Overview

Applying Formal Methods: Part I

Overview

- Access-control properties
  - Semantics: Kripke model
  - Model-checking problem
    - Given a Kripke model and an access-control property, for which world in the model is the property true?

Overview (cont.)

- Model-checking problem (cont.)
  - The key to solving the model-checking problem is to implement the evaluation function, where creating a Set data type is the beginning step
- Reasoning: inference rules, formal proofs, and proof trees

The End

Kripke Models

Access-Control Properties and Kripke Models

Informal Meanings

- Propositional variables can be used to represent requests:
  rff might represent "It is good to allow read access to file foo".
- P says φ ("Principal P makes the statement φ")
  P may explicitly utter φ, or it's safe to act as if P uttered φ.

  Fran says *rff*

  Sam says (*Fran* says *rff*)

- P controls φ ("Principal P has jurisdiction over statement φ", or "Principal P is authorized to make statement φ")
  If P says φ, then we consider φ to be true.

  Fran controls *rff*

  Sam controls (*Fran* controls *rff*)

- $P \Rightarrow Q$ ("P speaks for Q")
  Any statements P makes can be safely ascribed to Q (i.e., acted upon as if Q had made it).

Towards More Precision

Informal meanings are helpful, but...

- Which formulas are true in all cases (i.e., tautologies)?
- Which formulas are never true (i.e., contradictions)?
- Which formulas logically follow from others?
- How can we reason precisely about these statements?
- How can we trust that our conclusions make sense?

To answer these questions, the logic needs formal semantics:

- Akin to truth tables for propositional logic
- Based on Kripke structures

Kripke Structures

A *Kripke structure* $\mathcal{M}$ is a three-tuple $\langle W, I, J \rangle$, where:

- W is a nonempty set, whose elements are called *worlds*.
- I : **PropVar** $\to \mathcal{P}(W)$ is an *interpretation* function that maps each propositional variable p to a set of worlds.
- J : **PName** $\to \mathcal{P}(W \times W)$ is a function that maps each principal name A into a relation on worlds (i.e., a subset of $W \times W$).

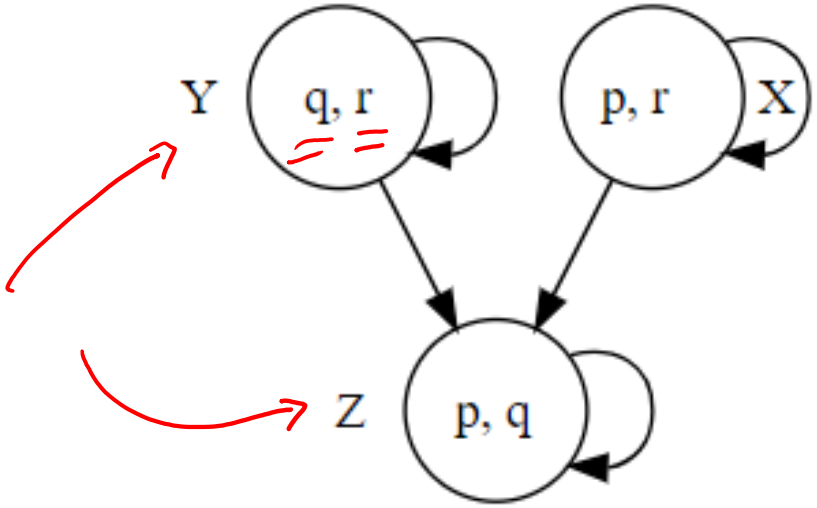
Intuition: "Truth tables" for modal logic!

- I is the Kripke-equivalent of a truth assignment: $I(p)$ is set of worlds in which p is true.
- $J(A)$ is relation describing how A views various worlds: each pair $(w, w') \in J(A)$ indicates that, when the current world is w, principal A believes current world might be w'.

Labeled Relations as Graphs

- W = {X, Y, Z}
- Relation *R* ($\subseteq$ W x W)
- *R* = { (X,X), (X,Z), (Y,Y), (Y,Z),(Z,Z)}
- *I* (p) = {X,Z}, *I* (q) = {Y,Z} and *I* (r) = {X,Y}

- A labeled relation diagram for *R* with proposition variables p, q, and r labeled according to *I*



Kripke Structures: Simple Example

$\mathcal{M}_0 = \langle W_0, I_0, J_0 \rangle$ is a Kripke structure, where:

- $W_0 = \{sw, sc, ns\}$
- $I_0 : \textbf{PropVar} \to \mathcal{P}(W_0)$, such that

$$I_0(go) \quad = \quad \{sw\}$$

- $J_0 : \textbf{PName} \to \mathcal{P}(W_0 \times W_0)$, such that

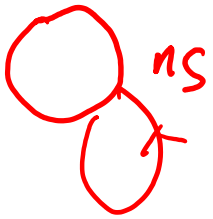
$$\begin{aligned} J_0(Gil) &= \{(sw, sw), (sc, sc), (ns, ns)\} \\ J_0(Flo) &= \{(sw, sw), (sw, sc), (sc, sw), (sc, sc), (ns, ns)\} \\ J_0(Hal) &= \{(sw, sw), (sc, sw), (ns, ns)\} \end{aligned}$$

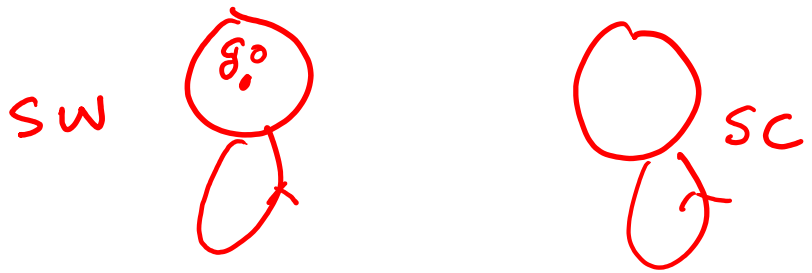
- *Gil* has perfect knowledge about the worlds.
- *Flo* has uncertainty in worlds *sw* and *sc*.
- *Hal* has a mistaken belief in world *sc*.

A Diagram for the Kripke Structure M_0

3 principals: Gil, Flo, Hal

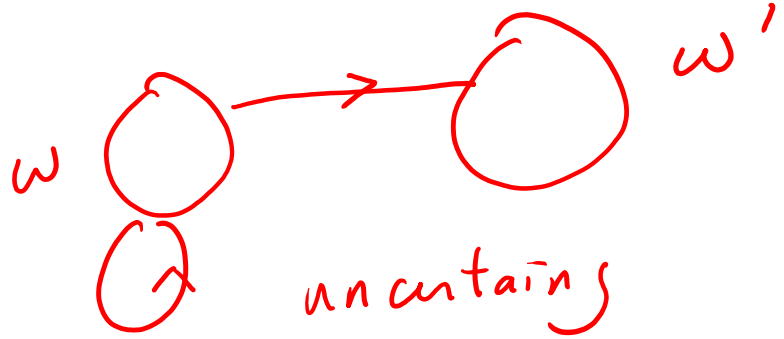
Draw $J(Gil)$

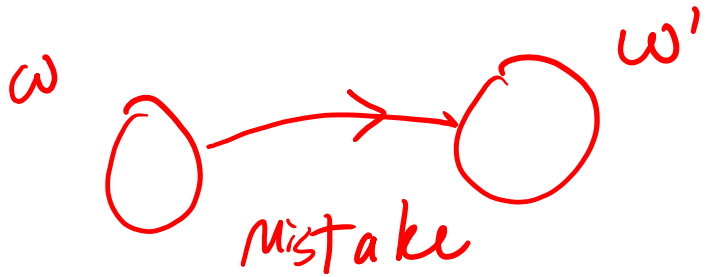




$(w, w') \quad w' = w$

$\in J(Gil)$





Kripke Structures: Another Example

$\mathcal{M}_1 = \langle W_1, I_1, J_1 \rangle$ is a Kripke structure, where:

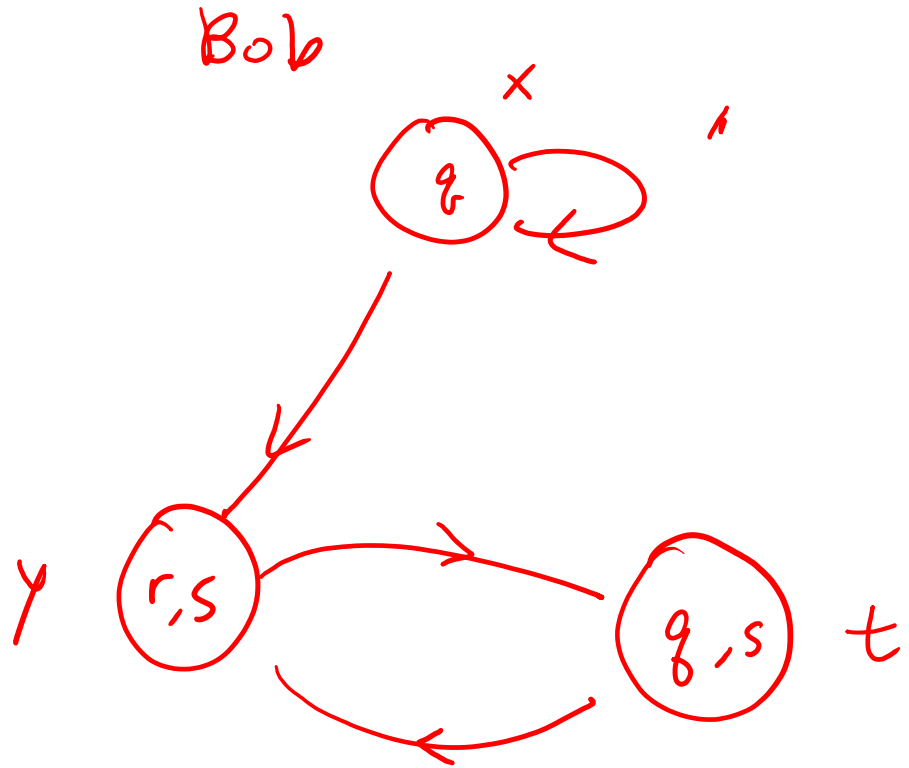
- $W_1 = \{x, y, t\}$
- I_1 : **PropVar** $\rightarrow \mathcal{P}(W_1)$, such that

$$\begin{aligned} I_1(q) &= \{x, t\} \\ I_1(r) &= \{y\} \\ I_1(s) &= \{y, t\} \end{aligned}$$

- J_1 : **PName** $\rightarrow \mathcal{P}(W_1 \times W_1)$, such that

$$\begin{aligned} J_1(Alice) &= \{(x,x),(y,y),(t,t)\} \\ J_1(Bob) &= \{(x,x),(x,y),(y,t),(t,y)\} \end{aligned}$$

A Diagram for the Kripke Structure M_1



Bob says φ

A.C property.

$E_M[\varphi]$

Go to use an evaluation to determine the set of world. -ing

the a property φ is true

The End

The Model-Checking Problem

Kripke Semantics

Kripke Semantics

Each Kripke structure $\mathcal{M} = \langle W, I, J \rangle$ gives rise to a function

$$\mathcal{E}_{\mathcal{M}}[\![-]\!] : \textbf{Form} \to \mathcal{P}(W),$$

where $\mathcal{E}_{\mathcal{M}}[\![\varphi]\!]$ is the set of worlds in which φ is true.

- $\mathcal{E}_{\mathcal{M}}[\![-]\!]$ is the Kripke-equivalent of rules for truth tables.
- $\mathcal{E}_{\mathcal{M}}[\![\varphi]\!]$ defined based on the structure of φ, plus the individual components of $\mathcal{M} = \langle W, I, J \rangle$.
- $\mathcal{M}$ satisfies φ provided that $\mathcal{E}_{\mathcal{M}}[\![\varphi]\!] = W$.
- φ is a tautology provided that every Kripke structure satisfies φ.

define meaning for P.C formula for the model $\mathcal{M}$

$M \models \varphi$

M models φ

meaning : $E_M(\varphi) = W$

↑ all possible world.

Example ("φ is tautology")

$\models \varphi$

φ is true for all Kripke Models.

The End

The Model-Checking Problem

The Evaluation Function: Definition and Calculations

Section 2.3.

Semantics: Definition of $\mathcal{E}_{\mathcal{M}}[\![-]\!]$

Form

$$
\begin{aligned}
\mathcal{E}_{\mathcal{M}}[\![p]\!] &= I(p) \\
\mathcal{E}_{\mathcal{M}}[\![\neg\varphi]\!] &= W - \mathcal{E}_{\mathcal{M}}[\![\varphi]\!] \\
\mathcal{E}_{\mathcal{M}}[\![\varphi_1 \wedge \varphi_2]\!] &= \mathcal{E}_{\mathcal{M}}[\![\varphi_1]\!] \cap \mathcal{E}_{\mathcal{M}}[\![\varphi_2]\!] \\
\mathcal{E}_{\mathcal{M}}[\![\varphi_1 \vee \varphi_2]\!] &= \mathcal{E}_{\mathcal{M}}[\![\varphi_1]\!] \cup \mathcal{E}_{\mathcal{M}}[\![\varphi_2]\!] \\
\mathcal{E}_{\mathcal{M}}[\![\varphi_1 \supset \varphi_2]\!] &= (W - \mathcal{E}_{\mathcal{M}}[\![\varphi_1]\!]) \cup \mathcal{E}_{\mathcal{M}}[\![\varphi_2]\!] \\
\mathcal{E}_{\mathcal{M}}[\![\varphi_1 \equiv \varphi_2]\!] &= \mathcal{E}_{\mathcal{M}}[\![\varphi_1 \supset \varphi_2]\!] \cap \mathcal{E}_{\mathcal{M}}[\![\varphi_2 \supset \varphi_1]\!]
\end{aligned}
$$

Similar to the rules for prop. logic formula

part I

$$
\begin{aligned}
\mathcal{E}_{\mathcal{M}}[\![P \text{ says } \varphi]\!] &= \{w | J(P)(w) \subseteq \mathcal{E}_{\mathcal{M}}[\![\varphi]\!]\} \\
\mathcal{E}_{\mathcal{M}}[\![P \text{ controls } \varphi]\!] &= \mathcal{E}_{\mathcal{M}}[\![(P \text{ says } \varphi) \supset \varphi]\!] \\
\mathcal{E}_{\mathcal{M}}[\![P \Rightarrow Q]\!] &= \begin{cases} W, & \text{if } J(Q) \subseteq J(P) \\ \emptyset, & \text{otherwise} \end{cases}
\end{aligned}
$$

part II

Match the intended meaning

Compound Principals

Example 2.11 page 27

Two operators combine principal expressions:

- $P \,\&\, Q$: informally denotes P **with** Q
- $P \mid Q$: informally denotes P **quoting** Q

Semantics:

$$J(P \,\&\, Q) = J(P) \cup J(Q)$$
$$J(P \mid Q) = J(P) \circ J(Q)$$

Semantics: Example 1

Recall $\mathcal{M}_1 = \langle W_1, I_1, J_1 \rangle$, where $W_1 = \{x, y, t\}$ and:

$$I_1(q) = \{x, t\}, \quad I_1(r) = \{y\}, \quad I_1(s) = \{y, t\}$$

In what worlds is $q \supset (r \wedge s)$ true?

$$\begin{aligned}
\mathcal{E}_{\mathcal{M}_1}[\![q \supset (r \wedge s)]\!] &= (W_1 - \mathcal{E}_{\mathcal{M}_1}[\![q]\!]) \cup \mathcal{E}_{\mathcal{M}_1}[\![r \wedge s]\!] \\
&= (W_1 - \mathcal{E}_{\mathcal{M}_1}[\![q]\!]) \cup (\mathcal{E}_{\mathcal{M}_1}[\![r]\!] \cap \mathcal{E}_{\mathcal{M}_1}[\![s]\!]) \\
&= (W_1 - I_1(q)) \cup (I_1(r) \cap I_1(s)) \\
&= (W_1 - \{x, t\}) \cup (\{y\} \cap \{y, t\}) \\
&= \{y\} \cup \{y\} \\
&= \{y\}
\end{aligned}$$

recursive definition

Semantics: Example 2

Recall $\mathcal{M}_1 = \langle W_1, I_1, J_1 \rangle$, where $W_1 = \{x, y, t\}$ and:

$$\begin{aligned} J_1(Alice) &= \{(x,x),(y,y),(t,t)\} \\ J_1(Bob) &= \{(x,x),(x,y),(y,x),(t,y)\} \end{aligned}$$

] world view

Therefore, we know that:

$$\begin{aligned} J_1(Alice)(x) &= \{x\} & J_1(Bob)(x) &= \{x,y\} \\ J_1(Alice)(y) &= \{y\} & J_1(Bob)(y) &= \{x\} \\ J_1(Alice)(t) &= \{t\} & J_1(Bob)(t) &= \{y\} \end{aligned}$$

↓
J (principal ↑)

In what worlds are the following true?

$Alice$ says $(q \supset (r \wedge s))$ $\quad$ Bob says $(q \supset (r \wedge s))$

Semantics: Example 2 (continued)

Recall that $\mathcal{E}_{\mathcal{M}_1}[\![q \supset (r \wedge s)]\!] = \{y\}$:

previous result

$$\begin{aligned}
&\mathcal{E}_{\mathcal{M}_1}[\![\mathit{Alice}\ \text{says}\ (q \supset (r \wedge s))]\!] \\
&\qquad = \{w \mid J_1(\mathit{Alice})(w) \subseteq \mathcal{E}_{\mathcal{M}_1}[\![q \supset (r \wedge s)]\!]\} \\
&\qquad = \{w \mid J_1(\mathit{Alice})(w) \subseteq \{y\}\} \\
&\qquad = \{y\}
\end{aligned}$$

$J_1(\mathit{Alice})$ computed earlier

$$\begin{aligned}
&\mathcal{E}_{\mathcal{M}_1}[\![\mathit{Bob}\ \text{says}\ (q \supset (r \wedge s))]\!] \\
&\qquad = \{w \mid J_1(\mathit{Bob})(w) \subseteq \mathcal{E}_{\mathcal{M}_1}[\![q \supset (r \wedge s)]\!]\} \\
&\qquad = \{w \mid J_1(\mathit{Bob})(w) \subseteq \{y\}\} \\
&\qquad = \{t\}
\end{aligned}$$

$J_1(\mathit{Bob})$

$\mathcal{E}_M(\varphi) = \{w : \varphi \text{ is true w.r.t to } M\}$

model M

ACL formula

The End

The Set Data Type I

The Set Data Type

A Data Type for Sets

- Sets are collections of elements. We consider those that:
  - Are finite
  - Contain elements of the same type
- Remarks
  - Sets and lists are similar, but not are the same
  - Sets are not necessarily ordered, and multiple occurrences of the same element are ignored

The Set Data Type

Representation

- Type newtype Set a = Set [a].
- The newtype construction has the same effect as using the data mechanism with one constructor (i.e., Set).
- To implement the usual set operations effectively, the type a should be a member of the Eq class.
- We may implement sets that ordered sets (i.e., a member of Ord if desired.

Set Operations

The Set data type is expected to support usual set operations; it will include:

- Set union (∪), intersection (∩), difference (–), and cardinality operations; we may also implement operations to compute symmetric difference and power sets
- The membership (ϵ), subset (⊆), and equal (=) relations
- Operation(s) to display (show) the elements of a set

Design Considerations

- Implement as an ordered set (or not)?
- How do you handle repetitions in the underlying list?
- Are there any additional operations/functions (e.g., higher-order functions) that will enhance set manipulations?
- Can we make ~~sure~~ use of functions from the Prelude?

Initial Steps

- Represent a set as an ordered list of elements without repetitions and declare Set as the data type

  ↪ newtype Set a = Set [a]

  where the underlying elements are stored in a list.
- Specify the types of each operations/functions to be implemented. e.g. ∪, ∩, − (set difference), Cardinality, etc..
- Identify functions from the Prelude for processing lists that may help. elem x lst ...

The End

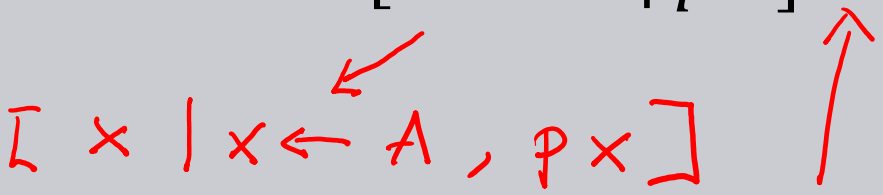
The Set Data Type II

More on the Set Data Type

List Comprehensions

List comprehensions allow many functions on lists to be defined by translating from the definition of a set.

Define a Set S via set-builder notation	Define a List L that represent S via list comprehension
$S = \{x \in A \mid p\ x\}$	$L = [\,x \leftarrow A \mid p\ x]$



Practice

Define the intersection of two sets formally in set notation and translate it into Haskell code.

A, B are sets

$$A \cap B = \{ x \mid (x \in A) \wedge (x \in B) \}$$

```
inter A B = [ x | x <- xs, elem x ys]
```

xs ys

The End

Reasoning and Access Control: Introduction

Inference Rules

How Do We Use the Logic?

No one wants to do analysis using Kripke structures:

- They're cumbersome to use.
- It's not clear which structure to use in a given situation.
- **However**, they provide **precise** meanings for our formulas.

Thus, to support a **feasible and legitimate** analysis, we need:

1. A set of rules that lets us manipulate formulas in our logic as a way of calculating the consequences of certain assumptions:

$$\frac{H_1 \quad \cdots \quad H_k}{C}$$

reasoning behind it.

IF $H_1 \ldots H_k$ are all true

Then the conclusion is true.

2. Confidence that the rules "make sense" and reflect the underlying semantics: any Kripke structure that satisfies $H_1, \ldots, H_k$ should also satisfy C.

Logical Rules: Page 1

$$\textit{Taut}\ \frac{}{\varphi}$$

if φ is an instance of a prop-logic tautology

$$\textit{Modus Ponens}\ \frac{\varphi \qquad \varphi \supset \varphi'}{\varphi'} \qquad \textit{Says}\ \frac{\varphi}{P\ \text{says}\ \varphi}$$

$$\textit{MP Says}\ \frac{}{(P\ \text{says}\ (\varphi \supset \varphi')) \supset (P\ \text{says}\ \varphi \supset P\ \text{says}\ \varphi')}$$

$$\textit{Speaks For}\ \frac{}{P \Rightarrow Q \supset (P\ \text{says}\ \varphi \supset Q\ \text{says}\ \varphi)}$$

$$P\ \text{controls}\ \varphi \stackrel{\text{def}}{=} (P\ \text{says}\ \varphi) \supset \varphi$$

Common Propositional-Logic Tautologies

FIGURE 3.2 Common propositional-logic tautologies

$p \vee \neg p$

$p \equiv (\neg\neg p)$

$p \supset (q \vee p)$

$p \supset (q \supset p)$

$(p \wedge q) \supset p$

$\neg(\neg p \wedge p)$

$p \supset (q \supset (p \wedge q))$

$(p \wedge q) \supset (p \supset q)$

$(p \wedge q) \supset (q \wedge p)$

$(p \equiv q) \supset (p \supset q)$

$((p \vee q) \wedge \neg p) \supset q$

$((p \supset q) \wedge (q \supset r)) \supset (p \supset r)$

Using the Taut Rule

- $(p \vee \neg p)$ is a tautology
- The taut rule can infer

$$(Paul \text{ controls } (read \wedge write)) \vee \neg(Paul \text{ controls } (read \wedge write))$$

- When applying the taut rule(s), we need to identify suitable tautologies

Logical Rules: Page 2

$$\textit{Transitivity of } \Rightarrow \quad \frac{P \Rightarrow Q \qquad Q \Rightarrow R}{P \Rightarrow R}$$

$$\textit{Equivalence} \quad \frac{\varphi_1 \equiv \varphi_2 \qquad \psi[\varphi_1/q]}{\psi[\varphi_2/q]}$$

Meaning of the notation

$\psi[\varphi/q]$ represents the result of replacing every occurrence of the variable q in ψ by the formula φ

Example:

$$(((R \text{ says } (t \supset r)) \wedge s) \supset (T \mid R \text{ controls } s))[t \supset r/s]$$

is

$$((R \text{ says } (t \supset r)) \wedge (t \supset r)) \supset (T \mid R \text{ controls } (t \supset r))$$

1st → 2nd: replace s (in 1st) by $(t \supset r)$ → get 2nd.

Proofs Using the Rules

A **formal proof** is a sequence of annotated statements in the logic, where each statement is one of the following:

- An assumption (annotated by "Assumption")
- Obtained by applying one of the axioms, definitions, or logical rules to previous statements in that sequence (annotated with appropriate rule)

Every formal proof yields a **theorem (derived inference rule)**

$$\frac{H_1 \quad \cdots \quad H_k}{C},$$

where $H_1, \ldots H_k$ are the only assumptions of the proof and C is the final statement.

The End

Reasoning and Access Control: Introduction

Formal Proofs via Inference Rules

Proofs Using the Rules

A **formal proof** is a sequence of annotated statements in the logic, where each statement is one of the following:

- An assumption (annotated by "Assumption")
- Obtained by applying one of the axioms, definitions, or logical rules to previous statements in that sequence (annotated with appropriate rule)

Every formal proof yields a **theorem (derived inference rule)**

$$\frac{H_1 \quad \cdots \quad H_k}{C},$$

where $H_1, \ldots H_k$ are the only assumptions of the proof and C is the final statement.

H_1

H_2

H_3

C

1 H_1 (assumption)

2. H_2 (assumption)

3. H_3 (assumption)

4 .
.
.
.

(n^{th} C) (May use 1, 2, 3 ... and existing Inference Rules)

(- -)

← Reasoning →

Figure 3.1 (Chapter)

Explanatn on Why those rules

are accepted

is discussed in Chapter 3. Section 1.

Logical Rules: Page 1

$$\textit{Taut}\quad \frac{}{\varphi}$$

if φ is an instance of a prop-logic tautology

$$\textit{Modus Ponens}\quad \frac{\varphi \qquad \varphi \supset \varphi'}{\varphi'} \qquad\qquad \textit{Says}\quad \frac{\varphi}{P \text{ says } \varphi}$$

$$\textit{MP Says}\quad \frac{}{(P \text{ says } (\varphi \supset \varphi')) \supset (P \text{ says } \varphi \supset P \text{ says } \varphi')}$$

$$\textit{Speaks For}\quad \frac{}{P \Rightarrow Q \supset (P \text{ says } \varphi \supset Q \text{ says } \varphi)}$$

$$P \text{ controls } \varphi \quad \stackrel{\text{def}}{=} \quad (P \text{ says } \varphi) \supset \varphi$$

Sample Formal Proof

Let's deduce A says s from the following two assumptions:

$$A \text{ says } (r \supset s) \qquad r$$

Theorem

1 2

3

A formal proof:

Assumptions

1.	A says $(r \supset s)$	Assumption
2.	r	Assumption
3.	$(A \text{ says } (r \supset s)) \supset (A \text{ says } r \supset A \text{ says } s)$	MP Says
4.	A says $r \supset A$ says s	1,3 Modus Ponens
5.	A says r	2 Says
6.	A says s	4,5 Modus Ponens

Conclusion

The resulting derived rule: $\dfrac{A \text{ says } (r \supset s) \qquad r}{A \text{ says } s}$

More Sample Proofs

Proofs for derived rules:

$$\frac{P \text{ controls } \varphi \quad P \text{ says } \varphi}{\varphi} \qquad \frac{\varphi_1 \quad \varphi_2}{\varphi_1 \wedge \varphi_2}$$

Proof of (A)

1. P controls φ	Assumption
2. P says φ	Assumption
3. $(P \text{ says } \varphi) \supset \varphi$	1 Def^n controls
4. φ	2,3 Modus Ponens

Proof of (B)

1. φ_1	Assumption
2. φ_2	Assumption
3. $\varphi_1 \supset (\varphi_2 \supset (\varphi_1 \wedge \varphi_2))$	Taut
4. $\varphi_2 \supset (\varphi_1 \wedge \varphi_2)$	1,3 Modus Ponens
5. $\varphi_1 \wedge \varphi_2$	2,4 Modus Ponens

The End

Reasoning and Access Control: Proofs

Derived Inference Rules

Some Useful Derived Rules

Conjunction $\dfrac{\varphi_1 \qquad \varphi_2}{\varphi_1 \wedge \varphi_2}$ *Double negation* $\dfrac{\neg\neg\varphi}{\varphi}$

Simplification $\dfrac{\varphi_1 \wedge \varphi_2}{\varphi_1}$ *Simplification* $\dfrac{\varphi_1 \wedge \varphi_2}{\varphi_2}$

Disjunction $\dfrac{\varphi_1}{\varphi_1 \vee \varphi_2}$ *Disjunction* $\dfrac{\varphi_2}{\varphi_1 \vee \varphi_2}$

Modus Tollens $\dfrac{\varphi \supset \varphi' \qquad \neg\varphi'}{\neg\varphi}$ *Hypothetical Syllogism* $\dfrac{\varphi_1 \supset \varphi_2 \qquad \varphi_2 \supset \varphi_3}{\varphi_1 \supset \varphi_3}$

Disjunctive Syllogism $\dfrac{\varphi_1 \vee \varphi_2 \qquad \neg\varphi_1}{\varphi_2}$ *Controls* $\dfrac{P \text{ controls } \varphi \qquad P \text{ says } \varphi}{\varphi}$

Figure 3.1 Basic Rules

Figure 3.5 derived Rules

Use both to construct proofs.

Working Through a Formal Proof

Assume the following:

- *Ed* controls (*Flo* controls *delete*)
- K_E says (*Flo* controls *delete*)
- $K_E \Rightarrow Ed$
- *Flo* says *delete*

Hypothesis

Let's prove that *delete* should be granted.

Conclusion

Working Through a Formal Proof: Part 2

1.	Ed controls $(Flo$ controls $delete)$	Assumption
2.	K_E says $(Flo$ controls $delete)$	Assumption
3.	$K_E \Rightarrow Ed$	Assumption
4.	Flo says $delete$	Assumption
5.	$K_E \Rightarrow Ed \supset$ $((K_E$ says $(Flo$ controls $delete)) \supset$ $(Ed$ says $(Flo$ controls $delete)))$	Speaks For
6.	$(K_E$ says $(Flo$ controls $delete))$ $\supset Ed$ says $(Flo$ controls $delete)$	3,5 Modus Ponens
7.	Ed says $(Flo$ controls $delete)$	2,6 Modus Ponens
8.	Flo controls $delete$	1, 7 Controls
9.	$delete$	4, 8 Controls

Construct proof

Forward

Backwards

Conclusion

check from Top to Bottom

Exercise

Consider the following collection of hypotheses:

1. MC controls (Lee controls $purchase$)
2. CA controls ($K_L \Rightarrow Lee$)
3. $K_{CA} \Rightarrow CA$
4. K_{CA} says ($K_L \Rightarrow Lee$)
5. K_L says $purchase$
6. MC says (Lee controls $purchase$)

Give a formal proof of the statement *purchase*, using only the inference rules and derived rules discussed in class.

1
2
3
4
5
6
7 Lee Controls purchase (1, 6, ∅ Controls)
8 CA says (K_L ⇒ Lee) (3,4 , derived speech for)
⋮

Hypothesis
H_1, H_6

purchase — Conclusion

The End

Reasoning and Access Control: Proofs

Formal Proofs and Proof Trees

Constructing a Formal Proof

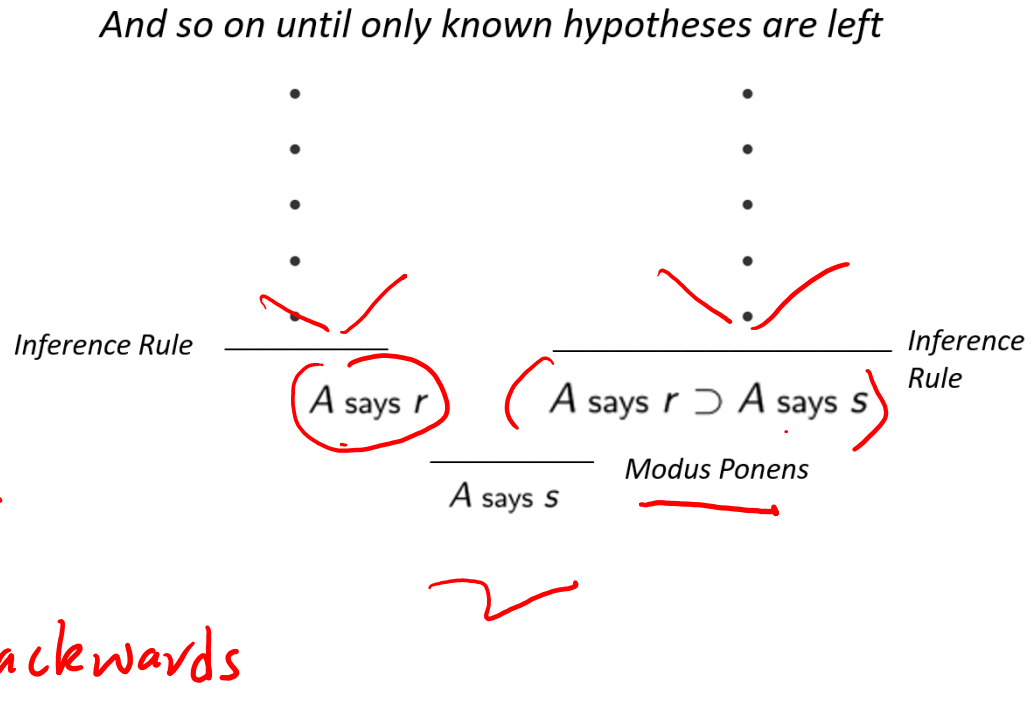
- How do you construct a formal proof?
- Finding a proof is like performing a search.
- Work from the hypotheses (forward).
- Work from the conclusion (backwards).
- Consider relevant inference rules by pattern matching.
- What is the overall *“structure”* of a proof?

Proof Trees

Revisit the proof for the derived rule

$$\frac{A \text{ says } (r \supset s) \qquad r}{A \text{ says } s}$$

A tree structure (proof tree)



good backwards

Work Backwards

- Construct proof trees

  Each step : match one inference

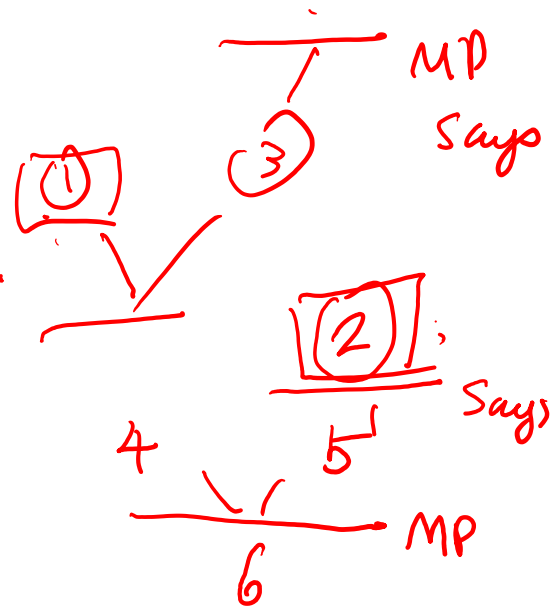
  : determine a list of "sub" goals

  (Hopefully easier)

Sample Formal Proof

Let's deduce $\underline{A \text{ says } s}$ from the following two assumptions:

$$A \text{ says } (r \supset s) \qquad r$$



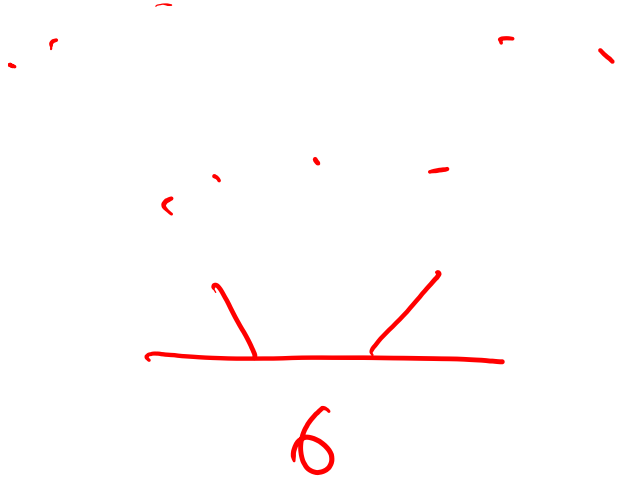
A formal proof:

1.	$A \text{ says } (r \supset s)$	Assumption
2.	r	Assumption
3.	$(A \text{ says } (r \supset s)) \supset (A \text{ says } r \supset A \text{ says } s)$	MP Says
4.	$A \text{ says } r \supset A \text{ says } s$	1,3 Modus Ponens
5.	$A \text{ says } r$	2 Says
6.	$A \text{ says } s$	4,5 Modus Ponens

The resulting derived rule: $\dfrac{A \text{ says } (r \supset s) \qquad r}{A \text{ says } s}$

Practice

Draw the proof tree for the sample formal proof given.



The End

Weekly Summary

Applying Formal Methods I

Summary

We apply formal methods to:

- Define the semantics of access-control logic through the introduction of Kripke models and the evaluation function.
- Identify the core question (the model-checking problem) and its solution.
- Develop a Set data type to implement the evaluation function, a key step to solve the model checking problem.
- Given a Kripke model and an access-control property, for which world in the model is the property true?

Overview

- Introduce the inference rules and show how to reason about access control by giving formal proofs.
- List and justify several derived inference rules to help construct formal proofs.
- Introduce proof trees as a tool to construct a formal proof.

The End