Informe del proyecto 3

González Isaac .López, Javier. Márquez, Nabil

December 11, 2016

Para la resolver los problemas propuestos se implementaron en python dos scripts, uno para crear todas las clausuas necesarias para generar las aristas que crearan la figura propuesta y un script para la visualización gráfica de la misma.

En main.py se encuentran implementadas las siguientes clases:

- Clause: repesentación de una cláusula genérica con coordenadas i,j. Permite operaciones como negar una proposición o agregarla al universo de expresiones.
- Q(i,j,d): cláusula que indica que la arista de la dirección d en la celda i,j debe aparecer en la figura. Además provee un método que implementa la cláusula 0 (unificación de representaciones), la cual lleva todas las aristas a una nueva cláusula con dirección sur o dirección este de ser posible.
- IjClause: cláusula de dos coordenadas con tipo
- Z(i,j): cláusula que indica que la celda i,j es interna al loop
- R(i,j,x,y): predicado que indica que desde la celda i,j se puede alcanzar a la celda x,y. Cada celda se alcanza a si misma

Adicionalmente se creo el enum Dir para representar los puntos cardinales.

En draw.py se encuentra implementado con la biblioteca pygames una representación gráfica de la salida de main.py

La solución implementada en main.py lee los casos provistos e itera las posiciones del mapa recibido agregando predicados por cada restricción númerica de aristas, también son agregadas las definciones de celdas interiores y exteriores, celdas alcanzables y por último casos bordes de celdas int. y ext (bordes y esquinas).

Cada nuevo predicado visto se agrega al universo de predicados, ya que para poder ejecutar el programa minsat se requiere un identificador único para cada uno y además es necesario poder recuperar este valor después de obtener la solcuón.

La salida del programa se escribe en formato SATLIB, el cual incluye un header indicando el formato en el que se encuentra la entrada a minsat (cnf), la cantidad de variables diferentes utilizadas y el número de predicados.

Desde python se obtiene la salida del minsat y se decodifica la respuesta, la cual es una lista de números positivos o negativos que indican que predicados son ciertos y falsos. En el script se eliminan los predicados que no se refieren a aristas en el problema y se ignoran las aristas negadas (que no se encuentran en la salida). Por último se imprime en pantalla la solución al problema indicando si una arista está o no iniciando por la primera fila de celdas colocando primero las aritas norte, luego las oeste (una por fila), las este y por último las de dirección sur.

Al correr el programa mediante la orden `python3 main.py < input.txt` se obtuvieron los resultados que se encuentran en *resultados.txt*, los casos marcados como UNSATISFIABLE no pudieron ser resueltos por choques entre los predicados suministrados a minsat.

Para correr el programa gráfico es necesario instalar pygames y utilizar el comando `python draw.py < example_output.txt`

El programa principal main.py hace uso de miniSat, el cual es el ejecutable que puede ser encontrado en la página oficial http://minisat.se/. La versión adjunta en el problema es para linux de 64, puede ser sustituida por cualquier otra mientras mantenga el nombre *miniSat* y sea ejecutable (ya que es lanzado desde python).

main.py crea un archivo intermedio int_file para poder comunicarse con minSat.

En este archivo comprimido también se pueden observar algunas imagenes generadas por draw.py a partir de nuestras soluciones proporcionadas.