Systems of Equivalence, Soundness and Completeness

We have seen many logical equivalences such as Double Negation, Commutation, or DeMorgan. These logical equivalences are meta-logical theorems: statements that say something about the truth-conditions of a pair of sentences, and that can be proven using formal semantics (e.g. by using a truth-table). In particular, if two statements φ and ψ have the exact same truth-conditions, we say that the two statements are logically equivalent, and we write this as $\varphi \Leftrightarrow \psi$.

Another meta-logical theorem we have seen is the transitivity of $\Leftrightarrow$: if $\varphi \Leftrightarrow \psi$ and $\psi \Leftrightarrow \lambda$, then $\varphi \Leftrightarrow \lambda$. The transitivity of $\Leftrightarrow$ allow us to combine already established logical equivalences to prove further logical equivalences. For example, by combing Double Negation and Commutation, we can prove that $\varphi \wedge \psi \Leftrightarrow \neg\neg(\psi \wedge \varphi)$: By Commutation: $\varphi \wedge \psi \Leftrightarrow \psi \wedge \varphi$, and by Double Negation: $\psi \wedge \varphi \Leftrightarrow \neg\neg(\psi \wedge \varphi)$. Hence, by the transitivity of $\Leftrightarrow$, we obtain that $\varphi \wedge \psi \Leftrightarrow \neg\neg(\psi \wedge \varphi)$.

Yet another meta-logical theorem is the Substitution Principle of Logical Equivalents, which states that if $\varphi \Leftrightarrow \psi$, then $S(\varphi) \Leftrightarrow S(\psi)$, where $S(\varphi)$ is a statement that contains φ as a component statement, and where $S(\psi)$ is the statement that results from substituting ψ for φ in $S(\varphi)$. Using this principle, we can show something like $\varphi \wedge \psi \Leftrightarrow \psi \wedge \neg\neg\varphi$: By Commutation: $\varphi \wedge \psi \Leftrightarrow \psi \wedge \varphi$, and by Double Negation: $\varphi \Leftrightarrow \neg\neg\varphi$. Thus, by the Substitution Principle: $\psi \wedge \varphi \Leftrightarrow \psi \wedge \neg\neg\varphi$. So, by transitivity of $\Leftrightarrow$, we obtain that $\varphi \wedge \psi \Leftrightarrow \psi \wedge \neg\neg\varphi$.

The above two proofs are quite tedious, and in practice we will hardly ever go into that much detail. In particular, in practice we will hardly ever explicitly refer to the transitivity theorem and substitution principle, but simply go through a sequence of steps to transform one expression into another one by the successive ‘application’ of equivalences. Thus, for example, the last proof from above would go something more like this: $\varphi \wedge \psi \Leftrightarrow$ (Commutation) $\psi \wedge \varphi \Leftrightarrow$ (Double Negation) $\psi \wedge \neg\neg\varphi$.

However, in treating the logical equivalences as rules for rewriting expressions, we are blurring the line between syntax and semantics: whereas the equivalences are meta-logical theorems (i.e. formal semantical results), we ended up treating them as formal syntactical transformation rules, which is not the same thing.

To do things right, then, we should carefully distinguish between the two. So, let $\varphi \wedge \psi \Leftrightarrow \psi \wedge \varphi$ be, as before, the formal semantical statement (and provable theorem) that the logical expressions $\varphi \wedge \psi$ and $\psi \wedge \varphi$ are logically equivalent. On the other hand, let $\varphi \wedge \psi \dashv\vdash \psi \wedge \varphi$ be a formal syntactical rule of an algebraic logical expressions rewriting system that states that any expression S can be rewritten as S’ that is the same as S, but with $\psi \wedge \varphi$ substituted for any one instance of $\varphi \wedge \psi$ that is a component part of S, and vice versa. To simplify the language, we will refer to $\varphi \wedge \psi \Leftrightarrow \psi \wedge \varphi$ as a ‘logical equivalence’, or simply as an ‘equivalence’, while $\varphi \wedge \psi$

$\dashv\vdash \psi \wedge \varphi$ will be referred to as an 'equivalence rule'. An equivalence rule is thus intended to be the syntactical counterpart of an equivalence.

However, 'intended' is the key word here! For example, as part of a syntactical algebraic system of statement transformations, there is of course nothing that prevents us from defining $\varphi \wedge \psi \dashv\vdash \psi \vee \varphi$ as a rule that allows us to change any conjunctions into disjunctions, and vice versa. Of course, this rule does not reflect any equivalence at all, and it is therefore also not clear what utility such an algebra would have, but the point is that it can be done. In particular, the point is that there is again an important difference between formal syntax and formal semantics: we can syntactically define things in any way we want, but what this semantically means remains to be seen!

Fortunately, having a clear difference between formal syntax and formal semantics, we can try to answer the question as to whether some equivalence rule reflects an actual equivalence or not. That is, we can ask, and answer, the following question: Suppose we have some equivalence rule $\varphi \dashv\vdash \psi$. Is it true that $\varphi \Leftrightarrow \psi$? If so, then we can call the equivalence rule sound. If not, then it is unsound.

More interestingly, suppose we have a whole algebraic system S with multiple equivalence rules. Let us call such a system a 'system of equivalence'. Now, let us write $\varphi \dashv\vdash_S \psi$ to indicate that it is possible, using a finite succession of applications of the equivalence rules of S, to transform φ into ψ and vice versa. We can now ask two important questions:

1. **Soundness** of S: For any φ and ψ: If $\varphi \dashv\vdash_S \psi$, then $\varphi \Leftrightarrow \psi$?

2. **Completeness** of S: For any φ and ψ: If $\varphi \Leftrightarrow \psi$, then $\varphi \dashv\vdash_S \psi$?

As we will see later, soundness and completeness are important properties of systems of inference. Indeed, usually logicians talk about these notions in the context of systems of inference, rather than equivalence. However, we can bring up these important notions in the context of equivalence as well, and the good news is that the proofs for soundness and completeness of particular systems of equivalence are somewhat easier than for systems of inference, so going over these proofs will be good practice for when we get into systems of inference. Of course, in order to do so, we will first need to define a particular system of equivalence.

Equivalence System E.S.

Let Equivalence System E.S. be the system that consists of the following equivalence rules:

Double Negation	$(\varphi \dashv\vdash \neg\neg\varphi)$
Commutation	$(\varphi \wedge \psi \dashv\vdash \psi \wedge \varphi$ and $\varphi \vee \psi \dashv\vdash \psi \vee \varphi)$
Association	$(\varphi \wedge (\psi \wedge \lambda) \dashv\vdash (\varphi \wedge \psi) \wedge \lambda$ and $\varphi \vee (\psi \vee \lambda) \dashv\vdash (\varphi \vee \psi) \vee \lambda)$
DeMorgan	$(\neg(\varphi \wedge \psi) \dashv\vdash \neg\varphi \vee \neg\psi$ and $\neg(\varphi \vee \psi) \dashv\vdash \neg\varphi \wedge \neg\psi)$
Distribution	$(\varphi \wedge (\psi \vee \lambda) \dashv\vdash (\varphi \wedge \psi) \vee (\varphi \wedge \lambda)$ and $\varphi \vee (\psi \wedge \lambda) \dashv\vdash (\varphi \vee \psi) \wedge (\varphi \vee \lambda))$
Idempotence	$(\varphi \wedge \varphi \dashv\vdash \varphi$ and $\varphi \vee \varphi \dashv\vdash \varphi)$
Complement	$(\varphi \wedge \neg\varphi \dashv\vdash \bot$ and $\varphi \vee \neg\varphi \dashv\vdash \top)$
Identity	$(\varphi \wedge \top \dashv\vdash \varphi$ and $\varphi \vee \bot \dashv\vdash \varphi)$
Annihilation	$(\varphi \wedge \bot \dashv\vdash \bot$ and $\varphi \vee \top \dashv\vdash \top)$
Inverse	$(\neg\bot \dashv\vdash \top$ and $\neg\top \dashv\vdash \bot)$
Adjacency	$((\varphi \vee \psi) \wedge (\varphi \vee \neg\psi) \dashv\vdash \varphi$ and $(\varphi \wedge \psi) \vee (\varphi \wedge \neg\psi) \dashv\vdash \varphi)$

Proof of Soundness of Equivalence System E.S.

In general, the soundness of some system (whether of inference of equivalence) is easier to prove than its completeness. Indeed, the soundness of any system of equivalence S simply depends on the soundness of its individual rules of equivalence: S is sound if and only if all its equivalence rules are sound.

To see this, consider some system S that has an unsound equivalence rule. That is, S has a rule $\varphi \dashv\vdash \psi$, but it is not true that $\varphi \Leftrightarrow \psi$. Well, then obviously $\varphi \dashv\vdash_S \psi$, but not $\varphi \Leftrightarrow \psi$. Hence, S itself is unsound.

On the other hand, suppose all rules of S are sound. Then we can use mathematical induction to show that S itself is sound as well. In particular, suppose $\varphi \dashv\vdash_S \psi$. Then φ can be transformed using the rules of S into ψ using k steps for some number k. We'll show that the result φ_i after each step i with $0 \leq i \leq k$ is equivalent to φ.

Base: k = 0. $\varphi_0 = \varphi$, so obviously $\varphi \Leftrightarrow \varphi_0$.

Step: The induction hypothesis is that $\varphi \Leftrightarrow \varphi_i$. We now have to show that $\varphi \Leftrightarrow \varphi_{i+1}$. φ_{i+1} was obtained by applying one of the equivalence rules of S to φ_i. Since each equivalence rule of S was sound, we have that $\varphi_i \Leftrightarrow \varphi_{i+1}$. Thus, by transitivity of $\Leftrightarrow$, and the induction hypothesis, it follows that $\varphi \Leftrightarrow \varphi_{i+1}$.

For our particular system E.S., we therefore simply have to show that each of its equivalence rules is sound. This is all easy to show: we have shown many of them already, and the others are left to the reader.

Proof of Completeness of Equivalence System E.S.

Ah, now things get interesting! We have to show that for any φ and ψ: If $\varphi \Leftrightarrow \psi$, then $\varphi \dashv\vdash_{E.S.} \psi$. How do we do that?

First of all, we have to note that when we say 'any' φ and ψ, we should obviously limit ourselves to those statement involving logical operators for which E.S. has defined any equivalence rules at all. That is, it would be rather unfair to call E.S. incomplete just because it cannot prove the contraposition equivalence of $\varphi \to \psi \;\Leftrightarrow\; \neg\psi \to \neg\varphi$, simply because it doesn't have any rules defined for the conditional! Given that E.S. deals only with the Boolean operators $\wedge$, $\vee$, and $\neg$, as well as the statements $\bot$ and $\top$, it stands to reason to restrict ourselves to statements that involve only those very logical symbols.

To do this, we can appeal to formal syntax. That is, we can use formal syntax to define the set of Boolean statements L_B (it should be obvious how to do this), and then rephrase our questions of completeness as follows: for any φ and $\psi \in L_B$: If $\varphi \Leftrightarrow \psi$, then $\varphi \dashv\vdash_{E.S.} \psi$?

To prove this result, first define the *canonical disjunctive normal form* CDNF(φ,ψ) of statement $\varphi \in L_B$ relative to a statement $\psi \in L_B$ as the statement that is a DNF of φ obtained as follows:

1. Consider A(φ, ψ) to be the set of atomic sentences occurring in either φ or ψ. If A(φ, ψ) is empty, then that means that both φ and ψ are expressions of the purely logical symbols $\wedge$, $\vee$, $\neg$, $\bot$ and $\top$. In that case, φ is equivalent to either $\bot$ or $\top$, and we'll set CDNF(φ,ψ) = $\bot$ or CDNF(φ,ψ) = $\top$ respectively. Otherwise, proceed as follows.
2. Create a truth-table with reference columns consisting of A(φ, ψ). Order the reference columns according to some ordering O. Fill out the truth-values in the reference columns by having the right-most column alternate every one, the second to last column alternate every 2, etc. (In short: make a truth-table as if one were to analyze both φ and ψ, using the standard practices with regard to creating and filling out the reference columns).
3. In this truth-table, work out the truth-conditions for φ.
4. For every row where φ works out to True, create the conjunction of literals corresponding to that row, ordered in the same way as the reference columns are ordered.
5. Finally, disjunct together all those conjunctions in the order as generated from top to bottom. If there are no rows where φ is True (i.e. if φ is a contradiction), then define CDNF(φ,ψ) = $\bot$.

In other words, we create a DNF of φ using a truth-table as we have seen in the slides/lecture, but 1) we impose a fixed order on the conjuncts and disjuncts, and 2) the DNF not only looks at the atomic sentences of φ, but also takes into account atomic variables occurring in ψ. It is easy to see that in doing so, if $\varphi \Leftrightarrow \psi$, then CDNF($\varphi$, ψ) = CDNF(ψ, φ), because if A(φ, ψ) is empty, then either CDNF(φ, ψ) = CDNF(ψ, φ) = $\bot$, or CDNF(φ, ψ) = CDNF(ψ, φ) = $\top$, and if A(φ, ψ)

is not empty, then the truth-table as generated by this method, and thus the corresponding CDNF, will be exactly the same for φ relative to ψ, as for ψ relative to φ.

We will now prove that for any φ and ψ: $\varphi \dashv\vdash_{E.S.}$ CNDF(φ, ψ). That is, we will show that if we have some statement φ, then E.S. can transform this into its canonical DNF relative to ψ.

To transform any statement $\varphi \in L_B$ into its Canonical DNF, follow the steps below, adding parentheses as needed, dropping parentheses where possible. Also, because of Association, we can act as if we are dealing with generalized conjunctions and disjunctions:

1. Use the Identity, Annihilation, and Inverse laws to simplify expressions involving $\bot$ and $\top$. Note that this will either eliminate all $\bot$ and $\top$'s, or one ends up with $\bot$ or $\top$ as the final statement. If the resulting statement is $\bot$, then the statement is in CDNF, and we can stop. We'll also stop if the statement is $\top$, and A(φ, ψ) is empty. If the statement is $\top$, and A(φ, ψ) is not empty, then use Complement to replace with $A \vee \neg A$ where $A \in$ A(φ, ψ), and proceed.
2. Put the statement into NNF by repeated applications of DeMorgan and Double Negation.
3. Put the statement into DNF by repeated applications of Distribution of $\wedge$ over $\vee$.
4. Use Adjacency to ensure that every conjunction contains a literal A or $\neg$A for every $A \in$ A(φ, ψ). That is: use Adjacency to 'add' literals that are 'missing' in any conjunction. E.g. if A(φ, ψ) ={P,Q, R}, then a conjunct such as $P \wedge Q$ becomes $(P \wedge Q \wedge R) \vee (P \wedge Q \wedge \neg R)$
5. Use Commutation to order the conjuncts in any conjunction according to O. E.g. $A \wedge B \wedge A \wedge \neg A$ becomes $A \wedge A \wedge \neg A \wedge B$
6. Use Idempotence to eliminate duplicate conjuncts. E.g. $A \wedge A \wedge \neg A \wedge B$ becomes $A \wedge \neg A \wedge B$
7. Use Complement and Annihilation to replace any conjunction containing a literal and its negation with a contradiction symbol. E.g. $A \wedge \neg A \wedge B$ becomes $\bot \wedge B$ becomes $\bot$.
8. Use Identity to get rid of all disjuncts that are $\bot$. If there is only one disjunct $\bot$ left, then the statement is in CDNF and the process can stop.
9. Use Idempotence to eliminate any duplicate disjuncts.
10. Finally, use Commutation to get all disjuncts in the 'right' order.

Finally, from this result, completeness of E.S. easily follows. If $\varphi \Leftrightarrow \psi$, then $\varphi \dashv\vdash_{E.S.}$ CNDF(φ, ψ) = CDNF(ψ, φ) $\dashv\vdash_{E.S.} \psi$, i.e. $\varphi \dashv\vdash_{E.S.} \psi$.

By adding the Implication and Equivalence equivalence principles as equivalence rules to this system, it should be obvious that we extend our result to include sentences involcing conditionals and biconditionals: as a first step process, first use Equivalence to convert every biconditional to conditionals, then use Implication to convert all conditional to Boolean operators, and then proceed as above.