

CSC258 Lecture Notes

Unified Algebra

An amusing paper on unified algebra can be found at www.cs.utoronto.ca/~hehner/BAUA.pdf. A much fuller description of unified algebra is at www.cs.utoronto.ca/~hehner/UA.pdf and that's where these notes are from.

Order of Evaluation

An algebra consists of expressions, and the expressions consist of operators and operands. Placing operators between operands makes some expressions ambiguous. For example, $2+3\times 4$ might mean that 2 and 3 are added, and then the result is multiplied by 4, or that 2 is added to the result of multiplying 3 by 4. To say which is meant, we can use parentheses: either $(2+3)\times 4$ or $2+(3\times 4)$. To prevent a clutter of parentheses, we decide on an order of evaluation. Here is the order of evaluation of all operators in these lecture notes.

0	constants $\top \perp 0 1 3.14$ and so on variables $x y$ and so on bracketed expressions $()$ within which the order of evaluation again applies	
1	prefix $+ -$	right to left
2	infix $\times / \wedge \vee$	left to right
3	infix $+ - \triangle \nabla$	left to right
4	infix $= \neq < > \leq \geq$	
5	outfix and prefix if then else	right to left

In the order of evaluation, infix $+$ can be found on level 3, and infix \times on level 2; that means, in the absence of parentheses, evaluate infix \times before infix $+$. The example $2+3\times 4$ therefore means $2+(3\times 4)$. Within levels 2 and 3 evaluation is from left to right. Within levels 1 and 5 evaluation is from right to left. On level 4, $x=y=z$ means $(x=y)\wedge(y=z)$ and similarly for the other operators and mixtures of operators on that level.

Format

To help the eye group the symbols properly, it is a good idea to leave space for absent parentheses. The spacing in expression $2 + 3\times 4$ is helpful; the spacing in $2+3 \times 4$ is misleading.

An expression that is too long to fit on one line must be broken into parts. There are several reasonable ways to do it; here is one suggestion. A long expression in parentheses can be broken at its main operator, which is placed under the opening parenthesis. For example,

$$\left(\begin{array}{l} \textit{first part} \\ + \textit{second part} \end{array} \right)$$

A long expression without parentheses can be broken at its main operator, which is placed under where the opening parenthesis belongs. For example,

$$\begin{array}{l} \textit{first part} \\ = \textit{second part} \end{array}$$

Attention to format makes a big difference in our ability to understand a complex expression.

Expressions and Values

An algebra consists of expressions, which are used to express values in the application domain. For example, the values may be amounts of water, or voltage, or frequency of vibration, or guilt and innocence. We must never use one expression to express more than one value; to do so would be a serious error called inconsistency. Sometimes we may not say what value an expression expresses; that is called incompleteness. For example, we will not be able to determine the value of $0/0$. (I prefer to avoid the question of whether $0/0$ has no value, or has a value but we cannot say what it is. It is of no interest whether an expression expresses a value if we cannot determine the value.) Here are four definitions.

Consistency:	at most one value can be determined for each expression
Completeness:	at least one value can be determined for each expression
Expressiveness:	at least one expression can be determined for each value
Uniqueness:	at most one expression can be determined for each value

Consistency is essential; completeness is not. Expressiveness is desirable; uniqueness is not. In general, several expressions may represent the same value. When we say that $2+3$ is 5 , we do not mean that $2+3$ and 5 are the same expression; clearly they are not. We mean that the value represented by $2+3$ is the value represented by 5 . When we say that $2+3$ has value 5 , we mean that expression $2+3$ represents the same value that 5 represents. We might just as well say that 5 has value $2+3$.

Variables and Instantiation

A variable is a kind of expression. Expressions represent values, and a variable represents an arbitrary value. A variable can be replaced by another expression. Replacing a variable by another expression is called instantiation. Expression -2 is called an instance of expression $-x$. Here is how instantiation works.

- We sometimes have to insert parentheses around expressions that are replacing variables in order to maintain the order of evaluation. Expression $-(2+3)$ is an instance of $-x$, but $-2+3$ is not.
- When the same variable occurs more than once in an expression, it must be replaced by the same expression at each occurrence. Expression $2+2$ is an instance of $x+x$, but $2+3$ is not. However, different variables may be replaced by the same or different expressions. Both $2+2$ and $2+3$ are instances of $x+y$.

Evaluation Rules

Here are the rules to determine the value of expressions.

Direct Rule An expression may be given a value by physical means, or by other means outside the algebra. This is the way an algebra is applied.

Example: By marking numbers along a stick we give them length values.

Example: We might decide to use \top to express truth and \perp to express falsity.

Indirect Rule An expression may be given a value by saying that it has the same value as another expression whose value is already known. This rule is used in two forms: value tables, and laws.

Example: In binary algebra, on one of the value tables, from the row labelled $x \wedge y$ and the column labelled $\top \top$, we will see that $\top \wedge \top$ is \top .

Example: From the first of the common laws, we will see that $x = x$ is \top .

Transparency Rule An expression does not change value when a subexpression is replaced by another expression with the same value.

Example: If x and y have the same value, then $x+z$ and $y+z$ have the same value.

Consistency Rule If it would be inconsistent for an expression to have a particular value, then it has another value. More generally, if it would be inconsistent for several expressions to have a particular assignment of values, then they have another assignment of values.

Example: In binary algebra, we will see from the value tables that if both x and $x \leq y$ are \top , then so is y .

Example: In binary algebra, we will see from the value tables that if $\neg x$ is \top , then x is \perp .

Example: In binary algebra, we will see from the value tables that if $x=y$ is \top , then x and y have the same value, and if $x=y$ is \perp , then x and y have different values.

Instance Rule If the value of an expression can be determined, then all its instances have that same value.

Example: Since $x=x$ is \top , therefore $x + y \times z = x + y \times z$ is \top .

Completion Rule If all ways of assigning values to its subexpressions give an expression the same value, then it has that value.

Example: In binary algebra, we will see from the value tables that $x \vee \neg x$ is \top .

Example: In binary algebra, we will see from the value tables that $x \wedge \neg x$ is \perp .

Binary Algebra

The expressions of binary algebra are called binary expressions. Binary expressions can be used to represent anything that comes in two kinds, such as true and false statements, high and low voltage, satisfactory and unsatisfactory computations, innocent and guilty behavior, north and south poles of magnets. In any application of binary algebra, the two things being represented are called the “binary values”. For example, in one application the binary values are truth and falsity; in another they are innocence and guilt. Binary expressions include:

\top	“top”
\perp	“bottom”
$\neg x$	“negate x ”
$x=y$	“ x equal y ”
$x \neq y$	“ x differ y ”
$x < y$	“ x below y ”
$x > y$	“ x above y ”
$x \leq y$	“ x below equal y ”
$x \geq y$	“ x above equal y ”

- $x \wedge y$ “ x min y ”; the top of the symbol is narrow; the symbol doesn't hold water
- $x \vee y$ “ x max y ”; the top of the symbol is wide; the symbol holds water
- $x \triangle y$ “ x neg min y ”
- $x \nabla y$ “ x neg max y ”

if x then y else z

The two simplest binary expressions are \top and \perp . Expression \top represents one binary value, and expression \perp represents the other. In the other binary expressions, the variables x , y , and z may be replaced by any binary expressions. Whichever value is represented by expression x , expression $\neg x$ represents the other value. This rule can be shown with the aid of a value table.

x	\top	\perp
$\neg x$	\perp	\top

This table says that $\neg \top$ represents the same value that \perp represents, and that $\neg \perp$ represents the same value that \top represents. We can similarly show how to evaluate other binary expressions.

xy	$\top\top$	$\top\perp$	$\perp\top$	$\perp\perp$
$x=y$	\top	\perp	\perp	\top
$x \neq y$	\perp	\top	\top	\perp
$x < y$	\perp	\perp	\top	\perp
$x > y$	\perp	\top	\perp	\perp
$x \leq y$	\top	\perp	\top	\top
$x \geq y$	\top	\top	\perp	\top
$x \wedge y$	\top	\perp	\perp	\perp
$x \vee y$	\top	\top	\top	\perp
$x \triangle y$	\perp	\top	\top	\top
$x \nabla y$	\perp	\perp	\perp	\top

xyz	$\top\top\top$	$\top\top\perp$	$\top\perp\top$	$\top\perp\perp$	$\perp\top\top$	$\perp\top\perp$	$\perp\perp\top$	$\perp\perp\perp$
if x then y else z	\top	\top	\perp	\perp	\top	\perp	\top	\perp

Preference

We have two binary values, and so far we have not shown any preference for one over the other. Now we shall show a preference for expressions with the value of \top in four ways. One way is to abbreviate the statement “Expression x has the same value as \top .” by just writing x , without saying anything about it. Whenever we just write a binary expression, we mean that it has the same value as \top (expresses the same value that \top expresses). For example, instead of saying “Expression $\top = \top$ has the same value as \top .” we just say “ $\top = \top$ ”.

Another way we show a preference is by the use of the words “solution” and “law”. A solution to a binary expression is an assignment of values to its variables that gives it the value of \top ; we have no name for an assignment that gives it the value of \perp . A law is a binary expression for which any assignment of values to its variables gives it the value \top , and so by the Completion Rule it too has the value \top ; we have no name for a binary expression that has the value \perp .

We often use the Indirect Rule by stating that an expression is a law, which means we are assigning it the same value as \top . If we want to assign \perp 's value to expression x , instead we state the law

state the law $\neg x$, and then rely on the Consistency Rule to say that x is \perp . In other algebras, if we want to say that x has the same value as y (not a binary value), instead we say that $x=y$ has the same value as \top by stating that $x=y$ is a law.

The final way we show a preference is in the applications of binary algebra. When we apply it to reasoning, we choose to use \top for true statements and \perp for false statements. When we use binary expressions as specifications, we choose to use \top for satisfactory objects, and \perp for unsatisfactory objects. When we use binary expressions to codify laws, we choose to use \top for innocent behavior and \perp for guilty behavior. In each case we could just as well have chosen to use \top and \perp the other way round, but the tradition is to use \top for the preferable alternative.

Ternary Algebra

Between the two values represented by \top and \perp , we now consider a third value, represented by 0 (pronounced “zero”). Ternary algebra can be applied to anything that comes in three kinds. In one application, the three expressions \top , 0 , and \perp represent the values “yes”, “maybe”, and “no”. In another, they represent the values “large”, “medium”, and “small”. An assignment of values to variables that gives an expression the value 0 is called a “root” of the expression.

The expressions of ternary algebra, called “ternary expressions”, include all those of binary algebra. To determine the value of these ternary expression, we extend the value tables.

x	\top	0	\perp							
$\neg x$	\perp	0	\top							
xy	$\top\top$	$\top 0$	$\top\perp$	$0\top$	00	$0\perp$	$\perp\top$	$\perp 0$	$\perp\perp$	
$x=y$	\top	\perp	\perp	\perp	\top	\perp	\perp	\perp	\top	
$x\neq y$	\perp	\top	\top	\top	\perp	\top	\top	\top	\perp	
$x<y$	\perp	\perp	\perp	\top	\perp	\perp	\top	\top	\perp	
$x>y$	\perp	\top	\top	\perp	\perp	\top	\perp	\perp	\perp	
$x\leq y$	\top	\perp	\perp	\top	\top	\perp	\top	\top	\top	
$x\geq y$	\top	\top	\top	\perp	\top	\top	\perp	\perp	\top	
$x\wedge y$	\top	0	\perp	0	0	\perp	\perp	\perp	\perp	
$x\vee y$	\top	\top	\top	\top	0	0	\top	0	\perp	
$x\Delta y$	\perp	0	\top	0	0	\top	\top	\top	\top	
$x\nabla y$	\perp	\perp	\perp	\perp	0	0	\perp	0	\top	

When the variables have binary values, each expression has the same value as it had in binary algebra; in that sense, we have extended binary algebra to ternary algebra in a consistent way. All our future extensions will likewise be consistent. The expression $x=\neg x$ has no solution in binary algebra because both assignments of binary values give it the value \perp ; in ternary algebra it has solution 0 . The expression $x\vee\neg x$ is a law of binary algebra because both assignments of binary values to variable x give it the value \top ; but it is not a law of ternary algebra because when x is 0 , $x\vee\neg x$ is 0 . By extending the algebra, we have gained some solutions and lost some laws.

We can add many new ternary expressions. For example, we can add approximate equality and addition modulo 3 with the value table:

xy	$\top\top$	$\top 0$	$\top\perp$	$0\top$	00	$0\perp$	$\perp\top$	$\perp 0$	$\perp\perp$
$x\approx y$	\top	0	\perp	0	0	0	\perp	0	\top
$x\oplus y$	\perp	\top	0	\top	0	\perp	0	\perp	\top

Four and More Values

To design a four-valued algebra, we have a choice. One way starts with ternary algebra and adds a new value \perp different from \top , 0 , and \perp , situated between \top and \perp , but unrelated to 0 in the ordering (this algebra is the same as a pair of binary algebras). Another way starts with binary algebra, and adds two new values represented by 1 (pronounced “one”) and -1 , with a total ordering. And there are many interesting algebras with more values. We now leap to an infinite-valued totally-ordered algebra. Value tables, which are already cumbersome for ternary algebra (**if x then y else z** takes 27 columns), become impossible with infinitely many values, so from now on we give values to new expressions by stating laws.

Common Laws

There are some laws of binary algebra that are not laws of any other algebra; for example

$x \vee \neg x$	excluded middle
$\neg(x \wedge \neg x)$	noncontradiction
$(x \leq y) = \neg x \vee y$	material
$((x=y)=z) = (x=(y=z))$	associative
$((x\neq y)\neq z) = (x\neq(y\neq z))$	associative
$(x=y) = (x \wedge y) \vee (\neg x \wedge \neg y)$	iff
$(x\neq y) = (x \wedge \neg y) \vee (\neg x \wedge y)$	exclusion

We have introduced binary and ternary expressions, and mentioned expressions of four or more values. We are about to introduce expressions of infinitely many values (numbers) by saying how to write them and giving their laws. There are many laws that are common to all of these algebras; for example

$\perp \leq x \leq \top$	extremes
$x \wedge \perp = \perp$	base
$x \vee \top = \top$	base
$x \triangle \perp = \top$	base
$x \nabla \top = \perp$	base
$x \wedge \top = x$	identity
$x \vee \perp = x$	identity
$(x = \top) = x$	identity
$(x \neq \perp) = x$	identity
$x = x$	reflexivity
$x \leq x$	reflexivity
$x \geq x$	reflexivity
$\neg(x < x)$	irreflexivity
$\neg(x > x)$	irreflexivity
$\neg\neg x = x$	double negation or self-inverse
$x \wedge x = x$	idempotence
$x \vee x = x$	idempotence
$(x=y) = (y=x)$	symmetry
$(x\neq y) = (y\neq x)$	symmetry

$x \wedge y = y \wedge x$	symmetry
$x \vee y = y \vee x$	symmetry
$x \Delta y = y \Delta x$	symmetry
$x \nabla y = y \nabla x$	symmetry
$\neg(x < y < x)$	antisymmetry
$\neg(x > y > x)$	antisymmetry
$\neg(x < y = x)$	exclusivity
$\neg(x > y = x)$	exclusivity
$(x < y) \vee (x = y) \vee (x > y)$	trichotomy
$(x \leq y) = (x < y) \vee (x = y)$	inclusivity
$(x \geq y) = (x > y) \vee (x = y)$	inclusivity
$(x > y) = (y < x)$	mirror
$(x \geq y) = (y \leq x)$	mirror
$(x < y) = \neg(x > y)$	reflection
$(x \wedge y = x) = (x \leq y) = (y = x \vee y)$	connection
$x \wedge (x \vee y) = x$	absorption
$x \vee (x \wedge y) = x$	absorption
$\neg(x = y) = (\neg x \neq \neg y)$	duality
$\neg(x \neq y) = (\neg x = \neg y)$	duality
$\neg(x < y) = (\neg x \leq \neg y)$	duality
$\neg(x \leq y) = (\neg x < \neg y)$	duality
$\neg(x > y) = (\neg x \geq \neg y)$	duality
$\neg(x \geq y) = (\neg x > \neg y)$	duality
$\neg(x \wedge y) = \neg x \vee \neg y$	duality, deMorgan
$\neg(x \vee y) = \neg x \wedge \neg y$	duality, deMorgan
$\neg(x \Delta y) = \neg x \nabla \neg y$	duality
$\neg(x \nabla y) = \neg x \Delta \neg y$	duality
$x \wedge (x \leq y) \leq y$	modus ponens
$(x \neq y) = \neg(x = y)$	negation
$x \Delta y = \neg(x \wedge y)$	negation
$x \nabla y = \neg(x \vee y)$	negation
$(x \wedge y) \wedge z = x \wedge (y \wedge z)$	associativity
$(x \vee y) \vee z = x \vee (y \vee z)$	associativity
$(x = y = z) \leq (x = z)$	transitivity
$(x < y < z) \leq (x < z)$	transitivity
$(x > y > z) \leq (x > z)$	transitivity
$(x \leq y \leq z) \leq (x \leq z)$	transitivity
$(x \geq y \geq z) \leq (x \geq z)$	transitivity
$x \wedge y \leq y \leq y \vee z$	specialization and generalization
$x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$	distribution or factoring
$x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z)$	distribution or factoring
$(x \leq y \wedge z) = (x \leq y) \wedge (x \leq z)$	distribution or factoring
$(x \leq y \vee z) = (x \leq y) \vee (x \leq z)$	distribution or factoring
$(x \wedge y \leq z) = (x \leq z) \vee (y \leq z)$	antidistribution
$(x \vee y \leq z) = (x \leq z) \wedge (y \leq z)$	antidistribution
$(w \wedge x) \vee (y \wedge z) \leq (w \vee y) \wedge (x \vee z)$	semi-commutative
(if \top then x else y) = x	base case
(if \perp then x else y) = y	base case
$\neg(\text{if } x \text{ then } y \text{ else } z) = \text{if } x \text{ then } \neg y \text{ else } \neg z$	base distributivity or factoring

There are laws of number algebra that are not laws of binary algebra, but only because they employ the symbols of number algebra (like $+$ and \times) that are not symbols of binary algebra. Any law of number algebra that employs only the symbols of binary algebra is also a law of binary algebra.

It is an interesting mathematical exercise to find a minimal set of laws for an algebra. But those who wish to use the algebra need to know many laws, and to them minimality is of no concern. In these lecture notes, no attention has been paid to minimality.

Number Algebra

We now introduce infinitely many values between \top and \perp . All operators apply to all values.

$\perp \dots -3 -2 -1 0 1 2 3 \dots \top$

(In these lecture notes, I consider only those numbers called “algebraic reals”.)

The expressions of number algebra are called number expressions. Number expressions can be used to represent anything that comes in various quantities, such as apples and water (\top represents an infinite quantity, and \perp represents an infinite deficit). Expressions are formed as follows.

any sequence of one or more decimal digits, such as 5296

any of the ways of forming an expression presented previously, such as

-5296 or $5296 \wedge 375$ or $5297 = 375$

$x+y$

“ x plus y ”

$x-y$

“ x minus y ”

$x \times y$

“ x times y ”

x/y

“ x divided by y ”, “ x over y ”

x^y

“ x to the power y ”

Anyone is welcome to invent new expressions and add them to the list.

Now that we have new expressions, we assign some of them the same value as \top . In these laws, d is a sequence of digits.

$$d0+1 = d1$$

counting

$$d1+1 = d2$$

counting

$$d2+1 = d3$$

counting

$$d3+1 = d4$$

counting

$$d4+1 = d5$$

counting

$$d5+1 = d6$$

counting

$$d6+1 = d7$$

counting

$$d7+1 = d8$$

counting

$$d8+1 = d9$$

counting

$$d9+1 = (d+1)0$$

counting

$$x+0 = x$$

identity

$$x+y = y+x$$

symmetry

$$x+(y+z) = (x+y)+z$$

associativity

$$(\perp < x < \top) \leq ((x+y = x+z) = (y=z))$$

cancellation

$$(\perp < x) \leq (\top + x = \top)$$

absorption

$$(x < \top) \leq (\perp + x = \perp)$$

absorption

$$x + y \wedge z = (x+y) \wedge (x+z)$$

distributivity or factoring

$$x + y \vee z = (x+y) \vee (x+z)$$

distributivity or factoring

$$x + (y \Delta z) = (x-y) \vee (x-z)$$

$$x + (y \nabla z) = (x-y) \wedge (x-z)$$

$x + (y \vee z) = (x-y) \wedge (x-z)$	
$w + (\text{if } x \text{ then } y \text{ else } z) = \text{if } x \text{ then } w+y \text{ else } w+z$	distributivity
$-x = 0 - x$	negation
$-(x+y) = -x + -y$	distributivity or factoring
$-(x-y) = -x - -y$	distributivity or factoring
$-(x \times y) = (-x) \times y$	associativity
$-(x/y) = (-x)/y$	associativity
$x-y = -(y-x)$	antisymmetry
$x-y = x + -y$	subtraction
$x + (y - z) = (x + y) - z$	associativity
$(\perp < x < \top) \leq ((x-y = x-z) = (y=z))$	cancellation
$(\perp < x < \top) \leq (x-x = 0)$	inverse
$(x < \top) \leq (\top - x = \top)$	absorption
$(\perp < x) \leq (\perp - x = \perp)$	absorption
$(\perp < x < \top) \leq (x \times 0 = 0)$	base
$x \times 1 = x$	identity
$x \times y = y \times x$	symmetry
$x \times (y+z) = x \times y + x \times z$	distributivity or factoring
$x \times (y \times z) = (x \times y) \times z$	associativity
$(\perp < x < \top) \wedge (x \neq 0) \leq ((x \times y = x \times z) = (y=z))$	cancellation
$(0 < x) \leq (x \times \top = \top)$	absorption
$(0 < x) \leq (x \times \perp = \perp)$	absorption
$x/1 = x$	identity
$(\perp < x < \top) \wedge (x \neq 0) \leq (x/x = 1)$	inverse
$x \times (y/z) = (x \times y)/z$	associativity
$(\perp < x < \top) \leq (x/\top = 0 = x/\perp)$	annihilation
$(\perp < x < \top) \leq (x^0 = 1)$	base
$x^1 = x$	identity
$x^{y+z} = x^y \times x^z$	exponents
$x^{y \times z} = (x^y)^z$	exponents
$\perp < 0 < 1 < \top$	direction
$(\perp < x < \top) \leq ((x+y < x+z) = (y < z))$	cancellation, translation
$(0 < x < \top) \leq ((x \times y < x \times z) = (y < z))$	cancellation, scale
$(x < y) \vee (x=y) \vee (x > y)$	trichotomy

Calculation

Given an expression, it is often useful to find a simpler expression with the same value. For example,

$x \times (z+1) - y \times (z-1) - z \times (x-y)$	
$= (x \times z + x \times 1) - (y \times z - y \times 1) - (z \times x - z \times y)$	distribute
$= x \times z + x - y \times z + y - z \times x + z \times y$	unity and double negation
$= x + y + (x \times z - x \times z) + (y \times z - y \times z)$	symmetry and associativity
$= x+y$	zero and identity

The entire five lines (without the hints that appear to the right) form one binary expression meaning

$(x \times (z+1) - y \times (z-1) - z \times (x-y) = (x \times z + x \times 1) - (y \times z - y \times 1) - (z \times x - z \times y))$
$\wedge ((x \times z + x \times 1) - (y \times z - y \times 1) - (z \times x - z \times y) = x \times z + x - y \times z + y - z \times x + z \times y)$
$\wedge (x \times z + x - y \times z + y - z \times x + z \times y = x + y + (x \times z - x \times z) + (y \times z - y \times z))$
$\wedge (x + y + (x \times z - x \times z) + (y \times z - y \times z) = x+y)$

By simply writing it, we are saying that it has the same value as \top . The hint “distribute” is intended to make it clear that

$$x \times (z+1) - y \times (z-1) - z \times (x-y) = (x \times z + x \times 1) - (y \times z - y \times 1) - (z \times x - z \times y)$$

is \top ; the hint “unity and double negation” is intended to make it clear that

$$(x \times z + x \times 1) - (y \times z - y \times 1) - (z \times x - z \times y) = x \times z + x - y \times z + y - z \times x + z \times y$$

is \top ; and so on. By the transitivity of $=$ and the Consistency Rule we see that

$$x \times (z+1) - y \times (z-1) - z \times (x-y) = x+y$$

is \top , and so $x \times (z+1) - y \times (z-1) - z \times (x-y)$ and $x+y$ have the same value.

We can use operators other than $=$ down the left side of a calculation, even a mixture, as long as there is transitivity. For example, if x is a real-valued variable,

$$\begin{aligned} & x \times (x + 2) && \text{distribute} \\ = & x^2 + 2 \times x && \text{identity and zero} \\ = & x^2 + 2 \times x + 1 - 1 && \text{factor} \\ = & (x+1)^2 - 1 && \text{a square is nonnegative} \\ \geq & -1 \end{aligned}$$

tells us that $x \times (x+2) \geq -1$ is \top .

The level of hint depends on the knowledge of the intended audience. A hint may refer to some laws, or to a calculation done elsewhere, or to some missing steps that a knowledgeable reader could reasonably be expected to supply.

Variation

One effective way of calculating is to increase or decrease an expression by increasing or decreasing a subexpression. We can increase $x \wedge y$ by increasing y . We can increase $-x$ by decreasing x . As an example, let a and b be binary.

$$\begin{aligned} & a \triangle (a \nabla b) \\ = & -(a \wedge -(a \nabla b)) && \text{decrease } a \nabla b \text{ to } a \text{ and so decrease the whole expression} \\ \geq & -(a \wedge -a) && \text{use a previous example} \\ = & -\perp \\ = & \top \end{aligned}$$

And so $a \triangle (a \nabla b)$ is above or equal to \top , and since there is nothing above \top , it is \top .

Here is a catalogue of variations for use in calculations.

- $-x$ varies inversely with x .
- $x+y$ varies directly with x and directly with y .
- $x-y$ varies directly with x and inversely with y .
- $x \wedge y$ varies directly with x and directly with y .
- $x \vee y$ varies directly with x and directly with y .
- $x \triangle y$ varies inversely with x and inversely with y .
- $x \nabla y$ varies inversely with x and inversely with y .
- $x < y$ varies inversely with x and directly with y .
- $x > y$ varies directly with x and inversely with y .
- $x \leq y$ varies inversely with x and directly with y .
- $x \geq y$ varies directly with x and inversely with y .
- if x then y else z** varies directly with y and directly with z .

Context

Consider an expression of the form $x \wedge y$ where x and y are binary. When we are simplifying x , we can suppose that y has value \top . If y really does have value \top , then we have done nothing wrong. If y has value \perp , then $x \wedge y$ has value \perp no matter which value x has; so no matter how we change x , we don't change the value of $x \wedge y$. For exactly the same reason, we can suppose that x has value \top when we are simplifying y . However, we cannot make both suppositions simultaneously and simplify both x and y at the same time. (If we could, then $x \wedge x$ could be simplified to \top .)

Here is an example.

$$\begin{aligned}
 & (x + x \times y + y = 5) \wedge (x - x \times y + y = 1) && \text{subtract and add } 2 \times x \times y \\
 = & (x - x \times y + y + 2 \times x \times y = 5) \wedge (x - x \times y + y = 1) && \text{use second part to simplify first} \\
 = & (1 + 2 \times x \times y = 5) \wedge (x - x \times y + y = 1) && \text{simplify} \\
 = & (2 \times x \times y = 4) \wedge (x - x \times y + y = 1) && \text{simplify} \\
 = & (x \times y = 2) \wedge (x - x \times y + y = 1) && \text{use first part to simplify second} \\
 = & (x \times y = 2) \wedge (x - 2 + y = 1) && \text{simplify} \\
 = & (x \times y = 2) \wedge (x + y = 3) \\
 \geq & (x=1) \wedge (y=2)
 \end{aligned}$$

We can generalize this sort of reasoning to apply to number expressions.

- In $x < y$,
 - when simplifying x , we can assume y is not \perp ;
 - when simplifying y , we can assume x is not \top .
- In $x > y$,
 - when simplifying x , we can assume y is not \top ;
 - when simplifying y , we can assume x is not \perp .
- In $x \leq y$,
 - when simplifying x , we can assume y is not \top ;
 - when simplifying y , we can assume x is not \perp .
- In $x \geq y$,
 - when simplifying x , we can assume y is not \perp ;
 - when simplifying y , we can assume x is not \top .
- In $x \wedge y$,
 - when simplifying x , we can assume y is not \perp ;
 - when simplifying y , we can assume x is not \perp .
- In $x \vee y$,
 - when simplifying x , we can assume y is not \top ;
 - when simplifying y , we can assume x is not \top .
- In $x \triangle y$,
 - when simplifying x , we can assume y is not \perp ;
 - when simplifying y , we can assume x is not \perp .
- In $x \nabla y$,
 - when simplifying x , we can assume y is not \top ;
 - when simplifying y , we can assume x is not \top .
- In **if x then y else z** ,
 - when simplifying y , we can assume x is not \perp ;
 - when simplifying z , we can assume x is not \top .

Textbook Terminology and Notations

unified algebra

binary
 expression
 law
 value table
 \top
 \perp
 $-$
 \times
 \wedge
 \vee
 \triangle
 ∇
 $=$
 \neq
 $<$
 $>$
 \leq
 \geq

textbook

boolean
 expression, proposition, sentence, term
 law, theorem, axiom, lemma
 truth table
 true, 1, tt, T
 false, 0, ff, F
 $-$, $!$, \sim , overscore, $'$, $\bar{}$
 \times , $*$, \cdot , no symbol
 \wedge , $\&$, AND, no symbol
 \vee , $|$, OR
 NAND
 NOR
 $=$, \equiv , \Leftrightarrow , XNOR
 \oplus , \neq , XOR
 not found in textbooks
 not found in textbooks
 \rightarrow , \Rightarrow , \therefore , \supset , IMPL
 not found in most textbooks

textbook

antecedent
 axiom
 boolean
 conjunct
 conjunction
 connective
 consequent
 disjunct
 disjunction
 entailment
 equivalence
 false
 formula
 gate
 implication
 implies
 inference
 lemma
 predicate
 proof
 proof rule
 proposition
 sentence
 term
 theorem
 true

unified algebra

binary left operand of \leq or right operand of \geq
 law
 binary
 binary operand of \wedge
 binary expression whose main operator is \wedge
 operator
 binary right operand of \leq or left operand \geq
 binary operand of \vee
 binary expression whose main operator is \vee
 binary expression whose main operator is \leq or \geq
 binary expression whose main operator is $=$
 \perp
 expression
 operator whose operands and result are binary
 binary expression whose main operator is \leq or \geq
 \leq
 binary expression whose main operator is \leq or \geq
 law used during calculation
 sometimes a function whose result is binary, sometimes a binary expression
 simplification to \top
 law
 binary expression
 binary expression
 number expression
 law
 \top

truth table	value table
valid	law
AND	\wedge
IMPL	\leq
NAND	\triangle
NOR	∇
OR	\vee
XOR	\neq
XNOR	$=$
0	sometimes \perp and sometimes 0
1	sometimes \top and sometimes 1
+	sometimes \vee and sometimes +
·	sometimes \wedge and sometimes \times
	sometimes no symbol at all is used to mean \wedge or \times
&	\wedge
	\vee
!	$-$
~	$-$
overscore	$-$
'	$-$
┘	$-$
*	\times
\equiv	$=$
\Leftrightarrow	$=$
\oplus	\neq
\neq	\neq
\rightarrow	\leq
\Rightarrow	\leq
\therefore	\leq
\cup	\leq