

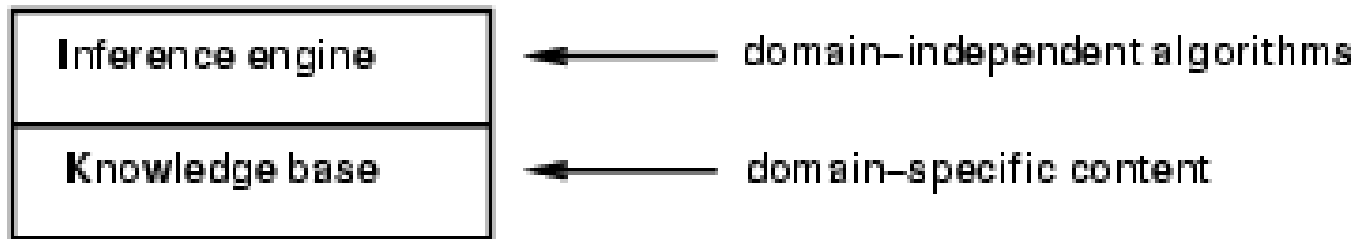
論理エージェント

Chapter 7

概要

- 知識ベースエージェント
- Wumpusの世界
- 論理一般 – モデルと伴意
- 命題(ブーリアン)論理
- 同値, 妥当性, 満足性
- 推論ルールと定理の証明
 - 前向き連鎖
 - 後向き連鎖
 - 融合

知識ベース



- 知識ベース= 形式言語での文の集合
- エージェント(あるいは他のシステム)構築への宣言的手法:
 - エージェントが知る必要があることをエージェントに告げる
- そして、エージェントは自身に何をすべきかを質問する – 答えは知識ベースから返ってくる
- エージェントは知識レベルで見られる
 - 即ち、どのように実現されているかではなく、エージェントが知っていることで
- あるいは実現レベルで見られる
 - 即ち、知識ベースのデータ構造とそれを操作しているアルゴリズム

簡単な知識ベースエージェント

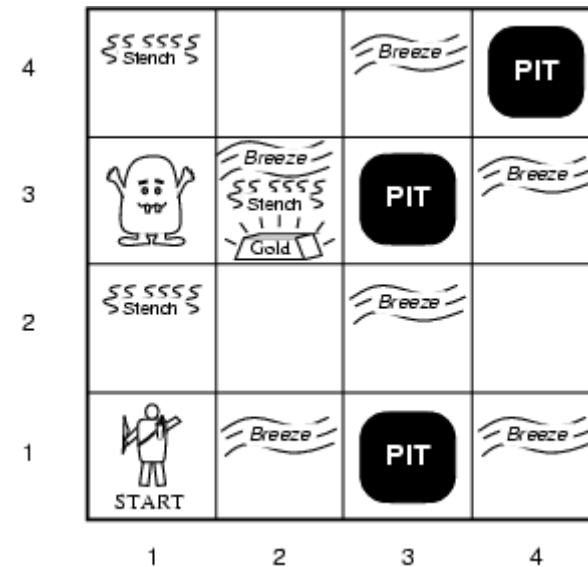
```
function KB-AGENT(percept) returns an action
  static: KB, a knowledge base
          t, a counter, initially 0, indicating time

  TELL(KB, MAKE-PERCEPT-SENTENCE(percept, t))
  action ← ASK(KB, MAKE-ACTION-QUERY(t))
  TELL(KB, MAKE-ACTION-SENTENCE(action, t))
  t ← t + 1
  return action
```

- エージェントは次のことができないといけない:
 - 状態、動作などを表すこと
 - 新しく知覚したものを受け入れること
 - 世界の内部表現を更新すること
 - 世界の隠れた性質を推論すること
 - 適切な動作を推論すること

Wumpusの世界のPEAS 記述

- Performance measure
 - 金は +1000, 死は -1000
 - ステップあたり -1, 矢を使用 -10
- Environment
 - wumpusの隣の四角はくさい
 - 窪みに近い四角はそよ風が吹いている
 - 四角に金があればそこは輝いている。逆も真
 - wumpusに向きあっているとき射撃で殺せる
 - 射撃には矢のみを使える
 - 掴むことで同じ四角の中の金を取る
 - 放すことで同じ四角の中の金を落とす
- Sensors: 悪臭、そよ風、輝き、段差、叫び声
- Actuators: 左折する、右折する、前進する、掴む、放す、射撃する



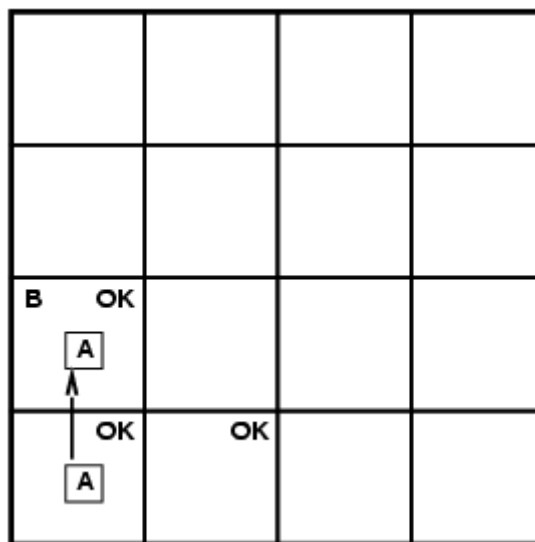
Wumpus の世界の性格

- 完全に観察可能 No – 局所的な知覚のみ
- 決定的 Yes – 結果は正確に示される
- エピソード的 No – 行動のレベルで順序的
- 静的 Yes – Wumpusと窪みは動かない
- 離散的 Yes
- 単一エージェント Yes – Wumpusは本質的に単一エージェントの自然な特徴である

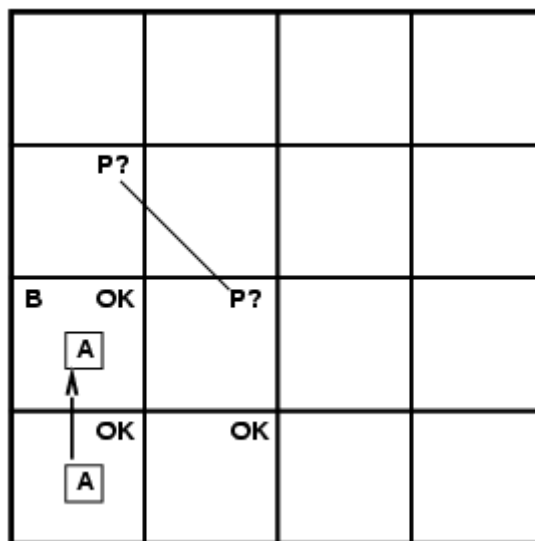
wumpusの世界の探索

OK			
OK A	OK		

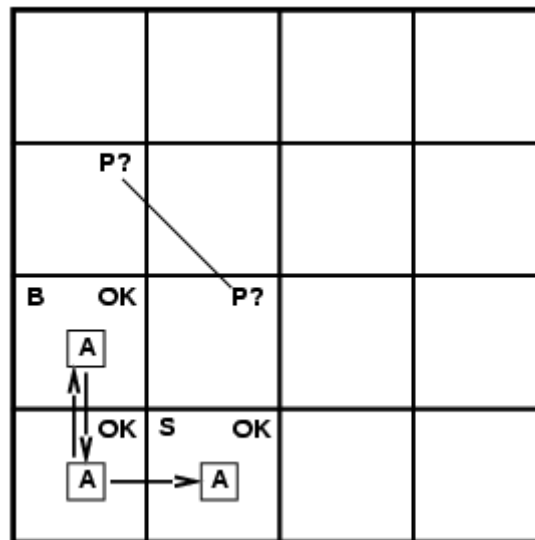
wumpusの世界の探索



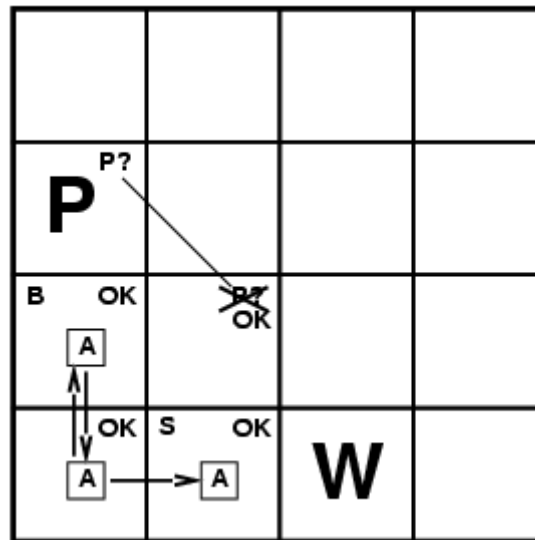
wumpusの世界の探索



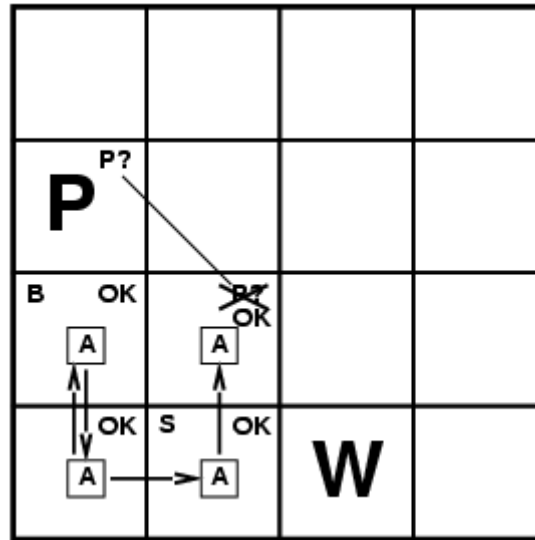
wumpusの世界の探索



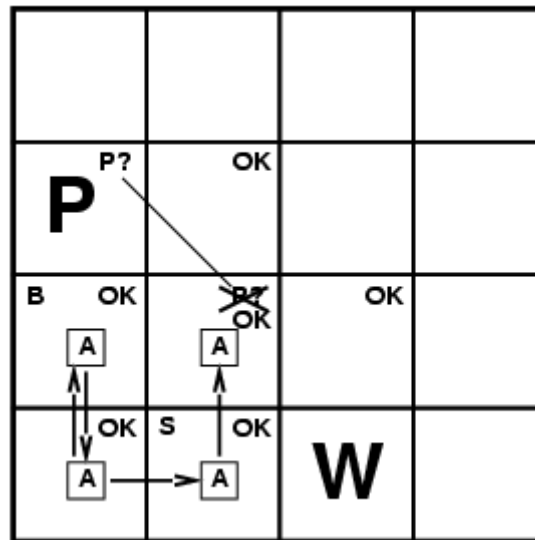
wumpusの世界の探索



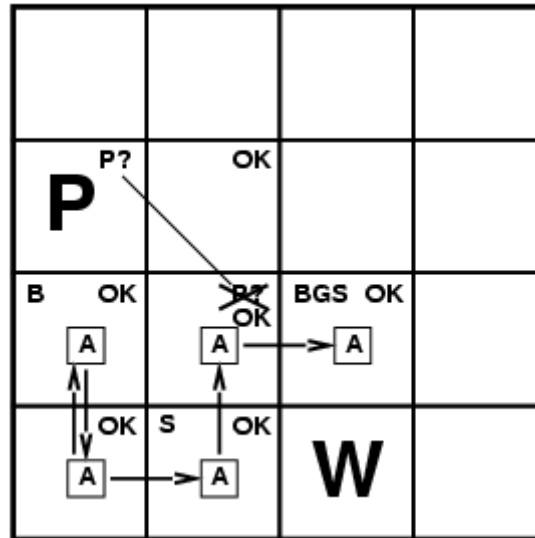
wumpusの世界の探索



wumpusの世界の探索



wumpusの世界の探索



論理一般

- 情報から結論が引き出せるとき、その情報を形式的に表現する言語を論理という
- 構文 は言語の文章を規定する
- 意味論は文の意味を規定する;
 - i.e., 世界の中で真を定義する
- E.g., 算術言語
 - $x+2 \geq y$ は文; $x^2+y > \{$ は文でない
 - $x+2 \geq y$ は真のときかつそのときに限り $x+2$ の数が y の数よりは小さくない
 - $x+2 \geq y$ は $x = 7, y = 1$ の世界で真である
 - $x+2 \geq y$ は $x = 0, y = 6$ の世界で偽である

伴意

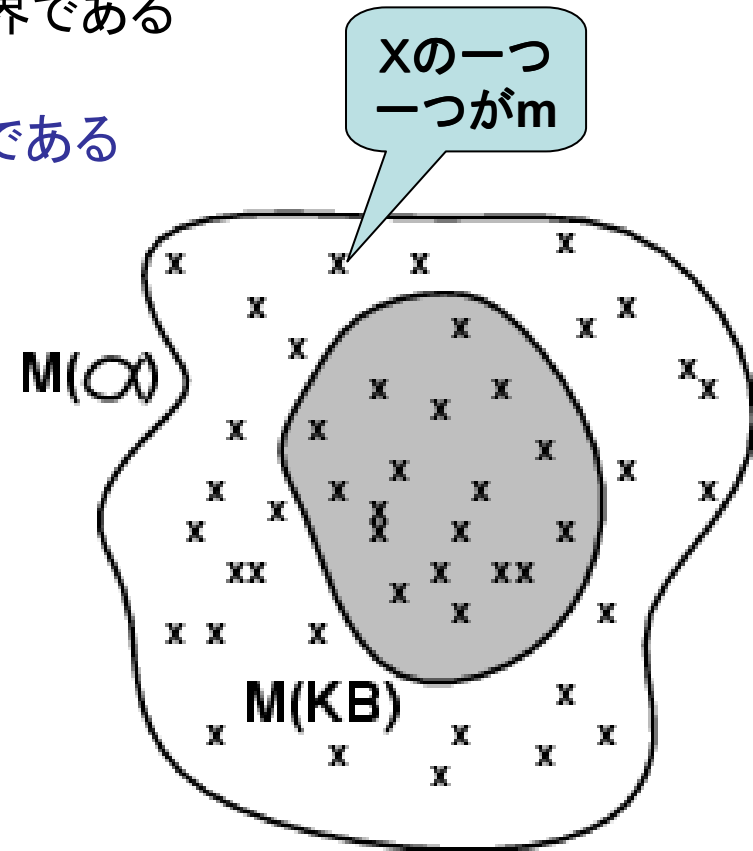
- 伴意 (Entailment) はあることが他のことに従うことを言う:

$$KB \vdash \alpha$$

- 知識ベース KB は文 α を伴意するときかつそのときに限り KB が真である全ての世界で α は真である
 - 例: “the Giants won” かつ “the Reds won” を含んでいる KB は “Either the Giants won or the Reds won” を伴意している
 - 例: $x+y = 4$ は $4 = x+y$ を伴意している
 - 伴意は意味に基づいた文 (i.e., 構文) の関係である

モデル

- 論理学者は典型的にモデルの観点から考えるが、そのモデルは真であると評価できる形式的に構造化された世界である
- α が m で真であるならば m は α のモデルである
- $M(\alpha)$ は α の全てのモデルの集合である
- $KB \vdash \alpha$ iff $M(KB) \subseteq M(\alpha)$
 - 例: $KB = \text{Giants won and Reds won}$
 $\alpha = \text{Giants won}$

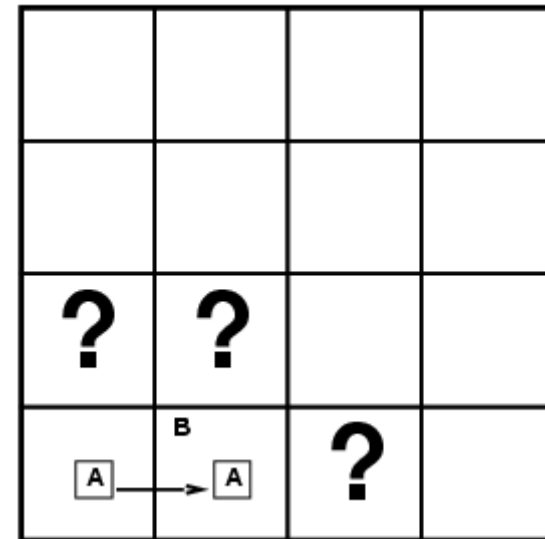


wumpusの世界の伴意

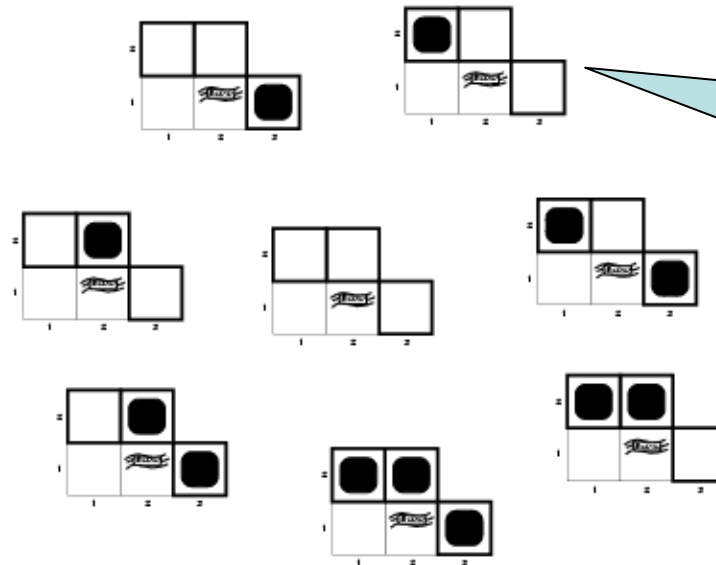
[1,1]で何も見つけない後で右に動いて [2,1]でそよ風という状態

窪みだけを仮定してKBに対する可能なモデルを考える

3つの場所に対するブール値の選択 ⇒ 8つの可能なモデル

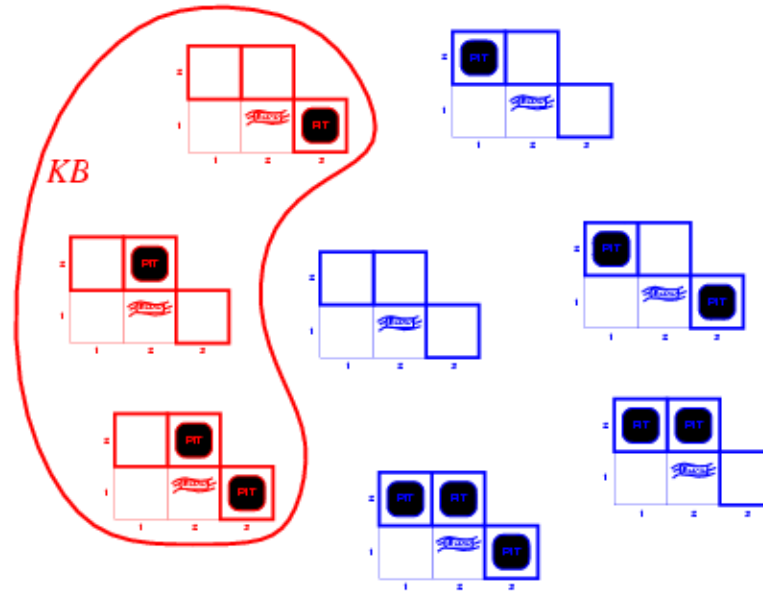


Wumpusのモデル



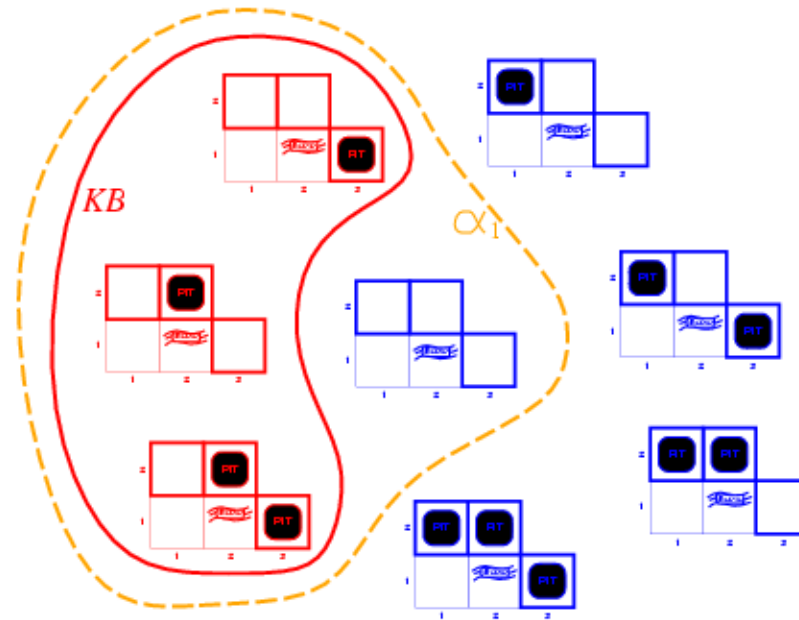
一つ一つが
モデル
全部で8つ
のモデル

Wumpusのモデル



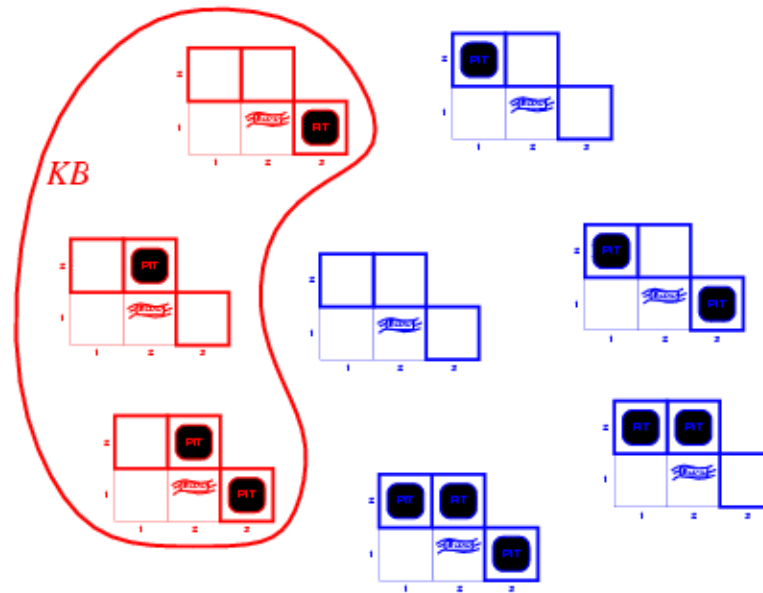
- $KB = \text{wumpusの世界のルール} + \text{観察}$

Wumpusのモデル



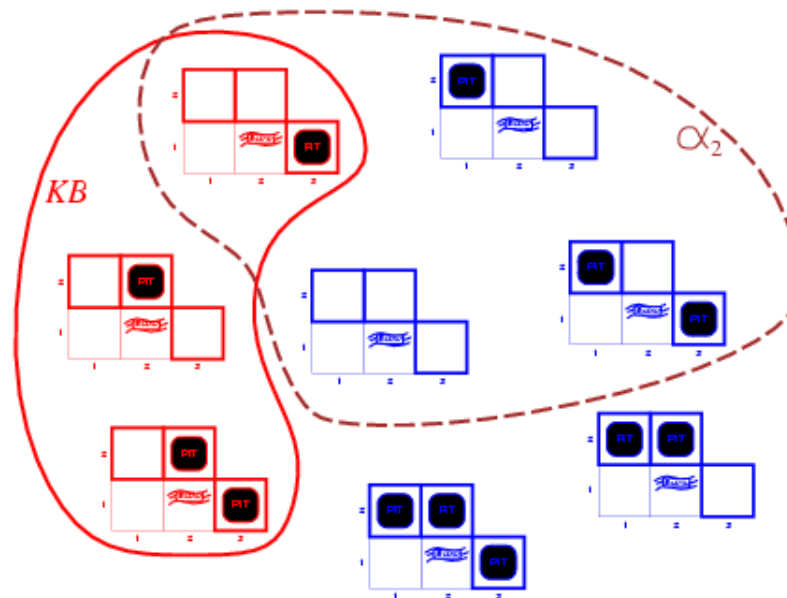
- KB = wumpusの世界のルール + 観察
- α_1 = “[1,2] is safe”, $KB \vdash \alpha_1$, モデルの検査によって証明される

Wumpusのモデル



- $KB = \text{wumpusの世界のルール} + \text{観察}$

Wumpusのモデル



- KB = wumpusの世界のルール + 観察
- α_2 = "[2,2] is safe", $KB \not\vdash \alpha_2$

推論

- $KB \vdash_i \alpha$ = 文 α は KB から手続き i で導出できる
- **健全性**: $KB \vdash_i \alpha$ であるときはいつでも $KB \models \alpha$ が真であるならば i は健全である
- **完全性**: $KB \models \alpha$ であるときはいつでも $KB \vdash_i \alpha$ が真であるならば i は完全である
- これからの話: 関心のあることのほとんどを表現でき、健全性と完全性を満たしている推論手続きを有している論理(一階述語論理)を定義する
- 即ち、 KB で分かっていることから答えが得られるような質問に、手続きは答える

命題論理: 構文

- 命題論理は単純な論理である— 基本的な考え方を説明する
- 命題論理のシンボル P_1, P_2 は文である
 - S が文なら, $\neg S$ は文である (negation)
 - S_1 と S_2 が文なら, $S_1 \wedge S_2$ は文である (conjunction)
 - S_1 と S_2 が文なら, $S_1 \vee S_2$ は文である (disjunction)
 - S_1 と S_2 が文なら, $S_1 \Rightarrow S_2$ は文である (implication)
 - S_1 と S_2 が文なら, $S_1 \Leftrightarrow S_2$ は文である (biconditional)

命題論理: 意味論

それぞれのモデルは各命題シンボルに対して真・偽を明確にする

E.g. $P_{1,2}$ $P_{2,2}$ $P_{3,1}$
false true false

これらの値で8つの可能なモデルを数え上げることができる

モデル m に関連して真を評価するための法則:

$\neg S$ は真であるときかつそのときに限り S は偽

$S_1 \wedge S_2$ は真であるときかつそのときに限り S_1 は真かつ S_2 は真

$S_1 \vee S_2$ は真であるときかつそのときに限り S_1 は真あるいは S_2 は真

$S_1 \Rightarrow S_2$ は真であるときかつそのときに限り S_1 は偽あるいは S_2 は真

即ち $S_1 \Rightarrow S_2$ は偽であるときかつそのときに限り S_1 は真かつ S_2 は偽

$S_1 \Leftrightarrow S_2$ は真であるときかつそのときに限り $S_1 \Rightarrow S_2$ は真かつ $S_2 \Rightarrow S_1$ は真

単純な再帰的プロセスは任意の文を評価する, 例:

$$\neg P_{1,2} \wedge (P_{2,2} \vee P_{3,1}) = \text{true} \wedge (\text{true} \vee \text{false}) = \text{true} \wedge \text{true} = \text{true}$$

結合子に対する真理表

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
<i>false</i>	<i>false</i>	<i>true</i>	<i>false</i>	<i>false</i>	<i>true</i>	<i>true</i>
<i>false</i>	<i>true</i>	<i>true</i>	<i>false</i>	<i>true</i>	<i>true</i>	<i>false</i>
<i>true</i>	<i>false</i>	<i>false</i>	<i>false</i>	<i>true</i>	<i>false</i>	<i>false</i>
<i>true</i>	<i>true</i>	<i>false</i>	<i>true</i>	<i>true</i>	<i>true</i>	<i>true</i>

Wumpus の世界の文

$[i, j]$ に窪みがあるなら $P_{i,j}$ は真である

$[i, j]$ でそよ風があるなら $B_{i,j}$ は真である

$\neg P_{1,1}$

$\neg B_{1,1}$

$B_{2,1}$

- “窪みは隣の四角にそよ風を起こす”

$$B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$$

$$B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$$

推論に対する真理表

$B_{1,1}$	$B_{2,1}$	$P_{1,1}$	$P_{1,2}$	$P_{2,1}$	$P_{2,2}$	$P_{3,1}$	KB	α_1
<i>false</i>	<i>false</i>	<i>false</i>	<i>false</i>	<i>false</i>	<i>false</i>	<i>false</i>	<i>false</i>	<i>true</i>
<i>false</i>	<i>false</i>	<i>false</i>	<i>false</i>	<i>false</i>	<i>false</i>	<i>true</i>	<i>false</i>	<i>true</i>
\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots
<i>false</i>	<i>true</i>	<i>false</i>	<i>false</i>	<i>false</i>	<i>false</i>	<i>false</i>	<i>false</i>	<i>true</i>
<i>false</i>	<i>true</i>	<i>false</i>	<i>false</i>	<i>false</i>	<i>false</i>	<i>true</i>	<u><i>true</i></u>	<u><i>true</i></u>
<i>false</i>	<i>true</i>	<i>false</i>	<i>false</i>	<i>false</i>	<i>true</i>	<i>false</i>	<u><i>true</i></u>	<u><i>true</i></u>
<i>false</i>	<i>true</i>	<i>false</i>	<i>false</i>	<i>false</i>	<i>true</i>	<i>true</i>	<u><i>true</i></u>	<u><i>true</i></u>
<i>false</i>	<i>true</i>	<i>false</i>	<i>false</i>	<i>true</i>	<i>false</i>	<i>false</i>	<i>false</i>	<i>true</i>
\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots
<i>true</i>	<i>true</i>	<i>true</i>	<i>true</i>	<i>true</i>	<i>true</i>	<i>true</i>	<i>false</i>	<i>false</i>

数え上げによる推論

- 全てのモデルの深さ優先数え上げは健全で完全である

```
function TT-ENTAILS?(KB,  $\alpha$ ) returns true or false
  symbols  $\leftarrow$  a list of the proposition symbols in KB and  $\alpha$ 
  return TT-CHECK-ALL(KB,  $\alpha$ , symbols, [])

function TT-CHECK-ALL(KB,  $\alpha$ , symbols, model) returns true or false
  if EMPTY?(symbols) then
    if PL-TRUE?(KB, model) then return PL-TRUE?( $\alpha$ , model)
    else return true
  else do
    P  $\leftarrow$  FIRST(symbols); rest  $\leftarrow$  REST(symbols)
    return TT-CHECK-ALL(KB,  $\alpha$ , rest, EXTEND(P, true, model)) and
      TT-CHECK-ALL(KB,  $\alpha$ , rest, EXTEND(P, false, model))
```

- n 個のシンボルに対して時間の複雑性は $O(2^n)$ で、空間の複雑性は $O(n)$ である

論理的同値

- 二つの文は論理的に同値であるときかつそのときに限り同じモデルで真である: $\alpha \equiv \beta$ iff $\alpha \vdash \beta$ and $\beta \vdash \alpha$

$$(\alpha \wedge \beta) \equiv (\beta \wedge \alpha) \quad \text{commutativity of } \wedge$$

$$(\alpha \vee \beta) \equiv (\beta \vee \alpha) \quad \text{commutativity of } \vee$$

$$((\alpha \wedge \beta) \wedge \gamma) \equiv (\alpha \wedge (\beta \wedge \gamma)) \quad \text{associativity of } \wedge$$

$$((\alpha \vee \beta) \vee \gamma) \equiv (\alpha \vee (\beta \vee \gamma)) \quad \text{associativity of } \vee$$

$$\neg(\neg\alpha) \equiv \alpha \quad \text{double-negation elimination}$$

$$(\alpha \Rightarrow \beta) \equiv (\neg\beta \Rightarrow \neg\alpha) \quad \text{contraposition}$$

$$(\alpha \Rightarrow \beta) \equiv (\neg\alpha \vee \beta) \quad \text{implication elimination}$$

$$(\alpha \Leftrightarrow \beta) \equiv ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)) \quad \text{biconditional elimination}$$

$$\neg(\alpha \wedge \beta) \equiv (\neg\alpha \vee \neg\beta) \quad \text{de Morgan}$$

$$\neg(\alpha \vee \beta) \equiv (\neg\alpha \wedge \neg\beta) \quad \text{de Morgan}$$

$$(\alpha \wedge (\beta \vee \gamma)) \equiv ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma)) \quad \text{distributivity of } \wedge \text{ over } \vee$$

$$(\alpha \vee (\beta \wedge \gamma)) \equiv ((\alpha \vee \beta) \wedge (\alpha \vee \gamma)) \quad \text{distributivity of } \vee \text{ over } \wedge$$

妥当性と満足性

文が**全ての**モデルで真であるなら、その文は**妥当**である

例: $True, A \vee \neg A, A \Rightarrow A, (A \wedge (A \Rightarrow B)) \Rightarrow B$

妥当性は**演繹法**を介して推論につなげられる:

$KB \vdash \alpha$ のときかつそのときに限り $(KB \Rightarrow \alpha)$ は妥当である

あるモデルで文が真であるなら、その文は**満足**である

例: $A \vee B, C$

いかなるモデルでも文が真でないなら、その文は**不満足**である

例: $A \wedge \neg A$

満足性は次を介して推論につなげられる:

$KB \vdash \alpha$ のときかつそのときに限り $(KB \wedge \neg \alpha)$ は不満足である

$KB \vdash \alpha$ (伴意: KB で真である世界で α は真)

iff $KB \Rightarrow \alpha$ は妥当 (伴意が成り立っている場合はいかなる場合でも真)

iff $(KB \wedge \neg \alpha)$ は不満足 (伴意が成り立っている場合はいかなる場合でも偽)

証明法

- 証明法は次の二つの種類に分けられる:
 - 推論法則を用いる
 - 古い文から新しい文の合法的(健全な)生成
 - 証明 = 推論法則を用いる順序は標準的な探索での操作に推論法則を用いることで実現
 - 典型的には、標準形への文の変形を必要とする
 - モデルの検査
 - 真理表の数え上げ (常に n の指数)
 - 改良された後戻り探索アルゴリズム
 - 例: Davis--Putnam-Logemann-Loveland (DPLL)
 - モデル空間での発見的方法 (健全だが完全ではない)
 - 例: min-conflicts-like hill-climbing algorithms

融合

連言標準形(CNF)

リテラル節の選言の連言

例: $(A \vee \neg B) \wedge (B \vee \neg C \vee \neg D)$

- 融合推論法則:

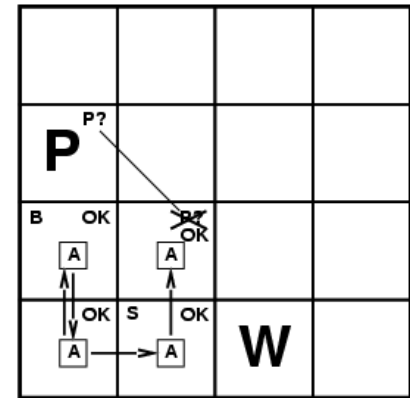
$$\frac{l_i \vee \dots \vee l_k, \quad m_1 \vee \dots \vee m_n}{l_i \vee \dots \vee l_{i-1} \vee l_{i+1} \vee \dots \vee l_k \vee m_1 \vee \dots \vee m_{j-1} \vee m_{j+1} \vee \dots \vee m_n}$$

ここで l_i と m_j は補のリテラル

例:
$$\frac{P_{1,3} \vee P_{2,2}, \quad \neg P_{2,2}}{P_{1,3}}$$

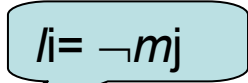
$l_i = \neg m_j$

- 融合は命題論理においては健全で完全である



融合

融合推論法則の完全性:

$$\frac{\begin{array}{l} \neg(l_i \vee \dots \vee l_{i-1} \vee l_{i+1} \vee \dots \vee l_k) \Rightarrow l_i \\ \neg m_j \Rightarrow (m_1 \vee \dots \vee m_{j-1} \vee m_{j+1} \vee \dots \vee m_n) \end{array}}{\neg(l_i \vee \dots \vee l_{i-1} \vee l_{i+1} \vee \dots \vee l_k) \Rightarrow (m_1 \vee \dots \vee m_{j-1} \vee m_{j+1} \vee \dots \vee m_n)}$$


連言標準形への変換

$$B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})\beta$$

1. \Leftrightarrow を消去, $(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$ で $\alpha \Leftrightarrow \beta$ を置換
 $(B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$
2. \Rightarrow を消去, $\neg \alpha \vee \beta$ で $\alpha \Rightarrow \beta$ を置換
 $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg(P_{1,2} \vee P_{2,1}) \vee B_{1,1})$
3. ドモルガンの法則と二重否定で \neg を中側に:
 $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge ((\neg P_{1,2} \wedge \neg P_{2,1}) \vee B_{1,1})$
4. $(\wedge \text{ over } \vee)$ に分配則を応用し、平坦化:
 $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg P_{1,2} \vee B_{1,1}) \wedge (\neg P_{2,1} \vee B_{1,1})$

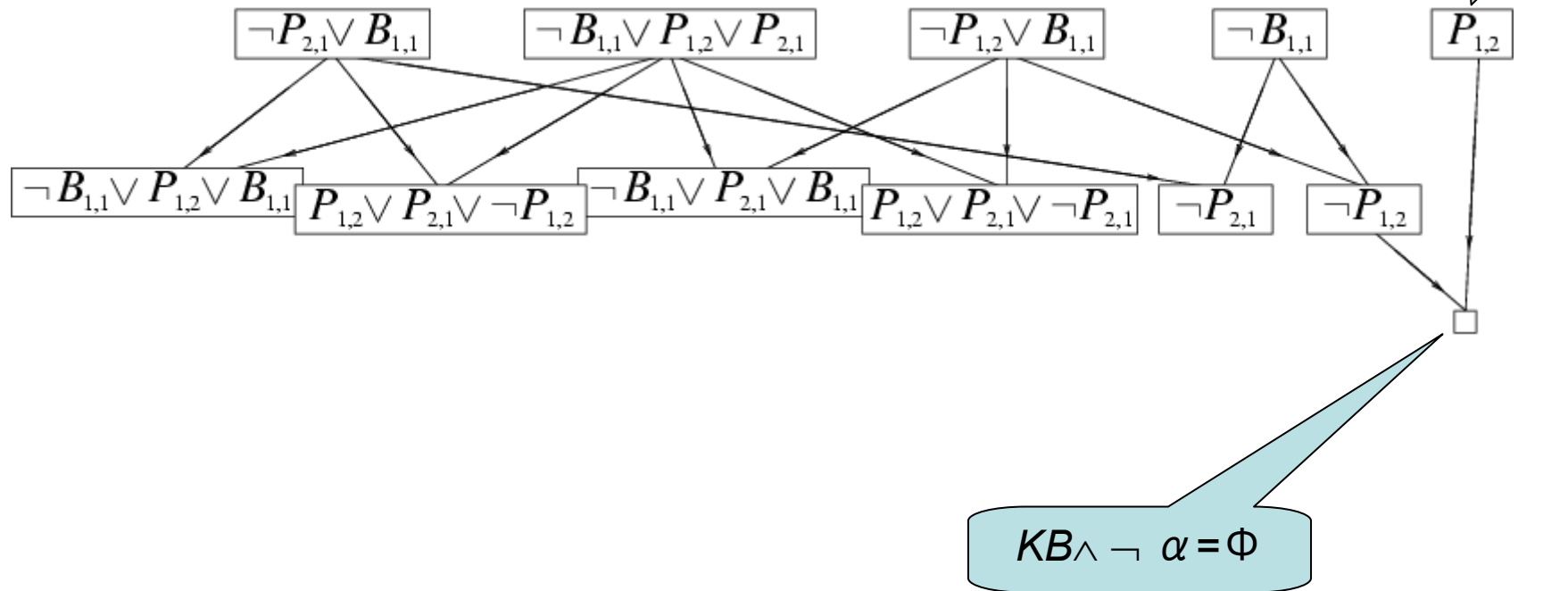
融合アルゴリズム

- 矛盾による証明。即ち、 $KB \wedge \neg \alpha$ が不満足であることを示す

```
function PL-RESOLUTION( $KB, \alpha$ ) returns true or false
   $clauses \leftarrow$  the set of clauses in the CNF representation of  $KB \wedge \neg \alpha$ 
   $new \leftarrow \{ \}$ 
  loop do
    for each  $C_i, C_j$  in  $clauses$  do
       $resolvents \leftarrow$  PL-RESOLVE( $C_i, C_j$ )
      if  $resolvents$  contains the empty clause then return true
       $new \leftarrow new \cup resolvents$ 
    if  $new \subseteq clauses$  then return false
   $clauses \leftarrow clauses \cup new$ 
```

融合の例

- $KB = (B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})) \wedge \neg B_{1,1}$
 $\alpha = \neg P_{1,2}$



前向き・後向き連鎖

- ホーン形式 (restricted)
KB = ホーン節の連言
 - ホーン節 =
 - 命題のシンボル; あるいは
 - (シンボルの連言) \Rightarrow symbol
 - E.g., $C \wedge (B \Rightarrow A) \wedge (C \wedge D \Rightarrow B)$
- 肯定式 (for Horn Form): ホーンの知識ベースに対して完全

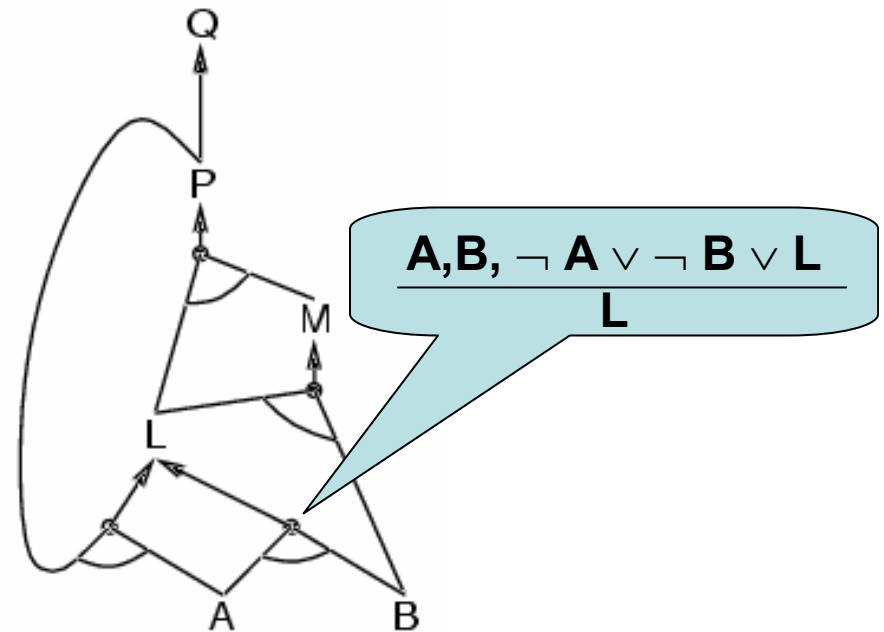
$$\frac{\alpha_1, \dots, \alpha_n, \quad \alpha_1 \wedge \dots \wedge \alpha_n \Rightarrow \beta}{\beta}$$

- 前向き連鎖や後向き連鎖で使われる
- これらのアルゴリズムは自然で、線形の時間で実行される

前向き連鎖

- 考え方: KB で前提が満たされた任意のルールを発火する
 - 質問が見出されるまで KB にその結果を加える

$P \Rightarrow Q$
 $L \wedge M \Rightarrow P$
 $B \wedge L \Rightarrow M$
 $A \wedge P \Rightarrow L$
 $A \wedge B \Rightarrow L$
 A
 B



前向き連鎖のアルゴリズム

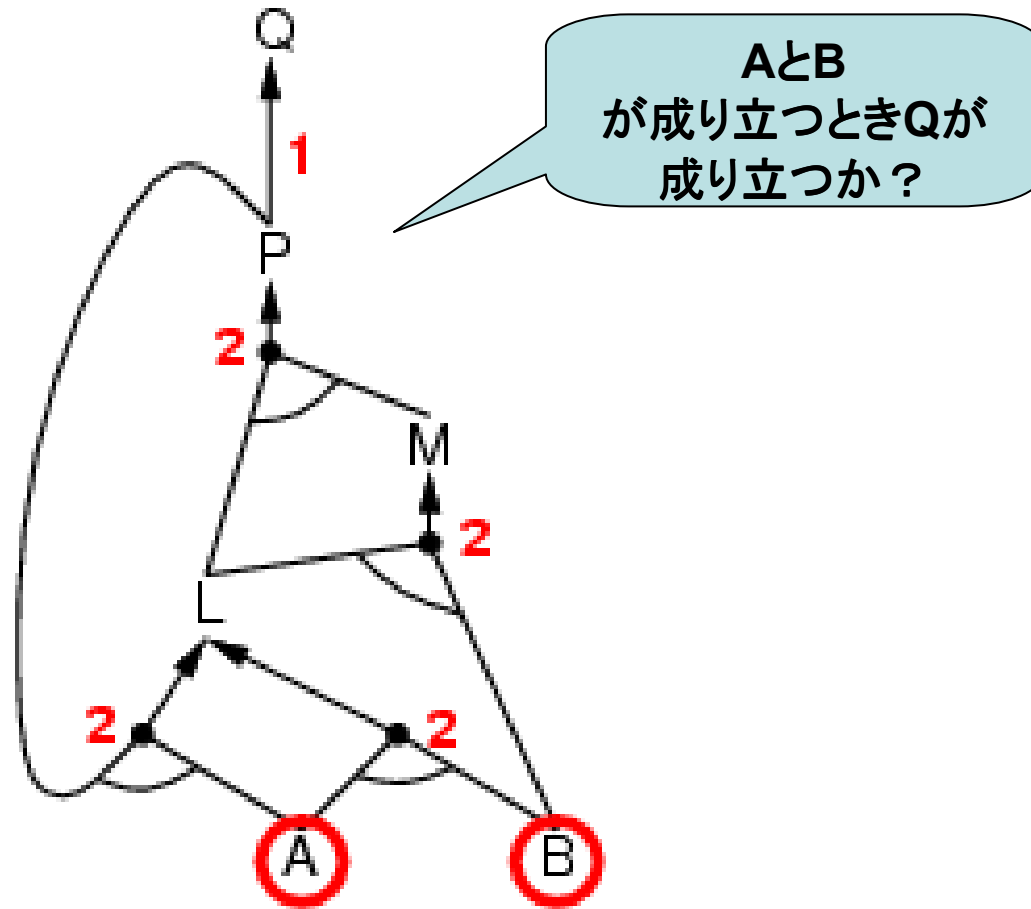
```
function PL-FC-ENTAILS?(KB, q) returns true or false
  local variables: count, a table, indexed by clause, initially the number of premises
                  inferred, a table, indexed by symbol, each entry initially false
                  agenda, a list of symbols, initially the symbols known to be true

  while agenda is not empty do
    p ← POP(agenda)
    unless inferred[p] do
      inferred[p] ← true
      for each Horn clause c in whose premise p appears do
        decrement count[c]
        if count[c] = 0 then do
          if HEAD[c] = q then return true
          PUSH(HEAD[c], agenda)

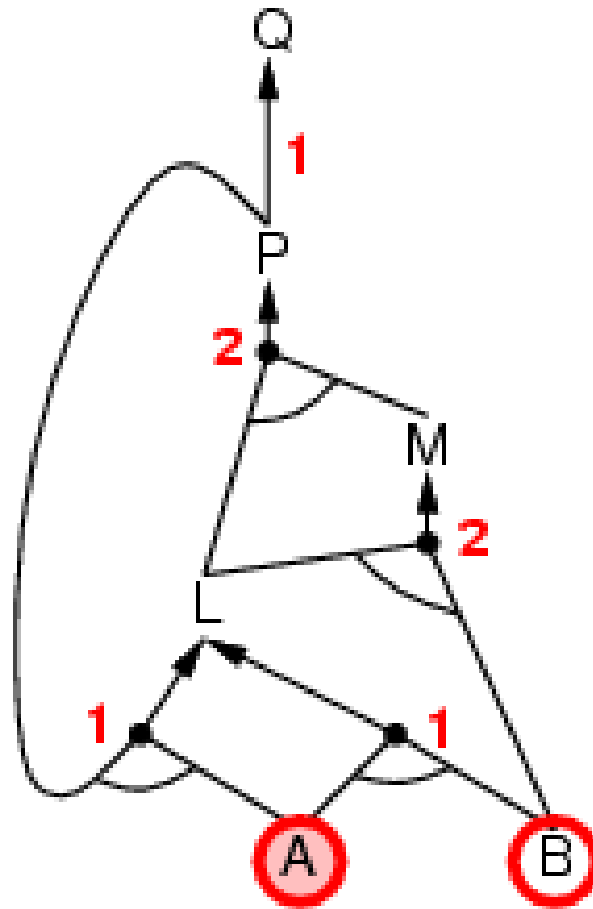
  return false
```

- 前向き連鎖はホーンKBに対して健全で完全である

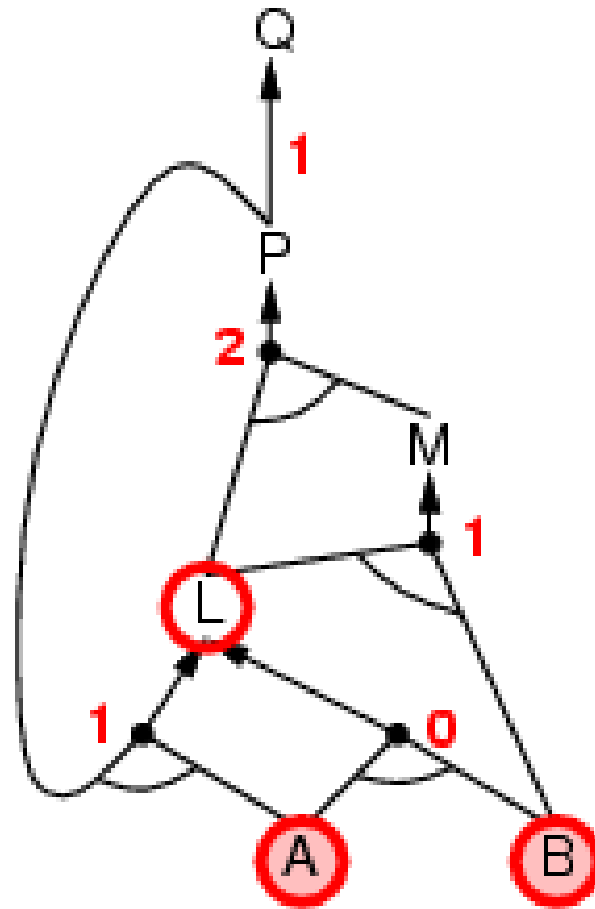
前向き連鎖の例



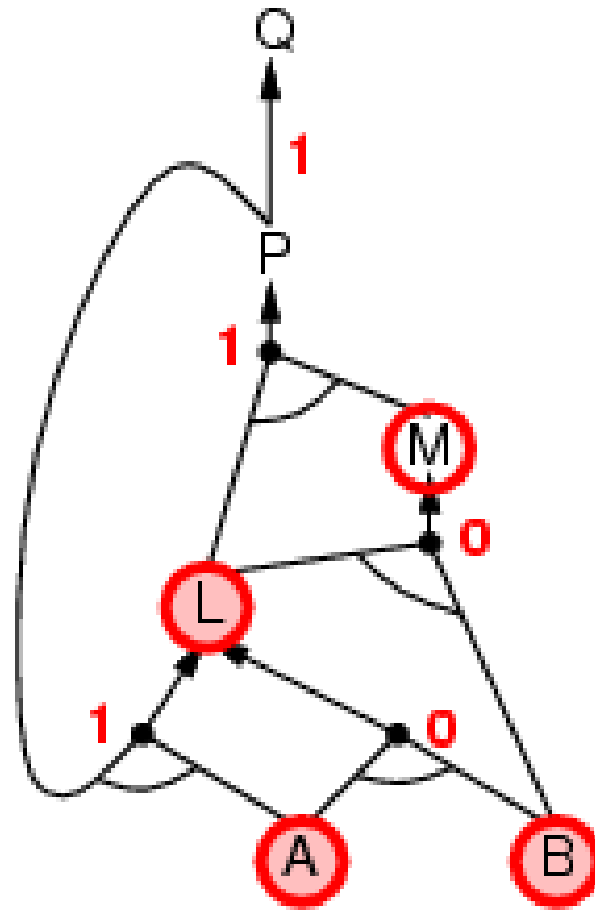
前向き連鎖の例



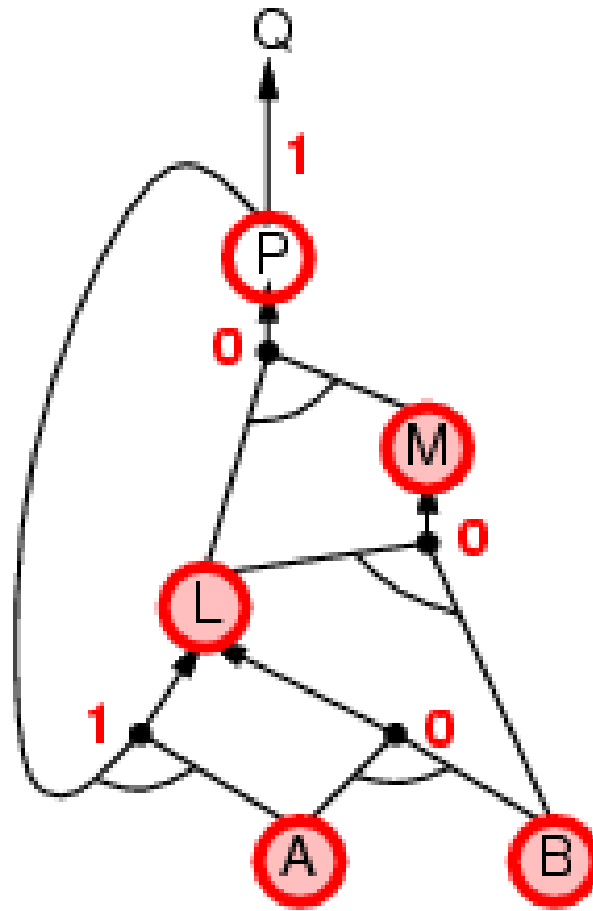
前向き連鎖の例



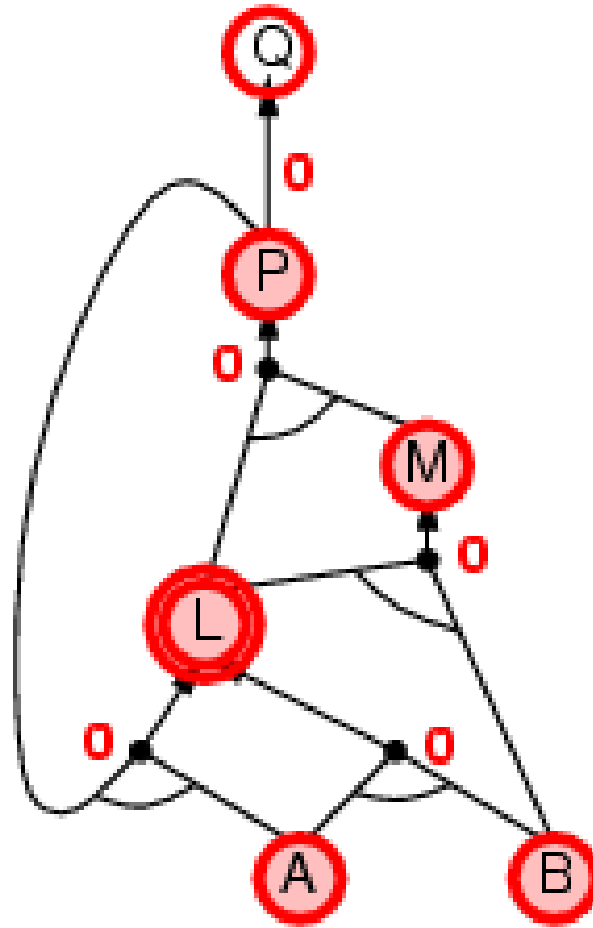
前向き連鎖の例



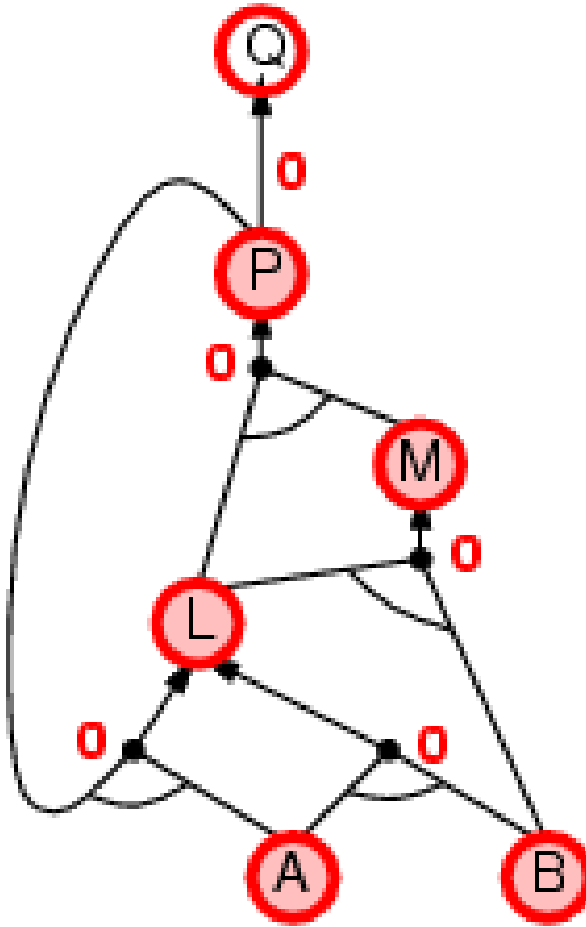
前向き連鎖の例



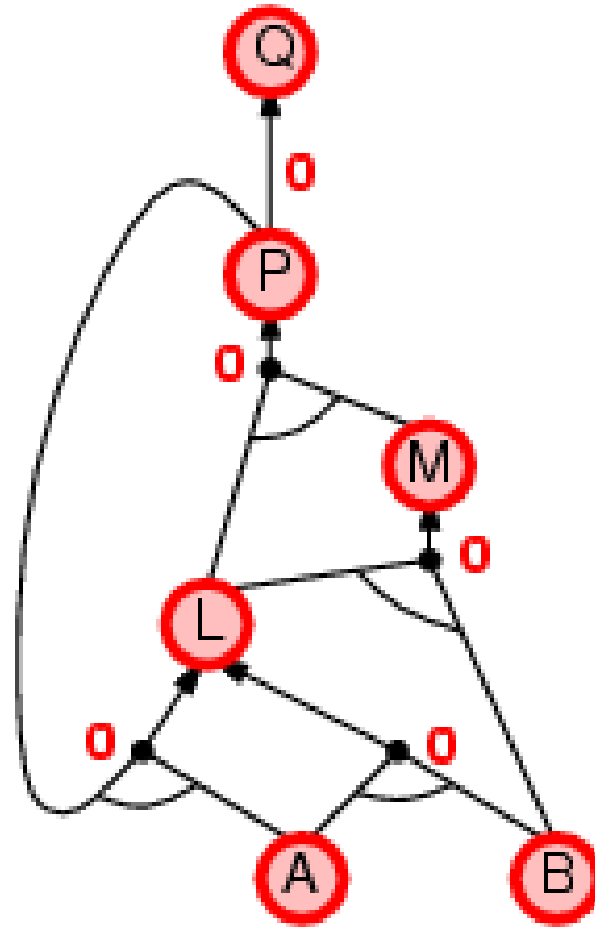
前向き連鎖の例



前向き連鎖の例



前向き連鎖の例



完全性の証明

- 前向き連鎖でKBによって伴意されている原始的な文を導き出す
 1. 前向き連鎖は新たな原始文が導き出されない**固定点**に到着する
 2. シンボルに真・偽を割当てモデル m として最終のゴールを考える
 3. 元のKBの全ての節は m で真である
$$a_1 \wedge \dots \wedge a_k \Rightarrow b$$
 4. 従って m はKBのモデルである
 5. $KB \vdash q$ ならば m を含んでいるKBの**全ての**モデルで q は真である

先の例ではモデルはQAB
(Q,A,Bとも真)

後向き連鎖

考え方: 質問 q から後向きに仕事をする:

後向き連鎖で q を証明するために

q がすでに既知であるか調べる

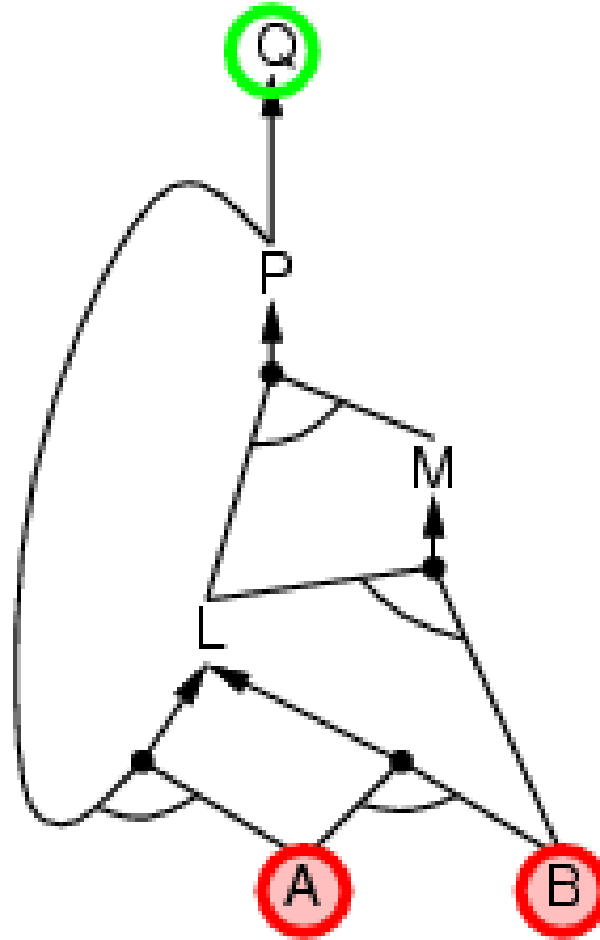
q に結論づけるあるルールの全ての前提を後向き連鎖で証明する

ループを避けるために: サブゴールがゴールスタックに既にあるかを調べる

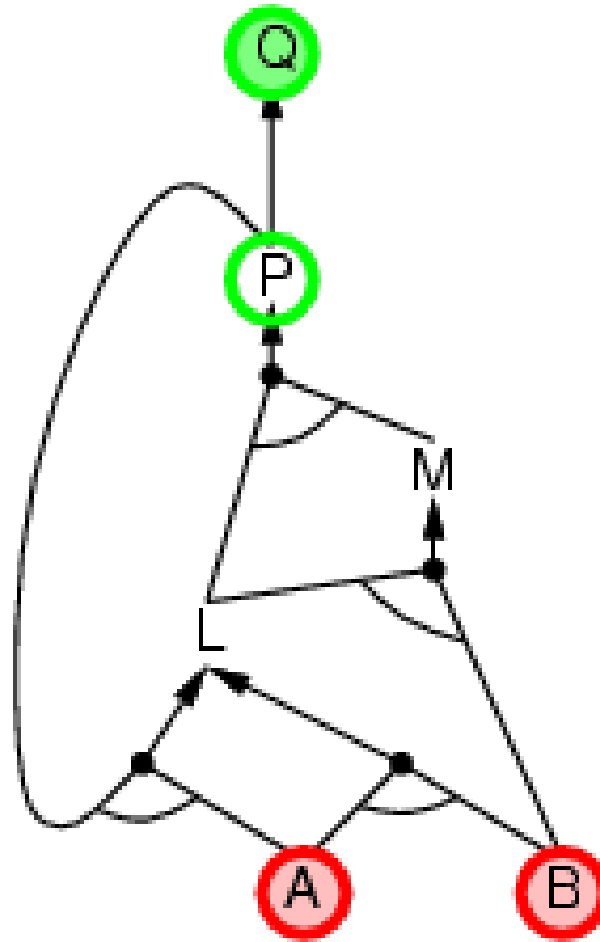
繰り返しの作業を避けるために: 新しいサブゴールが次に該当しているかを調べる

1. すべてに証明されているか
2. 既に失敗しているか

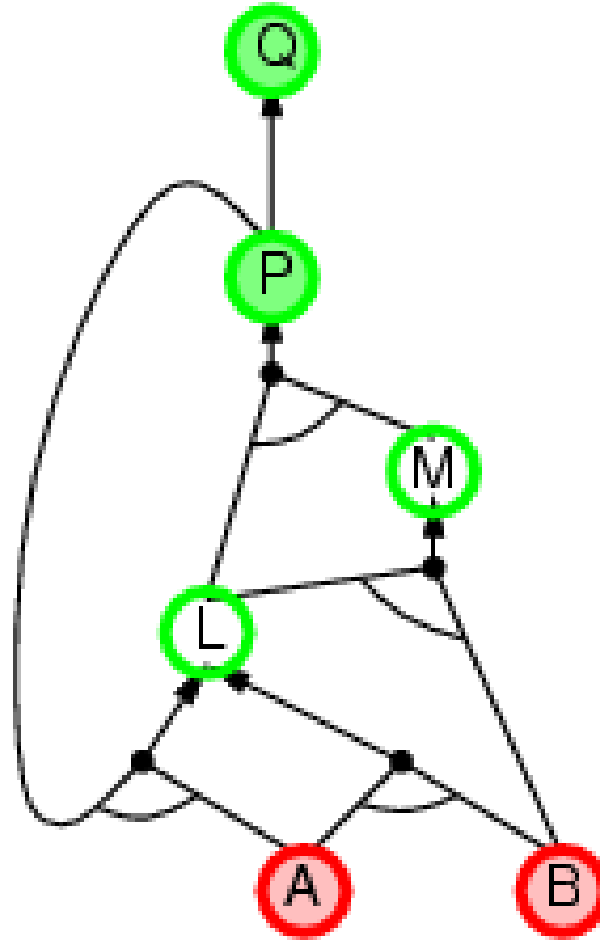
後向き連鎖の例



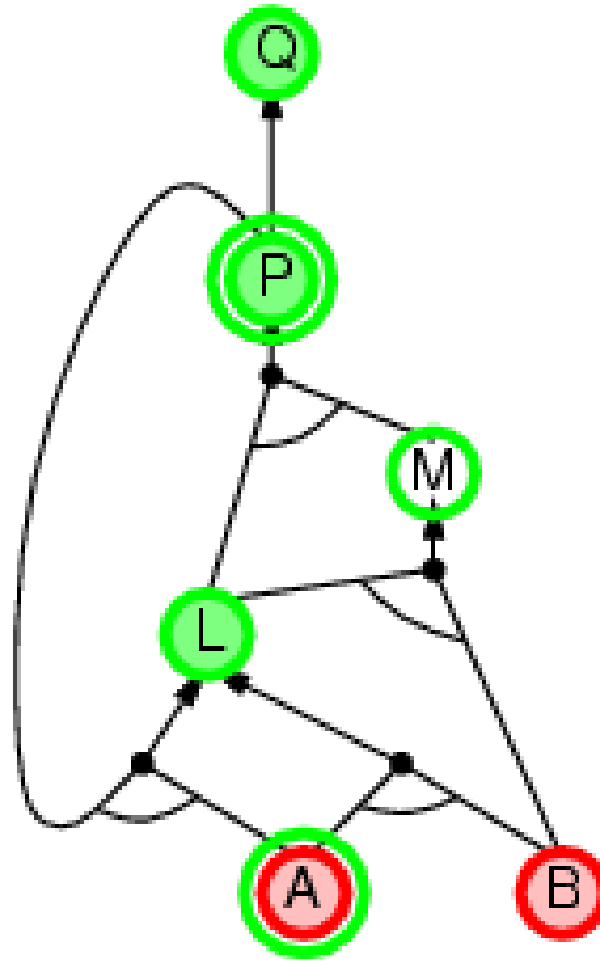
後向き連鎖の例



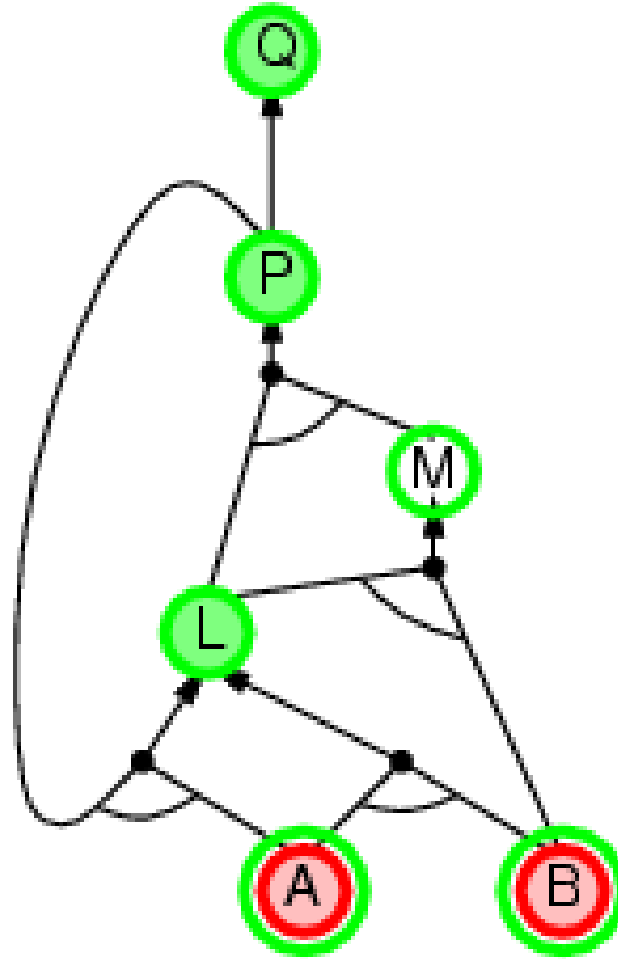
後向き連鎖の例



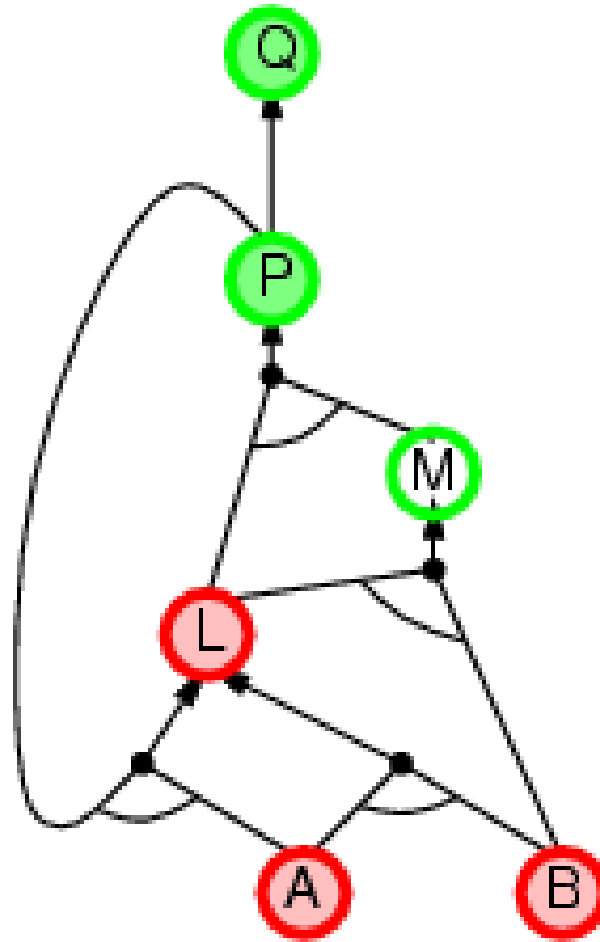
後向き連鎖の例



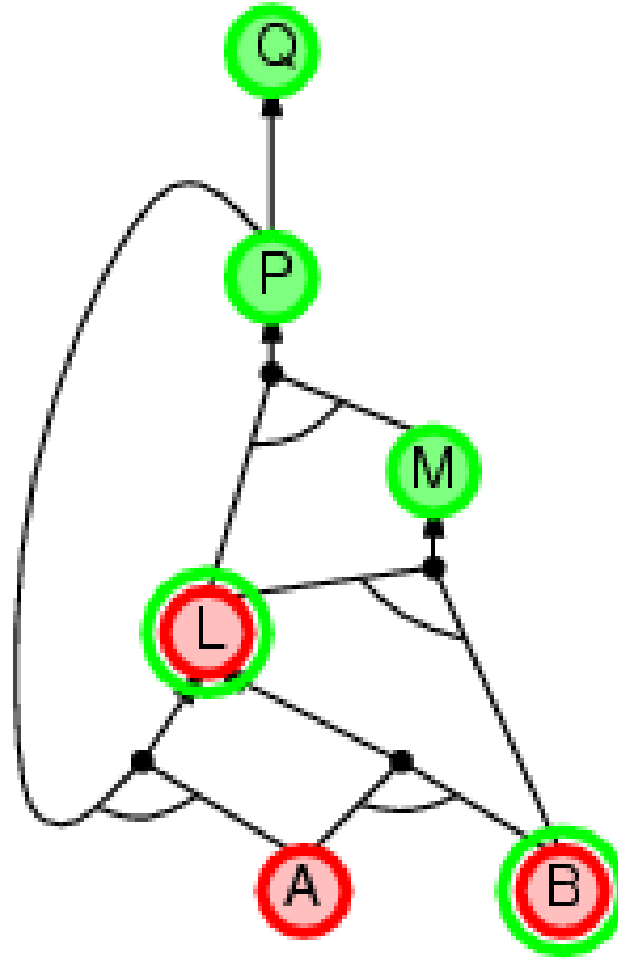
後向き連鎖の例



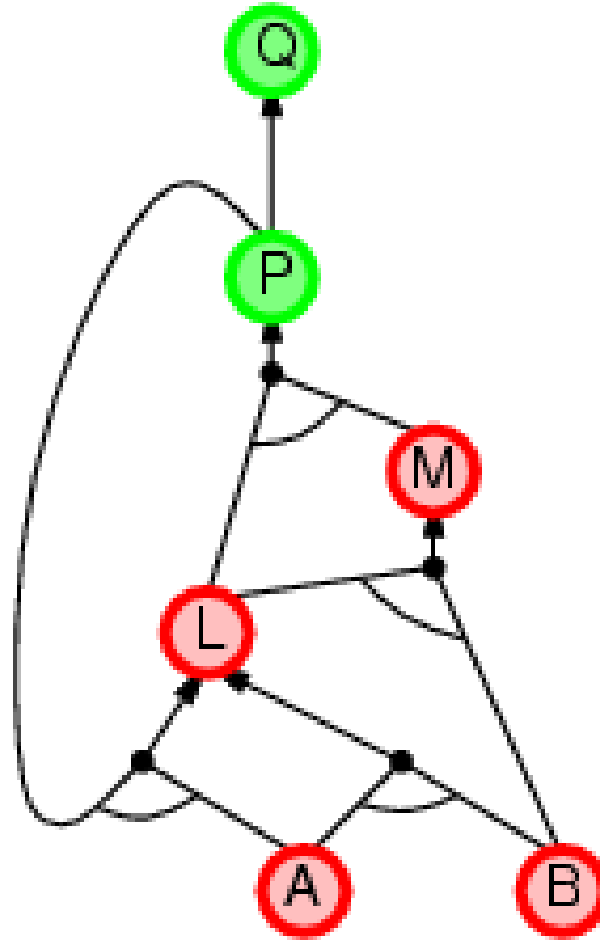
後向き連鎖の例



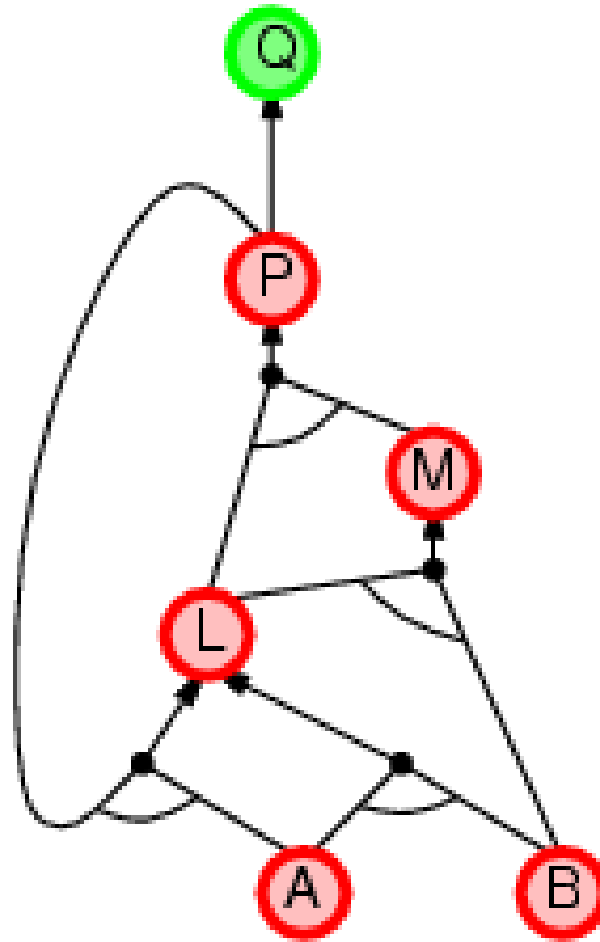
後向き連鎖の例



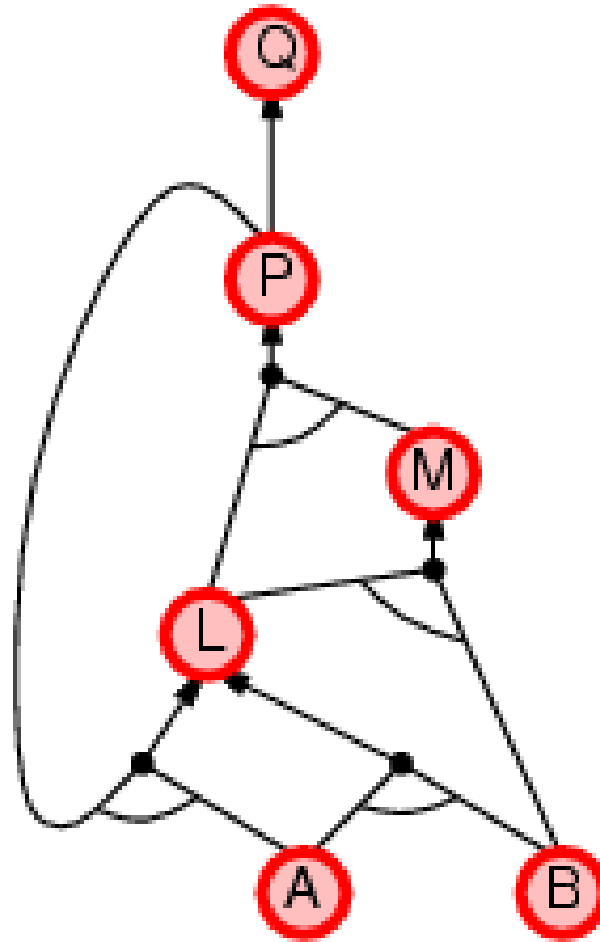
後向き連鎖の例



後向き連鎖の例



後向き連鎖の例



前向き・後向き連鎖

- 前向き連鎖はデータ駆動で自動的に無意識のプロセスである
 - e.g., 物体認識, ルーチンの決定
- ゴールに無関係の作業をたくさんする可能性がある
- 後向き連鎖はゴール駆動で問題の解決に適切である
 - e.g., 私の鍵はどこにあるか? どのようにして博士課程のプログラムに登録したらよいか?
- 後向き連鎖の複雑性はKBの大きさに線形よりはずっと小さい

効率的な命題論理の推論

命題論理の推論に対する2種類の効率的なアルゴリズム:

完全な後戻り探索アルゴリズム

- DPLL アルゴリズム (Davis, Putnam, Logemann, Loveland)
- 不完全な局所探索アルゴリズム
 - WalkSAT algorithm

DPLL アルゴリズム

入力された命題論理の文が満足であれば、決定的である

真理表の数え上げでの改良:

1. 早期の終了

任意のリテラルが真であれば節は真
任意の節が偽であれば文は偽

2. 純粹シンボルでの発見的方法

純粹シンボル: すべての節に同じ符号で現れる

例: 3つの節($A \vee \neg B$), ($\neg B \vee \neg C$), ($C \vee A$)で, AとBは純粹であり, Cは純粹でない
純粹シンボルのリテラルを真とする

3. 単一節での発見的方法

単一節: 節に一つのリテラル

単一節での唯一のリテラルは真でないといけない

DPLLアルゴリズム

function DPLL-SATISFIABLE?(*s*) **returns** *true* or *false*

inputs: *s*, a sentence in propositional logic

clauses ← the set of clauses in the CNF representation of *s*

symbols ← a list of the proposition symbols in *s*

return DPLL(*clauses*, *symbols*, [])

function DPLL(*clauses*, *symbols*, *model*) **returns** *true* or *false*

if every clause in *clauses* is true in *model* **then return** *true*

if some clause in *clauses* is false in *model* **then return** *false*

P, *value* ← FIND-PURE-SYMBOL(*symbols*, *clauses*, *model*)

if *P* is non-null **then return** DPLL(*clauses*, *symbols*-*P*, [*P* = *value* | *model*])

P, *value* ← FIND-UNIT-CLAUSE(*clauses*, *model*)

if *P* is non-null **then return** DPLL(*clauses*, *symbols*-*P*, [*P* = *value* | *model*])

P ← FIRST(*symbols*); *rest* ← REST(*symbols*)

return DPLL(*clauses*, *rest*, [*P* = *true* | *model*]) **or**

DPLL(*clauses*, *rest*, [*P* = *false* | *model*])

WalkSATアルゴリズム

- 不完全な局所検索アルゴリズム
- 評価関数: 不満足な節の数を最小化するときの最小衝突の発見的方法
- 貪欲とランダムとのバランス

WalkSATアルゴリズム

```
function WALKSAT(clauses, p, max-flips) returns a satisfying model or failure  
inputs: clauses, a set of clauses in propositional logic  
         p, the probability of choosing to do a “random walk” move  
         max-flips, number of flips allowed before giving up  
  
model ← a random assignment of true/false to the symbols in clauses  
for i = 1 to max-flips do  
    if model satisfies clauses then return model  
    clause ← a randomly selected clause from clauses that is false in model  
    with probability p flip the value in model of a randomly selected symbol  
        from clause  
    else flip whichever symbol in clause maximizes the number of satisfied clauses  
return failure
```

困難な満足性問題

- ランダムな3-連言標準形の文が満足であるかを考える

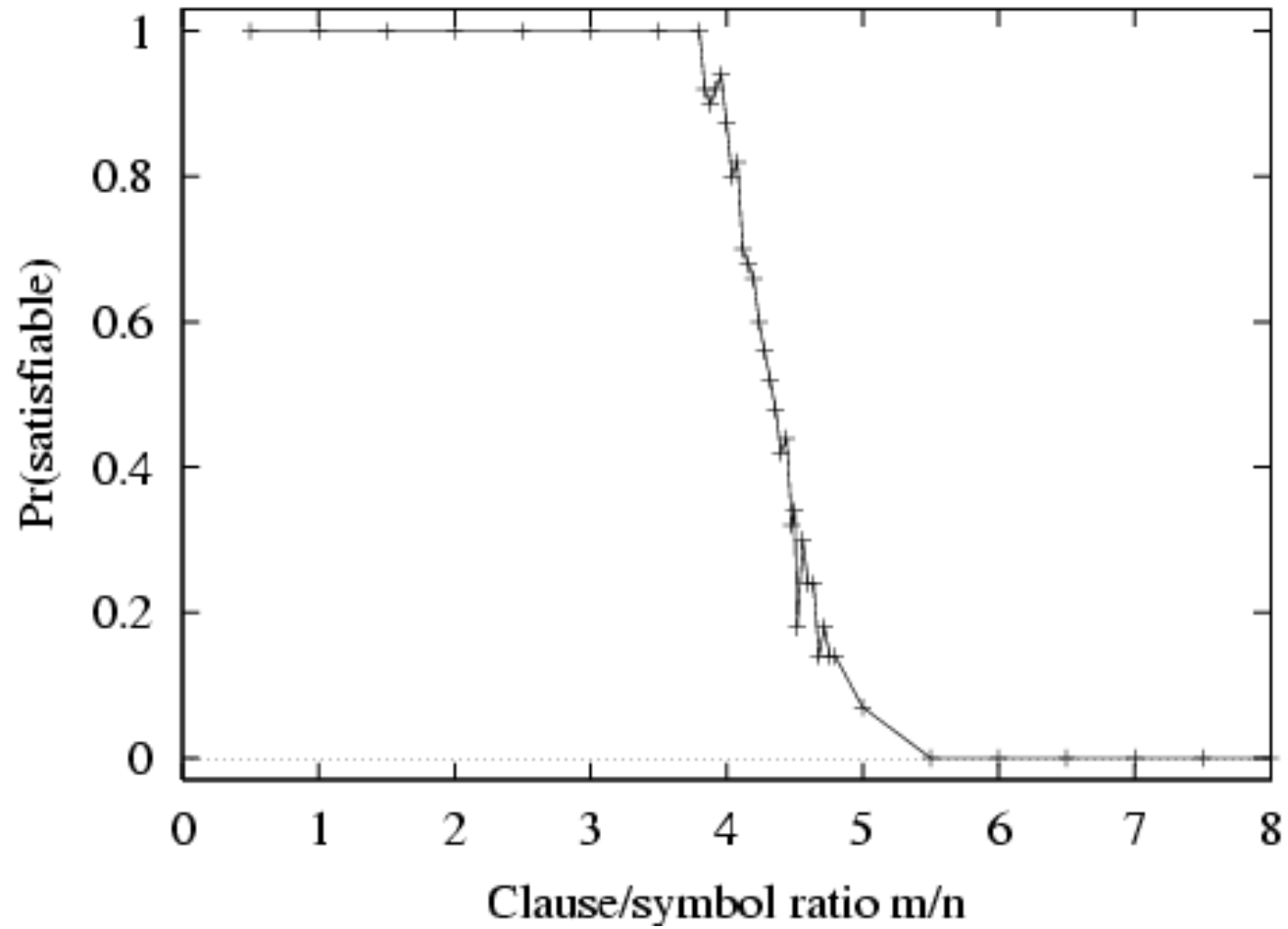
$$(\neg D \vee \neg B \vee C) \wedge (B \vee \neg A \vee \neg C) \wedge (\neg C \vee \neg B \vee E) \wedge (E \vee \neg D \vee B) \wedge (B \vee E \vee \neg C)$$

m = 節の数

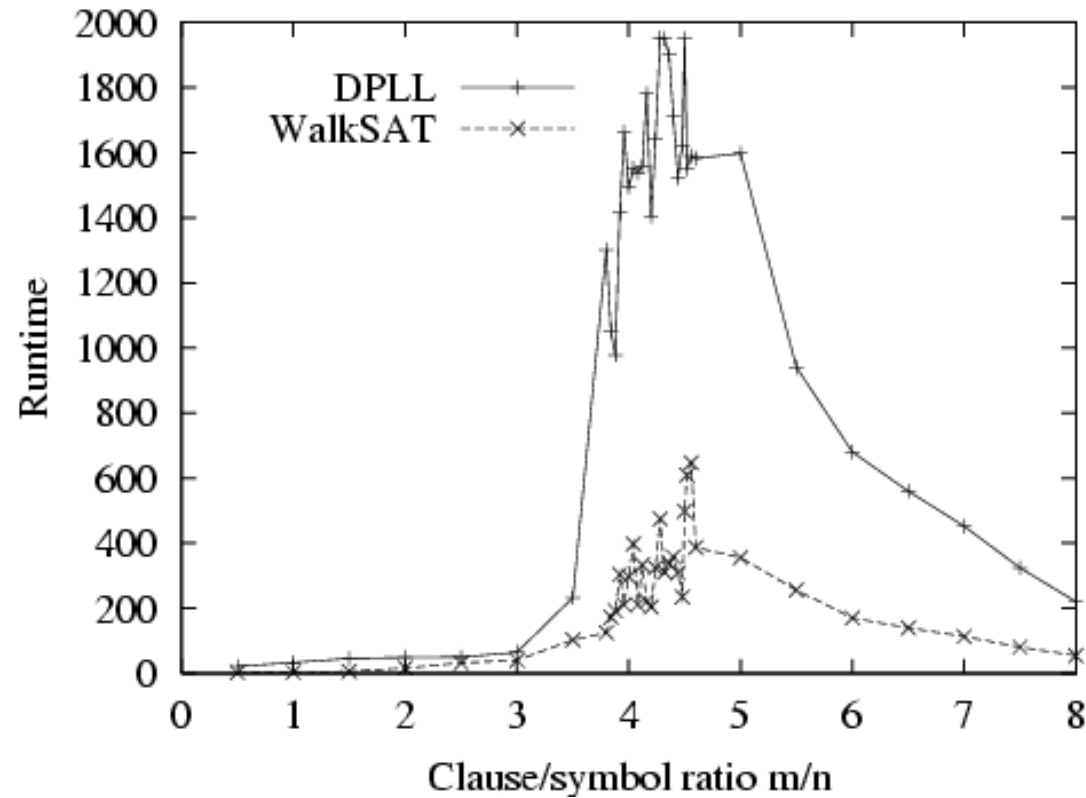
n = シンボルの数

- 困難な問題は $m/n = 4.3$ (critical point) のクラスに近いときである

困難な満足性問題



困難な満足性問題



- 満足性のランダムな3-連言標準形の100の文に対する平均の実行時間、ここで $n = 50$

wumpusの世界での推論ベース のエージェント

命題論理を用いてのwumpusの世界のエージェント:

$$\neg P_{1,1}$$

$$\neg W_{1,1}$$

$$B_{x,y} \Leftrightarrow (P_{x,y+1} \vee P_{x,y-1} \vee P_{x+1,y} \vee P_{x-1,y})$$

$$S_{x,y} \Leftrightarrow (W_{x,y+1} \vee W_{x,y-1} \vee W_{x+1,y} \vee W_{x-1,y})$$

$$W_{1,1} \vee W_{1,2} \vee \dots \vee W_{4,4}$$

$$\neg W_{1,1} \vee \neg W_{1,2}$$

$$\neg W_{1,1} \vee \neg W_{1,3}$$

...

⇒ 64の独立した命題シンボル, 155の文

```

function PL-WUMPUS-AGENT(percept) returns an action
inputs: percept, a list, [stench, breeze, glitter]
static: KB, initially containing the “physics” of the wumpus world
         x, y, orientation, the agent’s position (init. [1,1]) and orient. (init. right)
         visited, an array indicating which squares have been visited, initially false
         action, the agent’s most recent action, initially null
         plan, an action sequence, initially empty

update x, y, orientation, visited based on action
if stench then TELL(KB, Sx,y) else TELL(KB, ¬ Sx,y)
if breeze then TELL(KB, Bx,y) else TELL(KB, ¬ Bx,y)
if glitter then action ← grab
else if plan is nonempty then action ← POP(plan)
else if for some fringe square [i,j], ASK(KB, (¬ Pi,j ∧ ¬ Wi,j)) is true or
         for some fringe square [i,j], ASK(KB, (Pi,j ∨ Wi,j)) is false then do
           plan ← A*-GRAPH-SEARCH(ROUTE-PB([x,y], orientation, [i,j], visited))
           action ← POP(plan)
else action ← a randomly chosen move
return action

```

命題論理の表現上の制約

- KBは全ての一つ一つの四角に対して“物理的な”文章を含んでいる
- それぞれの時刻 t にそしてそれぞれの場所 $[x,y]$ に対して
$$L_{x,y}^t \wedge FacingRight^t \wedge Forward^t \Rightarrow L_{x+1,y}^t$$
- 節の急速な増加

まとめ

- 論理エージェントは新しい情報を導き出すために、そして、決定をするために知識ベースに推論を適応する
- 論理の基本的な概念:
 - 構文: 文の形式的な構造
 - 意味論: モデルに関連しての文の真
 - 伴意: 二つの文があったとき一方の文が真となる全ての世界で他方の文も真となる
 - 推論: 他の文から文を導き出すこと
 - 健全性: 伴意された文のみを導出は生成する
 - 完全性: 伴意された文すべてを導出は生成する
- Wumpusの世界は部分的で否定の情報や場合によっては推論を必要とする
- 融合は命題論理に対して完全である。前向き・後向き連鎖はホーン節に対して線形な時間であり、完全である
- 命題論理は表現力に欠ける