CS228 Logic for Computer Science 2020

Lecture 12: Encoding into SAT problem

Instructor: Ashutosh Gupta

IITB, India

Compile date: 2020-02-06

Topic 12.1

Encoding in SAT

SAT encoding

Since SAT is a NP-complete problem, therefore any NP-hard problem can be encoded into SAT in polynomial size.

Therefore, we can solve hard problems using SAT solvers.

We will look into a few interesting examples.

Objective of an encoding.

- Compact encoding (linear if possible)
- Redundant clauses may help the solver
- Encoding should be "compatible" with CDCL

Encoding into CNF

CNF is the form of choice

- Most problems specify collection of restrictions on solutions
- Each restriction is usually of the form

$$\text{if-this} \Rightarrow \text{then-this}$$

The above constraints are naturally in CNF.

"Even if the system has hundreds and thousands of formulas, it can be put into CNF piece by piece without any multiplying out"

– Martin Davis and Hilary Putnam

Exercise 12.1

Which of the following two encodings of $ite(p, q, r)$ is in CNF?

1. $(p \wedge q) \vee (\neg p \wedge r)$
2. $(p \Rightarrow q) \wedge (\neg p \Rightarrow r)$

Coloring graph

Problem:

color a graph$(\{v_1, \ldots, v_n\}, E)$ with at most d colors such that if $(v_i, v_j) \in E$ then color of v_i is different from v_j.

SAT encoding

Variables: p_{ij} for $i \in 1..n$ and $j \in 1..d$. p_{ij} is true iff v_i is assigned jth color.
Clauses:

- Each vertex has at least one color

$$\text{for each } i \in 1..n \qquad (p_{i1} \vee \cdots \vee p_{id})$$

- if $(v_i, v_j) \in E$ then color of v_1 is different from v_2.

$$(\neg p_{ik} \vee \neg p_{jk}) \qquad \text{for each } k \in 1..d, \quad (v_i, v_j) \in 1..n$$

Exercise 12.2

a. Encode: "every vertex has at most one color."

b. Do we need this constraint to solve the problem?

Pigeon hole principle

Prove:
if we place $n+1$ pigeons in n holes then there is a hole with at least 2 pigeons

The theorem holds true for any n, but we can prove it for a fixed n.

SAT encoding
Variables: p_{ij} for $i \in 0..n$ and $j \in 1..n$. p_{ij} is true iff pigeon i sits in hole j.
Clauses:

- Each pigeon sits in at least one hole

$$\text{for each } i \in 0..n \qquad (p_{i1} \vee \cdots \vee p_{in})$$

- There is at most one pigeon in each hole.

$$(\neg p_{ik} \vee \neg p_{jk}) \qquad \text{for each } k \in 1..n, \quad i < j \in 1..n$$

Topic 12.2

Cardinality constraints

Cardinality constraints

$$p_1 + ... + p_n \bowtie k$$

where $\bowtie \in \{<, >, \leq, \geq, =, \neq\}$

Encoding $p_1 + ... + p_n = 1$

- At least one of p_i is true

$$(p_1 \vee .... \vee p_n)$$

- Not more than one p_is are true

$$(\neg p_i \vee \neg p_j) \qquad i, j \in \{1, .., n\}$$

Exercise 12.3

a. What is the complexity of at least one constraints?

b. What is the complexity of at most one constraints?

Sequential encoding of $p_1 + .. + p_n \leq 1$

The earlier encoding of at most one is quadratic. We can do better by introducing auxiliary (fresh) variables.

Let s_i be a fresh variable to indicate that the count has reached 1 by i.

The following constraints encode $p_1 + .. + p_n \leq 1$.

$$
\begin{array}{rcl}
 & (p_1 \Rightarrow s_1) & \wedge \\
\bigwedge_{1<i<n}(\quad ((p_i \vee s_{i-1}) \Rightarrow s_i) & \wedge & (s_{i-1} \Rightarrow \neg p_i) \quad) \\
 & \wedge & (s_{n-1} \Rightarrow \neg p_n)
\end{array}
$$

If $p_i = 1$, for each $j \geq i$, $s_j = 1$.

If already seen a one, no more ones.

Exercise 12.4

a. Give a satisfying assignment when $p_3 = 1$ and all other ps are 0.

b. Give a satisfying assignments of s_is when all ps are 0.

c. Convert the constraints into CNF

Bitwise encoding of $p_1 + ... + p_n \leq 1$

Let $m = \lceil \ln n \rceil$.

- Consider bits $r_1, ..., r_m$
- For each $i \in 1...n$, let $b_1, ..., b_m$ be the binary encoding of $(i-1)$. We add the following constraints for p_i to be 1.

$$(p_i \Rightarrow (r_1 = b_1 \wedge ... \wedge r_m = b_m))$$

Example 12.1

Consider $p_1 + p_2 + p_3 \leq 1$.
$m = \lceil \ln n \rceil = 2$.

We get the following constraints.
$(p_1 \Rightarrow (r_1 = 0 \wedge r_2 = 0))$
$(p_2 \Rightarrow (r_1 = 0 \wedge r_2 = 1))$
$(p_3 \Rightarrow (r_1 = 1 \wedge r_2 = 0))$

$\rightsquigarrow$

Simplified
$(p_1 \Rightarrow (\neg r_1 \wedge \neg r_2))$
$(p_2 \Rightarrow (\neg r_1 \wedge r_2))$
$(p_3 \Rightarrow (r_1 \wedge \neg r_2))$

Exercise 12.5

What are the variable and clause size complexities?

Solving Sudoku using SAT solvers

Example 12.2

4	2	6	5	7	1	3	9	8
8	5	7	2	9	3	1	4	6
1	3	9	4	6	8	2	7	5
9	7	1	3	8	5	6	2	4
5	4	3	7	2	6	8	1	9
6	8	2	1	4	9	7	5	3
7	9	4	6	3	2	5	8	1
2	6	5	8	1	4	9	3	7
3	1	8	9	5	7	4	6	2

Sudoku

- *Variables:* $v_{i,j,k} \in \mathcal{B}$ *and* $i, j, k \in \{1, ..., 9\}$
- *If* $v_{i,j,k} = 1$, *column* i *and row* j *contains* k.
- *Value in each cell is valid:*

$$\sum_{k=1}^{9} v_{i,j,k} = 1 \qquad i, j \in \{1, .., 9\}$$

- *Each value used exactly once in each row:*

$$\sum_{i=1}^{9} v_{i,j,k} = 1 \qquad j, k \in \{1, .., 9\}$$

- *Each value used exactly once in each column:*

$$\sum_{j=1}^{9} v_{i,j,k} = 1 \qquad i, k \in \{1, .., 9\}$$

- *Each value used exactly once in each* 3×3 *grid*

$$\sum_{s=1}^{3} \sum_{r=1}^{3} v_{3i+r,3j+s,k} = 1 \quad i, j \in \{0, 1, 2\}, k \in \{1, .., 9$$

Encoding $p_1 + ... + p_n \leq k$

There are several encodings

- Generalized pairwise
- Sequential counter
- Operational encoding
- Sorting networks
- Cardinality networks

Exercise 12.6

Given the above encodings, how to encode $p_1 + ... + p_n \geq k$?

Generalized pairwise encoding for $p_1 + ... + p_n \leq k$

No $k + 1$ variables must be true at the same time.

For each $i_1, ..., i_{k+1} \in 1..n$, we add the following clause

$$(\neg p_{i_1} \vee \cdots \vee \neg p_{i_{k+1}})$$

Exercise 12.7

How many clauses are added for the encoding?

Sequential counter encoding for $p_1 + ... + p_n \leq k$

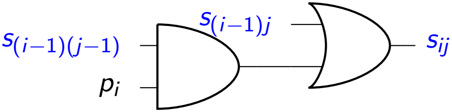
Let variable s_{ij} encode that the sum upto p_i has reached to j or not.

- Constraints for first variable p_1

$$(p_1 \Rightarrow s_{11}) \wedge \bigwedge_{j \in [2,k]} \neg s_{1j}$$

- Constraints for p_i, where $i > 1$

$$((p_i \vee s_{(i-1)1}) \Rightarrow s_{i1}) \wedge \bigwedge_{j \in [2,k]} ((\underbrace{p_i \wedge s_{(i-1)(j-1)}}_{add\ +1} \vee s_{(i-1)j}) \Rightarrow s_{ij})$$



Sequential counter encoding for $p_1 + ... + p_n \leq k$ (II)

- If the sum has reached to k at $i - 1$, no more ones

$$(s_{(i-1)k} \Rightarrow \neg p_i)$$

Exercise 12.8

What is the variable/clause complexity?

Operational encoding for $p_1 + ... + p_n \leq k$

Sum the bits using full adders. Compare the resulting bits against k.

Produces $O(n)$ encoding, however the encoding is not considered good for sat solvers, since it is not arc consistent.

Arc-consistency

Let $C(Ps)$ be a problem with variables $Ps = p_1, ..., p_n$.

Let $E(Ps, Ts)$ be encoding of the problem, where variables $Ts = t_1, ..., t_k$ are introduced by the encoding.

Definition 12.1

We say $E(Ps, Ts)$ is arc-consistent if for any partial model m of E

1. *If $m|_{Ps}$ is inconsistent with C, then unit propagation in E causes conflict.*
2. *If $m|_{Ps}$ is extendable to m' by local reasoning in C, then unit propagation in E obtains m'' such that $m''|_{Ps} = m'$.*

Unit propagation == Local reasoning

Example: arc-consistency

Example 12.3

Consider problem $p_1 + ... + p_n \leq 1$

An encoding is arc-consistent if

1. *If at any time two* p_i*s are made true, unit propagation should trigger unsatisfiability*
2. *If at any time* p_i *is made true, unit propagation should make all other* p_j*s false*

Commentary: The unit propagation in the encoding must mimic local reasoning of the problem. Intuitively, the encoding must not make the life of solver harder. If local reasoning can deduce something, then unit propagation must also deduce it. For more discussion, look in `http://minisat.se/downloads/MiniSat+.pdf` page5

Example: non arc-consistent encoding

Example 12.4

Consider problem $p_1 + p_2 + p_3 \leq 0$

Let us use full adder encoding

$$\begin{aligned} sum =& (p_1 \oplus p_2 \oplus p_3) \\ carry =& (p_1 \wedge p_2) \vee (p_2 \wedge p_3) \vee (p_1 \wedge p_3) \\ & \neg sum \wedge \neg carry \end{aligned}$$

Clearly p_1, p_2, p_3 *are* 0.

But, $\underbrace{\textit{the unit propagation without any decisions}}_{\textit{Local reasoning}}$ *does not give the model.*

Exercise 12.9

Does Tseitin encoding preserve the arc-consistency?

Topic 12.3

Input Format

DIMACS Input format

Example 12.5

Input CNF

```
c
c this is a comment
c
p cnf 4 6
-2   3  0
 1   3  0
-1   2  3 -4 0
-1  -2  0
 1  -2  0
 2  -3  0
```

Declares number of variables and clauses.

Each row is a clause ending with 0

Clause is $p_2 \vee \neg p_3$

Topic 12.4

Problems

SAT encoding: n queens

Exercise 12.10

Encode N-queens problem in a SAT problem.
N-queens problem: Place n queens in $n \times n$ chess such that none of the queens threaten each other.

SAT encoding: overlapping subsets

Exercise 12.11

For a set of size n, find a maximal collection of k sized sets such that any pair of the sets have exactly one common element.

SAT encoding: setting a question paper

Exercise 12.12

There is a datbase of questions with the following properties:

- *Hardness level* $\in$ {*Easy,Medium,Hard*}
- *Marks* $\in \mathbb{N}$
- *Topic* $\in \{T_1, ..., T_t\}$
- *LastAsked* $\in$ *Years*

Make a question paper with the following properties

- *It must contain x% easy, y% medium, and z% difficult marks.*
- *The total marks of the paper are given.*
- *The number of problems in the paper are given.*
- *All topics must be covered.*
- *No question that was asked in last five years must be asked.*

Write an encoding into SAT problem that finds such a solution. Test your encoding on reasonably sized input database. Devise a strategy to evaluate your tool and report plots to demonstrate the performance.

SAT encoding: finding a schedule

Exercise 12.13

An institute is offering m courses.

- *Each has a number of contact hours == credits*

The institute has r rooms.

- *Each room has a maximum student capacity*

The institute has s weekly slots to conduct the courses.

- *Each slot has either 1 or 1.5 hour length*

There are n students.

- *Each student have to take minimum number of credits*
- *Each student has a set of preferred courses.*

Assign each course slots and a room such that all student can take courses from their preferred courses that meet their minimum credit criteria. Write an encoding into SAT problem that finds such an assignment . Test your encoding on reasonably sized input. Devise a strategy to evaluate your tool and report plots to demonstrate the performance.

SAT encoding: synthesis by examples

Exercise 12.14

Consider an unknown function $f : \mathcal{B}^N \to \mathcal{B}$. Let us suppose for inputs $I_1, ..., I_m \in \mathcal{B}^N$, we know the values of $f(I_1), .., f(I_m)$.

a) Write a SAT encoding of finding a k-sat formula containing ℓ clauses that represents the function.

b) Write a SAT encoding of finding an NNF (negation normal form, i.e., $\neg$ is only allowed on atoms) formula of height k and width ℓ that represents the function.(Let us not count negation in the height.)

c) Write a SAT encoding of finding a binary decision diagram of height k and maximum width ℓ that represents the function.

Test your encoding on reasonably sized input. Devise a strategy to evaluate your tool and report plots to demonstrate the performance.

SAT encoding: Rubik's cube

Exercise 12.15

Write a Rubik's cube solver using a SAT solver

- *Input:*
  - *start state,*
  - *final state, and*
  - *number of operations k*
- *Output:*
  - *sequence of valid operations or*
  - *"impossible to solve within k operations"*

Test your encoding on reasonably many inputs. Devise a strategy to evaluate your tool and report plots to demonstrate the performance.

End of Lecture 12