CS 5860 Atomic evidence, more examples of computational meaning of logical formulas, proofs

Thur Sept 22, 2011

PLAN

1. Review atomic evidence and evidence semantics
2. More examples
3. Comparing tableau proof and computational evidence
4. Proof rules for Refinement Logic (Computational Tableaux)

Atomic evidence

For the simple logic of $\&, \vee, \Rightarrow, \bot$ we don't have detailed knowledge of atomic evidence, that is, evidence for the propositional variables like P, Q, R, X, Y, Z, etc. When we want to say that P is known, we say $p_i \in [P]$ and mean that p_i is atomic evidence. Like P itself, we don't analyze the structure of p_i further.

In the previous lecture we picked specific examples of atomic propositions such as $0=0$, $0=1$, $3<5$, etc. The evidence for $0=0$ is " the symbols on each side of $=$ are identical." This is so primitive a computation that we just say $[0=0] = \{\text{are\_identical}\}$ or $\{\text{axiom}\}$.

We examined the more complex case of $n<m$, say $3<5$, and defined $3<5$ as $\exists p:\mathbb{N}^+.(3+p=5)$, take $p=2$. We the proved informally $n<m \Rightarrow n<m+1$; this requires showing $\exists p:\mathbb{N}^+.(n+p=m) \Rightarrow \exists p':\mathbb{N}^+.(n+p'=m+1)$. It is clear how to prove this. Let $n+p=m$, then $n+(p+1)=m+1$.

CS 5860

Thur Sept 22 - Continued 2.

Computational evidence for $P \vee (Q \& R) \Rightarrow (P \vee Q) \& (P \vee R)$.

The computational meaning of the formula is the ML type where the functions are total and do not give exceptions. Here is the function in two notations.

ML notation

```
\x. if isl(x) then <x, x>
      else let x = <q, r> in <inr(q), inr(r)>
```

Nuprl notation

```
λ(x. decide(x; p. <inl(p), inl(p)>;
                    qr. spread(qr; q, r. <inr(q), inr(r)>)))
```

Aside, for those who studied the Smullyan tableau rules, here is a tableau proof.

```
                    1. F( P∨(Q&R) ⇒ (P∨Q)&(P∨R))
                    2. T(P∨(Q&R))   from 1
                    3. F((P∨Q)&(P∨R))   from 1
       ______________________________________________________
     4. TP  from 2                                5. T(Q&R)  from 2
   __________________
14. F(P∨Q) from 3   15. F(P∨R) from 3            6. TQ  from 5
                                                 7. TR  from 5
17. FP              16. FP  from 15          ______________________________
  ⋇ 4,17              ⋇ 4,16               8. F(P∨Q) from 3      9. F(P∨R) from 3
                                           12. FP                10. FP
                                           13. FQ                11. FR
                                              ⋇ 6,13                ⋇ 7,11
```

closed tableau

CS 5860

Thur Sept. 22, 2011 continued 3. Another example

"Currying" $\underset{f}{((P\&Q) \Rightarrow R)} \Rightarrow \underset{p}{P} \Rightarrow (\underset{q}{Q} \Rightarrow R)$

ML program as evidence

```
\f. \p. \q.(f <p,q>)
```

Nuprl program

```
λ(f. λ(p. λ(q. ap(f; <p,q>) ) ) )
```

"Un-Curry" $(P \Rightarrow \underset{f}{(Q \Rightarrow R)}) \Rightarrow \underset{x}{(P\&Q)} \Rightarrow R$

ML program

```
\f. \x. ( let x = <p,q> in (f p) q)
```

Nuprl program

```
λ(f. λ(x. spread(x; p,q. ap(ap(f;p); q) ) ) )
```

Defining negation "computationally."

Let False be an atomic constant which has no evidence.
To say that P is false we show (P ⇒ False).
Define ~P as (P ⇒ False)
The evidence type for False is empty, say { } or Void.

Here is a theorem involving False. $(P \,\&\, {\sim}P) \Rightarrow \text{False}$, that is $(P \,\&\, (P \Rightarrow \text{False})) \Rightarrow \text{False}$. The evidence is

Nuprl program : `λ(h. spread(h; p, np. ap(np; p)))`

ML does not have a void type, but we can use the type constant any. for False

ML program `\h. let h = <p, np> in (np p)`

CS 5860

Thur Sept 22, 2011 continued 4

More about negation

Notice that there is no evidence for (P ∨ ~P). The evidence would be either inl(p) or inr(np) for some evidence p for P or some function np: P ⇒ False. We can know this for constants. Suppose we define True = (False ⇒ False) then λ(x.x) ∈ True. So we know (True ∨ ~True), the evidence is inl(λ(x.x)).

For other constants we have no idea, e.g.

(SAT ∈ PTime) ∨ ~(SAT ∈ PTime)

Here are some theorems we can prove about negation

1. ~(P ∨ Q) ⇒ ~P & ~Q ((P ∨ Q) ⇒ False) ⇒ (P ⇒ False) & (Q ⇒ False)

Nuprl program

λ(h. <λ(p. h(inl(p))), λ(q. h(inr(q)))>)

2. (P ⇒ Q) ⇒ (~Q ⇒ ~P) exercise

3. ~~(P ∨ ~P) same as ((P ∨ (P ⇒ False)) ⇒ False) ⇒ False

Check that this program works

λ(h. ap(h; inr(λ(p. ap(h; inl(p)))))) !

The proof rules will help us "type check" this program.

CS 5860 Supplementary Material (optional)

Thur Sept. 22, 2011

Beth Tableaux (Constructive Rules)

T&
$$\frac{S,\ T(X \& Y)}{S,\ TX,\ TY}$$

F&
$$\frac{F\ S,\ F(X \& Y)}{S,\ FX \quad \big| \quad S,\ FY}$$

T∨
$$\frac{S,\ T(X \vee Y)}{S, TX \quad \big| \quad S, TY}$$

F∨
$$\frac{S,\ F(X \vee Y)}{S,\ FX,\ FY}$$

(two goals)

T⇒
$$\frac{S,\ T(X \Rightarrow Y)}{S, FX \quad \big| \quad S, TY}$$

F⇒
$$\frac{S,\ F(X \Rightarrow Y)}{S_T,\ TX,\ FY}$$

T∼
$$\frac{S,\ T(\sim X)}{S,\ FX}$$

F∼
$$\frac{S,\ F{\sim}X}{S_T,\ TX}$$

$$S_T = \{ TX \mid TX \in S \}$$