Tareas 3 y 7
Heurísticas para SAT y algoritmos aleatorizados

Pamela Jocelyn Palomo Martínez

1. Búsqueda dispersa

La meta-heurística elegida para intentar resolver instancias de SAT o, en su defecto, de MAX-SAT fue una búsqueda dispersa (Scatter search, SS). El SS se resume en el Algoritmo 1.

Algorithm 1 Búsqueda dispersa

Input:
p ▷ Tamaño del conjunto inicial de soluciones
r ▷ Tamaño del conjunto de referencia
$P \leftarrow$Construcción(p).
$P \leftarrow$Mejora(P)
$R \leftarrow$CrearConjuntoReferencia(P, r)
repeat
    $S \leftarrow$GenerarSubconjuntos(R)
    $R' \leftarrow$Combinación(S)
    $R' \leftarrow$Mejora(R')
    $R \leftarrow$ActualizarConjuntoReferencia(R, R')
until R no se actualiza
return La solución mejor evaluada del conjunto de referencia

Cada una de las componentes del algoritmo se explica a detalle a continuación:

- **Construcción(i).** Genera i soluciones de la siguiente manera: para cada variable se genera un número aleatorio en el conjunto $\{0,1\}$ proveniente de una distribución uniforme. Si el número aleatorio es cero, la varible es falsa; si es 1, la variable es verdadera.
- **Mejora(X).** Para cada solución del conjunto X se ejecuta el siguiente algoritmo: Si la variable i es falsa, se hace verdadera y si es verdadera, se hace falsa. Si el cambio aumenta el número de claúsulas satisfechas, se acepta y se fija $i = 1$, si no se hace $i = i + 1$. Al inicio del algoritmo i es igual a 1, el procedimiento termina cuando i excede al número de variables.
- **CrearConjuntoReferencia(P, r).** El conjunto de referencia es un subconjunto de r elementos del conjunto de soluciones iniciales. Los mejores $r/2$ y los peores $r/2$ elementos de P evaluados con respecto al número de claúsulas que satisfacen conforman el conjunto de referencia.
- **GenerarSubconjuntos(R).** De R se generan $r = |R|$ subconjuntos. El conjunto de referencia es ordenado de manera descendiente con respecto al número de claúsulas que satisface, entonces el primer subconjunto estará formado por el par de soluciones que se encuentran en las posiciones 1 y $|R|$, respectivamente; el segundo subconjunto, por las soluciones en las posiciones 2 y $|R|-1$ y así, sucesivamente.
- **Combinación(S).** La combinación de cada par de soluciones (X_1, X_2) resulta en una nueva solución X_3 y se realiza de la siguiente manera: Si la variable i, si tiene el mismo valor en X_1 y X_2, su valor en X_3 será el mismo, si no, se asignará aleatoriamente.
- **ActualizarConjuntoReferencia(R, R').** Sea $r = |R|$. Los elementos de R' son agregados a R si son diferentes a cada elemento del mismo. Sea r' la nueva cardinalidad del conjunto de referencia, si $r' > r$, se eliminan los $r' - r$ elementos peor evaluados del conjunto. Es decir, R mantiene su cardinalidad durante todo el proceso. Si los elementos de R no cambian, entonces se dice que no se actualiza el conjunto.

2. Análisis de un algoritmo aleatorizado para fijar el tamaño del conjunto de referencia

El SS utiliza dos parámetros que deben ser calibrados, los cuales son el tamaño de la población inicial y el tamaño del conjunto de referencia. En esta sección se hace uso de un algoritmo aleatorizado para fijar el tamaño del conjunto de referencia.

Sea X un conjunto de variables booleanas y C un conjunto de claúsulas. Sin pérdida de generalidad, se asume que las claúsulas están en la forma normal conjuntiva. Para cada claúsula $i \in C$, sea n_i el número de variables que forman i.

Considere el algoritmo utilizado para generar las soluciones iniciales. Si n^* es el mínimo número de variables contenidas en alguna claúsula, es decir $n^* = \underset{i \in C}{min}\{n_i\}$, la probabilidad de que una claúsula arbitraria i sea satisfecha cumple la siguiente desigualdad:

$$P(C_i) \geq 1 - \frac{1}{2^{n^*}}. \tag{1}$$

En el SS nos interesa contar con soluciones iniciales que generen una gran cantidad de claúsulas satisfechas con la esperanza de que los movimientos de combinación de soluciones generarán una nueva que satisfaga todas las claúsulas (o la mayor cantidad posible). En particular, supongamos que queremos generar suficientes soluciones para asegurar con una probabilidad de 0.95 que cada claúsula será satisfecha por al menos una de las soluciones. Ahora bien, note que si $n^* \geq 5$, la probabilidad de que una sola solución satisfaga una claúsula es mayor que 0.96875. Sin embargo, para valores de n^* menores que 5 se tiene que generar más de una solución. La probabilidad de que dadas m soluciones, una claúsula sea satisfecha, está dada por:

$$P(C_i)_m \geq 1 - \frac{1}{2^{m \times n^*}}. \tag{2}$$

Con base a lo anterior, si $n^* = 2$, se deben generar al menos tres soluciones para que con una probabilidad de 0.984375 cada claúsula sea satisfecha por al menos una solución. Por otro lado, si n^* es igual a 3 o 4, se deben generar dos soluciones como mínimo.

El análisis anterior, nos permite determinar que no se necesita un conjunto de soluciones con gran cardinalidad para que este sea de buena calidad. Como en la literatura del SS se ha notado que una buena composición del conjunto de referencia contiene la mitad de las soluciones de buena calidad y la otra mitad son soluciones diversas, y como se ha mostrado que se requiren al menos tres soluciones para tener un conjunto de buena calidad, en la tarea 3 se fijó el tamaño del conjunto de referencia a seis. Esto es importante, ya que se reduce el esfuerzo computacional al realizar menos operaciones que si se eligiera un número mayor para el tamaño del conjunto de referencia.

La Tabla 1 contiene los promedios del número de claúsulas satisfechas en 10 ejecuciones del algoritmo por cada instancia, variando el tamaño del conjunto de referencia entre 6, 12, 24 y 48 soluciones y dejando fijo el tamaño del conjunto de soluciones iniciales en 100. El nombre de las instancias sigue el formato «tipo_formato_variables_clausulas», donde tipo es el tipo de problema 2SAT, 3SAT o SAT, el formato es CNF, variables es el número de variables y clausulas, el número de cláusulas. Las instancias marcadas en negrita indican los únicos casos en los que hubo una ligera mejora por aumentar el tamaño del conjunto de referencia.

Instancia	$r = 6$	$r = 12$	$r = 24$	$p = 48$
2_SAT_CNF_50_50	50	50	50	50
2_SAT_CNF_50_100	96	96	96	96
2_SAT_CNF_50_200	187	187	187	187
2_SAT_CNF_100_50	50	50	50	50
2_SAT_CNF_100_100	98.6	98.9	98.9	99
2_SAT_CNF_100_200	194.3	194.3	194.4	195
2_SAT_CNF_200_50	50	50	50	50
2_SAT_CNF_200_100	100	100	100	100
2_SAT_CNF_200_200	200	200	200	200
3_SAT_CNF_50_50	50	50	50	50
3_SAT_CNF_50_100	100	100	100	100
3_SAT_CNF_50_200	199.2	199.4	199.7	200
3_SAT_CNF_100_50	50	50	50	50
3_SAT_CNF_100_100	100	100	100	100
3_SAT_CNF_100_200	200	200	200	200
3_SAT_CNF_200_50	50	50	50	50
3_SAT_CNF_200_100	100	100	100	100
3_SAT_CNF_200_200	200	200	200	200
SAT_CNF_50_50	50	50	50	50
SAT_CNF_50_100	99	99	99	99
SAT_CNF_50_200	199	199	199	199
SAT_CNF_100_50	50	50	50	50
SAT_CNF_100_100	100	100	100	100
SAT_CNF_100_200	200	200	200	200
SAT_CNF_200_50	50	50	50	50
SAT_CNF_200_100	100	100	100	100
SAT_CNF_200_200	200	200	200	200

Cuadro 1: Promedio del número de claúsulas satisfechas en 10 corridas del algoritmo variando el tamaño del conjunto de referencia

Es importante recalcar que el tamaño del conjunto de referencia afecta al número de operaciones ejecutadas en las líneas 3, 5, 6, 7 y 8 del Algoritmo 1. Algunas veces, cuando se trabaja con metaheurísticas, puede convenir realizar más operaciones a cambio de obtener soluciones de mayor calidad; sin embargo, el análisis previo y los experimentos computacionales muestran que, aunque se incremente 2, 4 u 8 veces el tamaño del conjunto de referencia, la calidad de la solución no aumenta o aumenta de manera no considerable.