Tema 1. Heurísticos en Optimización Combinatorial

Abdelmalik Moujahid, Iñaki Inza y Pedro Larrañaga
Departamento de Ciencias de la Computación e Inteligencia Artificial
Universidad del País Vasco–Euskal Herriko Unibertsitatea

1.1 Introducción

En este tema se van a presentar las ideas básicas de las técnicas de optimización heurística en dominios discretos, de ahí el título del tema, *Heurísticos en Optimización Combinatorial*.

Conviene recordar que la formulación general de un problema de Investigación Operativa consiste en optimizar $f(\mathbf{x})$, siendo:

$$\begin{aligned} f: \quad & D_1 \times D_2 \times \ldots \times D_n \longrightarrow \mathbb{R} \\ & \mathbf{x} = (x_1, x_2, \ldots, x_n) \longrightarrow f(\mathbf{x}) \end{aligned}$$

Es habitual que $f(\mathbf{x})$ esté sujeta a un conjunto de restricciones:

$$\begin{aligned} h_i(\mathbf{x}) &< b_i \qquad i = 1, \ldots, l \\ h_i(\mathbf{x}) &> b_i \qquad i = l+1, \ldots, m \\ h_i(\mathbf{x}) &= b_i \qquad i = m+1, \ldots, r \end{aligned}$$

En un problema de optimización combinatoria que cumple que $D_i \subset \mathbb{N} (i = 1, \ldots, n)$ y por tanto el espacio de soluciones está formado por permutaciones o por productos cartesianos de subconjuntos de $\mathbb{N}$.

1.2 Heurísticos

Tratando de buscar la etimología de la palabra heurístico nos encontramos que proviene del griego *heuriskein*, término que puede ser traducido por encontrar o descubrir. También la expresión utilizada por Arquímedes, *heureka* – que significa "lo he encontrado" – al descubrir el denominado de principio de Arquímedes, está relacionada con el término heurístico.

Mucho más cercano en el tiempo, el término heurístico fue utilizado a mediados del siglo pasado por dos de los precursores de la Inteligencia Artificial. Polya en 1957 en su obra *How to Solve It* lo utiliza por vez primera en este contexto. Años más tarde Simon (1963) define heurístico como un proceso que puede resolver un problema dado pero no ofrece garantías de hacerlo.

La definición que hoy en día más se ajusta al sentido con el que se usa la palabra heurístico es la siguiente: procedimiento simple, a menudo basado en el sentido común, que se supone que va a ofrecer una buena solución (no necesariamente la óptima) de un modo fácil y rápido a problemas difíciles.

La justificación de métodos heurísticos se fundamenta en el hecho de la existencia de problemas de optimización pertenecientes a la categoría denominada NP. Es decir,

problemas para los que no existe un algoritmo de resolución que sea polinomial con el tamaño del problema. Si se demuestra que un problema de optimización pertenece a esta categoría de problemas NP, es práctica habitual el abordarlo por medio de heurísticos de optimización. Desde un punto de vista práctico, para aquellos problemas donde la búsqueda exhaustiva sea ineficiente o para aquellos problemas donde la cardinalidad del espacio de búsqueda aumenta exponencialmente con el tamaño del problema, va a tener sentido la utilización de algoritmos heurísticos.

Por lo que respecta a las *ventajas* derivadas de la utilización de heurísticos para resolver problemas de optimización, quizás la más importante sea la mayor flexibilidad en el manejo del problema que se deriva de las mismas en contraposición con el encorsetamiento subyacente a las técnicas de la investigación operativa clásica. Además, las ideas en las que se basan estas técnicas heurísticas son más comprensibles al compararlas con las técnicas matemáticas de resolución.

El gran *inconveniente* derivado de la utilización de los métodos heurísticos radica en que no es posible conocer a priori la calidad de la solución obtenida con los mismos, desconociéndose por tanto la cercanía de dicha solución con respecto al óptimo global.

Tratando de establecer unos criterios genéricos acerca de en qué condiciones se aconseja utiliza estos procedimientos heurísticos, tenemos que su uso es adecuado cuando se verifica una o más de las siguientes condiciones:

i) No existe un método exacto de resolución, o en el caso de que dicho método exacto exista, el mismo requiere de mucho gasto computacional y/o de memoria.

ii) No es necesario encontrar la solución óptima, en el sentido del óptimo global sino que es suficiente con obtener una solución suficientemente buena.

iii) Los datos son poco fiables y por tanto no tiene sentido el tratar de encontrar el óptimo global para dichos datos, ya que el mismo no sería más que una aproximación al óptimo global que correspondería a los datos correctos.

iv) Existen limitaciones de tiempo en proporcionar la respuesta y/o de memoria en el ordenador que va a efectuar los cálculos.

v) Se va a utilizar el resultado proporcionado por el heurístico de optimización como solución inicial para un algoritmo exacto de tipo iterativo, el cual reducirá considerablemente el número de iteraciones si parte de una solución inicial suficientemente buena.

Desde un punto de vista de estrategias usadas para llevar a cabo búsquedas heurísticas podemos distinguir básicamente 3 tipos: constructivas, mejora de una solución y basadas en poblaciones.

La idea básica en las búsquedas *heurísticas constructivas* consiste en ir añadiendo componentes individuales a la solución inicial hasta que se obtiene una solución final o factible. Un ejemplo de búsqueda heurística constructiva es la búsqueda voraz.

En las búsquedas *heurísticas basadas en la mejora de una solución*, se parte de una solución para en cada paso buscar en la vecindad de la misma una solución mejor, la cual en el caso de que exista reemplaza a la solución actual.

En la búsqueda *heurística basada en poblaciones* se trabaja con una población de

indivíduos que evolucionan a lo largo de las generaciones.

1.3 El problema del viajante de comercio

En el problema del viajante de comercio conocido el coste de trasladarse de una ciudad a otra, se trata de encontrar la gira que visitando todas las ciudades una y sólo una vez, tenga asociada un coste mínimo.

Este problema se conoce bajo el acrónimo de TSP (*Travelling Salesman Problem*) (Lawler y col., 1985), si bien aparece referenciado por vez primera en el año 1932, en la actualidad sigue siendo motivo de estudio, ya que un buen número de problemas de optimización en Inteligencia Artificial se relacionan con la búsqueda de la permutación óptima.

Más formalmente el problema se plantea de la manera siguiente: Dadas n ciudades y una matriz de costos cuyos elementos se denotan por c_{ij} (con $i, j = 1, \ldots, n$), se trata de encontrar la permutación $\pi^* = (\pi^*(1), \ldots, \pi^*(n))$ con menor costo asociado. Es decir:

$$\pi^* = (\pi^*(1), \ldots, \pi^*(n)) = argmin_\pi \sum_{i=1}^{n-1} c_{\pi(i)\pi(i+1)} + c_{\pi(n)\pi(1)}$$

A modo de ejemplo vamos a considerar la siguiente matriz de costos, correspondiente a un TSP con $n = 6$ ciudades:

$$\begin{pmatrix} 0 & 6 & 2 & 1 & 4 & 10 \\ 6 & 0 & 3 & 4 & 3 & 1 \\ 2 & 3 & 0 & 2 & 8 & 3 \\ 1 & 4 & 2 & 0 & 5 & 6 \\ 4 & 3 & 8 & 5 & 0 & 9 \\ 10 & 1 & 3 & 6 & 9 & 0 \end{pmatrix}$$

1. Escoger una ciudad al azar como ciudad de partida.
2. Mientras queden ciudades por visitar, escoger de entre las no visitadas la que se encuentre a menor distancia de la última visitada.
3. Mostrar la solución

Figura 1.1: Pseudocódigo de un algoritmo heurístico constructivo voraz para el TSP

La Figura 1.1 muestra el pseudocódigo de un algoritmo heurístico constructivo voraz. Siguiendo dicho algoritmo y suponiendo que la ciudad de partida es la cuarta, la gira que se obtiene es la siguiente:

$$4 - 1 - 3 - 2 - 6 - 5$$

El costo asociado a dicha gira es:

$$c_{41} + c_{13} + c_{32} + c_{26} + c_{65} + c_{54} = 1 + 2 + 3 + 1 + 9 + 5 = 21$$

Dos cuestiones básicas a tener en cuenta en relación con este algoritmo son que por una parte no hay garantía de que se alcance el óptimo global con el mismo, y por otra parte que el *algoritmo es determinista*, es decir, fijada la ciudad inicial - y a no

ser que haya empates - siempre se alcanza la misma solución.

Tratando de evitar el que el algoritmo se vaya a quedar atrapado en óptimos locales, se puede plantear una *versión estocástica* del algoritmo heurístico constructivo voraz anterior sin más que considerar que en el Paso 2 del algoritmo previo no vamos a escoger la ciudad, que de entre las no visitadas, esté a mínima distancia de la última visitada, sino que vamos a seleccionar de entre las no visitadas al azar y con una probabilidad inversamente proporcional a la distancia con respecto a la última ciudad visitada. Así por ejemplo, si volvemos a considerar a la ciudad cuarta como la inicial, las probabilidades de seleccionar las ciudades uno, dos, tres, cinco y seis son respectivamente de $\frac{k}{1}$, $\frac{k}{4}$, $\frac{k}{2}$, $\frac{k}{5}$ y $\frac{k}{6}$, verificándose además que $\frac{k}{1} + \frac{k}{4} + \frac{k}{2} + \frac{k}{5} + \frac{k}{6} = 1$. De manera análoga se seleccionarían el resto de ciudades, hasta obtener una gira.

La Figura 1.2 muestra el pseudocódigo de un *algoritmo de mejora iterativa* de una

---

1. Seleccionar al azar una permutación de ciudades π.
2. Repetir
   Obtener la permutación π' vecina de π con menor costo
   Hacer $\pi := \pi'$
   Hasta que el costo de ninguna permutación vecina sea menor que el costo de la solución actual

---

Figura 1.2: Algoritmo de mejora iterativa de una solución para el TSP

solución basado en un optimizador local, con el cual se sustituye la solución actual por la mejor de sus soluciones vecinas, hasta que esto no es posible. Este algoritmo se denomina habitualmente algoritmo de ascenso por la colina (*hillclimbing*).

Podríamos pensar en una *versión estocástica* del mismo sin más que permitiendo saltar a una solución vecina de la actual, escogida al azar, sin que necesariamente dicha solución constituya la mejor de todas las soluciones pertenecientes a la vecindad de la solución actual.

Otra posibilidad, denominada *threshold accepting* pasa por aceptar saltos a soluciones vecinas cuya evaluación supere el α por ciento (por ejemplo, el 95 %) de la evaluación de la actual solución.

O incluso podríamos saltar a soluciones que estando en la vecindad de la solución actual tengan peor función objetivo. Parece intuitivo permitir estos cambios a soluciones con peor función objetivo con una baja probabilidad para aquellas soluciones cuya evaluación esté muy alejada de la evaluación de la solución actual.

La Figura 1.3 muestra dos vecindades de la solución $\mathbf{x}$. Al llevar a cabo un algoritmo de mejora iterativa como el de la Figura 1.2 con la vecindad más pequeña se quedaría atrapado en el óptimo local x', mientras que con la vecindad mayor se alcanzaría el óptimo global x^*.

1.4 Metaheurísticos

Durante las dos últimas décadas se han venido desarrollando distintos algoritmos heurísticos de optimización, algunos de los cuales son genéricos e independientes del problema que tratan de optimizar. A este tipo de algoritmos heurísticos se les conoce bajo el nombre de metaheurísticos. La gran ventaja de los mismos radica en que una pequeña modificación o adaptación con respecto de la formulación general es sufi-

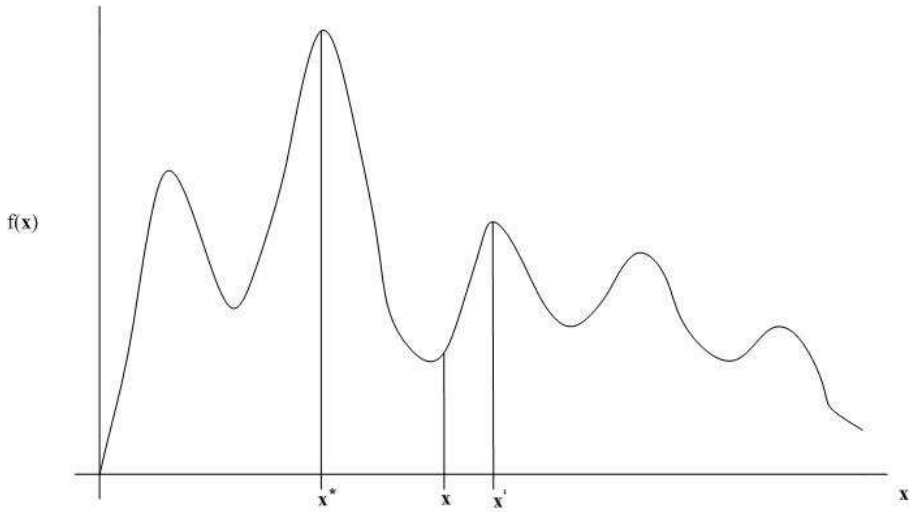

Figura 1.3: Influencia de la vecindad en el heurístico de mejora iterativa de una solución

ciente para que puedan ser aplicados a un problema concreto.

Algunos de estos metaheurísticos son los siguientes: enfriamiento estadístico (Kirkpatrick y col. 1983), algoritmos genéticos (Goldberg, 1989), algoritmos de estimación de distribuciones (Larrañaga y Lozano, 2002), estrategias evolutivas (Rechenberg, 1973), programación genética (Koza, 1992), vecindad variable (Hansen y Mladenovic, 2001) y búsqueda tabú (Glover y Laguna 1997).

La Tabla 1.1 recoge una clasificación de los mismos teniendo en consideración por una parte el hecho de que utilicen estocasticidad dentro del algoritmo heurístico, así como en base a si usan una población de individuos en lugar de un único individuo.

Metaheurístico	Estocástico	Población
Enfriamiento Estadístico	sí	no
Algoritmos Genéticos	sí	sí
Algoritmos de Estimación de Distribuciones	sí	sí
Estrategias Evolutivas	sí	sí
Programación Genética	sí	sí
Vecindad Variable	no	no
Búsqueda Tabú	no	no

Tabla 1.1: Características de metaheurísticos

Referencias

- F. Glover, M. Laguna (1997). *Tabu Search*. Kluwer Academic Publisher.
- D. E. Goldberg (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison Wesley.
- P. Hansen, N. Mladenovic (2001). Variable neighborhood search. Principles and applications. *European Journal of Operational Research*, 130, 449–467.
- S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi (1983). Optimization by Simulated Annealing. *Science*, 220, 671–680.
- J. R. Koza (1992). *Genetic Programming*. MIT Press.
- P. Larrañaga, J. A. Lozano, (2002). *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation*. Kluwer Academic Publisher.
- E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy-Kan, D. B. Shmoys (1985). *The Travelling Salesman Problem*. Wiley.
- I. Rechenberg (1973). *Evolutionsstrategien – Optimierung Technischer Systeme nach Principien der Biologischen Information*. Friedrich Frommann Verlag.