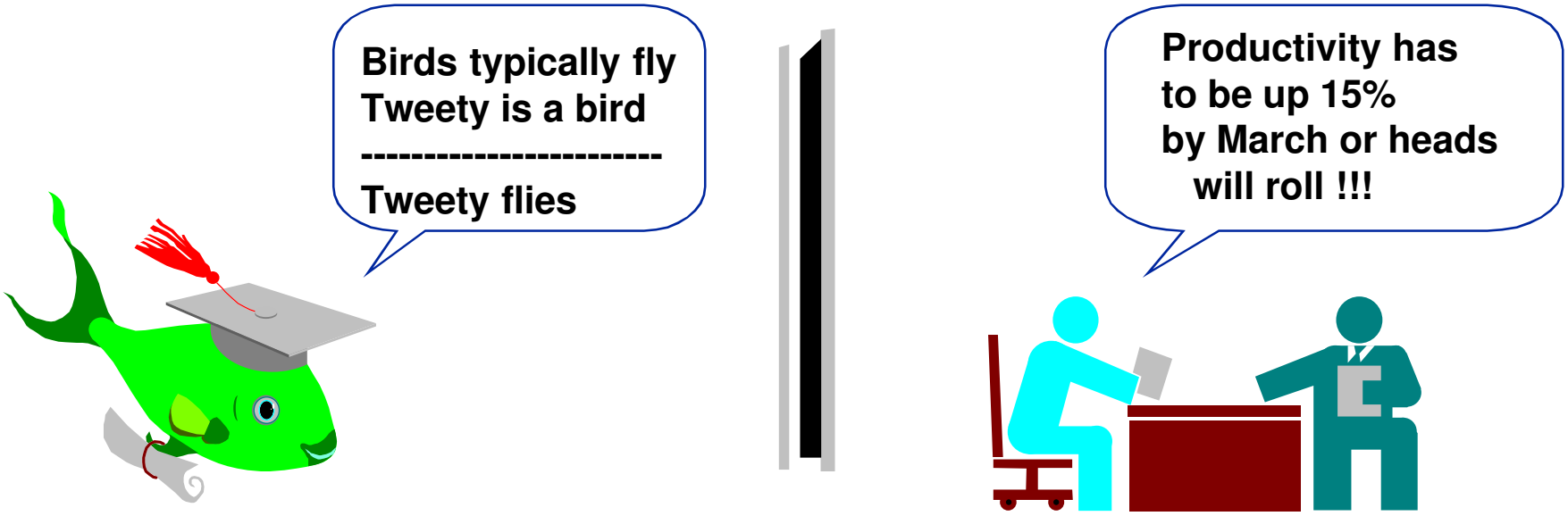
Inheritance Comes of Age:

Applying Nonmonotonic Techniques to Problems in Industry

Leora Morgenstern
IBM T.J. Watson Research Center



- **nonmonotonic logic --- plausible reasoning --- is invisible to industry**

In general, AI overhyped
(housekeeping robots vs. collecting tennis balls)
but at least it's there:

- **<u>expert systems</u> for medical diagnosis**
  circuit configuration
  financial applications
- **<u>dictation systems</u>**

Where is nonmonotonic reasoning?

Should we care that nonmnotonic reasoning is invisible?

(This isn't anything new)

Back in the 80's we didn't care:

- **AI was hot**
- **Research money was plentiful**

But in the 90's:

- **funding has shrunk**
- **no tolerance for programs that don't promise & deliver**

Should we care that nonmnotonic reasoning is invisible?

(This isn't anything new)

Back in the 80's we didn't care:

- AI was hot
- Research money was plentiful

But in the 90's:

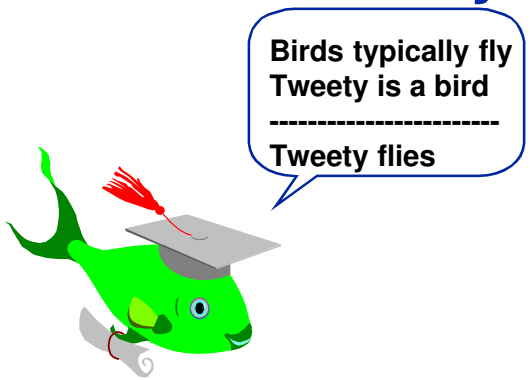
- funding has shrunk
- no tolerance for programs that don't promise & deliver

DANGER !!!!

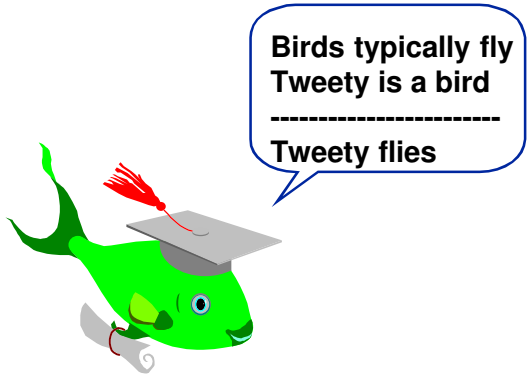
- nonmonotonic research will be marginalized
- funding will wither away
- field will shrink
- death of a once vibrant and exciting field

How to reestablish Nonmonotonic Reasoning as a central part of mainstream AI?

- figure out why nonmonotonic reasoning and industry are so far apart



- fix the problems



OUTLINE

- reason for gap between nonmon and industry
- general strategy to bridge the gap
- example of a nonmonotonic system in industry
  - benefits inquiry in medical insurance corporation
  - extension of inheritance with exceptions:
    inheritance of well-formed formulae
  - new paradigm for nonmonotonic reasoning
- generalization of system
- general lessons for connecting nonmon and industry

Analyzing the gap between nonmon and industry

Nonmonotonic reasoning introduced in 70's to capture plausible/default reasoning

McCarthy
McDermott&Doyle
Reiter

- **reasoning with default rules, exceptions**
- **reasoning with incomplete information**
- **jumping to conclusions**
- **retracting conclusions if necessary**

Analyzing the gap between nonmon and industry

Nonmonotonic reasoning introduced in 70's to capture plausible/default reasoning

McCarthy
McDermott&Doyle
Reiter

- reasoning with default rules, exceptions
- reasoning with incomplete information
- jumping to conclusions
- retracting conclusions if necessary

Example:

If I take flight 104 to London, I will arive at 7:50 am

But that isn't true: what if takeoff is delayed? the weather is bad? there is congestion at Heathrow?

Example of nonmonotonic reasoning:

If I take flight 104 to London, I will arive at 7:50 am

But that isn't true: what if takeoff is delayed? the weather is bad? there is congestion at Heathrow?

Okay, so I'll write the rule this way:

If I take flight 104 to London
and takeoff is on time
and the weather is good
and there's not much congestion at Heathrow
then I will arrive at 7:50 am

But that won't work either --- what if an engine fails? or the plane runs out of fuel?

- problem in enumerating all possible conditions (Knowledge rep problem)
- problem in figuring out that they all hold (incomplete knowledge problem)

Still, we're okay at predicting when planes will land

How can we do this prediction?

The airplane example, continued:

If I take flight 104 to London,
and nothing goes wrong,
then I'll land at 7:50 am

The trick is to assume that nothing is going to go wrong,
so we can infer the plane will land at 7:50 am

But something could go wrong!!!
(And then our system is inconsistent)

So, we have to give a way out:
if we find out we're wrong, retract the wrong conclusion

The example, formalized:

flight-104(MyFlight)

flight-104(x) : on-time(x)

---

on-time(x)

the inference rule!!! It means:
if x is of type flight104
and it's consistent that it's on time
then conclude that it's on time

You can now conclude:

on-time(MyFlight)

BUT:

You might also have rules like:

Bad-weather(destination(x)) ==> not on-time(x)

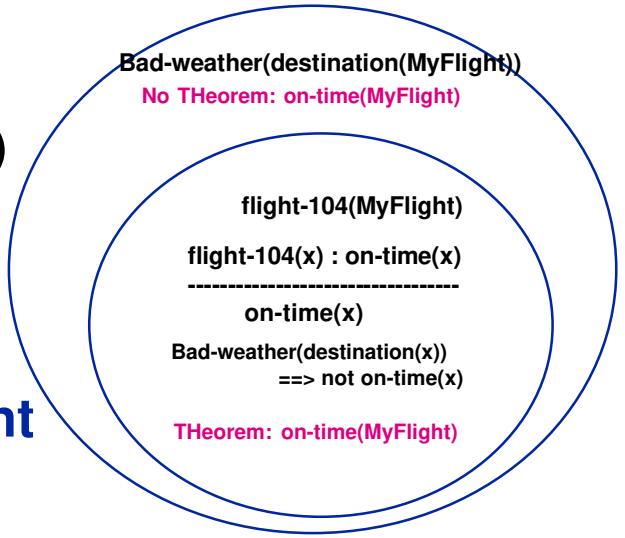
Then, if you add an axiom such as

Bad-weather(destination(MyFlight))

on-time(MyFlight) will no longer be consistent

You will have to withdraw that conclusion

That's what makes this logic nonmonotonic



Analyzing the gap between nonmon and industry

Nonmonotonic reasoning introduced in 70's to capture plausible/default reasoning

McCarthy
McDermott&Doyle
Reiter

- **reasoning with default rules, exceptions**
- **reasoning with incomplete information**
- **jumping to conclusions**
- **retracting conclusions if necessary**

Aim: more powerful logic, more flexible programs
Expectation: tool of software engineers (like o-o)

Analyzing the gap between nonmon and industry

Nonmonotonic reasoning introduced in 70's to capture plausible/default reasoning

McCarthy
McDermott&Doyle
Reiter

- **reasoning with default rules, exceptions**
- **reasoning with incomplete information**
- **jumping to conclusions**
- **retracting conclusions if necessary**

Aim: more powerful logic, more flexible programs
Expectation: tool of software engineers (like o-o)

No secret ---- we all admit this (almost) ....
But AI is all about getting things done !

Analyzing the gap between nonmon and industry

WHY?

1. **Nonmonotonic reasoning focusses on toy problems**

2. **Industry doesn't understand commonsense reasoning**

3. **NMR techniques haven't scaled up to industry**

1. **Nonmonotonic reasoning focusses on toy problems of commonsense reasoning**

Examples:

- **Tweety**
- **Yale Shooting Problem**

1. Nonmonotonic reasoning focusses on toy problems of commonsense reasoning

Examples:

Tweety Problem

From: Tweety is a bird
Birds Typically Fly

Infer: Tweety Flies

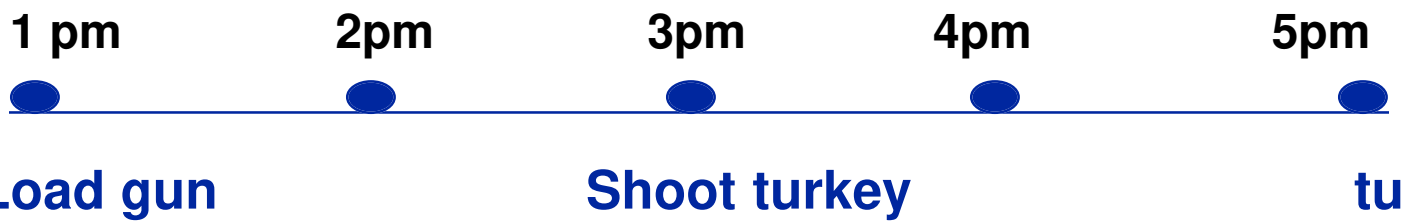
Retract Tweety Flies **if find out that** Tweety is a penguin

- **still a benchmark after all these years!!!**

1. Nonmonotonic reasoning focusses on toy problems of commonsense reasoning

Examples:

Yale Shooting Problem



Shooting a loaded gun always results in death;
Guns usually stay loaded; turkeys usually stay alive.

Can we infer that the turkey dies at 3 pm?

1. Nonmonotonic reasoning focusses on toy problems of commonsense reasoning

Tweety

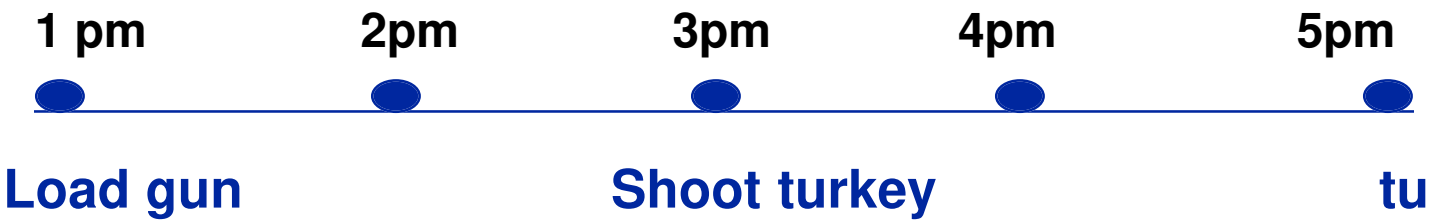
From:

Tweety is a bird
Birds Typically Fly

Infer:

Tweety Flies

Yale Shooting Problem



Why should industry invest in technology based on this?

2. Industry doesn't care about commonsense reasoning

Industry is concerned with

- **diagnosing bacterial infections**

- **determining where oil is**

- **predicting the stock market**

These may be easier than the Yale shooting problem, but that doesn't help either nmr or industry

2. Industry doesn't care about commonsense reasoning

Industry is concerned with

- **diagnosing bacterial infections**

- **determining where oil is**

- **predicting the stock market**

These may be easier than the Yale shooting problem, but that doesn't help either nmr or industry

Irony: **<u>nonmonotonic reasoning is present here</u>**
e.g., predicting (the stock market, oil) in absence of complete info is nonmonotonic

Nonmonotonic research has ignored industry problems, so there is no evidence of being able to help industry

3. NMR techniques haven't scaled up to industry

Nonmonotonic reasoning crippled by

- decidability problems
- tractability problems
- lack of good tools

3. NMR techniques haven't scaled up to industry

Nonmonotonic reasoning crippled by

- **decidability problems**

  nonmonotonic predicate logic is undecidable
  (appeal to consistency of first-order logic)

- **tractability problems**

  determining whether formula is in extension
  of propositional default theory is Σ_2^P -complete

- **lack of good tools**

  There are good results!! polynomial algorithms for
  - **inheritance with exceptions** (Horty, Touretzky, Thomason; Stein ($O(n^5)$))
  - **logic programming** prop,non-rec. Horn($O(n)$)
    courteous logic programs($O(n^2)$)(Grosof)

But no corresponding tools; must code from scratch

This can kill an applied non-mon project at the start

nomonotonic reasoning is isolated from industry because

- **nmr and industry focus on different problems**
- **few effective algorithms and tools**

nomonotonic reasoning is isolated from industry because

- **nmr and industry focus on different problems**
- **few effective algorithms and tools**

Strategy for integration becomes clear:

First,

- **study problems in industry**
- **select subset where nonmon can contribute**
- **focus on those problems**

Second,

- **design efficient algorithms for tractable subtheories**
- **develop industrial strength code**

Do Both!!! (Ideally, at the same time)

To date: only a handful of industry problems with nonmonotonic solutions

Focus here: detailed examination of problem & solution

Problem:
Benefits Inquiry in Medical Insurance Corp.

Benefits Inquiry: What had been done

- **Code-based system**

- customer provides codes for procedure and diagnosis

- customer service rep uses a table-lookup system to determine coverage

Drawbacks:

- must have hard-to-get info
- granularity problem
- can't ask questions based on non-code information

  Can a 44-year old woman have a mammogram if she has had one 2 years ago?

- difficult to update (10000's procedure codes; 10000's diagnosis codes)

Benefits Inquiry for medical insurance

We need to reason about

- **which medical services are covered / excluded**

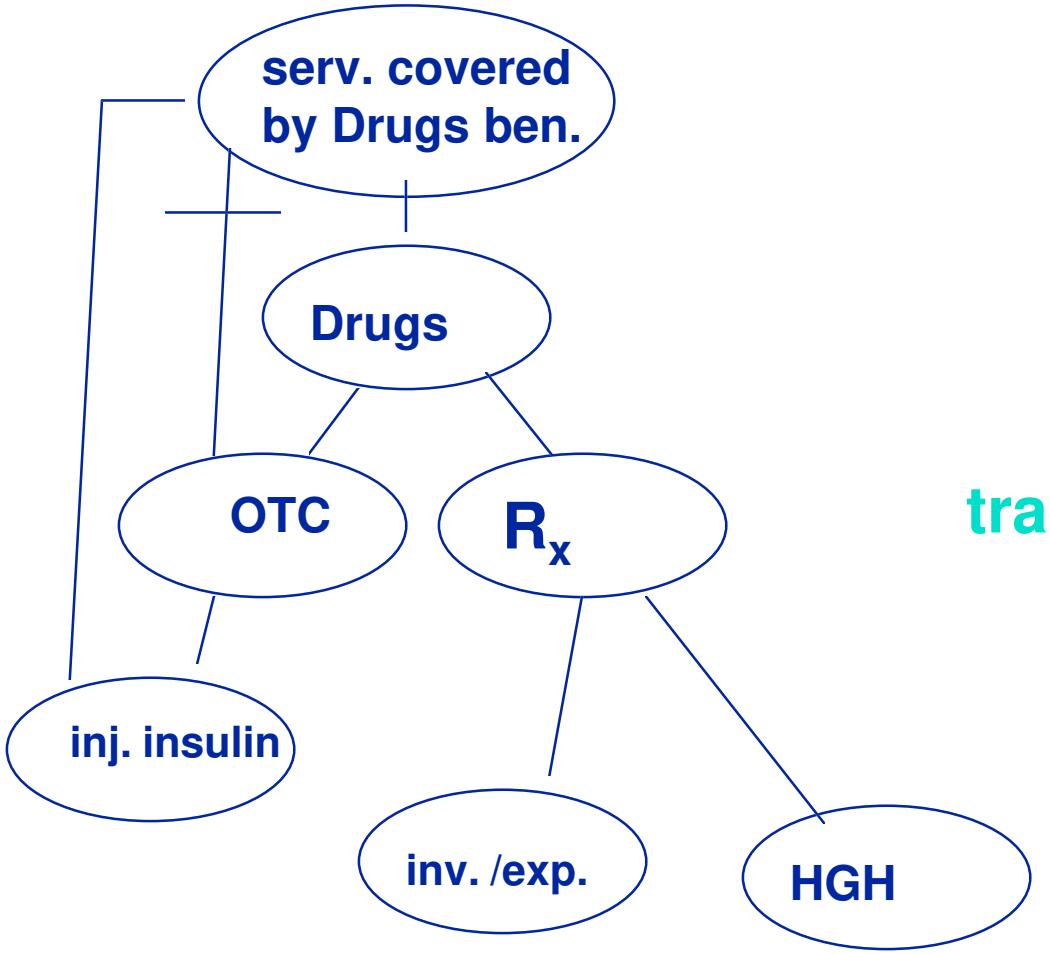
- **which rules and regulations apply to medical services**

Benefits Inquiry for medical insurance:

How does nonmonotonic reasoning enter the picture?

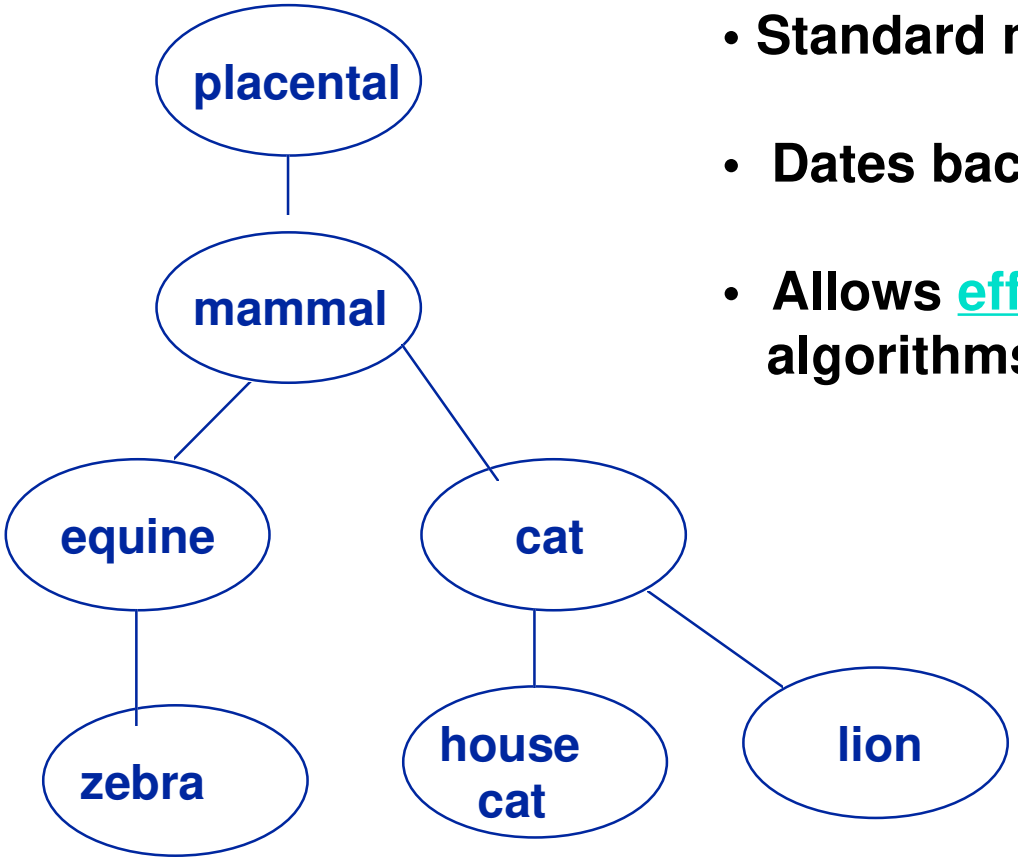
Medical services: taxonomony with exceptions (plus)

(Horty, Touretzky, Thomason; Stein)



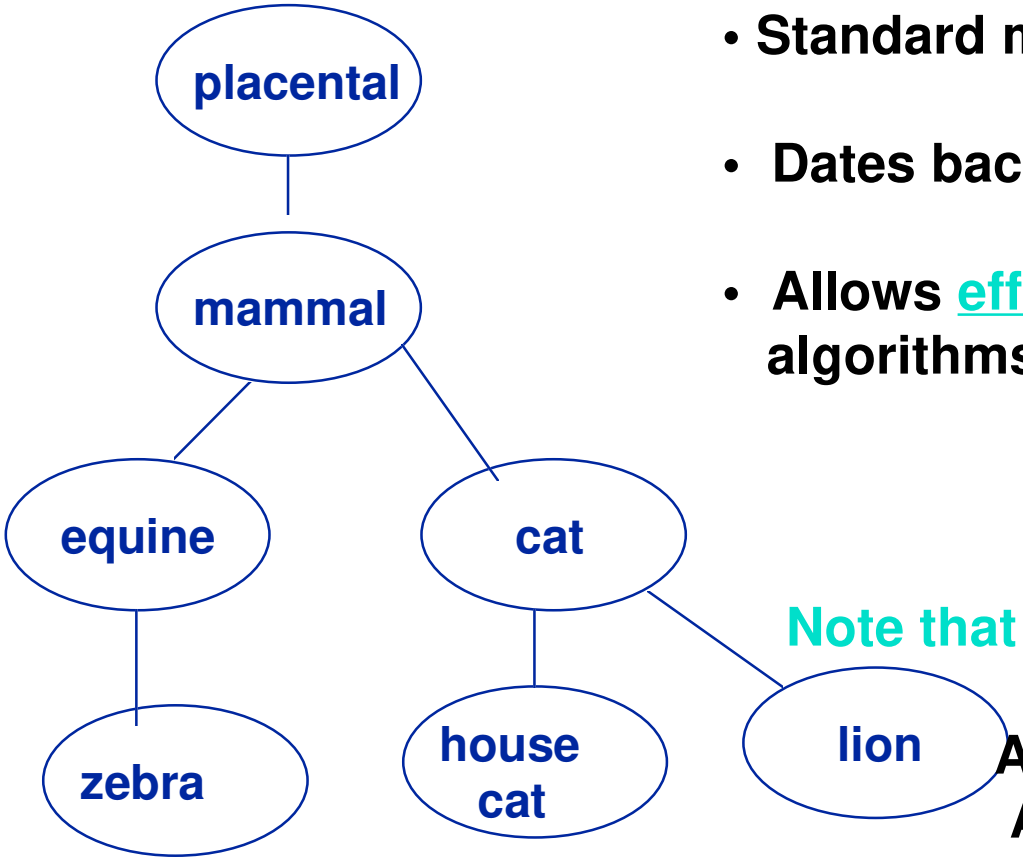
tractable form of nonmon

Brief Review of Inheritance



- Standard monotonic inheritance
- Dates back to Aristotle
- Allows efficient, graph-based algorithms for reasoning

Brief Review of Inheritance

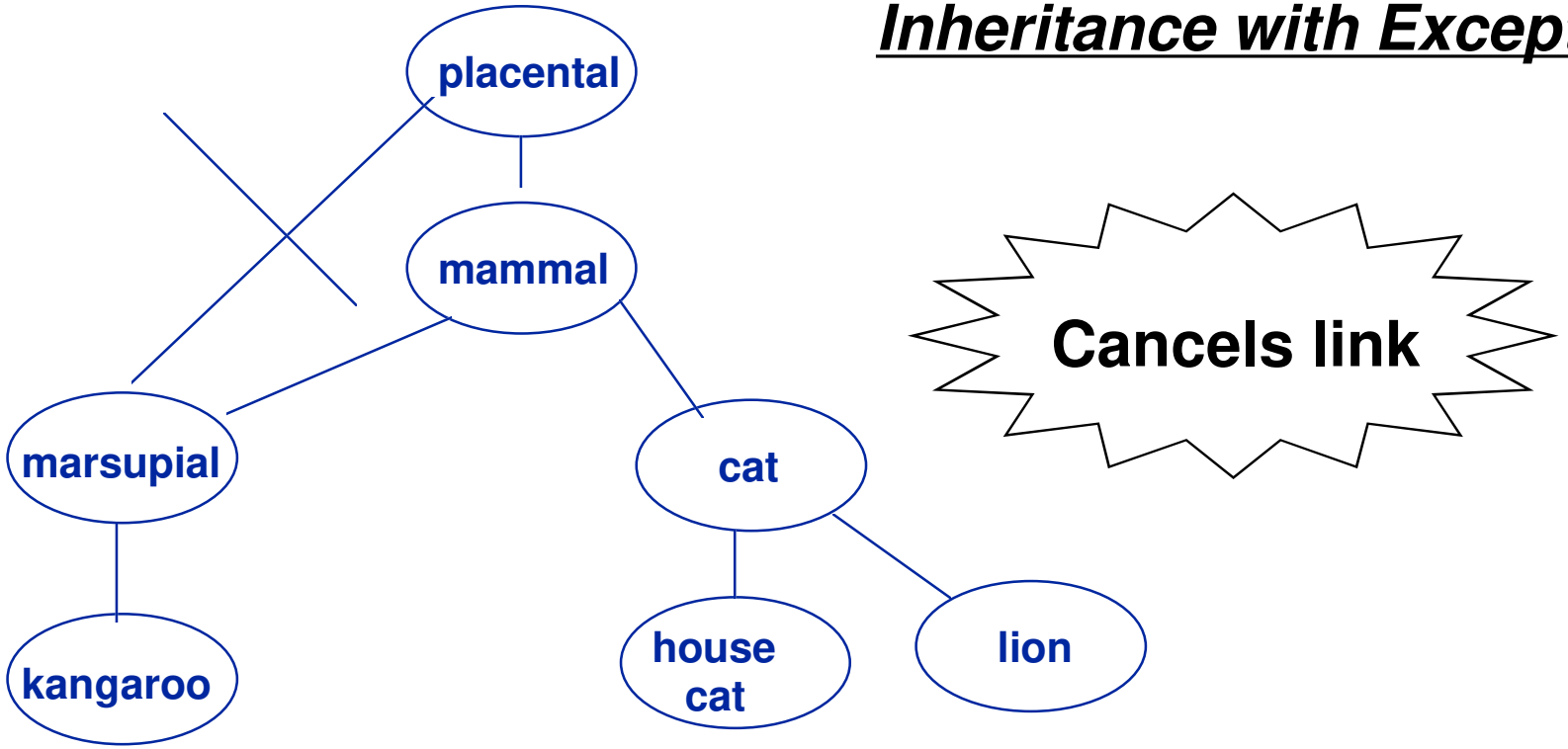


- Standard monotonic inheritance
- Dates back to Aristotle
- Allows efficient, graph-based algorithms for reasoning

Note that this translates into logical rules:

All zebras are equines
All lions are cats
All equines are mammals
All mammals have placentas ....

Review of Inheritance

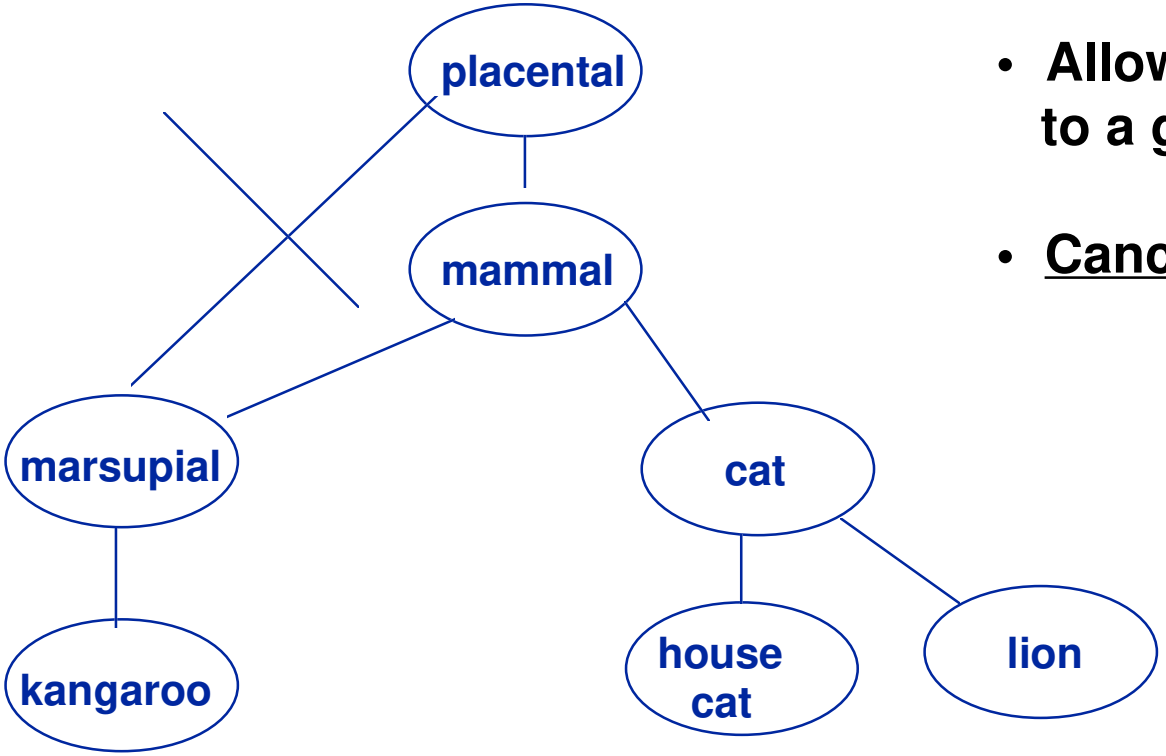


Inheritance with Exceptions

Cancels link

All cats are mammals; all kangaroos are marsupials;
all marsupials are mammals;
~~all~~ most mammals have placentas
marsupials do not have placentas

Review of Inheritance



Inheritance with Exceptions

- **Allows us to specify exceptions to a general class**
- **Cancels link: nonmonotonicity**

We could write this in a nonmonotonic logic:

mammal(x) : placental(x)
----
placental(x)

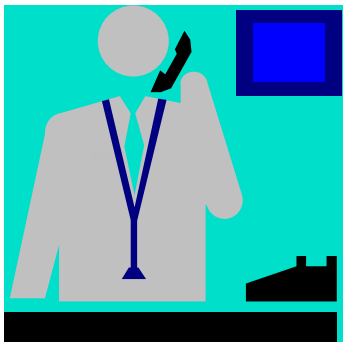
marsupial(x) ==> not placental(x)
kangaroo(x) ==> marsupial(x)

So if Kiri is a kangaroo, we conclude that Kiri is not placental

To date: only a handful of industry problems with nonmonotonic solutions

Focus here: detailed examination of problem & solution

Problem:
Benefits Inquiry in Medical Insurance Corp.



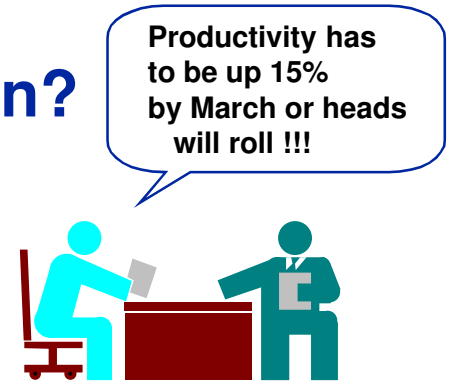
Patients ask Customer Service Reps:

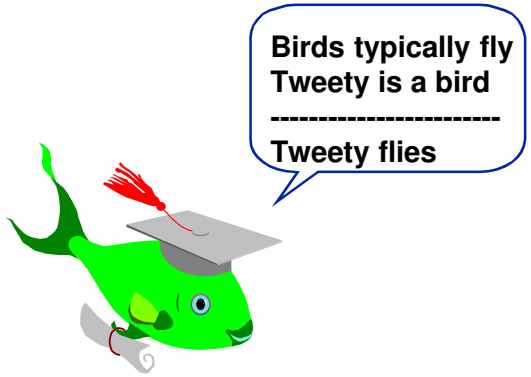
- **is my son's tonsillectomy covered inpatient?**
- **how long can I stay in the hospital after a standard delivery?**
- **what will the copay be for an experimental drug?**

Difficult problem due to

- **many business regulations for each insurance product**
- **many insurance products**
- **insurance products constantly change**

What lessons can we learn?





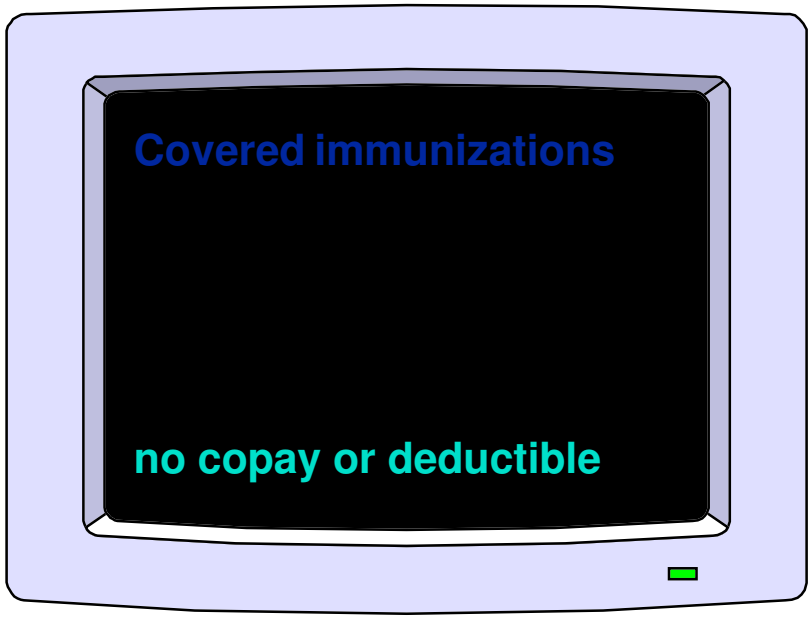
- Keep our eyes open for problems
- Basic research is still crucial
- Proper balance is important, but hard to maintain
- Still need: KR, knowledge engineering, etc.
- Tools!!!

Benefits Inquiry: What had been done

- Nothing
- Placing charts and manuals on line
- “Desktop” text-based systems
- Code-based systems

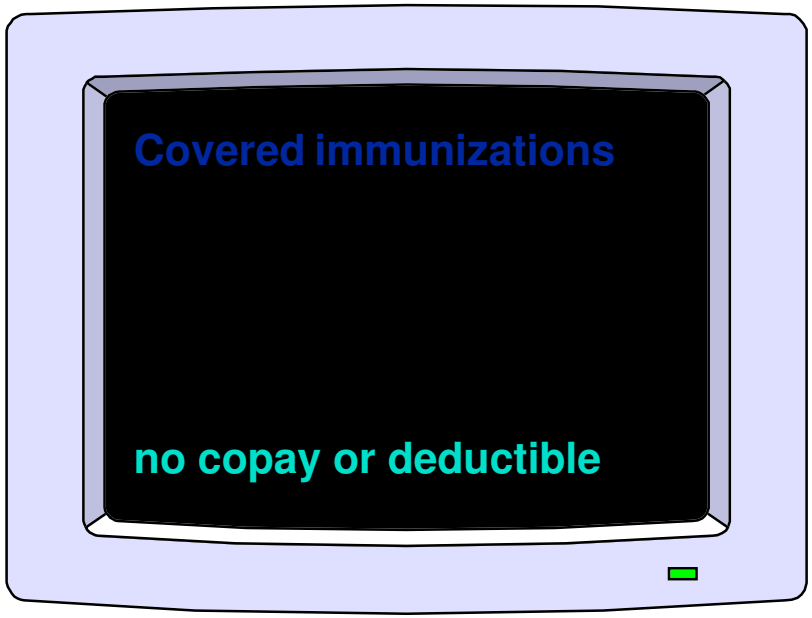
Benefits Inquiry: What had been done

- "Desktop" text-based systems
  - information divided into subject areas
    preventive care, immunization, drugs
  - subjects areas associated with screenful of text



Benefits Inquiry: What had been done

- "Desktop" text-based systems
  - information divided into subject areas
    preventive care, immunization, drugs
  - subjects areas associated with screenful of text



Drawbacks:

- only contains small portion of info
- no explicit connection between linked areas (preventive, immunization)
- hard to update
- only handles frequently asked questions

Benefits Inquiry: What had been done

- **Code-based system**

- customer provides codes for procedure and diagnosis

- customer service rep uses a table-lookup system to determine coverage

Benefits Inquiry for medical insurance

Project Goal:

- **supports benefits inquiry**
- **avoids drawbacks**

Benefits Inquiry for medical insurance

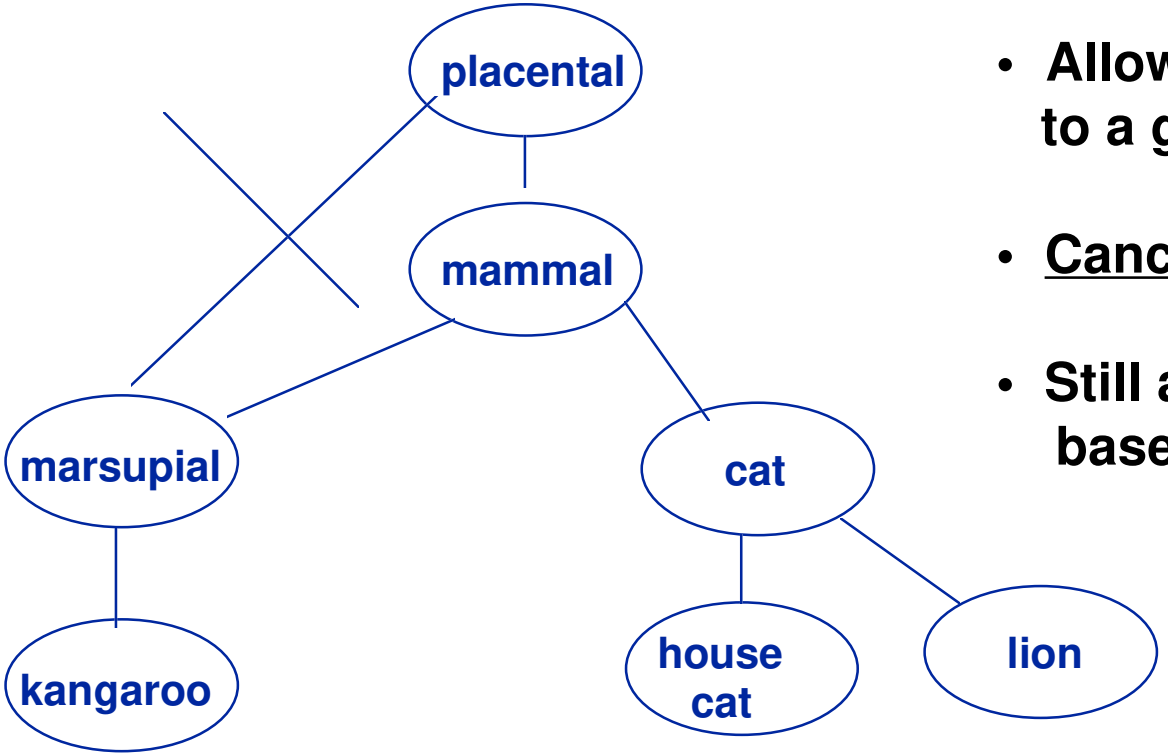
Project Goal:

- supports benefits inquiry
- avoids drawbacks

Needed: system that

- allows questions at varying levels of granularity
- gives unambiguous answers
- allows representation of large amount of information
- supports navigation around large information space
- supports connections among related topics
- suports easy updates and modifications

Review of Inheritance



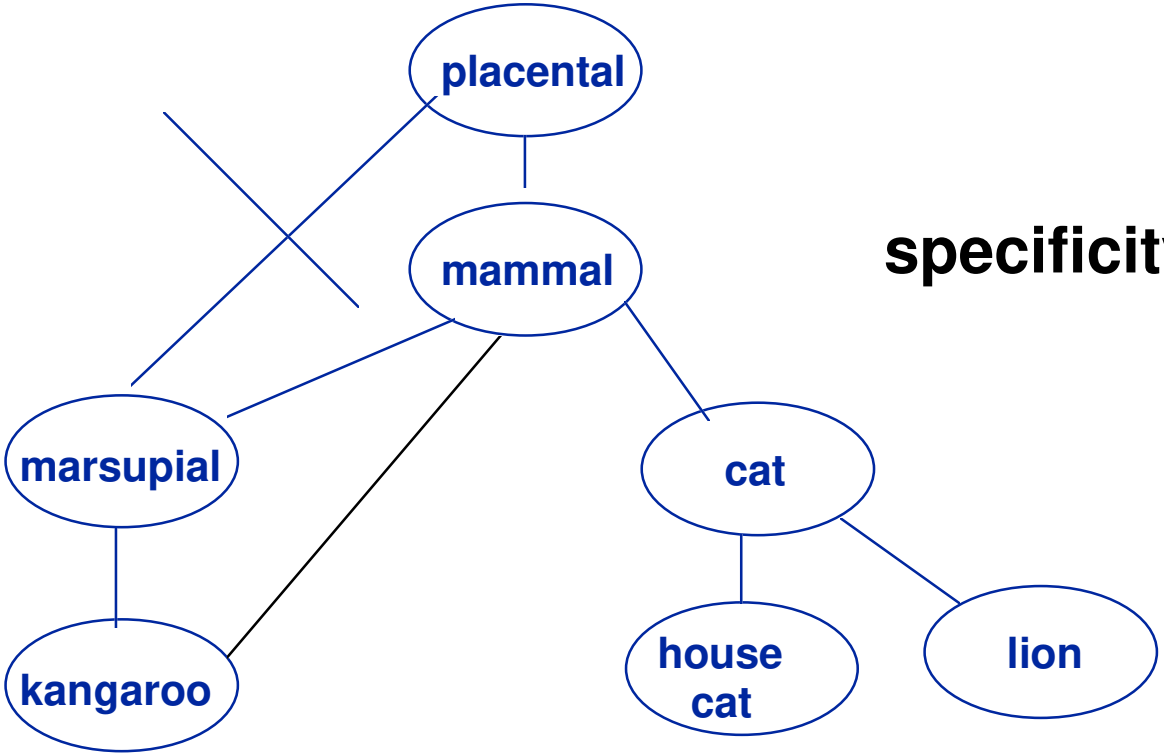
Inheritance with Exceptions

- **Allows us to specify exceptions to a general class**
- **Cancels link: nonmonotonicity**
- **Still allows efficient reasoning based on specificity**

That is, we have nice, graph algorithms that figure out the answers in reasonable polynomial time

Review of Inheritance

Inheritance with Exceptions

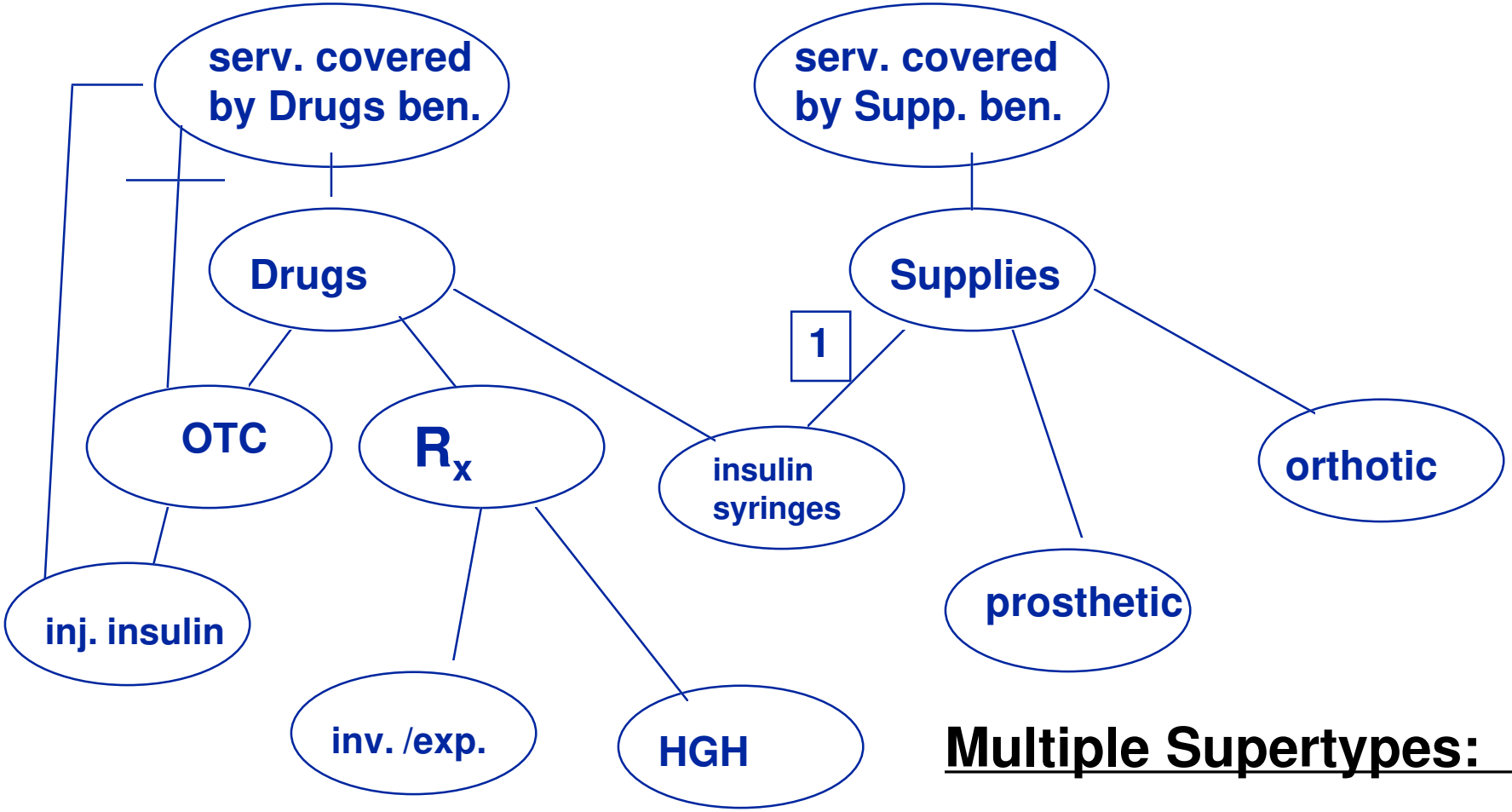


specificity vs. shortest-path

Benefits Inquiry for medical insurance:

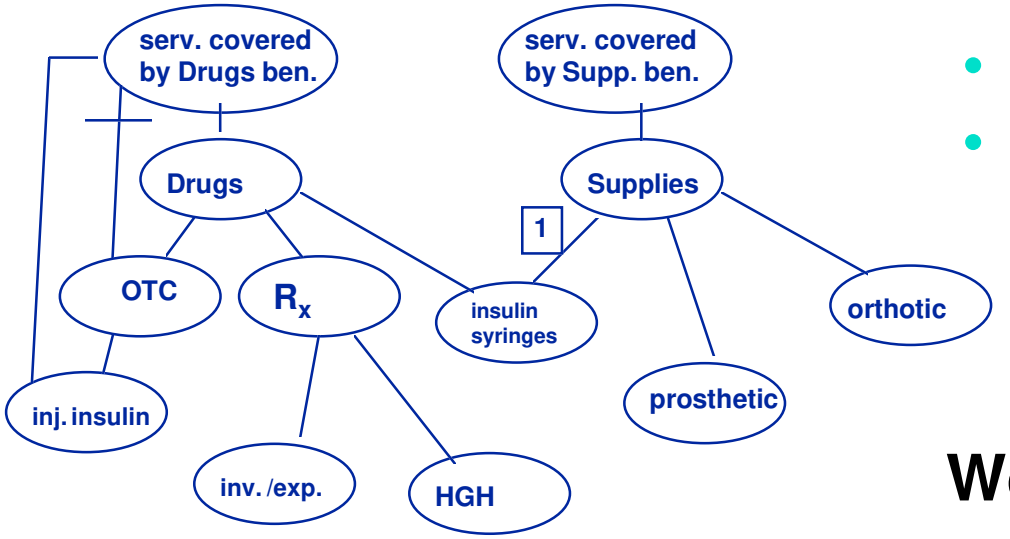
How does nonmonotonic reasoning enter the picture?

Medical services: taxonomony with exceptions plus



Multiple Supertypes: DAG

Benefits Inquiry



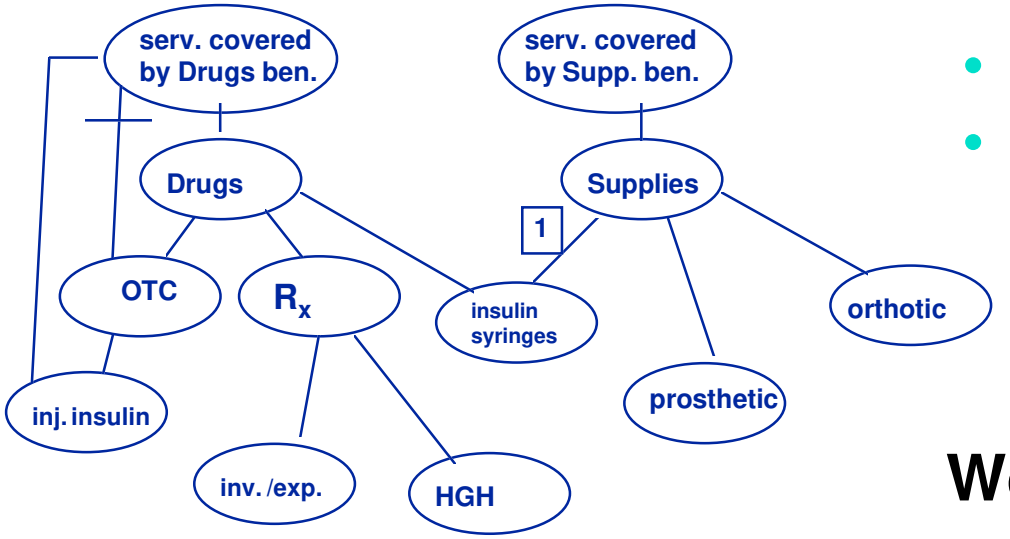
Inheritance with exceptions solves part of the problem:

- which services are covered?
- by which benefit?

Won't solve:

- which regulations apply?
- how large is the benefit?

Benefits Inquiry



Inheritance with exceptions solves part of the problem:

- which services are covered?
- by which benefit?

Won't solve:

- which regulations apply?
- how large is the benefit?

We'll need something more powerful
(building on inheritance with exceptions)----
But this shows that nonmon can be useful in industry

Benefits Inquiry:

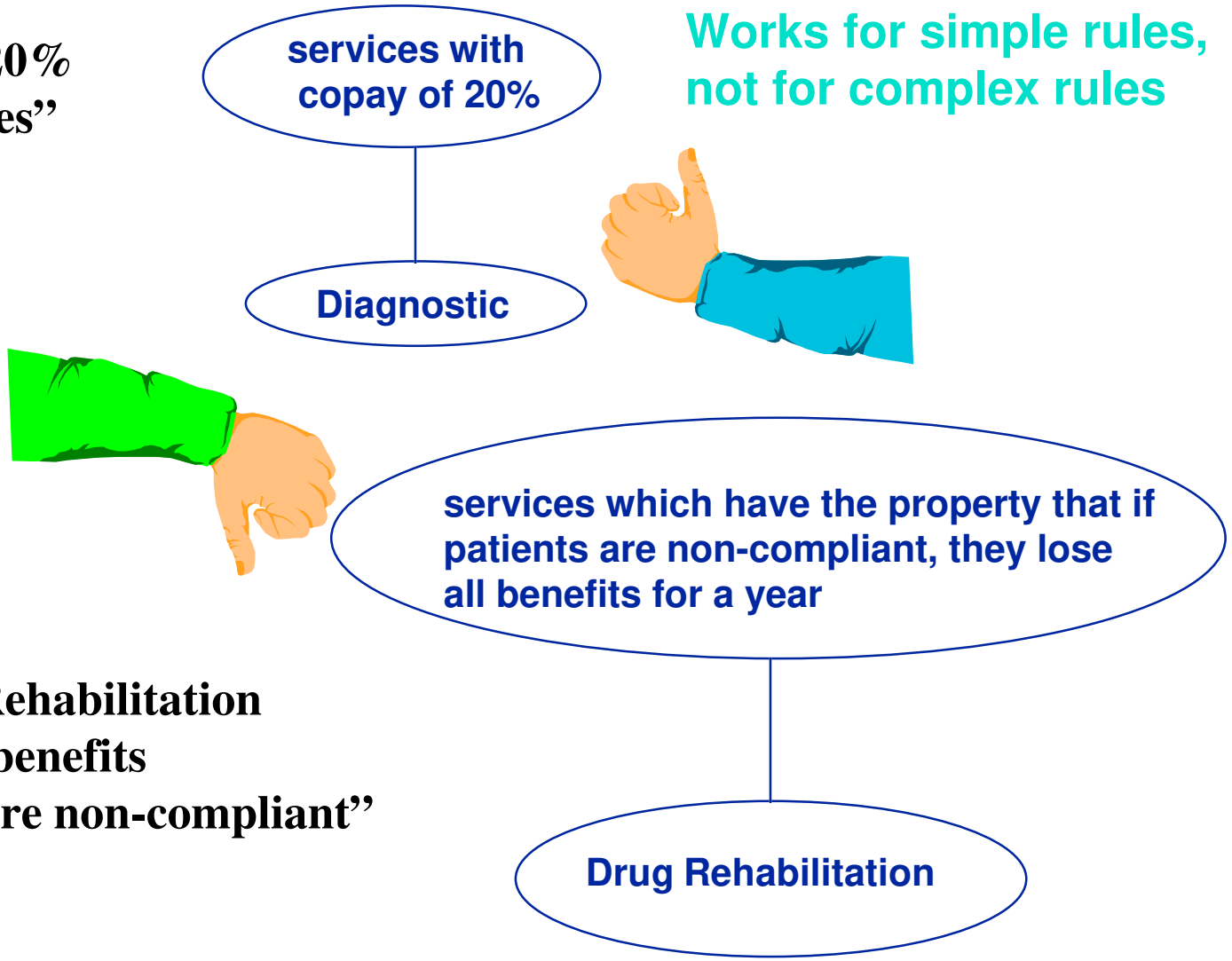
- How to represent business rules

- How to reason with business rules

How to represent business rules

- **in semantic net? can we represent rules as nodes?**

"There is a copay of 20% for diagnostic services"



"Patients in Drug Rehabilitation programs lose all benefits for a year if they are non-compliant"

Alternative: Attach business rules to nodes in semantic net

new structure: FAN

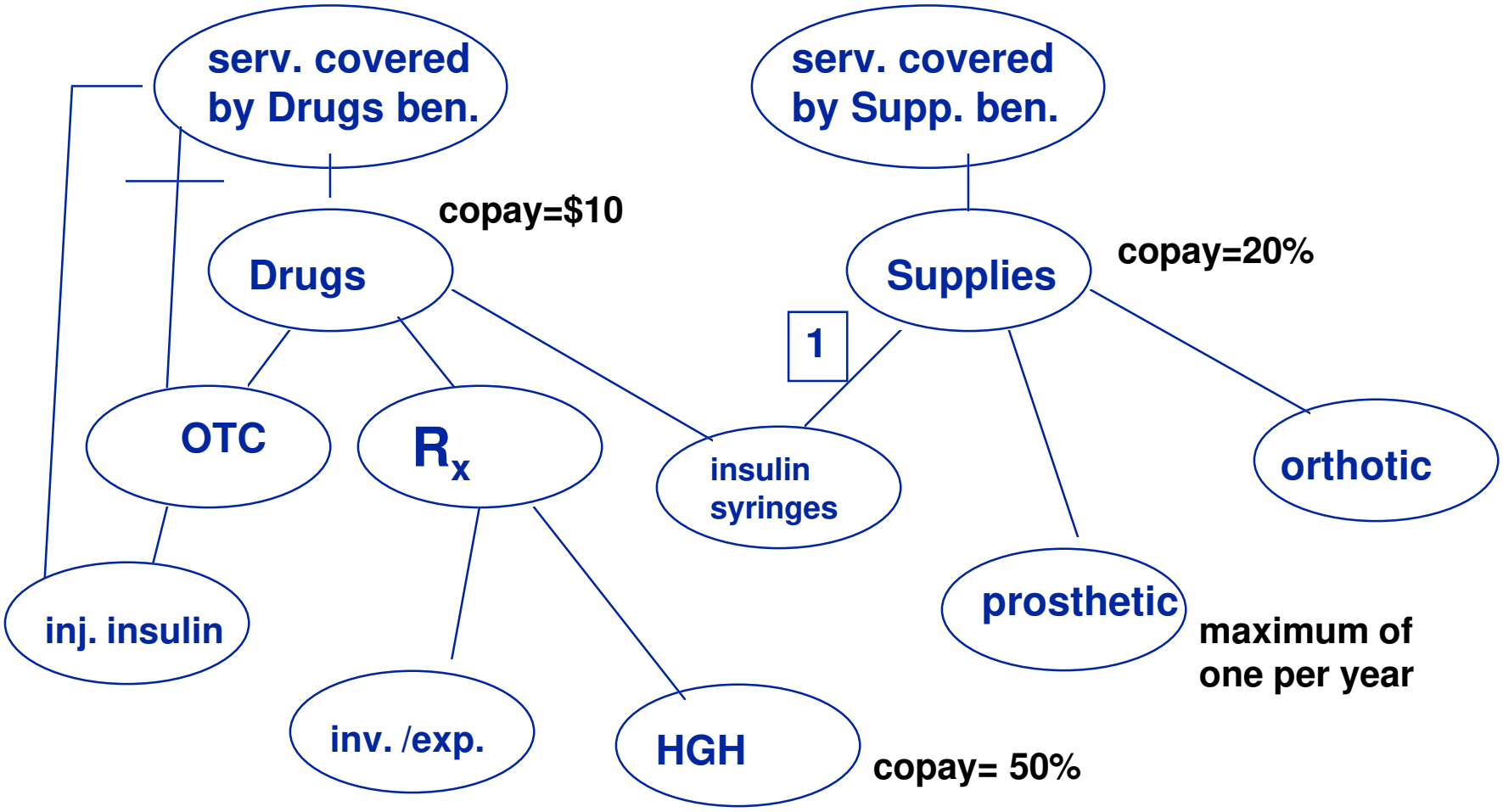
Formula Augmented (Semantic) Network

- N --- a set of nodes
- W --- a set of well-formed formulae
- B --- a background context
- L1 --- a set of links conecting nodes
- L2 --- a set of links connecting nodes to sets of wffs
- O --- set of partial orders on some members of L1
  (lets us prefer certain links/paths)

Example of a FAN

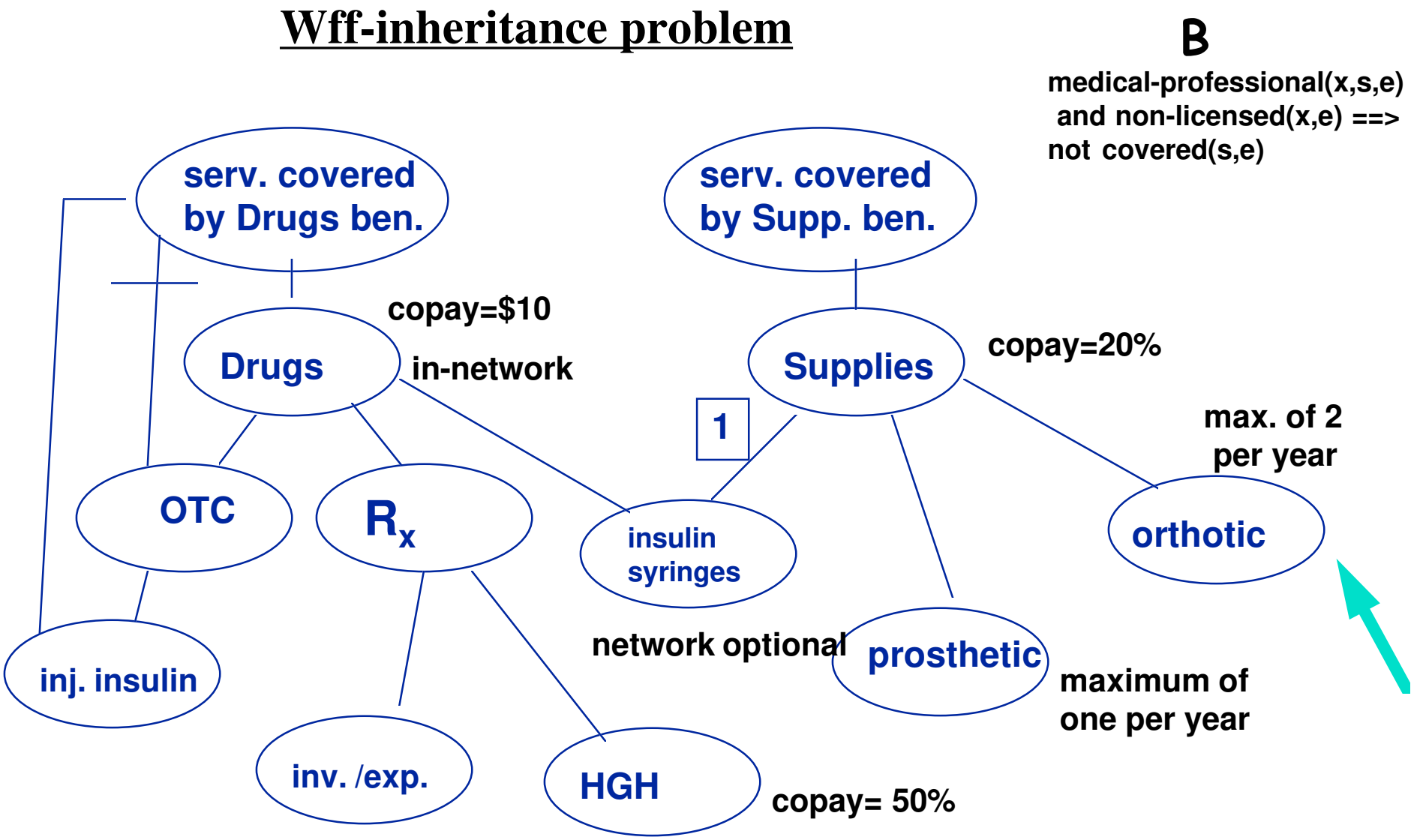
B

medical-professional(x,s,e)
and non-licensed(x,e) ==>
not covered(s,e)



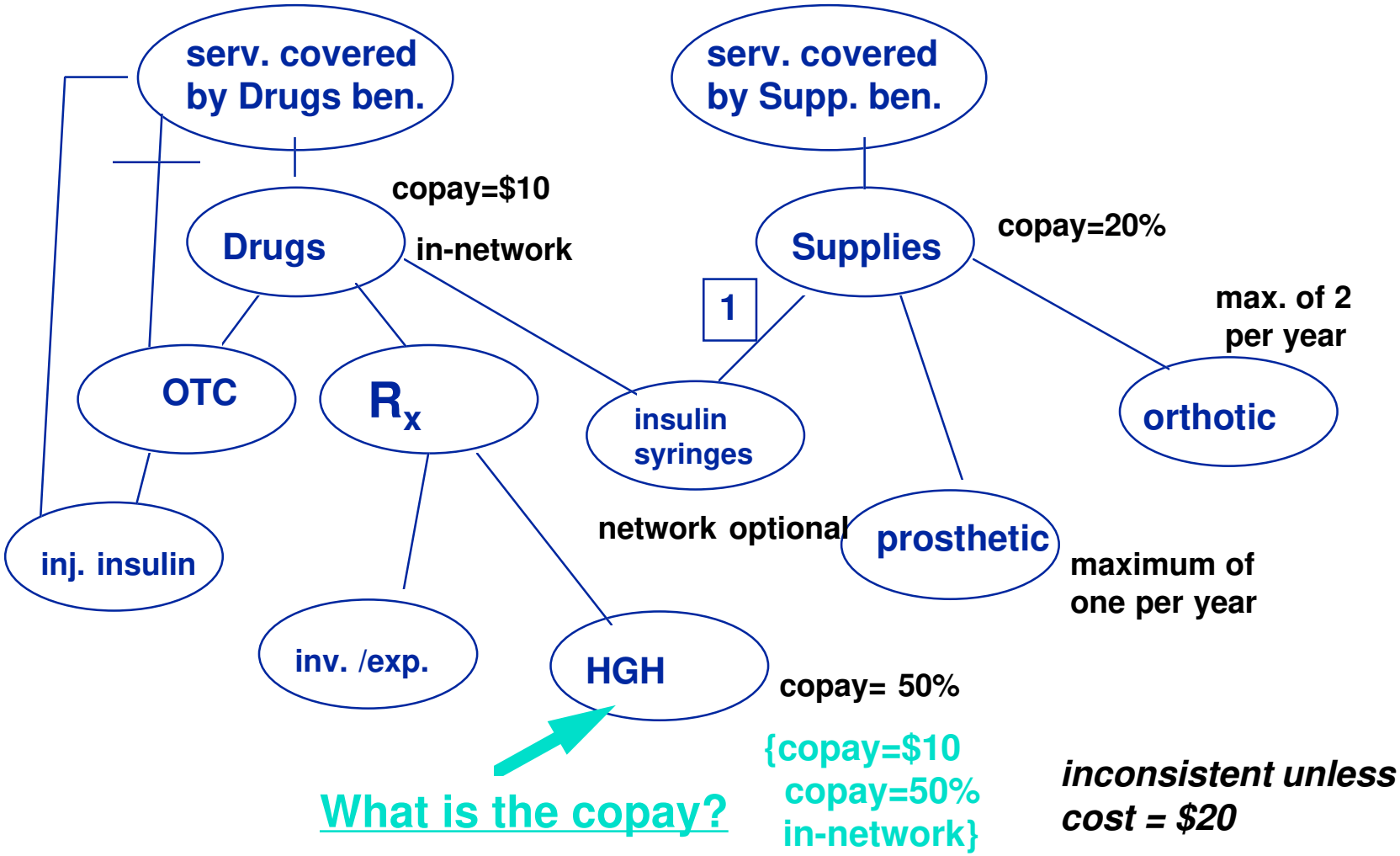
Reasoning in a FAN:
Determining which rules apply to a (focus) node

Wff-inheritance problem



Naive approach: just travel up dag, collecting rules as you go along

Inconsistent!!!



- **Wff-inheritance: must take maximal consistent subset of rules**
- **more than 1 maximal subset; how to choose?**

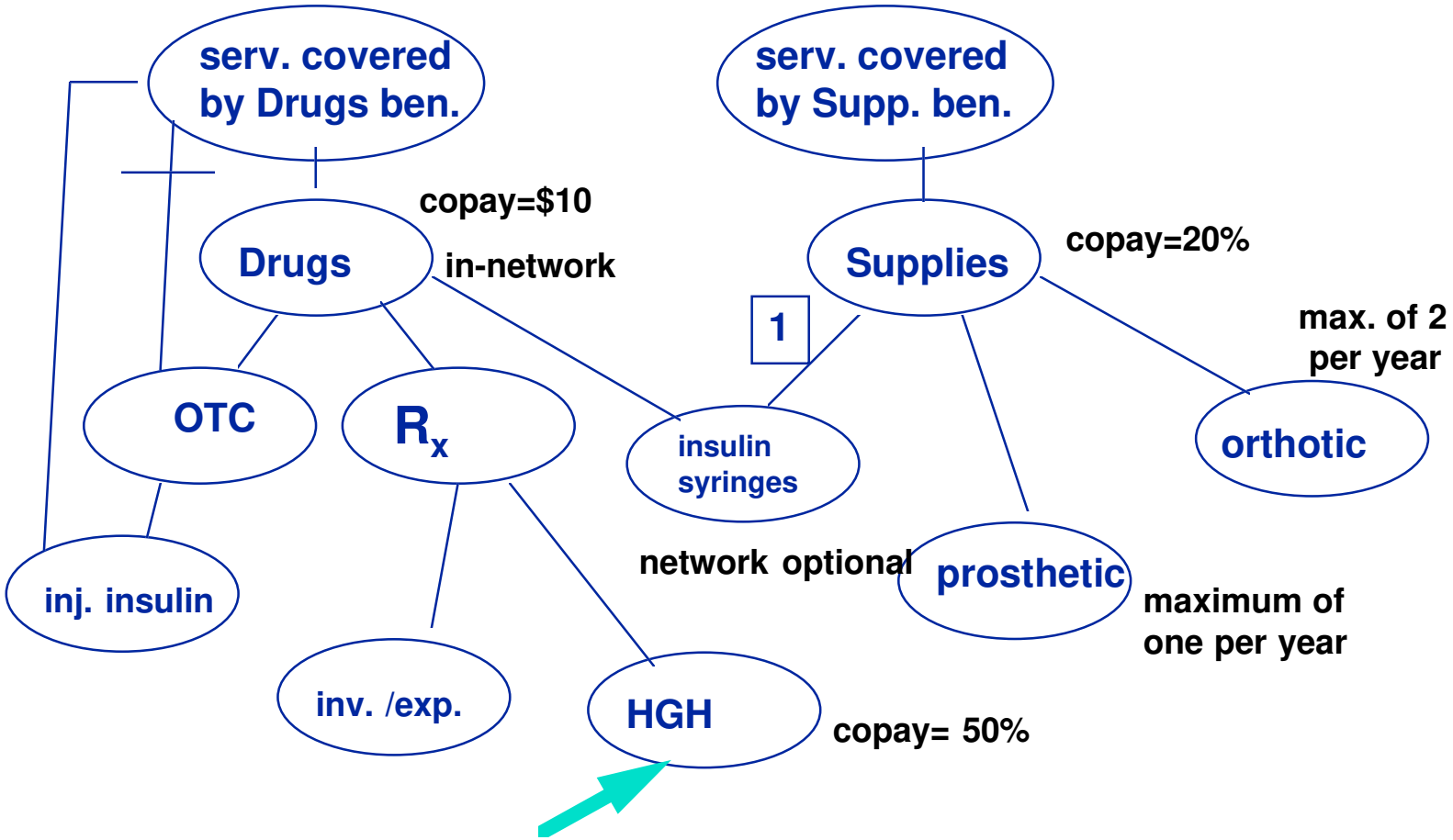
{copay=$10
copay=50%
in-network}

{copay=$10
in-network}

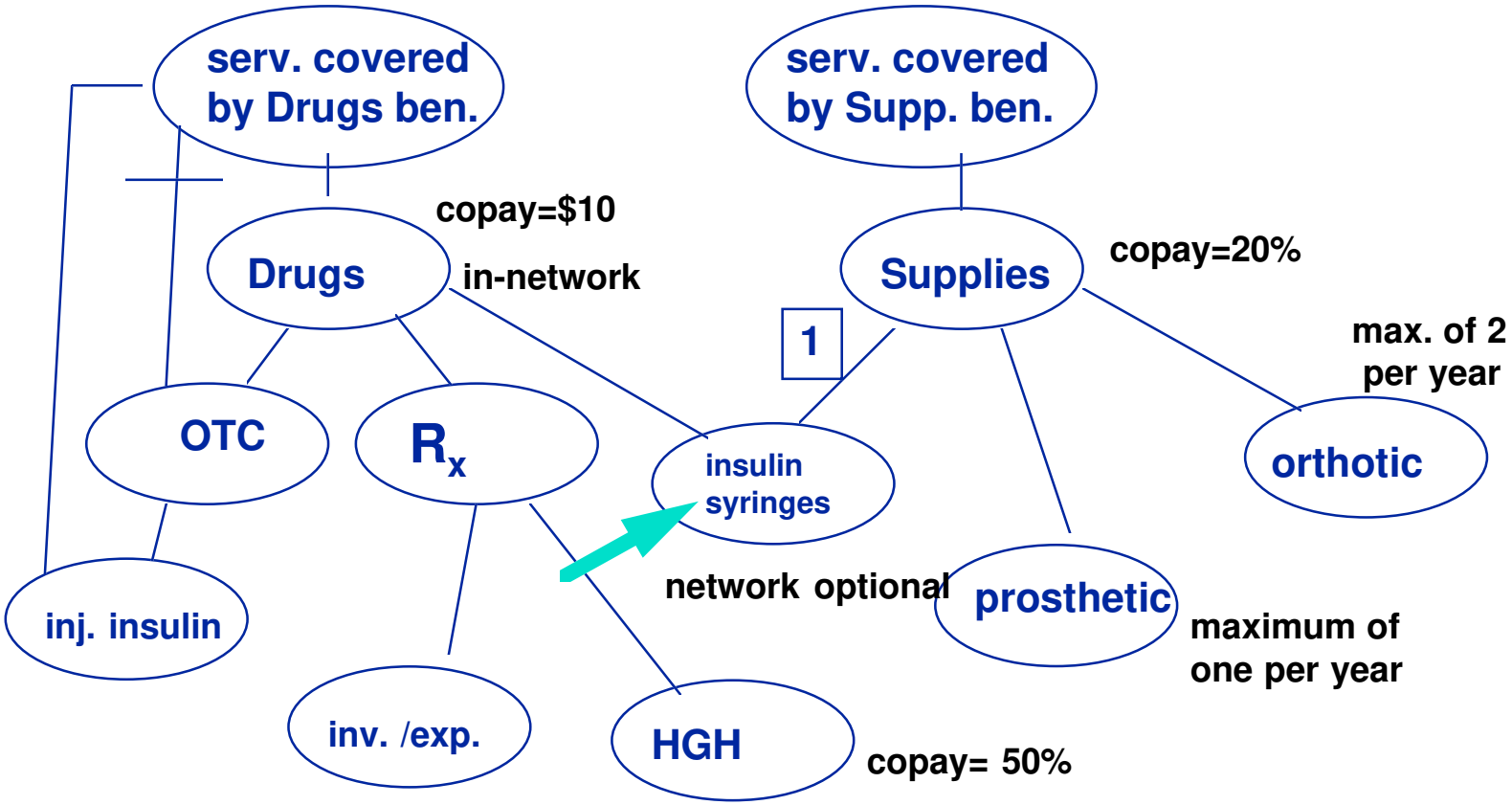
{copay=50%
in-network}

- **preference criteria**
  - **specificity**
  - **path preference**

Preference criterion of specificity



Preference criterion of path preference



What is the copay?

copay=20%
is on a more preferred path than
copay=$10

Other preference criteria:

- some classes of wffs have higher priorities
  (medical rules vs. administrative rules)

- some models are preferred
  (in benevolent insurance companies,
  argument for getting reimbursed wins over
  argument against getting reimbursed)

We can formalize this using a definition of a preferred maximally consistent subset .....

Assume that S1, S2 are sets such that S1 U S2 is inconsistent wrt B. Let X1 and X2 be max. cons. subsets of S1 U S2 wrt B. Let R1 be a subset of S1, R2 a subset of S2, s.t. R1 = X1 -X2; R2 = X2 -X1. That is, R1 and R2 are all that distinguish X1 from X2. Then if S1 is preferred to S2, X1 is a pmcs of S1 U S2. Otherwise, if S1 U S2 is consistent wrt B, S1 U S2 is a (in this case, the) pmcs of S1 and S2.

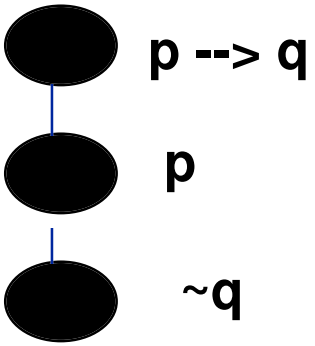
But this won't be very useful in practice.
We need

- **a procedure to compute a pmcs of 2 sets**
- **a procedure to traverse the network**

The Algorithm:

- iterative traversal

- downward traversal isn't just inefficient; it's wrong



- dag means we have to handle fork points; path preference gives ordering of recursive traversal

- can't collect rules from nodes only on pre-empted paths, so we must preprocess network (Stein, 1992)

The Algorithm:

Procedure Compute Ψ(N):

Preprocess the FAN to remove preempted, conflicted links
Ψ(N) := wffs(N)
FAN-traverse(N)

Procedure FAN-traverse(x):

```
If x has not been visited
   mark x as visited
   Ψ(N):= pmcs(Ψ(N),wffs(x))
   if x is a root then return
                  else determine all nodes yi such that x --> yi
                       do topological sort of all k links x --> yi
                       for i:= 1 to k do FAN-traverse(yi)
```

Computational issues

- **Determining a preferred maximally consistent subset means checking for consistency**
  - **---- undecidable in first-order case**
  - **---- intractable in propositional case**
    - **divide-and-conquer (rule types, iterative traversal)**
    - **Selman&Kautz's efficient method for consistency checking**
    - **later: Grosof's polynomial methods**

How does this fit into standard non-monotonic reasoning?

- all (most) business rules are defaults
- effectively reasoning with a huge number of default rules
- most nonmonotonic theories are intractable,
  but some are less intractable than others
- connection to work of
  Brewka, Delgrande & Schaub, Geffner, Grosof
  (equivalency results in the works (Grosof))

Going forward from Benefits Inquiry

- **How can we generalize?**

  --- similar domains

  --- similar techniques

- **What lessons can we learn?**

Generalizing Benefits Inquiry: Similar Domains

- **Other parts of the insurance industry**

  life insurance, property and casualty insurance

- **Other tasks for insurance**

  adjudication
  dynamic product configuration
  (composition and subtyping)

Generalizing Benefits Inquiry: Similar Techniques

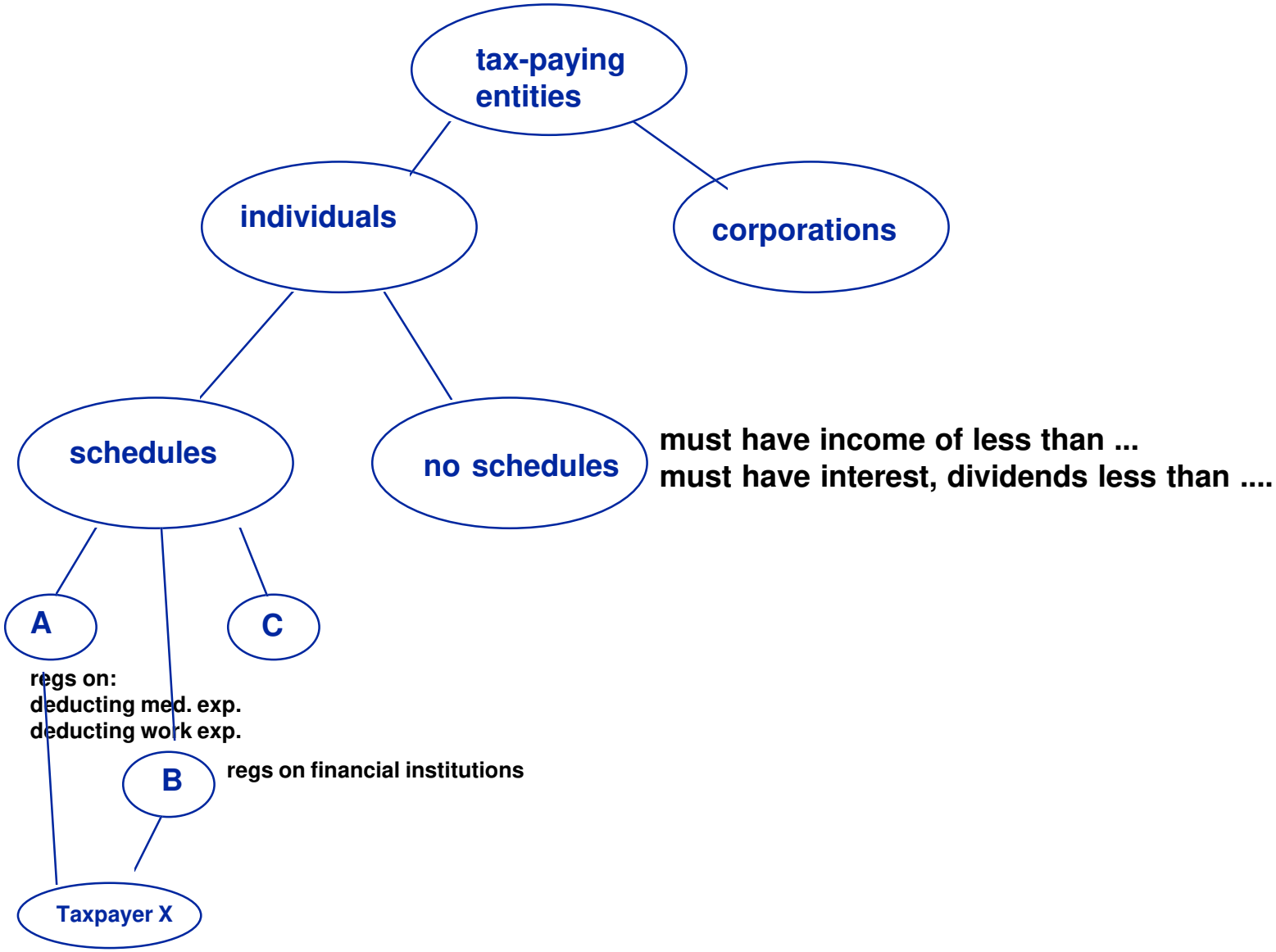
Can use this technique (FANs and wff-inheritance) if

- **there is a large amount of taxonomic information**
- **there is a significant amount of non-taxonomic info**
- **taxonomic & non-taxonomic info are conceptually linked**
- **non-taxonomic info can be mapped into wffs**

Generalizing Benefits Inquiry: Similar Techniques

Potential Domains:

- **Legal Reasoning**
- **Medical Reasoning**
- **Reasoning in Business Organizations**

Legal Reasoning (statutory & regulatory law)

Example: Taxes