

# 学習での知識

Chapter 19

# 学習の論理的形式

- ゴールと仮説はなにか
  - ゴールは質問の予測- WillWait
- 事例の類別と同値な関係をもたらず論理式を見出すことが学習
- 各仮説は、質問の候補となる定義としての式を提案する
  - $\forall r \text{ WillWait}(r) \Leftrightarrow \text{Pat}(r, \text{Some}) \vee$   
 $\text{Pat}(r, \text{Full}) \wedge \text{Hungry}(r) \wedge \text{Type}(r, \text{French}) \wedge$   
...

# 学習の論理的形式

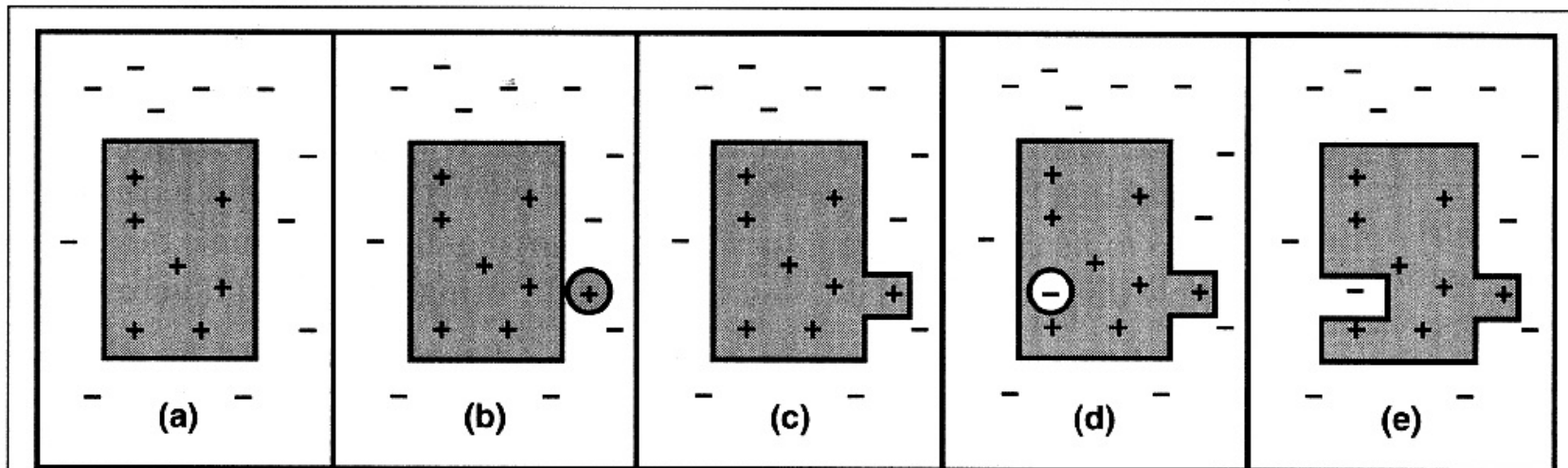
- 仮説空間は全ての仮説の集合であって、仮説の集合を受け入れるように学習のアルゴリズムは設計されている。
- 仮説の一つは正しい:  
 $H_1 \vee H_2 \vee \dots \vee H_n$
- 各 $H_i$  は事例の有る集合を予知する- ゴールの述語の拡張
- 異なった拡張である二つの仮説はお互いに論理的に一貫していないか、あるいは、論理的に同値である

# 事例は何か

- 事例は論理的な記述であり、ゴールの概念はその記述に適合したりあるいはしなかったりする。
  - $\text{Alt}(X1) \wedge \neg \text{Bar}(X1) \wedge \neg \text{Fri/Sat}(X1) \wedge \dots$
- 理想的には全ての事例に一致する仮設を見つきたい
- $f$  と  $h$  の関係: ++, --, +- (false negative), -+ (false positive). 最後の二つでは  $f$  と  $h$  は一貫性がない

# 現時点での最善の仮説探索

- 一つの仮説を維持する
- 新しい事例がきたとき、一貫性があるように調整する (Fig 19.1)
  - 肯定の事例の一般化
  - 否定の事例の特殊化



**Figure 18.10** (a) A consistent hypothesis. (b) A false negative. (c) The hypothesis is generalized. (d) A false positive. (e) The hypothesis is specialized.

# 一般化

- 定義: 仮説 $H_1$  は仮説  $H_2$  の一般化であるときかつそのときに限り仮説 $H_1$ の全ての節 $C_1$ と仮説 $H_2$ の全ての節 $C_2$ とに対して $\forall x C_2(x) \Rightarrow C_1(x)$
- 節 $C_2$ を含むように $H_1$ を一般化するためには  $C_2 \Rightarrow C_1$ であるような新しい節 $C_1$ を見つけ出す。これは選言となっている $C_2$ から一つ以上のリテラルを捨てることで実現される

$C_2(x)$  is *Alternate(x)  $\wedge$  Patrons(x, Some)*

then  $C_1(x)$  can be *Patrons(x, Some)*

# 特殊化

- 定義: 仮説 $H_1$  は仮説  $H_2$  の特殊化であるときかつそのときに限り仮説 $H_1$ の全ての節 $C_1$ と仮説 $H_2$ の全ての節 $C_2$ とに対して $\forall x C_1(x) \Rightarrow C_2(x)$
- 節 $C_2$ を含むように $H_1$ を特殊化するためには  $C_1 \Rightarrow C_2$ であるような新しい節 $C_1$ を見つけ出す。これは連言となっている $C_2$ から一つ以上のリテラルを除くことで実現される

$C_2(x)$  is  $Patrons(x, Some) \vee Alternate(x)$

then  $C_1(x)$  can be  $Patrons(x, Some)$

# 一般化と特殊化

- 関係:

- $a$ は $(a \wedge b)$ の一般化である  $(a \wedge b) \Rightarrow a$
- $(a \wedge b)$ は $a$ の特殊化である  $(a \wedge b) \Rightarrow a$
- $a$ は $(a \vee b)$ の特殊化である  $a \Rightarrow (a \vee b)$
- $(a \vee b)$ は $a$ の一般化である  $a \Rightarrow (a \vee b)$
- Note:  $b$  can also be negated  $(a \wedge \sim b) \Rightarrow a$

- 特殊化と一般化は対称的で推移的である
- それぞれの操作に対していくつかの選択がある

# 現時点での最善の仮説探索

- アルゴリズム (Fig 19.2, page 681)
  - 新しい事例をとるごとに、これまでの全ての事例と合わせて一貫性が保たれるかを調べる必要がある

```
function Current-Best-Learning(examples) returns hypothesis  $H$   
   $H :=$  hypothesis consistent with first example  
for each remaining example  $e$  in examples do  
  if  $e$  is false positive for  $H$  then  
     $H :=$  choose a specialization of  $H$  consistent with examples  
  else if  $e$  is false negative for  $H$  then  
     $H :=$  choose a generalization of  $H$  consistent with examples  
  if no consistent generalization/specialization found then fail  
end  
return  $H$ 
```

# 席を待つ例

- Fig 18.3 for Current-Best-Learning

Example	Attributes										Target
	<i>Alt</i>	<i>Bar</i>	<i>Fri</i>	<i>Hun</i>	<i>Pat</i>	<i>Price</i>	<i>Rain</i>	<i>Res</i>	<i>Type</i>	<i>Est</i>	<i>Wait</i>
$X_1$	T	F	F	T	Some	\$\$\$	F	T	French	0-10	T
$X_2$	T	F	F	T	Full	\$	F	F	Thai	30-60	F
$X_3$	F	T	F	F	Some	\$	F	F	Burger	0-10	T
$X_4$	T	F	T	T	Full	\$	F	F	Thai	10-30	T
$X_5$	T	F	T	F	Full	\$\$\$	F	T	French	>60	F
$X_6$	F	T	F	T	Some	\$\$	T	T	Italian	0-10	T
$X_7$	F	T	F	F	None	\$	T	F	Burger	0-10	F
$X_8$	F	F	F	T	Some	\$\$	T	T	Thai	0-10	T
$X_9$	F	T	T	F	Full	\$	T	F	Burger	>60	F
$X_{10}$	T	T	T	T	Full	\$\$\$	F	T	Italian	10-30	F
$X_{11}$	F	F	F	F	None	\$	F	F	Thai	0-10	F
$X_{12}$	T	T	T	T	Full	\$	F	F	Burger	30-60	T

- 問題: 非決定的、簡便で正しいことの保証がない、バックトラックを必要とする

# 席を待つ例

1.  $X_1$ : positive example, select one literal

$$H_1: \forall x \text{ WillWait}(x) \iff \text{Alternate}(x)$$

2.  $X_2$ : negative example,  $H_1$  is false positive, specialize

$$H_2: \forall x \text{ WillWait}(x) \iff \text{Alternate}(x) \wedge \text{Patrons}(x, \text{Some})$$

3.  $X_3$ : positive example,  $H_2$  is false negative, generalize

$$H_3: \forall x \text{ WillWait}(x) \iff \text{Patrons}(x, \text{Some})$$

4.  $X_4$ : positive example,  $H_3$  is false negative, generalize

$$H_4: \forall x \text{ WillWait}(x) \iff \text{Patrons}(x, \text{Some})$$

$$\vee [\text{Patrons}(x, \text{Full}) \wedge \text{Fri/Sat}(x)]$$

$$H_4': \forall x \text{ WillWait}(x) \iff \sim \text{WaitEstimate}(x, 30-60)$$

$$H_4'': \forall x \text{ WillWait}(x) \iff \text{Patrons}(x, \text{Some})$$

$$\vee [\text{Patrons}(x, \text{Full}) \wedge \text{WaitEstimate}(x, 10-30)]$$

etc...

# 最小決定探索

- $h$ を最善の推量として保持することには問題がある。従って、できるだけ可能なものとして保持することが可能か
- バージョン空間 (候補消去) アルゴリズム
  - 漸次的増加
  - 最小関与
- 間隔から領域集合へ
  - G-setとS-set
    - $S_0$  – なにも含んでいない最も特殊な集合  $\langle 0, 0, \dots, 0 \rangle$
    - $G_0$  – 全てを含んでいる最も一般的な集合  $\langle ?, ?, \dots, ? \rangle$
  - 間にある全ては事例と一致することが保証されている
- バージョン空間は漸次的増加で $S_0$ を一般化し $G_0$ を特殊化する

# アルゴリズム

**function** Version-Space-Learning(*examples*)

**returns** version space  $V$

$V :=$  the set of all hypothesis

**for each** example  $e$  in *examples* **do**

**if**  $V$  is not empty **then**

$V :=$  Version-Space-Update( $V, e$ )

**end return**  $V$

**function** Version-Space-Update( $V, e$ )

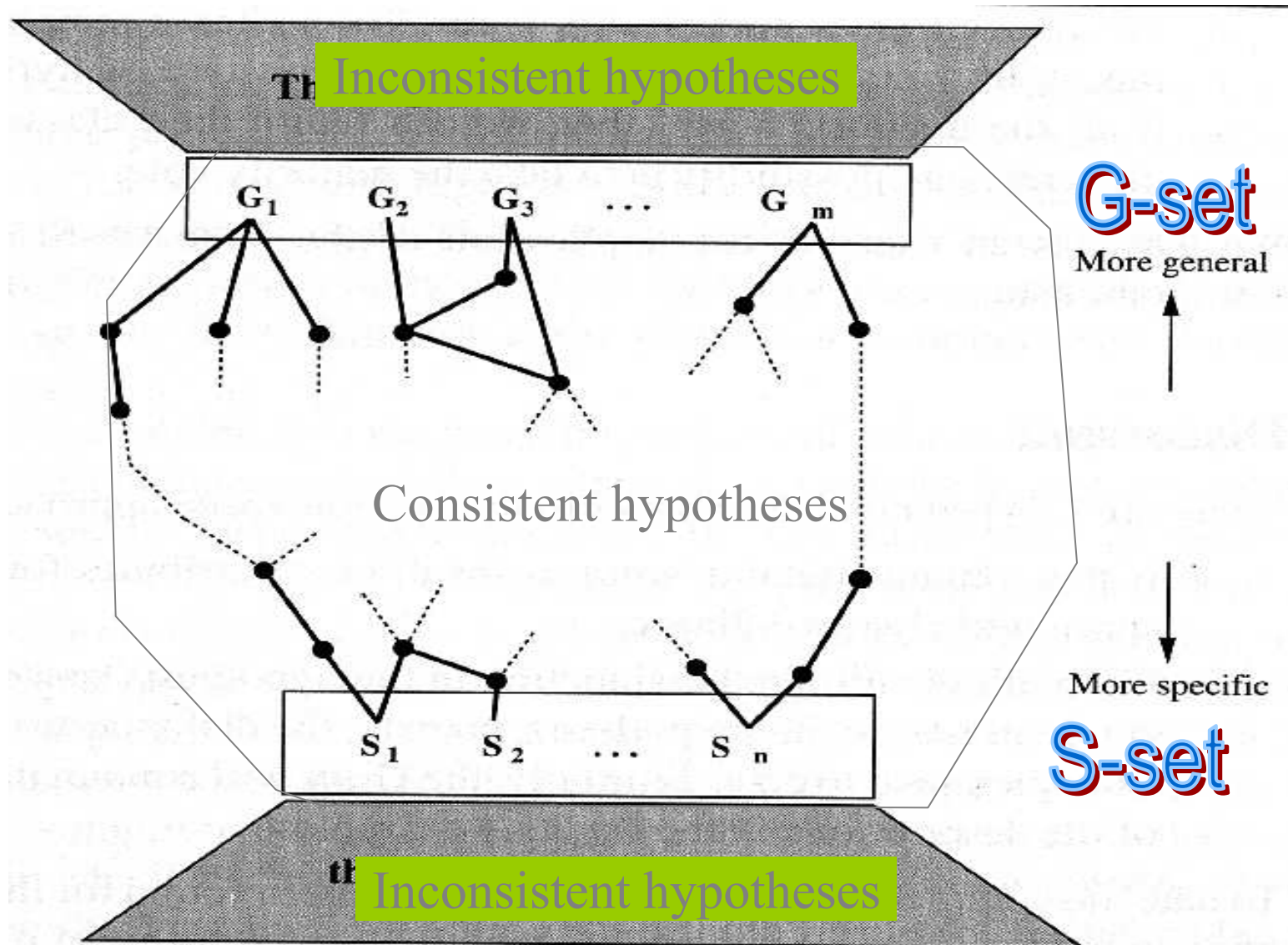
**returns** updated version space  $V'$

$V' := \{ h \text{ in } V : h \text{ is consistent with } e \}$

# バージョン空間

- Tom Mitchellの本にある4つの場合による例
- 一般化と特殊化 (次のスライド)
  - $S_i$ に対してfalse positiveは余りにも一般的なので、それを廃棄する。
  - $S_i$ に対してfalse negativeは余りにも特殊なので、それを最小限一般化する
  - $G_i$ に対してfalse positiveは余りにも一般的なので、それを最小限特殊化する
  - $G_i$ に対してfalse negativeは余りにも特殊なので、それを廃棄する。
- 停止のときは
  - 一つの概念になったとき ( $S_i = G_i$ )
  - バージョン空間が壊れたとき ( $G$ が  $S$ よりも特殊になったなど)
  - 事例が尽きた
- 主要な問題: 雑音を扱えない

# バージョン空間



# バージョン空間

```
function Version-Space-Learning(examples)  
    returns boundary version spaces (G-set, S-set)  
G-set := {True}; S-set := {False}  
for each example e in examples do  
    if G-set and S-set are empty return Fail  
    else (G-set, S-set) :=  
        Version-Space-Boundary-Update(e, G-set, S-set)  
end  
return (G-set, S-set)
```

```
function Version-Space-Boundary-Update(e, G-set, S-set)  
    returns updated boundary version spaces (G-set, S-set)  
    (see next slide)
```

# 予備知識を使う

- 決定木と論理的記述学習に対しては予備知識を想定していなかった
- 予備知識を持っているとき、それをどう扱うのか
- 関数学習に対応して論理的な定式化を必要とする

# 論理設定での帰納的学習

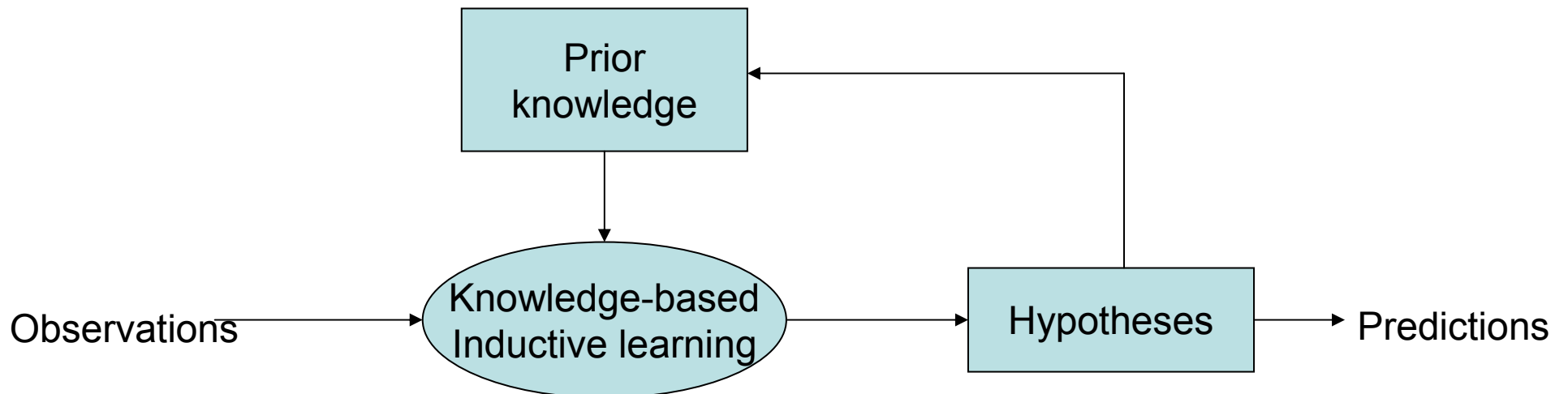
- 目標は仮説を見つけることである。その仮説は記述が与えられたとき事例の類別を説明する

*Hypothesis*  $\wedge$  *Description*  $\models$  *Classifications*

- 記述 – 全ての事例の記述の連言
- 類別 – 全ての事例の類別の連言

# 累積的学習過程

- 新しい手法は既に何かを知っていて、さらに何かを学ぼうとしているエージェントを設計すること
- 直感的には、*既に知っていることは正しいという前提を置けば*、知識を使わない方法よりも速く、よりよい方法である。
- 漸次的増加の知識にあわせてこの累積的学習をどのように実現したらよいのか



# 知識を使う例

- たった一つの観察から一般的な結論に到達すること
  - そのような経験はないか
- ブラジルへの旅行は: 言語と名前
  - ?
- 薬理的には無知だが診断的には高度な知識を有する医学生...
  - ?

# 一般的なスキーム

- 説明に基づいた学習 (EBL)
  - $Hypothesis \wedge Description \models Classifications$
  - $Background \models Hypothesis$ 
    - 事例からは事実上何も学ばない
- 関係に基づいた学習 (RBL)
  - $Hypothesis \wedge Descriptions \models Classifications$
  - $Background \wedge Descriptions \wedge Classifications \models Hypothesis$ 
    - 本質的には演繹的
- 知識に基づいた帰納的学習 (KBIL)
  - $Background \wedge Hypothesis \wedge Descriptions \models Classifications$

# 帰納的論理プログラミング

- 帰納的論理プログラミングは仮説を汎用的な一階述語論理に定式化する
  - － 決定木と同じように他はもっと制約された言語である
- 背景知識は学習の複雑さを減少させるために使われる:
  - － 背景知識はさらに仮説空間を縮小させる
  - － 背景知識は小さい仮説空間を見つけるのに役立つ
  - － これらは、背景知識が正しいことを前提にしている

# 説明に基づいた学習

- 個々の観察から一般的な法則を抽出する方法
- 似たような問題をその次はもっと速く解けることがゴール
- 記憶化 – 結果を蓄積することでスピードアップを図り、スクラッチから問題を解くことを避ける
- 説明に基づいた学習は観察から法則を得ることで、歩を進める

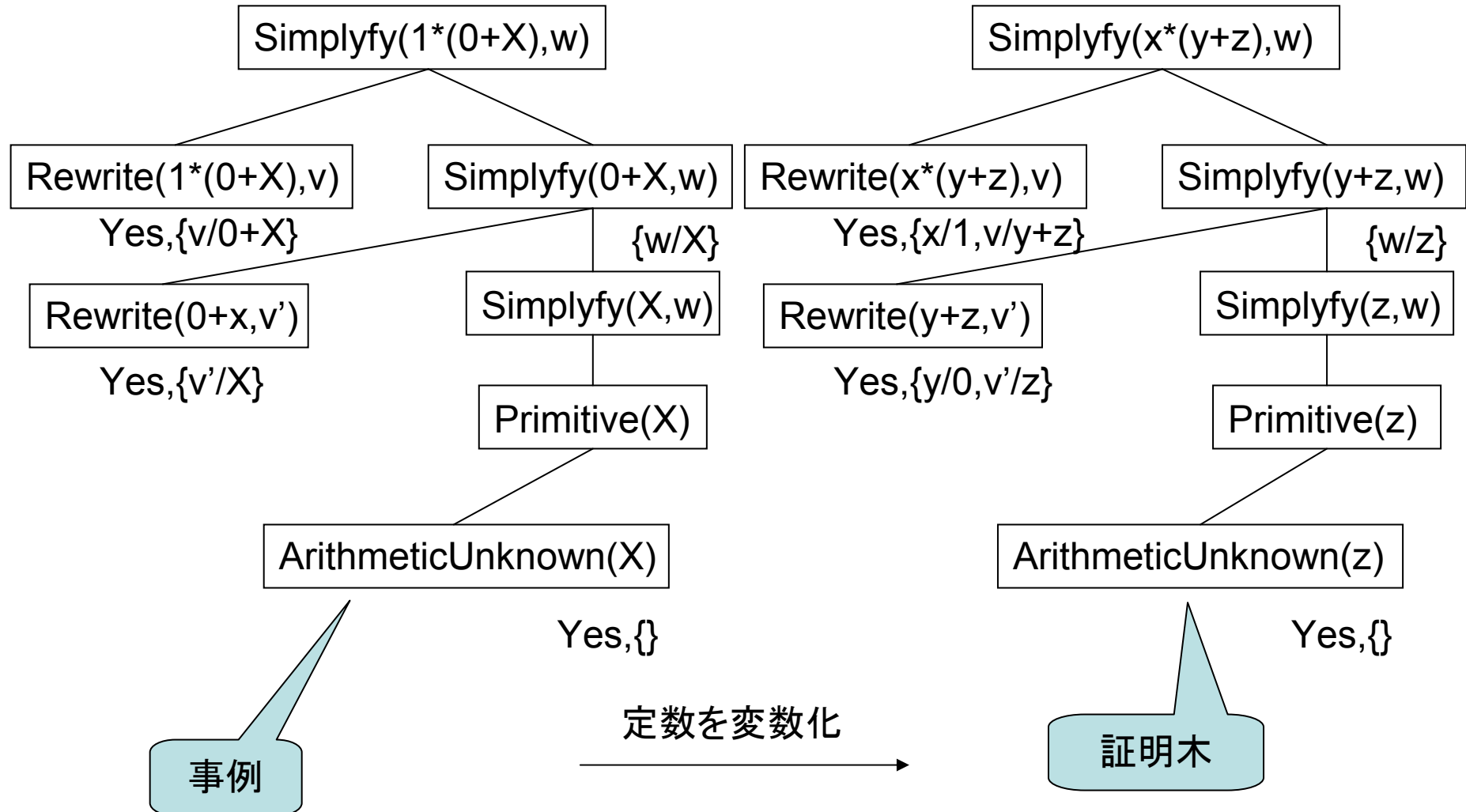
# なぜ説明に基づいた学習か

- あることがなぜ良い考え方なのかを説明することは、考え方を思いつくよりはずっと容易である
- 一度あることが理解されれば、それを一般化できるし、他の環境で使うこともできる
- 一般的な法則を事例から抽出する
- 説明に基づいた学習は最初の本の定数を変数化することでもうひとつの証明木を作り出す
- 例(2ページ後のスライド)

# 説明に基づいた学習

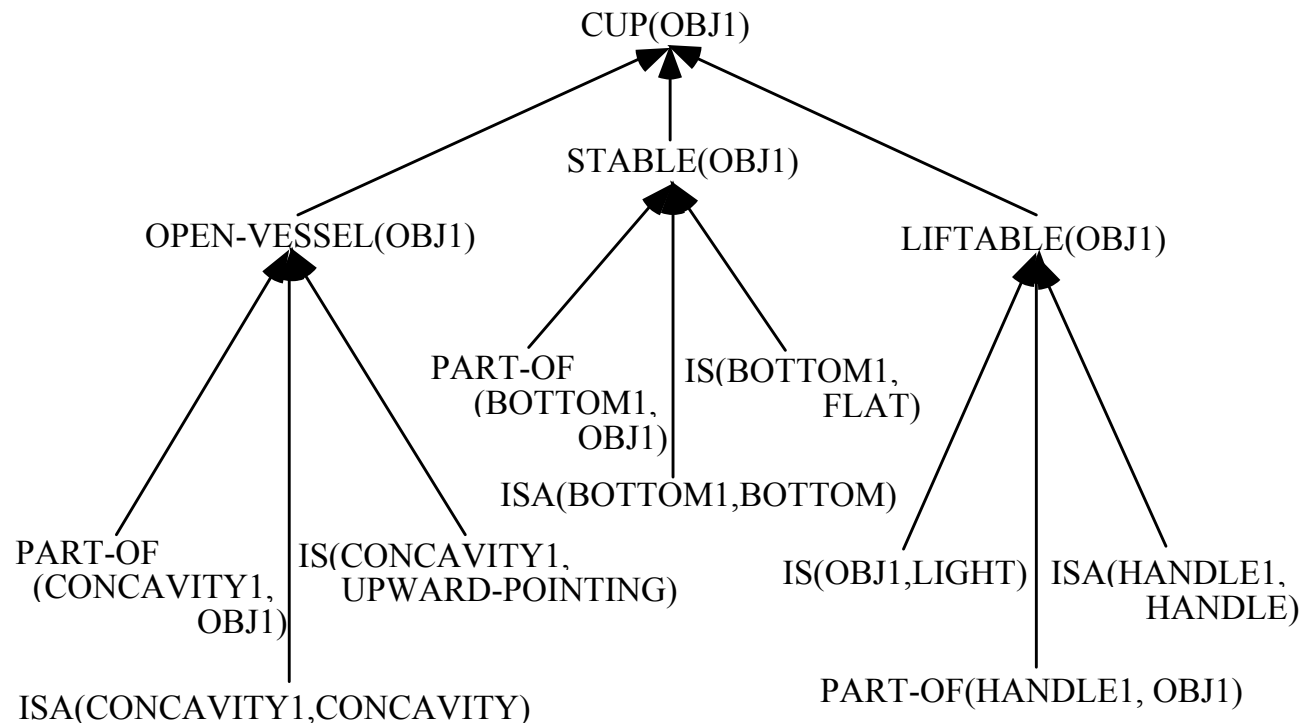
- $1X(0+X)$ が $X$ であることを証明して、一般的な証明木を作成する
  - 知識ベースに次の背景知識が用意されている
    - $\text{Rewrite}(u,v) \wedge \text{Simplify}(v,w) \Rightarrow \text{Simplify}(u,w)$
    - $\text{Primitive}(u) \Rightarrow \text{Simplify}(u,u)$
    - $\text{ArithmeticUnknown}(u) \Rightarrow \text{Primitive}(u)$
    - $\text{Number}(u) \Rightarrow \text{Primitive}(u)$
    - $\text{Rewrite}(1xu,u).$
    - $\text{Rewrite}(0+u,u).$

# 説明に基づいた学習



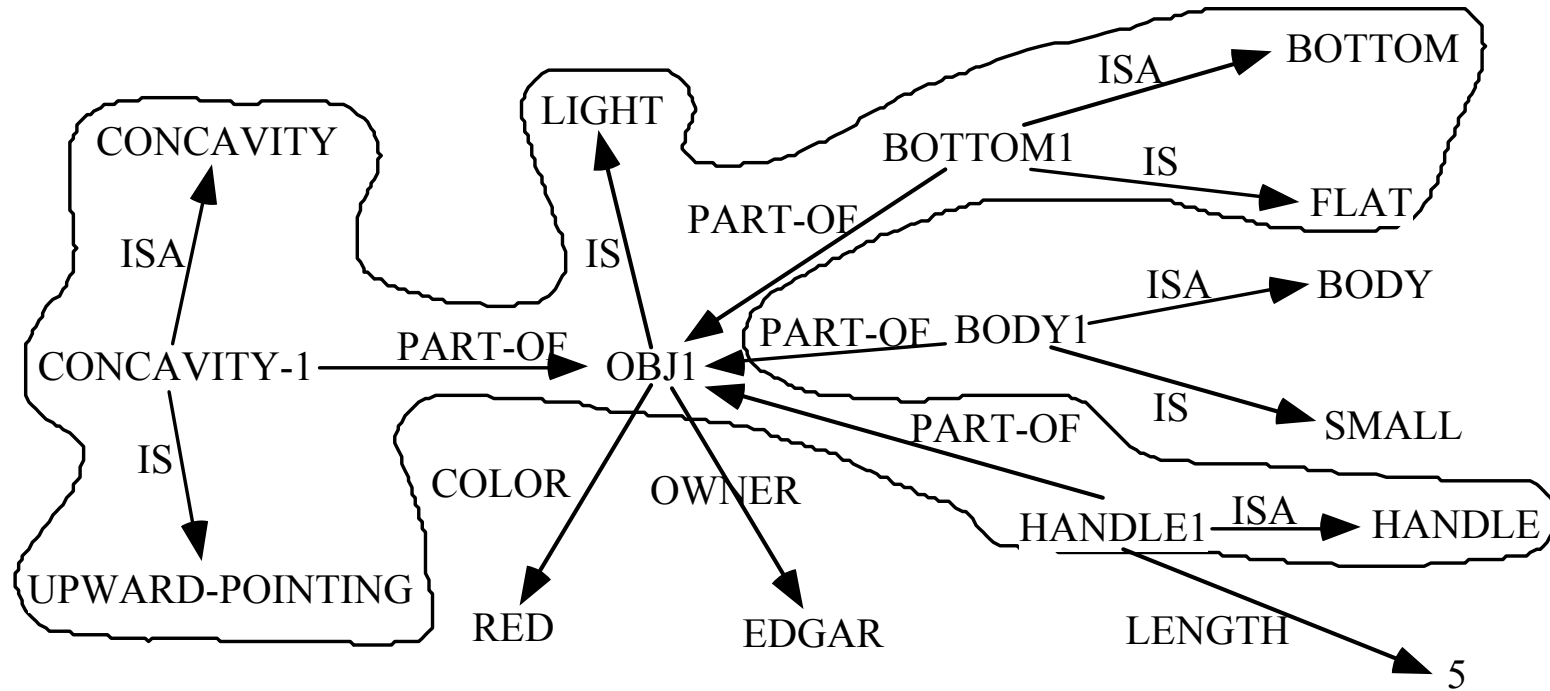
# 説明に基づいた学習

- 事例からカップの証明木を作成する
  - 証明木の葉は、カップと認知するために訓練として与えられた事例の特徴を記述したものである。証明木を組み立てることで、カップに関係する特徴が取り出される



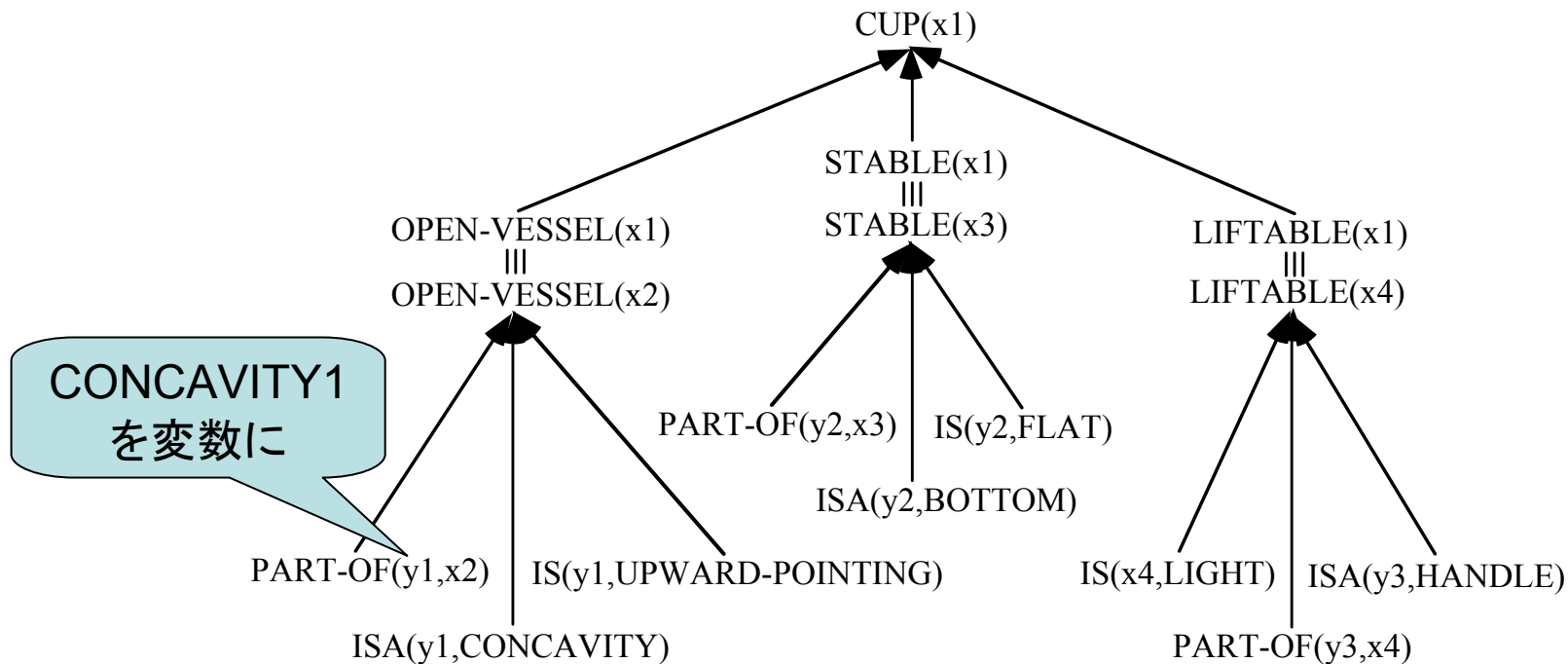
# 説明に基づいた学習

- カップを表わす意味ネットワーク。サークル内はカップに関する特徴



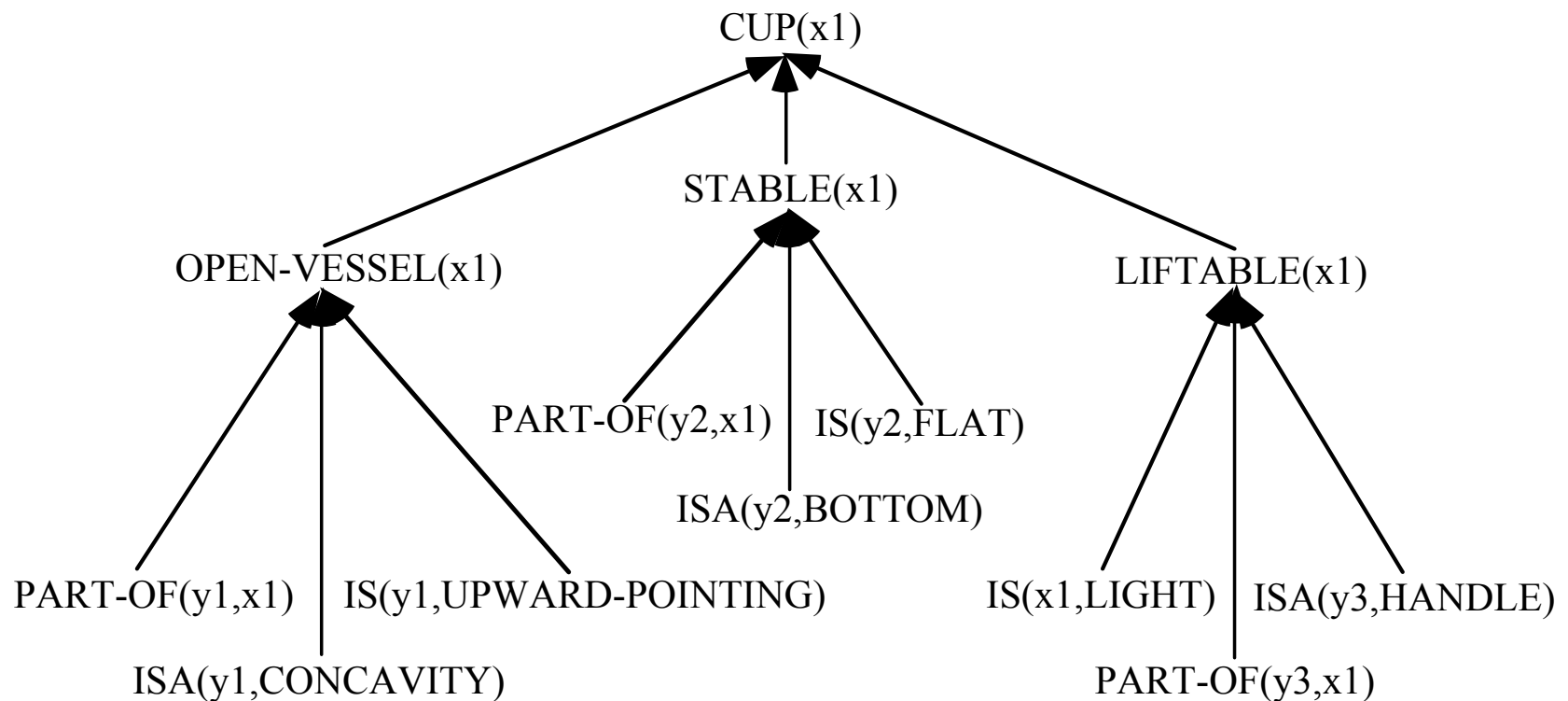
# 説明に基づいた学習

- 真を保ちながら証明木をできるだけ一般化する
  - それぞれの規則のインスタンスを一般的なパターンで置き換える
  - これらのパターンの最も一般的な単一化を考える



# 説明に基づいた学習

- 一般化された証明木の葉は概念としてのカップの操作上の定義を表す  
"x1"y1"y2"y3, [PART-OF(y1, x1) & ISA(y1, CONCAVITY) &  
IS(y1, UPWARD-POINTING) & PART-OF(y2, x1) &  
ISA(y2, BOTTOM) & IS(y2, FLAT) & IS(x1, LIGHT) &  
PART-OF(y3, x1) & ISA(y3, HANDLE) => CUP(x1)]



# 説明に基づいた学習の基本

- 事例が与えられたとき、背景知識を用いて証明木を構成する
- 並行して、ゴールが変数化されたものとなるように、証明木を一般化する
- 新しい法則を構成する (*leaves => the root*)
- ゴールの中の変数によらずに真である条件を捨てる

# 説明に基づいた学習の効率

- 法則の選択(証明木の各所から新たな法則を得ることが出来るが)
  - － 余りにも多すぎる法則は -> 遅い推論
  - － 導出された法則の利得は - 顕著な高速化
  - － できるだけ一般的に
- 操作性 - サブゴールはそれを簡単に解くための操作上の手段
  - － 操作性と一般性とのトレードオフ
- 説明に基づいた学習が効率のよい方法であるかという点が問われている。経験によれば、うまく構成すれば、高速になる。例えば、話し言葉でのスウェーデン語から英語への実時間での変換が可能である

# 関連情報を用いての学習

- 背景知識: 国の中では同じ言語が話される

$$\text{Nat}(x,n) \wedge \text{Nat}(y,n) \wedge \text{Lang}(x,l) \Rightarrow \text{Lang}(y,l)$$

- 観察: 国籍が与えられれば、言語は完全に決められる

– Given Fernando is Brazilian & speaks Portuguese

$$\text{Nat}(\text{Fernando},B) \wedge \text{Lang}(\text{Fernando},P)$$

- 論理的に次のことが導き出せる

$$\text{Nat}(y,B) \Rightarrow \text{Lang}(y,P)$$

# 関数従属

- 関形の形式化: 決定 – 言語(Portuguese)は国 (Brazil)から導かれる  
Nationality(x, Brazil) => Language(x, Portuguese)
- 決定は述語の関係である
- 対応する一般化は決定と記述から論理的に導き出せる

# 関数従属

- Fernando一人から全てのブラジル人を一般化した  
が、しかし、すべての国に対してではない。  
従って、決定は考慮すべき仮説空間を限定している
- 目標となる述語に関連する仮説を構成するための十分な基準語彙を決定は明確にする。
- 仮説空間の大きさを小さくすることは目標となる述語を学びやすくする
  - $n$ 個のブール性質に対して、決定が $p$ 個の性質を含んでいるなら、節約は $O(2^{n-p})$ となる

# 関連情報を使った学習

- 決定  $P \wedge Q$  が成り立つとき、ある事例が  $P$  に一致するならば、それらは  $Q$  にも一致しなければならない
- 観察と一貫する最も簡単な決定を見出す
  - 述語から、決定空間を探索する
  - アルゴリズム - (次のページ)
  - 時間の複雑さは  $n$  から  $p$  を選び出すこと
- 性質の選択は、機械学習、パターン認識、統計での研究課題である

# アルゴリズム

## (最小の一貫性のある決定)

**function** Minimal-Consistent-Det( $E, A$ ) **returns** set of attributes

**inputs:**  $E$ , a set of examples

$A$ , a set of attributes, of size  $n$

**for**  $i \leftarrow 0, \dots, n$  **do**

for each subset  $A_i$  of  $A$  of size  $i$  do

if Consistent-Det?( $A_i, E$ ) **then return**  $A_i$

---

**function** Consistent-Det?( $A, E$ ) **returns** a truth-value

**inputs:**  $A$ , a set of attributes

$E$ , a set of examples

**local variables:**  $H$ , a hash table

**for each** example  $e$  **in**  $E$  **do**

if some example on  $H$  has the same value as  $e$  for the attributes  $A$   
but a different classification **then return** *false*

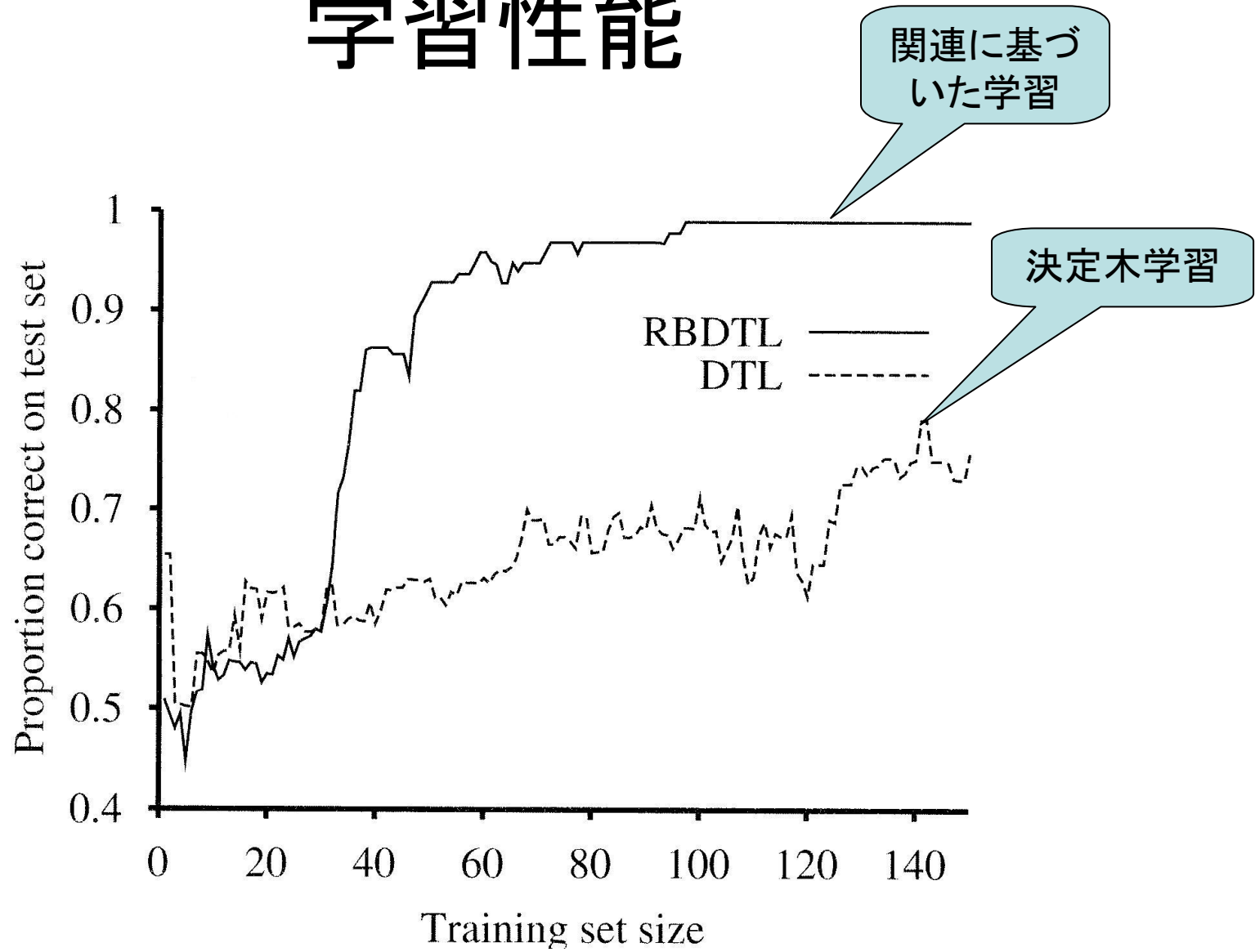
store the class of  $e$  in  $H$ , indexed by the values for attributes  $A$  of the example  $e$

**return** *true*

# 関連情報を使った学習

- 関連に基づいた学習と決定木学習を結びつけたもの-> RBDTL
- 学習性能は向上 (次のスライド).
  - 訓練集合の大きさでの性能の差
  - 利得: 時間節約, 適合しすぎの可能性が少ない
- 関連に基づいた学習での他の問題
  - 雑音処理
  - 他の背景知識の利用
  - 属性に基づいたアルゴリズムから一階述語論理へ

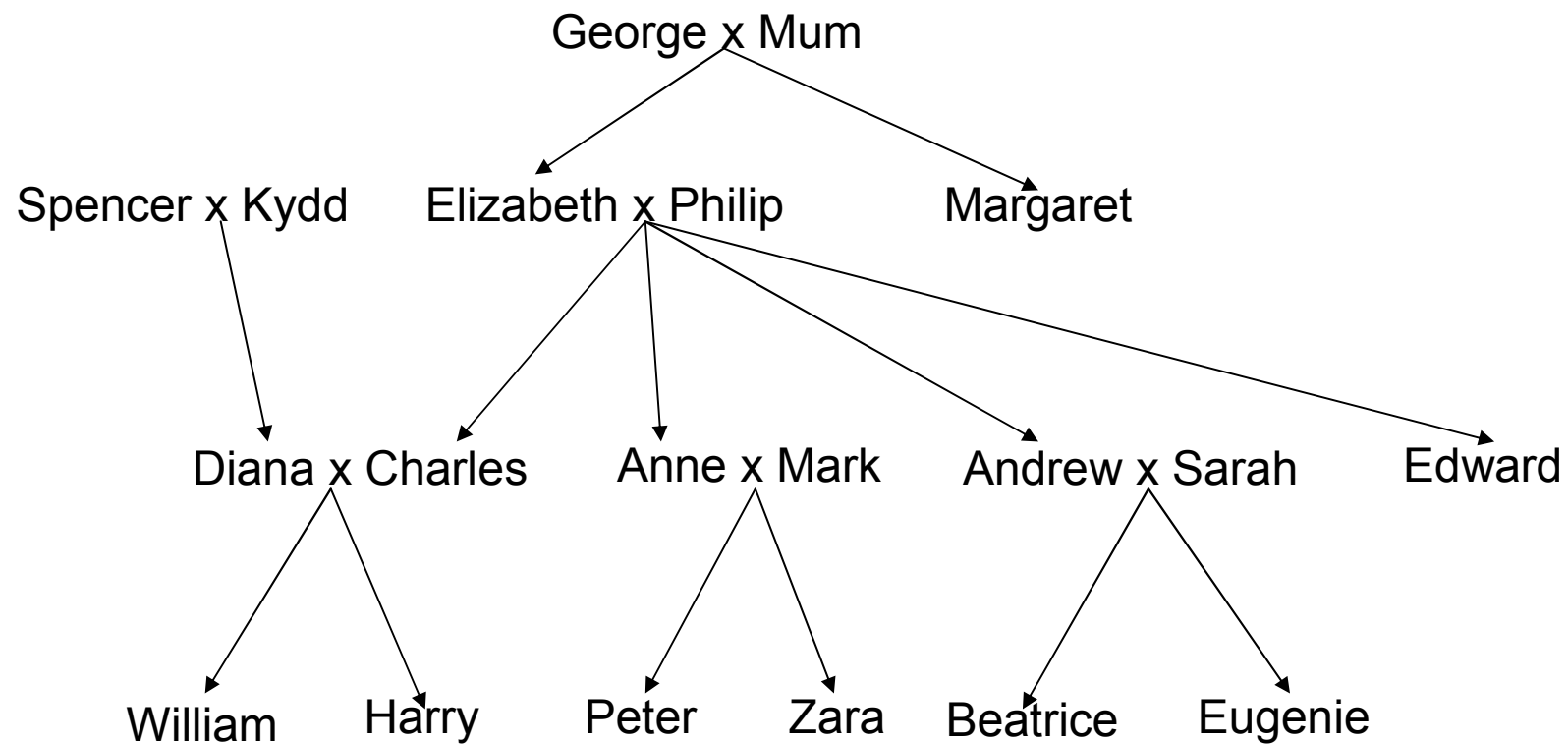
# 学習性能



# 帰納的論理プログラミング

- 帰納的方法と一階述語論理を結合
- 帰納的論理プログラミングは定理を論理プログラムとして表す
- 帰納的論理プログラミングは一般的でありかつ一階述語論理の定理を事例から演繹的に導き出す完全なアルゴリズムである
- 属性に基づいたアルゴリズムで失敗した領域で、帰納的論理プログラミングを使えば、うまく学習する
- 例 – 家系図 (次のスライド)

# 帰納的論理プログラミング



# 帰納的論理プログラミング

- 記述

Father(Philip,Charles)	Father(Philip,Anne)
Mother(Mum,Margaret)	Mother(Mum,Elizabeth)
Married(Diana,Charles)	Married(Elizabeth,Philip)
Male(Philip)	Male(Charles)
Female(Beatrice)	Female(Margaret)

- 分類

GrandParent, BrotherInLaw, Ancestorなどを知りたい

- 事例

Grandparent(Num,Charles) Grandparent(Elizabeth,Peter)  
→ Grandpparent(Mum,Harry) → Grandparent(Spencer, Peter)など

- 背景知識がなければ

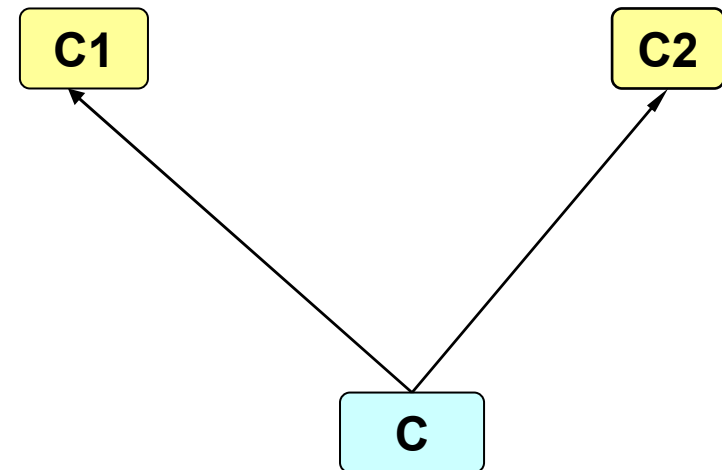
$$\begin{aligned} \text{Grandparent}(x,y) \Leftrightarrow & \{ \exists z \text{ Mother}(x,z) \wedge \text{Mother}(z,y) \} \\ & \vee \{ \exists z \text{ Mother}(x,z) \wedge \text{Father}(z,y) \} \\ & \vee \{ \exists z \text{ Father}(x,z) \wedge \text{Mother}(z,y) \} \\ & \vee \{ \exists z \text{ Father}(x,z) \wedge \text{Father}(z,y) \} \end{aligned}$$

- 背景知識に

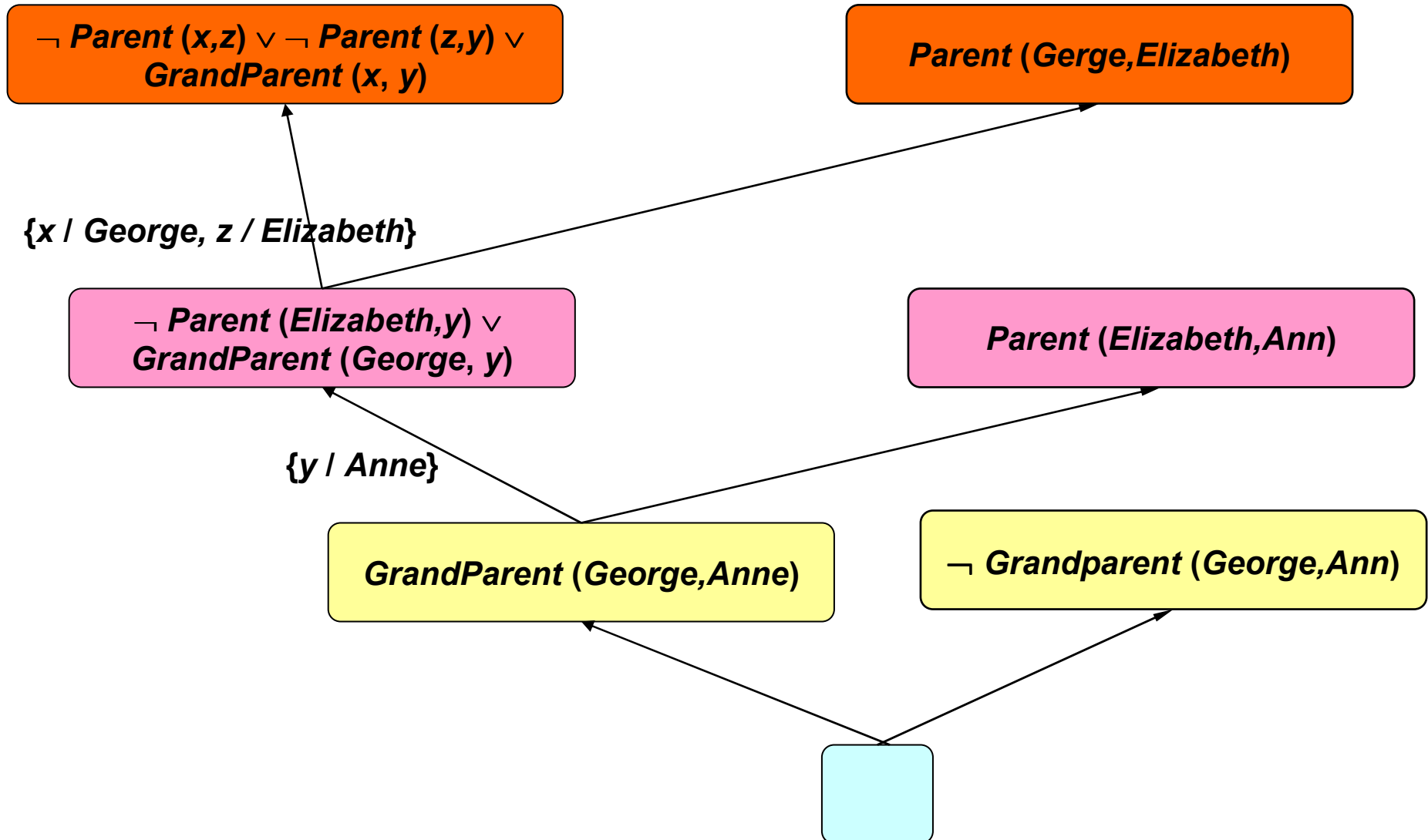
$\text{Parent}(x,y) \Leftrightarrow \{ \exists z \text{ Mother}(x,y) \vee \text{Father}(x,y) \}$ があれば  
 $\text{Grandparent}(x,y) \Leftrightarrow \{ \exists z \text{ Parent}(x,z) \wedge \text{parent}(z,y) \}$ となる

# 逆伴意

- 分類が  $\text{Background} \wedge \text{Hypothesis} \wedge \text{Descriptions}$  から引き出せるなら、反駁を用いてこれを証明することができる
- 証明を後ろ向きに行えるなら、証明が次のように進む仮説空間を見つけることができる
  - $C \rightarrow C1 \text{ and } C2$
  - $C \text{ and } C2 \rightarrow C1$
- 逆証明の生成
  - 家系図 (次のスライド)

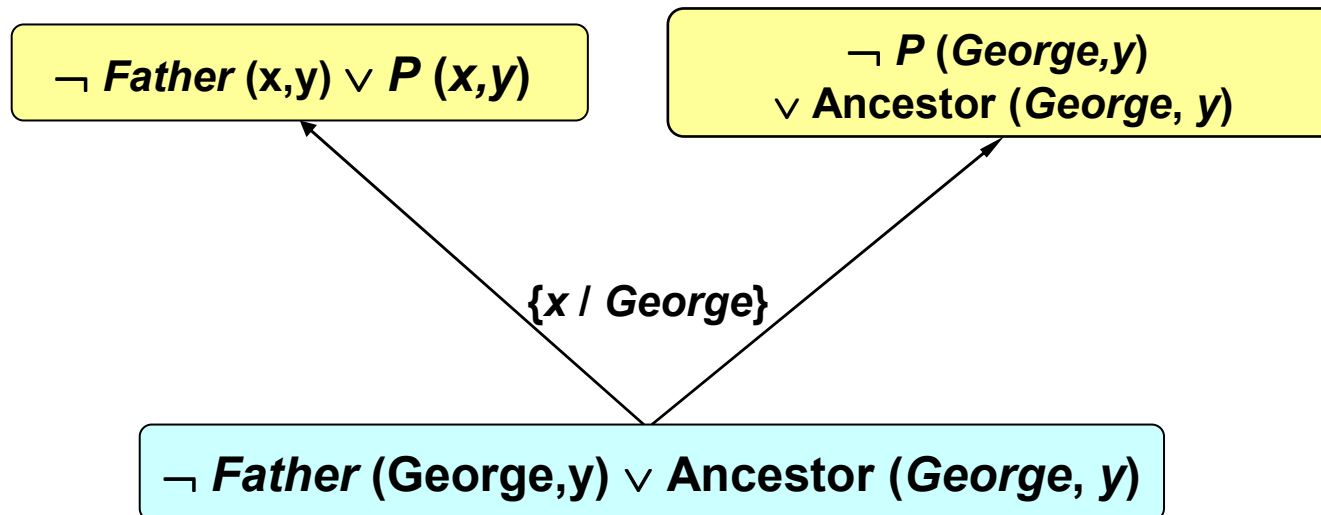


# 逆伴意



# 逆伴意

- 逆伴意は探索を伴う
  - 逆伴意の各ステップは非決定的である
    - あるCとC1に対して、たくさんのC2がある
- 逆伴意で新しい知識を発見する
  - 簡単ではない - a monkey and a typewriter
- 逆伴意で新しい述語を発見する
  - 下の図
- 背景知識の使用が可能な場合には、大きな利点がある



# トップダウン学習

- 決定木での方法を用いての一次ケースでの一般化
  - 一般的ルールで始めて、データに合うように特殊化する
  - 属性ではなく一次リテラルを使う。仮説空間は決定木ではなく節である
- 例:  $\Rightarrow$ grandfather(x,y) (次のスライド)
  - 肯定と否定の事例
  - 左側に一度に一つのリテラルを加える
    - 例: Father (x,y)  $\Rightarrow$  Grandfather(x,y)
    - リテラルをどのように選ぶか (Algorithm on page 702)
  - 法則はいくつかの肯定の事例とは一致し、どの否定の事例とも一致しない。
  - トップダウン学習はカバーされた肯定の事例を消去し、繰り返す

# トップダウン学習

- 肯定の事例  
    <George, Anne>, <Philip, Peter>, ...
- 否定の事例  
    <George, Elizabeth>, <Harry, Zara>, ...
- 最初の節を作る  
    =>grandfather(x,y)
- 左にリテラルを  
    Father(x,y) => grandfather(x,y)  
    Parent(x,z) => grandfather(x,y)  
    Father(x,z) => grandfather(x,y)
- 一番目の式は肯定の事例のどれとも一致しない。2番目と3番目は一致するが、否定の事例で見ると2番目は一致が少ないので、3番目を取り、さらに一つのリテラルを加える  
    Father(x,z)  $\wedge$  Parent(z,y) => grandfather(x,y)

# まとめ

- 累積学習で背景知識を用いる
- 背景知識は仮説空間を小さくする
- 背景知識は伴意の制約のように異なった論理的役割を担う。
- 説明に基づいた学習、関係に基づいた学習、知識に基づいた帰納的学習
- 簡便な定理が表現されるように、帰納的論理プログラミングは、新しい述語を生成する

# 付録

# 帰納的論理プログラミング (ILP)

- FOIL [Quinlan, 90]
  - トップダウンの帰納的論理プログラミングシステム
- GOLEM [Feng & Muggleton, 92]
  - ボトムアップの帰納的論理プログラミングシステム
- PROGOL [Muggleton, 96]
  - 逆伴意に基づいた統一手法 (a well founded theoretical approach)

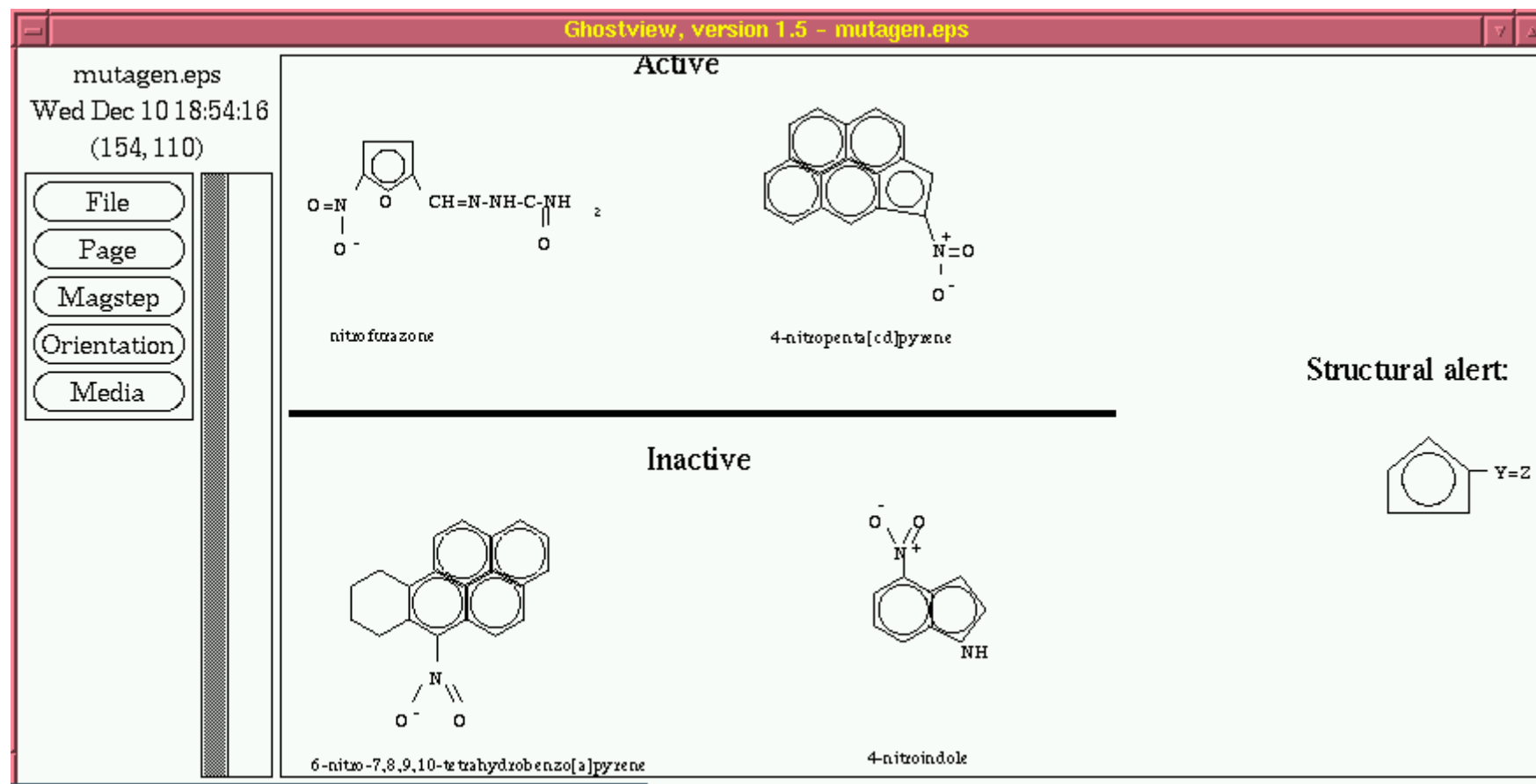
# 実用的な応用

- プログラム合成
  - きわめて困難
  - サブタスク: デバッグ、検証、...
- 機械学習
  - e.g., ゲームを学ぶ
- データ・マイニング
  - 大規模な構造的なデータでのマイニング

# 応用例:変異原性 の推察

- 分子の集合が与えられる
- あるものはDNAで突然変異を起こす (これらは変異原性である)、他はそうはならない
- これらを分子構造を元に区別することを試みる
- Srinivasan et al., 1994: “structural alert”を発見

# 応用例:変異原性の推察



# 応用例:活性基の発見

- Muggleton、他(1996)の発見
- 分子の中から“活性基”を発見
  - ある他の分子のドックとなるサブ構造を特定する
  - プロテイン同士の作用
- 分子を構成する原子のリスト: 要素, 3次元の座標, ...
- 背景知識はユークリッド距離、...

# 分子の例: (Muggleton et al. 1996)

