ETH zürich

Alexander J. Summers

Program Verification

Exercise Solutions 1: SAT Solving Algorithms

Assignment 1 (Tseitin CNF Conversion)

1. The conversion of $A \Leftrightarrow B$ to $(\neg A \vee B) \wedge (A \vee \neg B)$ increases the size, as does distributing disjunctions over conjunctions.

2. A formula $(p_{11} \wedge p_{12}) \vee (p_{21} \wedge p_{22}) \vee \ldots \vee (p_{n1} \wedge p_{n2})$ results in the formula (including 2^n clauses): $\bigwedge_{i_1,\ldots,i_n \in \{1,2\}} (p_{1i_1} \vee p_{2i_2} \vee \ldots \vee p_{ni_n})$

3. The clauses (which are themselves the translation of $(y \Rightarrow p \vee q) \wedge (p \vee q \Rightarrow y)$ are: $(\neg y \vee p \vee q) \wedge (\neg p \vee y) \wedge (\neg q \vee y)$

4. Processing the original formula should be performed until the rewritten version is in CNF (it could be continued further, but there is no need). In particular, if the original formula is already in CNF, we don't need to introduce any variables or make any changes.

5. We obtain $(q_1 \vee q_2 \vee \ldots \vee q_n) \wedge \bigwedge_{i \in \{1,\ldots,n\}} (\neg q_i \vee p_{i1} \vee p_{i2}) \wedge (\neg p_{i1} \vee q_i) \wedge (\neg p_{i2} \vee q_i)$ (in which $q_1, \ldots, q_n$ are fresh propositional variables).

6. The newly-introduced variables are tightly correlated with the original ones in the resulting formula; via unit propagation, choosing the truth values of one (or in the worst case, two) of p_{i1}, p_{i2}, q_i in a candidate model will force us to choose values for the others.

Assignment 2 (Applying SAT algorithms)

1. You can probably find a less verbose way of representing your solution; here we really show the individual steps explicitly. Note that after the first time (shown as separate steps for illustration), whenever we add a decision literal (the last DPLL rule) we will immediately apply unit propagation using the added unit clause.

   Here is one possible "run" of the algorithm (to save space, we just write a comma-separated set of clauses to represent the overall conjunction of clauses). We use overlining to indicate

clauses/literals which were added in the previous step, and underlining to indicate those removed in the *next* step.

$\{(n \vee p), (\neg n \vee p \vee q), (\neg n \vee p \vee \neg q), (\neg p \vee r), (\neg u \vee t), (\neg r \vee \neg s \vee t), (q \vee s),$
$(\neg p \vee t \vee u), (\neg p \vee \neg t \vee \neg u), (\neg r \vee \neg t \vee u)\}$

Add n *as a decision literal; current model is:* $\{n\}$

$\{\underline{(n \vee p)}, (\underline{\neg n} \vee p \vee q), (\underline{\neg n} \vee p \vee \neg q), (\neg p \vee r), (\neg u \vee t), (\neg r \vee \neg s \vee t), (q \vee s),$
$(\neg p \vee t \vee u), (\neg p \vee \neg t \vee \neg u), (\neg r \vee \neg t \vee u), \overline{(n)}\}$

(n) *is a unit clause: apply unit propagation; current model is:* $\{n\}$

$\{\underline{(p \vee q)}, \underline{(p \vee \neg q)}, (\underline{\neg p} \vee r), (\neg u \vee t), (\neg r \vee \neg s \vee t), (q \vee s),$
$(\underline{\neg p} \vee t \vee u), (\underline{\neg p} \vee \neg t \vee \neg u), (\neg r \vee \neg t \vee u)\}$

Add p *as a decision literal and apply unit propagation; current model is:* $\{n, p\}$

$\{\underline{(r)}, (\neg u \vee t), (\underline{\neg r} \vee \neg s \vee t), (q \vee s), (t \vee u), (\neg t \vee \neg u), (\underline{\neg r} \vee \neg t \vee u)\}$

(r) *is a unit clause: apply unit propagation; current model is:* $\{n, p, r\}$

$\{(\neg u \vee t), \underline{(\neg s \vee t)}, (q \vee \underline{s}), (t \vee u), (\neg t \vee \neg u), (\neg t \vee u)\}$

Add $\neg s$ *as a decision literal and apply unit propagation; current model is:* $\{n, p, r, \neg s\}$

$\{(\neg u \vee t), \underline{(q)}, (t \vee u), (\neg t \vee \neg u), (\neg t \vee u)\}$

(q) *is a pure literal: apply pure literal rule; current model is:* $\{n, p, r, \neg s, q\}$

$\{\underline{(\neg u \vee t)}, \underline{(t \vee u)}, (\underline{\neg t} \vee \neg u), (\underline{\neg t} \vee u)\}$

Add t *as a decision literal and apply unit propagation; current model is:* $\{n, p, r, \neg s, q, t\}$

$\{\underline{(\neg u)}, (\underline{u})\}$

$(\neg u)$ *is a unit clause: apply unit propagation; current model is:* $\{n, p, r, \neg s, q, t, \neg u\}$

$\{(\bot)\}$ *(*)*

We have an empty clause: backtrack decision literal t*; model is:* $\{n, p, r, \neg s, q\}$
...

...
$\{(\neg u \vee \underline{t}), (\underline{t} \vee u), \underline{(\neg t \vee \neg u)}, \underline{(\neg t \vee u)}\}$

Add $\neg t$ *as a decision literal and apply unit propagation; current model is:* $\{n, p, r, \neg s, q, \neg t\}$

$\{\underline{(\neg u)}, \underline{(u)}\}$

$(\neg u)$ *is a unit clause: apply unit propagation; current model is:* $\{n, p, r, \neg s, q, \neg t, \neg u\}$

$\{(\bot)\}$

We have an empty clause: backtrack decision literal $\neg s$*; model is:* $\{n, p, r\}$

$\{(\neg u \vee t), \underline{(\neg s \vee t)}, \underline{(q \vee s)}, (t \vee u), (\neg t \vee \neg u), (\neg t \vee u)\}$

Add s *as a decision literal and apply unit propagation; current model is:* $\{n, p, r, s\}$

$\{\underline{(\neg u \vee t)}, \underline{(t)}, \underline{(t \vee u)}, (\underline{\neg t} \vee \neg u), (\underline{\neg t} \vee u)\}$

(t) *is a unit clause: apply unit propagation; current model is:* $\{n, p, r, s, t\}$

$\{\underline{(\neg u)}, \underline{(u)}\}$

$(\neg u)$ *is a unit clause: apply unit propagation; current model is:* $\{n, p, r, s, t, \neg u\}$

$\{(\bot)\}$

We have an empty clause: backtrack decision literal p*; model is:* $\{n\}$

$\{(\underline{p} \vee q), (\underline{p} \vee \neg q), \underline{(\neg p \vee r)}, (\neg u \vee t), (\neg r \vee \neg s \vee t), (q \vee s),$
$\underline{(\neg p \vee t \vee u)}, \underline{(\neg p \vee \neg t \vee \neg u)}, (\neg r \vee \neg t \vee u)\}$

Add $\neg p$ *as a decision literal and apply unit propagation; current model is:* $\{n, \neg p\}$

$\{\underline{(q)}, (\underline{\neg q}), (\neg u \vee t), (\neg r \vee \neg s \vee t), \underline{(q \vee s)}, (\neg r \vee \neg t \vee u)\}$

(q) *is a unit clause: apply unit propagation; current model is:* $\{n, \neg p, q\}$

$\{(\bot), (\neg u \vee t), (\neg r \vee \neg s \vee t), (\neg r \vee \neg t \vee u)\}$

We have an empty clause: backtrack decision literal n*; model is:* $\{\}$
...

...

$\{(\underline{n} \vee p), \underline{(\neg n \vee p \vee q)}, \underline{(\neg n \vee p \vee \neg q)}, (\neg p \vee r), (\neg u \vee t), (\neg r \vee \neg s \vee t), (q \vee s),$
$(\neg p \vee t \vee u), (\neg p \vee \neg t \vee \neg u), (\neg r \vee \neg t \vee u)\}$

Add $\neg n$ as a decision literal and apply unit propagation; current model is: $\{\neg n\}$

$\{\underline{(p)}, (\underline{\neg p} \vee r), (\neg u \vee t), (\neg r \vee \neg s \vee t), (q \vee s),$
$(\underline{\neg p} \vee t \vee u), (\underline{\neg p} \vee \neg t \vee \neg u), (\neg r \vee \neg t \vee u)\}$

(p) is a unit clause: apply unit propagation; current model is: $\{\neg n, p\}$

$\{\underline{(r)}, (\neg u \vee t), (\underline{\neg r} \vee \neg s \vee t), (q \vee s), (t \vee u), (\neg t \vee \neg u), (\underline{\neg r} \vee \neg t \vee u)\}$

(r) is a unit clause: apply unit propagation; current model is: $\{\neg n, p, r\}$

$\{\underline{(\neg u \vee t)}, \underline{(\neg s \vee t)}, (q \vee s), \underline{(t \vee u)}, (\underline{\neg t} \vee \neg u), (\underline{\neg t} \vee u)\}$

Add t as a decision literal and apply unit propagation; current model is: $\{\neg n, p, r, t\}$

$\{(q \vee s), \underline{(\neg u)}, (\underline{u})\}$

$(\neg u)$ is a unit clause: apply unit propagation; current model is: $\{\neg n, p, r, t, \neg u\}$

$\{(q \vee s), (\bot)\}$

We have an empty clause: backtrack decision literal t; model is: $\{\neg n, p, r\}$

$\{(\neg u \vee \underline{t}), (\neg s \vee \underline{t}), (q \vee s), (\underline{t} \vee u), \underline{(\neg t \vee \neg u)}, \underline{(\neg t \vee u)}\}$

Add $\neg t$ as a decision literal and apply unit propagation; current model is: $\{\neg n, p, r, \neg t\}$

$\{\underline{(\neg u)}, (\neg s), (q \vee s), (\underline{u})\}$

$(\neg u)$ is a unit clause: apply unit propagation; current model is: $\{\neg n, p, r, \neg t, \neg u\}$

$\{(\neg s), (q \vee s), (\bot)\}$

We have an empty clause and have exhausted all backtracking options: return `unsat`

2. Initially, the CDCL algorithm will perform identical steps to the DPLL algorithm (essentially duplicating the first page of solution to the previous part, up to the first conflict, marked *(*)* above). We don't show these steps again. The implication graph built by these steps is shown in Figure 1.

   From this point *(*)* onwards, the behaviour of the algorithm is different. Based on analysing the conflict, we backtrack *both* decision literals t (relevant) and the prior decision $\neg s$

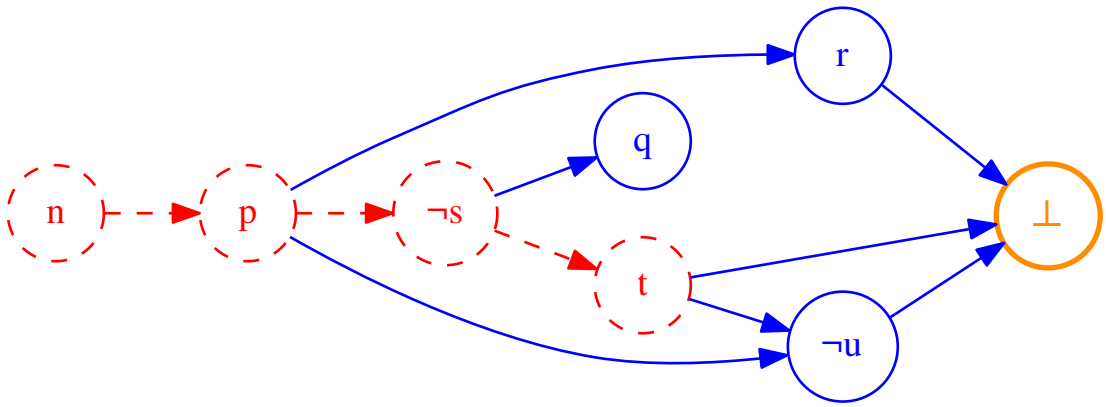

Figure 1: The implication graph at the point *(*)*.

(irrelevant for the conflict). We can also *learn* the clause $(\neg p \vee \neg t)$ as a result of the conflict (which depended on the decision literals p and t). Note that this learned clause should be kept also when subsequent back-jumps are made, as if it had been in the input problem originally. We will then proceed by first flipping the value of the relevant backtracked literal t:

... current model is: $\{n, p, r\}$

$\{(\neg u \vee \underline{t}), (\neg s \vee \underline{t}), (q \vee s), (\underline{t} \vee u), \underline{(\neg t \vee \neg u)}, \underline{(\neg t \vee u)}, \underline{\overline{(\neg p \vee \neg t)}}\}$

Add $\neg t$ *as a decision literal and apply unit propagation; current model is:* $\{n, p, r, \neg t\}$

$\{\underline{(\neg u)}, (\neg s), (q \vee s), \underline{(u)}\}$

$(\neg u)$ *is a unit clause: apply unit propagation; current model is:* $\{n, p, r, \neg t, \neg u\}$

$\{(\neg s), (q \vee s), (\bot)\}$

We have an empty clause: the relevant decision literals are $p, \neg t$
In this case, the learned clause is: $\neg p \vee t$
We backtrack decision literal p*; model is:* $\{n\}$

$\{(\underline{p} \vee q), (\underline{p} \vee \neg q), \underline{(\neg p \vee r)}, (\neg u \vee t), (\neg r \vee \neg s \vee t), (q \vee s),$
$\underline{(\neg p \vee t \vee u)}, \underline{(\neg p \vee \neg t \vee \neg u)}, (\neg r \vee \neg t \vee u), \underline{(\neg p \vee \neg t)}, \underline{(\neg p \vee t)}\}$

Add $\neg p$ as a decision literal and apply unit propagation; current model is: $\{n, \neg p\}$

$\{\underline{(q)}, \underline{(\neg q)}, (\neg u \vee t), (\neg r \vee \neg s \vee t), \underline{(q \vee s)}, (\neg r \vee \neg t \vee u)\}$

(q) is a unit clause: apply unit propagation; current model is: $\{n, \neg p, q\}$

$\{(\bot), (\neg u \vee t), (\neg r \vee \neg s \vee t), (\neg r \vee \neg t \vee u)\}$

We have an empty clause: the relevant decision literals are $n, \neg p$
In this case, the learned clause is: $\neg n \vee p$
We backtrack decision literal n*; model is:* $\{\}$
$\{\underline{(n \vee p)}, \underline{(\neg n \vee p \vee q)}, \underline{(\neg n \vee p \vee \neg q)}, (\neg p \vee r), (\neg u \vee t), (\neg r \vee \neg s \vee t), (q \vee s),$
$(\neg p \vee t \vee u), (\neg p \vee \neg t \vee \neg u), (\neg r \vee \neg t \vee u), (\neg p \vee \neg t), (\neg p \vee t), \underline{(\neg n \vee p)}\}$

Add $\neg n$ as a decision literal and apply unit propagation; current model is: $\{\neg n\}$

$\{\underline{(p)}, \underline{(\neg p} \vee r), (\neg u \vee t), (\neg r \vee \neg s \vee t), (q \vee s),$
$\underline{(\neg p} \vee t \vee u), \underline{(\neg p} \vee \neg t \vee \neg u), (\neg r \vee \neg t \vee u), \underline{(\neg p} \vee \neg t), \underline{(\neg p} \vee t)\}$

(p) is a unit clause: apply unit propagation; current model is: $\{\neg n, p\}$
...

...
$\{(r), \underline{(\neg u \vee t)}, \underline{(\neg r \vee \neg s \vee t)}, (q \vee s), \underline{(t \vee u)}, \underline{(\neg t} \vee \neg u), (\neg r \vee \underline{\neg t} \vee u), \underline{(\neg t)}, \underline{(t)}\}$

(t) is a unit clause: apply unit propagation; current model is: $\{\neg n, p, \neg t\}$

$\{(r), (q \vee s), (\neg u), (\neg r \vee u), (\bot)\}$

We have an empty clause and have exhausted all backtracking options: return `unsat`

Note that the CDCL search is shorter: once we begin to learn clauses, the depth of path that needs to be explored during search attempts tends to get shorter. If useful clauses are learned, this effect can become more significant for larger examples.

Assignment 3 (Correctness of DPLL)

1. Every recursive step of the algorithm either terminates the algorithm or reduces the number of propositional variables in the formula to be processed; no steps of the algorithm add new such variables. Since each step also recurses at most twice (when backtracking), the algorithm terminates: the number of variables in the formula is a termination measure.

2. We prove the following, stronger property: for a given input formula A and model M, if the algorithm returns `sat` and a model M', we have $M \subseteq M'$ and $M' \models A$.

We prove this by induction on the definition of the algorithm; since it terminates, this proof method is valid (we could also use induction on the number of variables in the current formula, given the answer to the previous part).

That is, we show that the desired property holds for any input A,M, assuming that it holds for all recursive calls made by the algorithm (our induction hypothesis).

We consider all five cases of the algorithm:

- The model we return is guaranteed to satisfy $\top$.
- Since we do not return `sat`, this case is trivial.
- (Pure Literal Rule) Assume we have a variable p which only occurs positively (the negative case is analogous). Let A' be the formula A after deleting all clauses containing p. Suppose that running the algorithm on A' and $M \cup \{p\}$ results in `sat` and a model M' (otherwise, we will not return `sat` anyway). Then, by our induction hypothesis, we have $M \cup \{p\} \subseteq M'$, and $M' \models A'$. Then we must have $p \in M'$, and so $M' \models A$ since it models any clause containing p.
- (Unit Propagation Rule) Assume that l is the (only) literal in the unit clause in A used in the rule. Let A' be the formula A processed according to the rule (all clauses with l as a literal are removed, and all clauses of the form $\tilde{l} \vee C$ are updated to just C). Suppose further that the recursive call using A' and $M \cup \{l\}$ returns $(\texttt{sat}, M')$. By our induction hypothesis, $M \cup \{l\} \subseteq M'$ and $M' \models A'$. Therefore $l \in M'$, and so M' satisfies all clauses which were removed from A to obtain A'. Furthermore, since M' must make all clauses in A' true, for any such clause B it will also make $l \vee B$ true. Therefore, $M' \models A$.
- (Decision Literal Rule) If the first recursive call returns $(\texttt{sat}, M')$, then by our induction hypothesis, $M' \models A \wedge p$, and so $M' \models A$ as required. Similarly, if the second recursive call returns `sat`. Otherwise, we will not return `sat` overall anyway, and have nothing to prove.