CS313K: Logic, Sets, and Functions

J Strother Moore
Department of Computer Sciences
University of Texas at Austin

(Lecture 11)

Cheat Sheets for the Rules of Inference

To prove ψ using:

- **Tautology**: find a tautology and instantiate it to get ψ.

- **Rewrite**:
  - put ψ into the form $\psi_h \rightarrow (\ldots \alpha' \ldots)$ where α' is the term you want to change,
  - pick some theorem in the form $\phi_h \rightarrow (\alpha = \beta)$ or $\phi_h \rightarrow (\alpha \leftrightarrow \beta)$,
  - match α with α' so that $\alpha/\sigma = \alpha'$,
  - relieve the hypotheses by proving $\psi_h \rightarrow \phi_h/\sigma$,
  - make sure the equivalence ("$=$" or "$\leftrightarrow$") is ok,
  - replace α' with β/σ.

- **Hypothesis**:
  - pick a term α' in the conclusion of ψ to rewrite with some hypothesis $(\alpha = \beta)$ or $(\alpha \leftrightarrow \beta)$,
  - make sure the equivalence ("$=$" or "$\leftrightarrow$") is ok,
  - replace α with β.

- **Cases**:
  - pick an exhaustive set of terms $(\phi'_1 \vee \ldots \vee \phi'_k)$
  - for each, prove $\phi'_i \rightarrow \psi$.

```
T1: (equal (rev (app x y))
           (app (rev y) (rev x)))

T2: (iff (true-listp (app x y))
         (true-listp y))

T3: (implies (true-listp x)
             (equal (rev (rev x)) x))
```

T1: (rev (app x y)) = (app (rev y) (rev x))

T2: (true-listp (app x y)) $\leftrightarrow$ (true-listp y)

T3: (true-listp x) $\longrightarrow$ (rev (rev x)) = x

Theorem:

```
(implies (and (true-listp a) (true-listp b))
         (true-listp (rev (app (rev a) b))))
```

```
(implies (and (true-listp a) (true-listp b))
         (true-listp (app (rev b)
                          (rev (rev a)))))
```

```
(implies (and (true-listp a) (true-listp b))
         (true-listp (rev (rev a))))
```

```
(implies (and (true-listp a) (true-listp b))
         (true-listp a))
```

```
(implies (and (true-listp a) (true-listp b))
         t)
```

t

The Rules of Inference are precisely described on pages 56–58.

The reason they're described precisely is so you can learn to do proofs without making mistakes.

I don't care if you learn the "implementation" of the rules. Who cares what π is in your steps? I don't!

But you must learn *how to use* the rules flawlessly and naturally.

Theorem:

```
(implies (and (true-listp a) (true-listp b))
         (true-listp (rev (app (rev a) b))))
```

Transformation 1 (Rewrite: Steps 1 and 2):

```
(implies (and (true-listp a) (true-listp b))
         (true-listp (rev (app (rev a) b))))
```

Rewrite at π=(2 1)

Transformation 1 (Rewrite: Steps 1 and 2):

```
(implies (and (true-listp a) (true-listp b))
         (true-listp (rev (app (rev a) b))))
```

Rewrite at π=(2 1) with

```
T1: (equal (rev (app x y))
           (app (rev y)
                (rev x)))
```

Transformation 1 (Rewrite: Steps 3 and 4):

```
(implies (and (true-listp a) (true-listp b))
         (true-listp (rev (app (rev a) b))))
```

Rewrite at π=(2 1) with

```
T1: (implies t                          ; φ_h
             (equal (rev (app x y))     ; α =
                    (app (rev y)        ; β
                         (rev x)))))
```

eqv = equal

σ={x $\leftarrow$ (rev a), y $\leftarrow$ b}

Transformation 1 (Rewrite: Steps 5 and 6):

```
(implies (and (true-listp a) (true-listp b))
         (true-listp (rev (app (rev a) b))))
```

Rewrite at π=(2 1) with

```
T1: (implies t                              ; φ_h
             (equal (rev (app x y))         ; α
                    (app (rev y)            ; β
                         (rev x))))
```

eqv = equal

σ={x $\leftarrow$ (rev a), y $\leftarrow$ b}

Prove: ((true-listp a) $\wedge$ (true-listp b)) $\rightarrow$ t

Transformation 1 (Rewrite: Step 7):

```
(implies (and (true-listp a) (true-listp b))
         (true-listp (rev (app (rev a) b))))
```

Rewrite at π=(2 1) with

```
T1: (implies t                              ; φ_h
             (equal (rev (app x y))         ; α
                    (app (rev y)            ; β
                         (rev x))))
```

eqv = equal

σ={x $\leftarrow$ (rev a), y $\leftarrow$ b}

β/σ = (app (rev b)
                (rev (rev a)))

Transformation 1 (Rewrite: Step 7):

```
(implies (and (true-listp a) (true-listp b))
         (true-listp (app (rev b)
                          (rev (rev a)))))
```

Rewrite at π=(2 1) with

```
T1: (implies t                              ; φ_h
             (equal (rev (app x y))         ; α
                    (app (rev y)            ; β
                         (rev x))))
```

eqv = equal

σ=$\{$x $\leftarrow$ (rev a), y $\leftarrow$ b$\}$

β/σ = (app (rev b)
                  (rev (rev a)))

Transformation 2:

```
(implies (and (true-listp a) (true-listp b))
         (true-listp (app (rev b)
                          (rev (rev a)))))
```

Transformation 2 (Rewrite: Steps 1,2,3,4,5,6):

```
(implies (and (true-listp a) (true-listp b))
         (true-listp (app (rev b)
                          (rev (rev a)))))
```

Rewrite at π= (2) with

```
T2: (iff (true-listp (app x y))     ; α ↔
         (true-listp y))            ; β
```

eqv = iff

σ = {x←(rev b), y←(rev (rev a))}

β/σ = (true-listp (rev (rev a)))

Prove (true-listp a) $\wedge$ (true-listp b) $\longrightarrow$ t

Transformation 2 (Rewrite: Steps 1,2,3,4,5,6):

```
(implies (and (true-listp a) (true-listp b))
         (true-listp (rev (rev a))))
```

Rewrite at π= (2) with

T2: (iff (true-listp (app x y)) ; $\alpha \leftrightarrow$
         (true-listp y))   ; β

eqv = iff

σ = $\{$x$\leftarrow$(rev b), y$\leftarrow$(rev (rev a))$\}$

β/σ = (true-listp (rev (rev a)))

Prove (true-listp a) $\wedge$ (true-listp b) $\rightarrow$ t

Transformation 3:

```
(implies (and (true-listp a) (true-listp b))
         (true-listp (rev (rev a))))
```

Transformation 3:

```
(implies (and (true-listp a) (true-listp b))
         (true-listp (rev (rev a))))
```

Transformation 3 (Rewrite: Steps 1,2,3,4,5):

```
(implies (and (true-listp a) (true-listp b))
         (true-listp (rev (rev a))))
```

Rewrite at π=(2 2) with

T3:(implies(true-listp x) ; ϕ_h

(equal (rev (rev x)) x));$\alpha = \beta$

eqv=equal

σ=$\{$x $\leftarrow$ a$\}$

β/σ=a

ϕ_h/σ = (true-listp a)

Transformation 3 (Rewrite: Steps 1,2,3,4,5):

```
(implies (and (true-listp a) (true-listp b))
         (true-listp (rev (rev a))))
```

Rewrite at π=(2 2) with

T3:(implies(true-listp x) ; ϕ_h

(equal (rev (rev x)) x));$\alpha = \beta$

eqv=equal

σ=$\{$x $\leftarrow$ a$\}$

β/σ=a

ϕ_h/σ = (true-listp a)

Prove (true-listp a) $\wedge$ (true-listp b)

$\longrightarrow$ (true-listp a)

Transformation 3 (Rewrite: Steps 1,2,3,4,5):

```
(implies (and (true-listp a) (true-listp b))
         (true-listp a))
```

Rewrite at π=(2 2) with

T3:(implies(true-listp x) ; ϕ_h
(equal (rev (rev x)) x));$\alpha = \beta$

eqv=equal

σ={x $\leftarrow$ a}

β/σ=a

ϕ_h/σ = (true-listp a)

Prove (true-listp a) $\wedge$ (true-listp b)
$\rightarrow$ (true-listp a)

Transformation 4:

```
(implies (and (true-listp a) (true-listp b))
         (true-listp a))
```

Transformation 4:

```
(implies (and (true-listp a)   ; α ↔ β (β = t
              (true-listp b))
         (true-listp a))       ; α
```

Use Hyp 1, δ = (true-listp a)

α = (true-listp a), β = t, eqv = iff

Transformation 4:

```
(implies (and (true-listp a)   ; α ↔ β (β = t
              (true-listp b))
         t)                    ; β
```

Use Hyp 1, δ = (true-listp a)

α = (true-listp a), β = t, eqv = iff

```
Transformation 5:
(implies (and (true-listp a)
              (true-listp b))
         t)

Taut: (implies p t)
```

σ=$\{$p$\leftarrow$(and (true-listp a) (true-listp b))$\}$

Transformation 6:

t

```
Thm (implies p t)
Proof:
(implies p t)
=  {rewrite with def implies}
(if p (if t t nil) t)

Case 1: p=nil
(if p (if t t nil) t)
= {by hyp 1}
(if nil (if t t nil) t)
= {by comp}
t
```

Case 2: p$\neq$nil (p $\leftrightarrow$ t)

```
(if p (if t t nil) t)
= {by hyp}
(if t (if t t nil) t)
= {by comp}
t
```

Q.E.D.

Your Questions

My Questions