Sequent Calculus

This tutorial concerns Boolean operations on predicates, and the semantic notion of satisfaction, $\vDash$.
It is **important** for you to have attempted the exercises before going to the tutorial.

Semantics

For any universe U, a predicate is a function `p :: U -> Bool`. Sometimes it is helpful to consider the corresponding subset of the universe $[\![p]\!] = \{x \in U \mid p\ x\}$.

We have defined boolean operations of negation, disjunction, and conjunction on predicates in terms of the corresponding operations on the Boolean type `Bool`.

```
neg :: (u -> Bool) -> (u -> Bool)
neg p x = not (p x)
```

```
(|:|) :: (u -> Bool) -> (u -> Bool) -> (u -> Bool)
(|:|) p q x = p x || q x

(&:&) :: (u -> Bool) -> (u -> Bool) -> (u -> Bool)
(&:&) p q x = p x && q x
```

```
bigOr :: [u -> Bool] -> (u -> Bool)
bigOr ps x = or [p x | p <- ps]

bigAnd :: [u -> Bool] -> (u -> Bool)
bigAnd ps x = and [p x | p <- ps]
```

We will use $\neg$, $\vee$, $\bigvee$, $\wedge$, $\bigwedge$ as mathematical notation for these operations on predicates. We also have predicates called top (`top :: u -> Bool`) and bottom (`bot :: u -> Bool`) defined by `top _ = True` and `bot _ = False`, corresponding to the entire set U and the empty set $\emptyset$. We use constants $\top$ and $\bot$ to represent `top` and `bot`. These constants and operations on predicates are distinguished (by their types) from the corresponding constants and operations on Booleans.

The semantic notion of satisfaction is a relation that may or may not hold between two finite sets (Γ and Δ) of predicates in a given universe U. It is written as a sequent $\Gamma \vDash \Delta$ and defined in terms of the corresponding relation $a \vDash b$ between single predicates.

$$\Gamma \vDash \Delta \textbf{ iff } \bigwedge \Gamma \vDash \bigvee \Delta$$

This means that we interpret the antecedents (predicates before the turnstile) as a conjunction, and the succedents (predicates after the turnstile) as a disjunction.

The following rules are sound:

$$\frac{}{\Gamma, a \vDash a, \Delta}\ I$$

$$\frac{\Gamma \vDash a, \Delta}{\Gamma, \neg a \vDash \Delta}\ \neg L \qquad \frac{\Gamma, a \vDash \Delta}{\Gamma \vDash \neg a, \Delta}\ \neg R$$

$$\frac{\Gamma, a, b \vDash \Delta}{\Gamma, a \wedge b \vDash \Delta}\ \wedge L \qquad \frac{\Gamma \vDash a \quad \Gamma \vDash b, \Delta}{\Gamma \vDash a \wedge b, \Delta}\ \wedge R$$

$$\frac{\Gamma, a \vDash \Delta \quad b \vDash \Delta}{\Gamma, a \vee b \vDash \Delta}\ \vee L \qquad \frac{\Gamma \vDash a, b, \Delta}{\Gamma \vDash a \vee b, \Delta}\ \vee R$$

Here, variables $a, b, \ldots$ are predicates, while Γ, Δ are (finite) sets of predicates. To make the notation less cluttered, we write Γ, p for $\Gamma \cup \{p\}$ and q, Δ for $\{q\} \cup \Delta$. The double line signifies a two-way rule: the conclusion is valid iff all of the assumptions are valid.

If a rule is sound, we can add Γ, Δ to the left, right (respectively) of each turnstile, and still have a sound rule. This follows from the fact, shown in class, that,

$$\Gamma, a \vDash b, \Delta \text{ holds in universe } U \textbf{ iff } a \vDash b \text{ holds in the sub-universe } \{x \in U \mid \bigwedge \Gamma\ x, \neg \bigvee \Delta\ x\}$$

Just as in algebra, where we manipulate arithmetic expressions to solve equations, we will manipulate logical expressions to solve problems in logic. In algebra, variables range over numbers. Expressions such as $\sqrt{b^2 - 4ac}$ are formed from constants 2, 4 and variables a, b, c, using operations $\sqrt{\ }$, $\times$, $-$, $(.)^{(.)}$. In propositional logic, we have variables a, b, c that range over predicates and two constants $\top, \bot$. Expressions are built from variables and constants using operations that include $\neg$, $\wedge$, $\vee$. Although we can, and will, use Boolean algebra to manipulate equations, it is easier in propositional logic to work with sequents, so we will start with these. We will return later to look at equations.

1 Reduction

As shown in class, we can use these rules to reduce any sequent to a conjunction of *simple* sequents. These simple sequents $\Gamma \vDash \Delta$ include no operations, Γ and Δ are simply finite sets of variables such that $\Gamma \cap \Delta = \emptyset$ (so the immediate rule does not apply).

In the simplest examples, this conjunction is empty; we reduce the starting sequent to an empty conjunction. This means that the sequent we started from is universally valid. For example, we can make the following reduction, starting from the sequent, $x \wedge (y \vee z) \vDash (x \wedge y) \vee (x \wedge z)$:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{x, y \vDash x, (x \wedge z)} \qquad \dfrac{}{x, y \vDash y, (x \wedge z)}}{x, y \vDash (x \wedge y), (x \wedge z)} \qquad \dfrac{\dfrac{}{x, z \vDash (x \wedge y), x} \qquad \dfrac{}{x, z \vDash (x \wedge y), z}}{x, z \vDash (x \wedge y), (x \wedge z)}}{x, (y \vee z) \vDash (x \wedge y), (x \wedge z)}}{x, (y \vee z) \vDash (x \wedge y) \vee (x \wedge z)}}{x \wedge (y \vee z) \vDash (x \wedge y) \vee (x \wedge z)}$$

To perform the reduction, we simply work backwards, at each step we choose a rule that will eliminate a connective $(\wedge, \vee, \neg)$ from the left or right side of the turnstile. When there is a choice, it doesn't matter which side we do first (but it may save duplication if you use rules with a single premise before those with two premises).

In this example, each branch of the tree ends in an instance of the immediate rule. For the immediate rule, there are no statements above the line, so the conclusion is valid. Working down the tree each inference line corresponds to a sound rule, so we see that the bottom line is valid.

1. Label each inference line in the tree with the name of the rule applied.
2. Reduce the following sequent, $(x \wedge y) \vee (x \wedge z) \vDash x \wedge (y \vee z)$, in similar fashion.
3. Is the equation, $x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$, valid in every universe? Explain your answer.

Use reductions to show that the following equations are valid in every universe.

4. $(x \wedge y) \vee (x \wedge z) = x \wedge (y \vee z)$
5. $\neg(x \wedge y) = \neg x \vee \neg y$
6. $\neg x \wedge \neg y = \neg(x \vee y)$

Sometimes the sequent we start from is not universally valid. For example, we can reduce the sequent $\vDash (a \wedge \neg b) \vee (\neg a \wedge b)$, to derive a new rule.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{a \vDash a,}}{\vDash a, \neg a} \qquad \vDash a, b}{\vDash a, \neg a \wedge b} \qquad \dfrac{\dfrac{a, b \vDash}{\vDash \neg b, \neg a} \qquad \dfrac{\dfrac{}{b \vDash b}}{\vDash \neg b, b}}{\vDash \neg b, \neg a \wedge b}}{\vDash a \wedge \neg b, \neg a \wedge b}}{\vDash (a \wedge \neg b) \vee (\neg a \wedge b)}$$

gives the rule $\dfrac{\vDash a, b \qquad a, b \vDash}{\vDash (a \wedge \neg b) \vee (\neg a \wedge b)}$

The reduction ends with two simple sequents. These form the premises for a new rule; the conclusion is the original sequent, which is satisfied iff the two premises are satisfied.

The expression, $(a \wedge \neg b) \vee (\neg a \wedge b)$, actually defines the **exclusive or** function, xor: $a \oplus b \equiv (a \wedge \neg b) \vee (\neg a \wedge b)$.

So we have derived the rule, $\dfrac{\models a, b \quad a, b \models}{\models a \oplus b}$, and hence, $\dfrac{\Gamma \models a, b, \Delta \quad \Gamma, a, b \models \Delta}{\Gamma \models a \oplus b, \Delta} \; \oplus R$

7. Reduce the sequent $(a \wedge \neg b) \vee (\neg a \wedge b) \models$, and use the result to derive a rule $(\oplus L)$.
8. We define $a \to b \equiv \neg a \vee b$. Use reductions of the sequents $\models \neg a \vee b$ and $\neg a \vee b \models$ to produce rules $(\to R)$ and $(\to L)$.
9. We define, $a \leftrightarrow b \equiv (a \wedge b) \vee (\neg a \wedge \neg b)$. Use reductions to derive rules $(\leftrightarrow R)$ and $(\leftrightarrow L)$
10. Is the trivial sequent $\emptyset \models \emptyset$, with the empty set on each side of the turnstile, valid in all models, valid in some models (if so describe them), or valid in no models?
11. If our original sequent is non-trivial, and includes no constants, then every sequent in the reduction tree will also be non-trivial. Explain why this is so.

If the expressions in our original sequent include constants, $\top, \bot$, then these will eventually occur naked on one or other side of the turnstile.

12. Give sound double-line rules $(\top L)$, $(\top R)$, $(\bot L)$, $(\bot R)$ that eliminate the constants.

Another way to use the result of a reduction of sequents of the form $\models \varphi$, is to move every atom in the premises of the derived rule to the right of the turnstile, using negation as necessary. This leads to a simple expression equivalent to φ.

For our example this gives the rule, $\dfrac{\models a, b \qquad \models \neg a, \neg b}{\models (a \wedge \neg b) \vee (\neg a \wedge b)}$, equivalent to $\dfrac{\models a \vee b \qquad \models \neg a \vee \neg b}{\models (a \wedge \neg b) \vee (\neg a \wedge b)}$.

Reading the premises as disjunctions, this means that semantically, $(a \wedge \neg b) \vee (\neg a \wedge b)$ is equivalent to $(a \vee b) \wedge (\neg a \vee \neg b)$

We can use this technique starting with any expression on the right-hand side of the turnstile, to find a conjunction of disjunctions of literals that is equivalent to the expression we start from. We call such a conjunction of disjunctions a conjunctive normal form (CNF).

Use this technique to give a CNF for the following expressions

13. $r \leftrightarrow (a \wedge b)$
14. $r \leftrightarrow (a \vee b)$
15. $r \leftrightarrow (a \to b)$
16. $r \leftrightarrow (\neg a)$

Use reductions to decide whether the equations below are valid (true in every universe).

17. $x \leftrightarrow y = (x \to y) \wedge (y \to x)$
18. $(x \to y) \to z = x \to (y \to z)$
19. $(x \leftrightarrow y) \leftrightarrow z = x \leftrightarrow (y \leftrightarrow z)$
20. $(x \oplus y) \oplus z = x \oplus (y \oplus z)$

This tutorial shows that any sequent can be reduced to a CNF. By falsifying any conjunct in the CNF we can falsify the original sequent. If the CNF is the empty conjunction, then we have shown that the original sequent is universally valid.

21. How could you use the techniques developed in this tutorial to check whether a sequent is satisfiable (valid in some model)?