Complejidad Computacional

El problema 3-Satisfiability

3SAT

Alberto Delgado Soler
Germán Alfonso Teixidó
Tomás González Martín
Pier Paolo Tarasco

November 6, 2019

1 Introduction

1.1 Description of the problem

The 3-satisfiability it's a special case of the satisfiability (SAT), in which each clause has exactly 3 literals. The SAT problem consist of verifying if, given a boolean expression with variables and without quantifiers, there is an assignment of values to the variables such that the expression evaluates to true.

1.2 SAT to 3-SAT

Let $U = \{u_1, u_2, ..., u_n\}$ be a set of variables, and $C = \{c_1, c_2, ..., , c_m\}$ a set of clauses, we would like to create a new set C' of clauses of three literals in a new set U', such that C' it's satisfiability if and only if C is satisfiability. To construct C' let's describe different cases, based on the number of variables in U and the number of variables needed to complete the triplets. Suppose that C contains k literals:

- Case 1. If $k = 1$ then $C_i = z_i$. In this case we need two more variables v_1 and v_2 to obtain four new clauses of three literals each.

  $$C' = \{v_1, v_2, z1\}, \{v_1, \overline{v_2}, z_1\}, \{\overline{v_1}, v_2, z1\}, \{\overline{v_1}, \overline{v_2}, z_1\}$$

  The reason of this is that v_1 as v_2 can be true or false. Because of this we have to represent each case (including the negated literals).

- Case 2. If $k = 2$ then $C_i = \{z_1, z_2\}$. We already have two literals, we only need one more, v_1. Thus, we add the two possible values of the new literal.

  $$C' = \{v_1, z_1, z_2\}, \{\overline{v_1}, z_1, z_2\}$$

- Case 3. If we have $k = 3$ we don't need any extra literal. The SAT problem already is in 3SAT.

- Case 4. If $k > 3$ we have to create "$k - 3$" additional variables, increasing the number of clauses to $k - 2$. The set C' will be made of three parts. The first part of the set, formed by the first two k literals, adding the first extra variable.

  $$\{z_1, z_2, v_1\}$$

  The other triplets (except the last one) will be in the form

  $$\{\overline{v_i}, z_{i+2}, v_{i+1}\}$$

  At last, the set is completed by:

  $$\{\overline{v_{k-3}}, z_{k-1}, z_k\}$$

  Thus, for the fourth case, where $k > 3$, we have the 3-SAT problem with this form:

  $$C' = \{z_1, z_2, v_1\} \cup \{\overline{v_i}, z_{i+2}, v_{i+1}\} \cup \{\overline{v_{k-3}}, z_{k-1}, z_k\}$$

1.3 Developed Code

The code, developed in the programming language *C++*.

1.3.1 Structure of the program

- */include*: files **.h* which contains the declarations of the classes utilized in the program.
- */obj*: contains the executable of the program, besides the input and output files, united to the object files (**.o**).
- */src*: files with the extension **.cpp* which contains the definition of the classes, and the main program.

1.3.2 Description of the main files

- *SAT*: Contains the formal definition of the SAT problem.
- *SAT3*: Being a special case of the SAT, inherit from it, implementing the concatenations of the clauses described before.
- *SAT2SAT3*: *SAT to 3SAT* Converts the problem from SAT to 3SAT utilizing the capacity of concatenating *SAT3*
- *input*: file which presents the problem in the conjunctive normal form (CNF) following the standard, where:
  - The first lines can have a C follow by a comment on each line.
  - After the letter p followed by cnf and then the number of the number of variables and the number of clauses.
  - Then there must be a line for each clause and, in each clause, the numbers of the variables that are in the clause are placed (with nothing or with a "-" in front to indicate that it is denied) and at the end of the clause the one "0" to indicate that the clause was terminated.