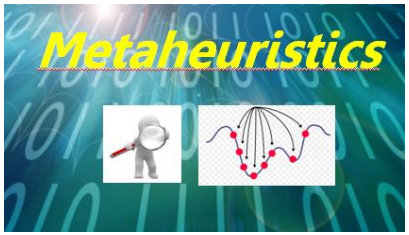
METAHEURÍSTICAS
2020 - 2021



- Tema 1. Introducción a las Metaheurísticas
- Tema 2. Modelos de Búsqueda: Entornos y Trayectorias vs Poblaciones
- Tema 3. Metaheurísticas Basadas en Poblaciones
- Tema 4: Algoritmos Meméticos
- Tema 5. Metaheurísticas Basadas en Trayectorias
- Tema 6. Metaheurísticas Basadas en Adaptación Social
- Tema 7. Aspectos Avanzados en Metaheurísticas
- Tema 8. Metaheurísticas Paralelas

Objetivos

- Analizar las características de las técnicas basadas en búsqueda local y las estructuras del entorno.
- Entender el funcionamiento y estructura general de los algoritmos de optimización simples basados en búsquedas locales.
- Ante un problema abordable mediante este tipo de técnica determinar la representación y operadores de vecindario más acordes.
- Entender el concepto de algoritmo basado en poblaciones.

ALGORÍTMICA

TEMA 2. Modelos de Búsqueda: Entornos y Trayectorias vs Poblaciones

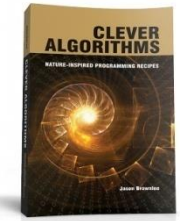
1. Introducción. Busqueda basada en Trajectorias vs Búsqueda basada en Poblaciones
2. Búsqueda Aleatoria *versus* Búsqueda Local
3. Métodos de Búsqueda Local Básicos

- **E.-G. Talbi. Metaheuristics. From design to implementation. Wiley, 2009**
- **J. Brownlee. Clever Algorithms (Nature-Inspired Programming Recipes). 2012, Brownlee.**

http://cleveralgorithms.com/

https://github.com/clever-algorithms/CleverAlgorithms

All algorithm descriptions include a working implementation of the algorithm in Ruby.



1. INTRODUCCIÓN

1.1. Término *"LOCAL"* . Estructura de entorno. Proceso de búsqueda

***1.2.* Búsqueda basada en Poblaciones**

1.1. Término *"LOCAL"* . Estructura de entorno. Proceso de búsqueda

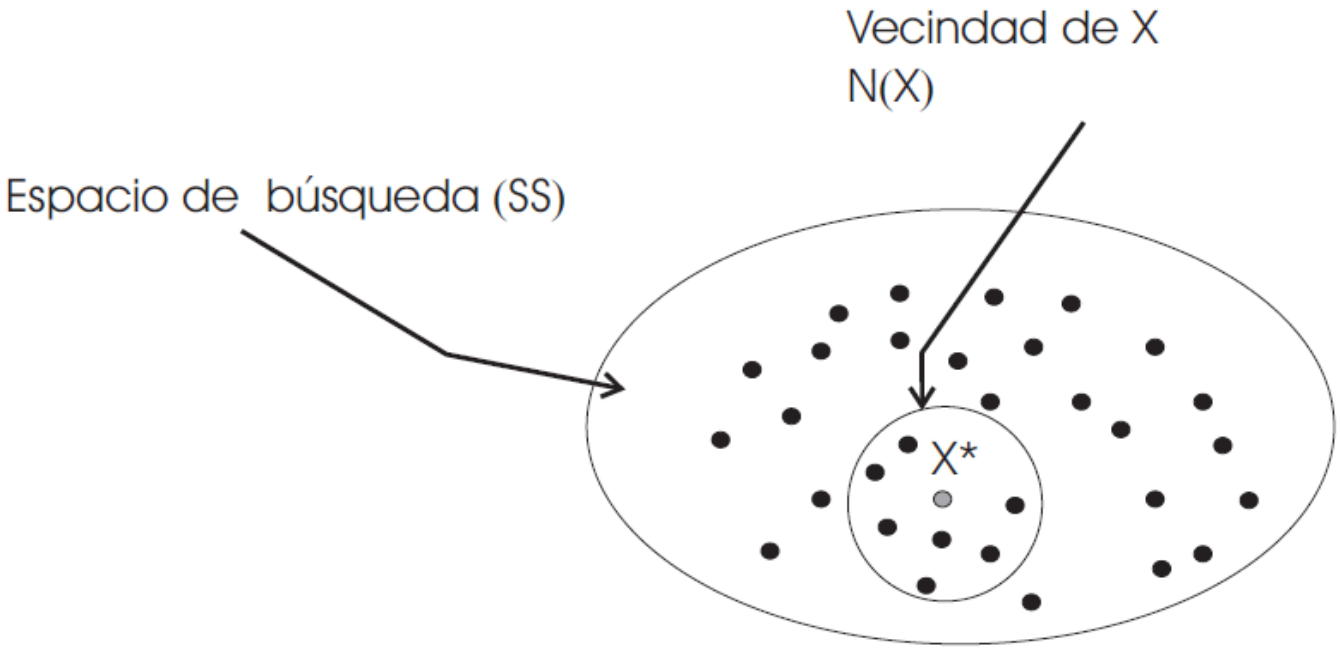
El término *"local"* se utiliza frecuentemente en los estudios teóricos y prácticos de las metaheurísticas de búsqueda.

Se asocia al uso de estructuras de entorno, reflejando el concepto de proximidad o vecindad entre las soluciones alternativas del problema.

1. INTRODUCCIÓN

1.1. Término *"LOCAL"*

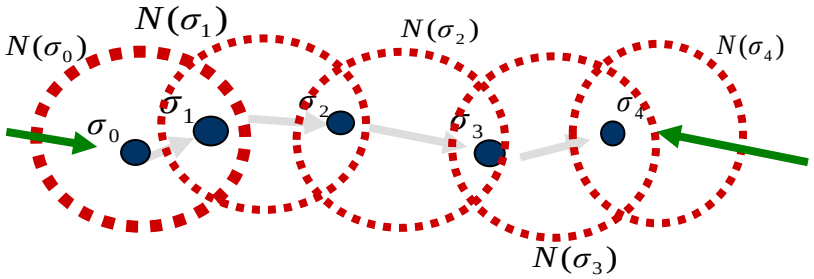
Todas las soluciones incluidas en el entorno de la solución actual, que viene delimitado por un operador de generación de soluciones, se denominan soluciones vecinas.



1. INTRODUCCIÓN

Los algoritmos basados en esta estrategia efectúan un estudio local del espacio de búsqueda, puesto que analizan el entorno de la solución actual para decidir cómo continuar el recorrido de la búsqueda.

Solución inicial



Óptimo local/global

Una búsqueda local es un proceso que, dada la solución actual en la que se encuentra el recorrido, selecciona iterativamente una solución de su entorno para continuar la búsqueda.

1. INTRODUCCIÓN

Una búsqueda local es un proceso que, dada la solución actual en la que se encuentra el recorrido, selecciona iterativamente una solución de su entorno para continuar la búsqueda.

Basta con diseñar la estructura de entorno para obtener un modelo genérico de algoritmo de búsqueda.

DESCRIPCIÓN

- **Se fija una codificación para las soluciones.**
- **Se define un operador de generación de vecino y, en consecuencia, se fija una estructura de entorno para las mismas.**
- **Se escoge una solución del entorno de la solución actual hasta que se satisfaga el criterio de parada.**

1. INTRODUCCIÓN

ELEMENTOS BÁSICOS EN EL PROCESO DE BÚSQUEDA

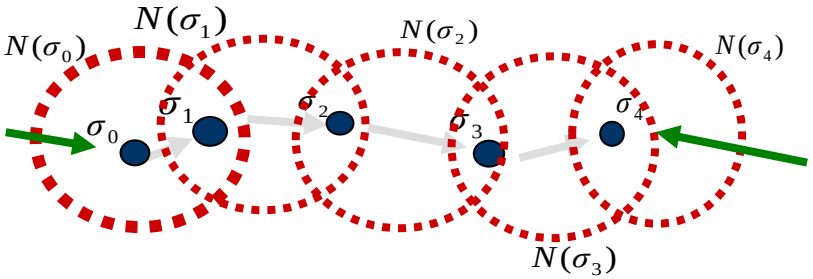
- **Proceso de elección de la solución inicial.**

- **Operador de vecino: Proceso de selección de solución/generación de una solución vecina:**

  S → S', S' ∈ E(S) (también notado N(S)).

- **Proceso de aceptación de solución vecina como solución actual.**

Solución inicial



Óptimo local/global

1. INTRODUCCIÓN

Procedimiento Búsqueda por Entornos
Inicio
GENERA(Solución Inicial)
Solución Actual ← Solución Inicial;
Mejor Solución ← Solución Actual;
Repetir
Solución Vecina ← GENERA_VECINO(Solución Actual);
Si Acepta(Solución Vecina)
entonces Solución Actual ← Solución Vecina;
Si Objetivo(Solución Actual) **es mejor que** Objetivo(Mejor Solución)
entonces Mejor Solución ← Solución Actual;
Hasta (Criterio de parada);
DEVOLVER (Mejor Solución);
Fin

1. INTRODUCCIÓN

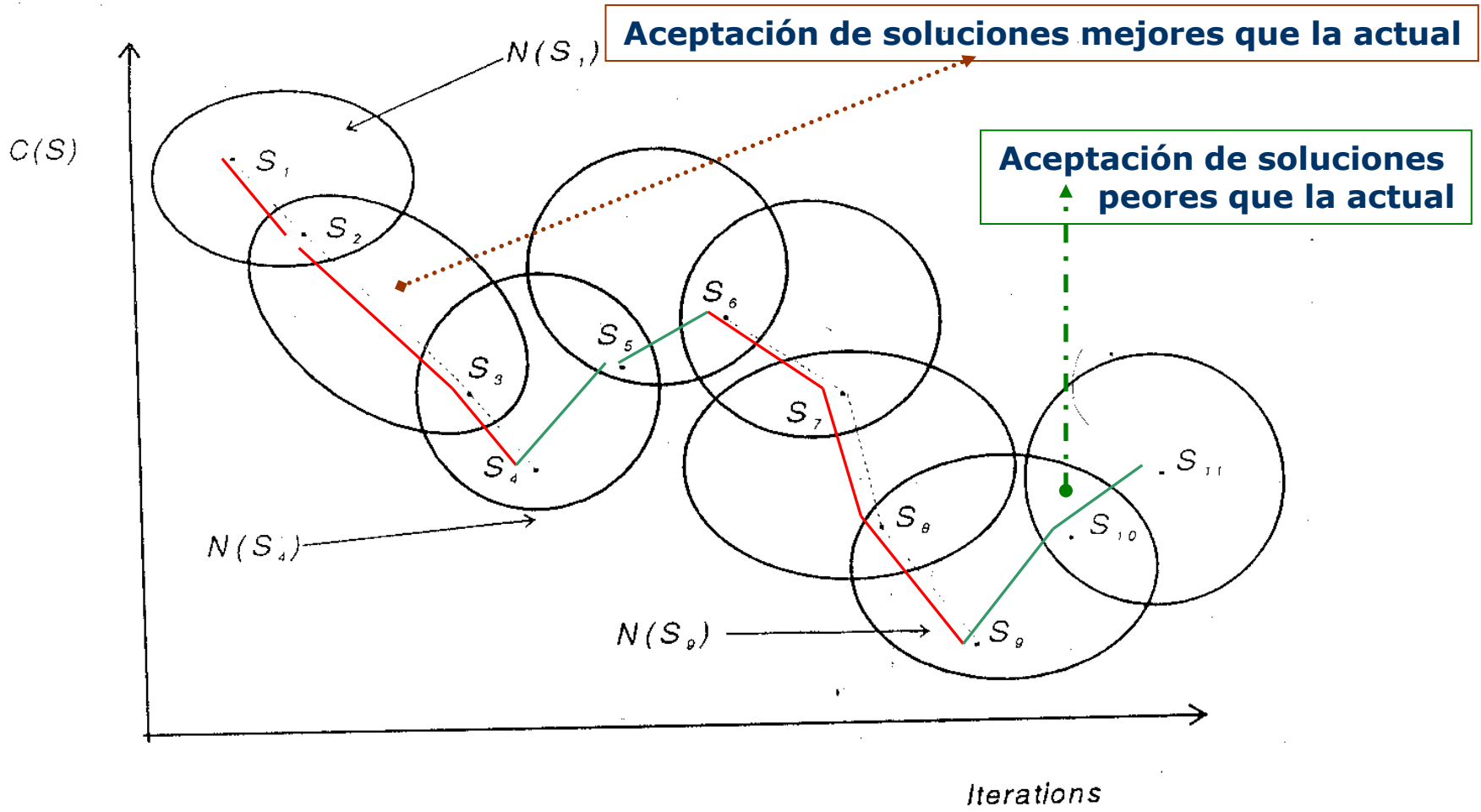
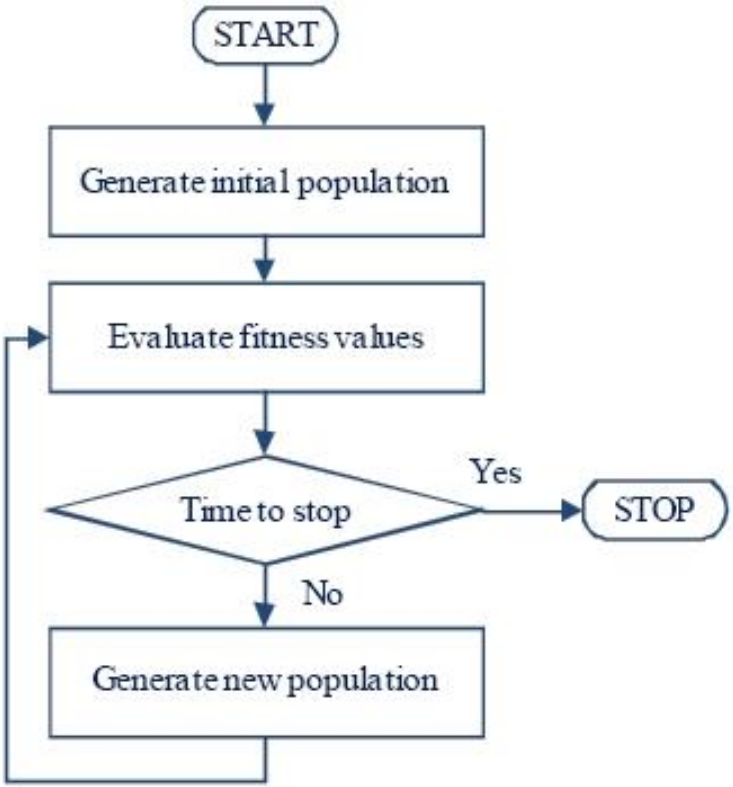


Figura que muestra una trayectoria de búsqueda basada en entornos

1. INTRODUCCIÓN

1.2. Búsqueda basada en Poblaciones

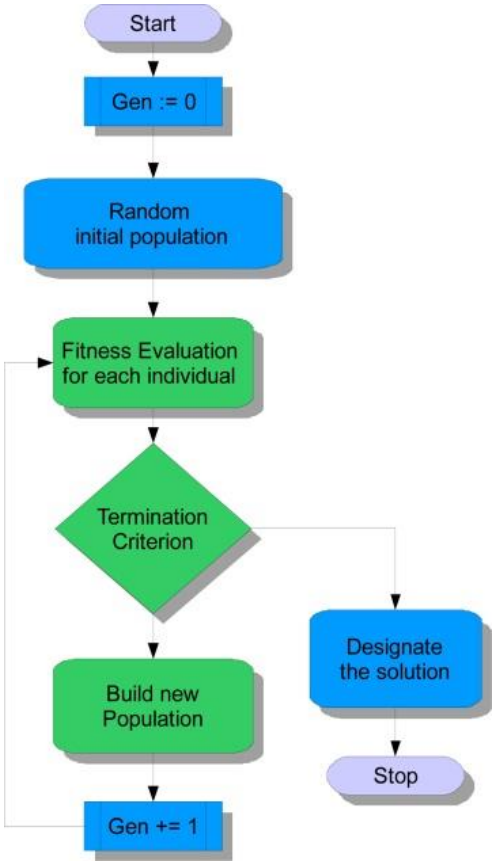


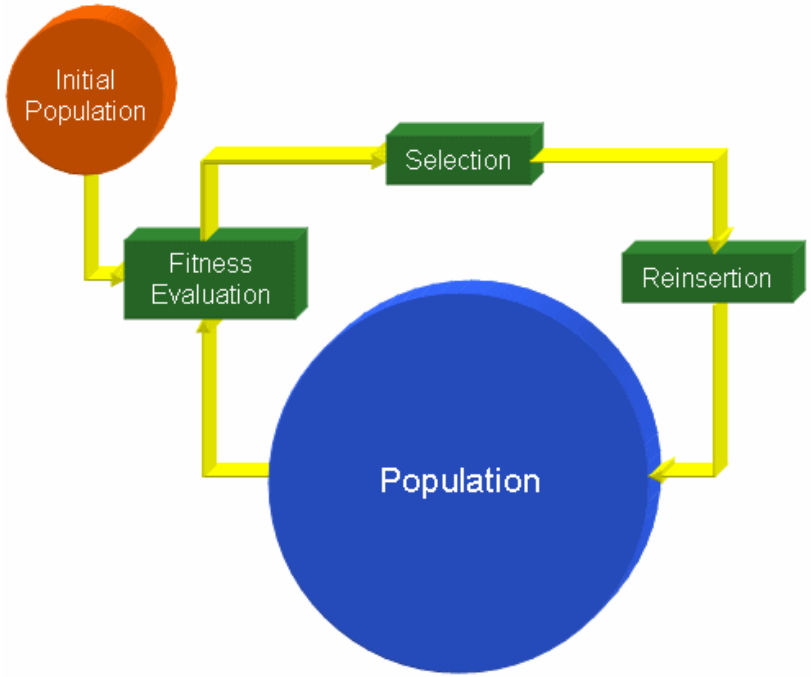
En un modelo de poblaciones de soluciones debemos definir cómo generar nuevas poblaciones.

Las nueva población debería estar asociada a la anterior y tener en cuenta la calidad de las anteriores soluciones.

1. INTRODUCCIÓN

1.2. Búsqueda basada en Poblaciones

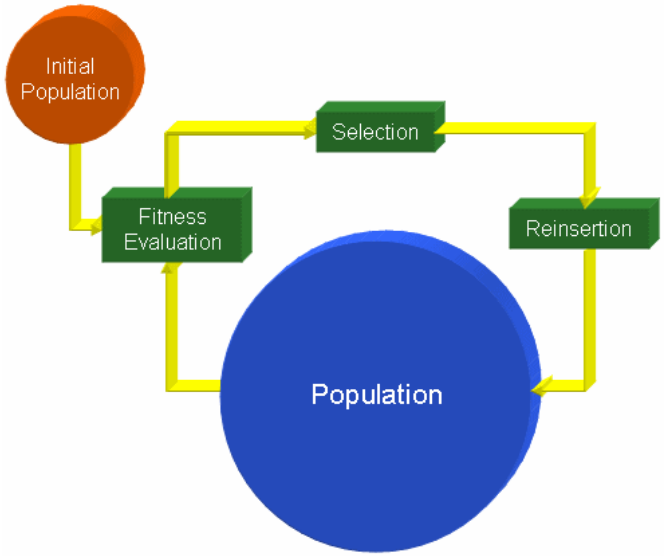


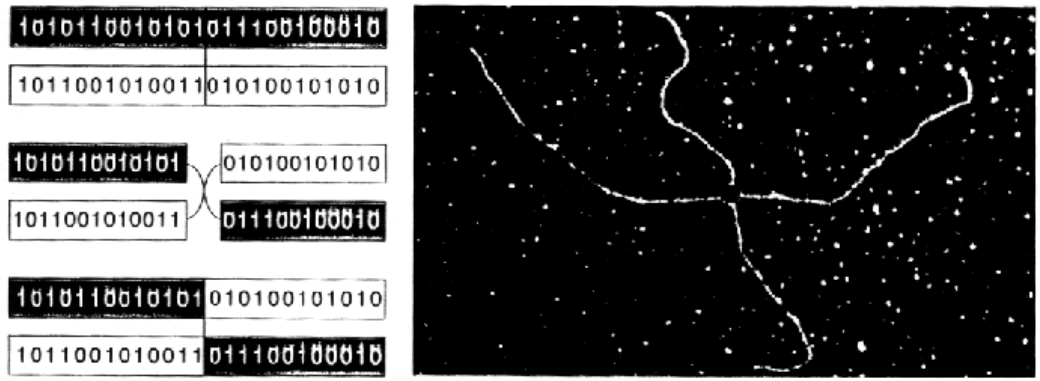


Se podría tener un modelo que seleccione soluciones, opere con ellas y puedan ser reinsertadas en la población dando lugar a una nueva población.

1. INTRODUCCIÓN

1.2. Búsqueda basada en Poblaciones



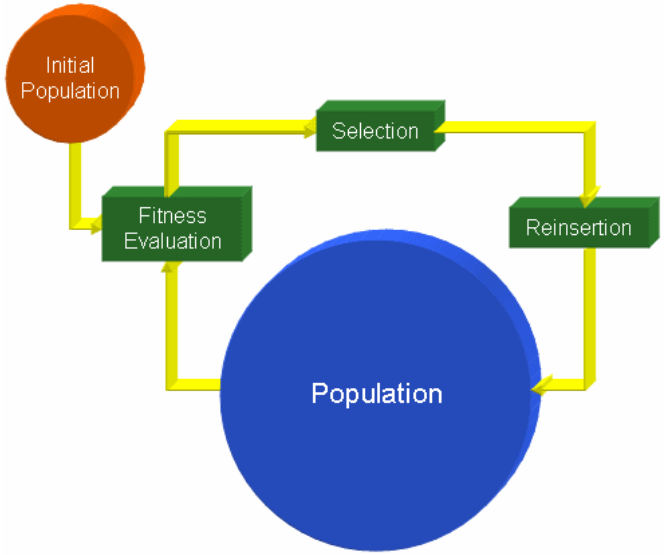


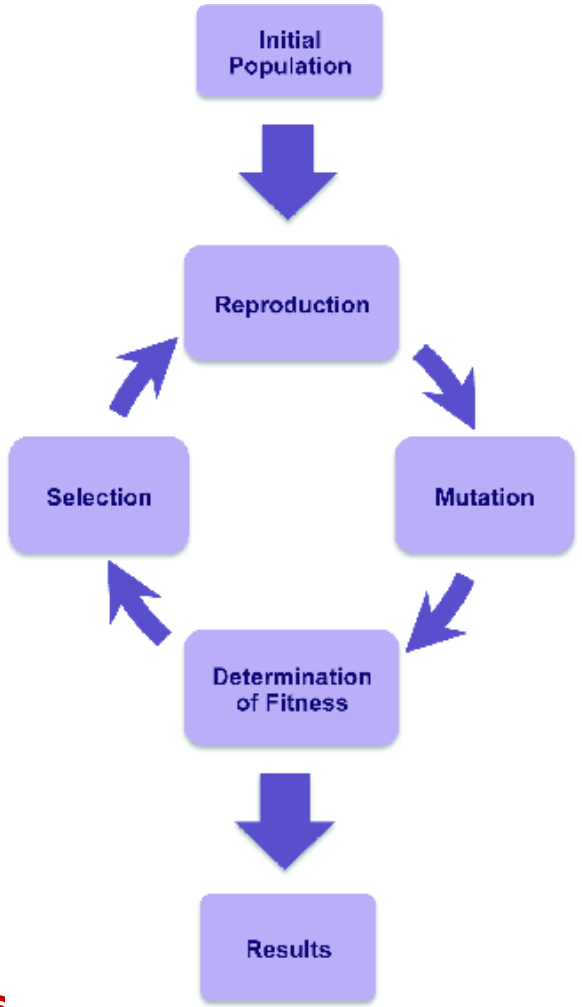
CROSSOVER is the fundamental mechanism of genetic rearrangement for both real organisms and genetic algorithms. Chromosomes line up and then swap the portions of their genetic code beyond the crossover point.

Se podría tener un modelo que seleccione soluciones, opere con ellas y puedan ser reinsertadas en la población dando lugar a una nueva población. Podemos imitar a la genética, cómo se combinan cromosomas (algoritmos genéticos)

1. INTRODUCCIÓN

1.2. Búsqueda basada en Poblaciones

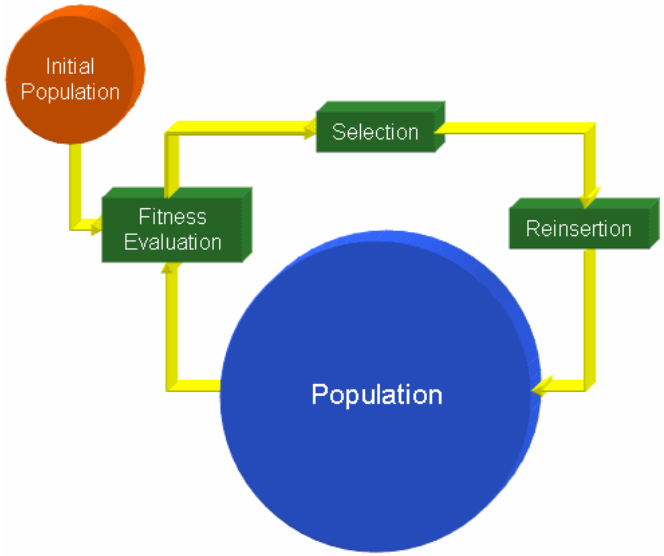




Se podría tener un modelo que seleccione soluciones, opere con ellas y puedan ser reinsertadas en la población dando lugar a una nueva población.
Podemos imitar a la evolución de especies y la genética, cómo se combinan cromosomas (algoritmos genéticos)

1. INTRODUCCIÓN

1.2. Búsqueda basada en Poblaciones



¿Otras propuestas de obtención de poblaciones?

Se podría tener un modelo que seleccione soluciones, opere con ellas y puedan ser reinsertadas en la población dando lugar a una nueva población. ~~**Podemos imitar a la evolución de especies y la genética, cómo se combinan cromosomas (algoritmos genéticos)**~~

Advances in Engineering Software
Volume 69, March 2014, Pages 46–61

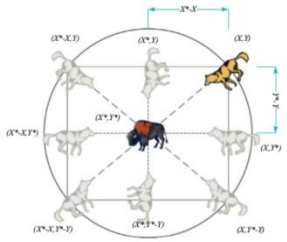
1. INTRODUCCIÓN

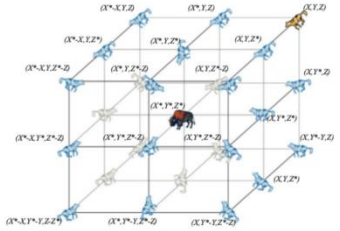
Grey Wolf Optimizer

Seyedali Mirjalili[a], Seyed Mohammad Mirjalili[b], Andrew Lewis[a]

¿Otras propuestas de obtención de poblaciones?

Grey Wolf Optimization





1. INTRODUCCIÓN

¿Otras propuestas de Metaheurísticas?

Published: 05 July 2020

Comprehensive Taxonomies of Nature- and Bio-inspired Optimization: Inspiration Versus Algorithmic Behavior, Critical Analysis Recommendations

Daniel Molina, Javier Poyatos, Javier Del Ser, Salvador García, Amir Hussain & Francisco Herrera

Cognitive Computation **12**, 897–939(2020) | Cite this article

Cornell University

arXiv.org > cs > arXiv:2002.08136

Search... Help | Advanc

Computer Science > Artificial Intelligence

[Submitted on 19 Feb 2020 (v1), last revised 20 Feb 2020 (this version, v2)]

Comprehensive Taxonomies of Nature- and Bio-inspired Optimization: Inspiration versus Algorithmic Behavior, Critical Analysis and Recommendations

Daniel Molina, Javier Poyatos, Javier Del Ser, Salvador García, Amir Hussain, Francisco Herrera

In recent years, a great variety of nature- and bio-inspired algorithms has been reported in the literature. This algorithmic family simulates different biological processes observed in Nature in order to efficiently address complex optimization problems. In the last years the number of bio-inspired optimization approaches in literature has grown considerably, reaching unprecedented levels that dark the future prospects of this field of research. This paper addresses this problem by proposing two comprehensive, principle-based taxonomies that allow researchers to organize existing and future algorithmic developments into well-defined categories, considering two different criteria: the source of inspiration and the behavior of each algorithm. Using these taxonomies we review more than three hundred publications dealing with nature-inspired and bio-inspired algorithms, and proposals falling within each of these categories are examined, leading to a critical summary of design trends and similarities between them, and the identification of the most similar classical algorithm for each reviewed paper. From our analysis we conclude that a poor relationship is often found between the natural inspiration of an algorithm and its behavior. Furthermore, similarities in terms of behavior between different algorithms are greater than what is claimed in their public disclosure: specifically, we show that more than one-third of the reviewed bio-inspired solvers are versions of classical algorithms. Grounded on the conclusions of our critical analysis, we give several recommendations and points of improvement for better methodological practices in this active and growing research field.

Comments: 76 pages, 6 figures
Subjects: **Artificial Intelligence (cs.AI)**
ACM classes: I.2.8
Cite as: arXiv:2002.08136 **[cs.AI]**
(or arXiv:2002.08136v2 **[cs.AI]** for this version)

Submission history

From: Daniel Molina Dr. [view email]
[v1] Wed, 19 Feb 2020 12:34:45 UTC (588 KB)
[v2] Thu, 20 Feb 2020 09:27:38 UTC (767 KB)

https://arxiv.org/abs/2002.08136

2. BÚSQUEDA ALEATORIA VERSUS BÚSQUEDA LOCAL

2.1. Búsqueda Aleatoria Pura

2.2. Búsqueda Aleatoria por Recorrido al Azar

En esta sección pretendemos estudiar el comportamiento de la búsqueda aleatoria, realizando un estudio de su eficacia y eficiencia, el cual justificará el uso de los procedimientos de búsqueda local.

2. BÚSQUEDA ALEATORIA VERSUS BÚSQUEDA LOCAL

2.1. Búsqueda Aleatoria Pura

- **Se elige aleatoriamente una muestra de soluciones del espacio de búsqueda y se devuelve la mejor.**

- **Se diría que el entorno de una solución es todo el espacio de búsqueda:**

E(s) = “todo el espacio de búsqueda”.

2. BÚSQUEDA ALEATORIA VERSUS BÚSQUEDA LOCAL

Procedimiento Búsqueda Aleatoria Pura

Inicio
GENERA(Solución Inicial)
Solución Actual ← Solución Inicial;
Mejor Solución ← Solución Actual;
Repetir
GENERA(Solución Actual); **%Generación Aleatoria**

Si Objetivo(Solución Actual) **es mejor que** Objetivo(Mejor Solución)
entonces Mejor Solución ← Solución Actual;
Hasta (Criterio de parada);
DEVOLVER (Mejor Solución);
Fin

2. BÚSQUEDA ALEATORIA VERSUS BÚSQUEDA LOCAL

EJEMPLO:

Búsqueda Aleatoria Pura para el Viajante de Comercio

Iteración	Solución
1	(1 2 4 3 8 5 9 6 7)
2	(9 6 4 7 8 5 1 2 3)
3	(2 4 1 5 8 3 9 7 6)
4	(4 7 5 1 8 3 2 6 9)
5	(7 6 9 5 8 3 1 2 4)
6	(8 3 7 2 1 5 7 6 9)
7	(2 5 1 3 9 8 4 6 7)
8	(1 4 2 3 8 5 6 9 7)
9	(3 4 2 1 5 8 7 9 6)
10	(7 4 9 3 8 5 6 2 1)

Una ejecución de la Búsqueda Aleatoria Pura

2. BÚSQUEDA ALEATORIA VERSUS BÚSQUEDA LOCAL

ESTUDIO TEÓRICO DE LA EFICIENCIA DE LA BÚSQUEDA ALEATORIA PURA

Si el problema tiene m soluciones y el óptimo es único, la probabilidad de que al generar aleatoriamente una solución se obtenga la óptima es 1/m.

Sean A_i los sucesos siguientes, que determinan la probabilidad absoluta de obtener el óptimo en la iteración i:

$$P(A_i) = 1/m \quad \forall i = 1, \ldots, n.$$

2. BÚSQUEDA ALEATORIA VERSUS BÚSQUEDA LOCAL

La probabilidad de obtener el óptimo en n iteraciones sería:

$$P(A_1 \cup A_2 \cup \ldots \cup A_n) =$$

$$\begin{aligned}
&= 1 - P((A_1 \cup A_2 \cup \cdots \cup A_n)^c) \\
&= 1 - P(A_1^c \cap A_2^c \cap \cdots \cap A_n^c) \\
&= 1 - P(A_1^c)P(A_2^c)\cdots P(A_n^c) \\
&= 1 - (1 - P(A_1))\,(1 - P(A_2))\cdots(1 - P(A_n)) \\
&= 1 - \left(1 - \frac{1}{m}\right)\left(1 - \frac{1}{m}\right)\cdots\left(1 - \frac{1}{m}\right) \\
&= 1 - \left(1 - \frac{1}{m}\right)^n
\end{aligned}$$

2. BÚSQUEDA ALEATORIA VERSUS BÚSQUEDA LOCAL

La expresión anterior se puede emplear para derivar el valor de *n* que, con una probabilidad suficientemente grande, garantiza la eficacia del método.

$$P(A_1 \cap A_2 \cap \ldots \cap A_n) = 1 - \left(1 - \frac{1}{m}\right)^n > 1 - \alpha$$

Si α es tal que $0 < \alpha < 1$ (probabilidad *a priori* de error en la búsqueda, es decir, probabilidad de que la BA no encuentre el óptimo), entonces:

$$\Longleftrightarrow \quad \alpha > \left(1 - \frac{1}{m}\right)^n$$

$$\Longleftrightarrow \quad log(\alpha) > log\left(\left(1 - \frac{1}{m}\right)^n\right)$$

$$\Longleftrightarrow \quad log(\alpha) > nlog\left(\left(1 - \frac{1}{m}\right)\right)$$

$$\Longleftrightarrow \quad n > \frac{log(\alpha)}{log\,(1 - 1/m)}$$

Es decir, n sería el nº de iteraciones necesario para garantizar la obtención del óptimo con probabilidad 1-α

2. BÚSQUEDA ALEATORIA VERSUS BÚSQUEDA LOCAL

EJEMPLO:

Valores de n (número de iteraciones necesario) para distintos valores de la probabilidad de fallo α y del tamaño del espacio de búsqueda m

P_{fallo} α	$P_{éxito}$ $1-\alpha$	nºsol m	nºIt. n
0.1	0.9	1000	2302
0.2	0.8	1000	1609
0.3	0.7	1000	1203
0.4	0.6	1000	916
0.1	0.9	2000	4605
0.2	0.8	2000	3219
0.3	0.7	2000	2408
0.4	0.6	2000	1833

P_{fallo} α	$P_{éxito}$ $1-\alpha$	nºsol m	nºIt. n
0.1	0.9	3000	6907
0.2	0.8	3000	4828
0.3	0.7	3000	3612
0.4	0.6	3000	2748
0.1	0.9	4000	9210
0.2	0.8	4000	6437
0.3	0.7	4000	4816
0.4	0.6	4000	3664

Número de iteraciones para la búsqueda aleatoria pura

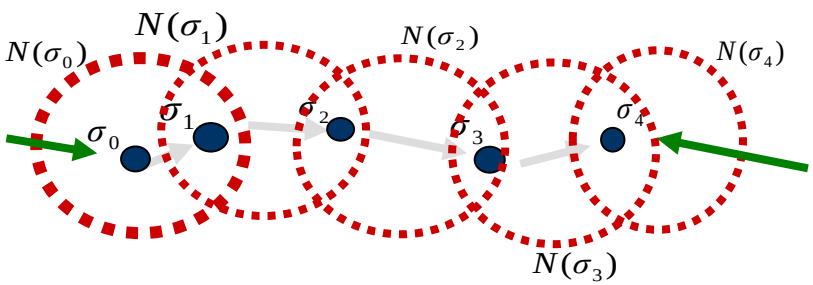
2. BÚSQUEDA ALEATORIA VERSUS BÚSQUEDA LOCAL

2.2. Búsqueda Aleatoria Por Recorrido al Azar

- **Se obtiene desde la descripción de una Búsqueda por Entornos.**
- **La solución inicial se genera aleatoriamente.**
- **El entorno de cualquier solución es propio (no consta de todas las soluciones del espacio de búsqueda).**
- **La solución vecina a la solución actual se escoge aleatoriamente dentro del entorno y se acepta automáticamente.**
- **Se almacena la mejor solución obtenida hasta el momento. Ésta es la solución que se devuelve finalmente.**

- **En definitiva, puede considerarse como una búsqueda local en la que se acepta el primer vecino generado, independientemente de que sea mejor o peor que la solución actual.**

Solución inicial



Óptimo local/global

3. MÉTODOS DE BÚSQUEDA LOCAL BÁSICOS

3.1. Introducción. Procedimiento Base

- **Consiste en el muestreo de soluciones vecinas mejores que la actual en el entorno de ésta.**
- **Hay dos versiones: del Mejor y del Primer Mejor.**
- **En ambos casos, el algoritmo devuelve la última solución visitada (no es necesario ir almacenando la mejor solución).**

3. MÉTODOS DE BÚSQUEDA LOCAL BÁSICOS

Procedimiento General Búsqueda Local

Inicio

GENERA(Solución Inicial);
Solución Actual ← Solución Inicial;

Repetir

GENERA_SOLUCIÓN_ENTORNO(Solución Vecina *tal que*
Objetivo(Solución Vecina) **mejor que** Objetivo(Solución Actual));

Si Objetivo(Solución Vecina) **mejor que** Objetivo(Solución Actual)
entonces Solución Actual ← Solución Vecina;

Hasta (Objetivo(Solución Vecina) **peor o igual que**
Objetivo(Solución Actual), $\forall$ S $\in$ E(Solución Actual));

DEVOLVER(Solución Actual);

Fin

3. MÉTODOS DE BÚSQUEDA LOCAL BÁSICOS

3.2. Búsqueda Local del Mejor

Steepest-Ascent Hill Climbing (best neighbor)

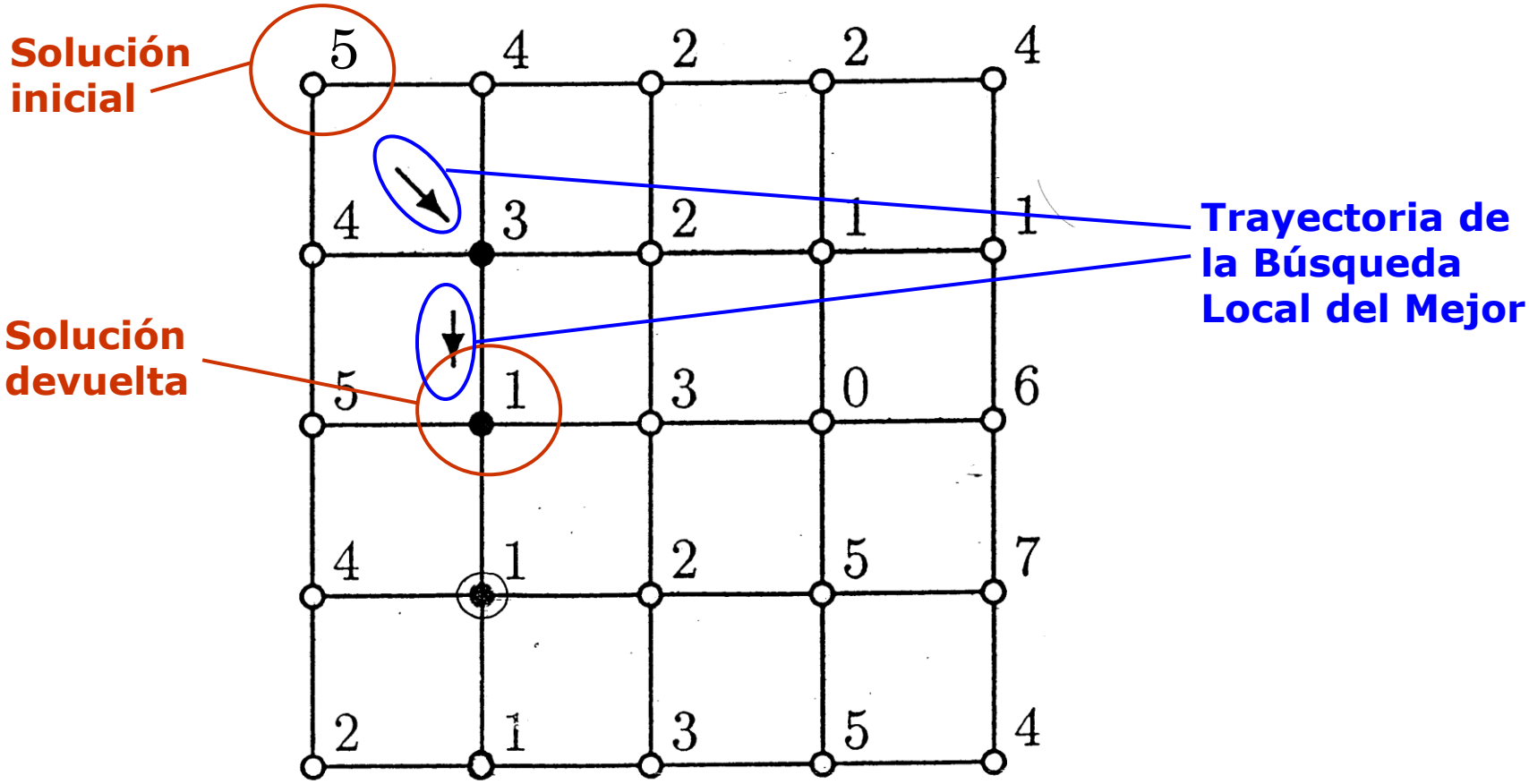
- **Genera el entorno completo de la solución actual y selecciona la mejor solución vecina.**
- **Si ésta es mejor que la solución actual, la sustituye y se continúa la iteración.**
- **En otro caso, el algoritmo finaliza.**

3. MÉTODOS DE BÚSQUEDA LOCAL BÁSICOS

Procedimiento Búsqueda Local del Mejor

Inicio
  GENERA(S_{act});
  Repetir
    Mejor Vecino $\leftarrow$ S_{act}
    Repetir para toda S' $\in$ E(S_{act})
      S' $\leftarrow$ GENERA_VECINO(S_{act});
      Si Objetivo(S') **mejor que** Objetivo(Mejor Vecino) entonces
        Mejor Vecino $\leftarrow$ S';
    Fin-Repetir-para
    Si Objetivo(Mejor Vecino) **mejor que** Objetivo(S_{act}) entonces
      S_{act} $\leftarrow$ Mejor Vecino;
  Hasta (Objetivo(Mejor Vecino) **peor o igual que** Objetivo(S_{act}));
  DEVOLVER(S_{act});
Fin

3. MÉTODOS DE BÚSQUEDA LOCAL BÁSICOS



EJEMPLO: E(s) = {si / si = (xi ± {0,1}, yi ± {0,1}) ∧ si ≠ s}

3. MÉTODOS DE BÚSQUEDA LOCAL BÁSICOS

3.3. Búsqueda Local del Primer Mejor

Simple Hill Climbing (first-best neighbor)

- **Se va generando paso a paso el entorno de la solución actual hasta que se obtiene una solución vecina que mejora a la actual o se construye el entorno completo.**
- **En el primer caso, la solución vecina sustituye a la actual y se continúa iterando.**
- **En el segundo, se finaliza la ejecución del algoritmo.**

3. MÉTODOS DE BÚSQUEDA LOCAL BÁSICOS

Procedimiento Búsqueda Local del Primer Mejor

Inicio
  GENERA(S_{act});

  Repetir

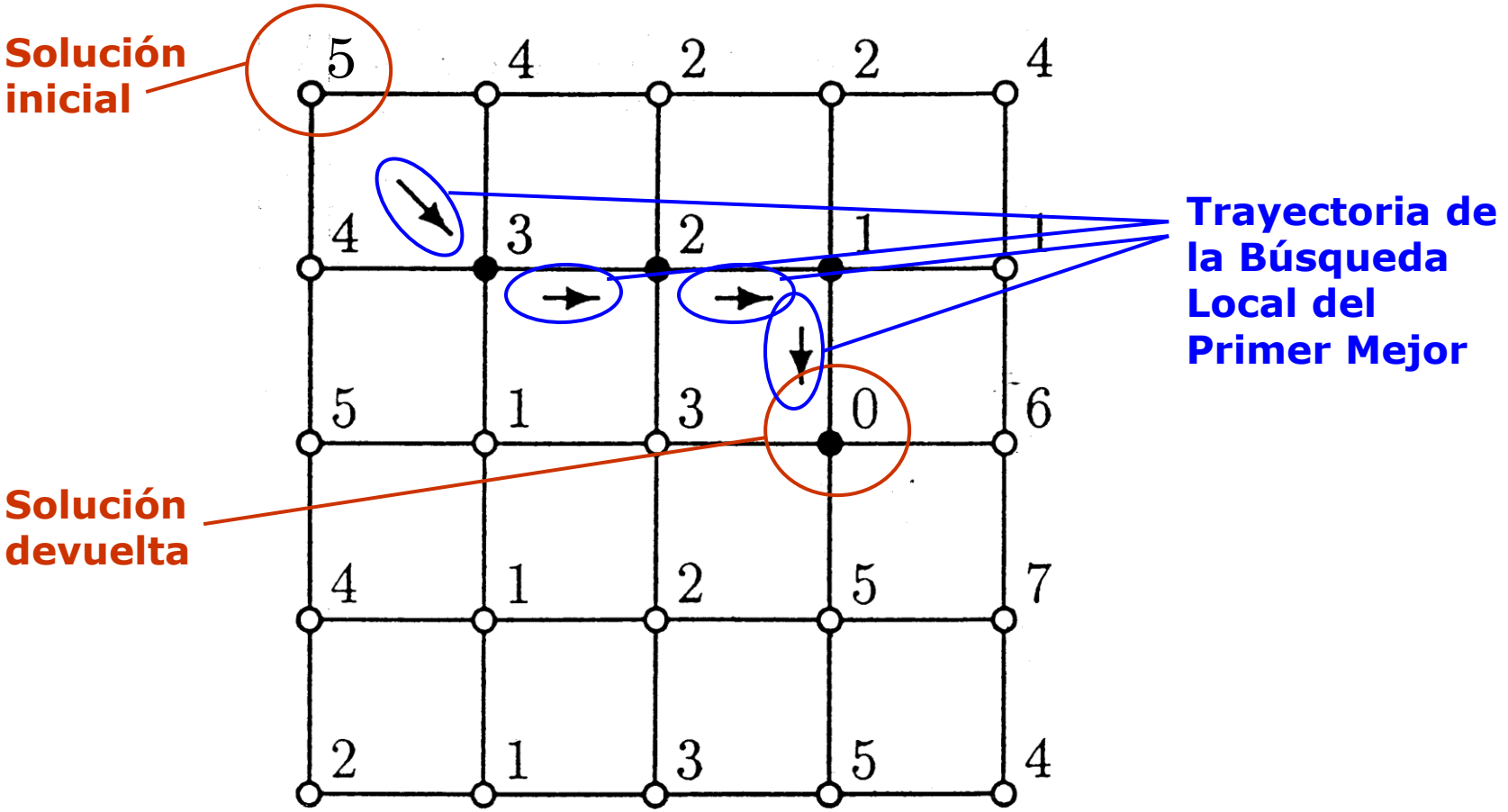
    Repetir
      S' ← GENERA_VECINO(S_{act});
    Hasta (Objetivo(S') **mejor que** Objetivo(S_{act})) **O**
        (se ha generado E(S_{act}) al completo)

    Si Objetivo(S') **mejor que** Objetivo(S_{act}) entonces
      S_{act} ← S';

  Hasta (Objetivo(S') **peor o igual que** Objetivo(S_{act}));

  DEVOLVER(S_{act});
Fin

3. MÉTODOS DE BÚSQUEDA LOCAL BÁSICOS

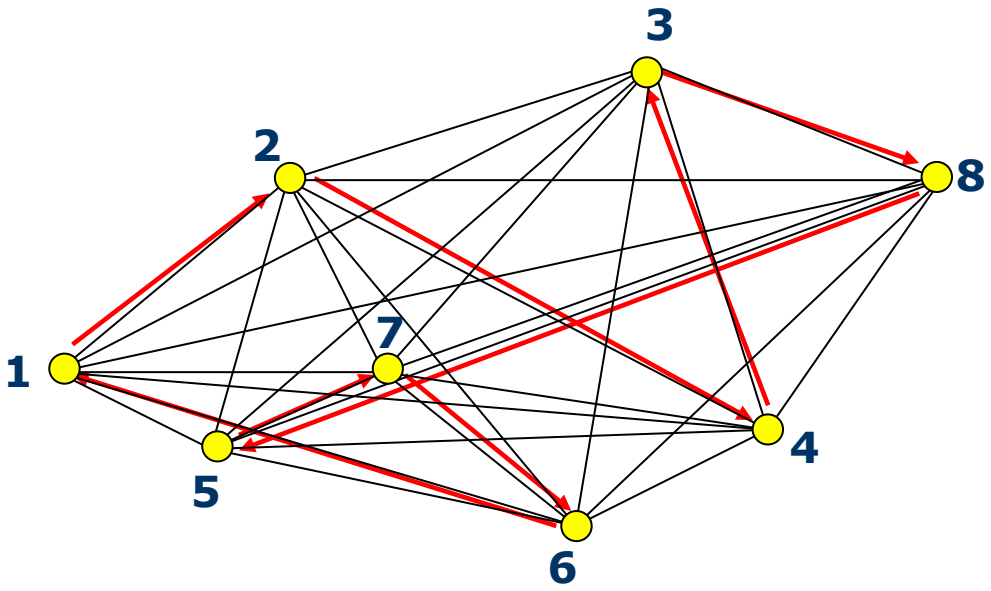


EJEMPLO: E(s) = {si / si = (xi ± {0,1}, yi ± {0,1}) ∧ si ≠ s}

3. MÉTODOS DE BÚSQUEDA LOCAL BÁSICOS

3.4. Ejemplo: Viajante de Comercio

- Ejemplo: Viajante de Comercio



- Representación de una solución: Camino

(1 2 4 3 8 5 7 6)

3. MÉTODOS DE BÚSQUEDA LOCAL BÁSICOS

1. Esquema de representación:
   Permutación de {1, ..., n}.

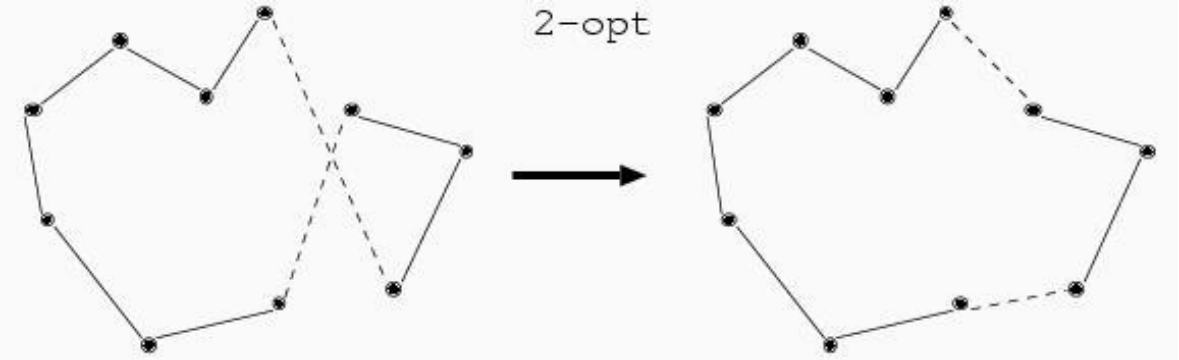
2. Función objetivo:

$$Min\ C(S) = \sum_{i=1}^{n-1} (D[S[i], S[i+1]]) + D[S[n], S[1]]$$

3. Mecanismo de generación de la solución inicial:
   Permutación *aleatoria*.

3. MÉTODOS DE BÚSQUEDA LOCAL BÁSICOS

4. Operador de generación de nuevas soluciones: escoger dos posiciones e intercambiar sus valores (2-opt):



5. Mecanismo de selección: Selección del mejor o el primer mejor.

6. Criterio de parada: Cuando el vecino generado no mejore a la solución actual.

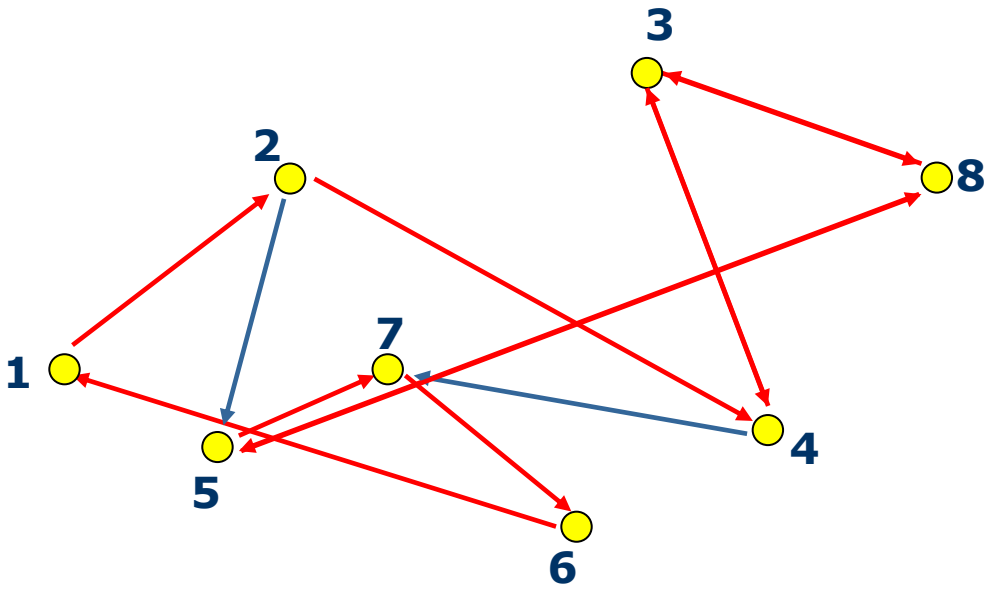
3. MÉTODOS DE BÚSQUEDA LOCAL BÁSICOS

Otro operador: Viajante de Comercio

- Inversión simple (2-Opt): se escoge una sublista y se invierte el orden

(1 2 <u>4 3 8 5</u> 7 6)

(1 2 <u>5 8 3 4</u> 7 6)

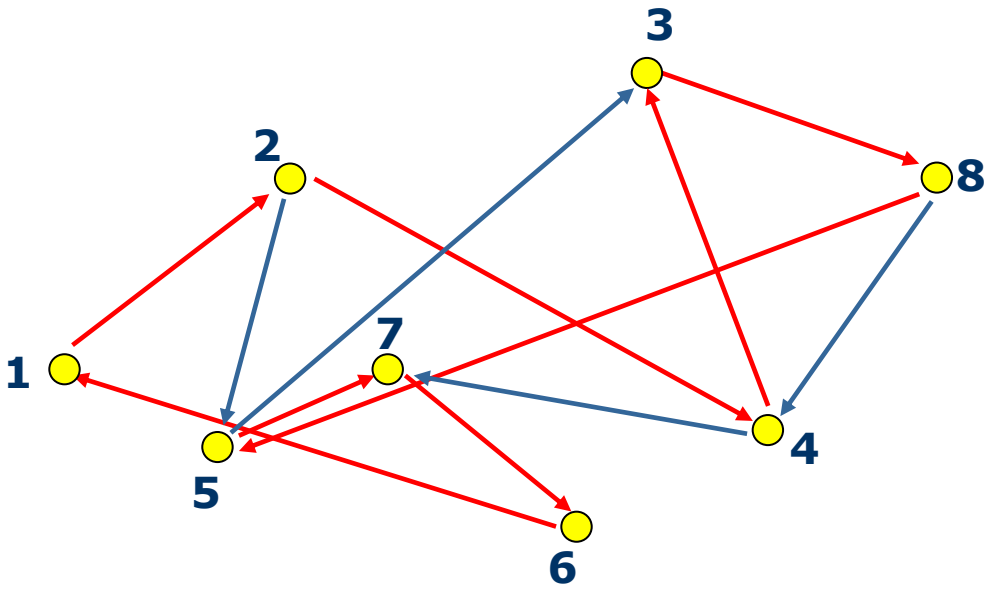


3. MÉTODOS DE BÚSQUEDA LOCAL BÁSICOS

- Intercambio: se escogen dos elementos y se intercambian

(1 2 <u>4</u> 3 8 <u>5</u> 7 6)

(1 2 <u>5</u> 3 8 <u>4</u> 7 6)



3. MÉTODOS DE BÚSQUEDA LOCAL BÁSICOS

TSP: Factorización del Movimiento 2-Opt

- Sea f(π) el coste de la solución original
- Para generar π', el operador de vecino elimina los arcos:

$$\pi(i)\pi(i+1) \text{ y } \pi(j)\,\pi(j+1),$$

- y restablece el circuito con los arcos:

$$\pi(i)\pi(j+1) \text{ y } \pi(j)\,\pi(i+1)$$

- Po $\Delta f(\pi) = c_{\pi(i)\pi(j+1)} + c_{\pi(i+1)\pi(j)} - (c_{\pi(i)\pi(i+1)} + c_{\pi(j)\pi(j+1)})$

- y la variación en el coste de f(π) es:

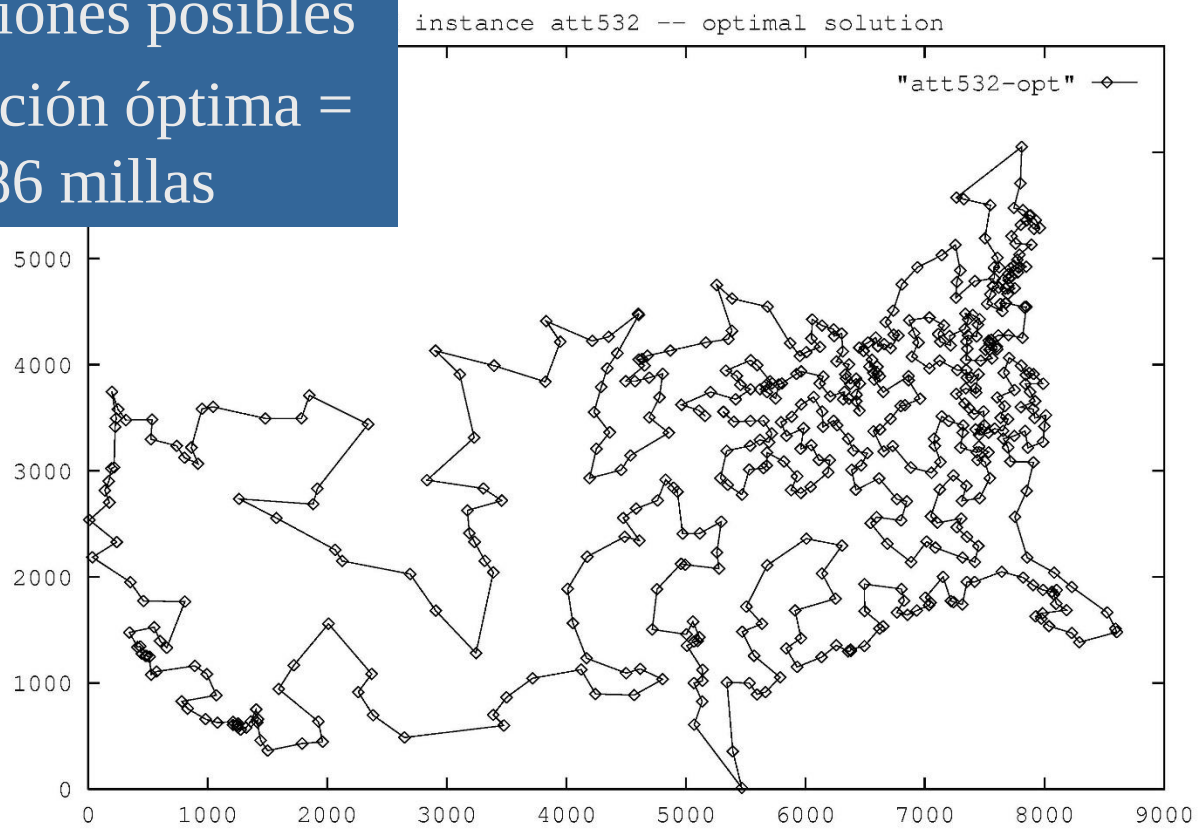
$$f(\pi') = f(\pi) + \Delta f(\pi)$$

3. MÉTODOS DE BÚSQUEDA LOCAL BÁSICOS

Solución óptima

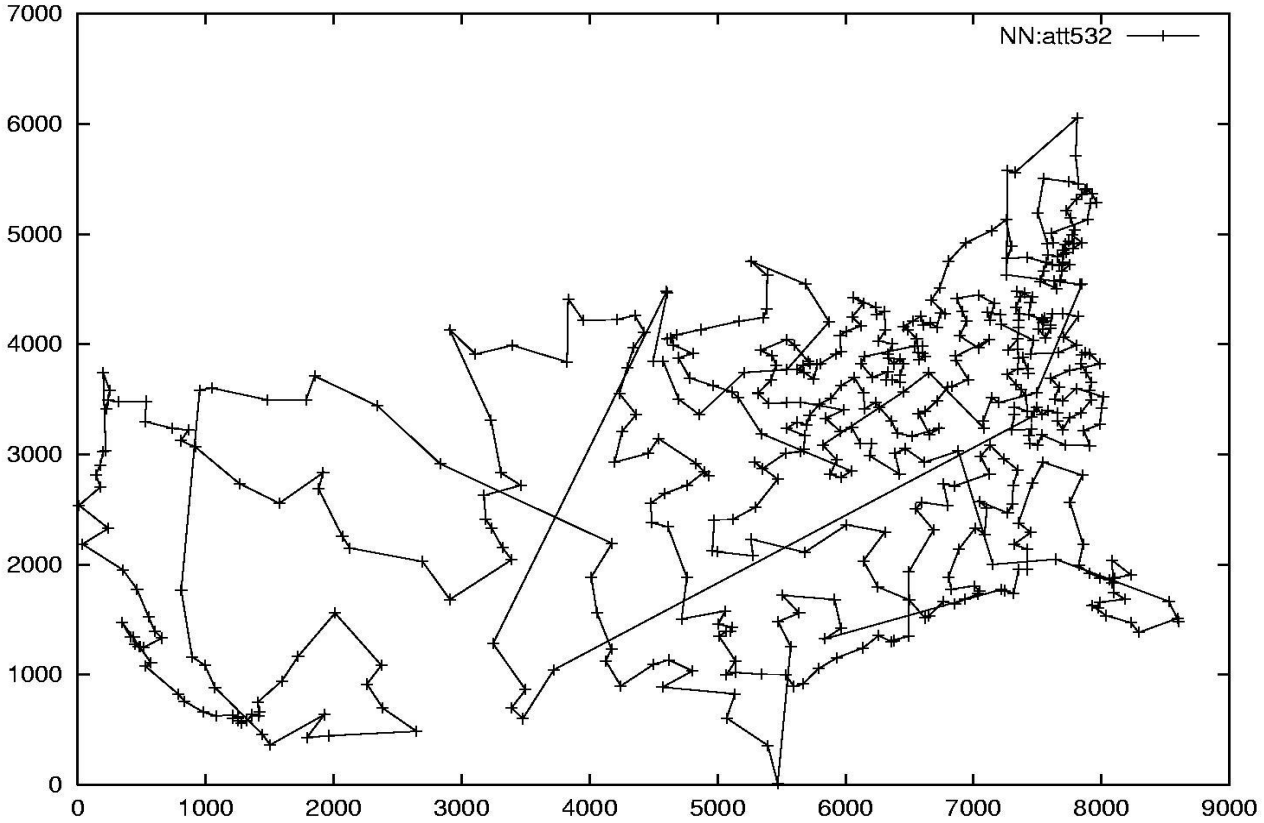
532! soluciones posibles

Coste solución óptima = 27.686 millas



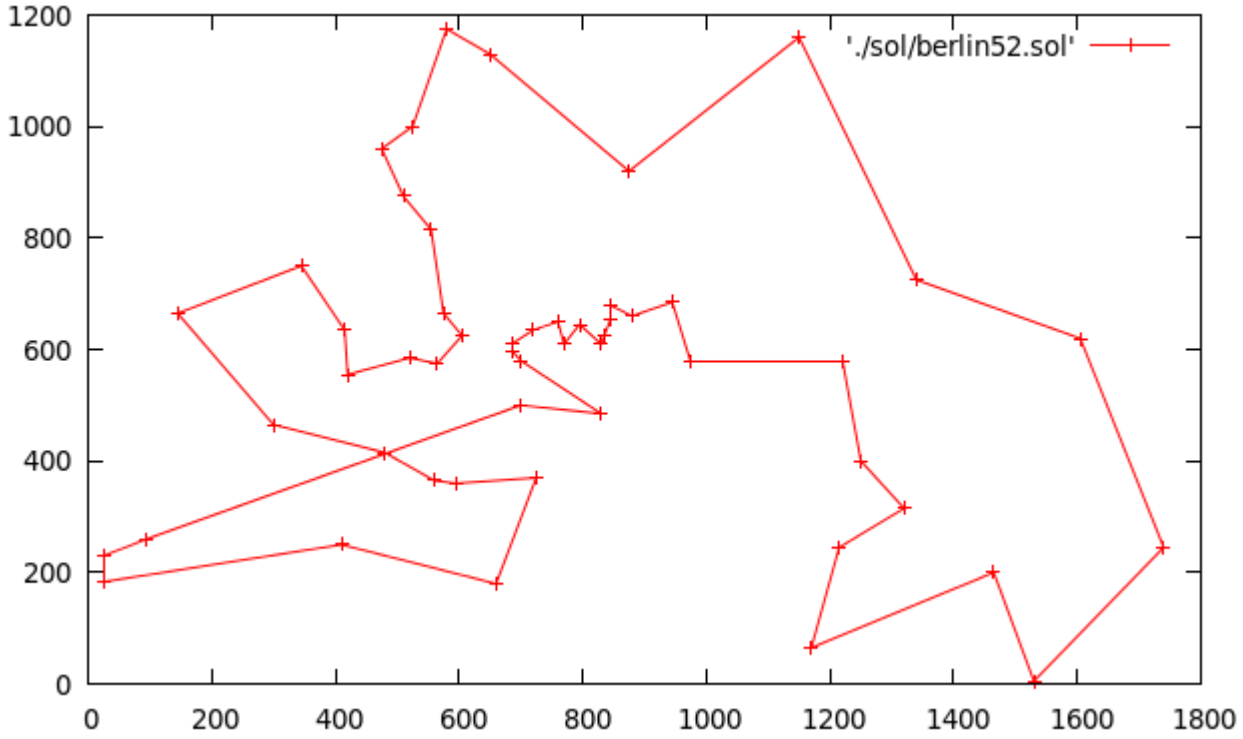
3. MÉTODOS DE BÚSQUEDA LOCAL BÁSICOS

Solución obtenida en una ejecución del algoritmo de Búsqueda Local del Mejor



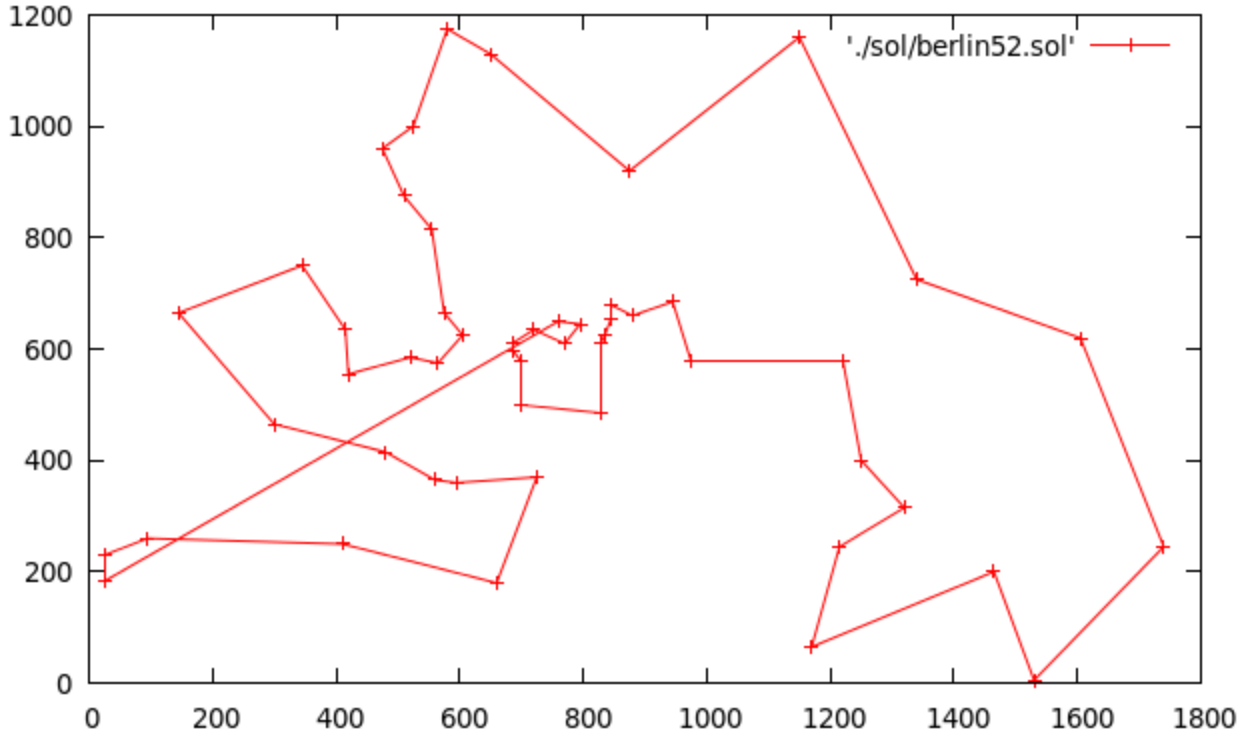
3. MÉTODOS DE BÚSQUEDA LOCAL BÁSICOS

TSP – Instancia Berlin52



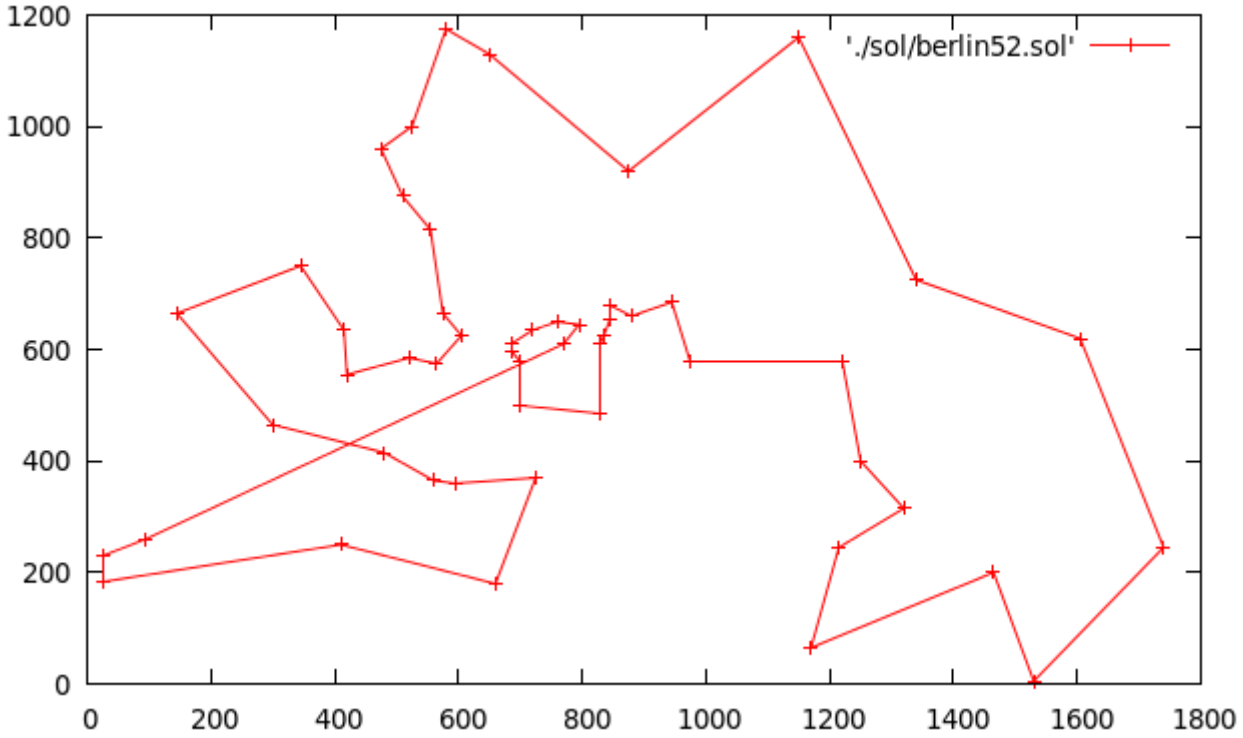
3. MÉTODOS DE BÚSQUEDA LOCAL BÁSICOS

Solución obtenida con la técnica de vecino más cercano



3. MÉTODOS DE BÚSQUEDA LOCAL BÁSICOS

Solución obtenida con la técnica de vecino más cercano y después la Búsqueda 2-OPT



3. MÉTODOS DE BÚSQUEDA LOCAL BÁSICOS

Preguntas:

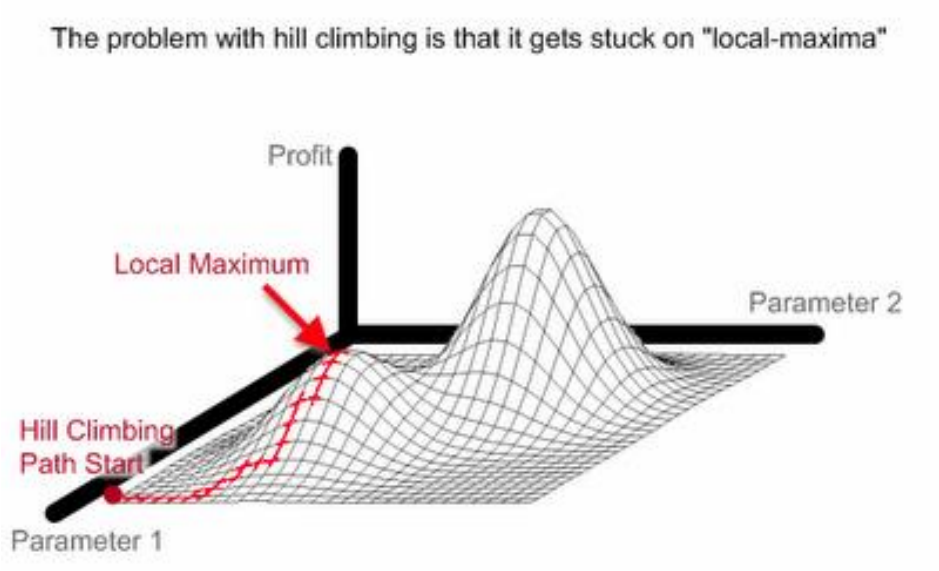
- ¿Cómo de diferente puede llegar a ser la solución vecina respecto de la solución actual según el operador de vecindario empleado?
- ¿Cuál es el tamaño del entorno generado por cada operador de vecino?
- El efecto al aplicar un operador concreto, ¿es el mismo en todos los problemas?

  Por ejemplo, ¿es lo mismo generar un vecino por intercambio de 2 posiciones en TSP que en otro problema?

3. MÉTODOS DE BÚSQUEDA LOCAL BASICOS

3.5. Problemas de la Búsqueda Local

- Suele caer en óptimos locales, que a veces están bastante alejados del óptimo global del problema



3. MÉTODOS DE BÚSQUEDA LOCAL BASICOS

3.5. Problemas de la Búsqueda Local

SOLUCIONES: 3 opciones para salir de los óptimos locales

- Permitir movimientos de empeoramiento de la solución actual

  Ejemplo: Enfriamiento Simulado, Búsqueda Tabú (T5).

- Modificar la estructura de entornos

  Ejemplo: Búsqueda Tabú, Búsqueda en Entornos Variables: VNS, …(T5)

- Volver a comenzar la búsqueda desde otra solución inicial

  Ejemplo: Búsquedas Multiarranque, ILS, VNS, … (T5).

Bibliografía general

[Tal09] E.-G. Talbi. Metaheuristics. From design to implementation. Wiley, 2009

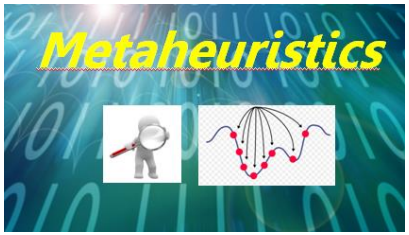
[Bro12] J. Brownlee. Clever Algorithms (Nature-Inspired Programming Recipes). 2012, Brownlee.

http://cleveralgorithms.com/

http://cleveralgorithms.com/

6. **Algorithms**
    1. Stochastic Algorithms
        1. Random Search
        2. Adaptive Random Search
        3. Stochastic Hill Climbing
        4. Iterated Local Search
        5. Guided Local Search

METAHEURÍSTICAS
2020 - 2021



- Tema 1. Introducción a las Metaheurísticas
- Tema 2. Modelos de Búsqueda: Entornos y Trayectorias vs Poblaciones
- Tema 3. Metaheurísticas Basadas en Poblaciones
- Tema 4: Algoritmos Meméticos
- Tema 5. Metaheurísticas Basadas en Trayectorias
- Tema 6. Metaheurísticas Basadas en Adaptación Social
- Tema 7. Aspectos Avanzados en Metaheurísticas
- Tema 8. Metaheurísticas Paralelas