Formula-Augmented Semantic Networks

A Flexible Method for Representing and Reasoning with Knowledge

Leora Morgenstern
IBM T.J. Watson Research Center

Claim:

We need to use knowledge representation techniques for practical applications

Knowledge Management, Intelligent Helpdesks, Customer Relationship Management, Knowledge-based Systems

Premise:

Large systems use knowledge

There is a need to represent knowledge so that it is:

- easy → to acquire, express, modify
- natural
- efficient → for reasoning purposes

Knowledge Representation = ? Formal Logic? Databases?

Formal knowledge representation

representing knowledge in a formal way so reasoning can be automated

Need formal way of

- using symbols to represent knowledge
- mapping formal "pieces of knowledge" to external reality
- inferring new fact from existing knowledge

Formal (classical) logic is one way we can do this, but not the only, or best way

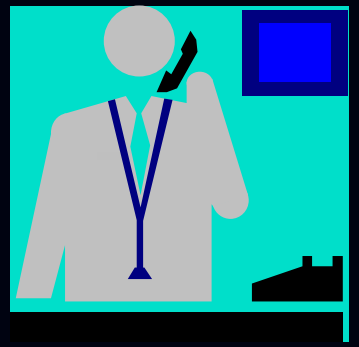
Trick: combine the best parts of formal logic with the best parts of other reasoning systems

Outline

- Example domain: benefits inquiry in medical insurance
- Issues in Representation
- FANs: Formula-Augmented Semantic Networks
- Other domains: Product Configuration,
  Intelligent Helpdesks
- FANs in general
- Remaining KR issues

Example Domain:
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

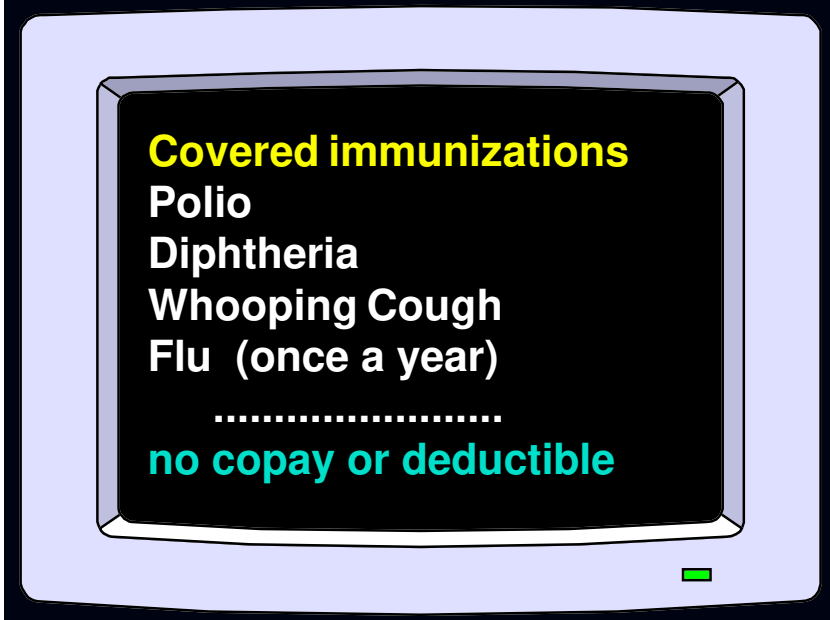
- many business regulations for each insurance product
- many insurance products
- insurance products constantly change

Benefits Inquiry: What had been done

- Nothing --- CSRs just wing it
- Placing charts and manuals on line
- “Desktop” text-based systems
- Code-based systems

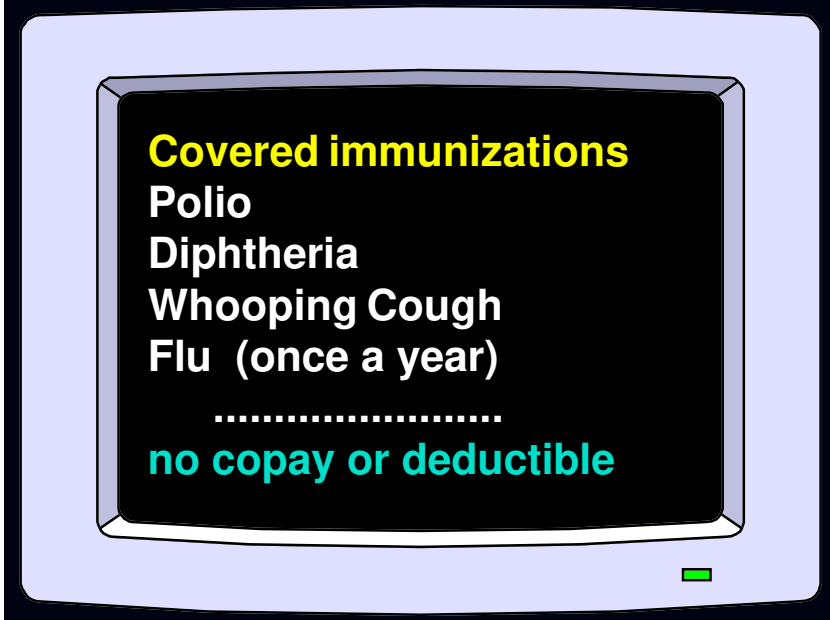
Benefits Inquiry: What had been done

- "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: 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

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

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

To be used by:

- customer service representatives
- policy modifers

Benefits Inquiry for medical insurance

We need to reason about

- **which medical services are covered / excluded**
- **which rules and regulations apply to medical services**

What type of representation?
What type of reasoning?

Two types of knowledge:

- business rules

rule-based systems

- taxonomic information (subclasses, superclasses)

inheritance systems

What type of reasoning to use?

- **Rule-based systems:**

Idea: write down all rules (as statements in classical logic)

Employed(Robert_Jones,IBM,1998) ==> Insured(Robert_Jones,1998)
Employed(Robert_Jones,IBM,1998)

(i,e)Service(e,Drugs,i) & Covered(e,i) ==> Copay(e,i) = $3

and then churn through theorem prover

Problems:

- **Efficiency problems**
- **Representation problems --- how do you talk about exceptions?**

E.g., how do you say
The copay is $3, except for experimental drugs?
and human growth hormone?

Nonmonotonic Logic

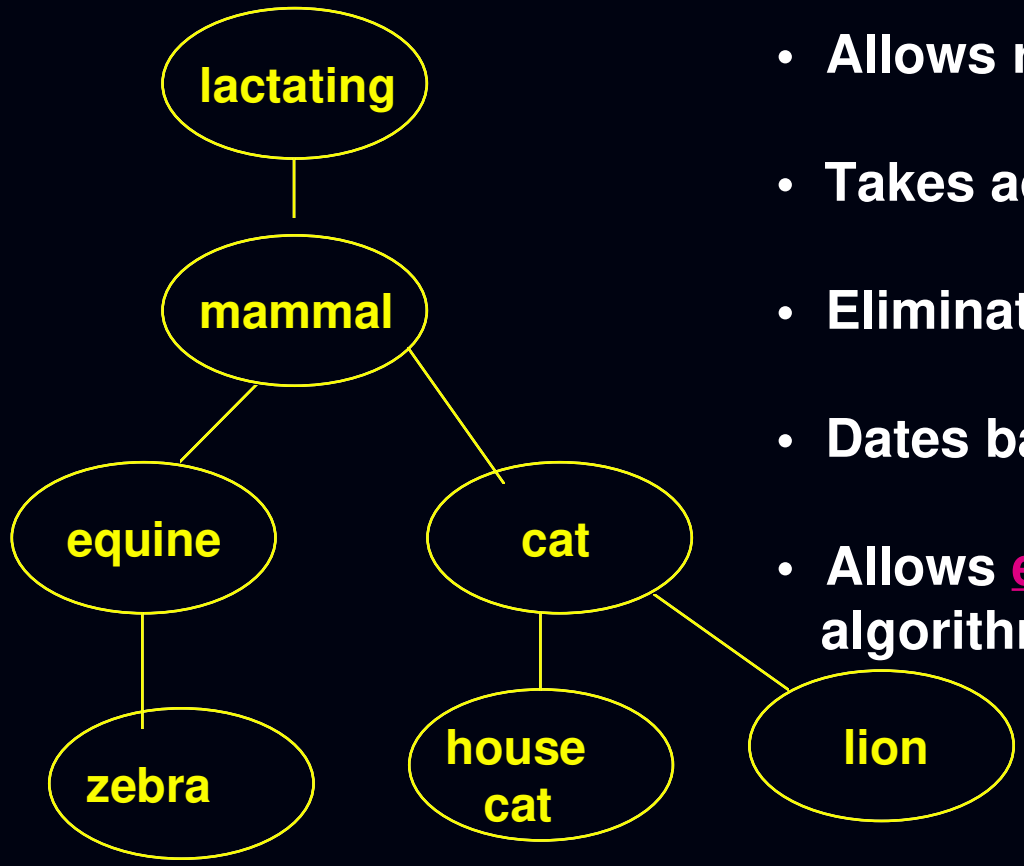
+ allows representation of exceptions
  so you can say
    all drugs are covered except for OTC drugs

+ gives a method for reasoning with exceptions
    ---- jump to conclusions with incomplete information
    ---- retract conclusions if necessary due to more info

-- but undecidable, intractable !
-- very tricky to formalize correctly
-- updating can be hard

Nevertheless, there is good news for some types of nonmonotonic reasoning:
inheritance & logic programming

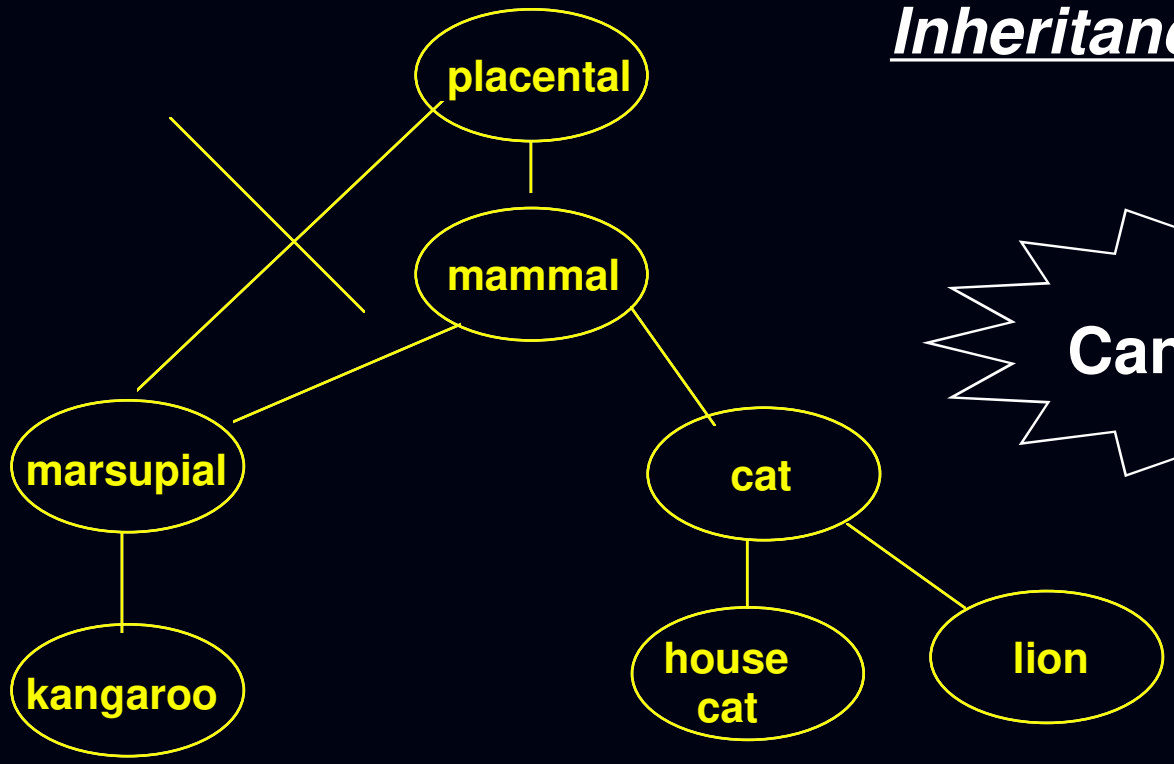
Inheritance Networks



- Standard monotonic inheritance
- Allows reasoning with sub,superclasses
- Takes advantages of commonalities
- Eliminates redundnacies
- Dates back to Aristotle
- Allows efficient, graph-based algorithms for reasoning

But what if there are exceptions?

Nonmonotonic inheritance network

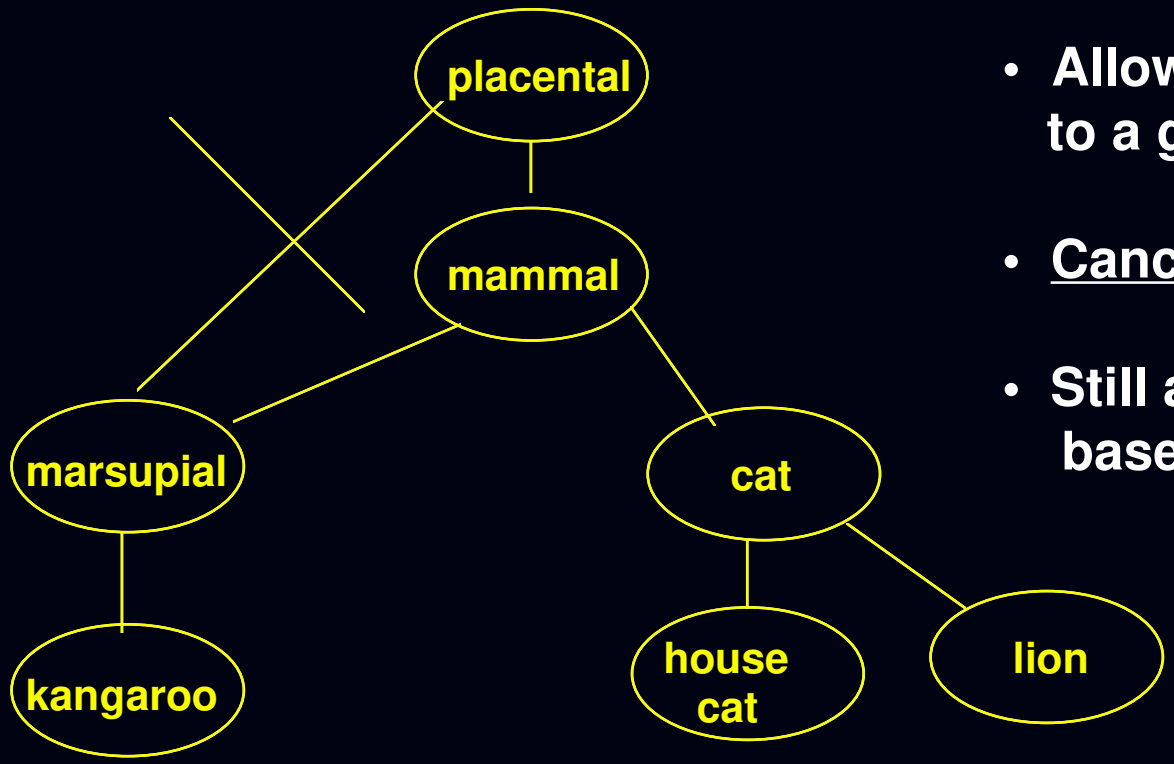


Inheritance with Exceptions

Cancels link

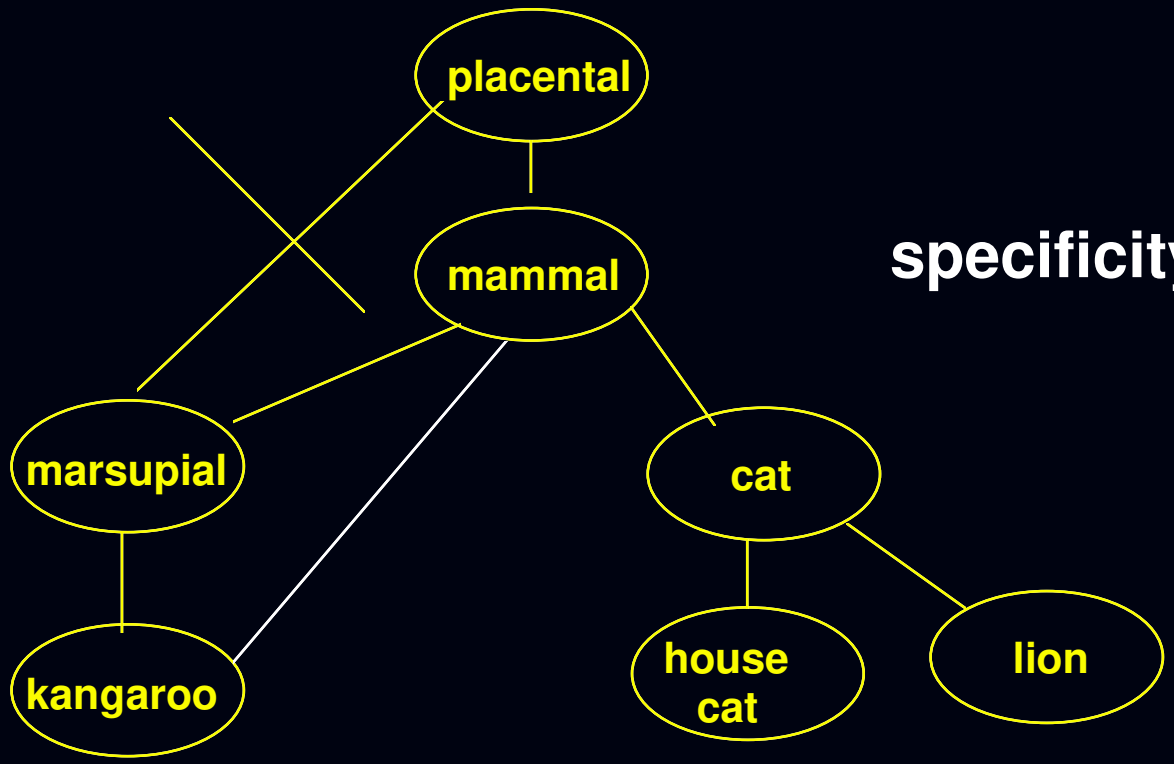
(adds links --> withdraw conclusions)

Inheritance with Exceptions



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

Inheritance with Exceptions



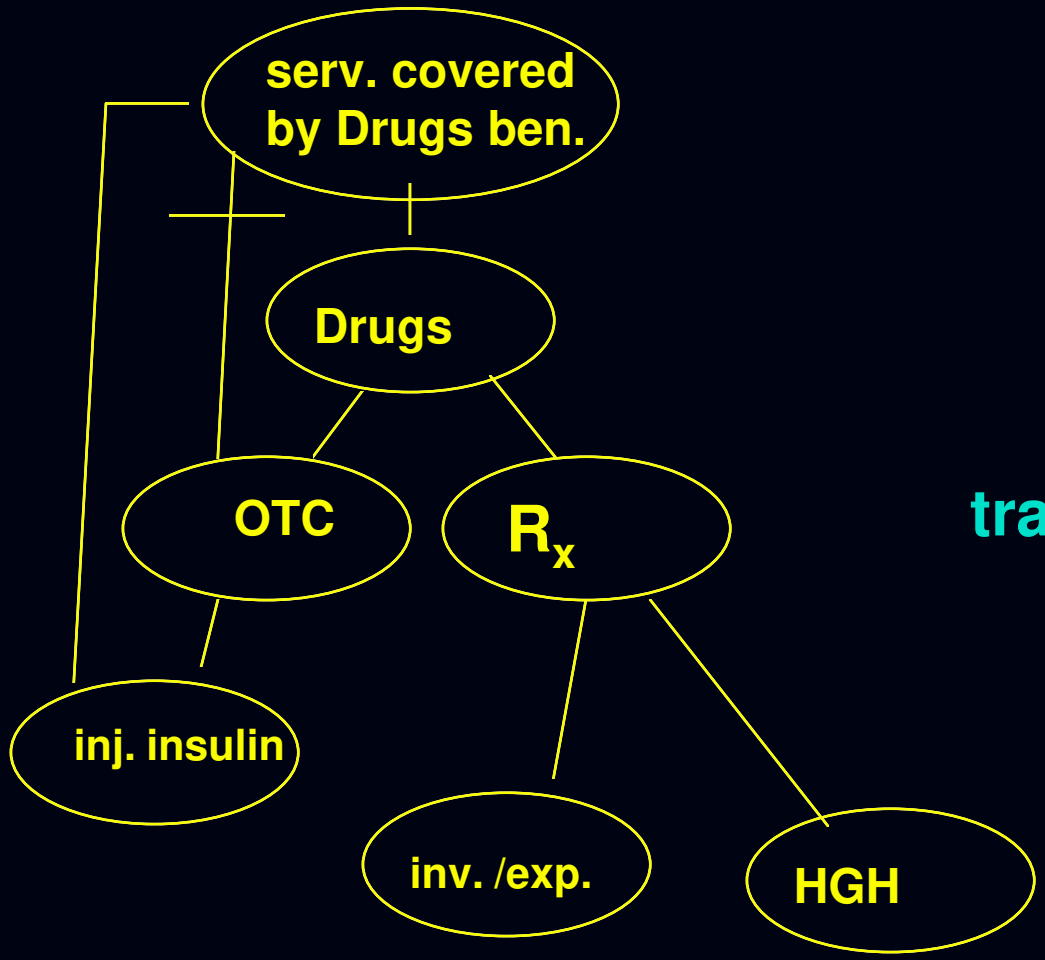
specificity vs. shortest-path

Benefits Inquiry for medical insurance:

Nonmonotonic inheritance is prevalent

Medical services: taxonomony with exceptions (plus)

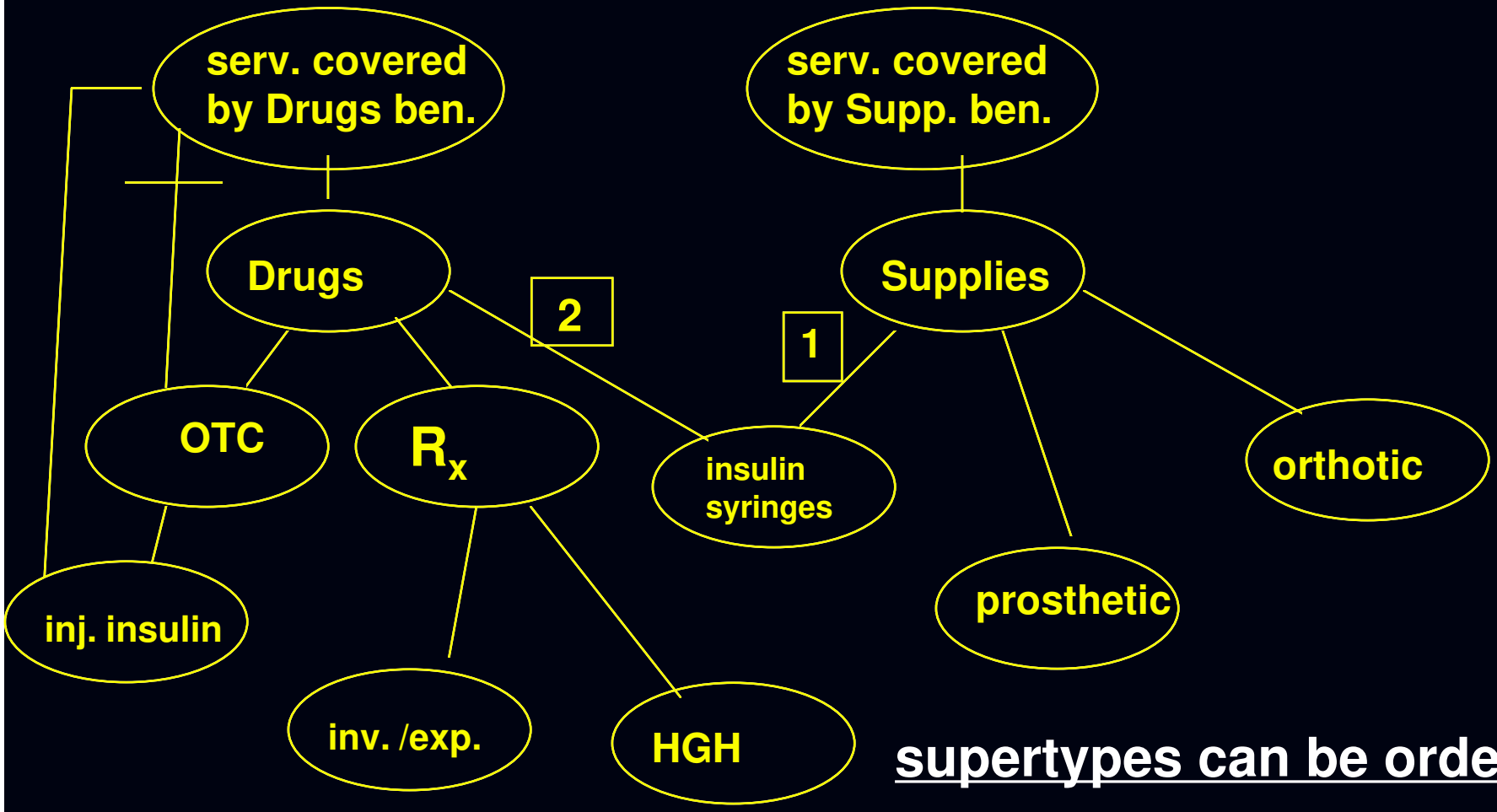
(Horty, Touretzky, Thomason; Stein)



tractable form of nonmon

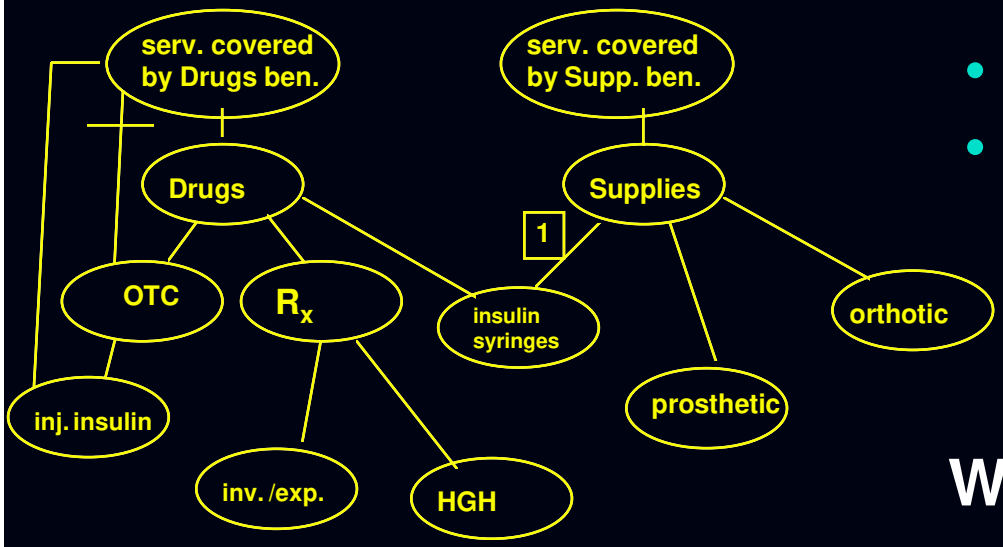
Benefits Inquiry for medical insurance:

Note that there are multiple supertypes: DAG, not tree



supertypes can be ordered

Benefits Inquiry



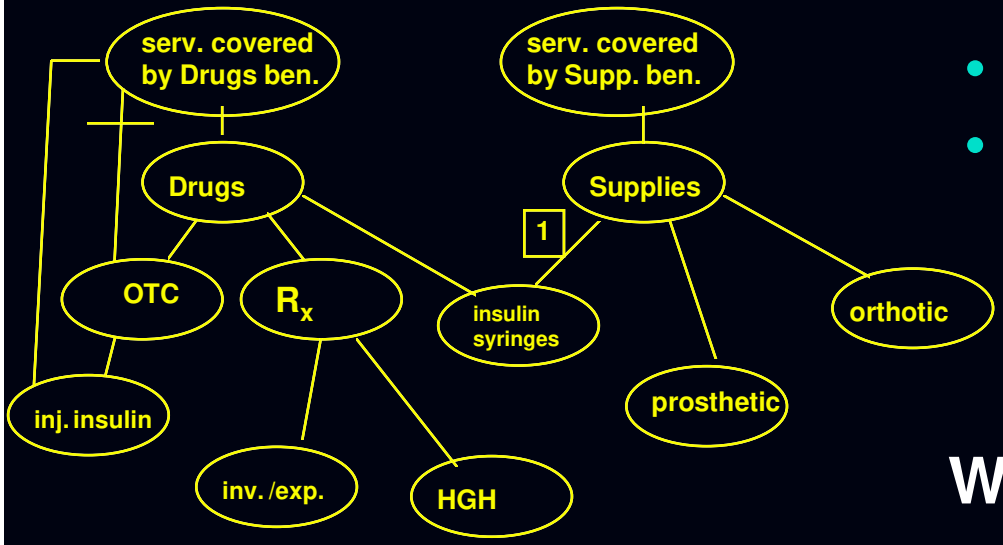
Inheritance with exceptions solves part of the problem:

- 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)

Two types of knowledge:

- taxonomic information (subclasses, superclasses)

  inheritance systems

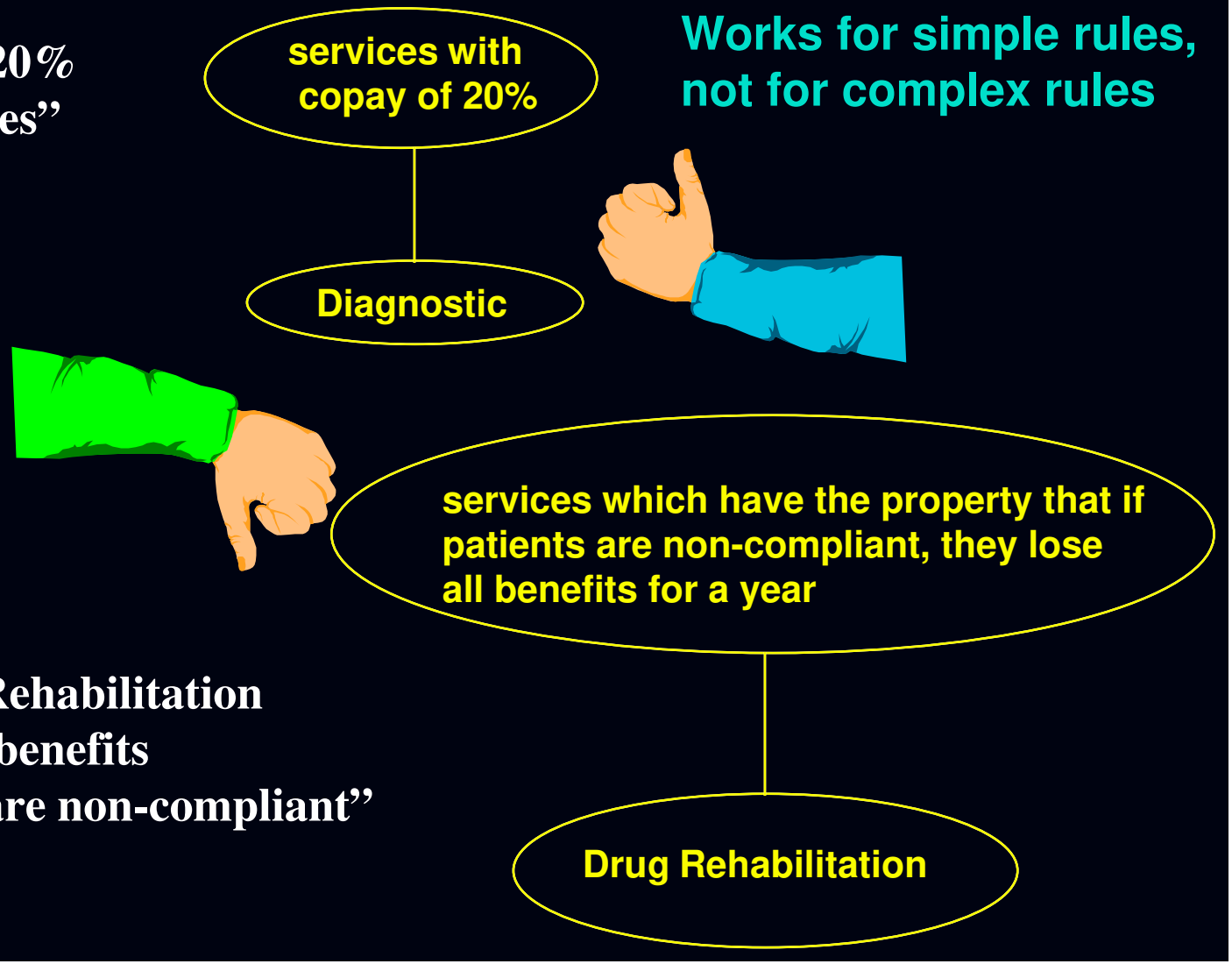
- business rules

  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"

How to represent business rules

- **in rule-based system? every rule written as a logical formula?**

--- hard to write down

--- intractable

--- lose all efficiency of semantic network

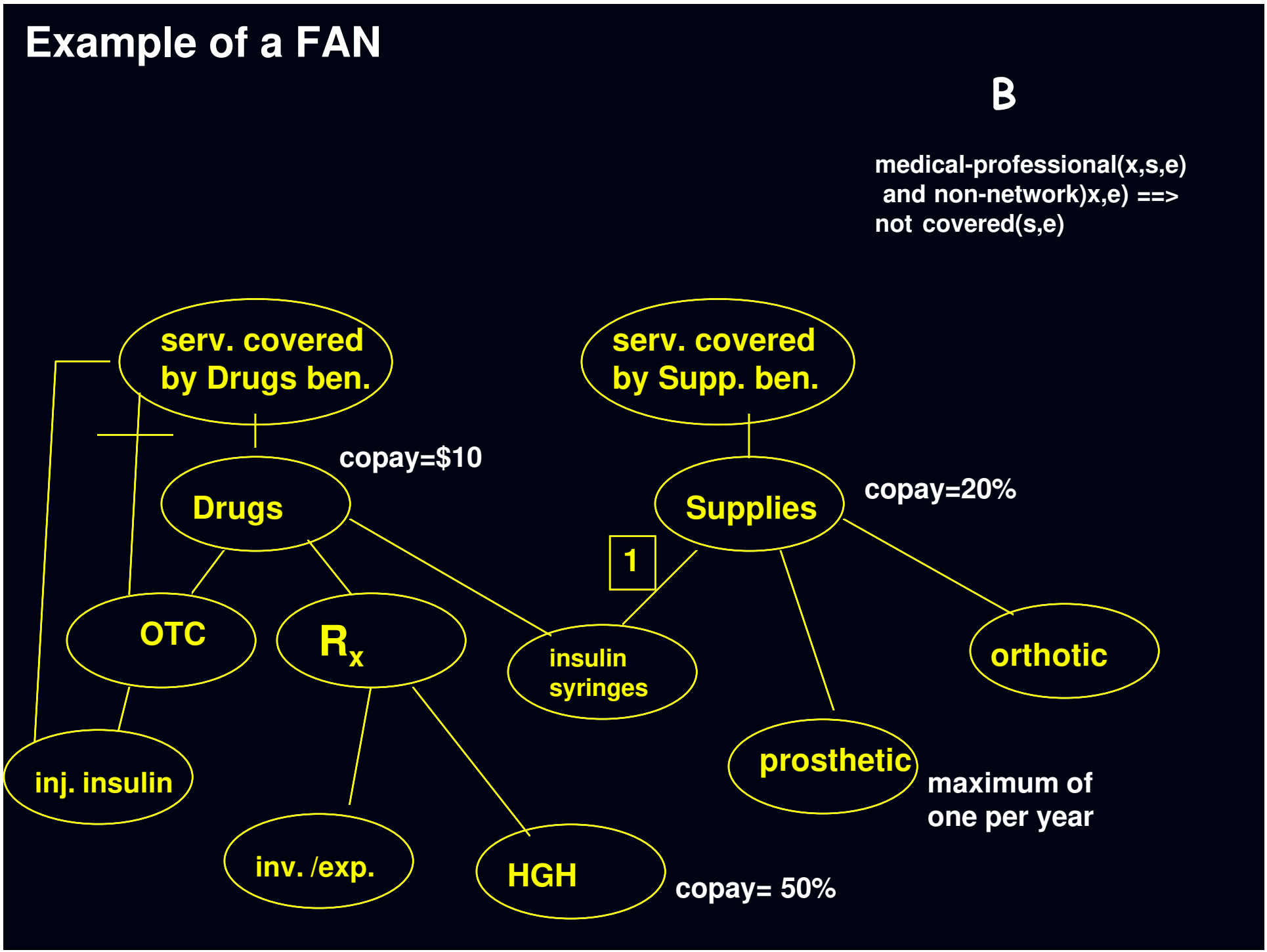
Alternative: Combine semantic net with rule-based system

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

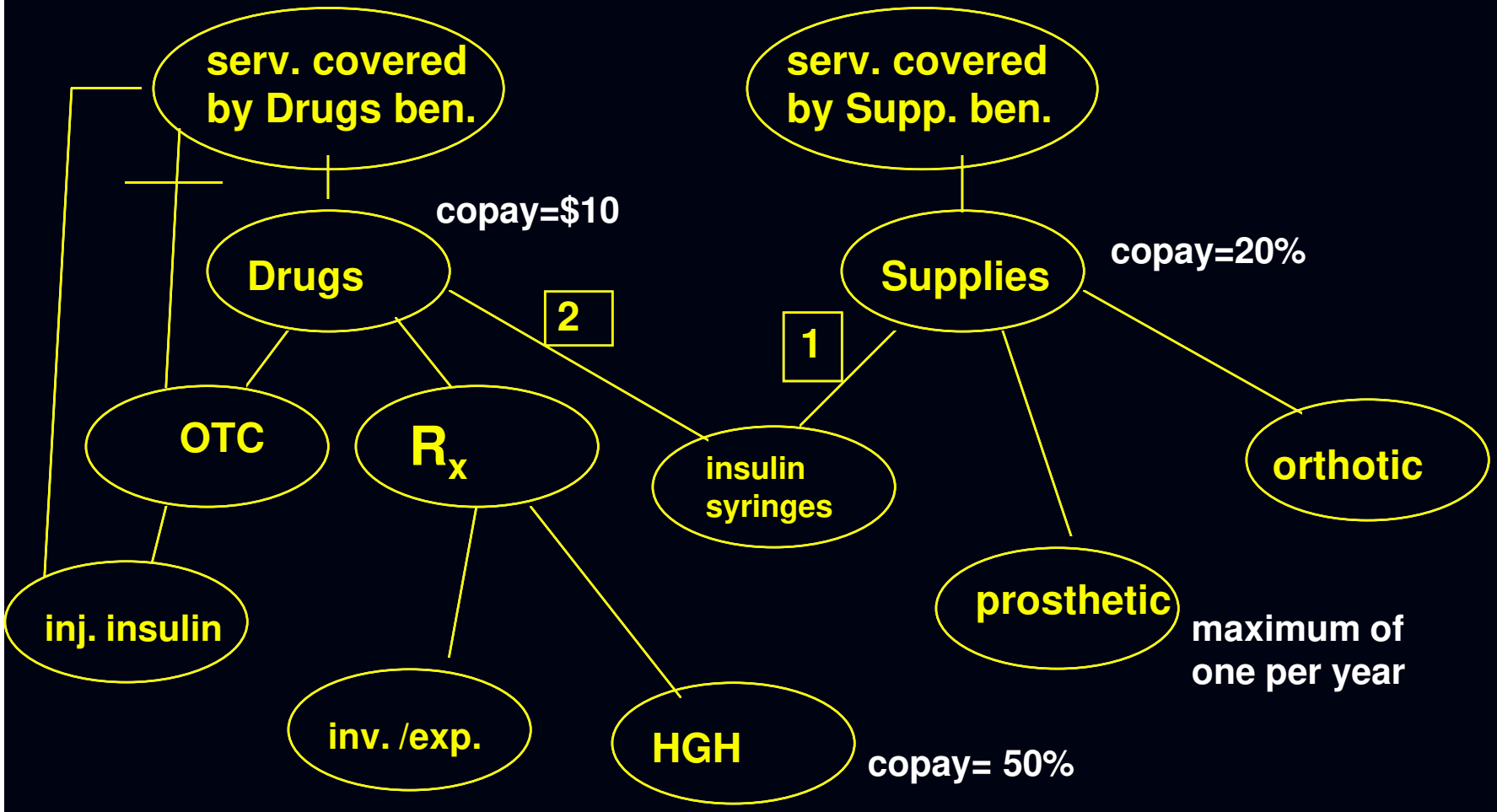


Note differences from semantic network:

- multiple parents
- order on multiple parents
- rules attached to nodes
- background rules

B

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

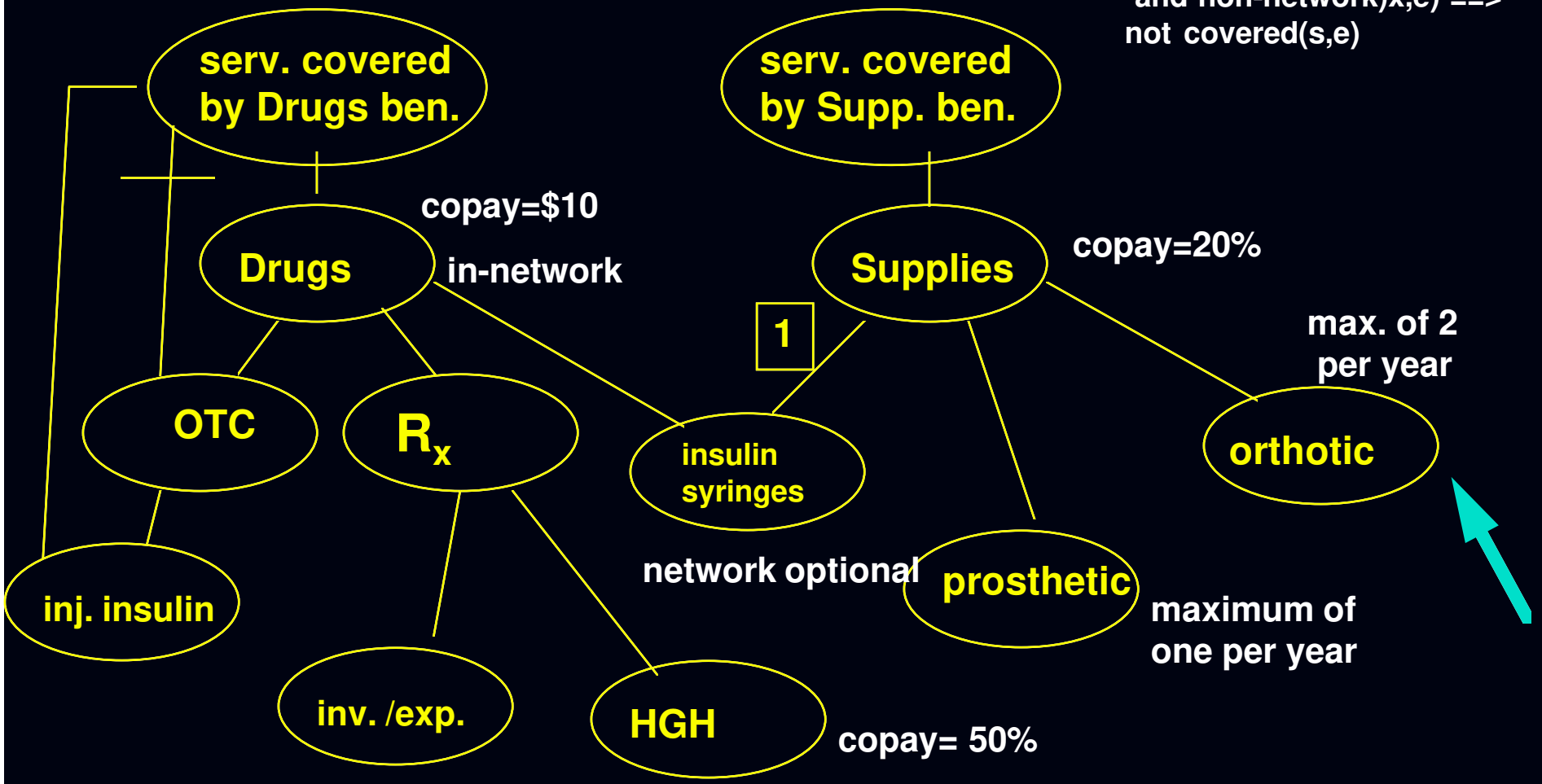


Reasoning in a FAN:
Determining which rules apply to a (focus) node

Wff-inheritance problem

B

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

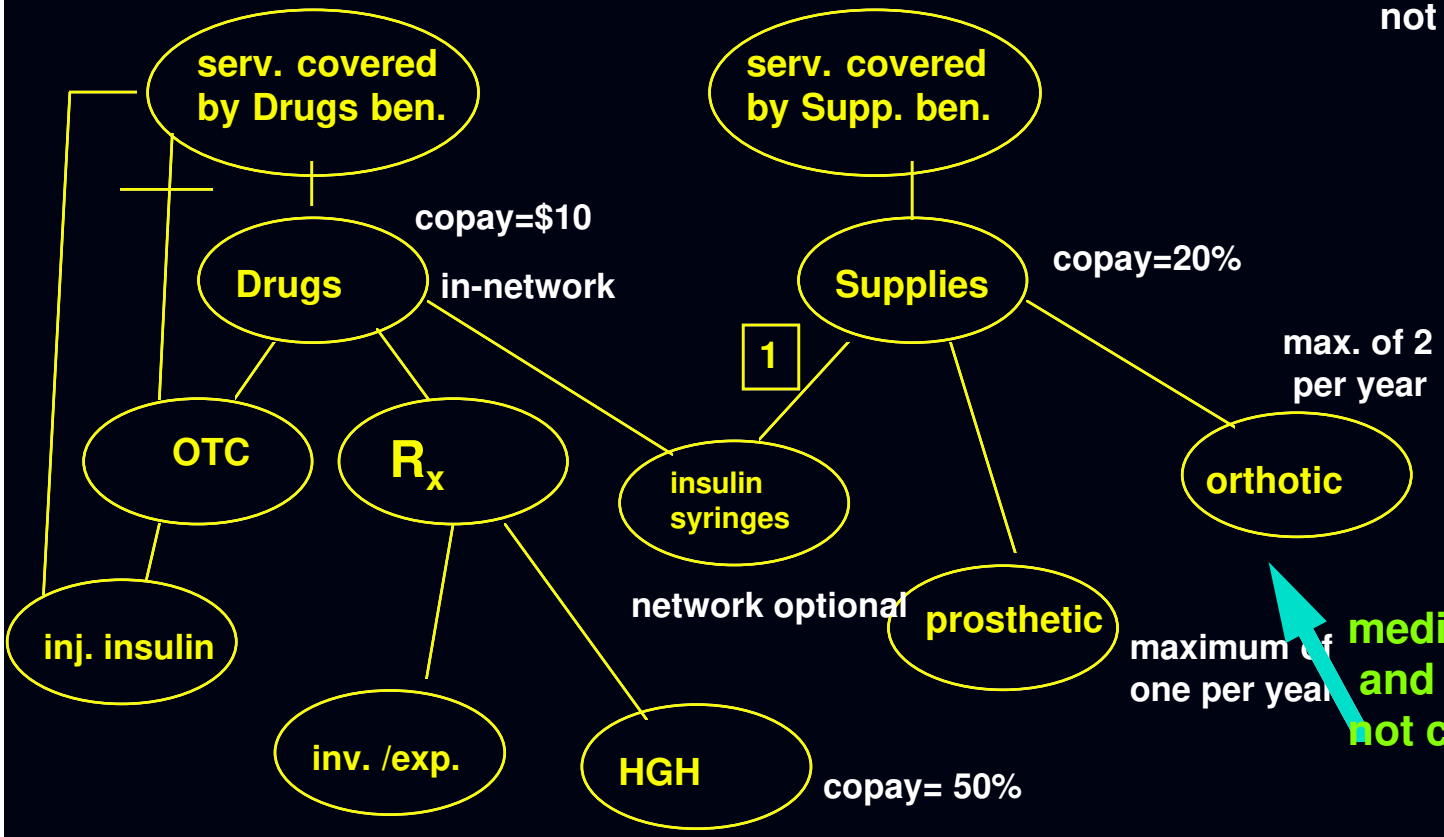


Reasoning in a FAN:
Determining which rules apply to a (focus) node

Wff-inheritance problem

B

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

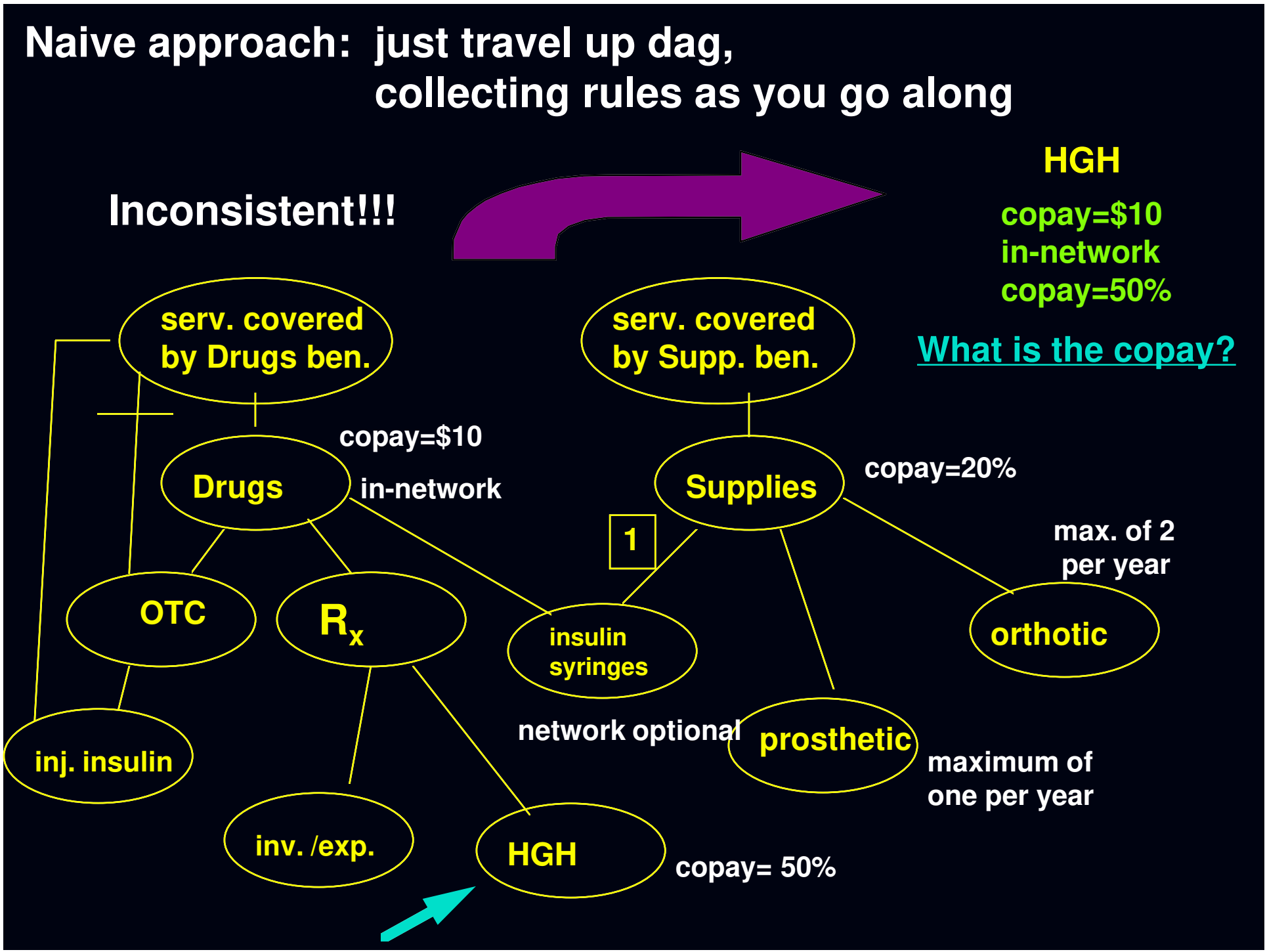


medical-professional(x,s,e) and non-network)x,e) ==> not covered(s,e)

copay=20%

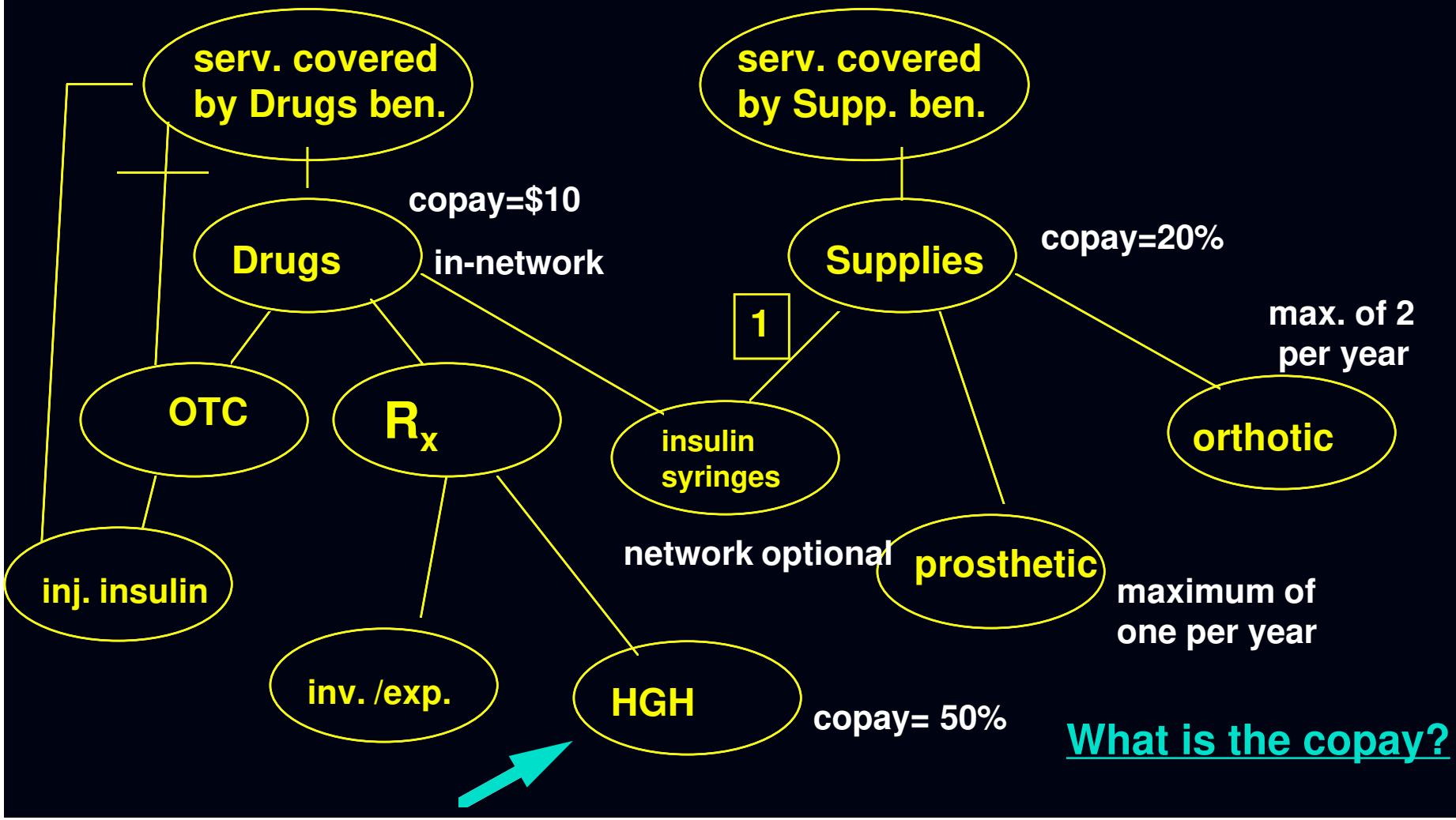
max of 2 per year

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

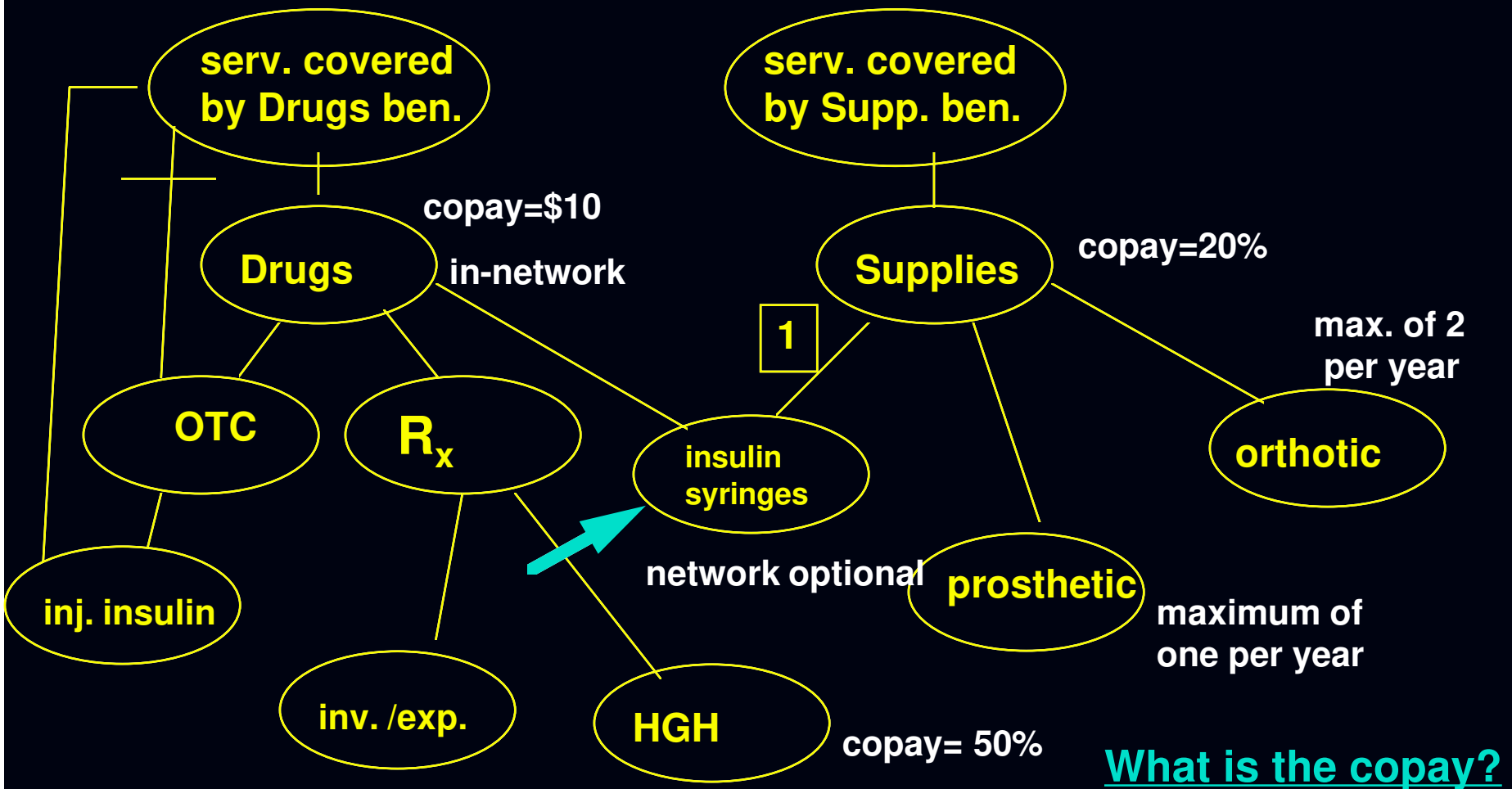


- Wff-inheritance: must take maximal consistent subset of rules
- more than 1 maximal subset; how to choose?
- preference criteria
    - specificity
    - path preference

Preference criterion of specificity



Preference criterion of path preference



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.

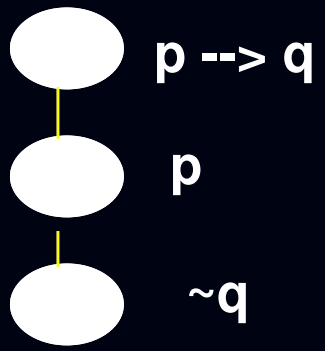
But this won't be very useful in practice.
We need

- **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

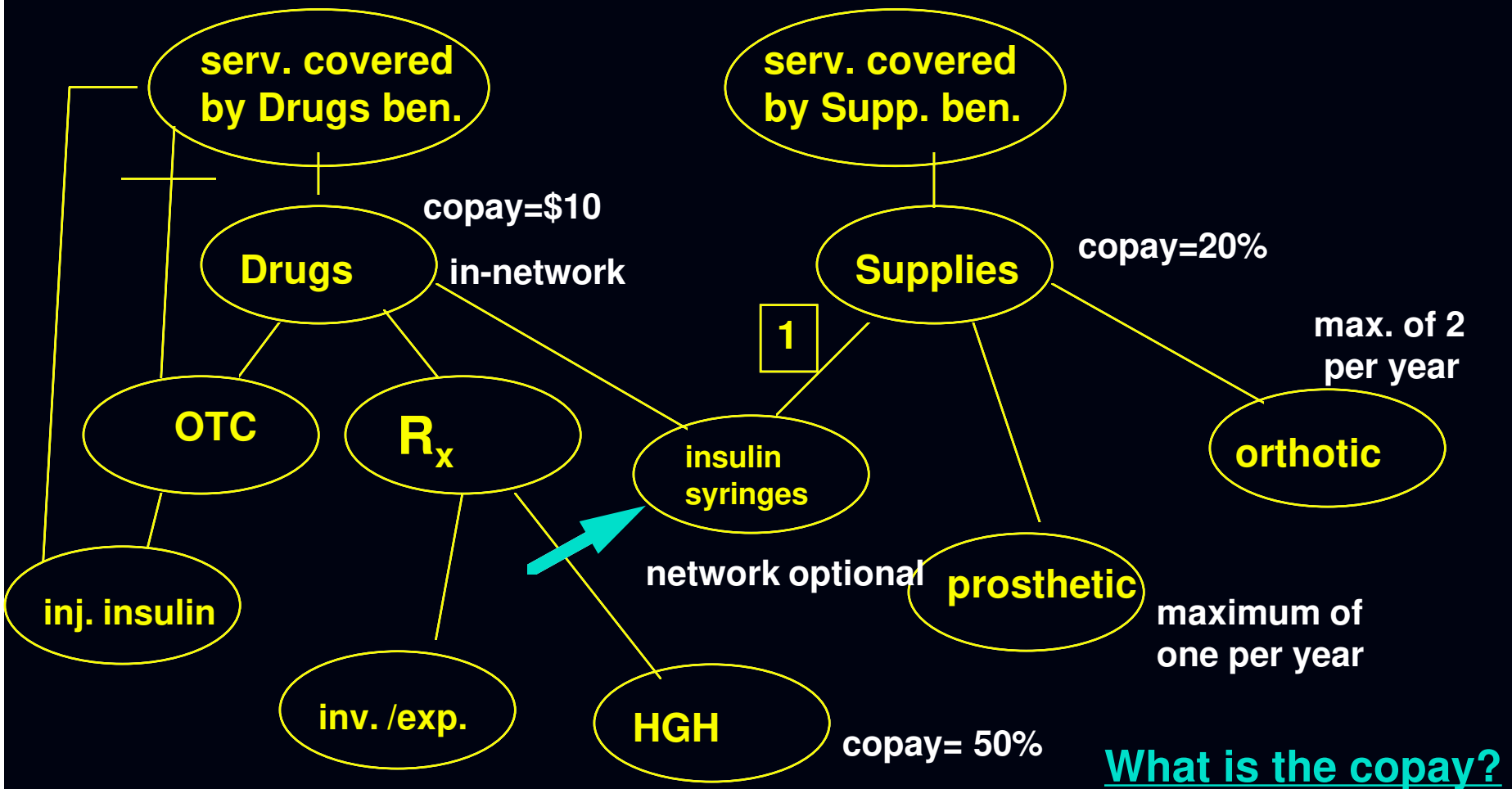


{p-->q,p} U {~q} inconsistent --- throw out p-->q or p?

- 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)

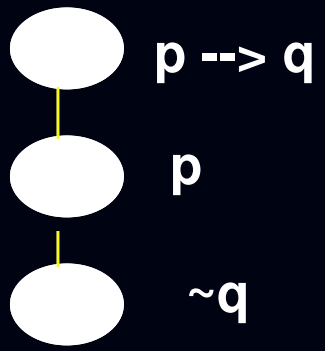
Preference criterion of path preference



The Algorithm:

- iterative traversal

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



{p-->q,p} U {~q} inconsistent --- throw out p-->q or p?

- 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 and other logic programming techniques

The Systems:

1. **Benftinq**

- developed for medical insurance company
- written in C (OS/2)
- included the following features:
  - ---- FAN structure, including rules at nodes
  - ---- included easy graphical user interface
  - ---- authoring tool for modifying network, rules
  - ---- limited rules inheritance
    (location, access, cost-share)
- rolled out to policy modifiers, CSRs
  - ---- great response

The Systems:

2.

BenInq

- all the functionality of Benftinq +
- had full rules inheritance (propositional)
- polynomial method of Kautz and Selman
- written in Smalltalk for purposes of collaborations with IBM Zurich
- used as basis for dynamic product configurator for life, property insurance

What wasn't solved for either system:

- Easing entering of rules into system

Domain experts (not knowledge engineers) entering arbitrary rules

The Knowledge Representation bottleneck

Need more research!!!

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

Example of such a domain: computer help desk

"How can I retrieve lost email?"
"Why is my 56.6 kbps modem receiving at 32 kbps?"
"How do I change my settings so I don't keep getting the IBM home page when Netscape comes up?"

What's necessary, for any domain:

- method of *categorizing*, *representing*, *updating*, *reasoning with* large amounts of information
- solutions to common problems readily available

Internet help desk

Sample questions (focus on email problems):

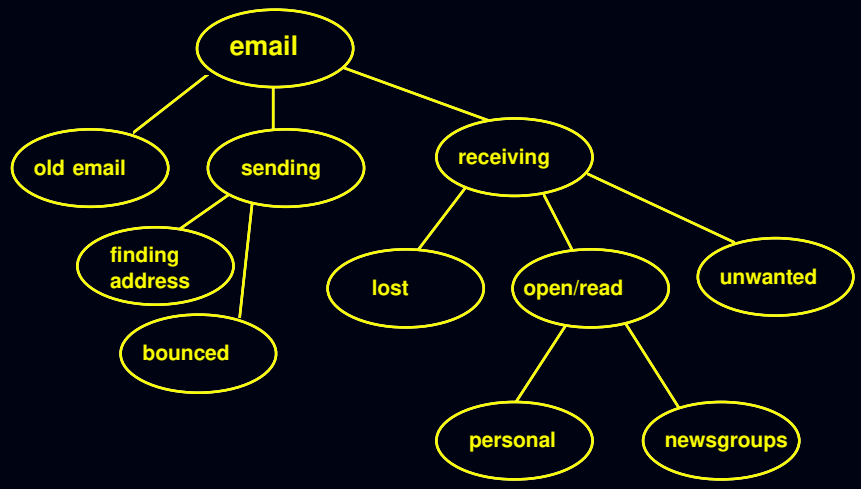
(source: log files, email to & from customer service reps, IBM Internet help desk 6/5/98-6/7/98)

"I lost all my new mail when the phone connection broke. Can I retrieve it?"
"I can't download email with 1 megabyte attachment."
"The system says I have mail, but nothing comes up"
"I can't delete my old email"
"I can't send email"

- Many different problems
- Common features, common (partial) solutions
  e.g. proper settings, connected through IBM dialer ...
- Need tool that takes advantage of commonalities

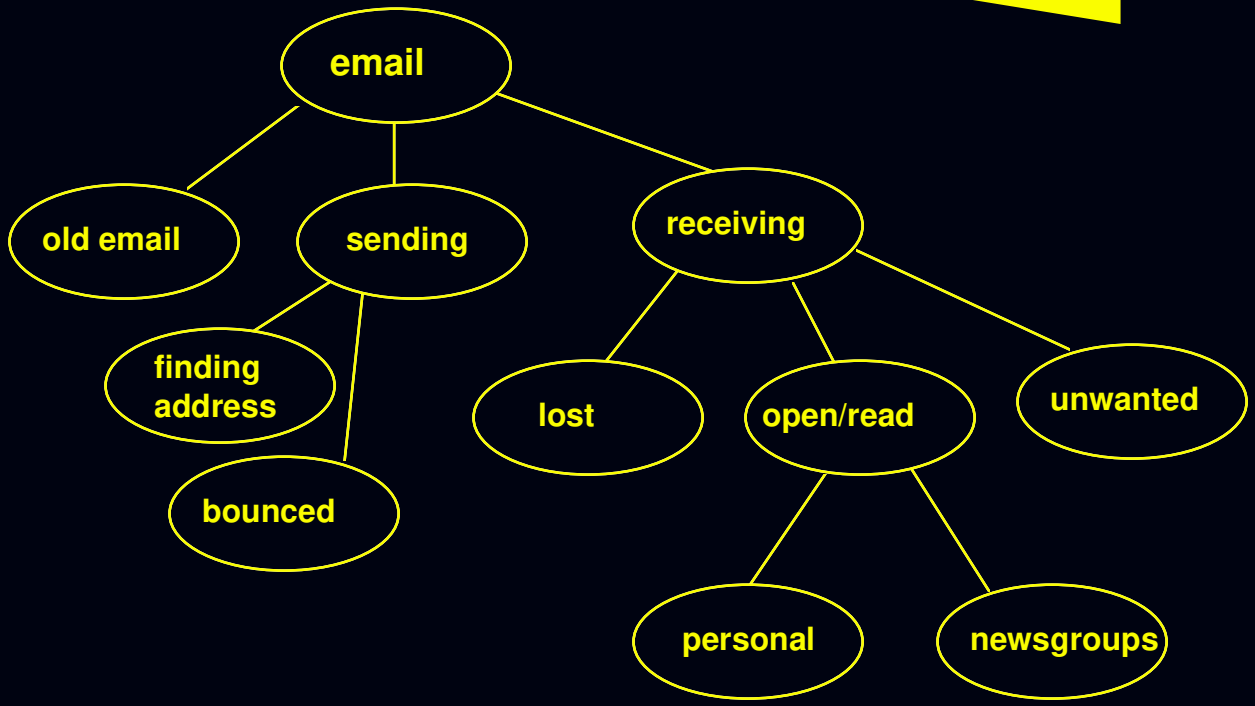
FANs

Formula-Augmented Semantic Networks



The help desk domain

a way of organizing the cases in a case-based reasoning system



exceptions in a FAN:
in general, must be connected through IBM dialer to do anything with email,
but can read personal mail (not newsgroups) even if connected through LAN

Generalizing Benefits Inquiry: Similar Techniques

Potential Domains:

- Legal Reasoning
- Medical Reasoning
- Reasoning in Business Organizations

FANs aren't a magic bullet

- domain may not be a FANs domain
- FANs may not solve the hard part of the problem

The other parts of the problem:

- ontology for networks --- basic concepts/nodes
- learning networks --- Bayesian techniques?
- representation of business rules
  - ----- knowledge engineering
  - ----- formal languages (ontology, etc.)
  - ----- mapping between natural lang. & formal languages

knowledge rep bottleneck

Addressing these problems will improve FANs