Assignment Project Exam Help

CS262 Logic and Verification

Lecture 5: Normal form algorithms

https://tutorcs.com

WeChat: cstutorcs

Generalized disjunctions/conjunctions

Let $X_1, X_2, \ldots, X_n$ be a sequence of propositional formulas.

We write a **generalized disjunction** as:

$[X_1, X_2, \ldots, X_n] := X_1 \vee X_2 \vee \cdots \vee X_n$ (recall associativity!)

We write a **generalized conjunction** as:

$\langle X_1, X_2, \ldots, X_n \rangle := X_1 \wedge X_2 \wedge \cdots \wedge X_n$

If $X_1, \ldots, X_n$ are literals, then $[X_1, X_2, \ldots, X_n]$ is called a **clause**, and $\langle X_1, X_2, \ldots, X_n \rangle$ is called a **dual clause**.

Valuations on generalized disjunctions/disjunctions

This is how valuations act on generalized disjunctions/disjunctions:

$v([X_1, \ldots, X_n]) = T$ if and only if $v(X_i) = T$ **for at least one** member of the list $X_1, \ldots, X_n$.

$v(\langle X_1, \ldots, X_n \rangle) = T$ if and only if $v(X_i) = T$ **for every** member of the list $X_1, \ldots, X_n$.

$v([\,]) = v(\bot) = F$ (neutral element of disjunction)

$v(\langle\,\rangle) = v(\top) = T$ (neutral element of conjunction)

We group all propositional formulas of the forms $(X \circ Y)$ and $\neg(X \circ Y)$ (where $\circ$ is one of $\{\wedge, \vee, \rightarrow, \leftarrow, \uparrow, \downarrow, \not\rightarrow, \not\leftarrow\}$, see the exercises) into two categories, those that act conjunctively, and those that act disjunctively:

Conjunctive			*Disjunctive*		
α	α_1	α_2	β	β_1	β_2
$X \wedge Y$	X	Y	$\neg(X \wedge Y)$	$\neg X$	$\neg Y$
$\neg(X \vee Y)$	$\neg X$	$\neg Y$	$X \vee Y$	X	Y
$\neg(X \rightarrow Y)$	X	$\neg Y$	$X \rightarrow Y$	$\neg X$	Y
$\neg(X \leftarrow Y)$	$\neg X$	Y	$X \leftarrow Y$	X	$\neg Y$
$\neg(X \uparrow Y)$	X	Y	$X \uparrow Y$	$\neg X$	$\neg Y$
$X \downarrow Y$	$\neg X$	$\neg Y$	$\neg(X \downarrow Y)$	X	Y
$X \not\rightarrow Y$	X	$\neg Y$	$\neg(X \not\rightarrow Y)$	$\neg X$	Y
$X \not\leftarrow Y$	$\neg X$	Y	$\neg(X \not\leftarrow Y)$	X	$\neg Y$

Valuations on α- and β-formulas

For every α- and β-formula and every valuation v, we have

$v(\alpha) = v(\alpha_1) \wedge v(\alpha_2)$

$v(\beta) = v(\beta_1) \vee v(\beta_2)$

I.e., we have $\alpha = \alpha_1 \wedge \alpha_2$ and $\beta = \beta_1 \vee \beta_2$ (logical equivalence).

Conjunctive normal form algorithm

Given any any propositional formula X, start with $\langle [X] \rangle$.

Given the current expansion $\langle D_1, \ldots, D_n \rangle$, select one of the disjunctions D_i that is not a disjunction of literals.

Select a non-literal N from D_i.

If $N = \neg\top$, then replace N by $\bot$.
If $N = \neg\bot$, then replace N by $\top$.
If $N = \neg\neg Z$, then replace N by Z.
If N is a β-formula, then replace N by the two formula sequence β_1, β_2 (**β-expansion**).
If N is an α-formula, then replace the disjunction D_i with two disjunctions, one in which N is replaced by α_1, and one in which N is replaced by α_2 (**α-expansion**).

Example 1 (CNF expansion)

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

$\langle[\neg(p \wedge \neg\bot) \vee \neg(\top \uparrow q)]\rangle = \ldots$

Example 2 (CNF expansion)

$(\neg p \to (q \not\leftarrow r)) \to ((p \downarrow q) \uparrow (p \to \neg r))$

$\langle [(\neg p \to (q \not\leftarrow r)) \to ((p \downarrow q) \uparrow (p \to \neg r))] \rangle = \ldots$

Here is a compact representation of these rules:

$$\frac{\neg\top}{\bot} \qquad \frac{\neg\bot}{\top} \qquad \frac{\neg\neg Z}{Z} \qquad \frac{\beta}{\begin{array}{c}\beta_1\\ \beta_2\end{array}} \qquad \frac{\alpha}{\alpha_1 \mid \alpha_2}$$

Correctness of this algorithm

Proposition: Throughout the algorithm, we produce a sequence of logically equivalent formulas.

Proof: We only verify the last two steps.

β-expansion:

$$\begin{aligned} D_i &= [\beta, X_2, \ldots, X_k] \\ &= [\beta_1 \vee \beta_2, X_2, \ldots, X_k] \\ &= [\beta_1, \beta_2, X_2, \ldots, X_k] \end{aligned}$$

α-expansion:

$$\begin{aligned} D_i &= [\alpha, X_2, \ldots, X_k] \\ &= (\alpha_1 \wedge \alpha_2) \vee (X_2 \vee \cdots \vee X_k) \\ &= (\alpha_1 \vee (X_2 \vee \cdots \vee X_k)) \wedge (\alpha_2 \vee (X_2 \vee \cdots \vee X_k)) \\ &= \langle [\alpha_1, X_2, \ldots, X_k], [\alpha_2, X_2, \ldots, X_k] \rangle \end{aligned}$$

□

Termination of the algorithm

Proposition: The algorithm terminates, regardless of which choices are made during the algorithm.

To prove this, we need to make a little detour.

Consider a rooted tree. A **branch** is a sequence of nodes starting at the root, descending towards one of the childen in each step (unless no children are present). We say that the tree is **finitely branching**, if every node has only finitely many children (possibly 0). A tree/branch is **finite** if it has a finite number of nodes; otherwise it is **infinite**.

Theorem (König's lemma): A tree that is finitely branching but infinite must have an infinite branch.

A game with balls

Consider the following game played with balls that have non-negative integer labels:

At the beginning, we have a box containing just one ball.

In each step, we may remove one ball from the box, and replace it by any (finite) number of balls having lower numbers.

Theorem: This game must end, regardless of which choices are made during the game.

Proof: Use König's theorem. □

Proposition: The algorithm terminates, regardless of which choices are made during the algorithm.

Proof: Define the **rank** of a propositional formula as follows:

Recursion anchor: $r(p) = r(\neg p) = 0$ for variables p; $r(\top) = r(\bot) = 0$; $r(\neg\top) = r(\neg\bot) = 1$.

Recursive step: $r(\neg\neg Z) = r(Z) + 1$
$r(\alpha) = r(\alpha_1) + r(\alpha_2) + 1$; $r(\beta) = r(\beta_1) + r(\beta_2) + 1$

For a generalized disjunction: $r([X_1, \ldots, X_n]) = \sum_{i=1}^{n} r(X_i)$

Examples:
$r((p \to \neg q) \wedge (\neg\neg r \wedge \neg\bot)) = \ldots$
$r([p \wedge q, \neg\top, \neg\neg r]) = \ldots$

Answers should be 5 and 3, respectively.

Termination of the algorithm

Proposition: The algorithm terminates, regardless of which choices are made during the algorithm.

Proof: ...continued...

Assign the current conjunction of disjunctions $\langle D_1, \ldots, D_n \rangle$ a sequence of n balls, by placing one ball labelled $r(D_i)$ for each disjunction D_i into a box.

Argue that each step of the algorithm corresponds to removing one ball from the box, and replacing it either by two balls with a lower number (α-expansion) or by one ball with a lower number (in all other cases).

Example: $\langle [p \wedge q, \neg\top, \neg\neg r], [p \to q] \rangle$

We have argued before that this game must end, so our algorithm must terminate. $\square$

Normal form algorithms

We have seen an algorithm to transform any formula into conjunctive normal form (CNF). How to do the same for disjunctive normal form (DNF)?

Recall the CNF algorithm: Start with $\langle [X] \rangle$, and repeatedly apply one of the following expansion rules:

$$\frac{\neg\top}{\bot} \qquad \frac{\neg\bot}{\top} \qquad \frac{\neg\neg Z}{Z} \qquad \frac{\beta}{\begin{array}{c}\beta_1\\ \beta_2\end{array}} \qquad \frac{\alpha}{\alpha_1 \mid \alpha_2}$$

Here is the DNF algorithm: Start with $[\langle X \rangle]$, and repeatedly apply one of the following expansion rules:

$$\frac{\neg\top}{\bot} \qquad \frac{\neg\bot}{\top} \qquad \frac{\neg\neg Z}{Z} \qquad \frac{\beta}{\beta_1 \mid \beta_2} \qquad \frac{\alpha}{\begin{array}{c}\alpha_1\\ \alpha_2\end{array}}$$

Example 1 (DNF expansion)

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

$[\langle\neg(p \wedge \neg\bot) \vee \neg(\top \uparrow q)\rangle] = \ldots$

Example 2 (DNF expansion)

$(\neg p \to (q \not\leftarrow r)) \to ((p \downarrow q) \uparrow (p \to \neg r))$

$[\langle(\neg p \to (q \not\leftarrow r)) \to ((p \downarrow q) \uparrow (p \to \neg r))\rangle] = \ldots$