Solving the Yale Shooting Problem

- The Yale Shooting Problem arises when a theory of temporal reasoning is integrated with nonmonotonic logic.

- The problem is a special case of the multiple extension problem. In the Nixon diamond, we had multiple extensions but they seemed reasonable. However, here some of the multiple extensions are unintuitive.

- To get rid of unwanted multiple extensions, we can either
  - ---- change the underlying theory so that the unwanted extensions don't show up
  - ---- place a new preference relation on models that will select only the desired models
  - ---- or some combination of the above

- Most solutions to the Yale Shooting Problem try to solve the problem through a better understanding of causation

Classes of solutions to the Yale Shooting Problem

- Chronological ignorance: Prefer those models where unexpected facts happen as late as possible (Shoham, Kautz: 1986)

- Explicit causation: Circumscribe the cause predicate. Prevents actions from causing unexpected events (Lifschitz, 1987)

- Prevent unexpected actions from happening.
  --- Prefer model in which “unmotivated actions” are minimal (Morgenstern, Stein, 1988)
  --- Circumscribe some sort of occurs predicate

- Probabilistic approaches (Pearl; 1988)

- Combinations of above strategies. Shanahan, 1997 is (currently) widely believed to have given a solution.

The Yale Shooting Problem

The Yale Shooting Problem, discovered by Hanks & McDermott in 1985/1986, shows that a straighforward nonmonotonic solution to the frame problem won't work.

Suppose we have the following theory: (With ab circumscribed)

Holds(s,fluent) & ~ab(fluent,action,s) ==> Holds(Result(s,action),fluent) ***Inertia***

Holds(S0,Alive) ***Fred (the turkey) is alive at S0***
Holds(Result(s,load),loaded) ***When you load a gun, it's loaded as a result***
Holds(s,loaded) ==> ab(alive,shoot,s) & Holds(Result(s,shoot),~alive) ***If you shoot a loaded gun at a turkey, it's dead as a result***

Also suppose we define situations as follows:

S1=Result(S0,load)
S2=Result(S1,wait)
S3=Result(S2,shoot)

The question is: is Fred dead or alive in S3?
We expect Fred to be dead, because he's been shot with a loaded gun.
But actually, it turns out that there's some ambiguity.

The Yale Shooting Problem: A Case of Multiple Extensions

Recall the Nixon diamond: Nixon is a Republican and a Quaker; Quakers are typically pacifists; Republicans are typically non-pacifists. We can't conclude anything about Nixon because of the ambiguity. A similar thing happens in the Yale Shooting Problem.

We have two minimal models (where ab is circumscribed)

M1: (the expected model)	M2: (the unexpected model)
S1=Result(S0,load)	S1=Result(S0,load)
S2=Result(S1,wait)	S2=Result(S1,wait)
S3=Result(S2,shoot)	S3=Result(S2,shoot)
Holds(S0,alive)	Holds(S0,alive)
Holds(S1,loaded)	Holds(S1,loaded)
Holds(S2,loaded)	<u>ab(loaded,wait,S1)</u>
<u>ab(alive,shoot,S2)</u>	Holds(S2, ~loaded)
Holds(S3,~alive)	Holds(S3,alive)

In M1, everything goes as planned. In M2, the abnormality happens earlier; somehow, the wait is abnormal to the gun being loaded. Which means that the shoot in S2 isn't abnormal for the turkey being alive.
But note that both models have exactly one ab statement!
Models aren't comparable; neither is a subset of the other; <u>they're both minimal!</u>

It's important to realize that both M1 and M2 are minimal.

M1: (the expected model)

S1=Result(S0,load)
S2=Result(S1,wait)
S3=Result(S2,shoot)
Holds(S0,alive)

Holds(S1,loaded)
Holds(S2,loaded)
<u>ab(alive,shoot,S2)</u>
Holds(S3,~alive)

M2: (the unexpected model)

S1=Result(S0,load)
S2=Result(S1,wait)
S3=Result(S2,shoot)
Holds(S0,alive)

Holds(S1,loaded)
<u>ab(loaded,wait,S1)</u>
Holds(S2, ~loaded)
Holds(S3,alive)

This example is so small, that it's hard to see (we only have 2 fluents), but we can imagine a 3rd fluent, say *gobbling*

Then a model like M3 would not be minimal: M1 < M3

M3: (a non-minimal unexpected model)

S1=Result(S0,load)
<u>ab(gobbling, load,S0)</u>
S2=Result(S1,wait)
S3=Result(S2,shoot)
Holds(S0,alive)

Holds(S1,loaded)
Holds(S2,loaded)
<u>ab(alive,shoot,S2)</u>
Holds(S3,~alive)

In this model, turkey stops gobbling after the load; truly unexpected.

Solving the YSP: Chronological Ignorance (Shoham; Kautz; 1986)

The idea: In the unexpected model, an abnormality occurs earlier than in the expected model.

M1: (the expected model)

S1=Result(S0,load)
S2=Result(S1,wait)
S3=Result(S2,shoot)
Holds(S0,alive)

Holds(S1,loaded)
Holds(S2,loaded)
<u>ab(alive,shoot,S2)</u>
Holds(S3,~alive)

abnormality happens at S2

M2: (the unexpected model)

S1=Result(S0,load)
S2=Result(S1,wait)
S3=Result(S2,shoot)
Holds(S0,alive)

Holds(S1,loaded)
<u>ab(loaded,wait,S1)</u>
Holds(S2, ~loaded)
Holds(S3,alive)

abnormality happens at S1

We prefer M1 because the abnormality happens later!
(Don't trouble trouble till trouble troubles you)

Formalizing chronological ignorance:

Define a model M1 as being <u>chronologically more ignorant</u> than M2
if M1 and M2 are the same up to some point ....
but then, a new fact is introduced in M2

Formally,
M1 is <u>chronologically more ignorant</u> than M2
if there exists a time point t0 such that:

Note: t0 is the point where things change

1. **for any (base) sentence P whose least time point precedes t0,**
   if M1 |= P then M2 |= P

2. **there is some (base) sentence P whose least time point is t0**
   such that M2 |= t0 but M1 $\not\models$ t0

M1 is a chronologically maximally ignorant model of P if
M |= P, and there is no M’ such that M’ is chronologically more ignorant
than M and M’ |= P

Evaluating chronological ignorance

- **Chronological ignorance solves the Yale Shooting Problem. M1 is preferred to M2. We throw out M2. The turkey is dead.**

- **Model preference nice, intuitive, but not really necessary: Kautz achieves much the same thing through circumscription (clips)**

- **Chronological ignorance doesn't really do what we want!**
  It's especially bad at backward reasoning.
  If something goes wrong, must reason that it happened at last possible minute.
  <u>Stolen car problem:</u> Park car at 9 am; return at 5 pm; car is gone.
  Did car disappear at 4:50 pm???
  CI works for prediction (forward temporal projection),
  but not for explanation (backward temporal projection)

- **Come to think of it, the intuitions behind chronological ignorance were never that clear anyway.**

- **So the next trend was toward a solution that captured the correct intuitions**

Adding, and circumscribing a causes predicate (Lifschitz, 1987)

Idea: add an explicit cause predicate that describes how actions change fluents. Since you don’t want fluents to change unless they “have to”, circumscribe the cause predicate.

In particular, since the wait action doesn’t cause anything, it can’t cause the gun to become unloaded.

Formally:

causes(load, loaded, true)
causes(shoot, loaded, false)
causes(shoot, alive, false)

precond(loaded, shoot)

success(a,s) <==> forall f (Precond(f,a) ==> holds(s,f))
affects(a,s) <==> sucess(a,s) & exists v(causes(a,f,v)
success(a,s) & causes(a,f,v) ==> (holds(result(s,a),f) <==> v= true)
~affects(a,f,s) ==> (holds(result(s,a),f) <==> holds(s,f))

We combine this theory of causation

causes(load, loaded, true)
causes(shoot, loaded, false)
causes(shoot, alive, false)

precond(loaded, shoot)

success(a,s) <==> forall f (Precond(f,a) ==> holds(s,f))
affects(a,s) <==> sucess(a,s) & exists v(causes(a,f,v)
success(a,s) & causes(a,f,v) ==> (holds(result(s,a),f) <==> v= true)
~affects(a,f,s) ==> (holds(result(s,a),f) <==> holds(s,f))

with

holds(S0, alive)
~holds(S0, loaded

And we then circumscribe causes & precond

Solves both the frame problem & the qualification problem without running into Yale Shooting Problem !!

M1

S1=Result(S0,load)
S2=Result(S1,wait)
S3=Result(S2,shoot)
Holds(S0,alive)

causes(load, loaded, true)
causes(shoot, loaded, false)
causes(shoot, alive, false)

success(load,S0)
affects(load, loaded, S0)
Holds(S1, loaded)
success(wait,S1)
Holds(S2, loaded)
success(shoot, S2)
affects(shoot, alive, S2)
Holds(S3, ~alive)

M2

S1=Result(S0,load)
S2=Result(S1,wait)
S3=Result(S2,shoot)
Holds(S0,alive)

causes(load, loaded, true)
causes(shoot, loaded, false)
causes(shoot, alive, false)

success(load,S0)
affects(load, loaded, S0)
Holds(S1, loaded)
success(wait,S1)
<u>causes(wait, loaded, false)</u>
<u>affects(wait, laoded, S1)</u>
Holds(S2, ~ loaded)
~success(shoot, S2)
~affects(shoot, alive, S2)
Holds(S3, alive)

The only way that the gun can become unloaded between S1 and S2 is if <u>causes(wait, loaded, false)</u> . But such models wouldn't be minimal for causes.

Evauating Lifschitz's solution

- **Solves the Yale Shooting Problem. (**And qualification problem as a bonus!)
- **Okay for backward reasoning. Doesn't run into Stolen Car Problem**
- **Still, there are anomalous results.**
  For example, if it does turn out that the turkey survives the shoot,
  the theory is now saddled with the fact <u>causes(wait,loaded,false)</u>
  Now wait will *always* causes waiting to unload the gun.
  That is, one time anomalous occurrences can't happen!!
- **So how did this happen? How does this intuitive theory entail such anomalous results? (Anomalous results usually means the underlying intuitions aren't so great after all.)**
- **There are really two intuitions we have to formalize.**
  1. **Actions don't cause the world to change in strange ways**
  2. **Unexpected actions don't happen**
- **Lifschitz formalizes 1. but not 2.**

Motivated Action Theory (Morgenstern, Stein; 1988, 1994)

Idea: prefer models if they contain as few unmotivated actions as possible.
The gun never got unloaded because of the wait; it must have been the case than an unload somehow happened. We examine models for these strange, unmotivated actions, and exclude them.

Formally,

- causal rules are of the form A & B ==> C, where A is the triggering action, B gives the preconditions of the action, and C gives the effect.
  C may itself be an action; we allow causal chains
- causal rules always go forward in time; we assume unit time
- explanation closure axioms give persistence of fluents

- a statement p is strongly motivated if it must be true; that is, true in all models
- a statement C is weakly motivated in a model M if there is a causal rule of the form A & B == C, A is strongly motivated and B is motivated in M
- a statement is motivated in M
  if strongly or weakly or semi- or existentially motivated in M
- M1 < M2 if for some unmotivated p, M2 |= p, M1 $\not\models$ p, and not M2 < M1

How this works in the Yale Shooting Problem:

Holds(S0,alive)
Occurs(S0,load)
Occurs(S1, wait)
Occurs(S2, shoot)

Occurs(t,load) ==> Holds(t+1, loaded)
Occurs(t,unload) ==> Holds(t+1,~loaded)
Occurs(t,shoot) ==> Holds(t+1,~loaded)
Occurs(t,shoot) & Holds(t,loaded) ==> Holds(t+1, ~alive)

Holds(t,loaded) & ~occurs(t,shoot) & ~occurs(t,unload) ==> Holds(t+1,loaded)
Holds(t,alive) & ~(occurs(t,shoot) and holds(t,loaded)) ==> Holds(t+1,alive)

How this works in the Yale Shooting Problem:

Holds(S0,alive)
Occurs(S0,load)
Occurs(S1, wait)
Occurs(S2, shoot)

Occurs(t,load) ==> Holds(t+1, loaded)
Occurs(t,unload) ==> Holds(t+1,~loaded)
Occurs(t,shoot) ==> Holds(t+1,~loaded)
Occurs(t,shoot) & Holds(t,loaded) ==> Holds(t+1, ~alive)

Holds(t,loaded) & ~occurs(t,shoot) & ~occurs(t,unload) ==> Holds(t+1,loaded)
Holds(t,alive) & ~(occurs(t,shoot) and holds(t,loaded)) ==> Holds(t+1,alive)

M1:

Holds(S0,alive)
Occurs(S0,load)
Holds(S1,loaded)
Occurs(S1, wait)
Holds(S2,loaded)
Occurs(S2, shoot)
Holds(S3, ~alive)

unmotivated action!

M2:

Holds(S0,alive)
Occurs(S0,load)
Holds(S1,loaded)
Occurs(S1, wait)
Occurs(S1,unload)
Holds(S2, ~loaded)
Occurs(S2, shoot)
Holds(S3, alive)

Since M2 has an unmotivated action, we prefer M1 to M2

Evaluating Motivated Action Theory

- Solves the Yale Shooting Problem
- Works for forward and backward reasoning, no anomalous side effects
- Allows causal chains (not the same as minimizing occurrence)
- But: syntactic form of causal rules
- But: need explanation closure axioms; no general principle of inertia
- Need to combine this theory with part of Lifschitz's

Shanahan's theory

- the most comprehensive theory of causation: contains predicates initiates, terminates, happens (occurs), holds

- circumscribe happens, initiates, terminates

- still uses explanation closure axioms

Probabilistic Solutions to the Yale Shooting Problem

- **Motivation: Doesn't this just point out how troublesome nonmonotonic logic is? A probabilistic logic should be able to easily dispense with all this nonsense.**

- **But probabilistic systems still have to deal with all the representational issues that plague nonmonotonic systems.**

- **For example .... Pearl's (1988) probabilistic solution to the YSP:**

LD0: gun is loaded at time t0; LD1: gun is laoded at time t1;
SH1; shoot occurs at time t1;
AL1; turkey is alive at time t1; AL2: turkey is alive at time t2;

P(LD1 | LD0) == high = 1 - epsilon1
P(AL2 | AL1) == high = 1 - epsilon2
P(AL2 | AL1, SH1, LD1) == low == epsilon3
P(AL2 | AL1, SH1, ~LD1) == high == 1 - epsilon4

This isn't enough of course; need to specify complete probabilistic model ,,, but consider the problems:

Problems with Pearl's probabilistic solution to the YSP

- persistence rules need
- worse than before: need to consider all possible combinations of fluents, actions for each fluent
- persistence rules are specific to each time point in history!
- need same set of causal rules as before
- so if we have to specify persistence rules and causal rules as before, what have we gained?
- plus, there are the general problems of probabilistic reasoning: difficulty of assigning probabilities, and reasoning with them
- the moral: probabilistic reasoning still needs a good underlying model/ theory of causation/ knowledge representation