Maggie Johnson
CS103A

Handout #7

Logical Consequence

Key Topics

* Logical & Tautological Equivalence
* Logical & Tautological Consequence
* Negation Normal Form
* Conjunctive & Disjunctive Normal Forms

- Logical & Tautological Equivalence

S and S' are sentences of FOL built up from atomic sentences using truth-functional connectives. To test for tautological equivalence, we construct a joint truth table and check the final columns.

S and S' are *tautologically equivalent* if and only if every row of the joint truth table assigns the same values to S and S'.

If S and S' are tautologically equivalent, then they are *logically equivalent*.

Some logically equivalent sentences are not tautologically equivalent.

JJ = James ^ boy(JJ)
JJ = James ^ boy(James)

- Logical & Tautological Consequence

Logical Consequence: A sentence S is a logical consequence of a set of premises if it is impossible for the premises all to be true while the conclusion is false.

Logical Equivalence: Two sentences are logically equivalent if they have the same truth value in all possible circumstances.

Logical Truth or Necessity: A sentence that is a logical consequence of any set of premises. That is, no matter what the premises may be, it is impossible for the conclusion to be false (e.g., a = a).

Logical Possibility: A sentence or claim is logically possible if there is a possible circumstance in which it is true.

Tautological Equivalence: S and S' are tautologically equivalent if and only if every row of the joint truth table assigns the same values to S and S'.

Tautological Consequence: Q is a tautological consequence of a set of premises P1, ...Pn, if and only if every row in a truth table that assigns T to P1,...Pn also assigns T to Q.

Is Tet(a) v Large(c) a tautological consequence of the following two premises?

Tet(a) v ~Small(b)
Small(b) v Large(c)

How can we tell?

- Negation Normal Form

Logically equivalent sentences can be substituted for one another in the context of a larger sentence and the resulting sentences will also be logically equivalent.

Identity Laws	p ^ T <=> p p v F <=> p
Domination Laws	p v T <=> T p ^ F <=> F
Idempotent Laws	p v p <=> p p ^ p <=> p
Double Negation	~(~p) <=> p
Commutative Laws	p v q <=> q v p p ^ q <=> q ^ p
Associative Laws	(p v q) v r <=> p v (q v r) (p ^ q) ^ r <=> p ^ (q ^ r)
Distributive Laws	p v (q ^ r) <=> (p v q) ^ (p v r) p ^ (q v r) <=> (p ^ q) v (p ^ r)
DeMorgan's Laws	~(p ^ q) <=> ~p v ~q ~(p v q) <=> ~p ^ ~q
Other Useful Equivalences	p v ~p <=> T p ^ ~p <=> F

S(P) = FOL sentence that contains (possibly complex) sentence P as a component part. S(Q) is a new FOL sentence that is the result of substituting Q (which is logically equivalent to P) into S(P): S(P) <=> S(Q).

Negation Normal Form**:** Using only DeMorgan's Laws and the double negation equivalence, we can take any sentence built up from v, ^ and ~ and transform it such that each ~ applies only to atomic sentences. To do it, just drive each ~ inward switching v and ^ according to DeMorgan's laws, and canceling any double negations.

~(~(p ^ q) v r)

- Conjunctive and Disjunctive Normal Form

CNF: A sentence that is a conjunction of one or more disjunctions of one or more literals.

DNF: A sentence that is a disjunction of one or more conjunctions of one or more literals.

Convert the following to CNF:

p v (q ^ ~(r ^ (~s v t)))

1) Push the first NOT down to a literal using DeMorgan's Laws:

p v (q ^ ~(r ^ (~s v t)))
p v (q ^ (~r v ~(~s v t)))
p v (q ^ (~r v (s ^ ~t)))

2) Use the distributive law for OR over AND to push the first OR below the first AND:

p v (q ^ (~r v (s ^ ~t)))
(p v q) ^ (p v (~r v (s ^ ~t)))

3) We regroup using the associative law [(p v q) v r <=> p v (q v r)]

(p v q) ^ ((p v ~r) v (s ^ ~t))

4) Finally use the distributive law again:

(p v q) ^ ((p v ~r v s) ^ (p v ~r v ~t))

Can a sentence be in both CNF and DNF?