Automated Reasoning 2018

Lecture 6: Encoding into SAT problem

Instructor: Ashutosh Gupta

IITB, India

Compile date: 2018-08-04

Content

- Encoding into SAT problem
- Encoding cardinality constraints
- DIMACS Input format

Topic 6.1

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 6.1

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

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 6.2

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 6.2

Cardinality constraints

Cardinality constraints

$$x_1 + ... + x_n \bowtie k$$

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

Encoding $x_1 + ... + x_n = 1$

- At least one of x_i is true

$$(x_1 \vee .... \vee x_n)$$

- Not more than one x_is are true

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

Exercise 6.3

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

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

Sequential encoding of $x_1 + .. + x_n \leq 1$

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 $x_1 + .. + x_n \leq 1$.

$$
\begin{array}{rcl}
(x_1 \Rightarrow s_1) & \wedge & \\
\bigwedge_{1<i<n}(\quad ((x_i \vee s_{i-1}) \Rightarrow s_i) & \wedge & (s_{i-1} \Rightarrow \neg x_i) \quad) \\
 & \wedge & (s_{n-1} \Rightarrow \neg x_n)
\end{array}
$$

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

If already seen a one, no more ones.

Exercise 6.4

a. Give a satisfying assignment when $x_3 = 1$ and all other xs are 0.

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

c. Convert the constraints into CNF

Bitwise encoding of $x_1 + ... + x_n \leq 1$

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 x_i to be 1.

$$(x_i \Rightarrow (r_1 = b_1 \land ... \land r_m = b_m))$$

Example 6.1

Consider $x_1 + x_2 + x_3 \leq 1$*.*
$m = \lceil \ln n \rceil = 2$*.*

We get the following constraints.

$(x_1 \Rightarrow (r_1 = 0 \land r_2 = 0))$
$(x_2 \Rightarrow (r_1 = 0 \land r_2 = 1))$
$(x_3 \Rightarrow (r_1 = 1 \land r_2 = 0))$

$\rightsquigarrow$

Simplified

$(x_1 \Rightarrow (\neg r_1 \land \neg r_2))$
$(x_2 \Rightarrow (\neg r_1 \land r_2))$
$(x_3 \Rightarrow (r_1 \land \neg r_2))$

Exercise 6.5

What are the variable and clause size complexities?

Encoding $x_1 + ... + x_n \leq k$

There are several encodings

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

Exercise 6.6

Given the above encodings, how to encode $x_1 + ... + x_n \geq k$?

Generalized pairwise encoding for $x_1 + ... + x_n \leq k$

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

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

Exercise 6.7

What many clauses are added for the encoding?

Sequential counter encoding for $x_1 + ... + x_n \leq k$

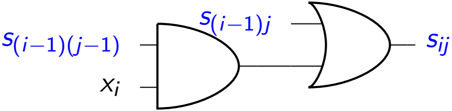
Let variable s_{ij} encode that the sum upto x_i has reached to j or not.

- Constraints for first variable x_1

$$(x_1 \Rightarrow s_{11}) \wedge \bigwedge_{j \in [2,k]} \neg s_{1j}$$

- Constraints for x_i, where $i > 1$

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



Sequential counter encoding for $x_1 + ... + x_n \leq k$ (II)

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

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

Exercise 6.8

What is the variable/clause complexity?

Cardinality constraints via sorted variables $O(n \ln^2 n)$

Let us suppose we have a circuit that produces sorted bits in decreasing order.

$$([y_1, .., y_n], Cs) := sort(x_1, ..x_n)$$

We can encode the cardinality constraints as follows

$$x_1 + .. + x_n \leq k \qquad \{y_{k+1} = 0\} \cup Cs$$
$$x_1 + .. + x_n \geq k \qquad \{y_k = 1\} \cup Cs$$

Exercise 6.9

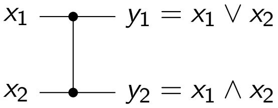
a. How to encode $x_1 + .. + x_n < k$

b. How to encode $x_1 + .. + x_n > k$

c. How to encode $x_1 + .. + x_n = k$

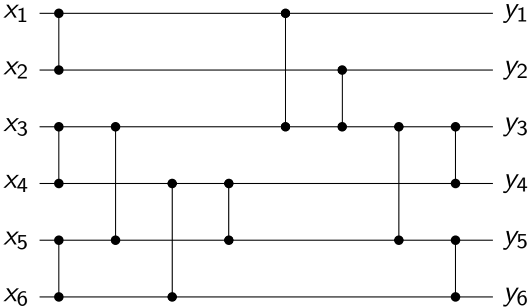
Sorting networks

The following circuit sorts two bits x_1 and x_2.



We can sort any number of bits by composing the circuit according to a sorting algorithm.

Example 6.2 *Sorting* 6 *bits using merge sort.*



Formal definition of sorting networks

base case:
$n = 1$

$$sort(x_1, x_2) \triangleq merge([x_1], [x_2]);$$

induction step:
$2n > 2$
Let,

sort/merge returns a vector of signals and a set of clauses.

$$([x'_1, ..., x'_n], Cs_1) := sort(x_1, .., x_n)$$
$$([x'_{n+1}, ..., x'_{2n}], Cs_2) := sort(x_{n+1}, .., x_{2n})$$
$$([y_1, ..., y_{2n}], Cs_M) := merge([x'_1, ..., x'_n], [x'_{n+1}, ..., x'_{2n}])$$

Then,

$$sort(x_1, .., x_{2n}) \triangleq ([y_1, .., y_{2n}], Cs_1 \cup Cs_2 \cup Cs_M)$$

Formally merge: odd-even merging network

Merge assumes that the input vectors are sorted.

base case:

$$merge([x_1],[x_2]) \triangleq ([y_1,y_2],\{y_1 \Leftrightarrow x_1 \wedge x_2, y_2 \Leftrightarrow x_1 \vee x_2\});$$

induction step:

Let

$$\begin{aligned}
([z_1,..,z_n],Cs_1) &:= merge([x_1,x_3...,x_{n-1}],[y_1,y_3,...,y_{n-1}]) \\
([z'_1,..,z'_n],Cs_2) &:= merge([x_2,x_4...,x_n],[y_2,y_4,...,y_n]) \\
([c_{2i},c_{2i+1}],CS^i_M) &:= merge([z_{i+1}],[z'_i]) \qquad \text{for each } i \in [1,n-1]
\end{aligned}$$

Then,

$$merge([x_1,...,x_n],[y_1,...,y_n]) \triangleq ([z_1,c_1,..,c_{2n-1},z'_n],Cs_1 \cup Cs_2 \cup \bigcup_i CS^i_M)$$

Cardinality Networks: a theoretical and empirical study, 2011, Constraints

Topic 6.3

Pseudo Boolean constraints

Pseudo-Boolean constraints

Let x_1,....,x_n be Boolean variables.

The following is a pseudo-Boolean constraint.

$$c_1x_1 + ...c_nx_n \leq c,$$

where $c_1, .., c_n, c \in \mathbb{Z}$.

How should we solve them?

- Using Boolean reasoning
- Using arithmetic reasoning

Here we will see the Boolean encoding for the constraints.

Observations on pseudo-Boolean constraints

- Replacing negative coefficients to positive

$$t - c_i x_i \leq c \quad \rightsquigarrow \quad t + c_i(\neg x_i) \leq c + c_i$$

- Divide the whole constraints by $d := gcd(c_1, ..., c_n)$.

$$c_1 x_1 + ... + c_n x_n \leq c \quad \rightsquigarrow \quad (c_1/d)x_1 + .. + (c_n/d)x_n \leq \lfloor c/d \rfloor$$

- Trim large coefficients to $c + 1$. Let us suppose $c_i > c$.

$$t + c_i x_i \leq c \quad \rightsquigarrow \quad t + (c+1)x_i \leq c$$

- Trivially true are replaced by $\top$. If $c >= c_i + ... + c_n$

$$c_1 x_1 + ... + c_n x_n \leq c \quad \rightsquigarrow \quad \top$$

- Trivially false are replace by $\perp$. If $c < 0$

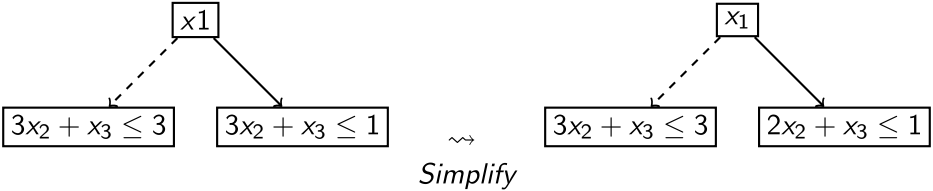
$$c_1 x_1 + ... + c_n x_n \leq c \quad \rightsquigarrow \quad \perp$$

Translating to decision diagrams

We choose a 0 and 1 for each variable to split cases and simplify.

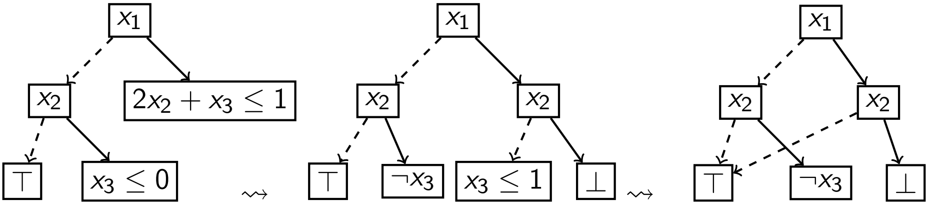
Example 6.3

Consider $2x_1 + 3x_2 + x_3 \leq 3$



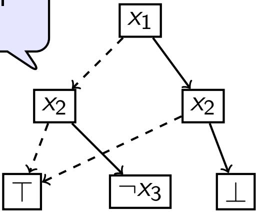
Example: translating to decision diagrams

We can split node left node $3x_2 + x_3 \leq 3$ further on x_2.



Example: decision diagrams to clauses

An auxiliary variable for each internal node



$$\rightsquigarrow \quad \begin{array}{l} (\neg x_1 \Rightarrow temp1) \land \\ (temp1 \land \neg x_2 \Rightarrow \top) \land \\ (temp1 \land x_2 \Rightarrow \neg x_3) \land \\ \\ (x_1 \Rightarrow temp2) \land \\ (temp2 \land \neg x_2 \Rightarrow \top) \land \\ (temp2 \land x_2 \Rightarrow \bot) \end{array}$$

Exercise 6.10

a. Simplify the clauses

b. Complexity of the translation from pseudo-Boolean constraints?

Topic 6.4

More problems

Solving Sudoku using SAT solvers

Example 6.4

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:*

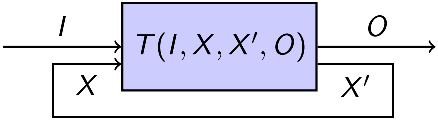
$$\sum_{j=1}^{9} v_{i,j,k} = 1 \qquad i, k \in \{1, .., 9\}$$

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

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

Bounded model checking

Consider a Mealy machine



- I is a vector of variables representing input
- O is a vector of variables representing output
- X is a vector of variables representing current state
- X' is a vector of variables representing next state

Prove: After n steps, the machines always produces output O that satisfies some formula $F(O)$.

Bounded model checking encoding

SAT encoding:

Variables:

- $I_0, \ldots, I_{n-1}$ representing input at every step
- $O_1, \ldots, O_n$ representing output at every step
- $X_0, \ldots, X_n$ representing internal state at every step

Clauses:

- Encoding system runs

$$T(I_0, X_0, X_1, O_1) \wedge \cdots \wedge T(I_{n-1}, X_{n-1}, X_n, O_n)$$

- Encoding property

$$\neg F(O_n)$$

If the encoding is unsat the property holds.

Topic 6.5

Input Format

DIMACS Input format

Example 6.5

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 6.6

Problems

SAT encoding: n queens

Exercise 6.11

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 6.12

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 6.13

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 6.14

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 6.15

Consider an unknown function $f : \mathcal{B}^N \rightarrow \mathcal{B}$. 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 6.16

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.

Search square of squares

Exercise 6.17

Squaring the square problem: "Tiling an integral square using only other smaller integral squares such that all tiles have different sizes."

Consider a square of size $n \times n$, find a solution of above problem using a SAT solver using tiles less than k.

Test your encoding on reasonably sized n and k. Devise an strategy to evaluate your tool and report plots to demonstrate the performance.

Encode Mondrian art

Exercise 6.18

Mondiran art problem: "Divide an integer square into non-congruent rectangles. If all the sides are integers, what is the smallest possible difference in area between the largest and smallest rectangles?"

Consider a square of size $n \times n$, find a Mondrian solution above k using a SAT solver.

End of Lecture 6