CSC B36 Additional Notes

proving a set of connectives complete, and not complete

⋆ Introduction

For this course, you are expected to formally prove that a given set of boolean connectives is complete. You are also expected to prove that a given set of connectives is *not* complete. These notes provide a guide to proving completeness and incompleteness for a set of connectives.

First … some definitions …

○ Boolean functions

Given an integer $n > 0$, a *boolean function* (of n inputs) is a function that takes n binary values as input and returns one binary value as output. I.e., the function maps from $\{0,1\}^n$ to $\{0,1\}$. For example, the *Agreement* function, defined by

$$Agreement(x,y,z) = \begin{cases} 1 & \text{if } x = y = z; \\ 0 & \text{otherwise,} \end{cases}$$

takes 3 binary values as input, and returns 1 if all 3 input values are equal, and returns 0 if the input values are not all the same.

○ Representing a boolean function

A propositional formula F with propositional variables $x_1, \cdots, x_n$ is said to *represent* a boolean function f of n inputs iff

for any truth assignment τ,
  τ satisfies F whenever $f(\tau(x_1), \cdots, \tau(x_n)) = 1$, and
  τ falsifies F whenever $f(\tau(x_1), \cdots, \tau(x_n)) = 0$.

Notice that logically equivalent formulas always represent the same boolean function.

○ Completeness for a set of connectives

A set C of connectives is said to be *complete* iff every boolean function can be represented by a propositional formula that uses only connectives in C. From the course notes, we have $\{\neg, \wedge\}$ and $\{\neg, \vee\}$ as examples of complete sets.

Note:
Any formula that uses *no connectives at all* also uses only connectives in *any* set of connectives.
E.g., the formula x uses only connectives in $\{\wedge, \vee\}$.

Abbreviation:
We use *uoc* as an abbreviation for *uses only connectives in.*
E.g., “F uoc C” means “F uses only connectives in C”.

⋆ Proving a set C is complete

To prove that a set C of connectives is complete, we start with a known complete set B of connectives.[1] Then we prove that

[1] For this courses, usually the only sets of connectives that we can assume to be complete are $\{\neg, \wedge\}$ and $\{\neg, \vee\}$.

for every formula F that uoc B, there exists a formula F' such that
F' uoc C and F' LEQV F.

Given any boolean function f, since B is complete, f can be represented by some formula, say F, that uoc B. Then by what we proved, there is some formula F' such that F' uoc C and F' LEQV F. Therefore every boolean function can be represented by some formula that uoc C as wanted.

Here are the steps to formally prove that a set C is complete.

1. Use structural induction to define the set $\mathcal{G}$ that uoc $\{\neg, \wedge\}$ or $\{\neg, \vee\}$ (the choice is yours; either is acceptable).
2. Use structural induction to prove that for every formula $F \in \mathcal{G}$, there exists a formula F' such that F' uoc C and F' LEQV F.
3. Our result follows from the fact that $\{\neg, \wedge\}$ (or $\{\neg, \vee\}$ if you chose it) is complete.

- **Example of a proof that a set is complete**

Consider the unary connective $\underline{0}$, where $\underline{0}P$ is always falsified, regardless of whether P is satisfied or falsified.

Here is a proof that $\{\underline{0}, \rightarrow\}$ is complete.

[step 1]
We define the set $\mathcal{G}$ of formulas that uoc $\{\neg, \vee\}$.

Let $\mathcal{G}$ be the smallest set such that
BASIS: If x is a propositional variable, then $x \in \mathcal{G}$.
INDUCTION STEP: If $F_1, F_2 \in \mathcal{G}$, then $\neg F_1, (F_1 \vee F_2) \in \mathcal{G}$.

[step 2]
Now we prove that for every formula $F \in \mathcal{G}$, there exists a formula F' such that
F' uoc $\{\underline{0}, \rightarrow\}$ and F' LEQV F.

BASIS: Let $F = x$, where x is a propositional variable.
Now consider $F' = x$.
Then F' uoc $\{\underline{0}, \rightarrow\}$ [F' uses no connectives at all]
and F' LEQV F [$F' = F$]
as wanted.

INDUCTION STEP: Let $F_1, F_2 \in \mathcal{G}$.
Suppose there are formulas F_1' and F_2' such that
F_1' and F_2' uoc $\{\underline{0}, \rightarrow\}$ and F_1' LEQV F_1 and F_2' LEQV F_2. [IH]

There are two cases to consider: $F = \neg F_1$ and $F = (F_1 \vee F_2)$.

Case 1: For $F = \neg F_1$, let $F' = (F_1' \rightarrow \underline{0}F_1')$.
Then F' uoc $\{\underline{0}, \rightarrow\}$ [by IH, F_1' uoc $\{\underline{0}, \rightarrow\}$]

and $F' = (F_1' \rightarrow \underline{0}F_1')$
LEQV $(F_1 \rightarrow \underline{0}F_1)$ [by IH, F_1' LEQV F_1]
LEQV $\neg F_1$ [$\underline{0}F_1$ is always falsified, so
$(F_1 \rightarrow \underline{0}F_1)$ is satisfied exactly when F_1 is falsified]
$= F$
as wanted.

Case 2: For $F = (F_1 \vee F_2)$, let $F' = ((F_1' \to \underline{0}F_1') \to F_2')$.

$$\begin{array}{lll}
\text{Then } F' \text{ uoc } \{\underline{0}, \to\} & & [\text{by IH, } F_1' \text{ and } F_2' \text{ uoc } \{\underline{0}, \to\}] \\
\text{and } F' = ((F_1' \to \underline{0}F_1') \to F_2') & & \\
\quad \textsc{leqv} \ ((F_1 \to \underline{0}F_1) \to F_2) & & [\text{by IH, } F_1' \ \textsc{leqv} \ F_1 \text{ and } F_2' \ \textsc{leqv} \ F_2] \\
\quad \textsc{leqv} \ (\neg F_1 \to F_2) & & [\text{by case 1, } \neg F_1 \ \textsc{leqv} \ (F_1 \to \underline{0}F_1)] \\
\quad \textsc{leqv} \ (\neg\neg F_1 \vee F_2) & & [\to \text{law}] \\
\quad \textsc{leqv} \ (F_1 \vee F_2) & & [\text{double negation}] \\
\quad = F & &
\end{array}$$

as wanted. □

[step 3]
Since $\{\neg, \vee\}$ is complete, therefore $\{\underline{0}, \to\}$ is also complete. □

○ Informally proving a set C is complete

The main ideas behind the above proof are that $\neg F$ LEQV $(F \to \underline{0}F)$ and $F_1 \vee F_2$ LEQV $((F_1 \to \underline{0}F_1) \to F_2)$.

In general, an informal proof that a set C is complete consists of showing how each connective in $\{\neg, \vee\}$ (or in $\{\neg, \wedge\}$) can be expressed equivalently in terms of the connectives in C.

⋆ Proving a set C is not complete

To prove that a set C of connectives is not complete, we start by finding a property (expressed as a predicate) that every formula that uoc C has, but not every formula in general. Then we prove that every formula that uoc C has the desired property. Finally, we give a specific formula F for which our property does not hold (by necessity, this F must use some connective that is not in C). Since every formula that uoc C must have the property, so no formula that uoc C represents the boolean function represented by F. Therefore C is not complete.

Here then are the steps to formally prove that a set C is not complete.

1. Use structural induction to define the set $\mathcal{H}$ of formulas that uoc C.
2. Define a predicate $P(F)$ that holds for every $F \in \mathcal{H}$, but not in general.
3. Use structural induction to prove that $P(F)$ holds for every formula $F \in \mathcal{H}$.
4. Give a specific formula F and show that $P(F)$ does not hold.
   Then our result follows as argued above.

○ Example of a proof that a set is not complete

Consider the unary connective $\underline{1}$, where $\underline{1}P$ is always satisfied, regardless of whether P is satisfied or falsified.

Here is a proof that $\{\underline{1}, \to\}$ is not complete.

[step 1]
We define the set $\mathcal{H}$ of formulas that uoc $\{\underline{1}, \to\}$.

> Let $\mathcal{H}$ be the smallest set such that
> BASIS: If x is a propositional variable, then $x \in \mathcal{H}$.
> INDUCTION STEP: If $F_1, F_2 \in \mathcal{H}$, then $\underline{1}F_1, (F_1 \to F_2) \in \mathcal{H}$.

[step 2]
For a formula F, we define predicate $P(F)$ as follows.

$P(F)$: $\tau_1^*(F) = 1$,

where τ_1 is the truth assignment that assigns 1 to every variable.
In other words, $P(F)$ says F is satisfied whenever all its variables are assigned TRUE.

[step 3]
We prove that $P(F)$ holds for every $F \in \mathcal{H}$.

BASIS: Let $F = x$, where x is a propositional variable.

$$\begin{aligned} \text{Then } \tau_1^*(F) &= \tau_1^*(x) && [F = x] \\ &= \tau_1(x) && [\text{definition of } \tau_1^* \text{ with argument } x] \\ &= 1 && [\text{definition of } \tau_1] \end{aligned}$$

as wanted.

INDUCTION STEP: Let $F_1, F_2 \in \mathcal{H}$.
Suppose $P(F_1)$ and $P(F_2)$. [IH]
I.e., τ_1 satisfies both F_1 and F_2.

There are two cases to consider: $F = \underline{1}F_1$ and $F = (F_1 \to F_2)$.

Case 1: For $F = \underline{1}F_1$, we have

$$\begin{aligned} \tau_1^*(F) &= \tau_1^*(\underline{1}F_1) && [F = \underline{1}F_1] \\ &= 1 && [\underline{1}F_1 \text{ is always satisfied}] \end{aligned}$$

as wanted.

Aside: IH was not used here. All steps are valid, even if τ_1 were any other truth assignment.

Case 2: For $F = (F_1 \to F_2)$, we have

$$\begin{aligned} \tau_1^*(F) &= \tau_1^*(F_1 \to F_2) && [F = (F_1 \to F_2)] \\ &= 1 && [\text{by IH, } \tau_1 \text{ satisfies } F_2\text{; so } \tau_1 \text{ also satisfies } (F_1 \to F_2)] \end{aligned}$$

as wanted. □

[step 4]
Now consider the formula $F = \neg x$.

$$\begin{aligned} \text{Then } \tau_1^*(F) &= \tau_1^*(\neg x) && [F = \neg x] \\ &= 0. && [\tau_1 \text{ satisfies } x\text{; so } \tau_1 \text{ falsifies } \neg x] \end{aligned}$$

Thus $P(F)$ does not hold.

Therefore $\{\underline{1}, \to\}$ is not complete. □

- **Informally proving a set C is not complete**

The main ideas behind the above proof lie in finding the predicate $P(F)$ and the specific formula $F = \neg x$.

In general, an informal proof that a set C is not complete consists of doing steps 2 and 4.