Trabajo Grupal 1

Modelado y resolución de problemas usando lógica

Coach: Javier Oliva (S1) / Ismael Correa (S2)

Dado un conjunto de proposiciones P, una fórmula $\alpha \in \mathcal{L}(P)$ está en forma normal conjuntiva (CNF) si y sólo si es una conjunción de cláusulas, donde cada cláusula es a su vez una disyunción de literales. Asimismo, una formula α está en 3-CNF si y sólo si cada cláusula está compuesta por exactamente 3 literales distintos. Por ejemplo, la fórmula $(x_1 \vee \neg x_2 \vee x_2) \wedge (\neg x_1 \vee x_2 \vee \neg x_3) \wedge (x_1 \vee x_2 \vee \neg x_3)$ está en 3-CNF.

Considere una representación para fórmulas en 3-CNF donde cada cláusula es representada por una lista de números enteros, representando negaciones de un literal con el signo negativo. Por ejemplo, la lista `[1, -2, -3]` representa la cláusula $(x_1 \vee \neg x_2 \vee \neg x_3)$. Para la representar la conjunción de cláusulas que componen la fórmula, se construye una lista de listas de enteros. Por ejemplo, la lista `[[1, -2, 3], [-1, 2, -3], [1, 2, -3]]` representa la formula en 3-CNF expuesta en el párrafo anterior.

1. Se le pide desarrollar un programa que, dada una cantidad de cláusulas $n \in \mathbb{N}$ y un conjunto de variables proposicionales $P = \{x_1, ..., x_k\}$ $(k \geq 3)$, genere formulas aleatorias en 3-CNF, siguiendo la representación de listas. Por ejemplo, una fórmula al azar con 4 cláusulas y 5 literales podría ser:

   $$(x_1 \vee \neg x_3 \vee x_5) \wedge (\neg x_2 \vee x_5 \vee \neg x_4) \wedge (x_3 \vee \neg x_1 \vee x_2) \wedge (\neg x_5 \vee \neg x_3 \vee x_2)$$

   y estaría representada por la lista `[[1, -3, 5], [-2, 5, -4], [3, -1, 2], [-5, -3, 2]]`.
2. Desarrolle un algoritmo de fuerza bruta que dada una formula en 3-CNF, representada como una lista de listas de enteros, verifique la satisfactibilidad de la misma. Su algoritmo debe retornar `false` en caso de que la fórmula sea insatisfactible, y `true`, acompañada de una valuación testigo, en caso contrario.
3. Haciendo uso del generador de fórmulas aleatorias en 3-CNF de la primera parte, construya fórmulas al azar para una cantidad de variables $k \in [3...x]$, con x en un rango adecuado (determinado por Ud.). Haciendo variar la cantidad de clausulas en cada caso, estudie cual es una razón adecuada de clausulas a variables, para obtener tanto fórmulas satisfacibles como insatisfacibles con similar frecuencia. Nota: No se le pide formalidad en los argumentos, solo un análisis experimental. Plantee una hipótesis y rescate observaciones mediante sus resultados experimentales.
4. Investigue cómo usar el SAT Solver `Minisat22` (o bien otros SAT Solvers disponibles a través de la interfaz PySat de Python) y compárelo con el SAT solver hecho por Ud. en el punto 2, por ejemplo, respecto a tiempo ejecución. Para esto genere fórmulas con diferentes cantidades de variables, y una cantidad de clausulas en

cada una coherente con sus descubrimientos realizados en la sección anterior, y ejecute ambos SAT Solvers para todas ellas. Realice un gráfico con el tiempo de ejecución de ambos SAT solvers como función de la cantidad de variables en las fórmulas generadas.

Sugerencias y Observaciones:

- La forma en que PySat representa fórmulas en CNF es idéntica a las del generador de fórmulas solicitado.
- El módulo StopWatch en Python le puede ser útil para cronometrar el tiempo de ejecución de su código.
- Para obtener mayor robustez en las mediciones de tiempo realizadas, se sugiere cronometrar su código. Dada una cantidad k de variables, considere diversas fórmulas con k variables y mida repetidamente el tiempo que toma con cada una. Considere la media o mediana de los tiempos obtenidos.
- Puede usar los siguientes dos resultados:
  - Dado un conjunto de proposiciones P tales que $|P| \geq 3$, si $\hat{\mathcal{L}}(P)$ es el lenguaje de fórmulas en 3-CNF sobre P, entonces toda fórmula insatisfacible en $\hat{\mathcal{L}}(P)$ debe tener al menos $2^3 = 8$ cláusulas.
  - Si $|P| \geq k$, y $\hat{\mathcal{L}}(P)$ es el lenguaje de fórmulas en K-CNF sobre P, entonces toda fórmula insatisfacible en $\hat{\mathcal{L}}(P)$ debe tener al menos 2^k cláusulas.