cs113 Lab4: By Amuldeep Dhillon

Text written to file build.sh

```
doctex lab.doc
pptexenv latex lab.tex
dvipdf lab.dvi
```

Bourne Shell

```
chmod 777 build.sh
./build.sh
```

SML

```
"Mapping Rules of Inference to Proof Power";
val premise = asm_rule;
val modus_ponens = ⇒_elim;
val modus_tollens = modus_tollens_rule;
val disjunctive_syllogism = ∨_cancel_rule;
val double_negation = ¬_¬_elim;
val disjunctive_addition = ∨_right_intro;
val conjunctive_addition = ∧_intro;
val conjunctive_simplification = ∧_left_elim;
val conjunctive_simplificationR = ∧_right_elim;
val hypothetical_syllogism = ⇒_trans_rule;
val double_negative = ¬_¬_elim;
val double_negativeI = ¬_¬_intro;
val demorgan_disjunction = rewrite_rule [¬_∨_thm];
val demorgan_conjunction = rewrite_rule [¬_∧_thm];
```

```
val it = "Mapping Rules of Inference to Proof Power": string
:) val premise = fn: TERM -> THM
:) val modus_ponens = fn: THM -> THM -> THM
:) val modus_tollens = fn: THM -> THM -> THM
:) val disjunctive_syllogism = fn: THM -> THM -> THM
:) val double_negation = fn: THM -> THM
:) val disjunctive_addition = fn: TERM -> THM -> THM
:) val conjunctive_addition = fn: THM -> THM -> THM
:) val conjunctive_simplification = fn: THM -> THM
:) val conjunctive_simplificationR = fn: THM -> THM
:) val hypothetical_syllogism = fn: THM -> THM -> THM
:) val double_negative = fn: THM -> THM
:) val double_negativeI = fn: THM -> THM
:) val demorgan_disjunction = fn: THM -> THM
:) val demorgan_conjunction = fn: THM -> THM
```

cs113 Lab4: Rules of Inference: Modus Ponens

- Modus Ponens

$$\begin{array}{l} p \rightarrow q \\ \underline{p \qquad} \\ \therefore q \end{array}$$

SML

```
"Modus Ponens";
val p1 = premise ⌜p⇒q⌝;
val p2 = premise ⌜p:BOOL⌝;
modus_ponens p1 p2;
```

```
:) "Modus Ponens";
val p1 = premise ⌜p⇒q⌝;
val p2 = premise ⌜p:BOOL⌝;
modus_ponens p1 p2;
val it = "Modus Ponens": string
:) val p1 = p ⇒ q ⊢ p ⇒ q: THM
:) val p2 = p ⊢ p: THM
:) val it = p ⇒ q, p ⊢ q: THM
```

- Modus Tollens

$$\begin{array}{l} p \rightarrow q \\ \sim q \\ \hline \therefore \sim p \end{array}$$

SML

```
"Modus Tollens";
val p1 = premise ⌜p⇒q⌝;
val p2 = premise ⌜¬q:BOOL⌝;
modus_tollens p1 p2;
```

```
:) "Modus Tollens";
val p1 = premise ⌜p⇒q⌝;
val p2 = premise ⌜¬q:BOOL⌝;
modus_tollens p1 p2;
val it = "Modus Tollens": string
:) val p1 = p ⇒ q ⊢ p ⇒ q: THM
:) val p2 = ¬ q ⊢ ¬ q: THM
:) val it = p ⇒ q, ¬ q ⊢ ¬ p: THM
```

- Disjunctive Addition

$$\frac{p}{\therefore p \lor q}$$

SML

"Disjunctive Addition";
val p2 = ⌜q:BOOL⌝;
val p1 = premise ⌜p:BOOL⌝;
disjunctive_addition p2 p1;

```
:) "Disjunctive Addition";
val p2 =  ⌜q:BOOL⌝;
val p1 = premise ⌜p:BOOL⌝;
disjunctive_addition p2 p1;
val it = "Disjunctive Addition": string
:) val p2 = ⌜q⌝: TERM
:) val p1 = p ⊢ p: THM
:) val it = p ⊢ p ∨ q: THM
```

- Conjunctive Addition

$$\begin{array}{l} p \\ q \\ \hline \therefore p \wedge q \end{array}$$

SML

"Conjunctive Addition";
val p1 = premise ⌜p:BOOL⌝;
val p2 = premise ⌜q:BOOL⌝;
conjunctive_addition p1 p2;

```
:) "Conjunctive Addition";
val p1 = premise ⌜p:BOOL⌝;
val p2 = premise ⌜q:BOOL⌝;
conjunctive_addition p1 p2;
val it = "Conjunctive Addition": string
:) val p1 = p ⊢ p: THM
:) val p2 = q ⊢ q: THM
:) val it = p, q ⊢ p ∧ q: THM
```

cs113 Lab4: Rules of Inference: Conjunctive Simplification

- Conjunctive Simplification

$$\frac{p \wedge q}{\therefore p}$$

SML

```
"Conjunctive Simplification";
val p1 = premise ⌜p ∧ q⌝;
conjunctive_simplification p1;
```

```
:) "Conjunctive Simplification";
val p1 = premise ⌜p ∧ q⌝;
conjunctive_simplification p1;
val it = "Conjunctive Simplification": string
:) val p1 = p ∧ q ⊢ p ∧ q: THM
:) val it = p ∧ q ⊢ p: THM
```

cs113 Lab4: Rules of Inference: Disjunctive Syllogism

- Disjunctive Syllogism

$$\begin{array}{l} p \vee q \\ \sim q \\ \hline \therefore p \end{array}$$

SML

```
"Disjunctive Syllogism";
val p1 = premise ⌜p∨q⌝;
val p2 = premise ⌜¬q:BOOL⌝;
disjunctive_syllogism p1 p2;
```

```
:) "Disjunctive Syllogism";
val p1 = premise ⌜p∨q⌝;
val p2 = premise ⌜¬q:BOOL⌝;
disjunctive_syllogism p1 p2;
val it = "Disjunctive Syllogism": string
:) val p1 = p ∨ q ⊢ p ∨ q: THM
:) val p2 = ¬ q ⊢ ¬ q: THM
:) val it = p ∨ q, ¬ q ⊢ p: THM
```

- Hypothetical Syllogism

$$\begin{array}{l} p \to q \\ q \to r \\ \hline \therefore p \to r \end{array}$$

SML

```
"Hypothetical Syllogism";
val p1 = premise ⌜p⇒q⌝;
val p2 = premise ⌜q⇒r⌝;
hypothetical_syllogism p1 p2;
```

```
:) "Hypothetical Syllogism";
val p1 = premise ⌜p⇒q⌝;
val p2 = premise ⌜q⇒r⌝;
hypothetical_syllogism p1 p2;
val it = "Hypothetical Syllogism": string
:) val p1 = p ⇒ q ⊢ p ⇒ q: THM
:) val p2 = q ⇒ r ⊢ q ⇒ r: THM
:) val it = p ⇒ q, q ⇒ r ⊢ p ⇒ r: THM
```

cs113 Lab4: Rules of Inference: Double Negative

- Double Negative

$$\frac{\sim\sim p}{\therefore p}$$

SML

| "*Double Negative*";
| *val p1 = premise* $\ulcorner \neg\neg p \urcorner$;
| *double_negative p1*;

```
:) "Double Negative";
val p1 = premise ⌜¬¬p⌝;
double_negative p1;
val it = "Double Negative": string
:) val p1 = ¬ ¬ p ⊢ ¬ ¬ p: THM
:) val it = ¬ ¬ p ⊢ p: THM
```

- Demorgan's

$$\frac{\sim (p \wedge q)}{\therefore \sim p \vee \sim q}$$

SML

$$"Demorgan's";$$
$$val\ p1 = premise\ \ulcorner\neg(p\wedge q)\urcorner;$$
$$demorgan\_conjunction\ p1;$$

```
:) "Demorgan's";
val p1 = premise ⌜¬(p∧q)⌝;
demorgan_conjunction p1;
val it = "Demorgan's": string
:) val p1 = ¬ (p ∧ q) ⊢ ¬ (p ∧ q): THM
:) val it = ¬ (p ∧ q) ⊢ ¬ p ∨ ¬ q: THM
```

cs113 Lab4: Problem 4.16

- Problem 4.16

$\sim p \vee q \rightarrow r$
$s \vee \sim q$
$\sim t$
$p \rightarrow t$
$\sim p \wedge r \rightarrow \sim s$

$\therefore \sim q$

```
val it = "Problem 4.16": string
:) :# val f = ⌜q⌝: TERM
val p1 = ¬ p ∨ q ⇒ r ⊢ ¬ p ∨ q ⇒ r: THM
:) val p2 = s ∨ ¬ q ⊢ s ∨ ¬ q: THM
:) val p3 = ¬ t ⊢ ¬ t: THM
:) val p4 = p ⇒ t ⊢ p ⇒ t: THM
:) val p5 = ¬ p ∧ r ⇒ ¬ s ⊢ ¬ p ∧ r ⇒ ¬ s: THM
:) val r1 = p ⇒ t, ¬ t ⊢ ¬ p: THM
:) val r2 = p ⇒ t, ¬ t ⊢ ¬ p ∨ q: THM
:) val r3 = ¬ p ∨ q ⇒ r, p ⇒ t, ¬ t ⊢ r: THM
:) val r4 = ¬ p ∨ q ⇒ r, p ⇒ t, ¬ t ⊢ ¬ p ∧ r: THM
:) val r5 = ¬ p ∧ r ⇒ ¬ s, ¬ p ∨ q ⇒ r, p ⇒ t, ¬ t ⊢ ¬ s: THM
:) val r6 = s ∨ ¬ q, ¬ p ∧ r ⇒ ¬ s, ¬ p ∨ q ⇒ r, p ⇒ t, ¬ t ⊢ ¬ q: THM
```

SML

```
"Problem 4.16";
val f = ⌜q:BOOL⌝
val p1 = premise ⌜¬p ∨ q⇒r⌝;
val p2 = premise ⌜s ∨ ¬q ⌝;
val p3 = premise ⌜¬t:BOOL⌝;
val p4 = premise ⌜p ⇒ t⌝;
val p5 = premise ⌜¬p ∧ r⇒ ¬s⌝;
val r1 = modus_tollens p4 p3;
val r2 = disjunctive_addition f r1;
val r3 = modus_ponens p1 r2;
val r4 = conjunctive_addition r1 r3;
val r5 = modus_ponens p5 r4;
val r6 = disjunctive_syllogism p2 r5;
```

- Problem 4.18

$$
\begin{array}{l}
\sim (p \vee q) \rightarrow r \\
\sim (p) \\
\sim (r) \\
\hline
\therefore q
\end{array}
$$

SML

```
"Problem 4.18";
val p1 = premise ⌜¬(p∨q)⇒r⌝;
val p2 = premise ⌜¬p:BOOL⌝;
val p3 = premise ⌜¬r:BOOL⌝;
val r1 = modus_tollens p1 p3;
disjunctive_syllogism (double_negation r1) p2;
```

```
val it = "Problem 4.18": string
:) val p1 = ¬ (p ∨ q) ⇒ r ⊢ ¬ (p ∨ q) ⇒ r: THM
:) val p2 = ¬ p ⊢ ¬ p: THM
:) val p3 = ¬ r ⊢ ¬ r: THM
:) val r1 = ¬ (p ∨ q) ⇒ r, ¬ r ⊢ ¬ ¬ (p ∨ q): THM
:) val it = ¬ (p ∨ q) ⇒ r, ¬ r, ¬ p ⊢ q: THM
```

- Problem 4.19

$p \wedge q$
$p \rightarrow \sim (q \wedge r)$
$s \rightarrow r$

$\therefore \sim s$

```
val it = "Problem 4.19": string
:) val p1 = p ∧ q ⊢ p ∧ q: THM
:) val p2 = p ⇒ ¬ (q ∧ r) ⊢ p ⇒ ¬ (q ∧ r): THM
:) val p3 = s ⇒ r ⊢ s ⇒ r: THM
:) val r1 = p ∧ q ⊢ p: THM
:) val r2 = p ⇒ ¬ (q ∧ r), p ∧ q ⊢ ¬ (q ∧ r): THM
:) val r3 = p ⇒ ¬ (q ∧ r), p ∧ q ⊢ ¬ q ∨ ¬ r: THM
:) val r4 = p ∧ q ⊢ q: THM
:) val r5 = p ∧ q ⊢ ¬ ¬ q: THM
:) val r6 = p ⇒ ¬ (q ∧ r), p ∧ q ⊢ ¬ r: THM
:) val r7 = s ⇒ r, p ⇒ ¬ (q ∧ r), p ∧ q ⊢ ¬ s: THM
```

SML

```
"Problem 4.19";
val p1 = premise ⌜p ∧ q⌝;
val p2 = premise ⌜p ⇒ ¬(q ∧ r)⌝;
val p3 = premise ⌜s ⇒ r⌝;
val r1 = conjunctive_simplification p1;
val r2 = modus_ponens p2 r1;
val r3 = demorgan_conjunction r2;
val r4 = conjunctive_simplificationR p1;
val r5 = double_negativeI r4;
val r6 = disjunctive_syllogism r3 r5;
val r7 = modus_tollens p3 r6;
```

cs113 Lab4: Predicate Calculus - Problem 5.8 a

Problem 5.8 a

Let D = {-48, -14,-8,0,1,3,16,23,26,32,36} Determine if this proposition is valid:

$\forall x \in D$, if x is odd then $x > 0$

SML

```
"Problem 5.8 a";
fun isOdd(x) = x mod 2 = 1;
val D = [~48, ~14,~8,0,1,3,16,23,26,32,36];
fun isGreater0 x = x > 0;
fun tautology (x::xs) = x andalso tautology(xs)
 | tautology ([]) = true;
tautology (map isGreater0 (filter isOdd D));
```

```
:) "Problem 5.8 a";
fun isOdd(x) = x mod 2 = 1;
val D = [~48, ~14,~8,0,1,3,16,23,26,32,36];
fun isGreater0 x = x > 0;
fun tautology (x::xs) = x andalso tautology(xs)
 | tautology ([]) = true;
tautology (map isGreater0 (filter isOdd D));
val it = "Problem 5.8 a": string
:) val isOdd = fn: int -> bool
:) val D = [~48, ~14, ~8, 0, 1, 3, 16, 23, 26, 32, 36]: int list
:) val isGreater0 = fn: int -> bool
:) :# val tautology = fn: bool list -> bool
:) val it = true: bool
```

- It is true

cs113 Lab4: Predicate Calculus - Problem 5.8 b

Problem 5.8 b

Let D = {-48, -14,-8,0,1,3,16,23,26,32,36} Determine if this proposition is valid:

$\forall x \in D$, if x is less than zero then x is even

SML

```
"Problem 5.8 b";
fun isEven(x) = x mod 2 = 0;
val D = [~48, ~14,~8,0,1,3,16,23,26,32,36];
fun isLesser0 x = x < 0;
fun tautology (x::xs) = x andalso tautology(xs)
 | tautology ([]) = true;
tautology (map isEven (filter isLesser0 D));
```

```
:) "Problem 5.8 b";
fun isEven(x) = x mod 2 = 0;
val D = [~48, ~14,~8,0,1,3,16,23,26,32,36];
fun isLesser0 x = x < 0;
fun tautology (x::xs) = x andalso tautology(xs)
 | tautology ([]) = true;
tautology (map isEven (filter isLesser0 D));
val it = "Problem 5.8 b": string
:) val isEven = fn: int -> bool
:) val D = [~48, ~14, ~8, 0, 1, 3, 16, 23, 26, 32, 36]: int list
:) val isLesser0 = fn: int -> bool
:) :# val tautology = fn: bool list -> bool
:) val it = true: bool
```

- It is true

cs113 Lab4: Predicate Calculus - Problem 5.8 c

Problem 5.8 c

Let D = {-48, -14,-8,0,1,3,16,23,26,32,36} Determine if this proposition is valid:

$\forall x \in D$, if x is even then $x <= 0$

SML

```
"Problem 5.8 c";
fun isEven(x) = x mod 2 = 0;
val D = [~48, ~14,~8,0,1,3,16,23,26,32,36];
fun isLesserAnd0 x = x <= 0;
fun tautology (x::xs) = x andalso tautology(xs)
 | tautology ([]) = true;
tautology (map isLesserAnd0 (filter isEven D));
```

```
:) "Problem 5.8 c";
fun isEven(x) = x mod 2 = 0;
val D = [~48, ~14,~8,0,1,3,16,23,26,32,36];
fun isLesserAnd0 x = x <= 0;
fun tautology (x::xs) = x andalso tautology(xs)
 | tautology ([]) = true;
tautology (map isLesserAnd0 (filter isEven D));
val it = "Problem 5.8 c": string
:) val isEven = fn: int -> bool
:) val D = [~48, ~14, ~8, 0, 1, 3, 16, 23, 26, 32, 36]: int list
:) val isLesserAnd0 = fn: int -> bool
:) :# val tautology = fn: bool list -> bool
:) val it = false: bool
```

- It is false because the value 16, 32, and 36 are all counterexamples

cs113 Lab4: Predicate Calculus - Problem 5.8 d

Problem 5.8 d
Let D = {-48, -14,-8,0,1,3,16,23,26,32,36} Determine if this proposition is valid:
if the ones digit of x is 2, then the tens digit is 3 or 4.

SML

```
"Problem 5.8 d";
fun ones_digit_is_2(x) = abs(x) mod 10 = 2;
val D = [~48, ~14,~8,0,1,3,16,23,26,32,36];
fun tens_digit_is_3_or_4(x) = abs(x) div 10 = 3
 orelse abs(x) div 10 = 4;
fun tautology (x::xs) = x andalso tautology(xs)
 | tautology ([]) = true;
tautology (map tens_digit_is_3_or_4
(filter ones_digit_is_2 D));
```

```
:) "Problem 5.8 d";
fun ones_digit_is_2(x) = abs(x) mod 10 = 2;
val D = [~48, ~14,~8,0,1,3,16,23,26,32,36];
fun tens_digit_is_3_or_4(x) = abs(x) div 10 = 3
 orelse abs(x) div 10 = 4;
fun tautology (x::xs) = x andalso tautology(xs)
 | tautology ([]) = true;
tautology (map tens_digit_is_3_or_4
(filter ones_digit_is_2 D));
val it = "Problem 5.8 d": string
:) val ones_digit_is_2 = fn: int -> bool
:) val D = [~48, ~14, ~8, 0, 1, 3, 16, 23, 26, 32, 36]: int list
:) :# val tens_digit_is_3_or_4 = fn: int -> bool
:) :# val tautology = fn: bool list -> bool
:) :# val it = true: bool
```

- It is true

Problem 5.8 e

Let D = {-48, -14,-8,0,1,3,16,23,26,32,36} Determine if this proposition is valid:
if the ones digit of x is 6, then the tens digit is 1 or 2.

SML

```
"Problem 5.8 e";
fun ones_digit_is_6(x) = abs(x) mod 10 = 6;
val D = [~48, ~14,~8,0,1,3,16,23,26,32,36];
fun tens_digit_is_1_or_2(x) = abs(x) div 10 = 1
 orelse abs(x) div 10 = 2;
fun tautology (x::xs) = x andalso tautology(xs)
 | tautology ([]) = true;
tautology (map tens_digit_is_1_or_2
(filter ones_digit_is_6 D));
```

```
:) "Problem 5.8 e";
fun ones_digit_is_6(x) = abs(x) mod 10 = 6;
val D = [~48, ~14,~8,0,1,3,16,23,26,32,36];
fun tens_digit_is_1_or_2(x) = abs(x) div 10 = 1
 orelse abs(x) div 10 = 2;
fun tautology (x::xs) = x andalso tautology(xs)
 | tautology ([]) = true;
tautology (map tens_digit_is_1_or_2
(filter ones_digit_is_6 D));
val it = "Problem 5.8 e": string
:) val ones_digit_is_6 = fn: int -> bool
:) val D = [~48, ~14, ~8, 0, 1, 3, 16, 23, 26, 32, 36]: int list
:) :# val tens_digit_is_1_or_2 = fn: int -> bool
:) :# val tautology = fn: bool list -> bool
:) :# val it = false: bool
```

- It is false because the value 36 is a counterexample