

Tutorial 3: SAT and SMT

September 25, 2026

Encoding Propositional Formulae

A proposition is defined by:

$$f ::= x \mid \neg f \mid f_1 \wedge f_2$$

Encoding Propositional Formulae

A proposition is defined by:

$$f ::= x \mid \neg f \mid f_1 \wedge f_2$$

You can define other operations in terms of these:

Encoding Propositional Formulae

A proposition is defined by:

$$f ::= x \mid \neg f \mid f_1 \wedge f_2$$

You can define other operations in terms of these:

$$f_1 \vee f_2 := \neg(\neg f_1 \wedge \neg f_2)$$

Encoding Propositional Formulae

A proposition is defined by:

$$f ::= x \mid \neg f \mid f_1 \wedge f_2$$

You can define other operations in terms of these:

$$f_1 \vee f_2 := \neg(\neg f_1 \wedge \neg f_2)$$

$$f_1 \rightarrow f_2 := \neg f_1 \vee f_2$$

Encoding Propositional Formulae

A proposition is defined by:

$$f ::= x \mid \neg f \mid f_1 \wedge f_2$$

You can define other operations in terms of these:

$$\begin{aligned} f_1 \vee f_2 &:= \neg(\neg f_1 \wedge \neg f_2) \\ f_1 \rightarrow f_2 &:= \neg f_1 \vee f_2 \\ f_1 \iff f_2 &:= (f_1 \rightarrow f_2) \wedge (f_2 \rightarrow f_1) \end{aligned}$$

Encoding Propositional Formulae

A proposition is defined by:

$$f ::= x \mid \neg f \mid f_1 \wedge f_2$$

You can define other operations in terms of these:

$$\begin{aligned} f_1 \vee f_2 &:= \neg(\neg f_1 \wedge \neg f_2) \\ f_1 \rightarrow f_2 &:= \neg f_1 \vee f_2 \\ f_1 \iff f_2 &:= (f_1 \rightarrow f_2) \wedge (f_2 \rightarrow f_1) \\ f_1 \oplus f_2 &:= \neg(f_1 \iff f_2) \end{aligned}$$

Conjunctive Normal Form (CNF)

A formula in CNF is a conjunction of clauses:

$$F := C_1 \wedge C_2 \wedge \dots \wedge C_n$$

Conjunctive Normal Form (CNF)

A formula in CNF is a conjunction of clauses:

$$F := C_1 \wedge C_2 \wedge \dots \wedge C_n$$

And a clause is a disjunction of literals:

$$C := l_1 \vee l_2 \vee \dots \vee l_n$$

Conjunctive Normal Form (CNF)

A formula in CNF is a conjunction of clauses:

$$F := C_1 \wedge C_2 \wedge \dots \wedge C_n$$

And a clause is a disjunction of literals:

$$C := l_1 \vee l_2 \vee \dots \vee l_n$$

A literal is merely a variable x or its complement $\bar{x}$.

From Formula to CNF

Remember we just need negation and conjunction.

From Formula to CNF

Remember we just need negation and conjunction.

$$\neg x = \bar{x}$$

From Formula to CNF

Remember we just need negation and conjunction.

$$\begin{aligned}\neg x &= \bar{x} \\ x \wedge y &= (\bar{x} \vee \bar{y} \vee c) \wedge (\bar{c} \vee x) \wedge (\bar{c} \vee y) \\ &\text{where } c \text{ is a fresh literal}\end{aligned}$$

From Formula to CNF

Remember we just need negation and conjunction.

$$\begin{aligned}\neg x &= \bar{x} \\ x \wedge y &= (\bar{x} \vee \bar{y} \vee c) \wedge (\bar{c} \vee x) \wedge (\bar{c} \vee y) \\ &\quad \text{where } c \text{ is a fresh literal}\end{aligned}$$

Everything else can be derived from these mappings!

An Example

Take the following non-CNF formula:

$$(z \vee w) \rightarrow x$$

An Example

Take the following non-CNF formula:

$$(z \vee w) \rightarrow x$$

Get rid of the implication:

$$\neg(z \vee w) \vee x$$

An Example

Take the following non-CNF formula:

$$(z \vee w) \rightarrow x$$

Get rid of the implication:

$$\neg(z \vee w) \vee x$$

Distribute negation:

$$(\neg z \wedge \neg w) \vee x$$

An Example

Take the following non-CNF formula:

$$(z \vee w) \rightarrow x$$

Get rid of the implication:

$$\neg(z \vee w) \vee x$$

Distribute negation:

$$(\neg z \wedge \neg w) \vee x$$

Replace conjunction and add constraints:

$$(c \vee x) \wedge (\neg\neg z \vee \neg\neg w \vee c) \wedge (\neg c \vee \neg z) \wedge (\neg c \vee \neg w)$$

An Example

Take the following non-CNF formula:

$$(z \vee w) \rightarrow x$$

Get rid of the implication:

$$\neg(z \vee w) \vee x$$

Distribute negation:

$$(\neg z \wedge \neg w) \vee x$$

Replace conjunction and add constraints:

$$(c \vee x) \wedge (\neg\neg z \vee \neg\neg w \vee c) \wedge (\neg c \vee \neg z) \wedge (\neg c \vee \neg w)$$

Simplify:

$$(c \vee x) \wedge (z \vee w \vee c) \wedge (\bar{c} \vee \bar{z}) \wedge (\bar{c} \vee \bar{w})$$

Bitvectors in CNF

Bitvectors are arrays of 0s and 1s:

$$(1110)_2 = 14$$

We can use these to model numbers in CNF!

Bitvectors in CNF

Bitvectors are arrays of 0s and 1s:

$$(1110)_2 = 14$$

We can use these to model numbers in CNF!

For example, take equality on $x_1x_2\dots x_n$ and $y_1y_2\dots y_n$:

$$(x_1 \iff y_1) \wedge (x_2 \iff y_2) \wedge \dots \wedge (x_n \iff y_n)$$

Bitvectors in CNF

Bitvectors are arrays of 0s and 1s:

$$(1110)_2 = 14$$

We can use these to model numbers in CNF!

For example, take equality on $x_1x_2\dots x_n$ and $y_1y_2\dots y_n$:

$$(x_1 \iff y_1) \wedge (x_2 \iff y_2) \wedge \dots \wedge (x_n \iff y_n)$$

What about other operations?

Adding Bitvectors

We can encode $x + y = z$ as:

$$\begin{aligned}z_1 &= x_1 \oplus y_1 \\z_2 &= x_2 \oplus y_2 \oplus \text{carry}_1 \\z_3 &= x_3 \oplus y_3 \oplus \text{carry}_2 \\&\dots \\z_n &= x_n \oplus y_n \oplus \text{carry}_{n-1} \\z_{n+1} &= \text{carry}_n\end{aligned}$$

Where carrying is defined by:

$$\begin{aligned}\text{carry}_1 &= x_1 \wedge y_1 \\ \text{carry}_{i+1} &= \text{AtLeastTwo}(\text{carry}_i, x_i, y_i)\end{aligned}$$

Proving Something!

How might we prove something useful?

$$(w > y \wedge x \geq z) \rightarrow (w + x > y + z)$$

SMT and the Garden Tree Problem

The problem specification:

1. A garden is a 5×5 space.
2. You may place as many trees as you like, each with height from 1 to 4.
3. Two trees of height n cannot be placed within n spaces of each other.
4. You want to maximize height.

Example Garden

One possible garden of height 34:

3	—	1	4	3
—	1	—	2	1
1	2	—	1	—
4	1	—	—	—
1	3	—	2	4

Example Garden

One possible garden of height 34:

3	—	1	4	3
—	1	—	2	1
1	2	—	1	—
4	1	—	—	—
1	3	—	2	4

Can we do any better?

Example Garden

One possible garden of height 34:

3	—	1	4	3
—	1	—	2	1
1	2	—	1	—
4	1	—	—	—
1	3	—	2	4

Can we do any better? Let's simplify.

Simplified Garden Tree Problem

The problem specification:

1. A garden is an $N \times N$ space.
2. You may place from a multiset of trees, each with height from 1 to 4 (e.g. $\{h_1 : 8, h_2 : 4, h_3 : 3, h_4 : 3\}$).
3. Two trees of height n cannot be placed within n spaces of each other.
4. You want to maximize height.

Constraints for Trees

Each tree has the following attributes:

- ▶ $x : \text{Nat}$ (the x coordinate of the tree)
- ▶ $y : \text{Nat}$ (the y coordinate of the tree)

And we have the following constraints:

- ▶ Trees must be in a valid coordinate.
- ▶ Every slot can only contain one tree.
- ▶ Trees with the same height cannot be too close to each other.

How can we do better?

Right now, we're providing the multiset.

How can we do better?

Right now, we're providing the multiset. What if we look for it?

How can we do better?

Right now, we're providing the multiset. What if we look for it?

Each tree has the following attributes:

- ▶ $x : \text{Nat}$ (the x coordinate of the tree)
- ▶ $y : \text{Nat}$ (the y coordinate of the tree)
- ▶ $\text{planted} : \text{Bool}$ (is the tree planted?)

And we have the following constraints:

- ▶ Trees must be in a valid coordinate.
- ▶ Every slot can only plant one tree.
- ▶ Planted trees with the same height cannot be too close.
- ▶ Total height of all planted trees must be $\geq \text{target}$.