Recibido: lunes, 05 noviembre 2007

ciencia :: matematicalia
revista digital de divulgación matemática
Vol. 3, nos. 4-5 (oct.-dic. 2007)

Matemáticas y Evolución: Algoritmos Genéticos

María Teresa Iglesias Otero
Departamento de Matemáticas
Universidade da Coruña
e-mail: totero@udc.es

A menudo nos enfrentamos a preguntas del tipo: ¿cuál es la mejor forma de...?, ¿cuál es el camino más corto para ir a...?, ¿cuál es la opción más barata entre...? En todas estas cuestiones subyace un *problema de optimización:* dado un conjunto de opciones, $\boldsymbol{D}$, (posibles soluciones a un problema concreto), hay que encontrar la mejor entre todas ellas (i.e.: en el espacio de búsqueda). Pero, ¿cómo medir la idea de *mejor*?

Cada ítem del espacio de búsqueda –cada posible solución de nuestro problema de la vida real– debería tener asociado un valor, de manera que la resolución del problema consistirá en buscar los elementos para los cuales ese valor es máximo (o mínimo, dependiendo de la situación en cuestión). Pero $\boldsymbol{D}$ puede ser muy general –finito o infinito–, puede consistir en números, vectores, grafos, o cualquier tipo de objeto que se desee estudiar: caminos de una ciudad a otra, por ejemplo. Por otra parte, necesitamos una regla, esto es, una función (pongamos f) que asocie a cada elemento de $\boldsymbol{D}$ un valor que mida la *calidad* de ese objeto respecto al problema que se pretende resolver. Este valor puede ser el precio de un producto, la distancia a recorrer de una ciudad a otra, etc.

Aunque podría considerarse cualquier conjunto de valores, lo más habitual en la práctica es trabajar con números reales; es decir, la función asigna a cada valor de $\boldsymbol{D}$ un número real. Esta función, que recibe nombres distintos dependiendo del contexto (función de coste, función de idoneidad), en el campo de la optimización funcional se denomina *función objetivo*.

Así, el problema de optimización asociado a nuestro problema se reformula como sigue: "Hay que encontrar el elemento (o elementos) de $\boldsymbol{D}$ para el cual f alcanza el valor máximo[1]". La cuestión es, pues, cómo proceder para encontrar el máximo.

Desde luego, si $\boldsymbol{D}$ es un subconjunto abierto del espacio real n-dimensional $\mathbf{R}^n$ disponemos de los métodos tradicionales: se trata de encontrar el valor $s \in \mathbf{R}^n$ tal que $\partial f/\partial x_1(s) = \ldots = \partial f/\partial x_n(s) = 0$.

Pero estamos suponiendo que la función es derivable, y en la mayor parte de las situaciones prácticas esta condición no es asumible. En una gran variedad de problemas la función con la que se trabaja no es derivable (en ocasiones ni siquiera es continua); además, en muchos casos, el espacio de búsqueda es discreto o finito. Por otra parte, aún cuando las derivadas existan, las ecuaciones resultantes pueden ser "horribles"[2], extremadamente difíciles de resolver, incluso numéricamente. Y, aun cuando encontremos una solución, ésta podría ser un óptimo local.

Afortunadamente, existen métodos "alternativos"; por ejemplo, los llamados *métodos del gradiente*. Uno de ellos es el *método del alpinista*[3]. *Grosso modo* este método escoge aleatoriamente un punto del espacio de búsqueda y se mueve iterativamente a puntos próximos de mayor calidad. El proceso se detiene cuando no hay mejoras. Obviamente el algoritmo puede quedar atrapado en máximos locales, pues termina en el óptimo más cercano al punto de partida.

No hemos considerado realizar una búsqueda exhaustiva ya que, claramente, sólo funciona si buscamos el elemento mejor en un conjunto finito con un cardinal muy pequeño. Baste considerar el siguiente ejemplo: *dado un conjunto de N ciudades y las distancias entre ellas, se trata de encontrar la ruta que, partiendo de una ciudad fija, permite visitar todas y cada una de las ciudades, una sola vez cada una, siendo mínima la suma de las distancias*.

Esta situación se conoce como *problema del viajante* o TSP[4]. Si tenemos cuatro ciudades es fácil comparar los $3!$ $(=6)$ itinerarios para encontrar el de longitud más corta. Pero en una situación un poco más realista, por ejemplo 100 ciudades, es obvio que ningún computador será capaz de encontrar la ruta más corta entre un total de $99! \simeq 10^{156}$

posibles[5] (de hecho, este problema es NP-completo[6]). Afortunadamente, para este tipo de problemas se han desarrollado excelentes heurísticas que proporcionan soluciones suficientemente buenas (subóptimos) en un tiempo razonable.

De todo lo anterior se observa que, para problemas específicos, existen algoritmos que proporcionan soluciones razonables pero que, con frecuencia, son *dependientes* de las características del problema (de su derivabilidad, por ejemplo). Para problemas realmente difíciles, en las últimas décadas se han desarrollado algoritmos *universales* que, siendo deterministas, incluyen alguna aleatorización en su inicio o en la dirección de la búsqueda, y que han demostrado ser más eficientes que los métodos tradicionales. Entre los algoritmos de este tipo se encuentran los *Algoritmos Genéticos* (abreviadamente AG).

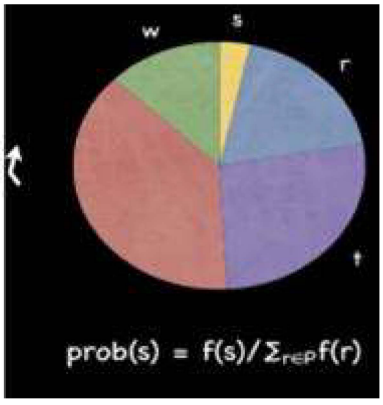
Los Algoritmos Genéticos se inspiran en la naturaleza: en el principio darwiniano de la supervivencia del más fuerte y en las leyes de Mendel. La idea es extremadamente simple: consideremos una población P de individuos de una cierta especie, con características (rapidez, inteligencia,...) que les hacen más o menos susceptibles de ser devorados por ciertos depredadores. Supongamos que podemos describir esas características a través de una función escalar f: $P \to \mathbf{R}$ de forma que el valor $f(p)$ ($p \in P$) sea mayor cuanto mayor sea la probabilidad que tiene p de sobrevivir. La población, desde luego, no es estática, y es de esperar que los individuos con mayor $f(p)$ sobrevivan, se reproduzcan y transmitan esas características a sus descendientes –hijos de padres altos se espera que sean altos, por ejemplo–, mejorando globalmente la calidad media de la población. Como, obviamente, también puede suceder que individuos rápidos tengan descendientes no tan rápidos, a veces individuos menos idóneos también sobreviven. Por otra parte, hay que tener en cuenta otro aspecto que dinamiza (diversifica) la población: la mutación. Aunque son hechos raros en la naturaleza, las mutaciones permiten la introducción de nuevo material genético, impidiendo la homogeneización de la población con el transcurso del tiempo.

Esta es la filosofía de actuación de los AG. Parten de una población P en el espacio de búsqueda y de una función f: $P \to \mathbf{R}$ que asigna a cada individuo un número real que medirá su idoneidad (o calidad) con respecto al problema a maximizar. Para ser estrictos, en lugar de trabajar con elementos del espacio de búsqueda, se trabaja con datos codificados, de forma que un computador los pueda procesar. Aunque existen otras muchas, la forma más habitual de codificar los elementos es utilizar el alfabeto binario; así, a cada elemento del espacio de búsqueda se le asocia un *individuo*[7] que no es otra cosa que una secuencia (cadena o vector) de ceros y unos. Cada posición de la cadena se denomina *gen* y, manteniendo la terminología de la Genética, cada uno de los valores que puede tomar el gen es un *alelo*.

El algoritmo actúa así: en primer lugar genera de forma aleatoria una población inicial $P(0)$ de tamaño N (se permiten cadenas repetidas) y calcula la imagen de cada elemento de $P(0)$ (se calculan a lo sumo N imágenes). Al igual que en Genética, el AG intercambia el "material genético" de los cromosomas mediante cruces y mutaciones generando una nueva población $P(1)$, y el proceso se repite hasta que se verifique la condición de parada fijada por el analista.

A semejanza de la naturaleza, no todo individuo se reproduce ni muta. El algoritmo *selecciona* aquellos individuos (constituyendo una población intermedia) en los que se efectuará la *recombinación* (cruce y/o mutación).

Existen distintos métodos de selección. Entre los más clásicos se cuentan el método de *selección por ruleta* y *selección mediante torneo*.



En el primero de ellos, se simula el juego de la ruleta sobre un círculo formado por sectores circulares de áreas acordes con la calidad relativa de cada elemento de la población. A mayor calidad relativa, mayor área del sector correspondiente –a mayor valor relativo

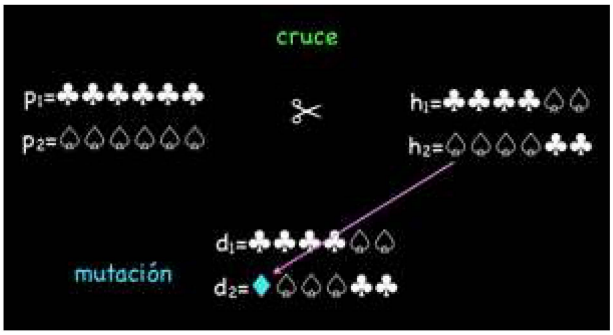
$$f(p)/\sum_{q \in P} f(q),$$

mayor probabilidad de ser seleccionado–. Se efectúan N tiradas y en cada una de ellas se escoge el elemento de la población correspondiente al sector en el que se ha detenido la bola. Los individuos mejores no sólo tienen más probabilidad de ser seleccionados, sino también de tener más copias en esta población intermedia. Individuos de calidades inferiores también pueden ser seleccionados aunque, obviamente, con probabilidad más baja.

La selección por torneo consiste básicamente en escoger un número m (generalmente m=2) de individuos y hacerles competir entre sí. Gana el más fuerte[8] (aquel cuyo valor $f(\cdot)$ es mayor). El torneo se repite hasta obtener N individuos que compongan la población intermedia sobre la que actuarán los *operadores genéticos*.

A continuación comienza el proceso reproductivo (*cruce*) –característica esencial de los AG, que los distingue de otros algoritmos evolutivos–. Existe una gran variedad de métodos de cruce, cuyo propósito común es intercambiar características de los

individuos. Por razones de brevedad exponemos el método clásico de *cruce por un punto*. Consideremos dos cadenas binarias de longitud L, $s=s_1s_2...s_L$ y $t=t_1t_2...t_L$ y escojamos (de forma aleatoria) un punto de cruce $1< i < L$; el proceso de cruce consiste en intercambiar las partes iniciales de s y t (los genes entre la posición 1 y la i-ésima, ambos inclusive). Este proceso produce los descendientes $s^*=t_1t_2...t_is_{i+1}...s_L$ y $t^*=s_1s_2...s_it_{i+1}...t_L$ que reemplazan a sus progenitores. Como no todos los individuos se reproducen, el analista ha de fijar una probabilidad de cruce.



El siguiente operador que interviene alterando la población es el operador *mutación* que, con una probabilidad p_m fijada muy baja, pretende evitar que el AG converja prematuramente a un resultado subóptimo. Exactamente como ocurre en la vida real, la mutación altera eventualmente un gen de un individuo. En la codificación binaria que estamos considerando aquí, alterar un gen equivale a intercambiar un 0 por un 1 (o viceversa) en una posición determinada de la cadena.

Tras todos estos pasos habremos obtenido una nueva generación $P(1)$ del mismo tamaño que $P(0)$ y de la que esperamos que su calidad media se haya incrementado. El proceso se repetirá hasta que se cumpla una condición de parada (que el valor medio de calidad de la población supere un valor umbral prefijado, por ejemplo). Formalmente, el procedimiento de actuación de los algoritmos genéticos clásicos es:

```
Procedimiento
  inicio
  t←0
  iniciar P(t)
  evaluar P(t)
  mientras no se verifique la condición de parada:
  t←t+1
  seleccionar P(t-1)* de P(t-1)
  aplicar cruce a P(t-1)*
  aplicar mutación a P(t-1)*
  P(t)←P(t-1)*
  fin
fin
```

En contraposición a métodos como el del alpinista, o a los métodos analíticos, que trabajan localmente, los AG funcionan “en paralelo”. Realizan la búsqueda muestreando simultáneamente distintas zonas del espacio de posibles soluciones. Esto les confiere un gran poder de búsqueda, combinando la exploración de áreas distintas con la explotación de las estructuras más idóneas.

Hay multitud de diseños de AG[9] que incluyen una gran variedad de operadores genéticos; sin embargo, la característica común a todos ellos es que con la ejecución de este tipo de algoritmos se espera que las poblaciones $P(t)$ incrementen su calidad media de generación en generación. Se pretende que, gradualmente, las generaciones contengan cada vez más individuos con una idoneidad más próxima a la del óptimo buscado. Desde luego, se plantea una cuestión fundamental: ¿por qué esto es así?

Para saber por qué funcionan los algoritmos genéticos es necesario tener en cuenta que en la naturaleza *los individuos que presentan semejanzas están emparentados*. En el campo de los AG, este principio se traduce en la identificación de individuos con estructuras –*esquemas*[10]– altamente idóneas para el problema a tratar. Matemáticamente, un esquema es un elemento del conjunto $\{0,1,\#\}^L$, es decir, una cadena de longitud L, formada por ceros, unos y almohadillas. Cada vez que aparece una almohadilla en una posición significa que ese gen puede tomar el valor cero o uno. Por ejemplo, $1\#\#0$ es el esquema de longitud 4 que representa a todas las cadenas que empiezan por 1 y acaban en 0: 1000, 1010, 1100 y 1101. Esto significa que un esquema identifica un conjunto de cadenas que siguen un patrón. Cada vez que el AG evalúa una cadena particular está evaluando implícitamente los esquemas de los que esa cadena es representante. Está, pues, procesando de forma paralela información sobre la calidad de ciertas estructuras.

Atendiendo al efecto de los operadores genéticos sobre los esquemas, observemos en primer lugar que es más probable que una mutación altere la estructura del esquema $010\#$ que la del esquema $\#\#\#1$, por ejemplo. A mayor número de bits definidos (distintos del símbolo #), mayor es la probabilidad de que una mutación altere la estructura del esquema. El número de bits definidos se conoce como el *orden del esquema*. Por ejemplo, $ord(1\#\#\#)=1$. Por otra parte, también es importante la separación entre los bits definidos de un esquema. A mayor distancia entre ellos, mayor facilidad de que un cruce destruya la estructura del esquema. La distancia entre el primero y el último bit definido de un esquema constituye la *longitud de definición del esquema: long*$(1\#\#0)=3$.

Parece, pues, que esquemas con valores bajos de estos dos parámetros (orden y longitud de definición) tienen más oportunidades de sobrevivir al proceso de recombinación. Si esos esquemas representan estructuras altamente idóneas (con idoneidad media igual o superior a la de la población), tendrán más probabilidades de ser seleccionados para constituir la siguiente generación. Esto es lo que *grosso modo* afirma el resultado teórico más importante que se ha formulado en el campo de los Algoritmos Genéticos: el *teorema de los esquemas*[11] de John Holland, establecido en 1975. Afirma este teorema que si las buenas estructuras de una población pueden describirse mediante esquemas simples (de orden y longitud de definición bajos), entonces esas buenas soluciones tenderán, con el tiempo, a dominar la población.

Este resultado es de valor teórico, pues en la práctica existen diversos factores –sobre los que todavía queda mucho por entender– que pueden complicar el proceso de convergencia de un AG, como la interrelación entre los bits (*epistasis*[12]), por ejemplo. No obstante, esto no resta valor a este tipo de algoritmos pues, con el formidable avance de la capacidad computacional de los ordenadores de hoy en día, son innumerables las aplicaciones prácticas en las que los Algoritmos Genéticos han demostrado su eficiencia. Su rango de aplicaciones se extiende a cuestiones tan dispares como el diseño del ala de un avión supersónico o el de la forma de una sala de conciertos con las mejores características de audición. También se han aplicado con éxito para diseñar turbinas para molinos de parques eólicos, o en ingeniería hidráulica para determinar el diámetro y la distribución de tuberías para redes de agua potable. En el campo médico, se han empleado para detectar variabilidades en el registro de imágenes médicas obtenidas mediante resonancia magnética. Como ejemplo final de su aplicación, permítasenos considerar la elaboración de horarios[13] para centros de enseñanza, tarea ingrata hasta no hace mucho tiempo, aliviada por programas comerciales que, a buen seguro, los jefes de estudios de enseñanza secundaria agradecerán.

Finalmente, incluimos algunas recomendaciones para quienes deseen iniciarse en el tema. Los siguientes enlaces son de fácil y agradable lectura:

A. Marczyk: *Algoritmos genéticos y computación evolutiva*, http://the-geek.org/docs/algen.

J.J. Merelo Guervós:
Informática evolutiva, http://geneura.ugr.es/~jmerelo/ie/intro.htm.
Informática evolutiva: algoritmos genéticos, http://geneura.ugr.es/~jmerelo/ie/ags.htm.

Para lecturas más técnicas, entre la amplia bibliografía existente –además de la de John Holland a la que ya se ha hecho referencia en el texto– destacamos el libro de David Edward Goldberg *Genetic Algorithms in Search, Optimization and Machine Learning*, de 1989 (Addison Wesley Longman, Inc.); así como *Genetic Algorithms+Data Structures=Evolution Programs*, de Zbigniew Michalewicz (Springer, 3ª. ed., 1996).

---

[1] No supone restricción considerar sólo el problema de maximización, pues minimizar una función equivale a maximizar su opuesta.

[2] Por ejemplo, $f(x)=x^3+\exp(\text{sen}(x))-2x$, cuya derivada igualada a cero es $3x^2+\cos(x)\exp(\text{sen}(x))-2=0$.

[3] *Hill-climbing method.*

[4] *Travelling Salesman Problem.*

[5] Se estima que el número total de átomos del universo es del orden de 10^{78}.

[6] Este tipo de problemas son extremadamente difíciles y su complejidad aumenta de forma exponencial al aumentar el conjunto de datos. De hecho, no existe un algoritmo que pueda resolver este problema en un tiempo polinomial.

[7] Heredando la terminología de la genética se suele denominar también *cromosoma* a la cadena binaria.

[8] Con este método de selección el peor individuo sólo se seleccionará si compite contra una copia de sí mismo.

[9] El procedimiento expuesto aquí corresponde al de un AG simple.

[10] Referimos al lector a la bibliografía de John Holland, matemático de la Universidad de Michigan, considerado el padre de los Algoritmos Genéticos.

[11] Para detalles ver *Adaptation in Natural and Artificial Systems*, MIT Press, 1975.

[12] Para detalles ver *A combinatorial approach to epistasis* (M.T. Iglesias et al.), Springer, 2005.

[13] Este problema (NP-completo) es uno de los que con más frecuencia aparecen en la literatura de Investigación de Operaciones.

Sobre la autora

María Teresa Iglesias Otero es licenciada en Ciencias Matemáticas por la Universidad de Santiago de Compostela y doctora en Informática por la Universidade da Coruña. En la actualidad es miembro del Departamento de Matemáticas de la Universidade da Coruña. Profesora titular del área de Matemática Aplicada, imparte su docencia en la Facultad de Informática de La Coruña.