EPISTEMIC LOGICS AND THEIR APPLICATIONS IN ARTIFICIAL INTELLIGENCE

Ernest Davis and Leora Morgenstern
New York University IBM T.J. Watson Research Ctr.

IJCAI 1993
Chamberry

What Are Epistemic Logics?

---Logics of Knowledge

---Logics of Belief

and their extensions

Logics used to reason *about* Knowledge and Belief

What is Reasoning about Knowledge?

- Reasoning about how agents ***use*** knowledge
- -- how they reason with their knowledge
- -- how they reason with partial knowledge

[Pierre doesn't know the directions to Lyon,
but does know the number of the automobile club]

Does not include
Knowledge-based systems et. al.
which *use* knowledge (facts) , but don't reason *about* it

Aim of Tutorial

--- motivate the problem

--- introduce concepts, vocabulary

--- examine major issues

--- point to tools, applications

Why do we want to reason about knowledge and belief?

- **Inherently important ---**
  central concern of philosophy, psychology

  --- how do agents acquire knowledge (Theaetetus) ?
  -- what is the relation between knowledge and belief ?
  -- do agents know all the consequences of their knowledge ?
  -- how do you explain inconsistent beliefs ?

- **Important for applications**

  --- AI applications :
  planning, speech act theory, CAI
  --- Other CS applications:
  distributed systems, security
  --- Applications outside AI:
  economics

AI Applications

- Planning
- Text Understanding
- Active Perception
- Speech Acts
- Intelligent Computer Aided Instruction
- Design of Intelligent Systems
- Nonmonotonic Logic

Planning

-- In perfect world (complete knowledge)
planning can be done
without reasoning about knowledge

-- Real world ---- incomplete knowledge
so planning agent must reason

does agent have enough knowledge to
perform action?
does other agent know enough to do action?

Knowledge Preconditions Problem for Actions and Plans

From the New York Times, Metropolitan Diary, Nov. 27, 1991

Dear Diary:

This is what happened the other day.

Richard locked himself out of his West Fourth Street apartment. The super wasn't around. Two hours later, Richard was still waiting in his lobby. Then Mary Anne, an upstairs neighbor, came home. She didn't have the keys to Richard's apartment, but she had keys to Carol's apartment next door to her. And Carol, she knew, had keys to Lydia's apartment on the floor below. And Lydia, Richard knew, had keys to his apartment.

So Mary Anne used her keys to get into Carol's apartment where she found a set of keys labeled "Lydia." Then Mary Anne and Richard went to Lydia's apartment where Richard was certain he would find the keys to his apartment. And so he did. A few minutes later he was unlocking his own door.

There's a moral here someplace, maybe about good neighbors, maybe about New York apartment dwellers. On the other hand, it could be a question. Like, didn't Lucy and Ethel have it easy?

The Moral:
Intelligent agents reason about knowledge and action

Active Perception

[Having intelligent control for the focus of the sensor]

Using knowledge of sensor characteristics and of external world,
Predict that a given focus for the sensor will gather a desired piece of knowledge

"I can find out what I need to know
-- for planning
-- for physical prediction
-- for disambiguating my perceptual interpretations
by focussing my camera 10 degrees to the right"

-- I can determine whether my wallet is in my pocket by feeling in my pocket
-- I can determine whether a region is a mark or a shadow by looking for the object casting the shadow

Speech Acts (Grice)

--- modelling communication, use of language

--explains how
"la neige est blanche" *means* snow is white
[in contrast to
"These clouds mean rain"]

based on *convention*; common knowledge of what a sentence means

-- explains why
"Dear Sir,
Mr. X has an excellent command of English and always comes to class"
is a bad letter of recommendation

***conversational implicature*; our expectations and knowledge**

Intelligent CAI
(Computer Aided Instruction)

Create Automated Tutor

Ideally

Would maintain a model of
- **-- what the student knows**
- **-- how the student reasons**
- **-- how the student learns**

Can initialize the model from
- **-- generic model of students**
- **-- specific student data (e.g. tests)**

Can update the model from knowing
- **-- what the student has been taught**
- **-- how the student responds**

Can plan an effective teaching strategy

Design of Intelligent Systems

Automate the construction of a specialized AI system

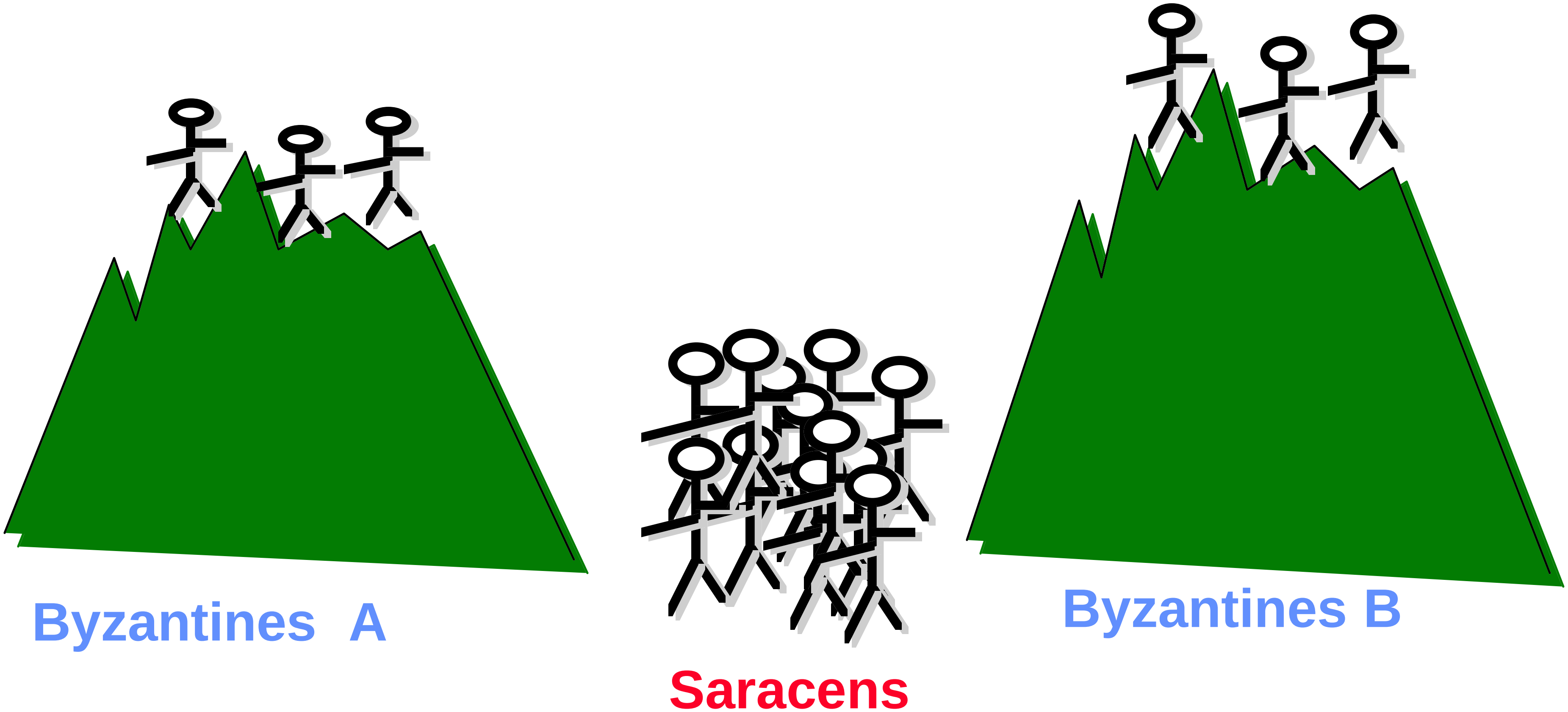
Given a specification of

-- the kind of knowledge that the system has
-- the evolution of the system's knowledge
-- the proper action of the system in a given state of knowledge

Design a knowledge-base architecture that implements this

Distributed Systems:
Byzantine Generals

Byzantines must coordinate attack;
otherwise, they'll be defeated



Distributed Systems: Byzantine Generals

- 2 Byzantine armies on opposite sides of Saracen army
- If both Byz. armies attack simultaneously, they win
- If only one attacks at a time, they'll be defeated

Objective: To decide on a time of simultaneous attack by sending messages back and forth

Difficulty: The general sending the messenger can't be sure that he will get through

Question: How long until they can coordinate an attack?

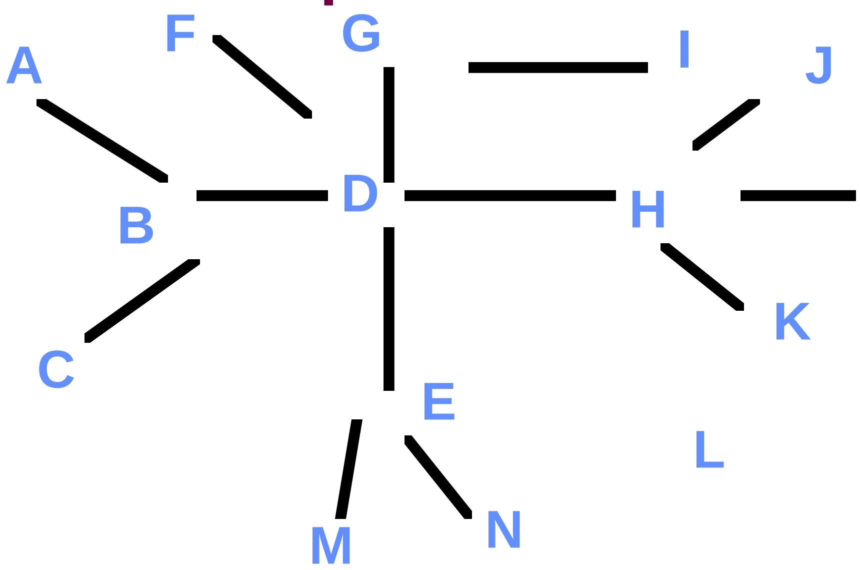
Theorem: There is no protocol which enables both generals to be sure that the other will attack

Relevance: components in computer system where messages don't always get through

Application: Distributed Systems

Given: A collection of nodes connected as a free tree.

Task: To impose a directed tree structure



Each node knows the general rules:
If u borders v then either u = parent(v) or v = parent(u);
The root has no parent; all other nodes have exactly 1 parent.

Protocol for u: if you border v, and you know your relation to v, communicate this to v.

Supports variety of information transmission patterns:

- **Assign one node to be root: info spreads from root**
- **Assign n-1 nodes to root; info spread up from leaves**

Applications of Epistemic Logic in General CS

Distributed Systems

Characterize a protocol for a distributed system in terms of

--What each element knows
--What each element wants to know
--What each element knows about what other elements know
--What is known by the union of all elements
--What is common knowledge throughout the system

Security

Guarantee that someone who does not know P (password) cannot find out Q (information)

Convince another agent that I know a solution to problem P without letting him know the solution

(Public key encryption, zero-knowledge proofs)

Applications of Epistemic Logic outside CS

GAME THEORY: (Partial knowledge games - Bridge, Poker)

Determine the likely action of the opponent based on his knowledge

Use one's knowledge effectively without revealing its source

ECONOMICS:

Determine the expected cost and value of a particular piece of knowledge

OUTLINE

- What are epistemic logics?
- Why are they interesting ?

--- AI Applications

- Representing knowledge

--- Modal Logics

--Syntax

--Semantics

state-based definition, possible worlds

--Extensions

quantification, time

--Applications

3 Wise Men, Byzantine Agreement

--Problems

--- Syntactic Logics

--Syntax and Semantics

-- Advantages and Disadvantages: Paradox

-- Resolution to Paradoxes

--- Additional Issues

--Dropping Consequential Closure

--Nonmonotonic Logics

- Using Representations

Important issues not covered here:

- connection of knowledge to other propositional attitudes: belief, hope, fear, desire
- [nonmonotonic and] probabilistic inference
- knowledge and perception
- natural language use of “know”

Representing Knowledge

- --- Modal Logics
  - --Syntax
  - --Semantics
    - state-based definition, possible worlds
  - --Extensions
    - quantification, time
  - --Applications
    - 3 Wise Men, Byzantine Agreement
  - --Problems
- --- Syntactic Logics
  - --Syntax and Semantics
  - -- Advantages and Disadvantages: Paradox
  - -- Resolution to Paradoxes
- --- Additional Issues
  - --Dropping Consequential Closure
  - --Nonmonotonic Logics

Representing Knowledge

How do we talk about knowledge?

LOGIC ---

Extending standard logic into logic of knowledge

Starting point: Propositional Logic

Propositions: P Q R

Connectives: ~ v & ==> <==>

Examples:

Temp = 25

Frog-Kermit v Temp = 25

Frog-Kermit ==> Green-Kermit

Note: no way to talk about knowledge

Extending Propositional Logic to Modal Logic of Knowledge

Add *modal operator Know* (applies to sentences)

Examples:

Know(Frog-Kermit)
Know(Frog-Kermit v ~ Frog-Kermit)
Temp = 25 & ~ Know(Temp = 25)
Know(~ Know(Frog-Kermit)) **[nested knowledge]**

Implicit agent; can make explicit

Know(Beth, Temp = 25)
Know(Sally, Frog-Kermit ==> Green-Kermit)
Know(Sally, Know(Beth, Temp = 25)) **[nested knowledge]**

Note: difficulty in representing knowledge

Referential Opacity

Most predicates are transparent;
you can substitute equals for equals.

John is the father of William
Color-of-eyes(John, Brown) is true just in case
Color-of-eyes(father(William),Brown) is true

Not true of Know

Scott is the author of Waverly --- but
Know(British(Scott)) may be true and
Know(British(author(Waverly)) may be false.

Also, the Morning Star is the same as the Evening Star
But Know(MorningStar = MorningStar) is true of all;
But Know(MorningStar = EveningStar) is not.

Know is Opaque

Modal Logic of Knowledge: Multiple Agents

<u>Mutual Knowledge</u>

2 or more agents know some fact

A and B mutually know P iff
Know(A,P) & Know(B,P)

very important for distributed systems!

<u>Common Knowledge</u>

2 or more agents know some fact and they know that they know, and so on ...

A and B have common knowledge of P iff
Know(A,P) and Know(B,P) and
Know(A, Know(B, Know(A, ... P]
Know(B, Know(A, Know(B, ... P]

How can we define the Know operator?

Intuitive definition of Know:

--- whatever is explicitly stated in a knowledge base

--- implicit knowledge in a propositional knowledge base

--- what can be derived

implicit knowledge definition is most accepted

We need to write down axioms to capture this concept of knowledge

[Possible] Axioms on Knowledge

1. **Veridicality**
   If A knows P, then P is true
2. **Consequential Closure**
   If A knows P_1 ... and A knows P_n and P_1 ... P_n |- Q
   then A knows Q
3. **Knowledge of Necessary Truths**
   If P is necessarily true, then A knows P
4. **Positive Introspection**
   If A knows P then A knows that A knows P
5. **Negative Introspection**
   If A does not know P, then he knows
   that he does not know P

Different subsets of these axioms form different systems of modal logic

1. **Veridicality**

   If A knows P, then P is true

2. **Consequential Closure**

   If A knows P_1 ... and A knows P_n and P_1 ... P_n |- Q then A knows Q

3. **Knowledge of Necessary Truths**

   If P is necessarily true, then A knows P

== T, a simple modal logic of knowledge

1. **Veridicality**

   If A knows P, then P is true

2. **Consequential Closure**

   If A knows P_1 ... and A knows P_n and P_1 ... P_n |- Q then A knows Q

3. **Knowledge of Necessary Truths**

   If P is necessarily true, then A knows P

4. **Positive Introspection**

   If A knows P then A knows that A knows P

== S4, popular modal logic of knowledge

1. Veridicality

If A knows P, then P is true

2. Consequential Closure

If A knows P_1 ... and A knows P_n and P_1 ... P_n |- Q
then A knows Q

3. Knowledge of Necessary Truths

If P is necessarily true, then A knows P

4. Positive Introspection

If A knows P then A knows that A knows P

5. Negative Introspection

If A does not know P, then he knows
that he does not know P

== S5, modal logic for ideal knowledge

2. Consequential Closure

If A believes P_1 ... and A believes P_n and P_1 ... P_n |- Q
then A believes Q

3. Knowledge of Necessary Truths

If P is necessarily true, then A believes P

4. Positive Introspection

If A believes P then A believes that A believes P

5. Negative Introspection

If A does not believe P, then he believes
that he does not believe P

== weak S5, better suited for belief

Note: an agent's beliefs are not necessarily true

What does <u>Know</u> mean?
When can we say that a sentence like

Know(John, temp = 25)

is true?

Several characterizations:

-- state-based
-- possible-worlds semantics

State-based definition of knowledge

[good for a machine]

Consider a device D that can be in one of a collection of states.
We say: D knows P
if the state of D is S and
whenever D is in state S, P is true

Examples:

D: a mercury thermometer S: the level of mercury

D knows temperature is 25C
because mercury points to 25C and
mercury only points to 25C when it is 25C

D knows temperature is not 0C
because because temp. is never 0 when mercury is at 25C

State-based definition of knowledge, cont.

Also ....

D knows that the temperature > 5C and < 100C
D knows that the temperature in degrees C is a square number
D knows that any planar map can be colored with 4 colors
D kows that either the sun is shining or it is not shining

And ...

D does not know that the sun is shining
because sometimes mercury points to 25 when it rains

State-based definition of knowledge

Other examples:

D : an inventory database
S : relation instance in the database

D knows that there are 5 widgets on the shelf

D knows that there are fewer than 8 widgets on the shelf

State-based definition of knowledge

Combined knowledge

Devices D1 and D2 together know P if
the state of D1 is S1
the state of D2 is S2
whenever the state of D1 is S1 and the state of D2 is S2, P is true

Example:
D1: an inside thermometer D2: an outside thermometer

D1 and D2 together know it's colder outside than inside
because D1's mercury points to 22
D2's mercury points to 10
whenever D1's mercury is at 22 and D2's mercury is at 10, it's colder outside than inside

Combined Knowledge: Example

D1: device in store that keeps track of purchases
D2: device in store that keeps track of incoming orders

D1 and D2 together knows that there are 10,000 items in the store
because D1 indicates 30,000 items have been purchased
D2 indicates 40,000 items have come in

State-based definition of knowledge
Common Knowledge

Devices D1 and D2 have common knowledge of P iff
- **state of D1 is an element of some set of states SS1**
- **state of D2 is an element of some set of states SS2**
- **whenever state of D1 is in SS1,**
  - **P is true and the state of D2 is in SS2**
- **whenever state of D2 is in SS2,**
  - **P is true and the state of D1 is in SS1**

Example:
- **D1: a digital clock that displays hour and minute**
- **D2: a digital clock that displays only the hour**

D1 and D2 have common knowledge that time is between 12:00 and 1:00 iff
- **SS1 = the set of all displays 12:xx on D1**
- **SS2 = the set of D2 displaying 12:00**

Applications of the State-based Definition: Distributed Systems

Let P = "Printer 0 is free"

Machine M1 knows p iff
the internal state of M1 is attained only when P is true.

M1 communicates P to M2 through message C if
M1 only sends C to M2 when M1 knows P and
Whenever M2 receives C from M1, it enters a state where
M2 knows P

M2 knows that M1 knows P if:
whenever M2 is in its current internal state,
internal state of M1 is one only attained when P is true

Protocol: rules of the form --
if Mx knows Pk, Mx communicates Cxyk to My

Verify concepts of protocols such as: if P true, then in
5 cycles, every machine will know that P is true

State-based definition of knowledge

What properties hold?

Important result:

Get: veridicality

consequential closure

necessary truths

positive introspection

negative introspection

That is, we get the modal logic S5 (perfect knowledge)

Problems with
State-based definition of knowledge

--- fine for machines;
unintuitive definition for intelligent agents

--- consequential closure:
all agents are perfect reasoners

--- necessary truths:
all agents know all axioms (universal facts)

--- negative introspection:
agents never have false beliefs

built into the semantics

Possible Worlds Definition of Knowledge

Kripke-Hintikka

idea: A knows P iff
P is true in all worlds that are knowledge-accessible for A

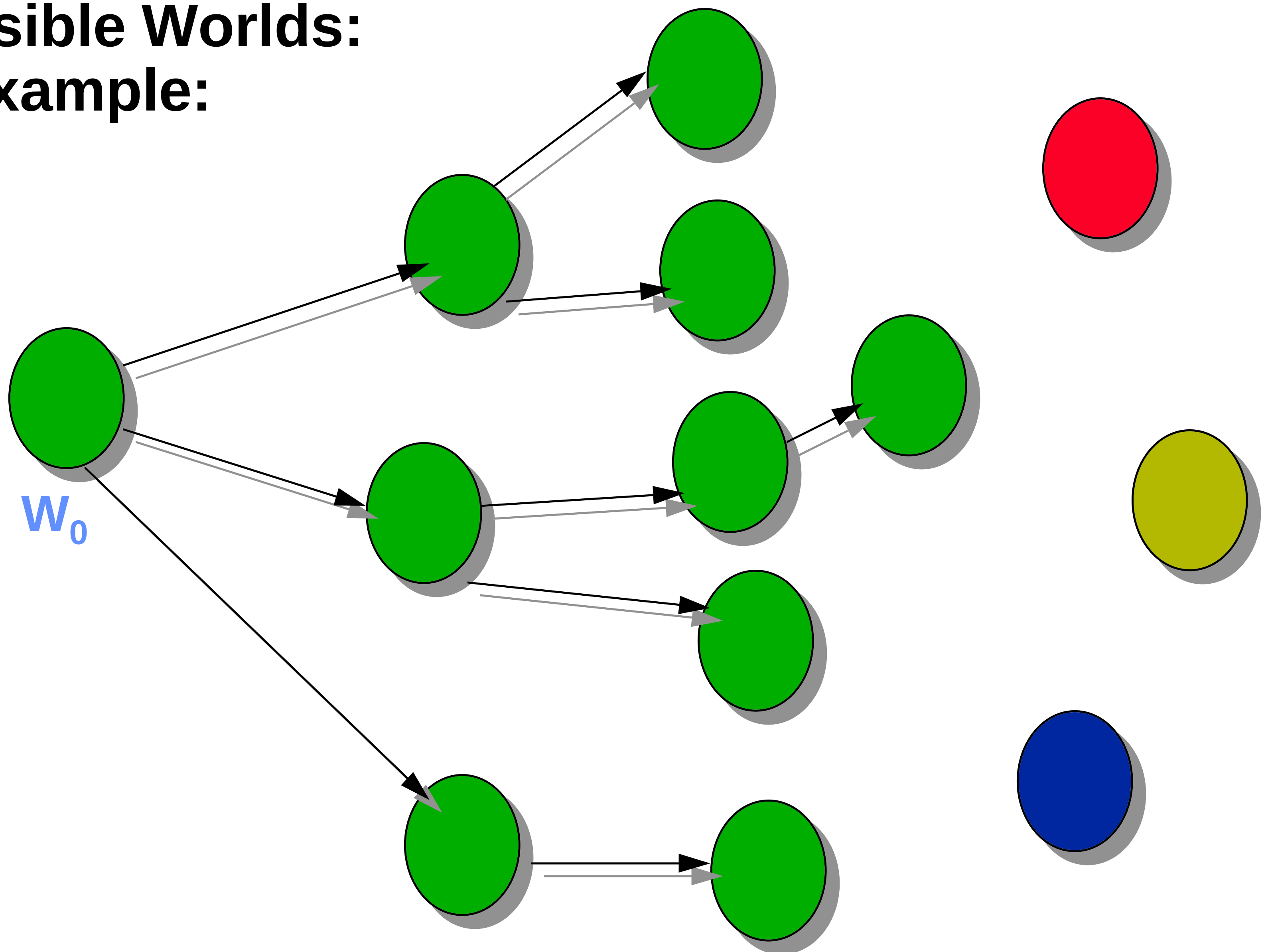
W_1 is knowledge-accessible from W_0 for A iff W_1 is consistent with everthing A knows in W_0 ; that is, for all A knows in W_0, he might as well be in W_1

Beth knows Kermit is green if he is green in all worlds that are knowledge-accessible to Beth

On the other hand, if in some knowledge-accessible world, Kermit is yellow, Beth doesn't know that Kermit is green

True(W_0, Know(A,P)) iff
forall W_1 K(a,W_0,W_1) ==> True(W_1,P)

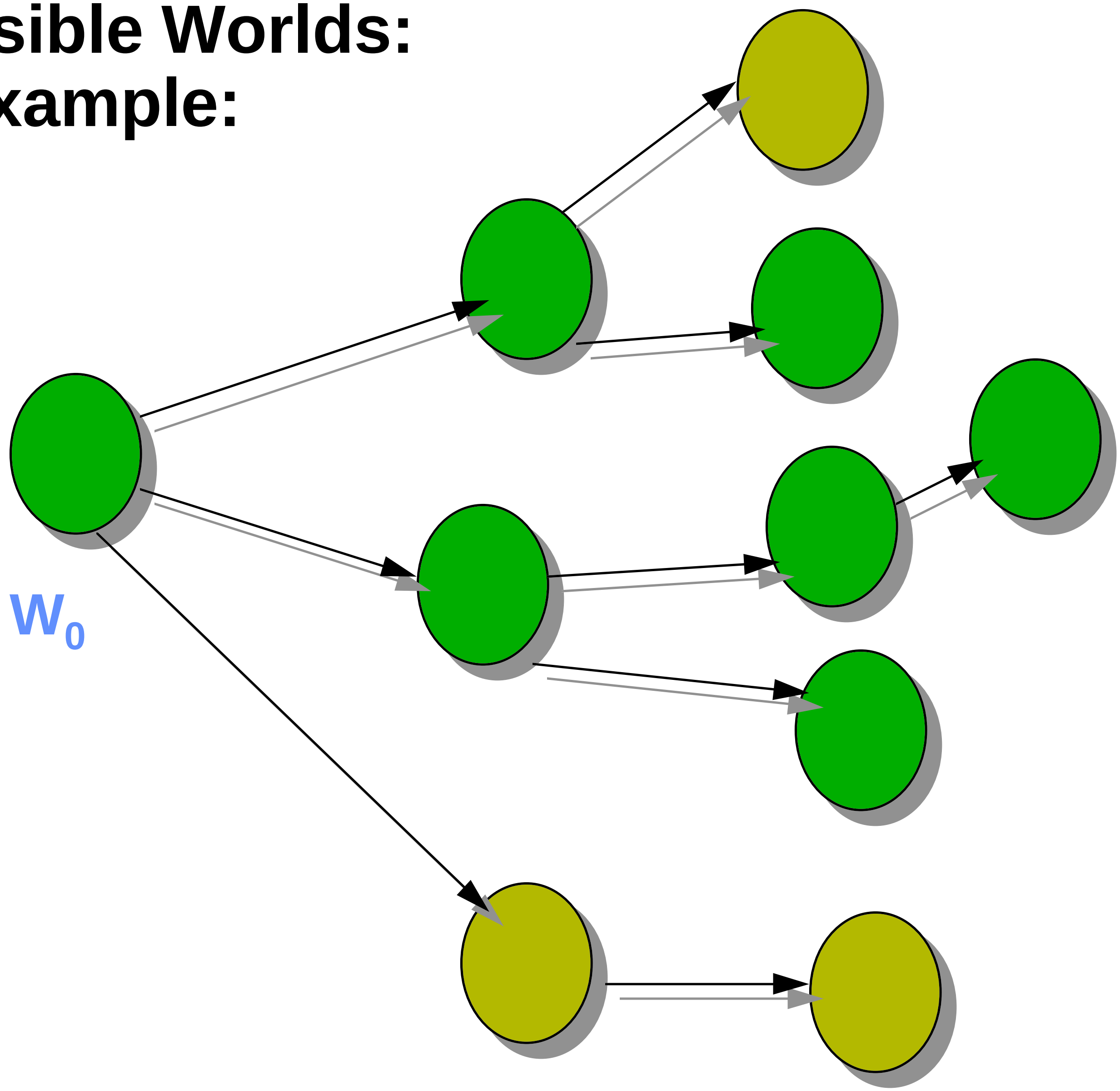
Possible Worlds: Example:



Beth knows that Kermit is green

(since Kermit is green is all words that are knowledge accessible to Beth from W_0)

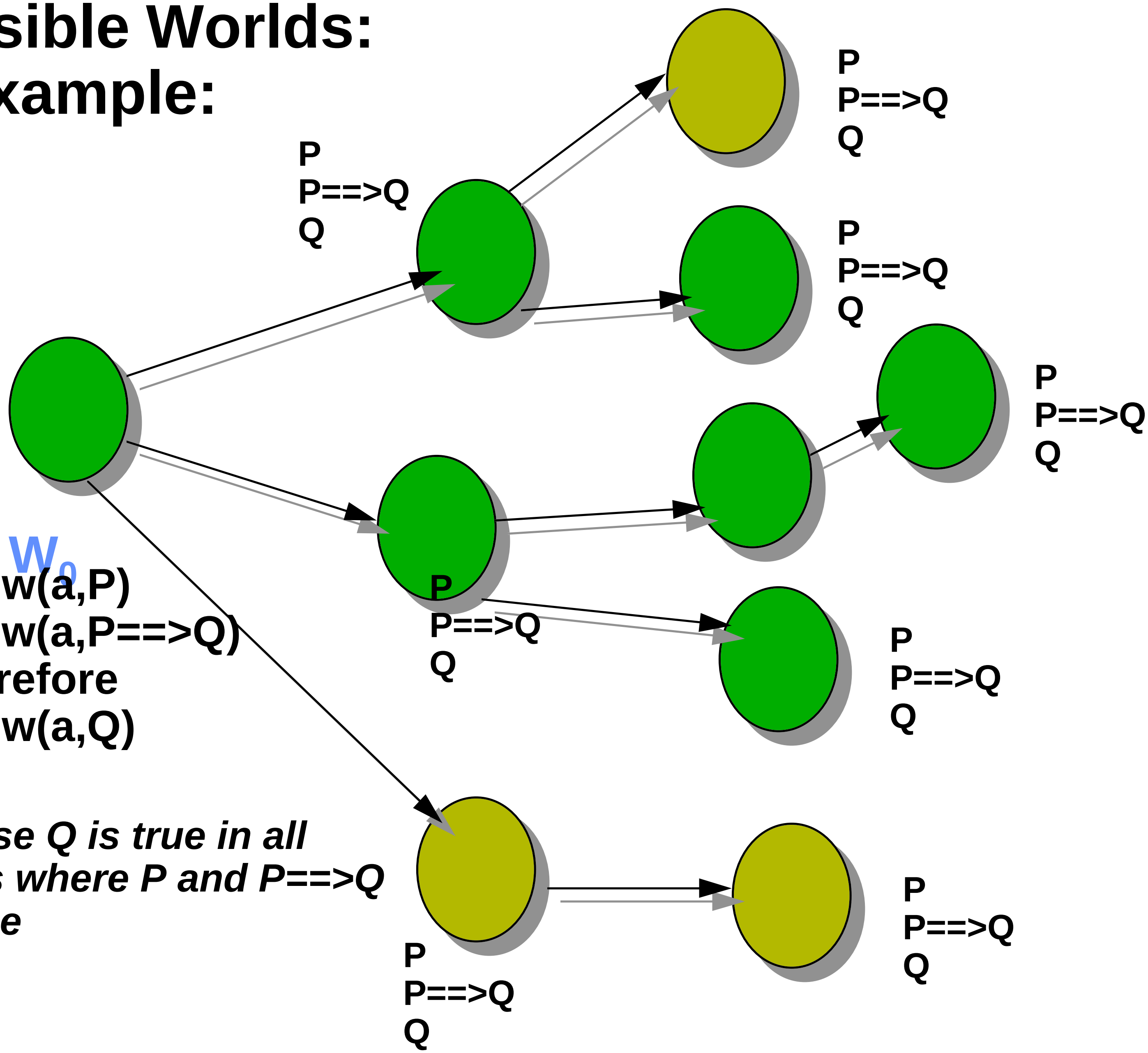
Possible Worlds:
Example:



Beth doesn't know whether Kermit is green

(since Kermit is green in some worlds; yellow in others)

Possible Worlds: Example:

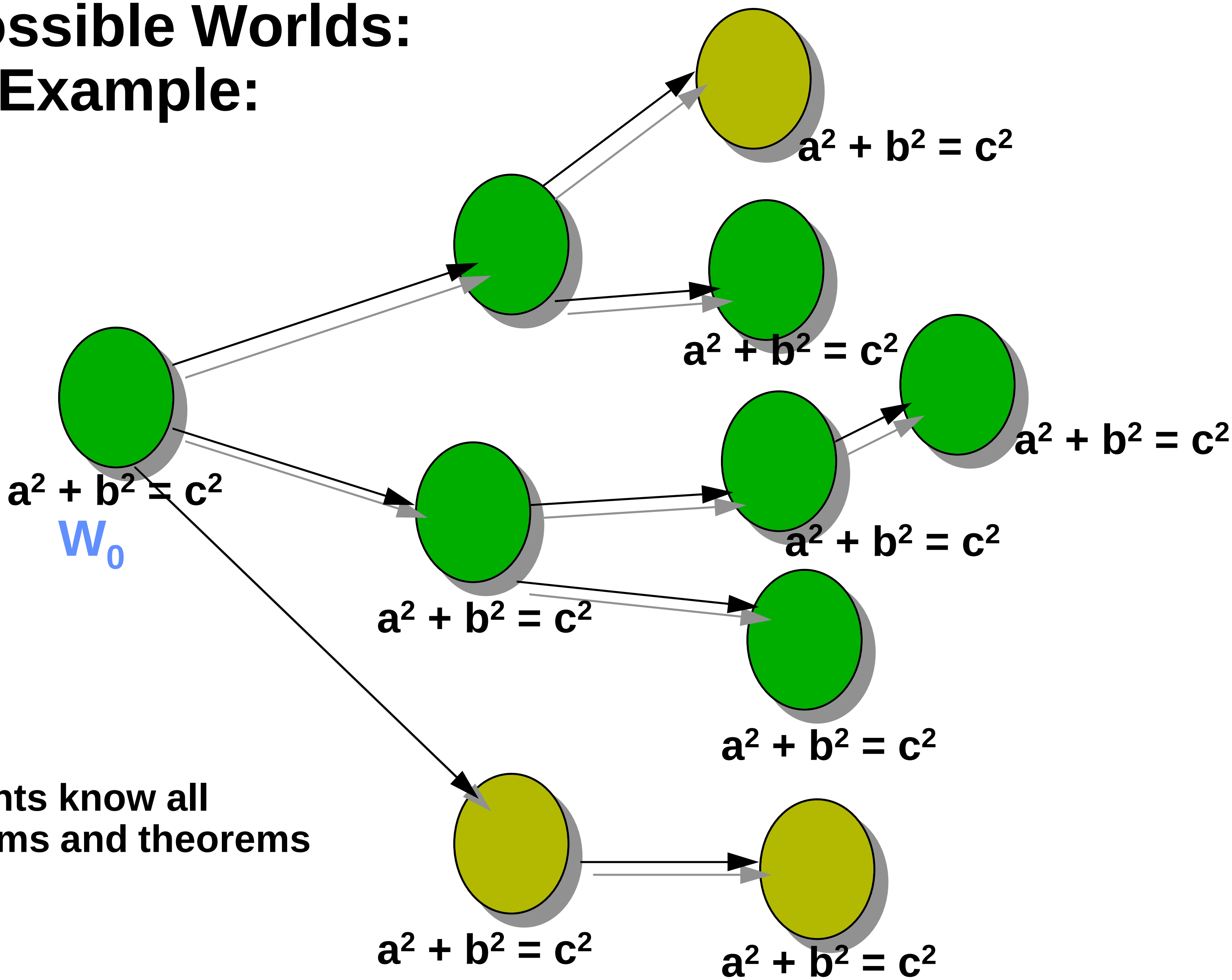


Know(a,P)
Know(a,P==>Q)
Therefore
Know(a,Q)

because Q is true in all worlds where P and P==>Q are true

Agents always know the consequences of their knowledge !!

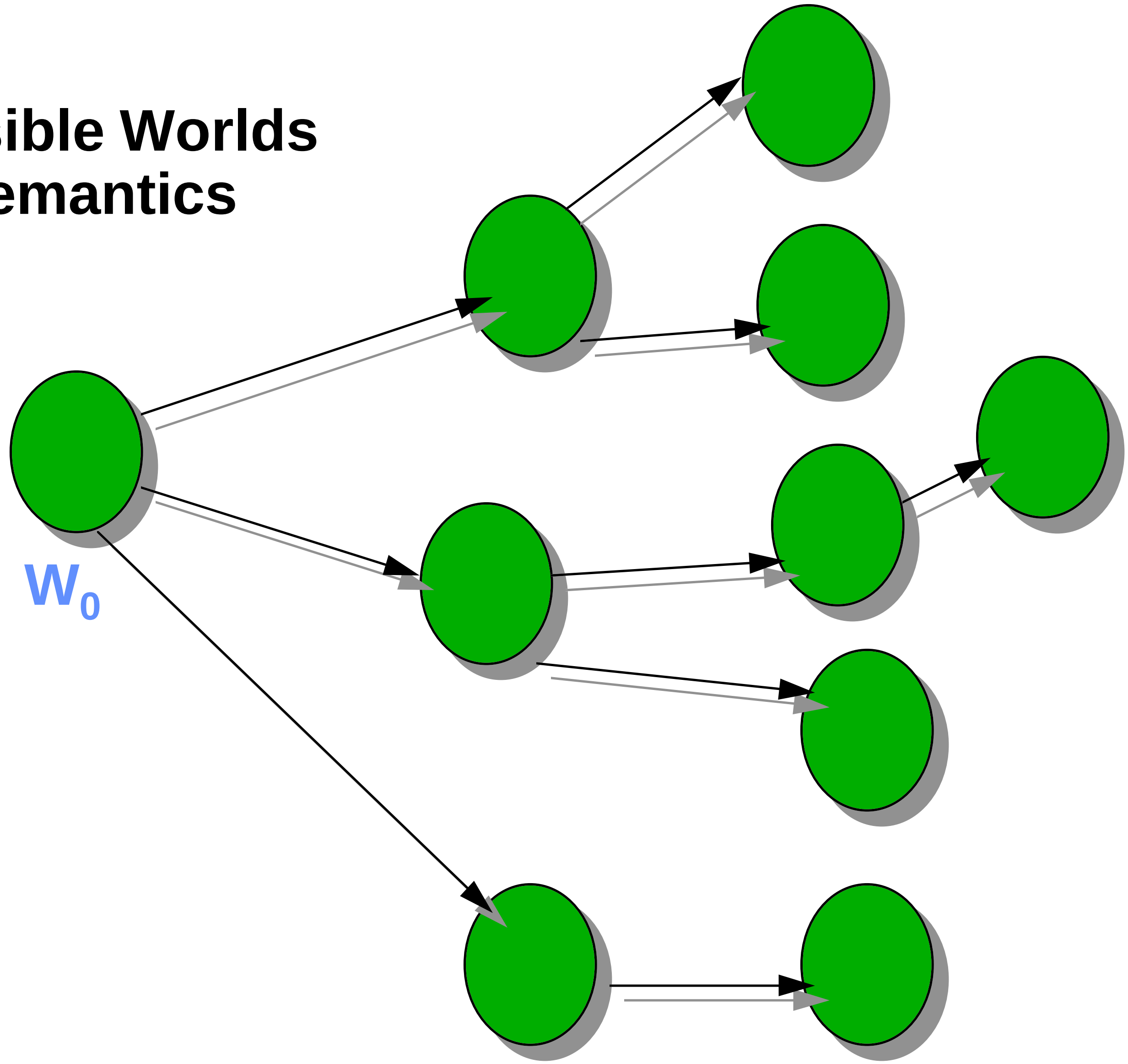
Possible Worlds: Example:



Agents know all axioms and theorems

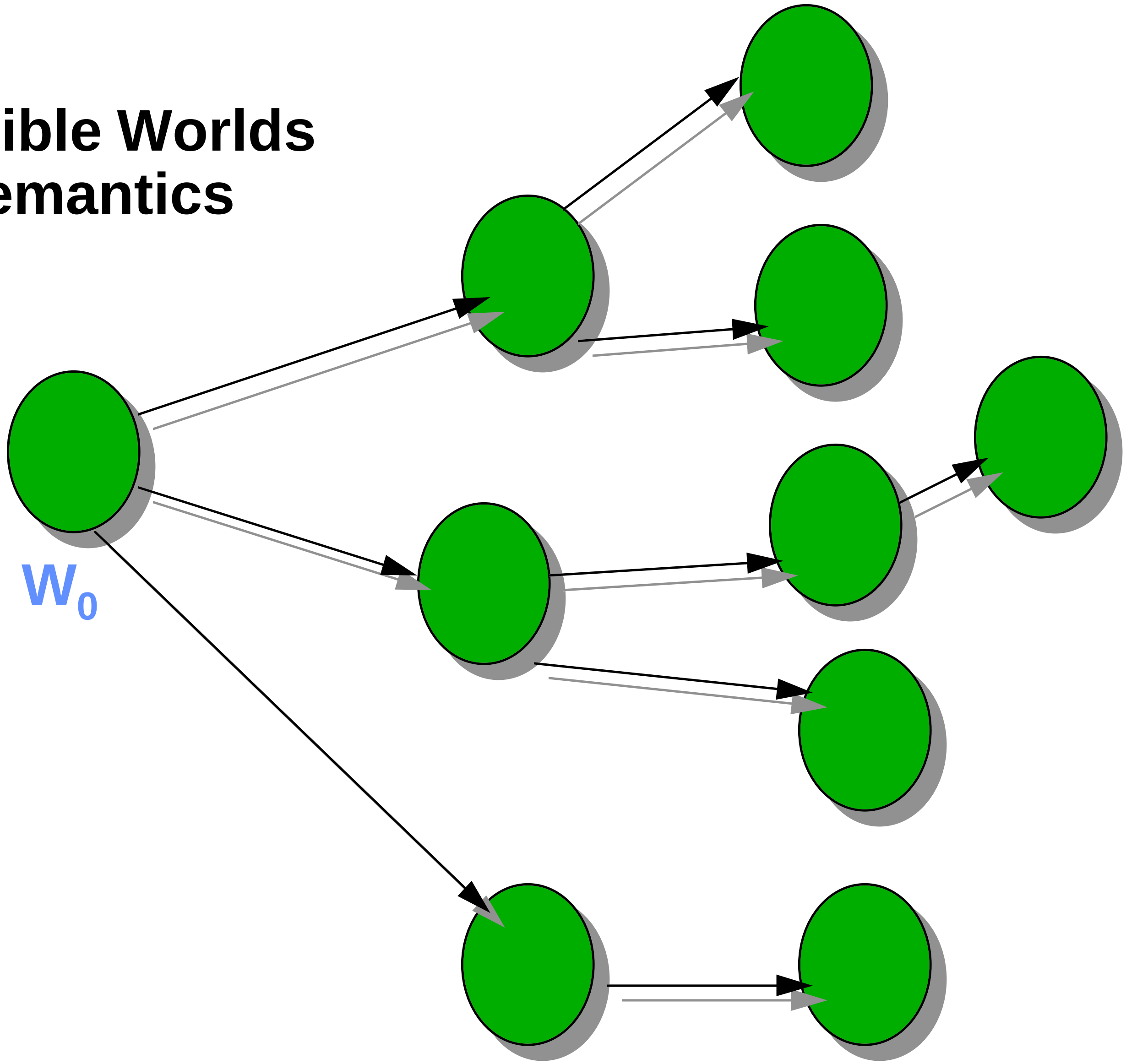
Since Pythagorean theorem is true in all worlds, Beth knows it (even if she's 5 years old)

Possible Worlds Semantics



Note: Different modal logics (subsets of axioms) correspond to properties of the knowledge accessibility relation

Possible Worlds Semantics

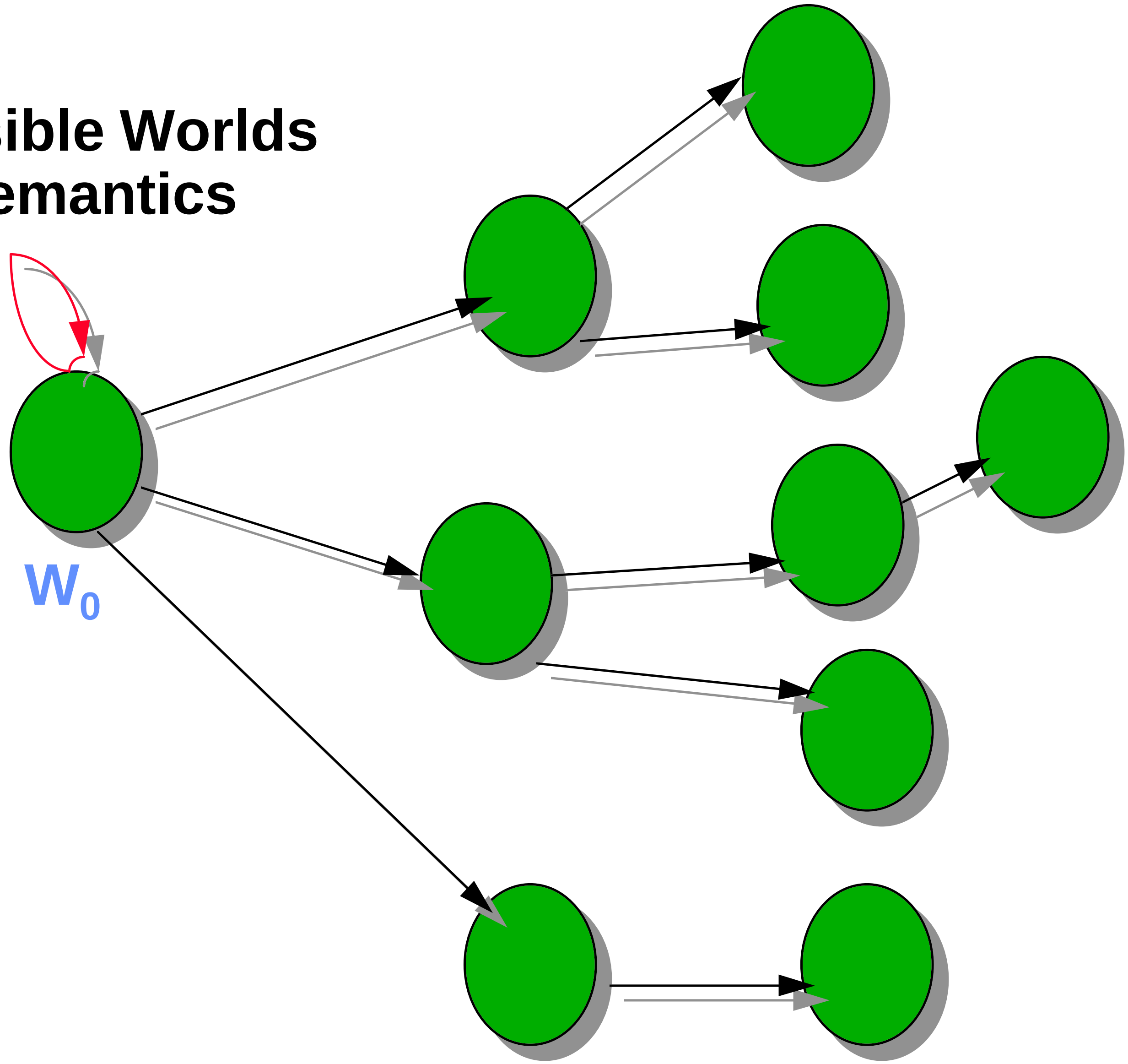


No restrictions:

Just get consequential closure and necessary truths

(weak T)

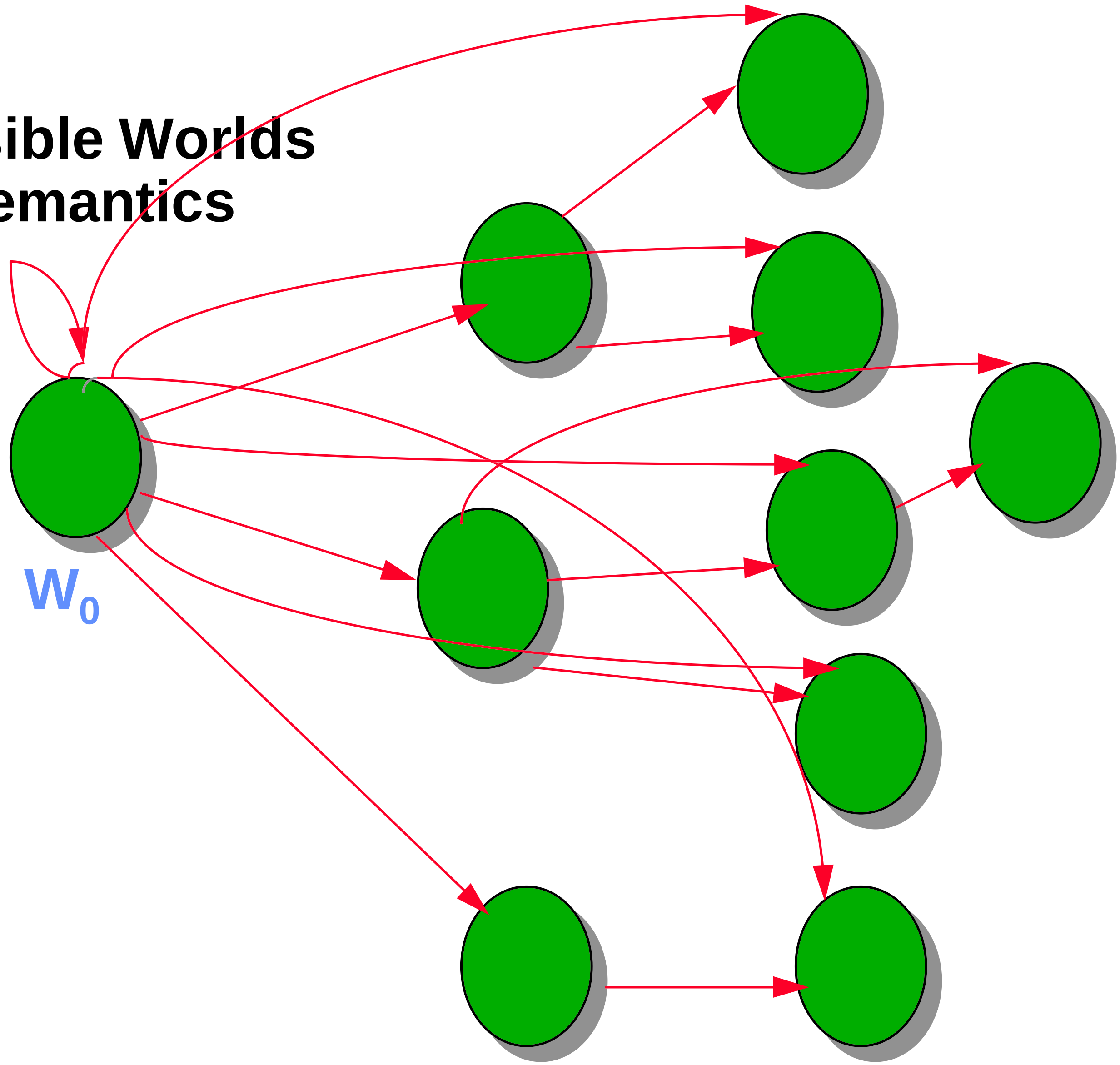
Possible Worlds Semantics



If K is reflexive, we get :

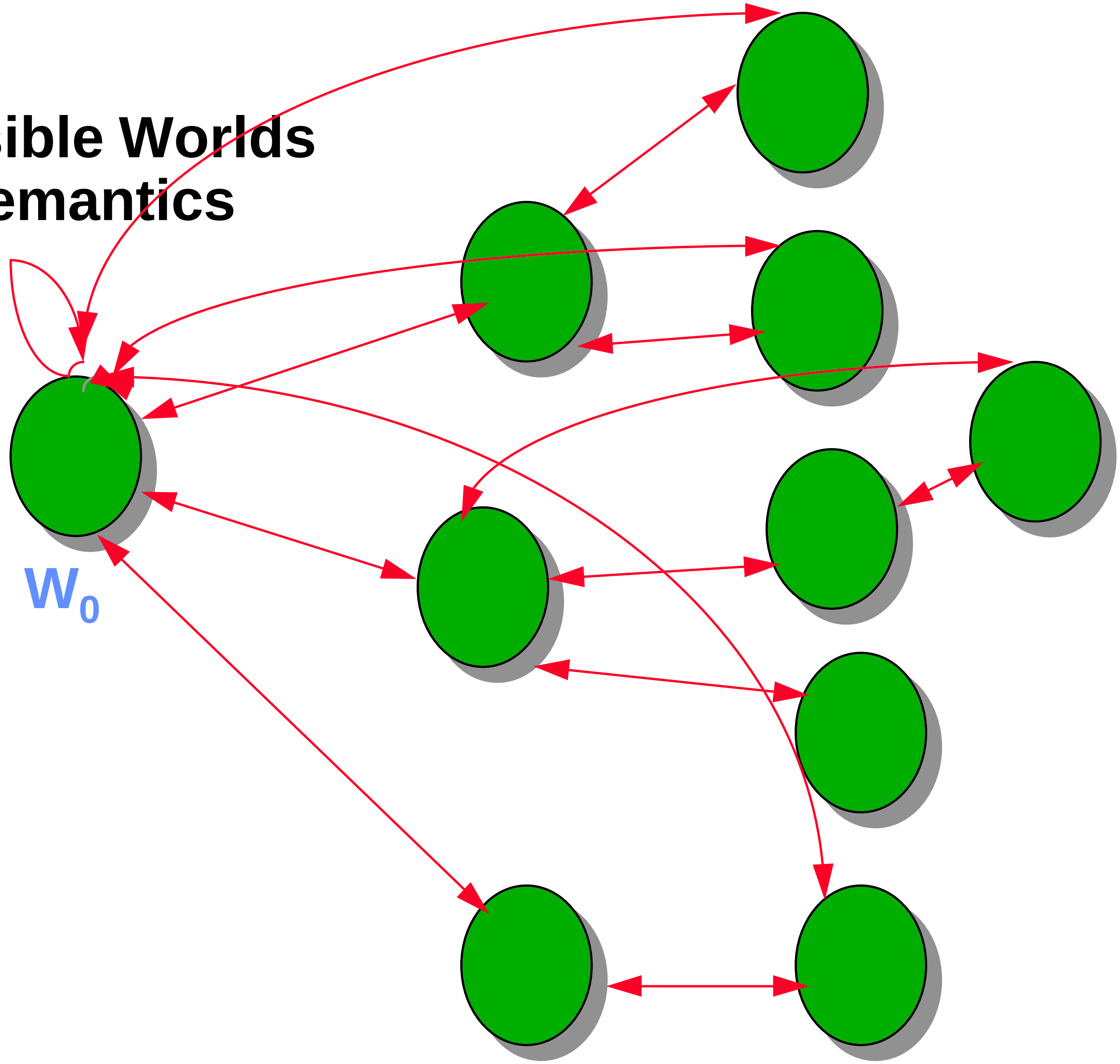
veridicality, consequential closure, necessary truths (T)

Possible Worlds Semantics



If K is reflexive and transitive we get :
veridicality, consequential closure, necessary truths, and positive introspection, (S4)

Possible Worlds Semantics



If K is reflexive, symmetric, and transitive we get :
veridicality, consequential closure, necessary truths, positive introspection, and negative introspection (S5)

Problems with
Possible Worlds definition of Knowledge

--- Is "knowledge-accessible" any more intuitive than knowledge?

--- Consequential closure:
all agents are perfect reasoners

--- Necessitation:
all agents know all axioms (universal facts)

Built into the semantics; can't take these out

[restricts us to a very small class of modal logics]

Extending Logics of Knowledge

Some Directions for Extensions:

-- Adding quantification
  Quantifying into epistemic contexts

-- Adding the concept of time

-- Dropping "perfect reasoner" assumption
  (consequential closure)

Adding Quantification

Before base logic was propositional logic

e.g. P = There's snow on the ground

Q = It's cold outside

P P & Q P v Q ~ P P ==> Q

Know (John, ~P) Know(John, P ==> Q)

Now base logic is predicate logic

(∀x) (Man(x) ==> Mortal(x))

(∃x) (Green(x))

Know(Beth, Blue(Toyota22))

Know (Susan, (∀ x) (Man(x) ==> Mortal(x))

Quantifying into Epistemic Contexts

Consider the following sentence:

John knows someone is blackmailing him

2 possible readings:

1. Know(John, ∃x Blackmailer(x,John)) (de dicto)
2. ∃x (Know(John, Blackmailer(x,John))) (de re)

Second reading more fit if
John knows who is blackmailing him

Note: 2. implies 1., but 1. does not imply 2

Quantifying into Epistemic Contexts

- When can we say

$\exists$x(Know(John,ϕ(x))
[$\exists$x (Know(John, Blackmailer(x)))]

We can deduce it from

Know(John, Blackmailer(Mr.Thorpe, John))

but not necessarily from

Know(John, Blackmailer(Murderer(Sam), John)))

How much knowledge does John have to have ?
constant? name? rigid designator?

Adding the concept of time

Till now: no concept of time

Know(Beth, Green(Kermit))

But -- facts of knowledge refer to *time*
People know different things at different times

Know(Beth, President(USA, Clinton))

is meaningless
When is Clinton President?
When does Beth know this?

Need to add concept of time

Adding time to epistemic logics

Many methods:

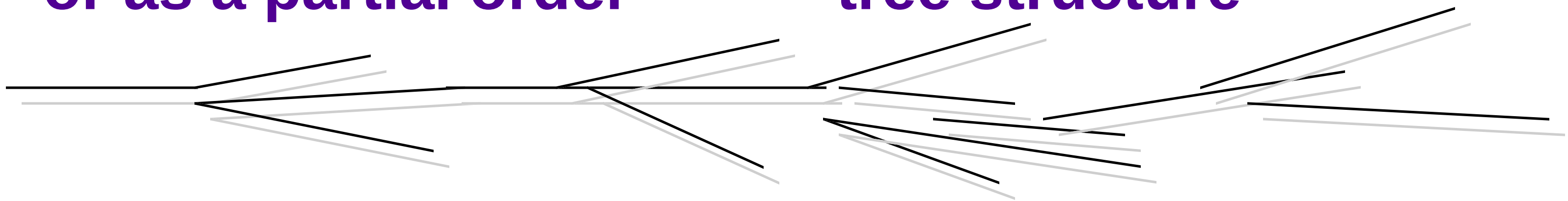
Adding extra temporal argument

Know(Beth, President(USA, Clinton, 1993)), 1993)

Know(Beth, President(USA, Bush, 1992)), 1993)

can imagine time as a total order --- time line

or as a partial order ---- tree structure



Adding Time

How is knowledge affected by time?

Perfect memory

Know(A,P,S1) & S1 < S2 ==> Know(A,P,S2)

Belief is also affected by time:

Changing your mind --

if A believes P, he believes he'll always believe P

Bel(A,P,S1) ==>
Bel(A, forall S1 S1 < S2 ==> Bel(A,P,S2), S1)

Future beliefs --

if A believes he'll believe P in future, he believes it now

Bel(A, ∃ S2 S1 < S2 and Bel(A,P,S2), S1)
==> Bel(A, P, S1)

Adding Time

Predicting the future:
reasoning about the effects of an action

How to say:
Susan knows that if she moves block A to block B,
block A will be on top of block B

Need to:
integrate logic of knowledge with logic of action

Situation Calculus

situation = instant of time

situations partially ordered by < :

branching model of time

Actions = functions on situations

E.g., Puton(A, B) maps situations in which blocks A and B are clear to situations in which block A is on top of block B

True-in(Result(put-on(A,B),S), on(A,B))

Know(Sam, True-in(Result(put-on(A,B),S), on(A,B)))

Application:
Three Wise Men Problem

In other guises: Dirty Children Problem
Cheating Husbands Problem

Idea: Three wise men are told that at least one has a black dot on his forehead.
Everyone can see if others have black dots, but no-one can see his own forehead.

Assume that we start at t = 0.

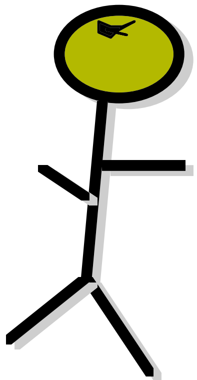
All are perfect reasoners.

Any round of reasoning takes one unit.

If all of the wise men have black dots, how long will it take them to realize? If 2 have dots? if 1 does?

Three Wise Men

BASE CASE: 1 Wise Man

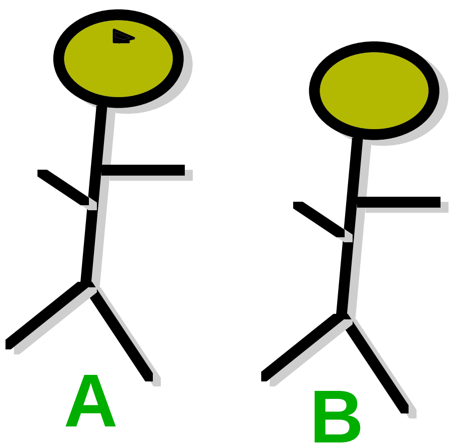


This is trivial; he knows he has a dot on his forehead so he says it right away, at t = 0.

Three Wise Men

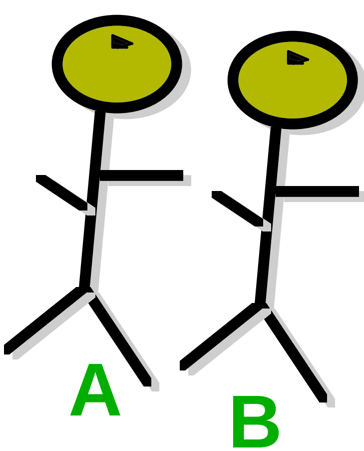
Now suppose there are 2 men

Case I: 1 man has a black dot



At time t = 0, A sees that B doesn't have a dot. Since he knows that one of them has a dot, he figures that he does. So at t = 1, A says: I have a black dot. (B can't figure anything out.)

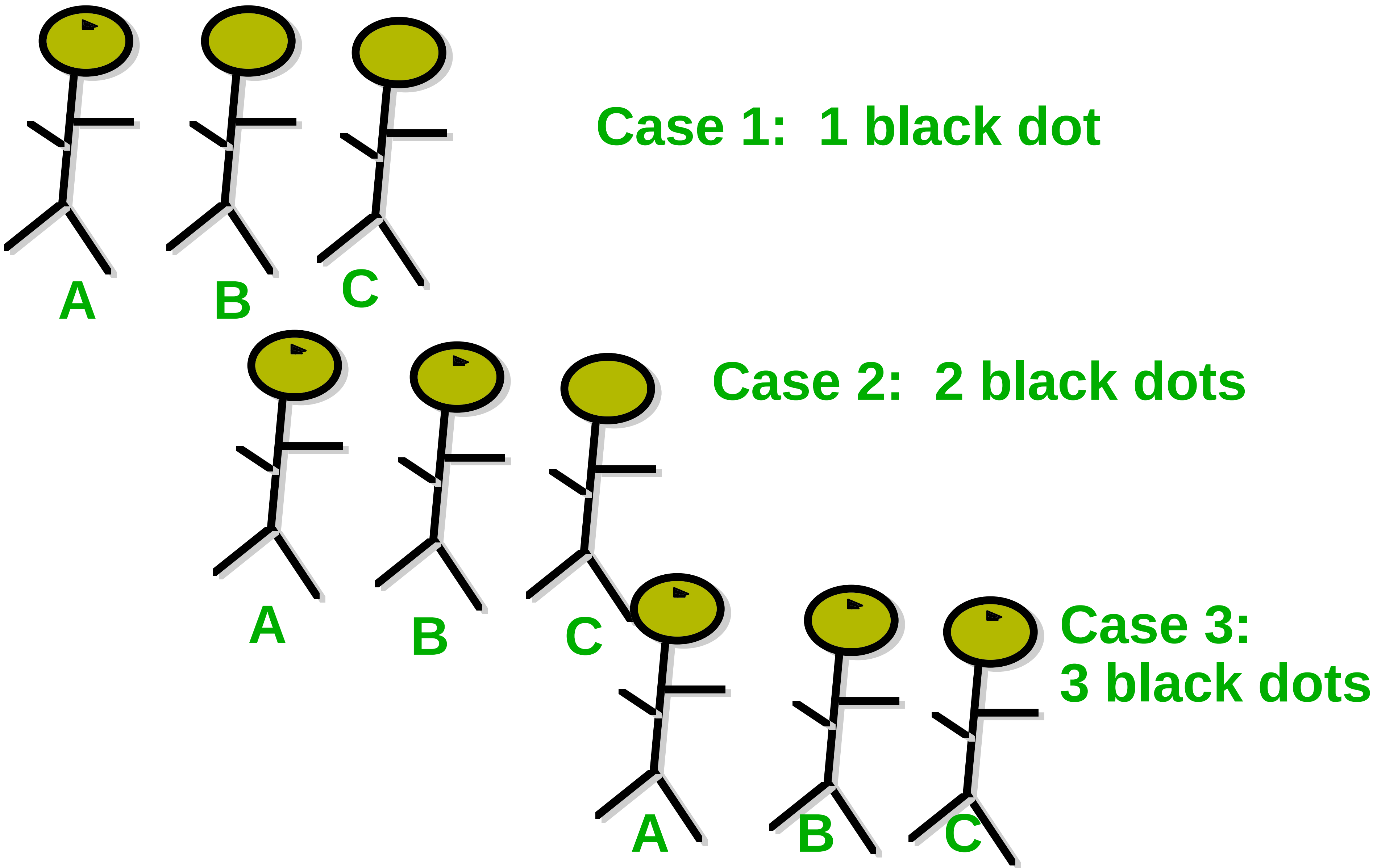
Case 2: both men have dots



At time t = 0, A sees that B has a dot. Thus, he doesn't know if he does or not. But at time t = 1, B is silent (he doesn't know if it's case 1 or case 2). So A knows that this *can't* be the same as case1; thus he must also have a dot. So he speaks out at time t = 2. B, doing the same reasoning, also speaks at t = 2.

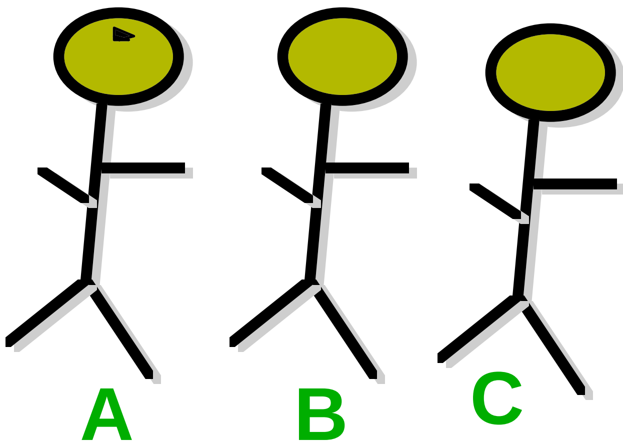
Three Wise Men

Now suppose there are 3 men



Three Wise Men

Case of 3 men

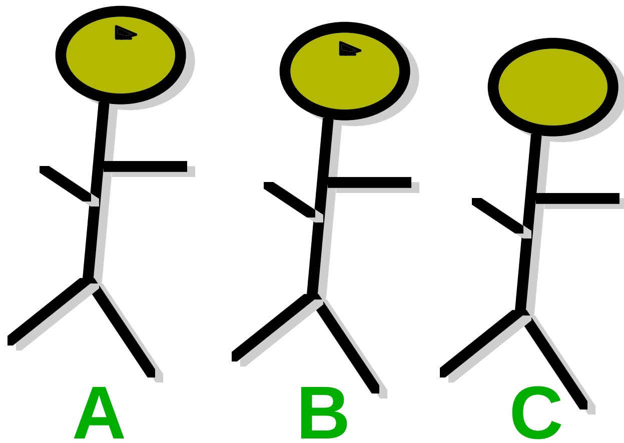


Case 1: 1 black dot

**At time t = 0, B and C each see one person with a dot.
So they may have dots on their forehead; they don't know.
But A doesn't see anyone with a dot on his forehead,
so he knows he must have a dot on his forehead.
So, at time t = 1, he speaks.**

Three Wise Men

Case of 3 men

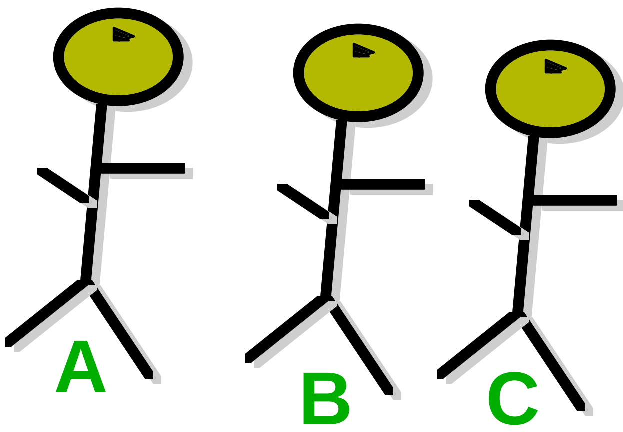


Case 2: 2 black dots

At time t = 0, everybody sees at least one person with a dot, so they don't know if they have dots.
A and B each see 1 person with a dot, so they know: either there is 1 person with a dot, or 2 people.
At t = 1, no-one speaks. So A [resp. B] knows it can't be that only B [A] has a dot. Because if that were the case, at time 1, B [A] would have spoken. Thus, there must be 2 people with dots -- i.e., A [B] has a dot too.
At t = 2, A   [and B!] speak.

Three Wise Men

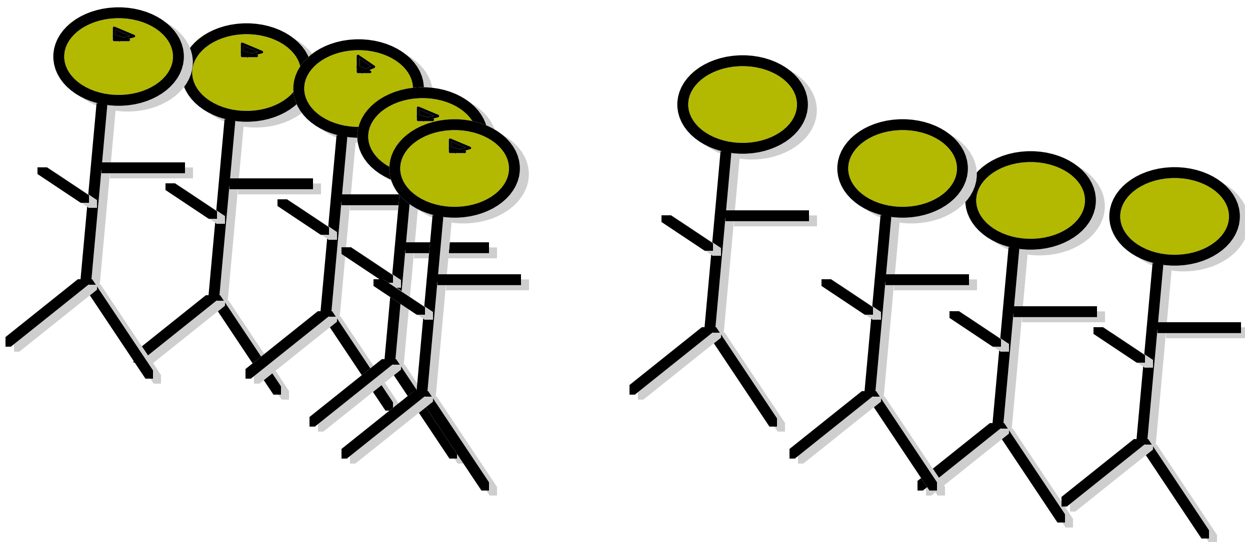
Case of 3 men



Case 3:
3 black dots

At t = 0, each of A, B, and C see that two other people have dots. So, A [resp. B, C] reasons as follows:
Either B and C have dots and I don’t, or we all have dots.
Now, if it were the case that I did not have a dot, this would reduce to case 2, and at time t = 2, B and C would speak.
When t = 2 passes, and B and C do not speak, A realizes that it is not case 2; that all three have dots.
B and C, reasoning similarly, come to the same conclusion.
Thus at t = 3, all speak.

N - Wise Man Problem



Assume N wise men. K have black dots on forehead.

Assuming - common knowledge of at least one black dot

all perfect reasoners

each round of reasoning takes 1 unit

Theorem: K men will speak at t = K

The crucial concepts: common knowledge

consequential closure

Three Wise Men --- Formulation in Logic

Language:

black(x) - X has a black dot on his forehead
speak(x,t) - X states the color on time T
t + 1 - successor of time T
0 - starting time
know(x,p,t) - X knows P at time T
know-whether(x,p,t) - X knows at T whether P holds

Axioms:

W1. know-whether(x,p,t) <==> [know(x,p,t) v ~know(x,p,t)]
(definition of know-whether: X knows whether P if he either knows P or he knows not P)

W2. speak(x,t) <==> know-whether(x,black(x),t)
(a wise man declares the color on his head iff he knows what it is)

Wise Men -- Logical formulation, cont.

W3. x < > y ==> know-whether(x,black(y),t)
(The wise men can see the color on everyone else's head)
W4. know-color(x,t) ==> speak(x,t)
(The wise men speak as soon as they figure it out)
W5. know-whether(y,speak(x,t),t+1)
(Each wise man knows what has been spoken)
W6. know(x,p,t) ==> know(x,p,t+1)
(The wise men do not forget what they know)
W7. know(x,black(w1) v black(w2) v black(w3), t)
(The wise men know that at least one of them has a black dot)
W8. if p is an instance of W1. -- W.8, then know(x,p,t)

Inference for 3 Wise Man Problem:

Lemma: If P is a theorem (can be inferred from 1 - 5, W.1 -- W.8, then know(x,p,t)

Proof: induction on length of inference (2,3, W.8)

Lemma 1.A ~black(w2) & ~black(w3) ==> speak(w2,0)

Proof: From W.7, w2 knows that
either w1, w2, or w3 has a black dot.
From W.3 and 1, w1 knows that neither w2 nor w3
has a black dot.
From 2 and 3, s2 knows that w1 has a black dot.
From W.2, w1 will speak.

Analogously
Lemma 1.B: ~black(w1) & ~black(w3) ==> speak(w2,0)
Lemma 1.C: ~black(w1) & ~black(w2) ==> speak(w3,0)

Inference for 3 wise men, cont.

Lemma 2.A:

~black(w3) ==> ∃ x speak(x,0) v speak(s1,1)

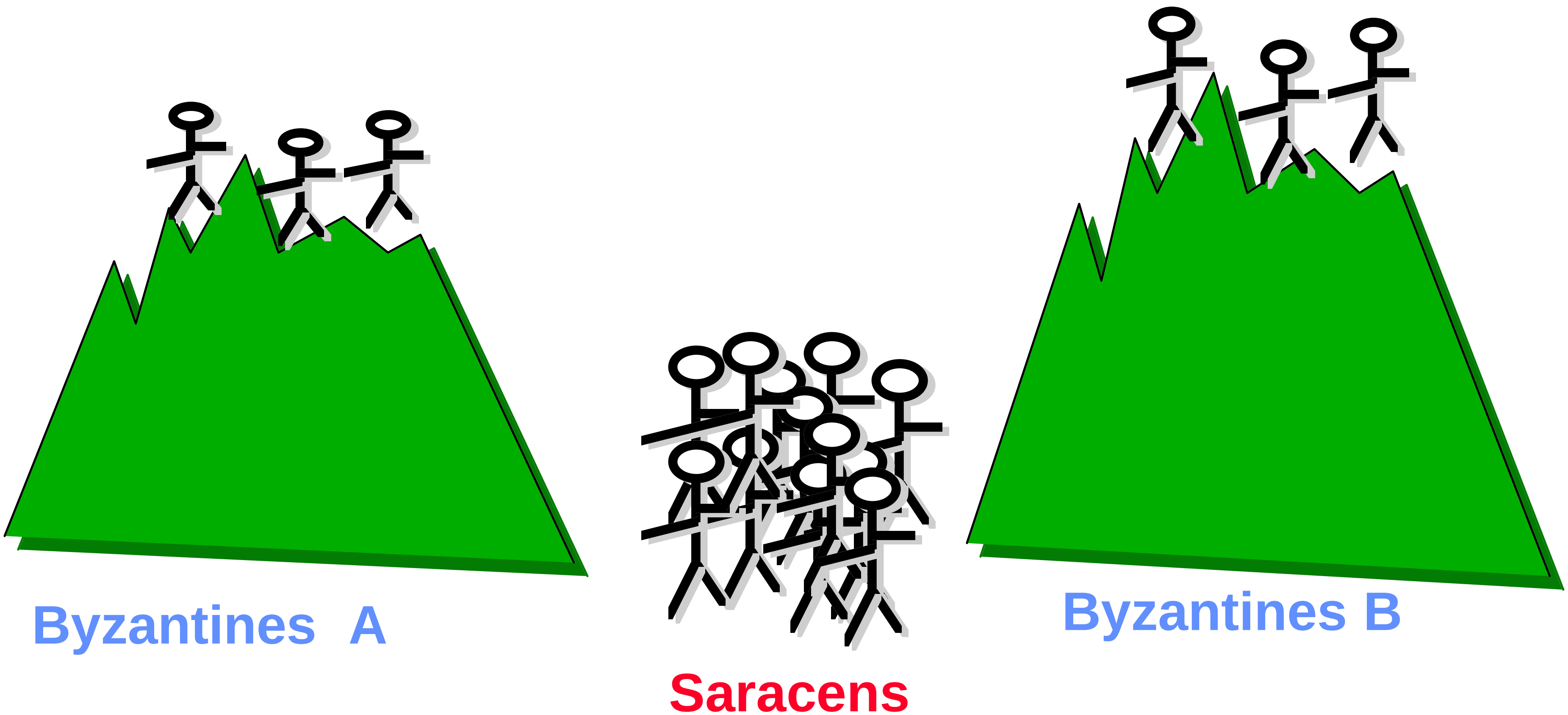
Proof:

From Lemma 1.A, if ~ black(w2) as well, then speak(w1,0);
From Lemma 1.B, if ~black(w1) as well, then speak(w2,0);
Suppose, then, that black(w1) & black(w2) & ~speak(w2,0).
From W.3, know(w1,black(w2),1) & know(w1,~black(w3),1)/
From W.5, know(w1, ~speak(w2,0),1).
By the lemma of necessitation, know(w1, Lemma 1.B, 1).
Using the contrapositive of Lemma 1.B and 2,
know(w1,black(w1),1).

And so on ....

Application:
Byzantine Problem

Byzantines must coordinate attack; otherwise, they'll be defeated



Byzantine Agreement:

t = 0: A sends B message: Attack at 6:00 AM (= M)
t = 1: B sends A message: received message
t = 2: A now knows that B received message,
but B doesn't know that A knows
A sends message to B that A received B's message
t = 3: B now knows that A knows that B knows M,
but A doesn't know that B now knows that
A knows that B knows that M
t = 2n: A sends message to B
A knows that B knows that ... that A knows (2n times)
(but not 2n+1 times)

Never reach common knowledge.

Thus, can't coordinate attack

Application:
Knowledge Preconditions for Actions and Plans

Interrelationship between Knowledge and Action

--- How does knowledge affect action ?
--- How do actions affect knowledge?

Focus of research:

--- agent wants to do an action
--- he doesn't know all that he needs to know
--- how can he get the action done anyway?

Knowledge Preconditions for Actions and Plans

Studied by McCarthy and Hayes;
Moore presented first concise solution.

Moore's theory based on

--- possible worlds theory of knowledge
--- situation calculus

situations = possible worlds

Moore:
Knowledge Preconditions Problem

(single agent case)

Basic idea:

You know how to do an action Dial(no(Suzanne))
iff you know executable procedure
[assumption: all agents know basic procedures]
iff you know what the parameters of the actions are

So you know how do perform Dial(no(Suzanne))
if you know what no(Suzanne) is

How do you know what the parameters of an action are?
You know what something is iff you know of a
rigid designator for that object

rigid designator = something that stays the same in
all possible worlds
(name, number, constant)

Know how to do Dial(no(Suzanne)) if know some number
equal to no(Suzanne)

Moore:
Knowledge Preconditions for Plans (single-agent)

Basic idea:
Knowledge Preconditions for Plans reduce to
Knowledge Preconditions for Actions

For example:

You know how to do sequence(act1, act2)
if you know how to do act1
and as a result of doing act1
you know how to do act2

Consider the plan
sequence(look_up_no(Suzanne), dial(no(Suzanne)))
You can perform the plan if you can do lookupno(Suzanne)
and you can then do dial(no(Suzanne))

Extension to Moore (Morgenstern) : Knowledge Preconditions for Multi-agent Plans

Example:

Pierre wants to drive to Lyon
He doesn't know the directions
He does know the number of the automobile club
How can he plan to drive to Lyon?

We want to show that Pierre can execute the following plan:

sequence(dial(Pierre, no(auto_club)),
ask(Pierre, officer(auto_club), directions(Lyon))),
tell(officer(auto_club), Pierre, directions(Lyon))))

What's needed:

ability to reason about one's own ability to do actions,
ability to <u>predict</u> other people's actions

Knowledge Preconditions for Multi-agent Plans

In general:

- --- need to know that you'll <u>be able</u> to do your part of the plan when it comes up
- --- need to predict that other agents <u>will</u> do their parts of the plan at the proper time

How to predict other agents' actions:

- --- consider interactions between knowledge, goals, and actions (BDI)
- --- agents typically act in their own interests
- --- will often accede to a request if there are no conflicting goals

Note: Importance of communication (establishing goals, relaying information)

Knowledge Preconditions for Multi-agent Plans

Consider Pierre's plan:

sequence(dial(Pierre, no(auto_club)),
ask(Pierre, officer(auto_club), directions(Lyon))),
tell(officer(auto_club), Pierre, directions(Lyon))))

Pierre can execute the plan if :

- **--- he knows the number of the auto club**
- **--- he knows how to ask for directions**
- **--- he can predict that once asked, the officer of the auto club will give him directions**
- **--- he knows that once the officer gives him directions, he will know how to get to Lyon**

works because officer is cooperative and knowledgeable, and knows how to give directions

Application: Speech Acts (Grice)

"Dear Sir,
Mr. X has an excellent command of English
and always comes to class"

Why will this doom Mr. X?

Cooperative Principle
Plus Maxims of Conversation:

Say as much as is needed, no more, no less

Since there is common knowledge of these maxims,
and Mr. X's teacher must know more about him,
his failure to say more must mean that there's nothing
more that is good to say.

Based on common knowledge of convention,
of maxims of conversation

Beyond Modal Logic

Disadvantages of Modal Logic:

1. **Inexpressive**
   Can't quantify over propositions
   Can't say, e.g.
   John knows something that Bill doesn't know

2. **Non-intuitive semantics**
   --- state-based
   --- possible worlds

3. **Undesirable consequences of semantics**
   --- necessary truths
   --- consequential closure

Representing Knowledge

- --- Modal Logics
  - --Syntax
  - --Semantics
    - state-based definition, possible worlds
  - --Extensions
    - quantification, time
  - --Applications
    - 3 Wise Men, Byzantine Agreement
  - --Problems
- → --- Syntactic Logics
  - --Syntax and Semantics
  - -- Advantages and Disadvantages: Paradox
  - -- Resolution to Paradoxes
- --- Additional Issues
  - --Dropping Consequential Closure
  - --Nonmonotonic Logics

Alternative to modal logic:
Syntactic Logic

standard predicate logic
can't say: Know(John, Frog(Kermit))
since Know is a predicate and
can't range over the sentence Frog(Kermit)

introduce an invertible map from sentences (wffs) to terms

[Godel mapping maps each wff onto an integer]

denote range of mapping function with quotation marks

Know(John, "Frog(Kermit)")

Tarskian semantics

Features of Syntactic Logics

Advantages:

1. Expressivity
   $\exists$x (Know(Bill,x) & ~ Know(John,x))
   forall x (Concerns(x,Radiology) ==> Know(Helene,x))

2. No need for necessitation, consequential closure

Disadvantages:

1. Messiness - quasi-quotation

2. Paradox

Tarskian Semantics considered advantage by some,
disadvantage by others

Messiness of Syntactic Logic

Saying simple things gets ugly.
Can’t just say [principle of positive introspection]:

forall a,x
Know(a,x) ==> Know(a,"Know(a,x)")

This would imply that
Know(John,"Frog(Kermit)") ==>
Know(John, "Know(a,x)")
which is not what we want and meaningless, too!

Need quasi-quotes, which allow us to substitute value of quoted string:
forall a,x
Know(a,x) ==> Know(a,"Know(@a,!p!)")

Paradox

akin to Liar Paradox ---

Everything I say is a lie

P iff ~True("!P!")

Knower Paradox

P iff Know(a,"~!P!")

P is true iff a knows that it is false

Comes from unrestricted use of quotation

Arises in many reasonable languages

Surprise Test Paradox:

You'll have a test someday next week,

but you won't know which

Pravda: Everything the Times says is a lie

New York Times: The Pravda sometimes lies

Whether or not these sentences are paradoxical depends on empirical facts about the world

e.g., has NY Times said one true fact?

and not only on structure of sentences

Resolutions to Paradox

--- reduce expressivity
(Tarski, Konolige)
no self-reflexive sentences

--- Three-valued logic: true, false, neither
(Kripke, Gupta, Herzberger, Barwise, Morgenstern)

--- Different semantics for Know, True
(Perlis)

No free lunch: drawbacks for each

Representing Knowledge

- --- Modal Logics
  - --Syntax
  - --Semantics
    - state-based definition, possible worlds
  - --Extensions
    - quantification, time
  - --Applications
    - 3 Wise Men, Byzantine Agreement
  - --Problems
- --- Syntactic Logics
  - --Syntax and Semantics
  - -- Advantages and Disadvantages: Paradox
  - -- Resolution to Paradoxes
- → --- Additional Issues
  - --Dropping Consequential Closure
  - --Nonmonotonic Logics

Dropping Consequential Closure

Until now, agents have been assumed to be perfect reasoners:

Know(P) & Know(P ==> Q) ==> Know(Q)

[consequential closure]

Clearly false:

--- agents make mistakes

--- if true, all agents should know whether ~~Fermat's last theorem~~ Riemann's Conjecture is true, but no-one does

--- agents have inconsistent beliefs but don't believe everything

---doesn't take into account time, resources, focus, etc.

Three types of incompleteness

(Konolige)

--- resource incompleteness

(running out of time to take a test)

--- fundamental logical incompleteness

(not knowing how to do integrals)

--- relevance incompleteness

(not knowing which facts to include)

Building a System without Consequential Closure

Issues:

- how can we drop consequential closure
- what can we replace it with

Dropping Consequential Closure

Difficult because it is built into possible worlds semantics, state-based definition of knowledge
<u>**part of every standard modal logic**</u>

Ways to proceed:

1. **Drop modal logic - go to syntactic logic**
   (Konolige, Haas, Elgot-Drapkin)

2. **Make a distinction between explicit and implicit knowledge**
   Implicit knowledge = standard concept of knowledge
   Introduce concept of awareness
   Explicit knowledge = awareness plus implicit knowledge
   Consequential closure for implicit knowledge only
   (Levesque, Halpern and Fagin)

Replacing Consequential Closure

<u>Problem</u>: If agents don't do perfect reasoning,
just what do they do?

<u>Proposal</u>: Limit reasoning rules in some way

--- restricted set of inference rules
- **e.g., math student might not know integrals**
  robot might not know path-finding algorithm

--- restricted resources
- **--- specifically time, number of steps**
- **--- clear that agents only have limited time to reason**
  (Elgot-Drapkin, Kraus, Nirkhe, and Perlis)

--- "need-to-know"
- **--- idea is that our reasoning is goal-oriented**
- **--- plan to reason**
  (Haas)

Problems with Alternatives to Consequential Closure:

--- restricted rules seem arbitrary,
counter-intuitive

--- can always find counterexamples

--- limited resources, e.g., limited number of steps:
what makes *n* the cutoff as opposed to *n+1* ?
If I know p, and q is *n* steps away, I'll know q.
But then won't I know r if r is 1 step away from q?

--- restricted reasoning rules:
logicians are thoroughly familiar with rules of logic,
and still aren't perfect reasoners.

--- "need to know" - agents seem to chain forward, too.

Nonmonotonic Logic

Commonsense reasoning

often draws conclusions on basis of partial information

- **Birds typically fly**
  Tweety is a bird
  ____
  Tweety flies

Counterexamples:
penguins, broken wings

- **If I turn the key in the ignition,**
  the car will start

Counterexamples:
dead battery, bad starter

Really:
If I turn the key in the ignition, and the starter works, and the battery works, and there's gas in the car and there's no potato in the tailpipe
.... and ... then the car will start

- **If I had an older brother, I'd know it**
  I don't know I have an older brother,
  so I infer that I don't have one

Counterexamples:
General Hospital,
Bill Clinton

Such reasoning (Tweety flying, my car starting, my lack of an older brother) can't be carried out in classical logic

Classical logic --- Drawing permanent conclusions based on complete information

What we need --- Drawing conclusions on basis of *incomplete* information --- later retract **Nonmonotonic Logic**

Classical Logic --- monotonic in set of assumptions
the more assumptions, the more conclusions

Nonmonotonic Logic --- nonmonotonic in set of assumptions
as you add assumptions, you may have to retract conclusions

$$\frac{\text{Bird(Tweety)}}{\text{Fly(Tweety)}}$$

but

Bird(Tweety), Penguin(Tweety) retracts Fly(Tweety)

How can we capture nonmonotonic reasoning?

1. **Default Logic (Reiter)**
   based on *default* rules:

   $$\frac{Bird(x) : Fly(x)}{Fly(x)}$$

   new type of inference rule

2. **NML (McDermott and Doyle)**
   based on idea of consistency

   Bird(x) & M(Fly(x)) ==> Fly(x)

   rules within the logic

3. **Circumscription (McCarthy)**
   restricts set of objects;
   in particular, abnormal objects

   Bird(x) and ~ ab(x) ==> Fly(x)

4. **Autoepistemic Logic (Moore)**
   if x is true, I'd know x
   (where x is an "important" statement)

   Allows inference from x not known to ~ x

Autoepistemic Logic (Moore)

Commonsense Reasoning:
based on one's beliefs --- or lack of them

e.g. how do I know I don't have an older brother?
If I had an older brother, I'd know about it

P == " I have an older brother "
P ==> Know(P)

Get from: P is not in my knowledge base
to: I don't know P: ~Know(P)

Note: nonmonotonic

If I later find out that a parent previously married and had children, I'd retract this conclusion
Nonmonotonic because *indexical*

Autoepistemic Logic --- how it works

based on logic of belief (L === belief)

set of formulas T that represent beliefs of reasoning agents should satisfy:

1. if P_1 ... P_n in T, and P_1 ... P_n |- Q, then Q in T (consequential closure)
2. if P in T, then LP in T (positive introspection)
3. if P not in T, then ~LP in T ("negative introspection")

Theories obeying 1. - 3. are stable.
If a stable theory is consistent, you also get:

4. if LP in T, then P in T
5. if ~LP in T, the P not in T

Def: T is grounded in set of premises A iff every formula of T is included in the tautological consequences of
A U {LP | P in T} U {~LP | P not in T}

Theorem: An AE theory T is sound w.r.t. set of premises A iff T is grounded in A

Autoepistemic Logic --- how it works

The older brother example:

P = "I have an older brother"
A = {P ==> LP}

By rule 1., P ==> LP in T.
Also, ~LP ==> ~P in T

Now P not in T. So by 3., ~LP in T.
So by 1., ~P in T, and by 2., L~P in T.

Result: You know that you do not have an older brother.
You have reasoned from your own lack of knowledge

Note: stable set semantics gives us weak S5:
preferred logic of belief

OUTLINE

- What are epistemic logics?
- Why are they interesting ?
  - --- AI Applications
- Representing knowledge
  - --- Modal Logics
    - --Syntax
    - --Semantics
      - state-based definition, possible worlds
    - --Extensions
      - quantification, time
    - --Applications
      - 3 Wise Men, Byzantine Agreement
    - --Problems
  - --- Syntactic Logics
    - --Syntax and Semantics
    - -- Advantages and Disadvantages: Paradox
    - -- Resolution to Paradoxes
  - --- Additional Issues
    - --Dropping Consequential Closure
    - --Nonmonotonic Logics
- Using Representations
  - --- automated reasoning
  - --- toy programs
- Summary

Issue: Construct Theorem Prover for Epistemic Logics

Problem: Complexity

--- very inefficient

--- theorem prover “driven” by axioms on knowledge

Idea: Circumvent

Approaches:

--- multiple contexts

--- direct representation, procedural attachment

--- inference with possible worlds

--- vivid reasoning

Multiple Contexts

Basic Idea:

For each state of knowledge, or state of imbedded knowledge, create a separate context of "object-level" facts.

Inference within a context uses "ordinary" automated reasoning.

Inference from one context to the next uses special-purpose inference.

Example of Multiple Contexts

Given:

In S1, A knows p.
In S1, B knows that p ==> q.
In S1, A knows that B knows that p ==> q.
A tells p to B during [S1,S2]

Initialize:

Context S1A (what A knows in S1): {p}
Context S1B (what B knows in S1): {p==>q}
Context S1AB (what A knows that B knows in S1): {p==>q}

Create corresponding contexts for time S2:

Context S2A = { ... }
Context S2B = { ... }
Context S2AB = { .... }

Example of multiple contexts, continued

Frame Inferences:

Context S2A : { p ... } (A still knows p)
Context S2B: { p ==> q ... } (B still knows p ==> q}
Context S2AB: { p ==> q ...} (A knows that B still knows that p ==> q)

Inferences associated with "tell" :

If X tells P to Y during [S1,S2] then in S2 Y knows P and X knows that Y knows P

Context S2B: { p ==> q, p} (B now knows P)
Context S2AB: { p ==> q, p) (A now knows that B knows p)

Modus Ponens within context:

Context S2B: { p ==> q, p, q} (B infers Q)
Context S2AB: { p ==> q, p, q} (A infers that B infers Q}

Implementation Remark:

Since different contexts are likely to share a lot of knowledge,inference will be more efficient if facts are labelled by context,as in CONNIVER and ATMS, rather than copying the whole knowledge base.

Limitations, Issues:

--- Limited expressivity:

Difficult to express

A knows p or A knows q

A knows who the president of the Congo is

The man with the white hat knows p

A will know p when the bell rings (partial spec. of time)

A knows that B does not know p

--- When do you generate new contexts?

--- What are the cross-context inference rules?

--- How is the closed-world assumption to be applied?

If S1A does not contain q, should we conclude

A knows in S1 that q is false? or

A does not know in S1 whether q is true or false?

If S1AB does not contain q, should we conclude

In S1, A knows that B knows that q is false? or

In S1, A knows that B does not know whether q? or

In S1, A does not know whether B knows q?

Explicit Syntactic Representation

Express arbitrary sentences about knowledge in syntactic representation
Use first-order theorem prover incorporating theory of strings

String operations implemented partly or wholly by procedural attachment
Axioms of knowledge implemented largely by special-purpose inference rules

Example:
Axiom 1: Joe knows that a person always knows whether he's hungry
Know(Joe,"forall x know-whether(x,"hungry(@x)")")
Axiom 2: Joe knows that Fred is hungry
Know(Joe,"hungry(Fred)")
To Prove: Joe knows that Fred knows that he is hungry
Know(Joe,"Know(Fred,"hungry(Fred)")")

Proof:

Applying the inference rule R1, consequential closure,
and the definition
know-whether(A,q) <==> Know(A,q) v Know(A, ~q)
to Axiom 1 gives:

3. Know(Joe,"forall x Know(x, "hungry(@x)" v
Know(x,"~hungry(@x)")")

Applying R1 plus the axiom of veridicality plus the
propositional axiom (P ==> Q) ==> (P ==> (P & Q)) to 3. gives

4. Know(Joe,
"forall x (hungry(x) & Know(x,"hungry(@x)")) v
~ hungry(x) & Know(x,~hungry(@x)"))")

Applying R1 to 2. and 4. gives

5. Know(Joe,"hungry(Fred) & Know(Fred,"hungry(Fred)")")

Applying R1 to 5 gives

6. Know(Joe,"Know(Fred,"hungry(Fred)")")

Problem: Immense search space. How to control search?

Inference with Possible Worlds

Technique: Translate all statements into first-order language of possible worlds.
Apply first-order theorem proving techniques.

Example:
Axiom 1: Joe knows that a person always knows if he's hungry

```
forall W1
  K(Joe,w0,W1) ==>
     forall x (( forall W2 K(X,W1,W2) ==> hungry(X,W2)) or
                     forall W3 K(X,W1,W3) ==> ~ hungry(X,W3)))
```

Axiom 2: Joe knows that Fred is hungry

```
forall W4 K(Joe,w0,W4) ==> hungry(Fred,W4)
```

To prove: Joe knows that Fred knows that he is hungry

```
forall W5 K(Joe,w),W5) ==>
    forall W6 K(Fred,W5,W6) ==> hungry(Fred,W6)
```

Skolemizing:

1. ~K(Joe,w0,W1) v ~K(X,W1,W2) v hungry(X,W2)
   v ~K(X,W1,W3)) v ~hungry(X,W3)

2. ~K(Joe,s0,S4) v hungry(fred,W4)

Negation of 3:

3A. K(Joe,w0,w5) {w5 and w6 are Skolem constants}
3B. K(fred,w5,w6)
3C. ~hungry(Fred,w6)

Skolemization of reflexivity:

4. K(X,W,W)

The resolution proof is then immediate

Comparison to syntactic representation

--- Much more controlled inference path

--- Somewhat less expressive language

--- Substantially less intuitive representation
and proof structure

Vivid Representation (Grove and Halpern, '92)

Construct an actual model of the theory as a set of possible worlds (or a collection of models).

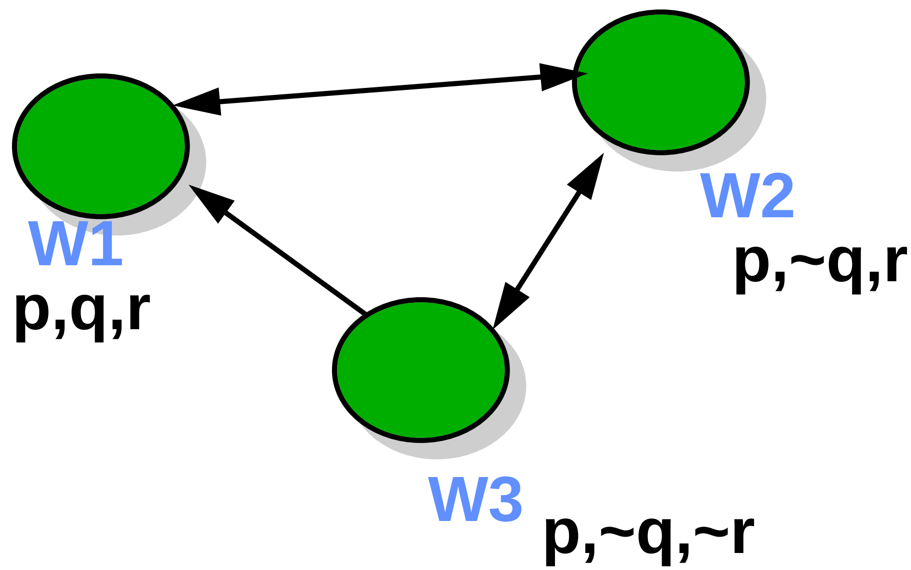
What is true in the model(s) may be a consequence of the theory.

Example:

Given:

know(a,p)
know(a, ((p & q) ==> r))
~ know(a,r)

S5 logic of knowledge



To show know(a,~r ==> ~q)) check that ~r ==> ~q holds in every accessible world.
To show ~know(a,q) show that q is false in some accessible world.

Problem: Distinguish between the consequences of the theory and random features of the model
(e.g., ~know(a, q <==> r) holds because q<==> r false in W2. But it's not a consequence of the theory.

OUTLINE

- What are epistemic logics?
- Why are they interesting ?
  - --- AI Applications
- Representing knowledge
  - --- Modal Logics
    - --Syntax
    - --Semantics
      - state-based definition, possible worlds
    - --Extensions
      - quantification, time
    - --Applications
      - 3 Wise Men, Byzantine Agreement
    - --Problems
  - --- Syntactic Logics
    - --Syntax and Semantics
    - -- Advantages and Disadvantages: Paradox
    - -- Resolution to Paradoxes
  - --- Additional Issues
    - --Dropping Consequential Closure
    - --Nonmonotonic Logics
- Using Representations
  - --- automated reasoning
  - --- toy programs ←
- Summary

Implementations

--- Restricted to toy programs

--- Planning

PAWTUCKET

UWL

UWL (Etzioni et. al.)
Modified TWEAK planner, find out variable bindings

Goal: (satisfy (color chair ?c)
(satisfy (color table ?c)
(handsoff (color table ?tc))

Make the chair the same color as the table,
but not by changing the color of the table

Actions: (SENSE-COLOR ?object ! color)
Effects: ((observe ?object !color))

Action: (GET-PAINT ? color)
Effects: (have-color ?color)

Name: (PAINT ?obj ?color)
Preconds: (satisfy ((have-color ?color)))
Effects(cause ((color ?obj ?color)))

UWL Plan:

(sense-color table !color)
(get-paint !color)
(paint chair ! color)

Summary

--- Logics of knowledge and belief are needed for many AI applications

--- planning, speech acts, distributed systems

--- Modal logics, Syntactic logics can be used to represent knowledge

--- Many extensions needed for commonsense reasoning:

--- time, default reasoning

--- Much future work ahead

--- concrete applications, multiple agents, consequential closure

Pawtucket (Davis, unpublished)

Situation:

John knows that Bill knows Mary's phone no
John knows that phone1 is a telephone

Wanted:

A plan for John to call Mary's no.

Causal rules:

A way to call x is to dial x's no. on the phonme
The preconditions of B telling P to A in S are
that A and B are at the same place
and that B knows P is true

Plan:

do(john,request(bill,do(bill, tell(john,a_q(n))))),
do(bill,tell(john,a_q(n))),
do(john, dial(a_q(n), phone1))