Computer-Aided Reasoning for Software

CSE507

SAT Solving Basics

Topics

Last lecture

- Going pro with solver-aided programming

Today

- Review of propositional logic
- Normal forms
- A basic SAT solver

topics

Review of propositional logic

- Syntax
- Semantics
- Satisfiability and validity
- Proof methods
- Semantic judgments

Syntax of propositional logic

$$(\neg p \wedge \top) \vee (q \rightarrow \bot)$$

Syntax of propositional logic

$$(\neg \boldsymbol{p} \wedge \top) \vee (\boldsymbol{q} \rightarrow \bot)$$

Atom — **truth symbols**: $\top$ ("true"), $\bot$ ("false")
propositional variables: $p, q, r, \ldots$

Literal — an atom α or its negation $\neg\alpha$

Formula — an atom or the application of a **logical connective** to formulas F_1, F_2:

$\neg F_1$	"not"	(negation)
$F_1 \wedge F_2$	"and"	(conjunction)
$F_1 \vee F_2$	"or"	(disjunction)
$F_1 \rightarrow F_2$	"implies"	(implication)
$F_1 \leftrightarrow F_2$	"if and only if"	(iff)

Semantics of propositional logic: interpretations

An **interpretation** I for a propositional formula F maps every variable in F to a truth value:

$$I : \{ p \mapsto \text{true}, q \mapsto \text{false}, \ldots\}$$

I is a **satisfying interpretation** of F, written as $I \vDash F$, if F evaluates to true under I.

I is a **falsifying interpretation** of F, written as $I \nvDash F$, if F evaluates to false under I.

A satisfying interpretation is also called a **model**.

Semantics of propositional logic: definition

Base cases:

- $I \vDash \top$
- $I \nvDash \bot$
- $I \vDash p$ iff $I[p] = \text{true}$
- $I \nvDash p$ iff $I[p] = \text{false}$

Inductive cases:

- $I \vDash \neg F$ iff $I \nvDash F$
- $I \vDash F_1 \wedge F_2$ iff $I \vDash F_1$ and $I \vDash F_2$
- $I \vDash F_1 \vee F_2$ iff $I \vDash F_1$ or $I \vDash F_2$
- $I \vDash F_1 \rightarrow F_2$ iff $I \nvDash F_1$ or $I \vDash F_2$
- $I \vDash F_1 \leftrightarrow F_2$ iff $I \vDash F_1$ and $I \vDash F_2$, or $I \nvDash F_1$ and $I \nvDash F_2$

Semantics of propositional logic: example

F: $(p \wedge q) \rightarrow (p \vee \neg q)$

I: $\{p \mapsto \text{true}, q \mapsto \text{false}\}$

?

$I \vDash F$

Satisfiability & validity of propositional formulas

F is **satisfiable** iff $I \models F$ for some I.

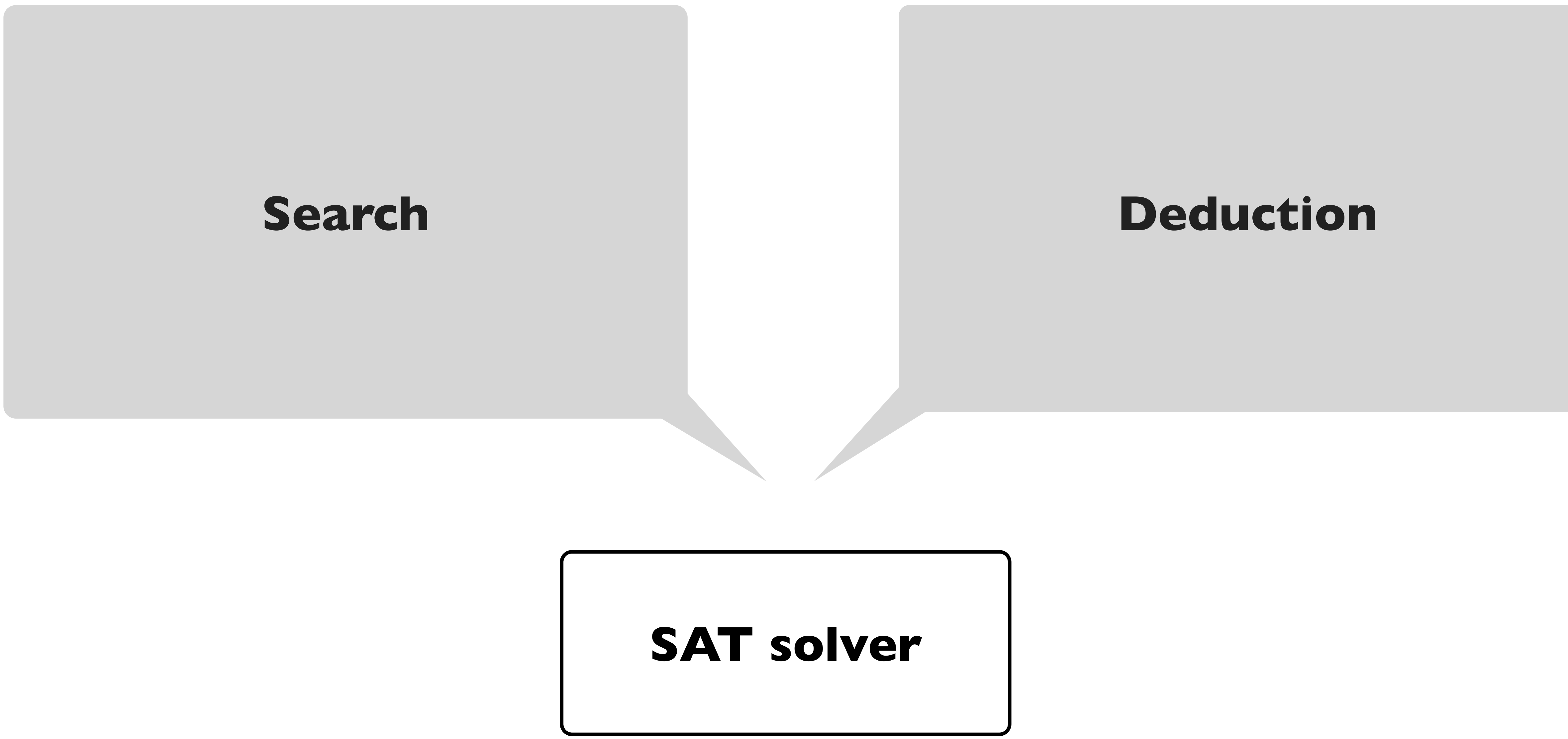
F is **valid** iff $I \models F$ for all I.

Duality of satisfiability and validity:

F is valid iff $\neg F$ is unsatisfiable.

If we have a procedure for checking satisfiability, we can also check validity of propositional formulas, and vice versa.

Techniques for deciding satisfiability & validity



Techniques for deciding satisfiability & validity

Search

Enumerate all interpretations (i.e., build a truth table), and check that they satisfy the formula.

Deduction

Assume the formula is invalid, apply proof rules, and check for contradiction in every branch of the proof tree.

SAT solver

Proof by search: enumerating interpretations

$F: \quad (p \wedge q) \rightarrow (p \vee \neg q)$

p	q	$p \wedge q$	$\neg q$	$p \vee \neg q$	F
0	0	0	1	1	1
0	1	0	0	0	1
1	0	0	1	1	1
1	1	1	0	1	1

Valid.

Proof by deduction: semantic arguments

$$\frac{I \models \neg F}{I \not\models F} \qquad \frac{I \not\models \neg F}{I \models F}$$

$$\frac{I \models F_1 \wedge F_2}{I \models F_1, I \models F_2} \qquad \frac{I \not\models F_1 \wedge F_2}{I \not\models F_1 \mid I \not\models F_2}$$

$$\frac{I \models F_1 \vee F_2}{I \models F_1 \mid I \models F_2} \qquad \frac{I \not\models F_1 \vee F_2}{I \not\models F_1, I \not\models F_2}$$

$$\frac{I \models F_1 \rightarrow F_2}{I \not\models F_1 \mid I \models F_2} \qquad \frac{I \not\models F_1 \rightarrow F_2}{I \models F_1, I \not\models F_2}$$

$$\frac{I \models F_1 \leftrightarrow F_2}{I \models F_1 \wedge F_2 \mid I \not\models F_1 \vee F_2} \qquad \frac{I \not\models F_1 \leftrightarrow F_2}{I \models F_1 \wedge \neg F_2 \mid I \models \neg F_1 \wedge F_2}$$

A **proof rule** consists of

- *premise*: facts that have to hold to apply the rule.
- *conclusion*: facts derived from applying the rule.

Commas indicate derivation of multiple facts; pipes indicate alternative facts (branches in the proof).

Proof by deduction: another example

$$\frac{I \models \neg F}{I \not\models F}$$

$$\frac{I \not\models \neg F}{I \models F}$$

$$\frac{I \models F_1 \wedge F_2}{I \models F_1, I \models F_2}$$

$$\frac{I \not\models F_1 \wedge F_2}{I \not\models F_1 \mid I \not\models F_2}$$

$$\frac{I \models F_1 \vee F_2}{I \models F_1 \mid I \models F_2}$$

$$\frac{I \not\models F_1 \vee F_2}{I \not\models F_1, I \not\models F_2}$$

$$\frac{I \models F_1 \rightarrow F_2}{I \not\models F_1 \mid I \models F_2}$$

$$\frac{I \not\models F_1 \rightarrow F_2}{I \models F_1, I \not\models F_2}$$

$$\frac{I \models F_1 \leftrightarrow F_2}{I \models F_1 \wedge F_2 \mid I \not\models F_1 \vee F_2}$$

$$\frac{I \not\models F_1 \leftrightarrow F_2}{I \models F_1 \wedge \neg F_2 \mid I \models \neg F_1 \wedge F_2}$$

Prove $p \wedge \neg q$ or find a falsifying interpretation.

1. $I \not\models p \wedge \neg q$ (assumed)
   a. $I \not\models p$ (1, $\wedge$)
   b. $I \not\models \neg q$ (1, $\wedge$)
      i. $I \models q$ (1b, $\neg$)

The formula is invalid, and $I = \{p \mapsto \text{false}, q \mapsto \text{true}\}$ is a falsifying interpretation.

Proof by deduction: another example

$$\frac{I \models \neg F}{I \not\models F} \qquad \frac{I \not\models \neg F}{I \models F}$$

$$\frac{I \models F_1 \wedge F_2}{I \models F_1, I \models F_2} \qquad \frac{I \not\models F_1 \wedge F_2}{I \not\models F_1 \mid I \not\models F_2}$$

$$\frac{I \models F_1 \vee F_2}{I \models F_1 \mid I \models F_2} \qquad \frac{I \not\models F_1 \vee F_2}{I \not\models F_1, I \not\models F_2}$$

$$\frac{I \models F_1 \rightarrow F_2}{I \not\models F_1 \mid I \models F_2} \qquad \frac{I \not\models F_1 \rightarrow F_2}{I \models F_1, I \not\models F_2}$$

$$\frac{I \models F_1 \leftrightarrow F_2}{I \models F_1 \wedge F_2 \mid I \not\models F_1 \vee F_2} \qquad \frac{I \not\models F_1 \leftrightarrow F_2}{I \models F_1 \wedge \neg F_2 \mid I \models \neg F_1 \wedge F_2}$$

1. $I \not\models (p \wedge (p \rightarrow q)) \rightarrow q$
2. $I \not\models q$ $(1, \rightarrow)$
3. $I \models (p \wedge (p \rightarrow q))$ $(1, \rightarrow)$
4. $I \models p$ $(3, \wedge)$
5. $I \models p \rightarrow q$ $(3, \wedge)$
   a. $I \not\models p$ $(5, \rightarrow)$
   b. $I \models q$ $(5, \rightarrow)$

We have reached a contradiction in every branch of the proof, so the formula is valid.

Semantic judgements

Formulas F_1 and F_2 are **equivalent**, written $F_1 \iff F_2$, iff $F_1 \leftrightarrow F_2$ is valid.

Formula F_1 **implies** F_2, written $F_1 \implies F_2$, iff $F_1 \rightarrow F_2$ is valid.

What do these definitions tell us in the context of this course?

$F_1 \iff F_2$ and $F_1 \implies F_2$ are **not** propositional formulas (not part of syntax). They are properties of formulas, just like validity or satisfiability.

review

Normal Forms (NNF, DNF, CNF)

Getting ready for SAT solving with normal forms

A **normal form** for a logic is a syntactic restriction such that every formula in the logic has an equivalent formula in the normal form.

Assembly language for a logic.

Three important normal forms for propositional logic:

- Negation Normal Form (NNF)
- Disjunctive Normal Form (DNF)
- Conjunctive Normal Form (CNF)

Negation Normal Form (NNF)

Atom := Variable | $\top$ | $\bot$

Literal := Atom | $\neg$Atom

Formula := Literal | Formula op Formula

op := $\wedge$ | $\vee$

The only allowed connectives are $\wedge$, $\vee$, and $\neg$.

$\neg$ can appear only in literals.

Conversion to NNF performed using **DeMorgan's Laws**:

$\neg(F \wedge G) \iff \neg F \vee \neg G$ $\neg(F \vee G) \iff \neg F \wedge \neg G$

Disjunctive Normal Form (DNF)

Atom := Variable | $\top$ | $\bot$

Literal := Atom | $\neg$Atom

Formula := Clause $\vee$ Formula

Clause := Literal | Literal $\wedge$ Clause

- Disjunction of conjunction of literals.
- Deciding satisfiability of a DNF formula is trivial.
- Why not SAT solve by conversion to DNF?

To convert to DNF, convert to NNF and distribute $\wedge$ over $\vee$:

$(F \wedge (G \vee H)) \iff (F \wedge G) \vee (F \wedge H)$

$((G \vee H) \wedge F) \iff (G \wedge F) \vee (H \wedge F)$

Conjunctive Normal Form (CNF)

Why CNF? Doesn't the conversion explode just as badly as DNF?

Atom := Variable | ⊤ | ⊥

Literal := Atom | ¬Atom

Formula := Clause ∧ Formula

Clause := Literal | Literal ∨ Clause

- Conjunction of disjunction of literals.
- Deciding the satisfiability of a CNF formula is hard.
- SAT solvers use CNF as their input language.

To convert to CNF, convert to NNF and distribute ∨ over ∧

$(F \vee (G \wedge H)) \iff (F \vee G) \wedge (F \vee H)$

$((G \wedge H) \vee F) \iff (G \vee F) \wedge (H \vee F)$

Equisatisfiability and Tseitin's transformation

Formulas F and G are **equisatisfiable** if they are both satisfiable or they are both unsatisfiable.

Tseitin's transformation converts a propositional formula F into an equisatisfiable CNF formula that is **linear** in the size of F.

Key idea: introduce **auxiliary variables** to represent the output of subformulas, and constrain those variables using CNF clauses.

Equisatisfiability and Tseitin's transformation

Formulas F and G are **equisatisfiable** if they are both satisfiable or they are both unsatisfiable.

Tseitin's transformation converts a propositional formula F into an equisatisfiable CNF formula that is **linear** in the size of F.

Key idea: introduce **auxiliary variables** to represent the output of subformulas, and constrain those variables using CNF clauses.

$x \rightarrow (y \wedge z)$

$a1$

$a1 \leftrightarrow (x \rightarrow a2)$

$a2 \leftrightarrow (y \wedge z)$

Equisatisfiability and Tseitin's transformation

Formulas F and G are **equisatisfiable** if they are both satisfiable or they are both unsatisfiable.

Tseitin's transformation converts a propositional formula F into an equisatisfiable CNF formula that is **linear** in the size of F.

$x \rightarrow (y \wedge z)$

$a1$
$a1 \rightarrow (x \rightarrow a2)$
$(x \rightarrow a2) \rightarrow a1$
$a2 \leftrightarrow (y \wedge z)$

Key idea: introduce **auxiliary variables** to represent the output of subformulas, and constrain those variables using CNF clauses.

Equisatisfiability and Tseitin's transformation

Formulas F and G are **equisatisfiable** if they are both satisfiable or they are both unsatisfiable.

Tseitin's transformation converts a propositional formula F into an equisatisfiable CNF formula that is **linear** in the size of F.

Key idea: introduce **auxiliary variables** to represent the output of subformulas, and constrain those variables using CNF clauses.

$x \rightarrow (y \wedge z)$

$a1$

$\neg a1 \vee (\neg x \vee a2)$

$(x \rightarrow a2) \rightarrow a1$

$a2 \leftrightarrow (y \wedge z)$

Equisatisfiability and Tseitin's transformation

Formulas F and G are **equisatisfiable** if they are both satisfiable or they are both unsatisfiable.

Tseitin's transformation converts a propositional formula F into an equisatisfiable CNF formula that is **linear** in the size of F.

$x \rightarrow (y \wedge z)$

$a1$
$\neg a1 \vee \neg x \vee a2$
$(x \wedge \neg a2) \vee a1$
$a2 \leftrightarrow (y \wedge z)$

Key idea: introduce **auxiliary variables** to represent the output of subformulas, and constrain those variables using CNF clauses.

Equisatisfiability and Tseitin's transformation

Formulas F and G are **equisatisfiable** if they are both satisfiable or they are both unsatisfiable.

Tseitin's transformation converts a propositional formula F into an equisatisfiable CNF formula that is **linear** in the size of F.

Key idea: introduce **auxiliary variables** to represent the output of subformulas, and constrain those variables using CNF clauses.

$x \rightarrow (y \wedge z)$

$a1$
$\neg a1 \vee \neg x \vee a2$
$x \vee a1$
$\neg a2 \vee a1$
$a2 \leftrightarrow (y \wedge z)$

Equisatisfiability and Tseitin's transformation

Formulas F and G are **equisatisfiable** if they are both satisfiable or they are both unsatisfiable.

Tseitin's transformation converts a propositional formula F into an equisatisfiable CNF formula that is **linear** in the size of F.

Key idea: introduce **auxiliary variables** to represent the output of subformulas, and constrain those variables using CNF clauses.

$x \rightarrow (y \wedge z)$

$a1$
$\neg a1 \vee \neg x \vee a2$
$x \vee a1$
$\neg a2 \vee a1$
$\neg a2 \vee y$
$\neg a2 \vee z$
$\neg y \vee \neg z \vee a2$

Another key feature of CNF: proof by resolution

Resolution rule

$$\frac{a_1 \lor \ldots \lor a_n \lor \beta \qquad b_1 \lor \ldots \lor b_m \lor \neg\beta}{a_1 \lor \ldots \lor a_n \lor b_1 \lor \ldots \lor b_m}$$

Proving that a CNF formula is valid can be done using just this one proof rule!

Apply the rule until a contradiction (empty clause) is derived, or no more applications are possible.

This procedure is sound and complete: it always produces a correct answer.

Another key feature of CNF: unit resolution

Resolution rule

$$\frac{a_1 \vee \ldots \vee a_n \vee \beta \qquad b_1 \vee \ldots \vee b_m \vee \neg\beta}{a_1 \vee \ldots \vee a_n \vee b_1 \vee \ldots \vee b_m}$$

Unit resolution rule

$$\frac{\beta \qquad b_1 \vee \ldots \vee b_m \vee \neg\beta}{b_1 \vee \ldots \vee b_m}$$

Unit resolution specializes the resolution rule to the case where one of the clauses is **unit** (a single literal).

SAT solvers use unit resolution in combination with backtracking search to implement a sound and complete procedure for deciding CNF formulas.

Unit resolution is a sound but incomplete rule of deduction, which is why we need search!

dpll

A basic SAT solver

Davis-Putnam-Logemann-Loveland (1962)

```
// Returns true if the CNF formula F is
// satisfiable; otherwise returns false.

DPLL(F)
  G ← BCP(F)
  if G = ⊤ then return true
  if G = ⊥ then return false
  p ← choose(vars(G))
  return DPLL(G{p ↦ ⊤}) ||
         DPLL(G{p ↦ ⊥})
```

Boolean constraint propagation applies unit resolution until fixed point.

If BCP cannot reduce *F* to a constant, we choose an unassigned variable and recurse assuming that the variable is either true or false.

If the formula is satisfiable under either assumption, then we know that it has a satisfying assignment (expressed in the assumptions). Otherwise, the formula is unsatisfiable.

Summary

Today

- Review of propositional logic
- Normal forms
- A basic SAT solver

Next Lecture

- A modern SAT solver