Analizis avanzado de algoritmos

Por: David Gomez 20359596

Recocido simulado

tecnica de busqueda meta-[heurística](https://es.wikipedia.org/wiki/Heur%C3%ADstica_(inform%C3%A1tica))

ASIGNACIÓN NÚMERO DOS

El problema

Se desea encontrar la secuencia de bits, de 16 dígitos de longitud, que contenga la mayor cantidad de subsecuencias 1 0. a partir de una secuencia aleatoria.

Si bien este problema pudiera parecer de solución trivial sirve como una base simple sobre la cual implementar el algoritmo de recocido simulado. Este es un [algoritmo de búsqueda](https://es.wikipedia.org/wiki/Algoritmo_de_b%C3%BAsqueda) meta-[heurística](https://es.wikipedia.org/wiki/Heur%C3%ADstica_(inform%C3%A1tica)) para problemas de [optimización](https://es.wikipedia.org/wiki/Optimizaci%C3%B3n_(matem%C3%A1tica)) global; el objetivo general de este tipo de algoritmos es encontrar una buena aproximación al valor óptimo de una [función](https://es.wikipedia.org/wiki/Funci%C3%B3n_matem%C3%A1tica) en un [espacio de búsqueda](https://es.wikipedia.org/wiki/Espacio_de_b%C3%BAsqueda) grande. A este valor óptimo se lo denomina "óptimo global"

Para este problema dicho óptimo global es un valor conocido, se trata de una secuencia que contiene 8 subsecuencias 10. El óptimo global puede utilizarse como una condición de parada, pero no es estrictamente necesario ya que idealmente el algoritmo de recocido simulado tendera hacia el óptimo global por su propia cuenta.

Descripción de la solución propuesta

Desarrollar un programa escrito en código java que implemente la técnica de recocido simulado para dar con el óptimo global a partir de una solución inicial generada aleatoriamente.

Se parte desde una solución o estado inicial situado o generado aleatoriamente en el espacio de solución del problema. En cada [iteración](https://es.wikipedia.org/wiki/Iteraci%C3%B3n), el método de recocido simulado evalúa las vecindades del estado o solución actual **S** y probabilísticamente decide entre efectuar una transición a un nuevo estado **S'** o quedarse en el estado **S**. Típicamente la comparación entre estados vecinos se repite hasta que se encuentre un estado óptimo que minimice la “energía” del sistema o hasta que se cumpla cierto tiempo computacional u otras condiciones.

Nos referimos a energía como el potencial de solución de un estado particular, aplicándose a este caso la energía se refiere al número de subsecuencias 10 en cada solución generada.

Para esta implementación se han establecido dos condiciones de parada: que se alcance un mínimo de temperatura de 0,0001 o que la solución actual encuentre el máximo global.

El vecindario de un estado s está compuesto por todos los estados a los que se pueda llegar a partir de s mediante un cambio en la conformación del sistema. El método de evaluación de estados vecinos es fundamental para encontrar una solución óptima global al problema dado. Los [algoritmos heurísticos](https://es.wikipedia.org/wiki/Algoritmo_heur%C3%ADstico), basados en buscar siempre un estado vecino mejor que el actual se detienen en el momento que encuentran un óptimo local. El problema con estos métodos es que no pueden asegurar que la solución encontrada sea un óptimo global, pues el espacio de búsqueda explorado no abarca todas las posibles variaciones del sistema.

Para este problema se escogió que el rango de la vecindad de una solución es de una diferencia de una subsecuencia de **10**, esto significa; por ejemplo, que para una solución que contenga 6 subsecuencias **10**, sus vecinas podrán ser cualquier secuencia que contenga o bien 5 o 7 subsecuencias **10**.

La decisión de pasar a un estado vecino no quedarse en el estado actual viene determinada como sigue:

Si *NE>E→P=1*  si no,

E se refiere en este caso al número de subsecuencias **10** que contiene la solución actual y NE se refiere al número de subsecuencias **10** que contiene la nueva solución. Tenemos que si la nueva solución es mejor que la solución actual, siempre se decidirá cambiar a esa solución.

T se refiere a la variable global de temperatura del sistema. T es un valor que desciende con cada iteración en un factor constante al que nos referiremos como Factor de enfriamiento.

La cualidad de esta función es que la probabilidad de explorar una nueva solución que es peor que la solución actual toma en cuenta cuan peor es la nueva solución y además crece en relación inversa a la temperatura del sistema. Esto significa que inicialmente nuestro sistema tendera a no permanecer demasiadas veces en una misma solución, permitiéndose explorar el espacio de soluciones, y a medida que la temperatura decrece se tendera a permanecer en aquellas soluciones que hayan sido las mejores.

Experimentos y resultados

Dos variables resultan determinantes para el algoritmo: la temperatura y Cooling rate o coeficiente de enfriamiento. Siendo T - (T\*CR)<<T

Otras variables como el espacio de vecindad se mantuvieron siempre fijo a 1

Puede observarse una relación entre la temperatura, el CR y el número de iteraciones. Es evidente que mientras más alta la temperatura más iteraciones realiza el algoritmo, lo inverso ocurre con el CR, mientras más pequeño resulta en más iteraciones. Los resultados parecen indicar además que mientras más pequeña sea la disminución de temperatura por iteración (CR), mejores serán los resultados.

|  |  |  |  |
| --- | --- | --- | --- |
| T = 1 Cooling rate = 0.999 | | T = 1 Cooling rate = 0.99 | |
| Subsecuencias 10 | Iteraciones | Subsecuencias 10 | Iteraciones |
| 8 | 479 | 7 | 63 |
| 8 | 511 | 7 | 89 |
| 8 | 465 | 7 | 95 |
| 8 | 489 | 7 | 62 |
| 8 | 476 | 8 | 62 |
| 8 | 439 | 7 | 64 |
| 8 | 496 | 8 | 56 |
| 8 | 425 | 7 | 79 |
| Media=8 | Media=472,5 | Media=7,25 | Media=71,25 |
| T = 5 Cooling rate = 0.999 | | **T = 5 Cooling rate = 0.99** | |
| Subsecuencias 10 | Iteraciones | Subsecuencias 10 | Iteraciones |
| 8 | 1358 | 7 | 154 |
| 8 | 1446 | 8 | 148 |
| 8 | 1334 | 7 | 162 |
| 8 | 1319 | 7 | 158 |
| 8 | 1295 | 7 | 148 |
| 7 | 1465 | 7 | 143 |
| 8 | 1230 | 8 | 128 |
| 8 | 1251 | 8 | 159 |
| Media=7,87 | Media=1337,25 | Media=7,37 | Media=150 |

Conclusiones

Se concluye que el recocido simulado es una técnica eficiente y de utilidad para realizar búsquedas en un espacio muy amplio que requiere de optimizar el espacio de búsqueda.