Lógica Computacional

Proyecto Final - Versión A (Haskell + MiniSAT)

Manuel Soto Romero José Alejandro Pérez Márquez Karla Denia Salas Jiménez
Erik Rangel Limón Dicter Tadeo García Rosas

Fecha de entrega: 09 de junio de 2024
Facultad de Ciencias UNAM

Objetivos

- Dar una breve introducción al problema de satisfacibilidad para la Lógica Proposicional SAT.
- Aprender a usar un solucionador SAT a través de Haskell.
- Dar solución a un problema reduciéndolo a un problema de satisfacibilidad en Lógica Proposicional y de esta manera, entender la relación entre algunos problemas del mundo real y el problema SAT.

Introducción

El *Teorema de Cook*[1] introduce el concepto de NP-completud, que establece lo siguiente:

> **Teorema 1. (Teorema de Cook)** *Un problema de decisión es NP-completo si pertenece a la clase de complejidad NP y si para todo problema en NP, éste puede ser reducido a él utilizando una transformación polinómica.*

El Teorema de Cook es de gran importancia, pues además de introducir el concepto de *NP*-completud, establece que el problema SAT es un problema *NP*-completo y de hecho, el primero en ser demostrado que pertenece a esta clase de complejidad.

Al ser un problema *NP*-completo el problema SAT, no existe un algoritmo eficiente que dada una instancia de este problema, pueda resolverlo. Sin embargo a lo largo de los años se han desarrollado sistemas capaces de *resolver* instancias del problema SAT de manera eficiente. A estos sistemas se les llama *solucionadores SAT*.

[1] También conocido como *Teorema de Cook-Levin* pues Leonid Levin lo demostró casi al mismo tiempo que Stephen Cook.

El problema SAT

En su forma más general, el problema SAT busca responder lo siguiente:

Dada una fórmula proposicional φ, ¿existe un estado de las variables $\mathscr{I}$ de tal manera que suceda que $\mathscr{I}(\varphi) = 1$?

A simple vista, podría parecer que el problema SAT es relevante sólo para el área de la Lógica Computacional, Teoría de la Complejidad y para las Ciencias de la Computación en general, no obstante, las aplicaciones en problemas de interés práctico son diversas, problemas tan simples como un sudoku o tan complicados como verificación de hardware de planeación (control de tráfico aéreo, logística, etc.) así como criptoanálisis, pueden ser resueltos mediante un solucionador SAT.

Como se recordará, uno de los motivos por los cuales se estudia a la lógica, es poder modelar problema, mediante un lenguaje, de manera que sea posible razonar formalmente sobre estos. A modo de ejemplo, consideremos el *problema del palomar*, que establece que es imposible colocar n palomas en m palomares de tal manera que cada paloma esté en un palomar diferente cuando $n > m$. Para los humanos, resolver este problema es trivial, sin embargo, para un solucionador SAT puede llegar a ser un problema muy complejo. Para codificar el problema anterior en una fórmula proposicional primero se deben definir las variables proposicional y su significado. En el caso del problema del palomar, sea $p_{i,j}$ la variable proposicional que representa que la i-ésima paloma se encuentra en el j-ésimo palomar. Ahora lo que se quiere, es poder codificar que:

- Cada paloma esté en un palomar

$$\bigwedge_{1\leq i\leq n} \bigvee_{1\leq j\leq m} p_{i,j}$$

  Con esta fórmula se está especificando que para cada paloma i, ésta tiene que estar en al menos un palomar.

- No puede haber dos palomas en el mismo palomar

$$\bigwedge_{1\leq j\leq m} \bigwedge_{1\leq i\leq n} \bigwedge_{1\leq k<i} \neg p_{i,j} \vee \neg p_{k,j}$$

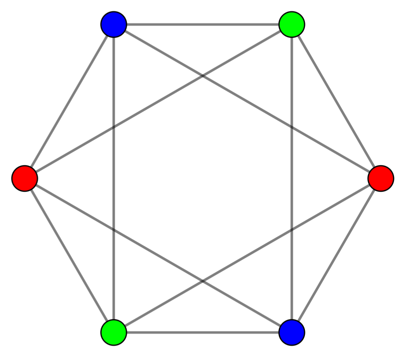
  De manera que la codificación del problema del palomar consiste en la conjunción de las dos fórmulas descritas anteriormente. Entonces, para determinar si dada una instancia del problema del palomar ésta tiene solución, lo que se haría es codificar la instancia en una fórmula proposicional y con la ayuda de un solucionador SAT determinar si es satisfacible, si lo es entonces el modelo que haga satisfacible a la fórmula también proporcionará la solución, de lo contrario no existe solución.

Ejercicio 1. Investigar en qué consiste en problema SAT, más allá de las notas de clase. Dar ejemplos de problemas relevantes que se hayan resuelto reduciéndo a este problema. Ojo: No es un resumen, ni se aceptará que copien y peguen respuestas. Debe citar absolutamente todas sus fuentes. Debe ser una aplicación real, citada en algún artículo.

Coloración de gráficas

El problema de la k-coloración de gráficas consiste en, dada una gráfica G con n vértices y m aristas, verificar si existe una asignación de k colores de los vértices de la gráfica de manera que para cada dos vértices adyacentes el color asignado sea diferente.

Por ejemplo, la siguiente gráfica está 3-coloreada.



Ejercicio 2. Dar una codificación apropiada para el problema de la k-coloración para una gráfica con v vértices y a aristas.

Ejercicio 3. Implementar una biblioteca en HASKELL para trabajar con gráficas no dirigidas. La representación queda a elección del grupo de alumnos.

Ejercicio 4. Definir una función `kColoracion :: Int → Grafica → [Coloracion]` que dado un número k y una gráfica, devuelva todas las k-coloraciones que pueden realizarse con dicha gráfica. Si la lista final es vacía, significa que la gráfica no puede ser k-coloreada.

Para resolver este problema:

1. Deberán usar su biblioteca de gráficas y la codificación del Ejercicio 2 para construir una fórmula proposicional que represente la noción *tener una k-coloración.*
2. Usando la biblioteca `MiniSat`, alimentar al solucionador SAT con la fórmula generada por el punto uno.
3. Usar la solución generada por el SAT solver para construir las k-coloraciones.

Entregables

1. **Reporte de investigación** con la siguiente estructura
   - Título propuesto
   - Autores
   - Institución
   - Resumen
   - Preliminares
     - Respuesta al ejercicio 1 donde hablen de MiniSAT y la aplicación que hayan encontrado. Debe ser una aplicación real, citada en algún artículo.
   - Implementación
     - Descripción del problema y su solución
     - Descripción de hallazgos, desafíos y aprendizajes encontrados durante la implementación.
   - Conclusiones
     - Analizar y comparar la eficacia de su solución.
   - Referencias: En caso de no incluirlas o citar incorrectamente se considerará plagio. Todo trabajo debe ser de su autoría **NO COPIAR Y PEGAR LOS RESULTADOS DE LA INVESTIGACIÓN.**
2. **Archivos** con la implementación. Puede ser un archivo comprimido pero deben tener todo ordenado y limpio.
3. **Video** donde expongan su trabajo, sus hallazgos, expliquen la solución que implementaron y ejemplos que consideren. El video debe tener una duración máxima de 10 minutos. **SE DEBERÁ AÑADIR EN CLASSROOM LA LIGA AL VIDEO, NO EL VIDEO DE LO CONTRARIO SE SATURARÁ LA PLATAFORMA Y DEJARÁ DE RECIBIR ENTREGAS.**

Evaluación

- Investigación sobre el problema SAT 20 %
- Investigación sobre aplicación del problema SAT 25 %
- Implementación práctica 25 %
- Análisis y documentación 20 %
- Presentación y comunicación de resultados 10 %