Práctica III

Simulación y Metaheurísticas

Metaheurísticas

José Ignacio Escribano

Móstoles, 16 de mayo de 2016

Índice de tablas

Índice

1. Introducción

En esta práctica implementaremos una de las metaheurísticas planteadas para el problema del p-hubs.

2. Resolución de la práctica

A continuación mostraremos la implementación del recocido simulado para el problema del p-hub. El código se encuentra en el método estático `recocidoSimulado` de la clase `Práctica3`.

```
static Solución recocidoSimulado(Enfriamiento.TIPOS_ENFRIAMIENTO tipo, Solución sol_ini, InstanciaPHub instancia,
int temp_ini, int nrep, double alpha, long tiempo) {

    double tk = 0;

    Solución x = sol_ini;

    int i = 1;

    switch (tipo) {
        case BOLTZMANN: tk = Enfriamiento.crierioBoltzmann(temp_ini, alpha, i); break;
        case CAUCHY: tk = Enfriamiento.esquemaCauchy(temp_ini, alpha, i); break;
        case LUNDYyMEES: tk = Enfriamiento.esquemaCauchy(temp_ini, alpha, i); break;
        case DESCENSO_GEOMETRICO: tk = Enfriamiento.crierioBoltzmann(temp_ini, alpha, i);
    }

    long fin = System.currentTimeMillis() + tiempo;

    double delta;

    do {
        int m = 0;
        do {
            // Generamos número aleatorio entre 0 y número de soluciones de la vecindad - 1
            Random r = new Random();
            List<Solución> vecindad = PHub.generarVecindad(x, instancia);
```

```java
27              int n = r.nextInt(vecindad.size());
28
29              Solución y = vecindad.get(n);
30
31              delta = y.getObjetivo() - x.getObjetivo();
32
33              if (delta < 0) {
34                  x = y;
35              } else {
36                  double u = r.nextDouble();
37                  if (u <= Math.exp(-delta / tk)) {
38                      x = y;
39                  }
40              }
41              m++;
42          } while (m != nrep);
43
44          switch (tipo) {
45              case BOLTZMANN: tk = Enfriamiento.crierioBoltzmann(temp_ini,
   ↪ alpha, i); break;
46              case CAUCHY: tk = Enfriamiento.esquemaCauchy(temp_ini, alpha,
   ↪ i); break;
47              case LUNDYyMEES: tk = Enfriamiento.esquemaCauchy(temp_ini,
   ↪ alpha, i); break;
48              case DESCENSO_GEOMETRICO: tk =
   ↪ Enfriamiento.crierioBoltzmann(temp_ini, alpha, i);
49          }
50          i++;
51      } while (System.currentTimeMillis() <= fin);
52
53      return x;
54  }
```

Esta función recibe un tipo de enfriamiento, una solución inicial, una instancia, una temperatura inicial, un número de repeticiones, un valor (parámetro de la función de enfriamiento) y el tiempo de ejecución del programa, en milisegundos.

En la línea 6 se establece que `x` es la solución inicial, y desde la línea 10 a la 16 se establece qué función de enfriamiento se utilizará para el recocido simulado.

Se elige una solución al azar de la vecindad de `x`, que llamaremos `y`, y se calcula la diferencia entre la solución nueva y la inicial. A este valor lo llamemos `delta`. Si `delta` es menor que cero, es decir $\delta = f(y) - f(x) < 0 \iff f(y) < f(x)$, es decir, se mejora la solución actual y se guarda. En caso contrario (si no se mejora la solución) se genera un número aleatorio en el intervalo $(0,1)$. Si este número aleatorio es menor que $e^{-\delta/t_k}$, se guarda la solución `y`. Notar que t_k es el valor de la temperatura en la iteración k. Este proceso se repite `nrep` veces, y todo lo anterior se repite durante el tiempo especificado. Éste es nuestro criterio de parada.

En cuanto la funciones de enfriamiento se ha definido una clase `Enfriamiento` dentro de la clase `Práctica3`. Se han implementado cuatro funciones de enfriamiento, que vienen dadas por las siguientes ecuaciones:

- Descenso geométrico: $t_{i+1} = \alpha t_i,\ \ \alpha \in [0.8, 0.99]$
- Criterio de Boltzmann: $t_i = \dfrac{t_0}{1 + \log i}$
- Esquema de Cauchy $t_i = \dfrac{t_0}{1 + i}$
- Lundy y Mees: $t_{i+1} = \dfrac{t_i}{1 + \beta t_i}$, con β muy pequeño

El código del criterio de Boltzmann se muestra a continuación:

```java
public static double crierioBoltzmann(double t0, double alpha, int i) {
    return t0 / (1 + Math.log(i));
}
```

Esta función recibe tres parámetros: la temperatura inicial, el valor del parámetro α y la iteración i. Se devuelve directamente el valor dado por la fórmula indicada anteriormente.

De forma similar se implementan los demás métodos.

2.1. Análisis de los resultados

Para evaluar este método ejecutamos durante 1 minuto (60000 milisegundos) cada una de las instancias con cada uno de las funciones de enfriamiento.

Los parámetros usados han sido los siguientes:

- Temperatura inicial = 10000
- $\alpha = \begin{cases} 0.99, & \text{si es descenso geométrico} \\ 0.001, & \text{si es Lundi y Mees} \end{cases}$
- Número de repeticiones = 250
- Tiempo de ejecución (condición de parada) = 60000 (1 minuto por instancia)

Tabla 1: Resultados de la ejecución del recocido simulado

Instancia	Función de enfriamiento				Media
	Descenso geométrico	Criterio de Boltzmann	Esquema de Cauchy	Lundi y Meer	
1	932.4007	1160.496	991.9606	831.4168	979.0685
2	986.0779	1054.8835	1105.693	883.8535	1007.6270
3	974.4974	994.4498	993.2027	825.3837	946.8834
4	1039.086	980.6139	1150.0283	1024.4868	1048.5538
5	929.1119	1037.8508	949.8733	712.1517	907.2469
6	977.0404	787.7134	1056.724	870.6946	923.0431
7	1021.4352	976.2935	982.2888	945.6088	981.4066
8	1056.8216	1057.9631	825.4254	977.5117	979.4304
9	981.1257	1074.5005	1041.2094	896.6903	998.3815
10	836.5384	832.3738	950.8749	757.631	844.3545
Media	973.4135	995.7138	1004.7280	872.5429	

Los resultados de la ejecución se pueden ver en la Tabla 1. En amarillo se muestra el mejor valor de la función objetivo de cada instancia.

Se puede observar que no se obtiene ninguna mejor instancia con el método del descenso geométrico, ni con el esquema de Cauchy, en las instancias 4 y 6, el mejor método es el criterio de Boltzmann, y en las instancias 1, 2, 3, 5, 7, 8, 9 y 10 se obtiene la mejor solución con el método de Landi y Mees. Es decir, con el método de Lundi y Meer se obtienen 8 mejores instancias, con el criterio de Boltzmann se obtienen 2 instancias y con el esquema de Cauchy y el descenso geométrico no se obtiene ninguna instancia. Además, el método de Lundi y Meer tiene la mejor media de valores de la función objetivo. Parece que el mejor de los cuatro métodos es el Lundi y Meer (con los parámetros definidos anteriormente).

Necesitamos conocer cuál es mejor valor de β que minimiza el valor de las instancias. Para ello, repetimos el proceso anterior para distintos parámetros de β, desde 1 hasta 10^{-10}. Los resultados se pueden ver en la Tabla 2.

Se puede observar que el parámetro con mayor número de instancias con la solución menor es $\beta = 0.1$, que parece un buen valor para este método, aunque se deberían realizar prueba de β en el intervalo $[0.1, 1]$ para comprobar si se consiguen soluciones mejores que las actuales.

Tabla 2: Resultados de la ejecución del recocido con el método simulado con la función de enfriamiento de Lundi y Meer para distintos valores del parámetro β

Instancia	$\beta = 1$	$\beta = 0.1$	$\beta = 0.01$	$\beta = 0.001$	$\beta = 10^{-4}$	$\beta = 10^{-5}$	$\beta = 10^{-6}$	$\beta = 10^{-7}$	$\beta = 10^{-8}$	$\beta = 10^{-9}$	$\beta = 10^{-10}$
1	708.4036	708.4036	708.4036	819.0216	888.7935	1537.5536	983.174	941.2269	960.708	1130.6897	1031.907
2	829.3524	829.3524	845.3045	959.4793	871.7137	997.4544	1044.877	1119.9853	1341.9047	1103.3533	889.3522
3	758.2295	758.2295	781.0541	866.4344	827.2772	857.758	931.0008	947.5278	1010.2579	923.8711	909.0433
4	851.259	764.0963	771.3152	852.2911	878.914	1113.301	929.5215	1178.9504	912.6937	1191.0961	1149.1484
5	671.1279	681.3047	671.3733	862.9865	785.1307	1035.2035	802.9528	904.586	989.3504	890.3637	851.9101
6	663.1406	663.1406	682.945	738.7844	857.4963	726.1366	826.3958	1178.5185	814.0449	883.9562	796.0767
7	807.82	793.2306	798.0115	826.2132	876.9625	887.4507	981.5936	1119.5116	832.6994	941.605	958.4476
8	829.1049	822.5995	827.965	846.9209	876.0916	962.5802	1004.3007	923.5164	954.9392	887.6405	917.5369
9	857.4355	840.0549	873.247	919.9782	1255.2681	912.541	1211.4791	1461.8266	962.1836	1066.2949	995.2715
10	824.7164	722.024	742.7818	958.8561	914.895	762.6091	909.6664	876.8673	1147.981	794.522	820.6012

3. Conclusiones

En esta práctica hemos visto cómo implementar una de las metaheurísticas vistas en clase: el recocido simulado. También hemos visto que existen diversos métodos para establecer el enfriamiento, dando todos ellos a distintos resultados, ya que la mayoría es una familia uniparamétrica. Esto hace que sea indispensable probar con distintas metaheurísticas (y sus distintos parámetros) y escoger la que mejor resultados obtenga para el problema en concreto.