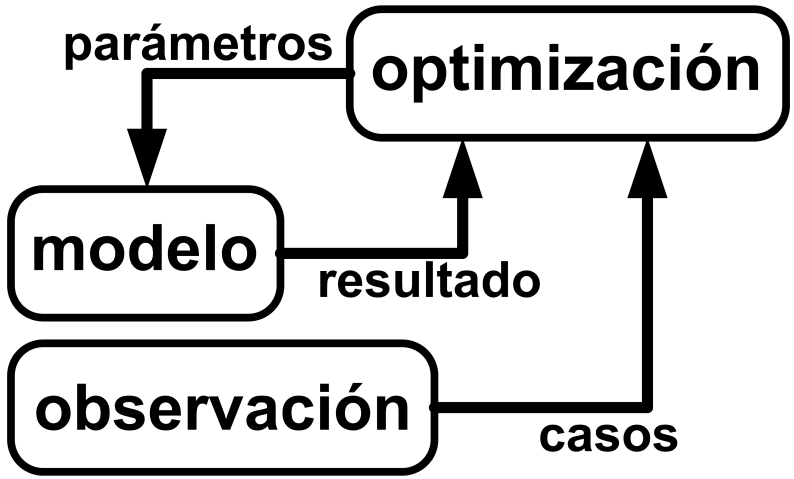
Metodología de Investigación de Operaciones

Pasos (a grandes rasgos):

1. Planteo y Análisis del problema a resolver.
2. Construcción de un modelo adecuado.
3. Obtención de datos y ajuste de parámetros del modelo.
4. Deducción de la(s) solucion(es).
5. Validación del modelo y evaluación de solucion(es).
6. Ejecución y Control de la(s) solucion(es).

Optimización:

concierne fundamentalmente etapas 2 y 4.



Problemas de Optimización

Problema

$\min f(x)$
tal que $x \in D$

Nomenclatura:

- $x = (x_1, x_2, ..., x_n)$ variables del problema.
- $f(x)$ función objetivo.
- D espacio de soluciones factibles.
- Valor óptimo de f : $f_0 = \min \{f(x) : x \in D\}$
- Conjunto de soluciones óptimas $S_0 = \{x \in D : f(x) = f_0\}$ (también llamadas soluciones globalmente óptimas).

Problemas de Programación Matemática

Problema

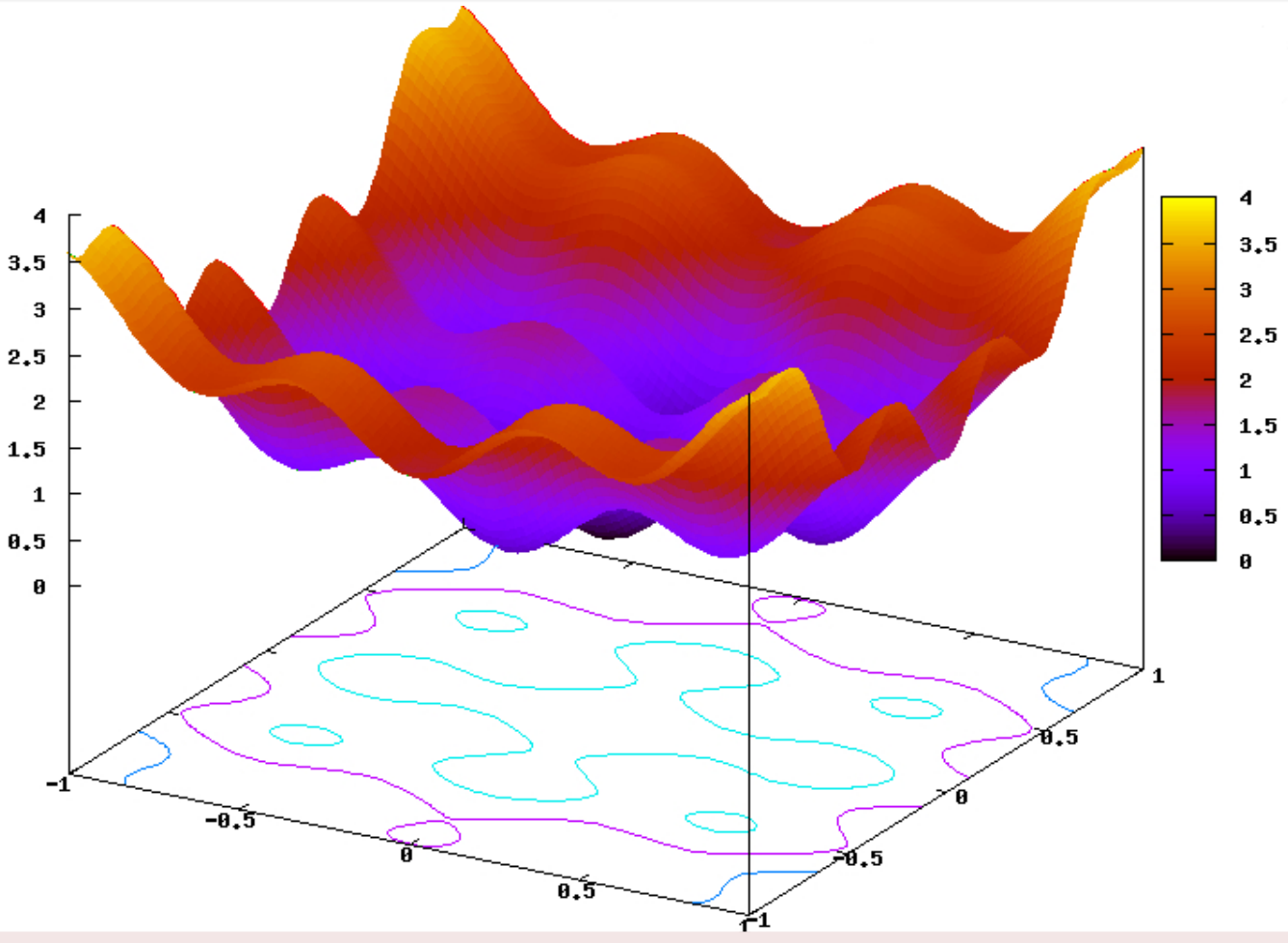
min $f(x)$
tal que $g(x) \leq 0$, $x \in X$

Nomenclatura:

- $x = (x_1, x_2, ..., x_n)$ variables del problema.
- $f(x)$ función objetivo.
- $g(x)$ restricciones del problema.
- X espacio de soluciones.
- $D = \{x \in X : g(x) \leq 0\}$ espacio de soluciones factibles.

Problemas de Programación Matemática (2)

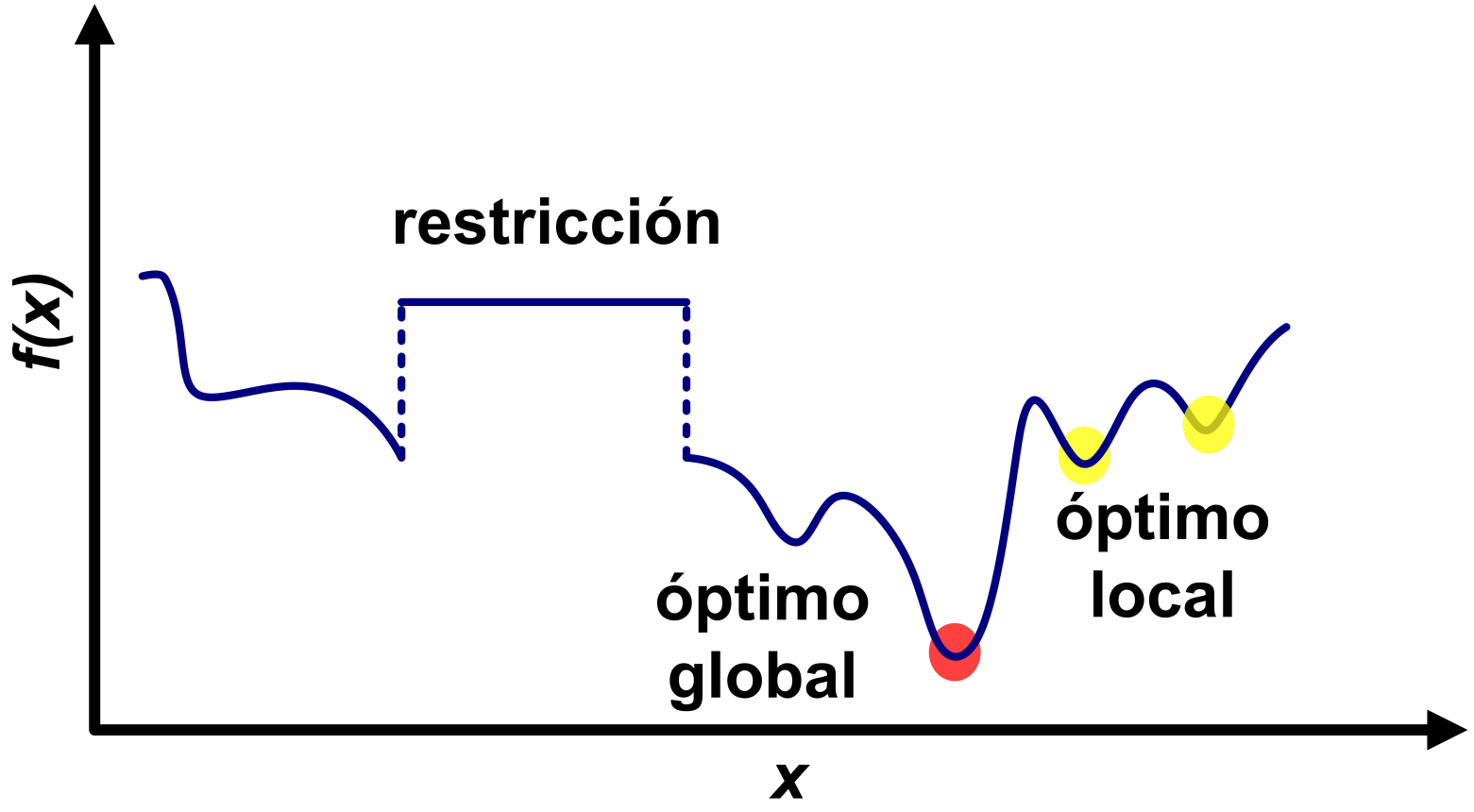
Graficamente,... ej. $x^2 + 2y^2 - 0.3cos(3\pi x) - 0.4cos(4\pi y) + 0.7$



Variables continuas.

Problemas de Programación Matemática (3)

Óptimos locales y globales...



Problemas de Optimización Combinatoria

Problema

min $f(x)$
tal que $x \in D$

Nomenclatura:

- $x = (x_1, x_2, ..., x_n)$ variables del problema.
- $f(x)$ función objetivo.
- Para todo i, $x_i \in D_i$ dominio de la variable, que es un conjunto discreto (finito o infinito).
- $X = D_1 x D_2 x \ldots x D_n$ espacio de soluciones (discreto).
- $D \subseteq X$ espacio de soluciones factibles.

Ejemplo: Problema del Viajante de Comercio (TSP - Travelling salesman problem)

Sean:

- Un conjunto de ciudades.
- Una ciudad origen O.
- Un conjunto de aristas que une las ciudades, con costos asociados.

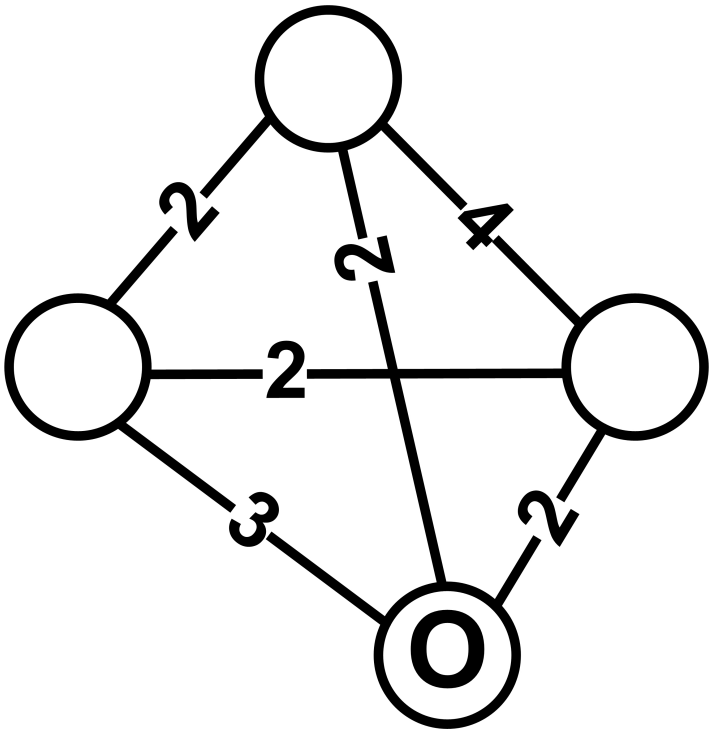
Problema del viajante:

recorrer todas las ciudades, comenzando en O y terminando en O, al menor costo posible.

Ejemplo: Problema del Viajante de Comercio (TSP) (2)

Instancia y solución

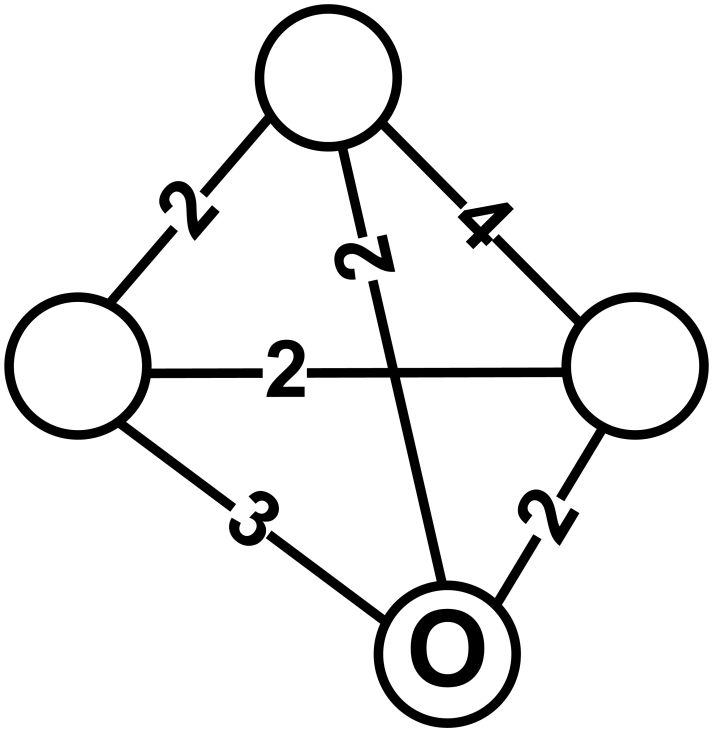
Instancia



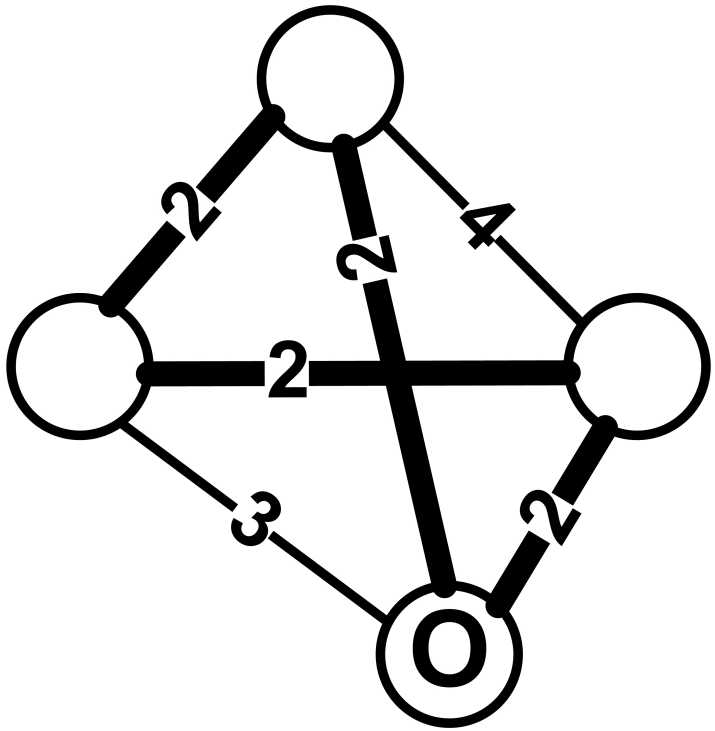
Ejemplo: Problema del Viajante de Comercio (TSP) (2)

Instancia y solución

Instancia



Una solución óptima (de valor 8)



Ejemplo: Problema del Viajante de Comercio (TSP) (3)

Formalización Miller-Tucker-Zemlin (MTZ) [MTZ60].

- Sea un grafo dirigido $G = (V, E)$, donde
  - $V = \{1, \ldots, N\}$
  - la ciudad origen es la 1
  - c_{ij} es el costo de ir de la ciudad i a la j
- Definimos las siguientes variables

$$x_{i,j} = 1 \text{ sii de la ciudad } i \text{ voy a la ciudad } j$$

$$u_i = k \text{ sii } i \text{ es la } k\text{-ésima ciudad del recorrido}$$

Ejemplo: Problema del Viajante de Comercio (TSP) (3)

Formalización Miller-Tucker-Zemlin (MTZ) [MTZ60]. Continuación

$$\min \sum_{i,j=1}^{N} c_{ij} x_{ij}$$

st:

$$\sum_{i=1}^{N} x_{i,j} = 1 \quad \forall i \neq j \in V \quad \text{// voy a toda ciudad una sola vez}$$

$$\sum_{j=1}^{N} x_{i,j} = 1 \quad \forall j \neq i \in V \quad \text{// salgo de toda ciudad una sola vez}$$

$u_1 = 1$ // siempre comienzo en la ciudad origen (1)

$2 \leq u_i \leq N \quad \forall i \neq 1 \in V$ // luego recorro el resto de las ciudades

$u_i - u_j + 1 \leq (N-1)(1 - x_{ij}) \quad \forall i \neq 1 \in V, \forall j \neq 1 \in V$

// si voy de i a $j \Rightarrow j$ está después

$x_{i,j} \in \{0,1\} \ \forall i,j \in N, u_i \in \{0,\ldots,N\} \forall i \in V$

Complejidad

Complejidad de un algoritmo:

Cantidad de operaciones elementales que se efectúan en el peor caso (en función del tamaño de los datos de entrada).

Complejidad de un problema:

Complejidad del algoritmo más eficiente para resolverlo.

Clases de complejidad:

- Clase *P* (polinomial)
- Clase *NP* (no-determinista polinomial)
- Clase *EXP* (exponencial)

Complejidad (2)

Clases de complejidad de problemas *P* y *NP*

Clase *P* (polinomial)

Conjunto de problemas de decisión que pueden ser resueltos en tiempo polinomial respecto al tamaño de la instancia por una maquina secuencial determinista.

Ejemplo de problema en *P*: Sorting

Clase *NP* (no-determinista polinomial)

Conjunto de problemas de decisión donde puede verificarse la correctitud de una solución en tiempo polinomial por una maquina secuencial determinista.
O lo que es lo mismo, pueden ser resueltos en tiempo polinomial por una maquina de Turing no determinsta.

Ejemplo de problema *NP*: Problema de satisfacibilidad booleana.

Complejidad (3)

Clases de complejidad de problemas *P* y *NP*

Hay problemas que se sabe no son *P* ni *NP*, por ejemplo: halting problem

Reducción de problemas

Un problema $P_1 \in NP$ puede ser "transformado" o reducido a otro problema $P_2 \in NP$ si existe un algoritmo polinomial que transforme cada instancia de P_1 en una instancia de P_2, manteniendo la correctitud en la solución.

Complejidad (4)

Problema abierto: $P \neq NP$?

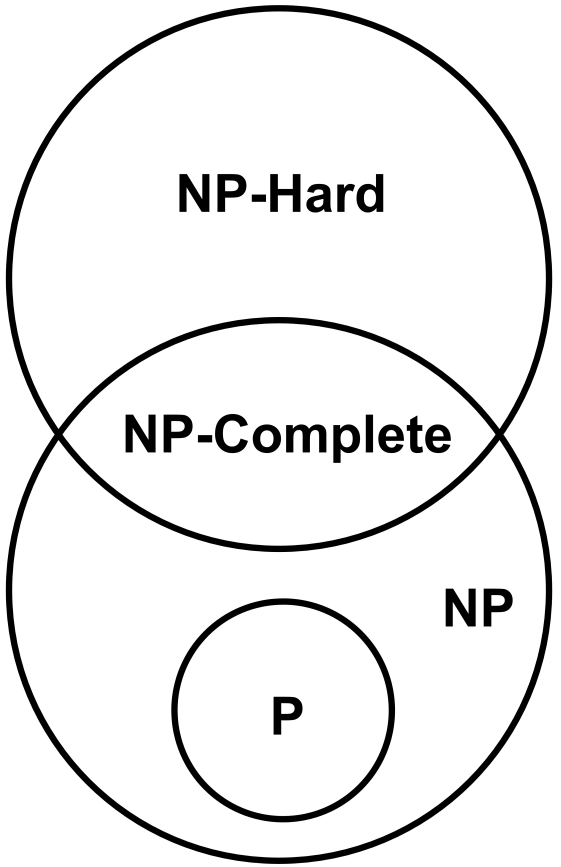
Problema abierto: $P \neq NP$?

Clases:

- $NP - completo$: problemas en NP, tales que todo problema en NP se pueden reducir a ellos
- $NP - dificil$: todo problema en NP se pueden reducir a ellos (no necesariamente NP)

Problemas de optimización:

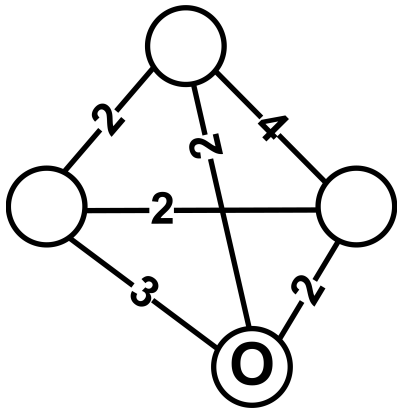
- algunos en P.
- muchos en $NP - completo$ o $NP - dificil$ (por ejemplo el TSP).



Ejemplo: Problema del Viajante de Comercio (TSP) (3)

Enumerado en TSP

- Problema
  - 4 ciudades
  - (4-1)!=6 recorridos
  - Evaluación=0.1s
  - Cómputo=0.1*6=0.6s

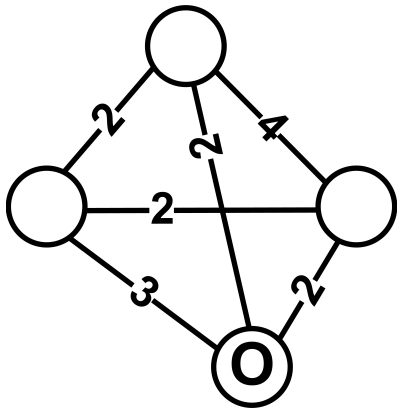


- Problema
  - 51 ciudades (grafo completo)
  - 50!=3.0E64 recorridos
  - Evaluación=0.00001s
  - Cómputo=9.6E51 años

Ejemplo: Problema del Viajante de Comercio (TSP) (3)

Enumerado en TSP

- Problema
  - 4 ciudades
  - (4-1)!=6 recorridos
  - Evaluación=0.1s
  - Cómputo=0.1*6=0.6s



- Problema
  - 51 ciudades (grafo completo)
  - 50!=3.0E64 recorridos
  - Evaluación=0.00001s
  - Cómputo=9.6E51 años

Problemas de Optimización NP-difíciles

El problema se dificulta cuando:

- el tamaño del problema es grande.
- el espacio de soluciones tiene muchas dimensiones.
- hay un conocimiento limitado de la función objetivo, $f()$:
  - no se cuenta con informacion de derivada,
  - puede ser discontinua, con ruidos, ...
- evaluar $f()$ es costozo en tiempo o procesamiento.
- $f()$ tiene (o se sospecha que tiene) malas propiedades:
  - muchas restricciones entrincadas,
  - muchos óptimos locales,
  - mesetas, barrancos, ...

¿Cuál es un objetivo "práctico" en la optimización global?

"Existe un objetivo (e.g. encontrar el mínimo valor posible de $f()$), existen recursos (e.g. número de intentos), y el problema es como usar esos recursos en forma óptima." A. Torn and A. Zilinskas

Métodos de solución

Existen muchas clases de problemas de optimización NP-difícil (y de algoritmos para resolverlos).

Métodos:

- Resolución analítica.
- Algoritmos exactos.
- Métodos de aproximación (p.ej, métodos numéricos).
- Simulación y otros métodos aleatorizados.
- Heurísticas.

Algoritmos exactos para Optimización Combinatoria

Algoritmos exactos

- Enumeración explícita o implícita de soluciones (programación dinámica, branch & bound).
- Algoritmos basados en Programación Matemática (simplex, punto interior, branch&cut, branch&cut&price).
- Otros específicos de cada problema.
- Únicos errores: redondeo, y eventualmente truncamiento.

Métodos de aproximación

Métodos de aproximación

- Encuentra solución con error máximo conocido a priori.
- En algunos casos, error de aproximación fijo.
- En otros, posible elegir trade-off entre error de aproximación y esfuerzo computacional (mayor esfuerzo, menor error).
- Problemas en la clase PTAS - “Polynomial Time Approximation Schemes”.

Métodos aleatorios

Métodos aleatorios

- Con cierta probabilidad, encuentra solución que tendrá un error máximo dado.
- Variantes:
  - Monte Carlo: siempre da una solución y estimación del error, con intervalo de confianza asociado; mayor esfuerzo computacional disminuye el error y aumenta la confianza.
  - Las Vegas: puede dar o no una solución en un tiempo dado (cuando la da, es exacta); mayor esfuerzo computacional aumenta la probabilidad de tener una respuesta.

Part II

METAHEURÍSTICAS

Contenido

2 Heurísticas
- Definición
- Ejemplo

3 Metaheurísticas
- General
- Búsqueda local (LS-Local Search)
- Recocido simulado (SA-Simulated Annealing)
- Búsqueda Tabú (TS-Tabu search)
- GRASP-Greedy Randomized Adaptive Search Procedure
- Computación evolutiva (EC-Evolutionary Computation)
- Colonias de Hormigas (ANT-Ant colony optimization)

4 Resumen
- Esquema de clasificación
- Intensificación y Diversificación
- ¿Preguntas?

Contenido

2 Heurísticas
- Definición
- Ejemplo

3 Metaheurísticas
- General
- Búsqueda local (LS-Local Search)
- Recocido simulado (SA-Simulated Annealing)
- Búsqueda Tabú (TS-Tabu search)
- GRASP-Greedy Randomized Adaptive Search Procedure
- Computación evolutiva (EC-Evolutionary Computation)
- Colonias de Hormigas (ANT-Ant colony optimization)

4 Resumen
- Esquema de clasificación
- Intensificación y Diversificación
- ¿Preguntas?

Contenido

2 Heurísticas
- Definición
- Ejemplo

3 Metaheurísticas
- General
- Búsqueda local (LS-Local Search)
- Recocido simulado (SA-Simulated Annealing)
- Búsqueda Tabú (TS-Tabu search)
- GRASP-Greedy Randomized Adaptive Search Procedure
- Computación evolutiva (EC-Evolutionary Computation)
- Colonias de Hormigas (ANT-Ant colony optimization)

4 Resumen
- Esquema de clasificación
- Intensificación y Diversificación
- ¿Preguntas?

Heurísticas

"heurística" deriva del griego heuriskein, que significa "encontrar" o "descubrir".

- Técnicas que busca soluciones de buena calidad (de valor cercano al óptimo?) a un costo computacional razonable, aunque sin garantizar la optimalidad de las mismas. En general, ni siquiera se conoce el grado de error.
- Clasificación de las heuristicas:
  - Métodos Constructivos: generan una solución (desde una solución vacia) agregando componentes hasta completar la solución. Son métodos rápidos, de baja calidad en los resultados.
  - Métodos de Búsqueda local: comienzan desde una solución e iterativamente van remplazando la solución actual con alguna solución parecida de mejor calidad.

Contenido

2 Heurísticas
- Definición
- Ejemplo

3 Metaheurísticas
- General
- Búsqueda local (LS-Local Search)
- Recocido simulado (SA-Simulated Annealing)
- Búsqueda Tabú (TS-Tabu search)
- GRASP-Greedy Randomized Adaptive Search Procedure
- Computación evolutiva (EC-Evolutionary Computation)
- Colonias de Hormigas (ANT-Ant colony optimization)

4 Resumen
- Esquema de clasificación
- Intensificación y Diversificación
- ¿Preguntas?

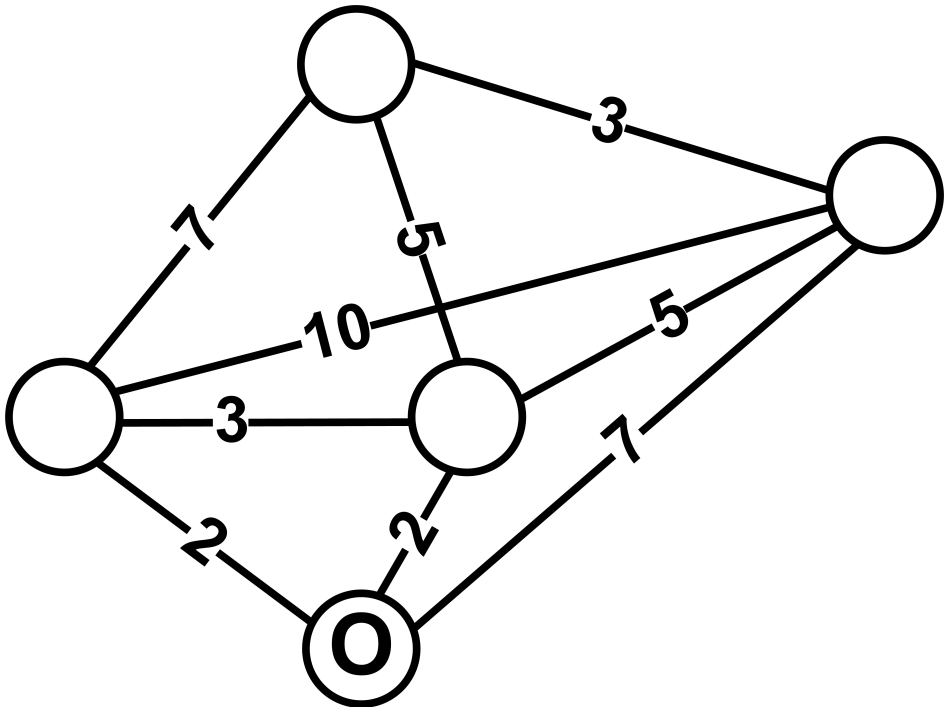
Ejemplo sencillo de heurística: método goloso (greedy)

Heuristica Greedy

- Idea: tratar de construir una solución eligiendo de manera iterativa los elementos componentes de menor costo.
- Para algunos problemas con estructura particular, la solución construida es una solución óptima. En general, no es el caso.

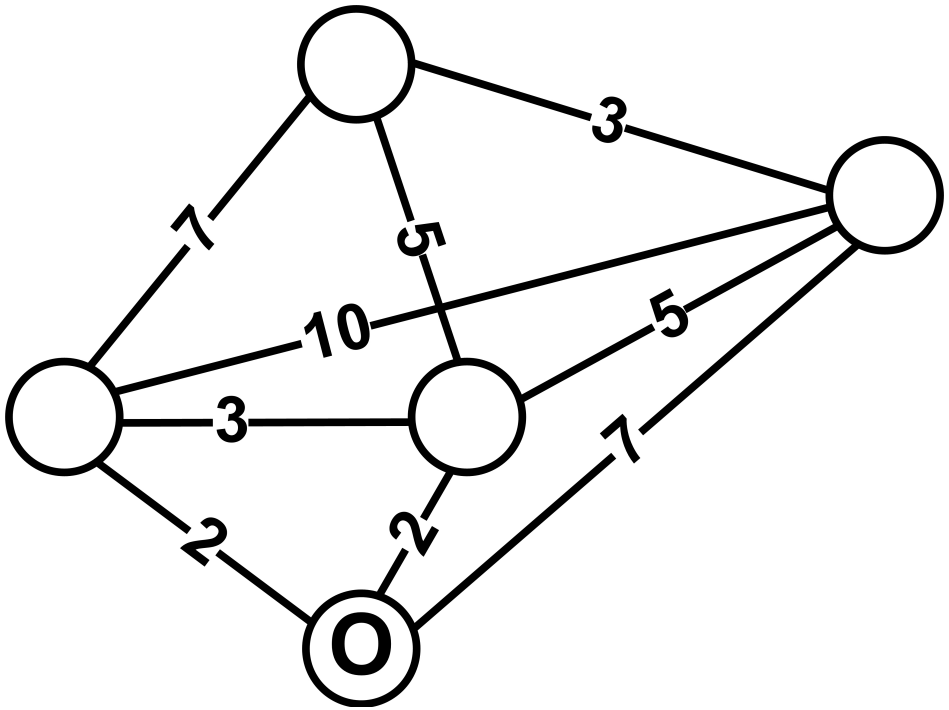
Ejemplo: Problema del Viajante de Comercio (TSP) (4)

Aplicación de Greedy



Ejemplo: Problema del Viajante de Comercio (TSP) (4)

Aplicación de Greedy

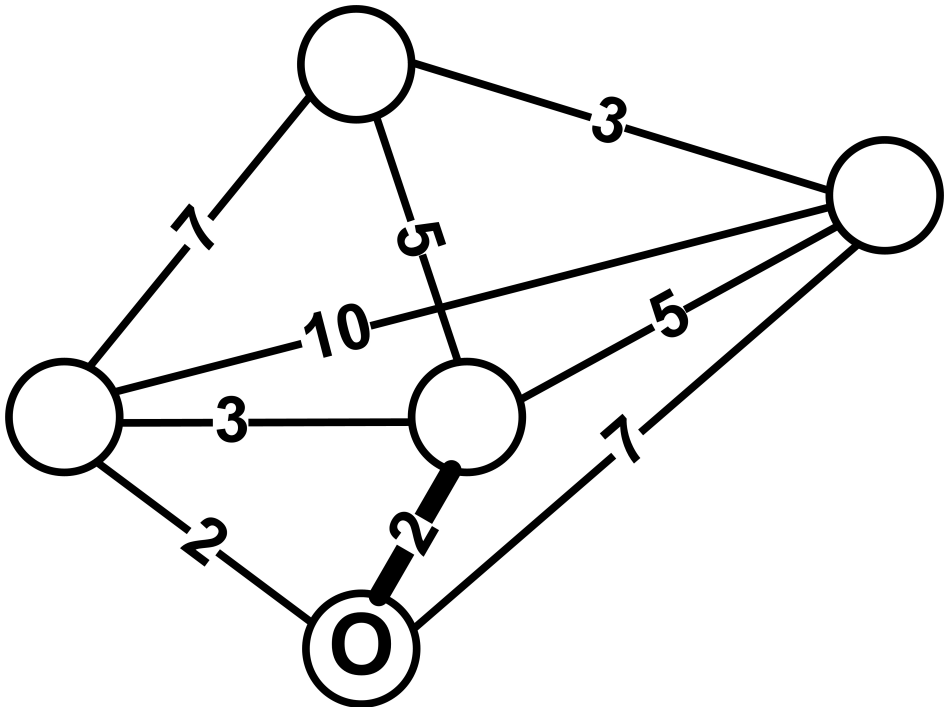


Algoritmo goloso:

Elijo la ciudad más próxima (menor costo) entre las aún no visitadas.

Ejemplo: Problema del Viajante de Comercio (TSP) (4)

Aplicación de Greedy

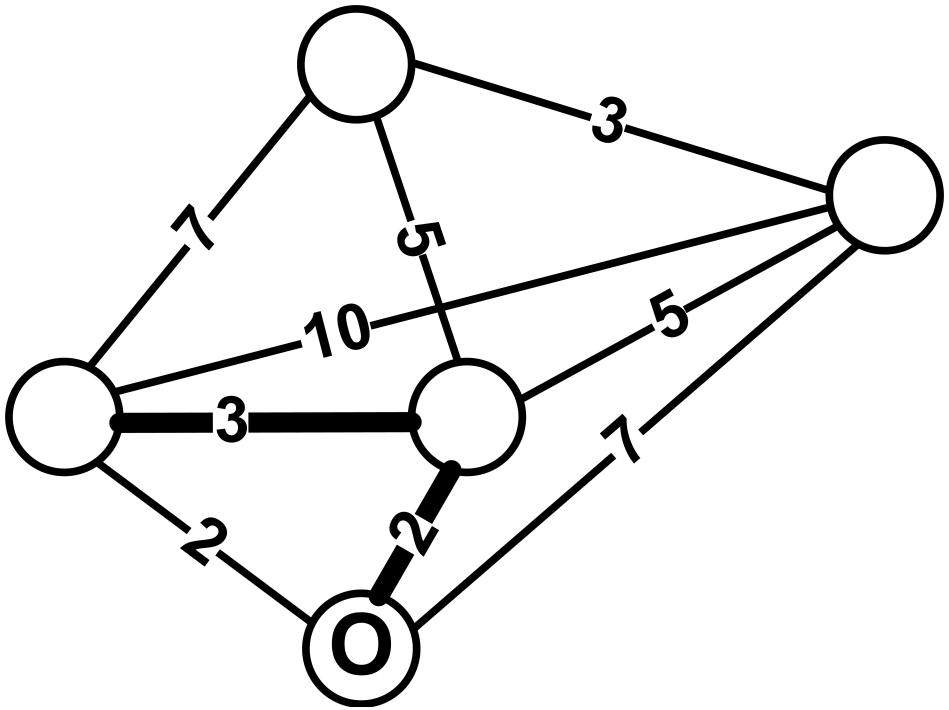


Algoritmo goloso:

Elijo la ciudad más próxima (menor costo) entre las aún no visitadas.

Ejemplo: Problema del Viajante de Comercio (TSP) (4)

Aplicación de Greedy

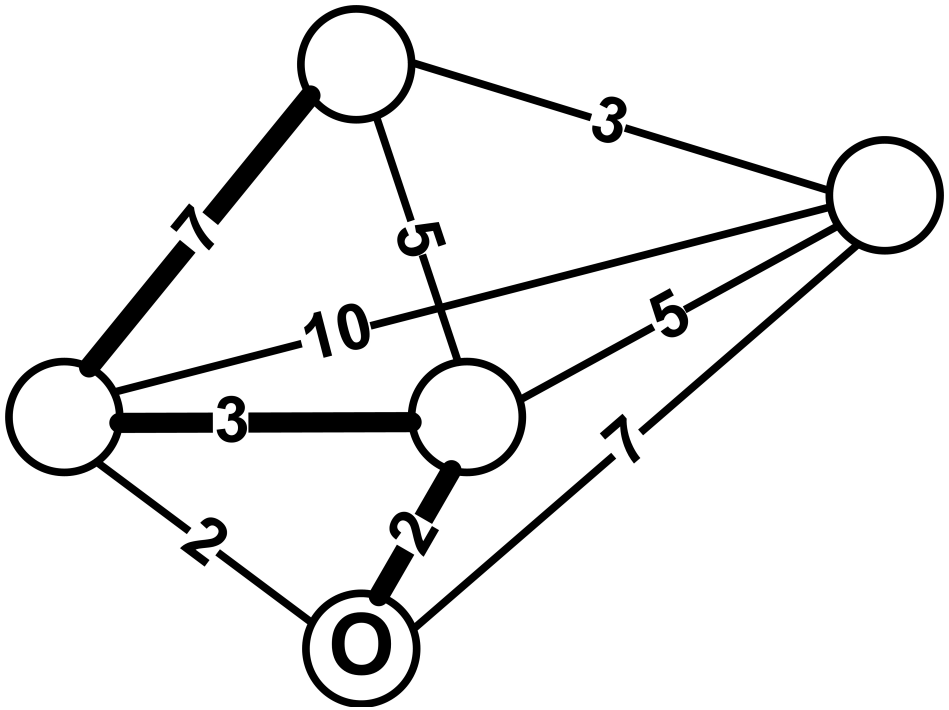


Algoritmo goloso:

Elijo la ciudad más próxima (menor costo) entre las aún no visitadas.

Ejemplo: Problema del Viajante de Comercio (TSP) (4)

Aplicación de Greedy

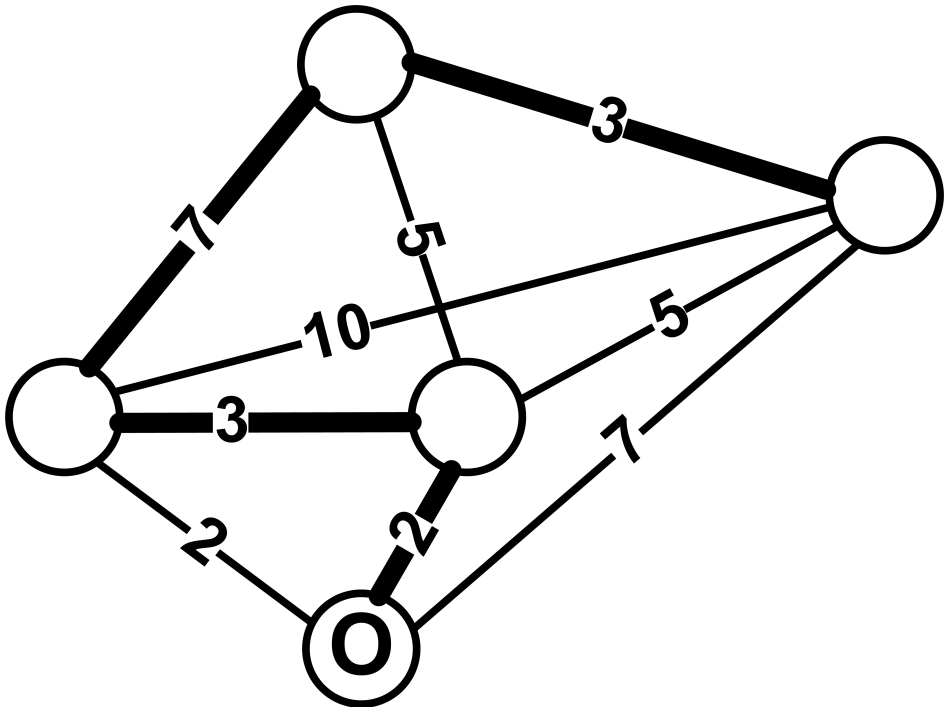


Algoritmo goloso:

Elijo la ciudad más próxima (menor costo) entre las aún no visitadas.

Ejemplo: Problema del Viajante de Comercio (TSP) (4)

Aplicación de Greedy

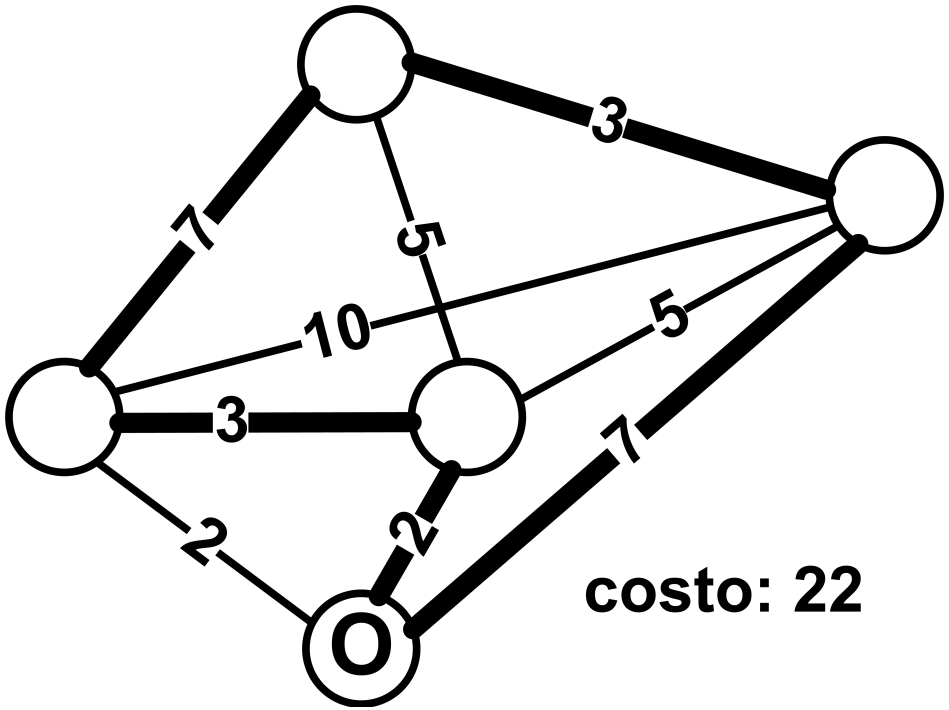


Algoritmo goloso:

Elijo la ciudad más próxima (menor costo) entre las aún no visitadas.

Ejemplo: Problema del Viajante de Comercio (TSP) (4)

Aplicación de Greedy

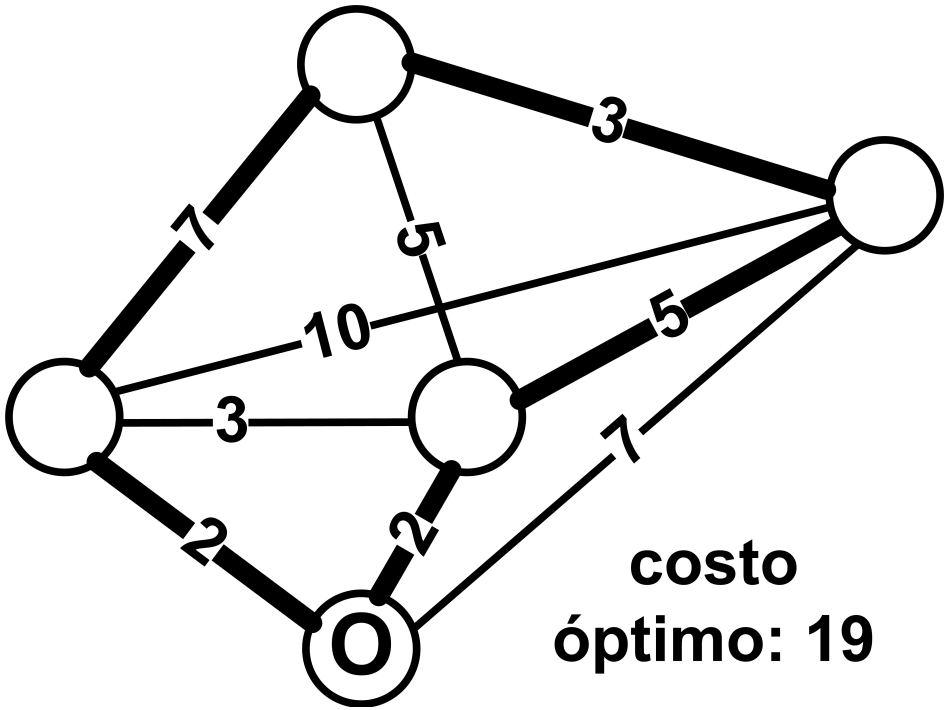


Algoritmo goloso:

Elijo la ciudad más próxima (menor costo) entre las aún no visitadas.

Ejemplo: Problema del Viajante de Comercio (TSP) (4)

Aplicación de Greedy



Algoritmo goloso:

Elijo la ciudad más próxima (menor costo) entre las aún no visitadas.

Contenido

2 Heurísticas
- Definición
- Ejemplo

3 Metaheurísticas
- General
- Búsqueda local (LS-Local Search)
- Recocido simulado (SA-Simulated Annealing)
- Búsqueda Tabú (TS-Tabu search)
- GRASP-Greedy Randomized Adaptive Search Procedure
- Computación evolutiva (EC-Evolutionary Computation)
- Colonias de Hormigas (ANT-Ant colony optimization)

4 Resumen
- Esquema de clasificación
- Intensificación y Diversificación
- ¿Preguntas?

Contenido

2 Heurísticas
- Definición
- Ejemplo

3 Metaheurísticas
- General
- Búsqueda local (LS-Local Search)
- Recocido simulado (SA-Simulated Annealing)
- Búsqueda Tabú (TS-Tabu search)
- GRASP-Greedy Randomized Adaptive Search Procedure
- Computación evolutiva (EC-Evolutionary Computation)
- Colonias de Hormigas (ANT-Ant colony optimization)

4 Resumen
- Esquema de clasificación
- Intensificación y Diversificación
- ¿Preguntas?

Metaheurística

Definición (?)

"Heurística de nivel más alto" (Glover, 1986).

Características:

- estrategia que guía el proceso de búsqueda, incluyendo en general heurísticas subordinadas.
- admite descripción a nivel abstracto (es de alto nivel).
- de uso genérico (no específica para un tipo de problema).
  - debe instanciarse para cada tipo de problema.

Metaheurísticas

Objetivos:

- Encontrar soluciones factibles de buena calidad (valor cercano al óptimo).
- Encontrar rápidamente soluciones factibles (puede ser en sí mismo un problema NP-difícil).
- Recorrer el espacio de soluciones sin quedar “atrapados” en una zona. Noción de:
  - exploración del espacio,
  - explotación de las soluciones obtenidas.

Historia de las Metaheurísticas

wikipedia. . .

1952 first works on stochastics optimization methods.
1954 use of evolution process on optimization (Barricelli).
1966 evolutionary programming (Fogel, Owens et Walsh).
1975 genetic algorithms (J. Holland).
1980 genetic programming (Smith).
1983 simulated annealing (Hastings, Kirkpatrick, Gelatt, Vecchi).
1986 first mention of the term meta-heuristic, during the conception of tabu search (Glover)
1988 first use of ants (F. Moyson, B. Manderick).
1989 memetic algorithm (P. Moscato).
1991 ant colony algorithms (M. Dorigo)
1995 Greedy randomized adaptive search procedure (Feo, Resende).
1997 cross entropy method (Rubinstein).
2004 Honey bee algorithm (S. Nakrani, S. Tovey).
2008 Firefly algorithm (X. S. Yang).

Clasificaciones de Metaheurísticas

Inspiradas (o no) en la naturaleza (sistemas biológicos, físicos o sociales).

- Algoritmos Genéticos - Evolución de las especies
- Recocido simulado - Enfriamiento de metales
- Colonias de hormigas - ídem.
- Búsqueda dispersa - no bio-inspirada.

Aleatorias vs. determinísticas.

- Aleatorias: Algoritmos Genéticos, Recocido simulado, Colonias de hormigas.
- Determinísticas: Búsqueda Tabú, Búsqueda dispersa.

Clasificaciones de Metaheurísticas (2)

Basadas en un individuo vs. basadas en poblaciones:

- Basadas en un individuo: Recocido simulado, GRASP, búsqueda tabú, búsqueda local y variantes.
- Basadas en poblaciones: Algoritmos Genéticos, Colonias de hormigas, Búsqueda dispersa.

Con memoria vs. sin memoria

- Sin memoria: Recocido simulado, GRASP, Algoritmos Genéticos, búsqueda local y variantes.
- Con memoria: Búsqueda tabú,Colonias de hormigas.

Clasificaciones de Metaheurísticas (3)

varias clasificaciones más. . .

- Función objetivo estática / dinámica.
- Espacio sin estructura / espacio estructurado.
- Uso de única función de vecindad / múltiples vecindades.
- . . .

Estructura del espacio

Problema de Optimización Combinatoria

min $f(x)$
tal que $x \in D \subseteq X$

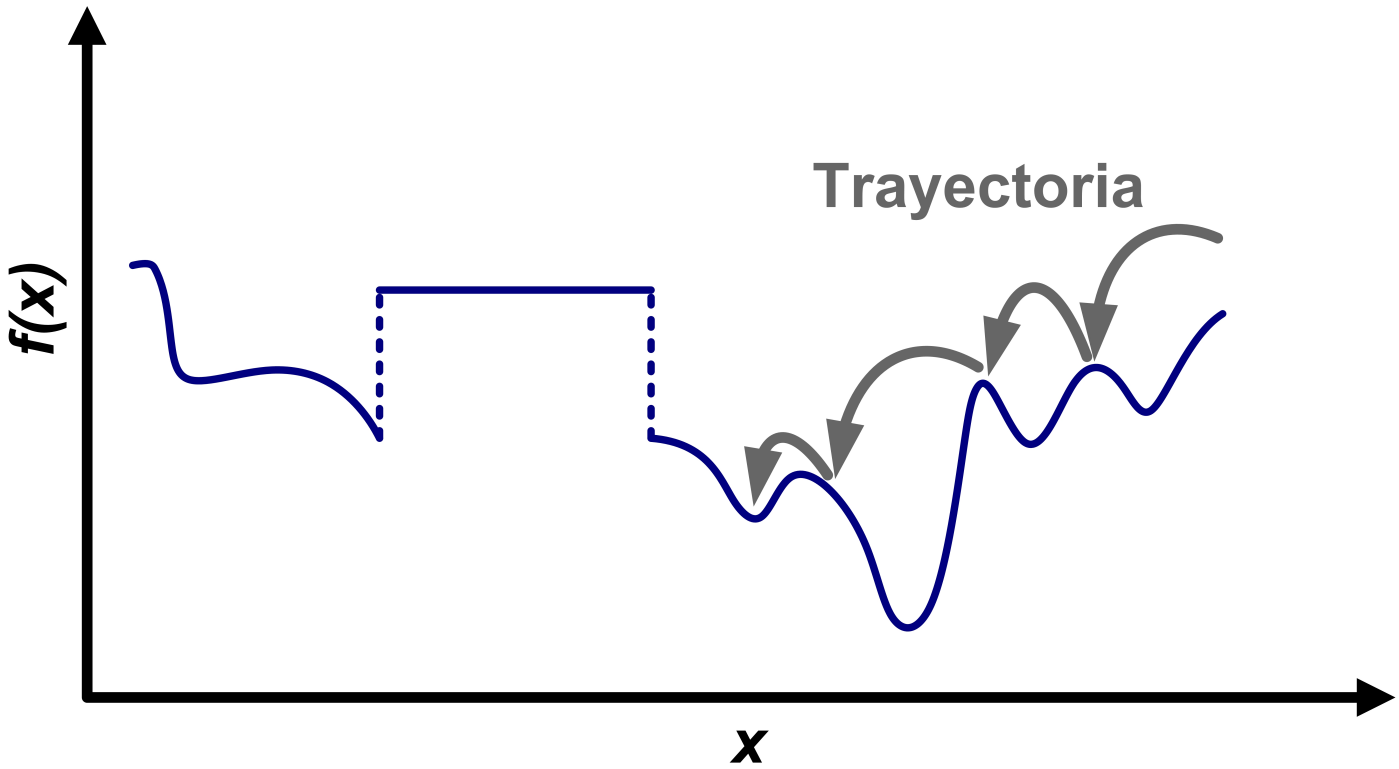
- El espacio de soluciones X es discreto (D, espacio de soluciones factibles, tambien discreto).
- Se necesita una estructura similar al concepto de entorno del continuo (ϵ,...).

La vecindad

- Estructura de vecindad: función $N : D \rightarrow 2^D$, que asigna a cada solución x un conjunto de "soluciones vecinas".
- Soluciones localmente mínimas (mínimos locales): x es mínimo local si $\forall y \in N(x), f(x) \leq f(y)$ (estricto si $<$).
- Mínimo global es la solución de valor mínimo entre todos los mínimos locales

Métodos de trayectoria

En cada iteración se intenta mejorar una solución.



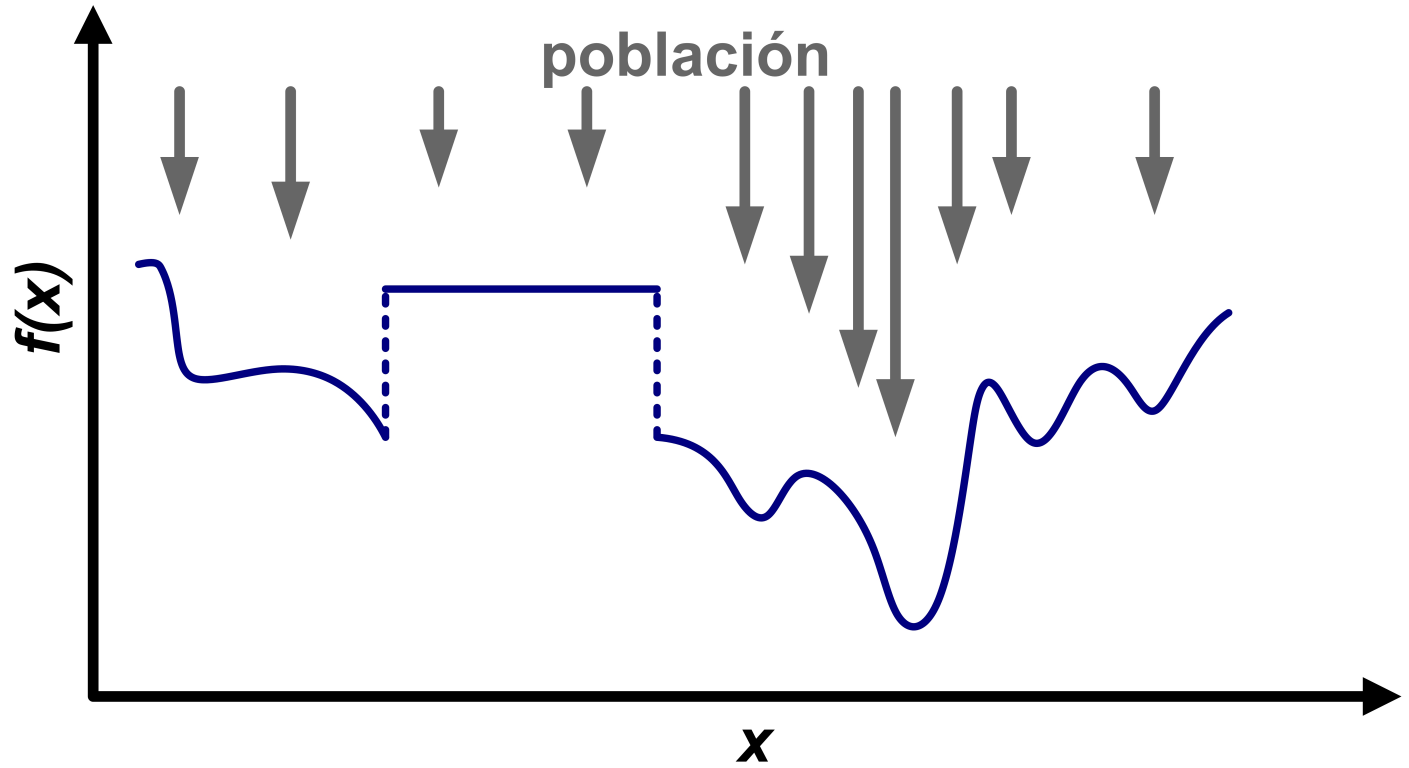
Métodos de trayectoria (2)

Ejemplos

- Método de búsqueda local básico
- Recocido simulado (SA - Simulated Annealing)
- Búsqueda Tabú (TS - Tabu search)
- Métodos de exploración local (ELS - Explorative local search)
  - GRASP - Greedy randomized adaptive search procedure
  - VNS - Variable neighborhood search
  - GLS - Guided local search
  - ILS - Iterated local search

Métodos basados en poblaciones

En cada iteración se intenta mejorar un conjunto (o población) de soluciones.



Métodos basados en poblaciones (2)

Ejemplos

- Computación evolutiva (EC)
  - Algoritmos genéticos
  - Algoritmos meméticos
  - Búsqueda dispersa (Scatter search) y caminos reconectados (path relinking)
  - Algoritmos evolutivos basados en modelos probabilísticos
- Colonias de hormigas (ANT - Ant colony optimization)

Algunas Metaheurísticas muy difundidas

Metaheuristicas que estudiaremos...

Métodos de trayectoria (un individuo):

- Búsqueda local y variantes.
- Recocido Simulado.
- Búsqueda Tabú.
- GRASP.

Métodos basados en poblaciones:

- Algoritmos Genéticos.
- Colonias de Hormigas.

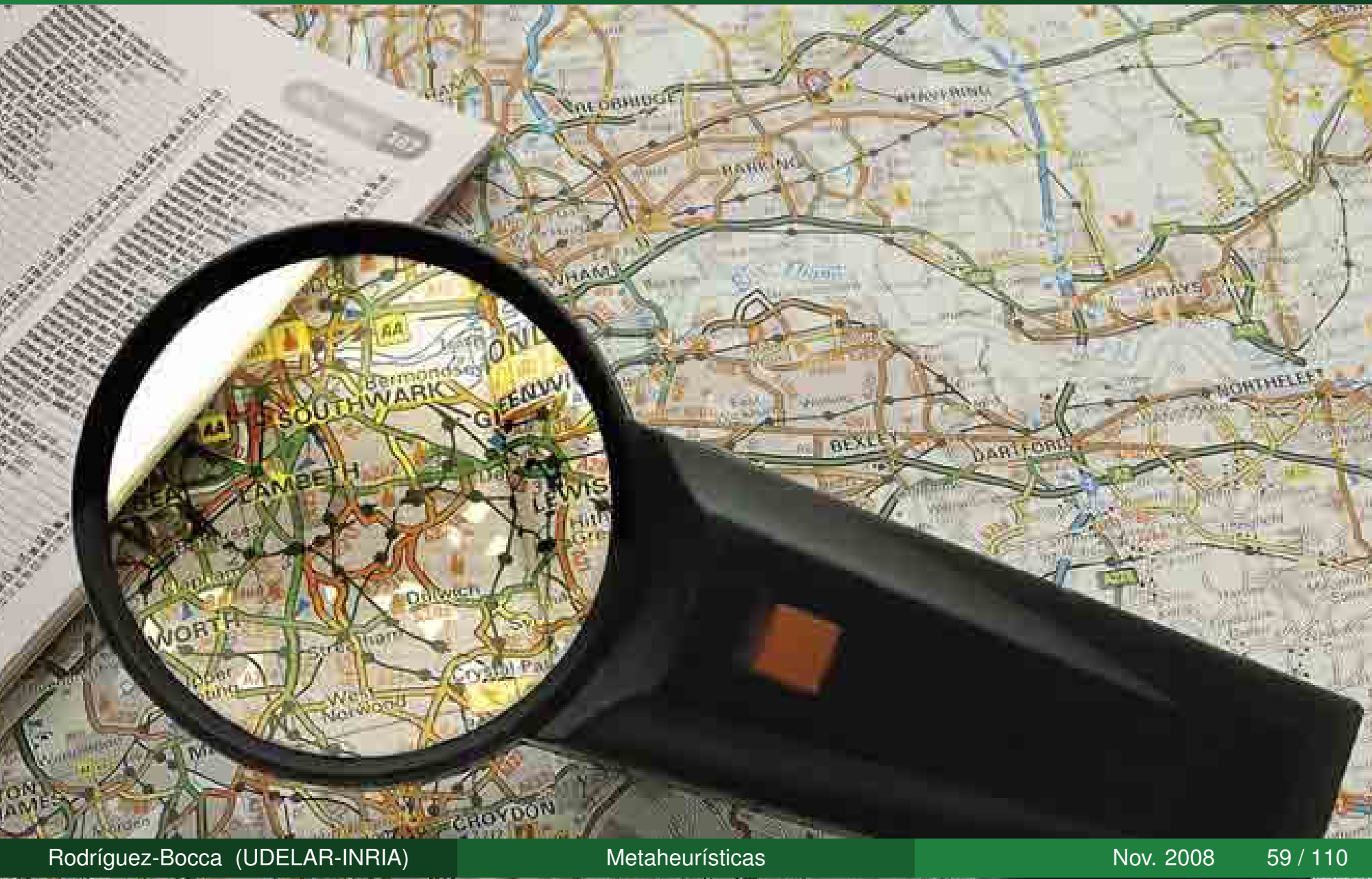
Contenido

2 Heurísticas
- Definición
- Ejemplo

3 Metaheurísticas
- General
- Búsqueda local (LS-Local Search)
- Recocido simulado (SA-Simulated Annealing)
- Búsqueda Tabú (TS-Tabu search)
- GRASP-Greedy Randomized Adaptive Search Procedure
- Computación evolutiva (EC-Evolutionary Computation)
- Colonias de Hormigas (ANT-Ant colony optimization)

4 Resumen
- Esquema de clasificación
- Intensificación y Diversificación
- ¿Preguntas?

Búsqueda local (LS-Local Search)



Búsqueda local (LS-Local Search)

- Tambien llamado mejora iterativa (Iterative Improvement).
- Los movimientos se realizan sólo si se mejora la solución.

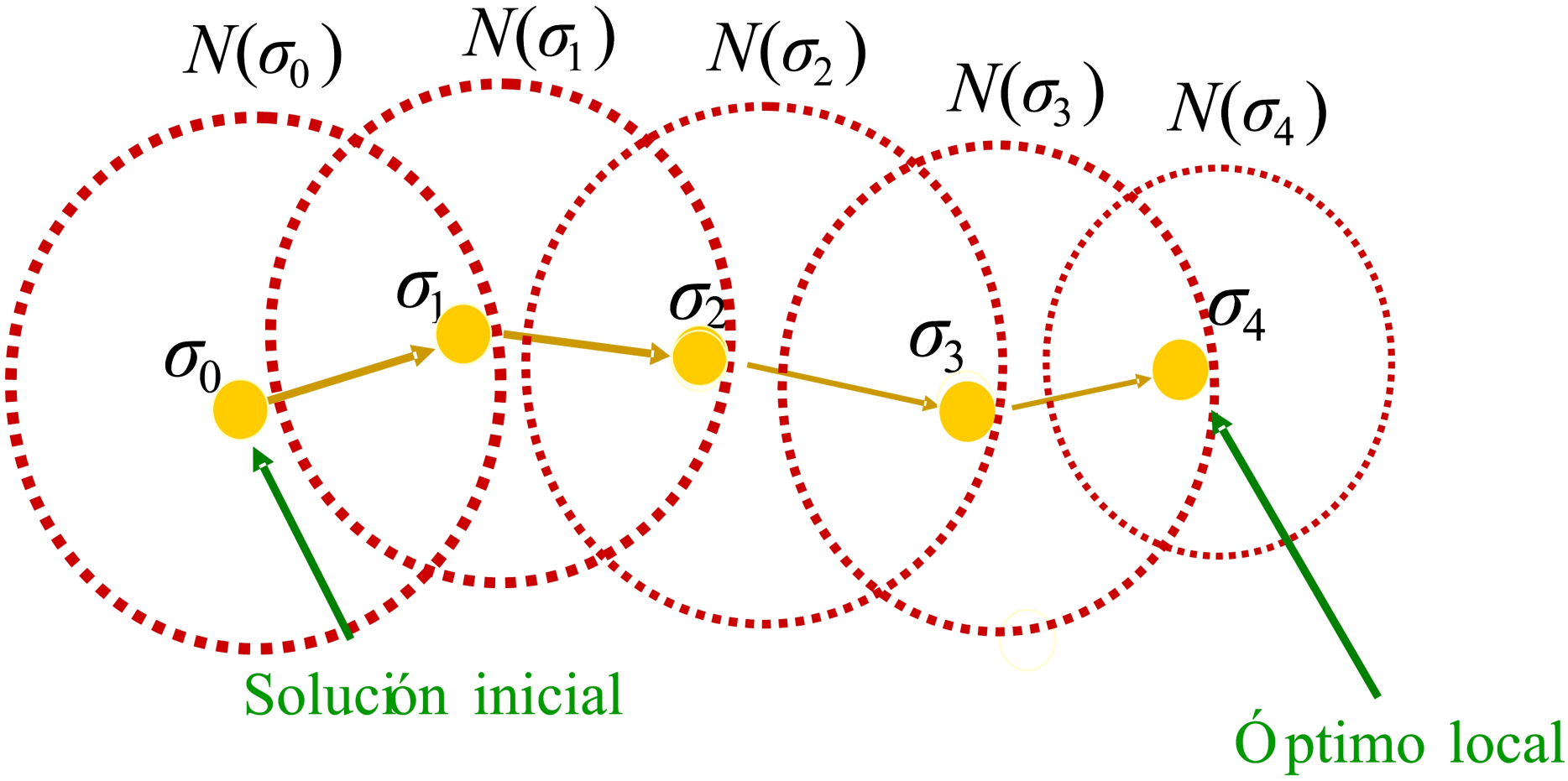
Algorithm LS

1: $s \leftarrow$ GenerateInitialSolution()
2: **repeat**
3: $\quad s \leftarrow$ Improve($s, N(s)$)
4: **until** no improvement is possible

Figure: LS - Local Search algorithm

Búsqueda local (2)

Noción de Trayectoria



LS queda atrapado en el mínimo local cercano a la solución inicial

Búsqueda local (3)

implementación para el TSP [Jay]...

- Solution Representation: A feasible solution is represented as a sequence of nodes, each node appearing only once and in the order it is visited.
  - The first and the last visited nodes are fixed to 1 (it is not specified in the solution representation and is always understood to be node 1).
- Initial Solution: A good feasible solution can be found quickly using a greedy approach
  - Starting with the first node in the tour, find the nearest node.
  - Each time find the nearest unvisited node from the current node until all the nodes are visited.
- Neighborhood: Any other solution that is obtained by a pair wise exchange of any two nodes in the solution.
  - This always guarantees that any neighborhood to a feasible solution is always a feasible solution (i.e, does not form any sub-tour).
  - There are C_2^{N-1} neighborhoods to a given solution.

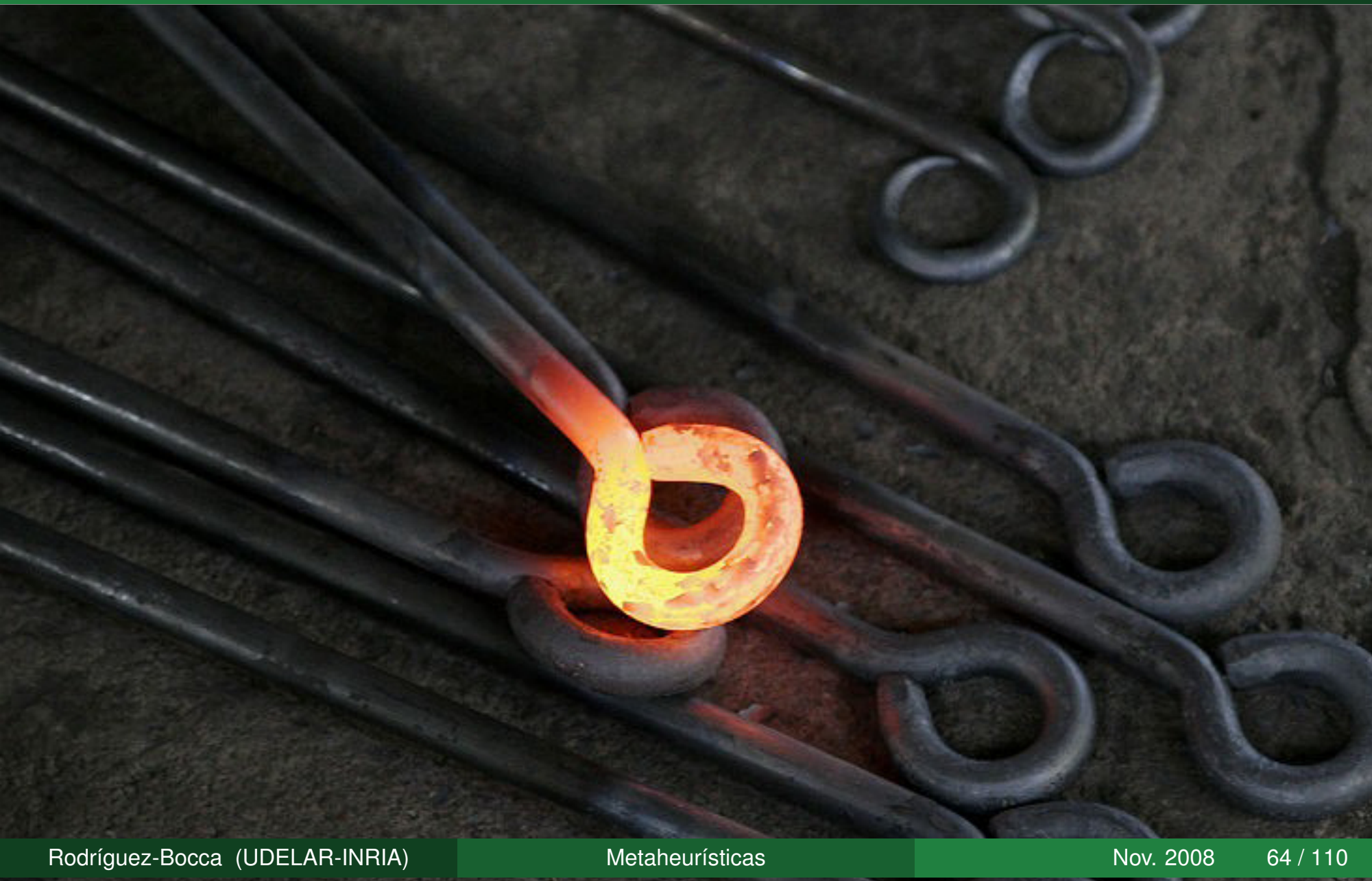
Contenido

2 Heurísticas
- Definición
- Ejemplo

3 Metaheurísticas
- General
- Búsqueda local (LS-Local Search)
- Recocido simulado (SA-Simulated Annealing)
- Búsqueda Tabú (TS-Tabu search)
- GRASP-Greedy Randomized Adaptive Search Procedure
- Computación evolutiva (EC-Evolutionary Computation)
- Colonias de Hormigas (ANT-Ant colony optimization)

4 Resumen
- Esquema de clasificación
- Intensificación y Diversificación
- ¿Preguntas?

Recocido simulado (SA-Simulated Annealing)



Recocido simulado (SA-Simulated Annealing)

- Es una de las metaheurísticas más antigua que incorpora una estrategia explícita para impedir óptimos locales.
- Basado en la física del calentamiento de metales. Se propone una similitud entre una buena estructura cristalina de metales y una buena estructura de soluciones para problemas de optimización combinatoria.
- Se trata de minimizar una función objetivo que representa la energía del sistema.

Recocido simulado (SA-Simulated Annealing)

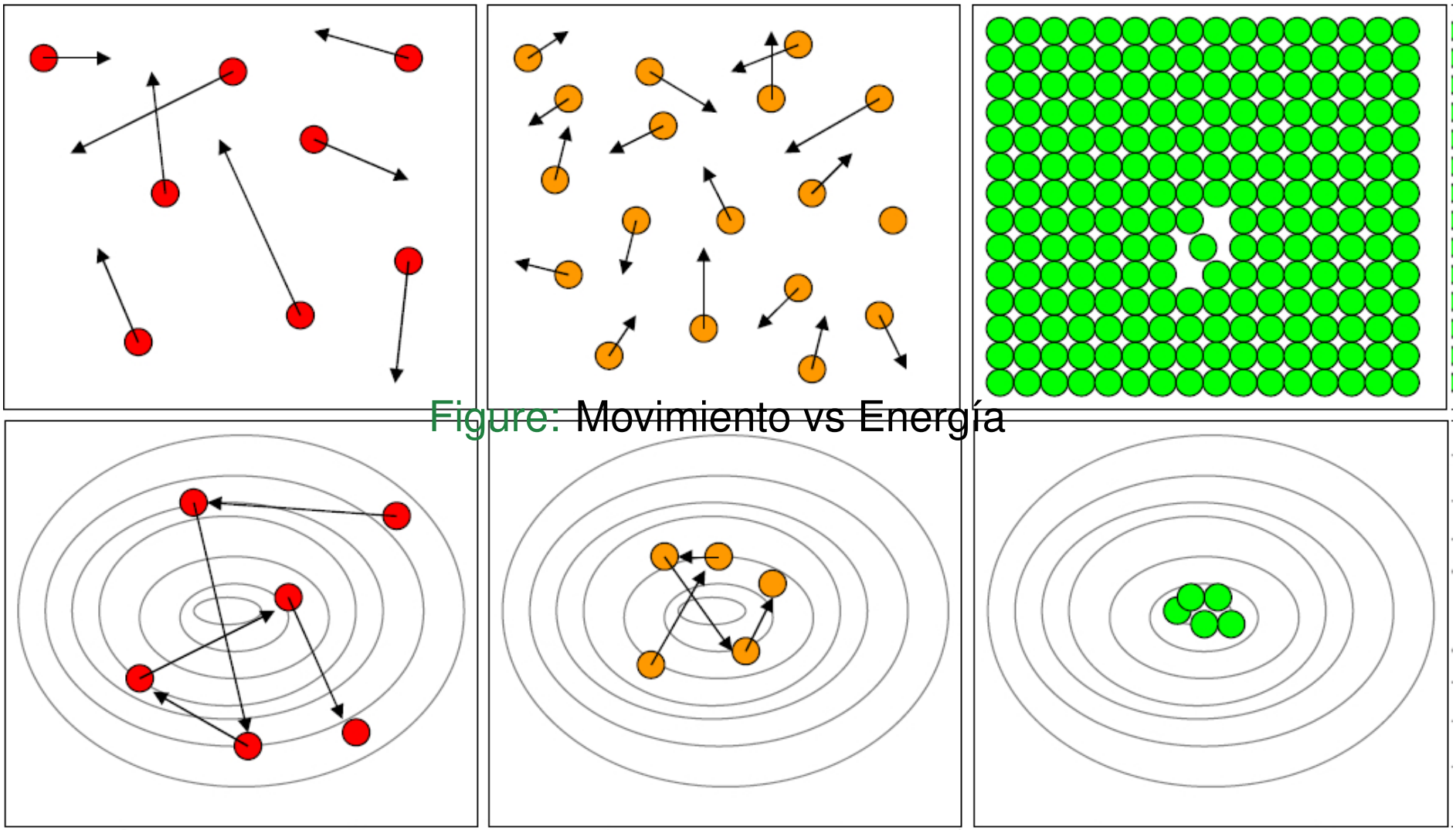


Figure: Movimiento vs Energía

Recocido simulado (2)

Idea

- Idea: permitir movimientos a soluciones que empeoren la función objetivo de forma de poder escapar a los óptimos locales.
- La probabilidad de realizar movimientos que empeoren decrece durante la búsqueda.

La temperatura del sistema

- Mientras que las soluciones que mejoran la función objetivo siempre son aceptadas, las soluciones que la empeoran son aceptadas con mayor probabilidad si la temperatura es más alta.
- Al comienzo, la temperatura es alta y cualquier transición entre estados es permitida
- La temperatura del sistema es controlada con enfriamiento sucesivo (función logarítmica) y recalentamientos periódicos, que permiten escapar de óptimos locales (estructura de soluciones).

Recocido simulado (3)

convergencia

- El proceso se puede describir en términos de cadenas de Markov de estado finito y se puede demostrar su convergencia.
- (pero un enfriamiento logarímico es muy lento, por tanto en la práctica no se aplica...)

Recocido simulado (4)

Algorithm SA

1: $s \leftarrow$ GenerateInitialSolution()
2: $T \leftarrow T_0$
3: **while** termination conditions not met **do**
4: Pick $s' \in N(s)$ at random
5: **if** $f(s') < f(s)$ **then**
6: $s \leftarrow s'$
7: **else**
8: Accept s' as new solution with probability $P(T, s', s)$
9: **end if**
10: Update(T)
11: **end while**

Figure: SA - Simulated Annealing algorithm

Recocido simulado (5)

implementación para el TSP [Jay]. . .

- Solution Representation, Initial Solution, and Neighborhood for SA are same as that for LS described before.
- Termination criteria: The algorithm terminates if it meets any one of the following criteria:
  1. It reaches a pre-specified number of iterations.
  2. There is no improvement in the solution for last pre-specified number of iterations.
  3. Fraction of neighbor solutions tried that is accepted at any temperature reaches a pre-specified minimum.

  Usually, the maximum number of iterations is kept large enough to allow the process to terminate either using criterion 2 or 3.

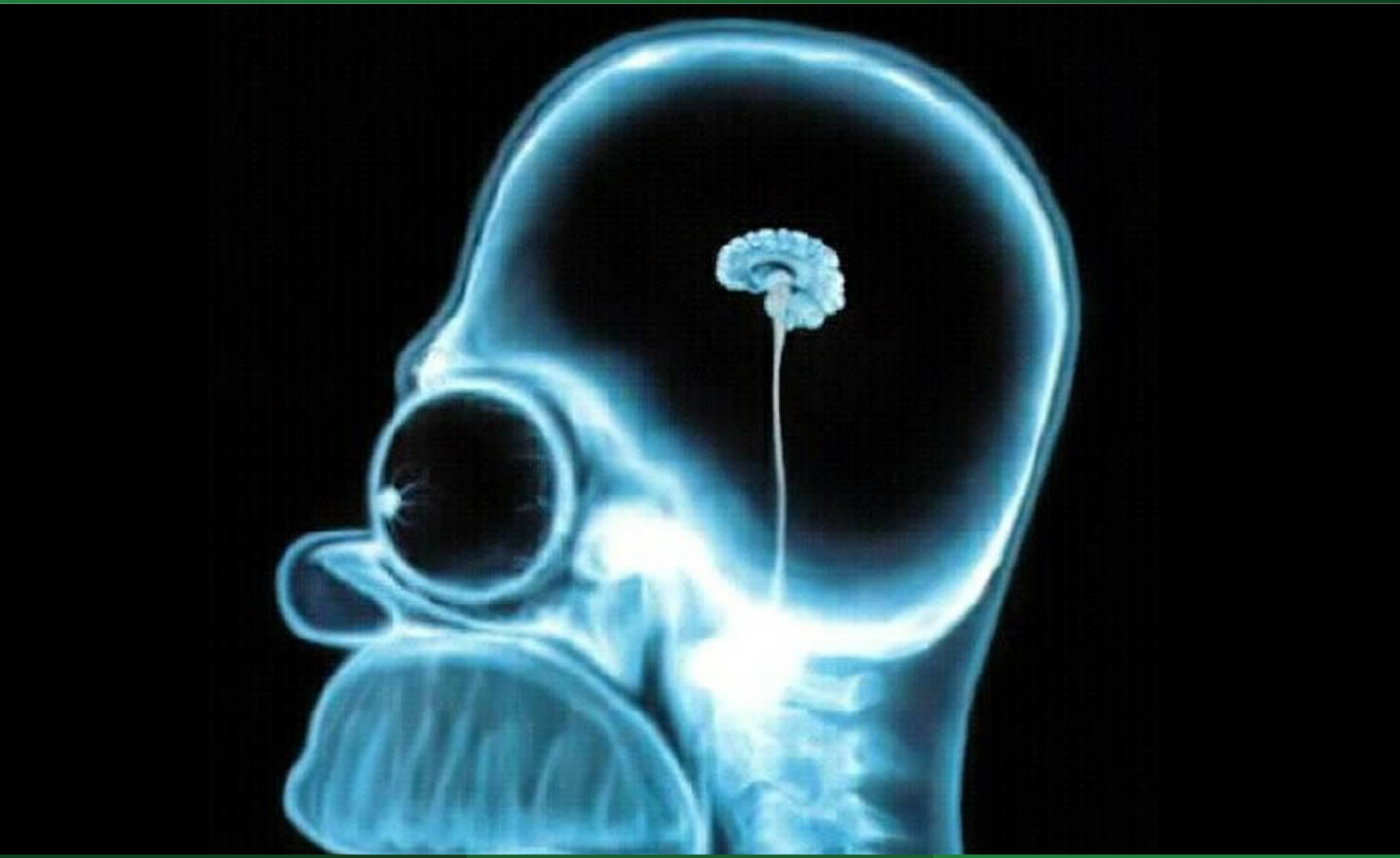
Contenido

2 Heurísticas
- Definición
- Ejemplo

3 Metaheurísticas
- General
- Búsqueda local (LS-Local Search)
- Recocido simulado (SA-Simulated Annealing)
- Búsqueda Tabú (TS-Tabu search)
- GRASP-Greedy Randomized Adaptive Search Procedure
- Computación evolutiva (EC-Evolutionary Computation)
- Colonias de Hormigas (ANT-Ant colony optimization)

4 Resumen
- Esquema de clasificación
- Intensificación y Diversificación
- ¿Preguntas?

Búsqueda Tabú (TS-Tabu search)



Búsqueda Tabú (TS-Tabu search)

- Utiliza una búsqueda local con memoria a corto plazo que le permite “escapar” de mínimos locales y evita ciclos.
- La memoria de corto plazo esta representada por la lista Tabú la cual registra las últimas soluciones “visitadas” e impide volver a ellas en los próximos movimientos.
- La lista Tabú se actualiza normalmente en forma FIFO.

Búsqueda Tabú (2)

Algorithm TS

1: $s \leftarrow$ GenerateInitialSolution()
2: $TabuList \leftarrow \emptyset$
3: **while** termination conditions not met **do**
4: $s \leftarrow$ BestImprovement($s, N(s) \setminus TabuList$)
5: Update($TabuList$)
6: **end while**

Figure: TS - Simple Tabu search algorithm

Búsqueda Tabú (3)

La lista tabú

La lista tabú

- El largo de la lista tabú controla la memoria del proceso de búsqueda. Una lista tabú corta, controla áreas reducidas del espacio de búsqueda y una larga, fuerza a una búsqueda en áreas mayores.
- El largo de la lista puede cambiar a lo largo del proceso de búsqueda.

Búsqueda Tabú (4)

implementación más realista...

Implementación

- Por razones de eficiencia, no se guarda la solución completa en la lista tabú, sino una parte de sus atributos. Se define una lista tabú de condiciones.
- En este proceso se pierde información y buenas soluciones pueden ser excluidas del conjunto permitido. Para reducir este problema, se define un criterio de aspiración que permitiría a una solución estar dentro del conjunto de soluciones permitidas aún cuando figure en la lista tabú.

Búsqueda Tabú (5)

implementación más realista...

Algorithm TS

1: $s \leftarrow$ GenerateInitialSolution()
2: InitializeTabuList($TL_1, \ldots, TL_r$)
3: $k \leftarrow 0$
4: **while** termination conditions not met **do**
5: $AllowedSet(s, k) \leftarrow \{z \in N(s) \setminus$ no tabu condition is violated or
: at least one aspiration condition is satisfied$\}$
6: $s \leftarrow$ BestImprovement($s, AllowedSet(s, k)$)
7: UpdateTabuListAndAspirationConditions()
8: $k \leftarrow k + 1$
9: **end while**

Figure: TS - Tabu search algorithm

Búsqueda Tabú (6)

implementación realista para el TSP [Jay]. . .

- Solution Representation, Initial Solution, and Neighborhood for TS are same as that for LS and SA described before.
- Tabu List: To prevent the process from cycling in a small set of solutions, some attribute of recently visited solutions is stored in a Tabu List, which prevents their occurrence for a limited period.
  - For our problem, the attribute used is a pair of nodes that have been exchanged recently.
- Aspiration criterion: Tabus may sometimes be too powerful: they may prohibit attractive moves, even when there is no danger of cycling, or they may lead to an overall stagnation of the searching process.
  - It may, therefore, become necessary to revoke tabus at times.
  - The criterion used in TSP is to allow a move, even if it is tabu, if it results in a solution with an objective value better than that of the current best-known solution.
- Termination criteria: The algorithm terminates if a pre-specified number of iterations is reached.

Contenido

2 Heurísticas
- Definición
- Ejemplo

3 Metaheurísticas
- General
- Búsqueda local (LS-Local Search)
- Recocido simulado (SA-Simulated Annealing)
- Búsqueda Tabú (TS-Tabu search)
- GRASP-Greedy Randomized Adaptive Search Procedure
- Computación evolutiva (EC-Evolutionary Computation)
- Colonias de Hormigas (ANT-Ant colony optimization)

4 Resumen
- Esquema de clasificación
- Intensificación y Diversificación
- ¿Preguntas?

GRASP - (Greedy Randomized Adaptive Search Procedure)

GRASP

- Clasificado como método de exploración local (ELS-Explorative local search)
- GRASP (Greedy Randomized Adaptive Search Procedure) es una metaheurística que combina procedimientos constructivos y de búsqueda local.
- GRASP es un procedimiento iterativo en dos fases:
  - una de construcción de la solución (heurística)
  - y otra de mejora.
- Heurística estática: asigna el score a los elementos solamente antes de comenzar la construcción.
- En GRASP, la construcción de la solución se caracteriza por ser dinámica y aleatoria.
- Los valores de la heurística son actualizados en cada iteración.

GRASP (2)

Algoritmo

Algorithm GRASP

```
1:  while termination conditions not met do
2:      s ← ConstructGreedyRandomizedSolution()
3:      ApplyLocalSearch(s)
4:      MemorizedBestFoundSolution()
5:  end while
```

Figure: GRASP - Greedy Randomized Adaptive Search Procedure

GRASP (3)

Construcción de una solución

Algorithm GRASP Construction

```
1:  s ← ∅    /* s denotes a partial solution in this case*/
2:  while termination conditions not met do
3:      RCL ← GenerateRestrictedCandidateList()
4:      x ← SelectElementAtRandom(RCL)
5:      s ← s ∪ {x}
6:      UpdateGreedyFunction(s)
7:  end while
```

Figure: Greedy Randomized Solution Construction

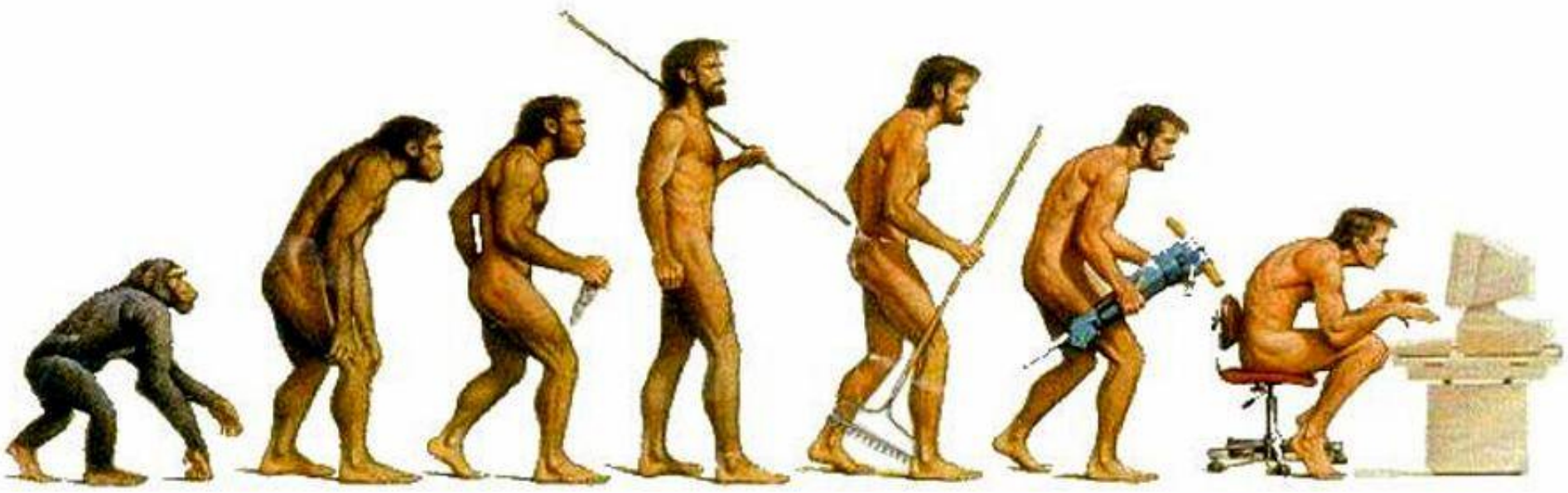
Contenido

2 Heurísticas
- Definición
- Ejemplo

3 Metaheurísticas
- General
- Búsqueda local (LS-Local Search)
- Recocido simulado (SA-Simulated Annealing)
- Búsqueda Tabú (TS-Tabu search)
- GRASP-Greedy Randomized Adaptive Search Procedure
- Computación evolutiva (EC-Evolutionary Computation)
- Colonias de Hormigas (ANT-Ant colony optimization)

4 Resumen
- Esquema de clasificación
- Intensificación y Diversificación
- ¿Preguntas?

Computación evolutiva (EC-Evolutionary Computation)



Computación evolutiva (EC-Evolutionary Computation)

- Amplio espectro de técnicas heurísticas que funcionan emulando mecanismos de evolución natural.
- Trabaja sobre una población de individuos o conjunto de soluciones, que evoluciona utilizando mecanismos de selección y construcción de nuevas soluciones candidatas mediante recombinación de características de las soluciones seleccionadas.

Etapas del mecanismo evolutivo:

1. Evaluación: función de fitness.
2. Selección : de individuos adecuados (de acuerdo al fitness) para la aplicación de operadores evolutivos.
3. Aplicación de operadores evolutivos.
4. Reemplazo o recambio generacional.

Computación evolutiva (2)

- Proceso evolutivo: decide en cada iteración qué individuos entrarán en la próxima población.
- Operadores evolutivos: determinan el modo en que el algoritmo explora el espacio de búsqueda.

Algoritmos genéticos:

- Cruzamiento es el operador principal, mientras que la mutación es un operador secundario, solamente para agregar mayor diversidad al mecanismo de exploración
- Grado de adaptación de un individuo se evalúa a través de la función de fitness

Algoritmos Genéticos (GA) - Genetic algorithm

Algorithm GA

1: $P \leftarrow$ GenerateInitialPopulation()
2: Evaluate(P)
3: **while** termination conditions not met **do**
4: $P' \leftarrow$ Recombine(P)
5: $P'' \leftarrow$ Mutate(P')
6: Evaluate(P'')
7: $P \leftarrow$ Select($P \cup P''$)
8: **end while**

Figure: GA - Genetic Algorithm

Contenido

2 Heurísticas
- Definición
- Ejemplo

3 Metaheurísticas
- General
- Búsqueda local (LS-Local Search)
- Recocido simulado (SA-Simulated Annealing)
- Búsqueda Tabú (TS-Tabu search)
- GRASP-Greedy Randomized Adaptive Search Procedure
- Computación evolutiva (EC-Evolutionary Computation)
- Colonias de Hormigas (ANT-Ant colony optimization)

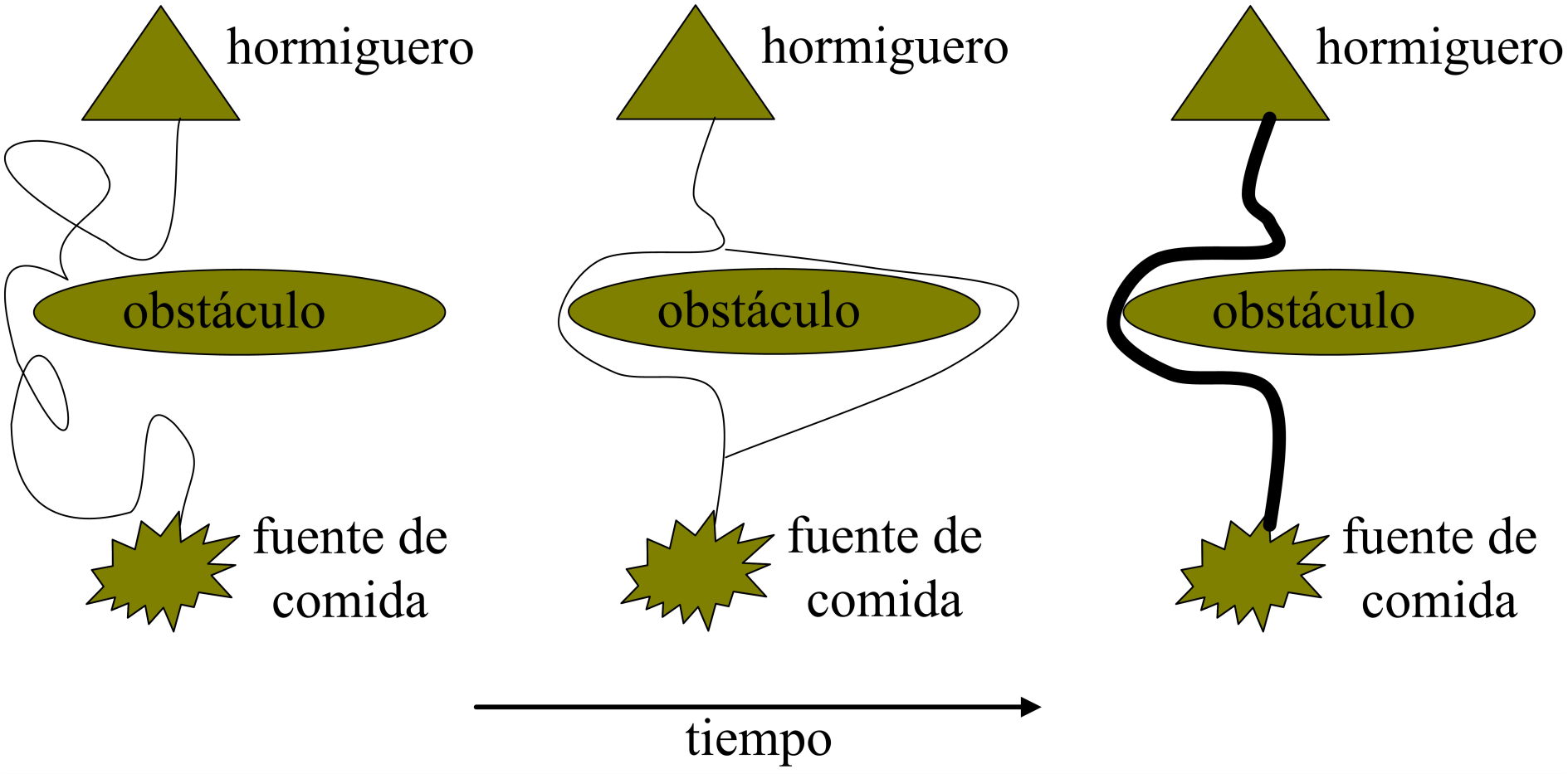
4 Resumen
- Esquema de clasificación
- Intensificación y Diversificación
- ¿Preguntas?

Colonias de Hormigas (ANT-Ant colony optimization)

Colonias de Hormigas (ANT-Ant colony optimization)

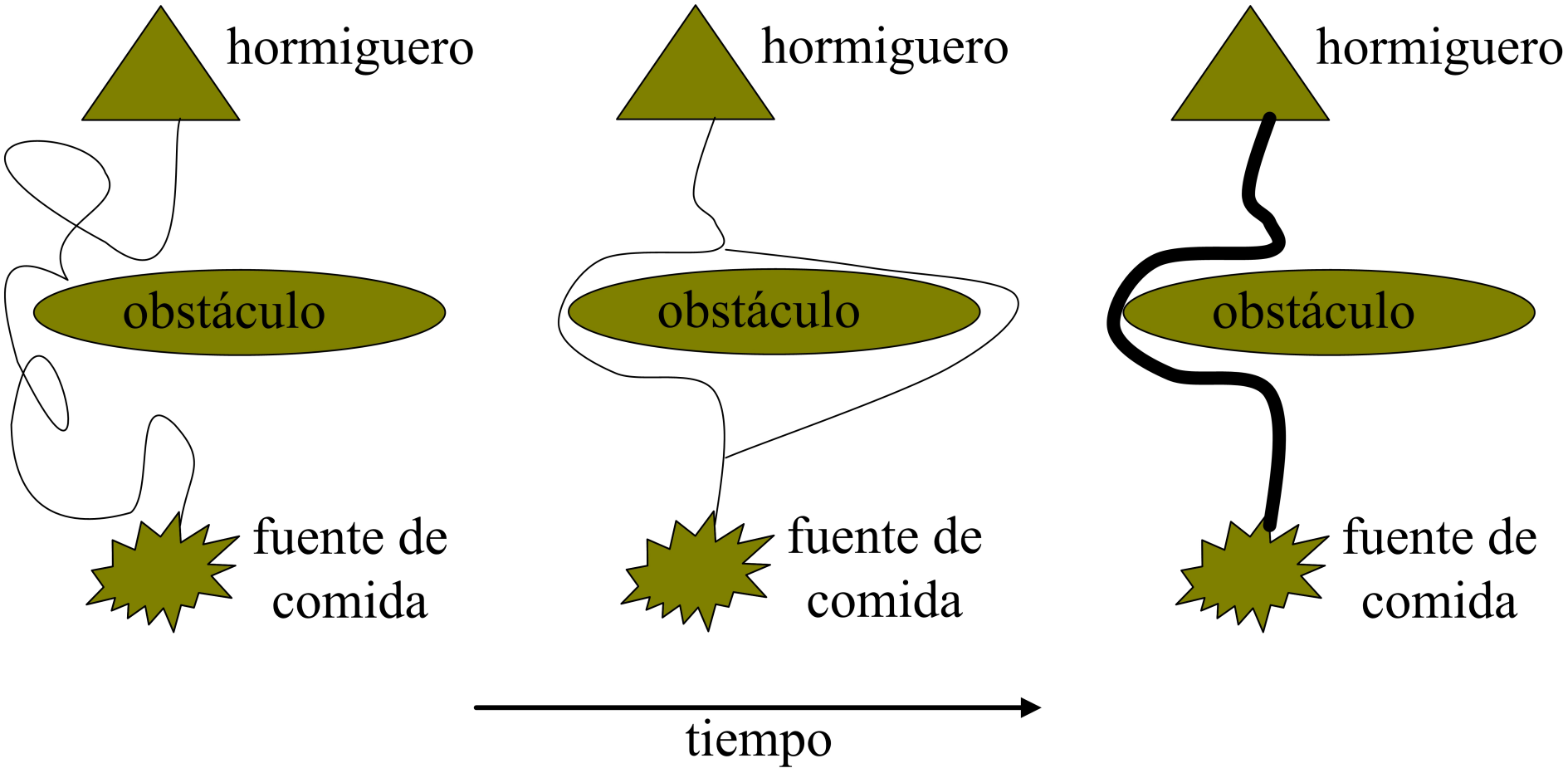
- Heurísticas que se basan en imitar el comportamiento de las hormigas.
- Si bien cada hormiga tiene individualmente capacidades básicas, la colonia en conjunto logra un comportamiento inteligente.
- A pesar de ser insectos casi ciegos logran encontrar el camino más corto entre el hormiguero y una fuente de comida y regresar, en base a una comunicación entre ellas de origen químico que se potencia con la presencia simultánea de muchas hormigas.
- Comunicación en base a feromonas, que dejan un “rastro” que sirve de referencia a otras hormigas.

Colonias de Hormigas (2)



Debido a la evaporación constante de la feromona, con el tiempo se elige el camino más corto.

Colonias de Hormigas (2)



Debido a la evaporación constante de la feromona, con el tiempo se elige el camino más corto.

Colonias de Hormigas (3)

Algoritmo

Algorithm ANT

```
1:  InitializePheromoneValues()
2:  while termination conditions not met do
3:      for all ants a ∈ A do
4:      s_a ← ConstructSolution(τ, η)
5:      end for
6:      UpdateOnlineDelayedPheromoneUpdate()
7:  end while
```

Figure: ANT-Ant System

Colonias de Hormigas (4)

Algoritmo mejorado

Algorithm ANT

```
1:   while termination conditions not met do
2:       ScheduleActivities
3:           ManageAntsActivity()
4:           EvaporatePheromone()
5:           DaemonActions()     /* optional */
6:       end ScheduleActivities
7:   end while
```

Figure: ANT-Ant colony optimization

Contenido

2 Heurísticas
- Definición
- Ejemplo

3 Metaheurísticas
- General
- Búsqueda local (LS-Local Search)
- Recocido simulado (SA-Simulated Annealing)
- Búsqueda Tabú (TS-Tabu search)
- GRASP-Greedy Randomized Adaptive Search Procedure
- Computación evolutiva (EC-Evolutionary Computation)
- Colonias de Hormigas (ANT-Ant colony optimization)

4 Resumen
- Esquema de clasificación
- Intensificación y Diversificación
- ¿Preguntas?

Contenido

2 Heurísticas
- Definición
- Ejemplo

3 Metaheurísticas
- General
- Búsqueda local (LS-Local Search)
- Recocido simulado (SA-Simulated Annealing)
- Búsqueda Tabú (TS-Tabu search)
- GRASP-Greedy Randomized Adaptive Search Procedure
- Computación evolutiva (EC-Evolutionary Computation)
- Colonias de Hormigas (ANT-Ant colony optimization)

4 Resumen
- Esquema de clasificación
- Intensificación y Diversificación
- ¿Preguntas?

Esquema de clasificación X/Y/Z

Notación de clasificación:

- Propuesto por Laguna (similar al esquema para filas de espera propuesto por Kendall):
  - X = A (adaptativo, con memoria) o M (sin memoria)
  - Y= N (búsqueda determinística) o S (muestreo aleatorio)
  - Z = 1 (basado en una solución) o P (basado en poblaciones).

Clasificación para las Metaheurísticas que estudiamos

- Búsqueda local y variantes (M/N/1).
- Recocido Simulado (M/S/1).
- Búsqueda Tabú (A/N/1).
- GRASP (A/S/1).
- Algoritmos Genéticos (A/S/P).
- Colonias de Hormigas (A/S/P).

Esquema de clasificación X/Y/Z

Notación de clasificación:

- Propuesto por Laguna (similar al esquema para filas de espera propuesto por Kendall):
  - X = A (adaptativo, con memoria) o M (sin memoria)
  - Y= N (búsqueda determinística) o S (muestreo aleatorio)
  - Z = 1 (basado en una solución) o P (basado en poblaciones).

Clasificación para las Metaheurísticas que estudiamos

- Búsqueda local y variantes (M/N/1).
- Recocido Simulado (M/S/1).
- Búsqueda Tabú (A/N/1).
- GRASP (A/S/1).
- Algoritmos Genéticos (A/S/P).
- Colonias de Hormigas (A/S/P).

Contenido

2 Heurísticas
- Definición
- Ejemplo

3 Metaheurísticas
- General
- Búsqueda local (LS-Local Search)
- Recocido simulado (SA-Simulated Annealing)
- Búsqueda Tabú (TS-Tabu search)
- GRASP-Greedy Randomized Adaptive Search Procedure
- Computación evolutiva (EC-Evolutionary Computation)
- Colonias de Hormigas (ANT-Ant colony optimization)

4 Resumen
- Esquema de clasificación
- Intensificación y Diversificación
- ¿Preguntas?

Intensificación y Diversificación

Intensificación y Diversificación

- Cada metaheurística utiliza una estrategia distinta para explorar el espacio
- La estrategias de búsqueda se diseñan balanceando estática o dinámicamente el aprovechamiento de la experiencia acumulada de búsqueda (intensificación) y la exploración del espacio de búsqueda (diversificación)

Intensificación y Diversificación (2)

Todas las metaheurísticas presentan un compromiso entre dos mecanismos:

- Diversificación $\sim$ exploración global
- Intensificación $\sim$ exploración local

Mecanismos de intensificación/diversificación:

- básicos o intrínsecos,
- especiales o estratégicos.

Intensificación/Diversificación (3)

Básicos

Recocido simulado (M/S/1):

- Intensificación: búsqueda local
- Balance dinámico controlado a través del parámetro T.
- Inicialmente alta diversificación y baja intensificación, al final del algoritmo se invierte relación

Búsqueda tabú (A/N/1):

- Intensificación: búsqueda local y función de aspiración
- Diversificación: a través de la o las listas tabú.
- Balance estático: a través del tamaño de la lista tabú. Listas tabú cortas: gran intensificación poca diversificación, listas largas a la inversa.

Intensificación/Diversificación (4)

Básicos

GRASP (A/S/1):

- Intensificación: búsqueda local en la fase de mejora y largo de la lista restringida de candidatos (RCL)
- Diversificación: implementada a través de la lista restringida de candidatos
- Balance estático: determinado por el largo de la lista restringida de candidatos
- Si la RCL es de largo 1, alta intensificación, a medida que aumenta su largo, aumenta la divesificación ya que la solución inicial para la búsqueda local se extrae randómicamente de la RCL

Intensificación/Diversificación (5)

Básicos

Computación evolutiva (A/S/P):

- Intensificación: a través del operador de selección que concentra la búsqueda en algunas áreas.
- Diversificación: en forma natural, a través de las poblaciones, a través de operadores: recombinación y modificación.
- Balance dinámico: Inicialmente alta diversificación y baja intensificación, al final del algoritmo se invierte relación.

Intensificación/Diversificación (6)

Básicos

Colonias de hormigas (A/S/P):

- Intensificación: a través de la feromona y las reglas de actualización que “refuerzan” las buenas soluciones.
- Diversificación: implementado a través del mecanismo probabilístico de construcción de soluciones.
- Balance dinámico: Inicialmente alta diversificación y baja intensificación, al final del algoritmo se invierte relación

Intensificación/Diversificación (7)

Ejemplo de Especiales

- SA: técnicas de recalentado y posterior enfriamiento (enfriamiento no monótono)
- TS: cambios dinámicos en el largo de la lista tabú (Reactive Tabu Search). El largo de la lista varía según las propiedades de la trayectoria en el espacio de búsqueda.
- EC: no son muy comunes estos mecanismos. Algunas aplicaciones cambian el operador de selección cuando el sistema esta próximo a la convergencia (equivalente a reiniciar el algoritmo).
- ANT: Al igual que los EC, no es usual interrumpir el proceso de convergencia. En los Ant Colony System , se usan mecanismos extra de diversificación durante la fase de construcción de la solución.

Intensificación/Diversificación (8)

Básicos - Resumen

Table: Intensificación/Diversificación para cada Metaheurística

	Intensificación	Diversificación
SA	búsqueda local básica	movimientos que empeoran solución (T)
TS	búsqueda local básica	lista tabú (memoria corto plazo)
ELS (GRASP)	búsqueda local básica	criterio de aceptación y perturbación
EC (AG)	selección	operaciones de modificación (ej. mutación)
ANT	reglas de actualización de la feromona	reglas de construcción probabilística

Resumen

- Los métodos basados en poblaciones son mejores en identificar posibles “buenas áreas” en el espacio de búsqueda (permiten dar “pasos largos”), que los métodos de trayectoria los cuales son mejores para explorar localmente las áreas.
- Las heurísticas “híbridas” tratan de aprovechar las bondades de ambos tipos de métodos y han tenido éxito en algunas aplicaciones, es un tipo de técnica a continuar mejorando en el futuro.

Resumen (2)

- El campo de las metaheurísticas a:
  - sido aplicado en una variedad enorme de problemas de optimización
  - demostrado buena performance en muchos problemas reales
- Las metaheurísticas son considerablemente más genericas que otros métodos:
  - por ejemplo: pueden ser aplicadas cuando no se conoce completamente la función objetivo.

Contenido

2 Heurísticas
- Definición
- Ejemplo

3 Metaheurísticas
- General
- Búsqueda local (LS-Local Search)
- Recocido simulado (SA-Simulated Annealing)
- Búsqueda Tabú (TS-Tabu search)
- GRASP-Greedy Randomized Adaptive Search Procedure
- Computación evolutiva (EC-Evolutionary Computation)
- Colonias de Hormigas (ANT-Ant colony optimization)

4 Resumen
- Esquema de clasificación
- Intensificación y Diversificación
- ¿Preguntas?

¿Preguntas?

Gracias!

Por su atención.

Bibliography

Sachin Jayaswal.
A Comparative Study of Tabu Search and Simulated Annealing for Traveling Salesman Problem.
Technical report.

C. E. Miller, A. W. Tucker, and R. A. Zemlin.
Integer programming formulations and traveling salesman problems.
ACM, 7:326–329, 1960.