

ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ И КОМПЈУТЕРСКО ИНЖЕНЕРСТВО

# Логички агенти и предикатно сметање



# Логички агенти

- Логичките агенти се агентите способни за расудување (reasoning)
- Тие расудуваат врз база на претходно стекнато знаење
- Логичките агенти се агенти кои се темелат врз знаењето (knowledge-based agents)



# Основни елементи на знаењето

- Претпоставка или предуслов
- Последица или заклучок
- Правило врз основа на кое од предусловите може да се изведе заклучокот
- Заклучок: логичките агенти заклучуваат (infer) како врз основа на базата на знаење да извлечат некој нов факт или да донесат одлука.



# Основни типови расудување

- Дедукција: извлекување заклучок врз основа на претпоставките и правилата (одделување)
- Индукција: одредување правило врз основа на постоечките претпоставки (обопштување)
- Абдукција: одредување на причините кои доведуваат до заклучокот (објаснување или дијагностика)



# Примери за дедукција

- Претпоставка: Сега врне.
- Правило: Кога врне, улиците се водени.
- Заклучок: Улиците се водени.
  
- Претпоставки:
  - Сите луѓе се живи суштества.
  - Живите суштества дишат.
- Правило: силогизам
- Заклучок: Луѓето дишат.



# Примери за индукција

## ■ Претпоставки:

- ☐ Луѓето дишат.
- ☐ Животните дишат.
- ☐ Растенијата дишат.
- ☐ Луѓето се живи суштества.
- ☐ Животните се живи суштества.
- ☐ Растенијата се живи суштества.

## ■ Заклучок:

- ☐ Сите живи суштества дишат.

# Пример за неточна индукција

- Канаринците се птици.
- Славејчињата се птици.
- Врапчињата се птици.
- Канаринците се мали.
- Славејчињата се мали.
- Врапчињата се мали.
  
- Сите птици се мали.



# Примери за абдукција

- Грипот предизвикува висока температура.
- Петре има висока температура.
- Причина:
  - ☐ Петре има грип.
  
- Грипот предизвикува висока температура.
- Ангината предизвикува висока температура.
- Петре има висока температура.
- Причина:
  - ☐ Петре има грип или
  - ☐ Петре има ангина.



# Точноста на трите типа расудување (reasoning)

## ■ Дедукција

- Секогаш кога претпоставките се постојано точни (но не и ако нивната вистинитост се менува со тек на времето)

## ■ Индукција

- Секогаш кога множеството на кое се однесува обопштувањето е комплетно

## ■ Абдукција

- Ако множеството причини е комплетно, тогаш може да се одредат кандидатите причини кои довеле до последицата (која е видлива).

# Организација на знаењето

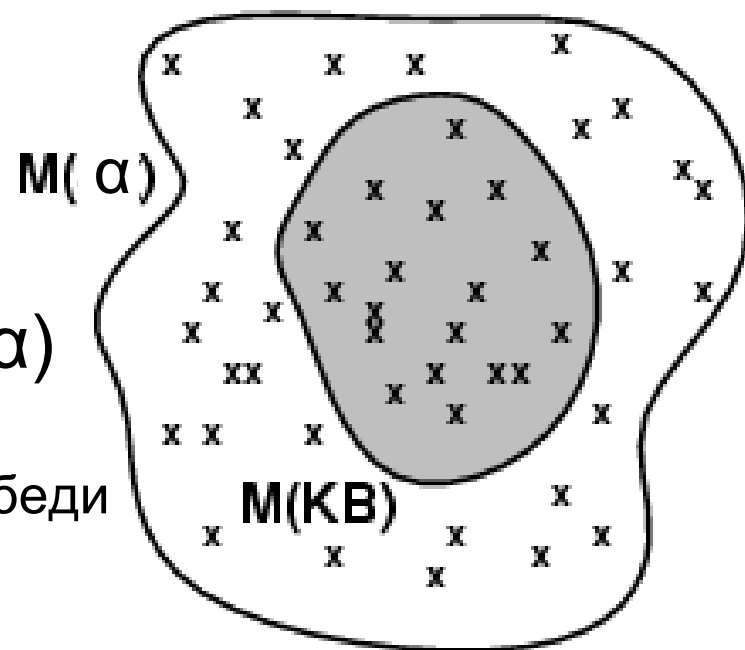
- База на знаење (knowledge base) = множество точни факти или тврдења со кои се опишува еден домен
- Правила за заклучување (inference rules) = општи алгоритми кои не зависат од доменот на системот

# Исказно сметање (propositional calculus)

- Исказно сметање, т.е. Булова логика
- Главен оператор: повлекува (entail)  $\models$
- Интерпретација:
  - базата на знаење повлекува еден исказ ако и само ако (акко) тој исказ е точен во сите светови во кои важи таа база на знаење.

# Модели

- Моделите се формално структурирани светови во однос на кои се оценува вистината
- Велиме дека  $m$  е модел на исказот  $\alpha$  ако  $\alpha$  е точен во  $m$
- $M(\alpha)$  е множество на сите модели на  $\alpha$
- Тогаш  $KB \models \alpha$  ако  $M(KB) \subseteq M(\alpha)$ 
  - пр.  $KB = \text{МЗТ победи}$  и  $\text{Работнички победи}$   
 $\alpha = \text{МЗТ победи}$



# Синтаксата на исказното сметање (1)

- Атоми:  $T$  (true) и  $F$  (false) и ознаки што започнуваат со голема буква:  $P, Q, R, P_1, P_2, \dots$
- Сврзници:  $\wedge, \vee, \neg, \Rightarrow$  и  $\Leftrightarrow$
- Реченица е секоја добро создадена формула дсф (well-formed formula wff)

# Синтаксата на исказното сметање (2)

- Секој атом е дсф.
- Ако  $a_1$  и  $a_2$  се дсф, тогаш дсф се и:  
 $a_1 \wedge a_2$ ,  $a_1 \vee a_2$ ,  $\neg a_1$ ,  $a_1 \Rightarrow a_2$ ,  $a_1 \Leftrightarrow a_2$
- Атомите и атомите пред кои стои негација се викаат литерали.
- Во импликацијата, првата дсф се вика претходник (antecedent), а втората последица или исход (consequent)

# Правила за заклучување

- $P, P \Rightarrow Q \quad / \quad Q$  (модус поненс)
  - $P, Q \quad / \quad P \wedge Q$  (“и” воведување)
  - $P, Q, P \wedge Q \quad / \quad Q \wedge P$  (комутативност)
  - $P \wedge Q \quad / \quad \text{важи и само } P \text{ или само } Q$  (“и” елиминација)
  - било од  $P$  или од  $Q \quad / \quad P \vee Q$  (“или” воведување)
  - $\neg (\neg P) \quad / \quad P$  (“не” елиминација)
- \* Имињата на правилата се според Nilsson. Кај него, дсф се означени со мали омеги.



# Логички еквиваленции (1)

- Импликацијата изразена како дисјункција

$$(P \Rightarrow Q) \Leftrightarrow (\neg P \vee Q)$$

- Комутативност и асоцијативност на:

- конјункцијата:

$$(P \wedge Q) \Leftrightarrow (Q \wedge P) \quad \text{и} \quad (P \wedge (Q \wedge R)) \Leftrightarrow ((P \wedge Q) \wedge R)$$

- дисјункцијата

$$(P \vee Q) \Leftrightarrow (Q \vee P) \quad \text{и} \quad (P \vee (Q \vee R)) \Leftrightarrow ((P \vee Q) \vee R)$$

# Логички еквиваленции (2)

- Дистрибутивност на конјункцијата во однос на дисјункцијата и обратно
  - конјункцијата во однос на дисјункцијата:
$$(P \wedge (Q \vee R)) \Leftrightarrow (P \wedge Q) \vee (P \wedge R)$$
  - дисјункцијата во однос на конјункцијата:
$$(P \vee (Q \wedge R)) \Leftrightarrow (P \vee Q) \wedge (P \vee R)$$
- Контрапозиција:
  - $(P \Rightarrow Q) \Leftrightarrow (\neg Q \Rightarrow \neg P)$

# Логички еквиваленции (3)

- Еквиваленцијата изразена како конјункција на импликации:

$$(P \Leftrightarrow Q) \Leftrightarrow ((P \Rightarrow Q) \wedge (Q \Rightarrow P))$$

- Де Морганови закони

- за конјункцијата:

$$(\neg(P \wedge Q)) \Leftrightarrow (\neg P \vee \neg Q)$$

- за дисјункцијата:

$$(\neg(P \vee Q)) \Leftrightarrow (\neg P \wedge \neg Q)$$



# Техники на расудувањето

- Верижење нанапред (forward chaining)
  - Верижење наназад (backward chaining)
  - Претпоставка: условно претставување на знаењето
- 
- Резолуција (resolution)
  - Претпоставка: конјунктивно претставување на знаењето



# Верижење нанапред

- Forward chaining
- Воспоставување верига од литерали и дисјункција на литерали од кои најмногу една е позитивна.
- Поаѓа од податоците, па што се добие - се добие (data driven reasoning)
- Знаењето се прикажува со И/ИЛИ дрва

# Знаењето може да се прикажува со И/ИЛИ дрва

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

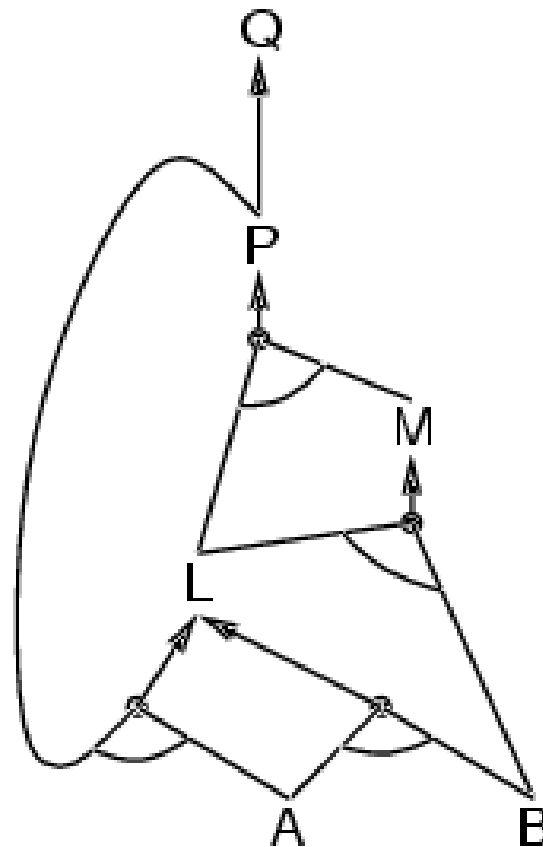
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

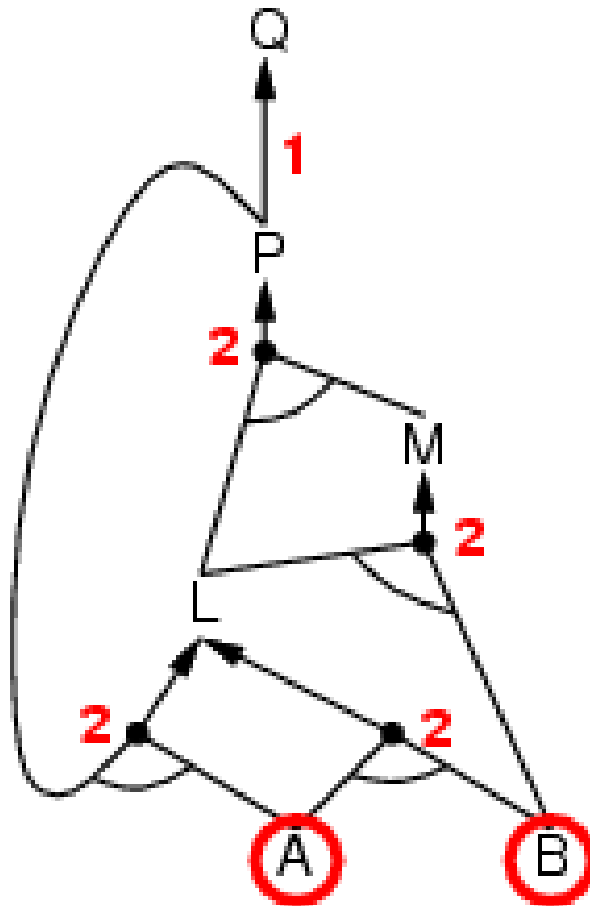
$$A \wedge B \Rightarrow L$$

$A$

$B$

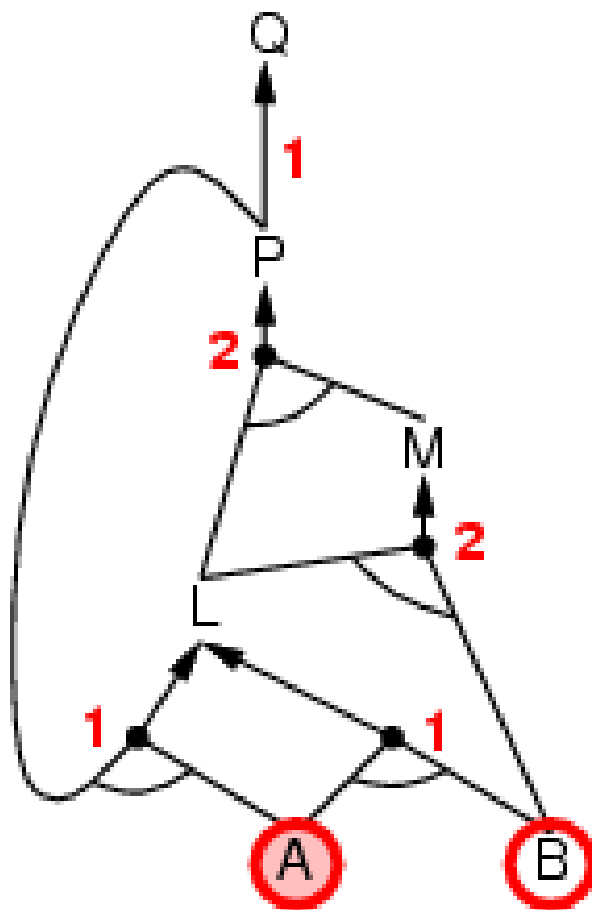


# Пример за верижење напред

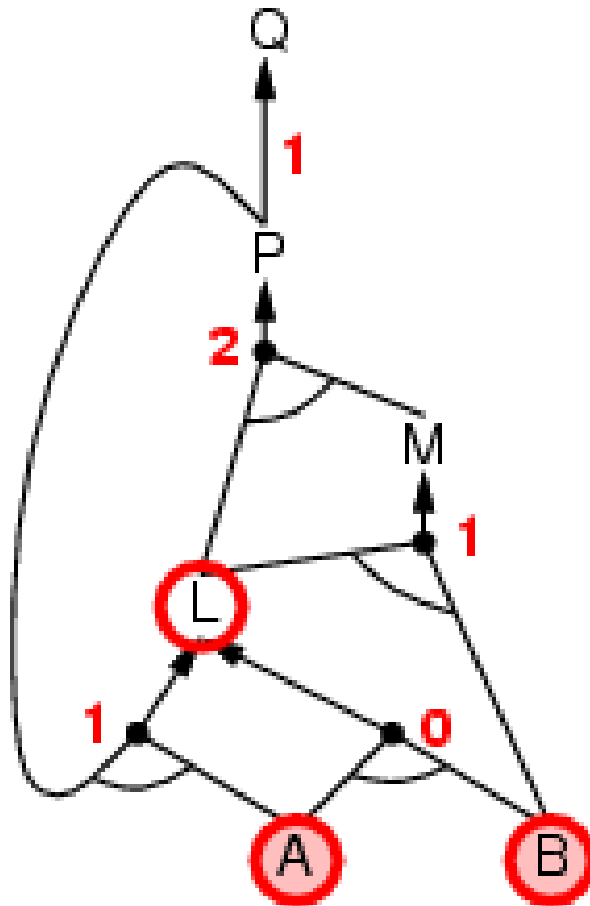




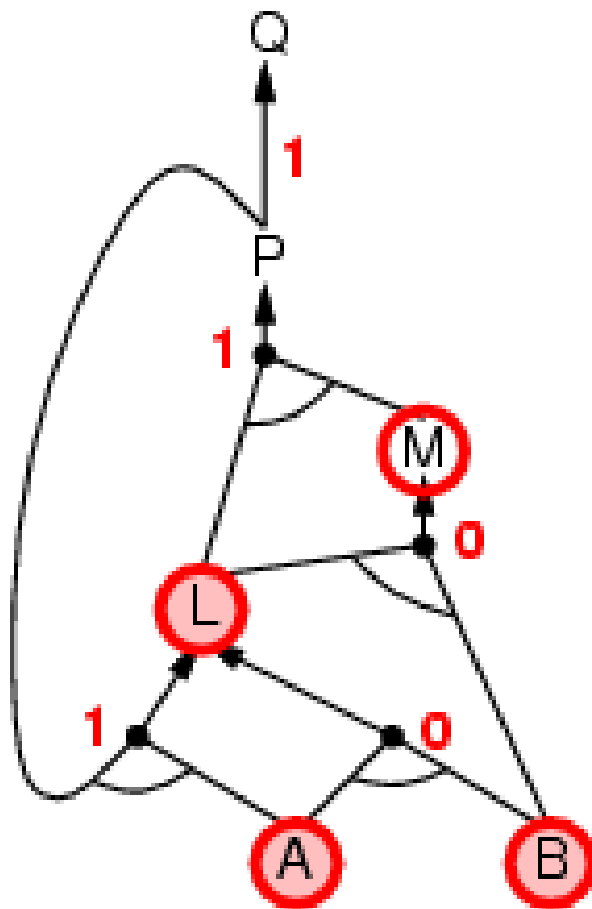
# Пример за верижење напред



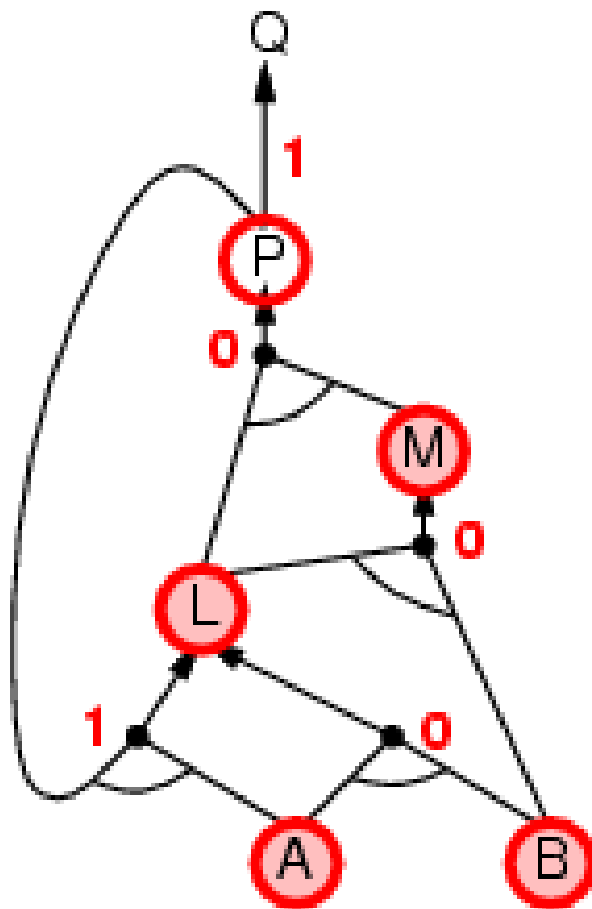
# Пример за верижење нанапред



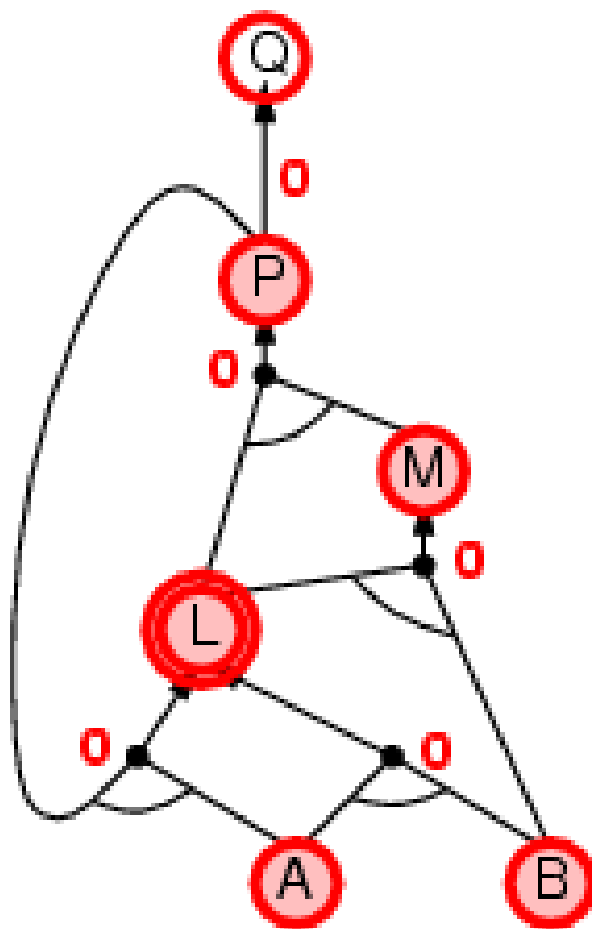
# Пример за верижење нанапред



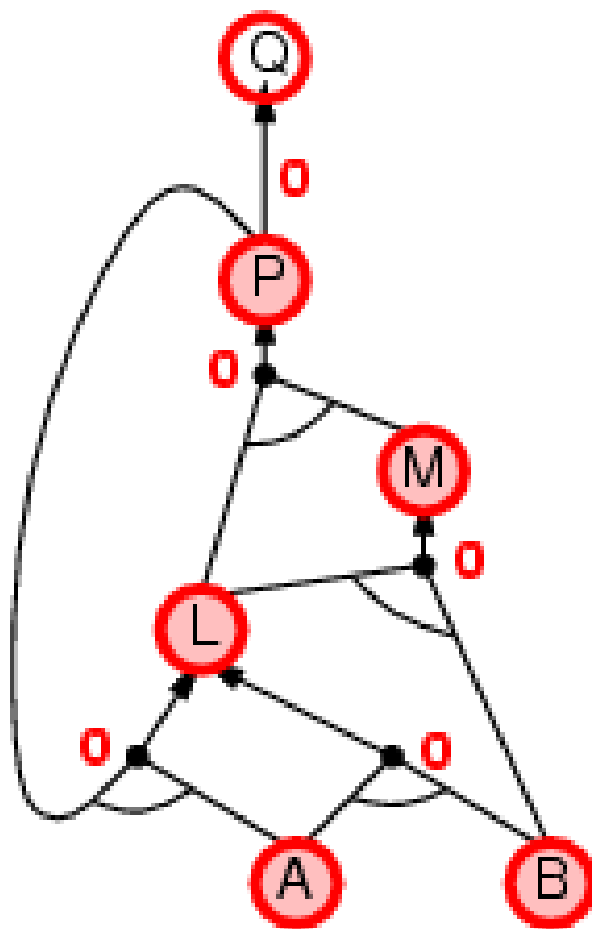
# Пример за верижење нанапред



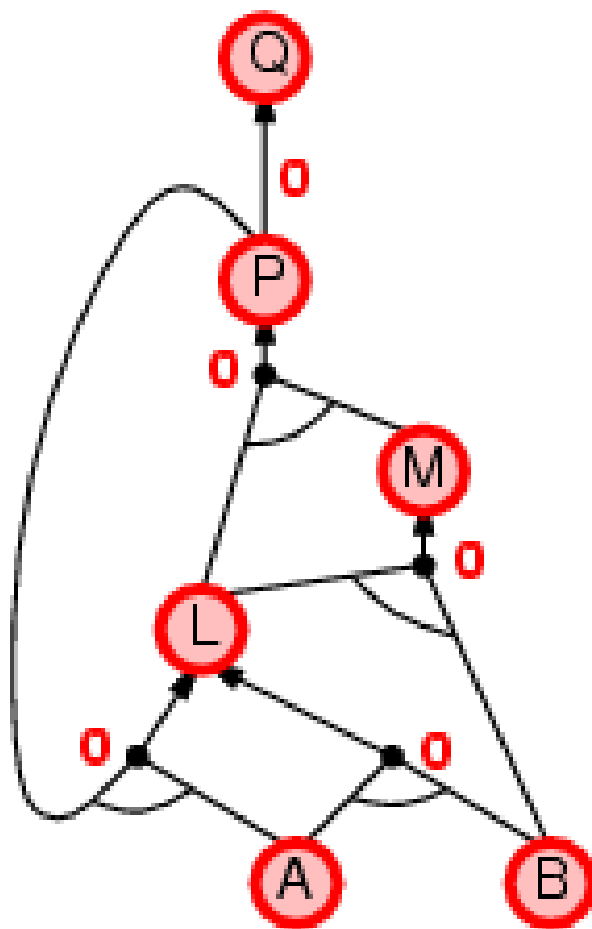
# Пример за верижење нанапред



# Пример за верижење нанапред



# Пример за верижење нанапред



# Верижење наназад

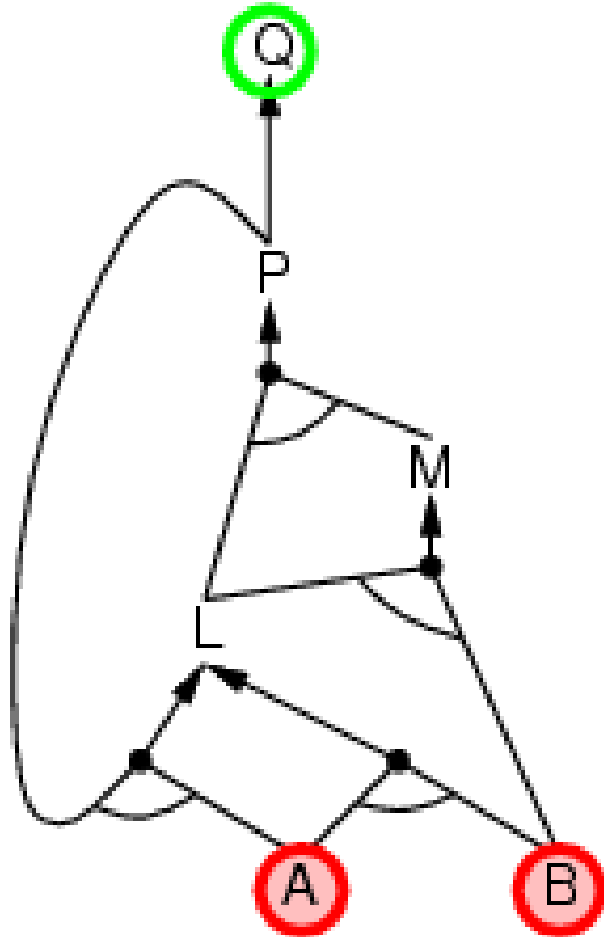
- Воспоставување верига од литерали и дисјункција на литерали од кои најмногу една е позитивна.
- Се стреми кон докажување на целите што резултираат со одреден настан (goal directed reasoning)
- Знаењето се прикажува со И/ИЛИ дрва



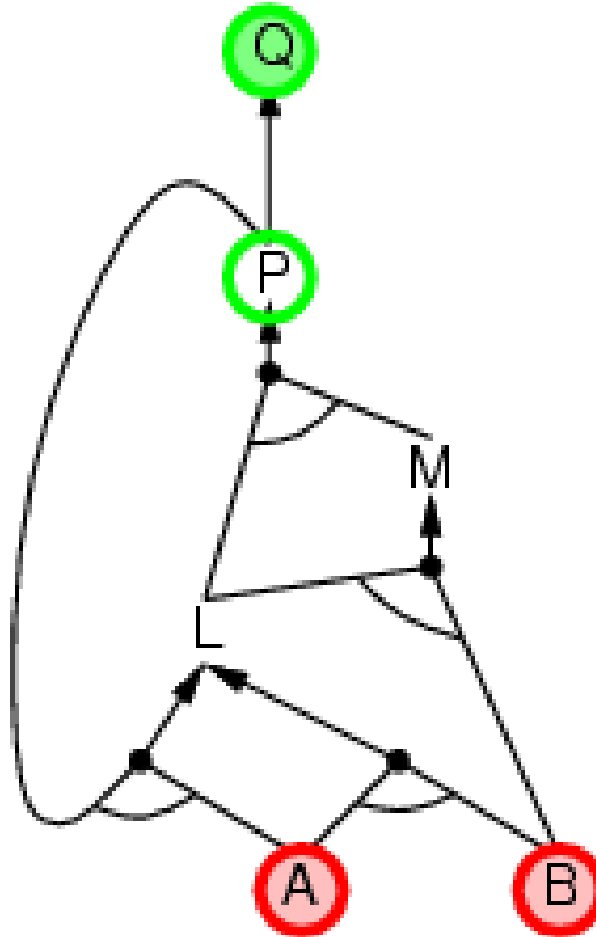
# Суштина на верижењето наназад

- Да се одговори на прашањето дали е точна целта.
- Стратегија: да се докаже целта, поаѓајќи од последиците кон причините
- Постои правило: Ако потцел, тогаш цел
- Ако во низата се најде факт кој ја потврдува потцелта, тогаш целта е исполнета.

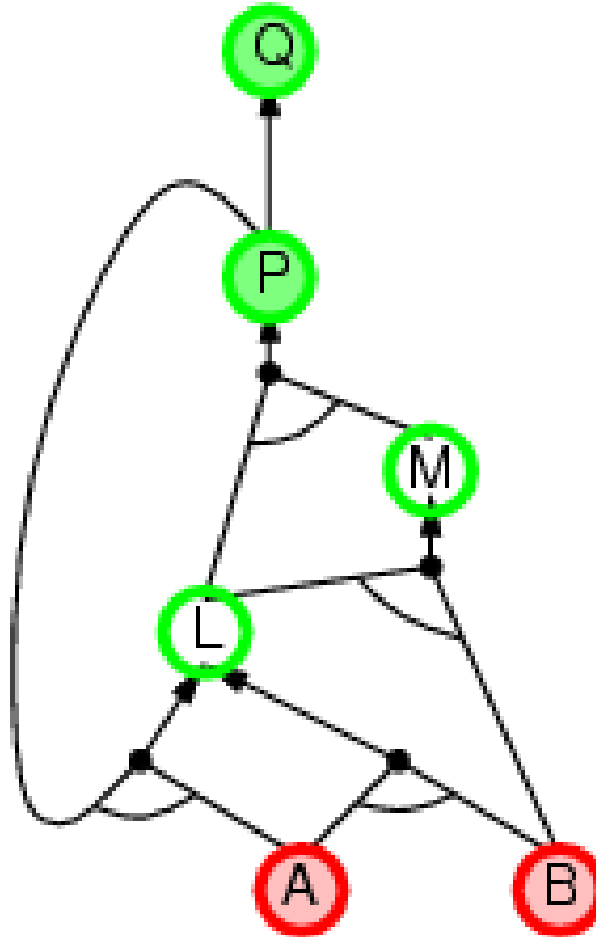
# Пример за верижење наназад



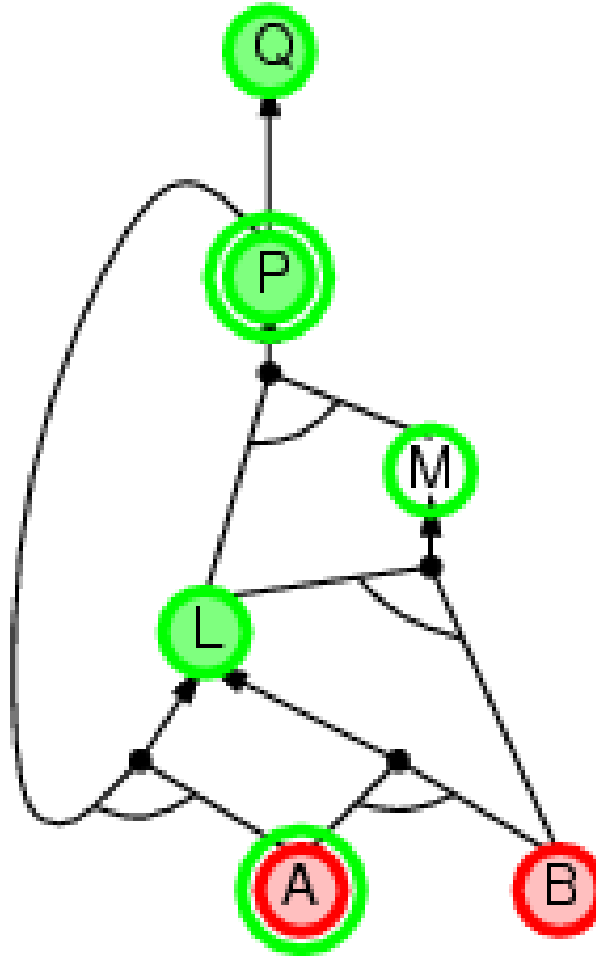
# Пример за верижење наназад



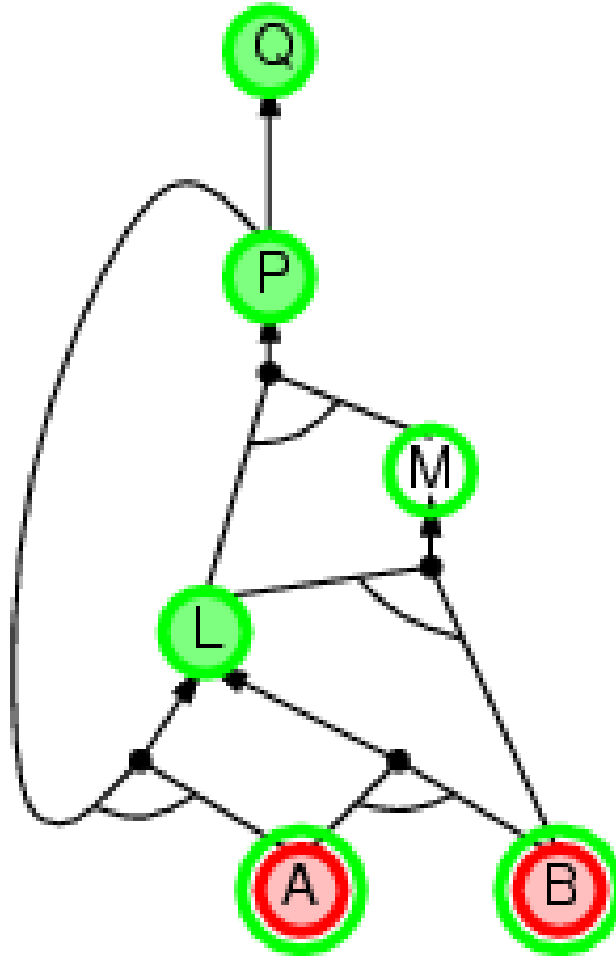
# Пример за верижење наназад



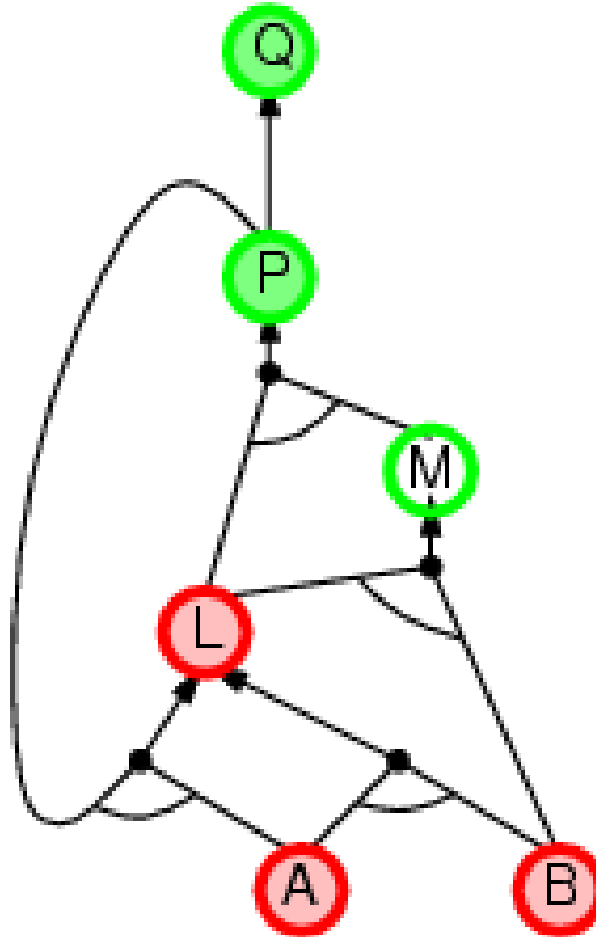
# Пример за верижење наназад



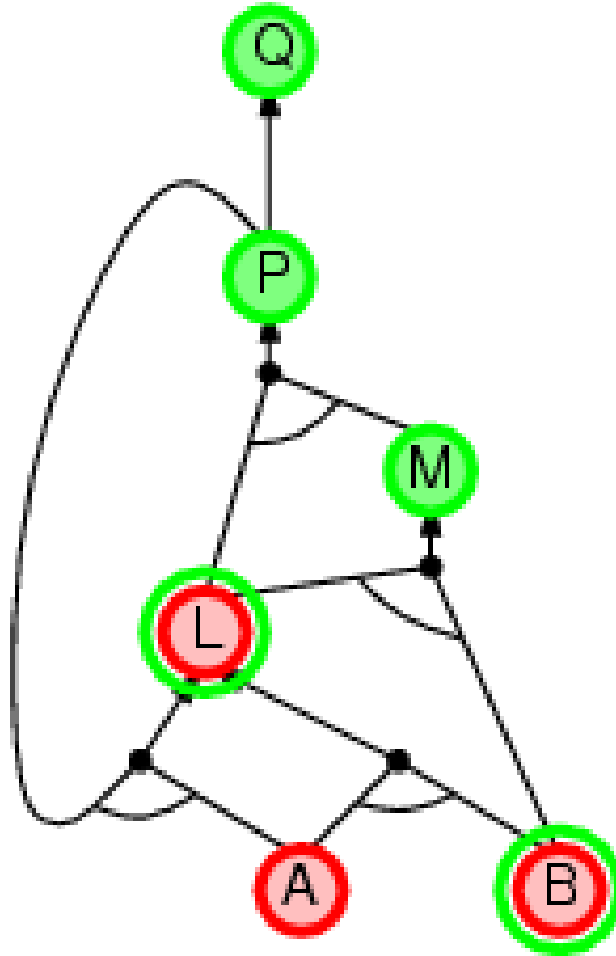
# Пример за верижење наназад



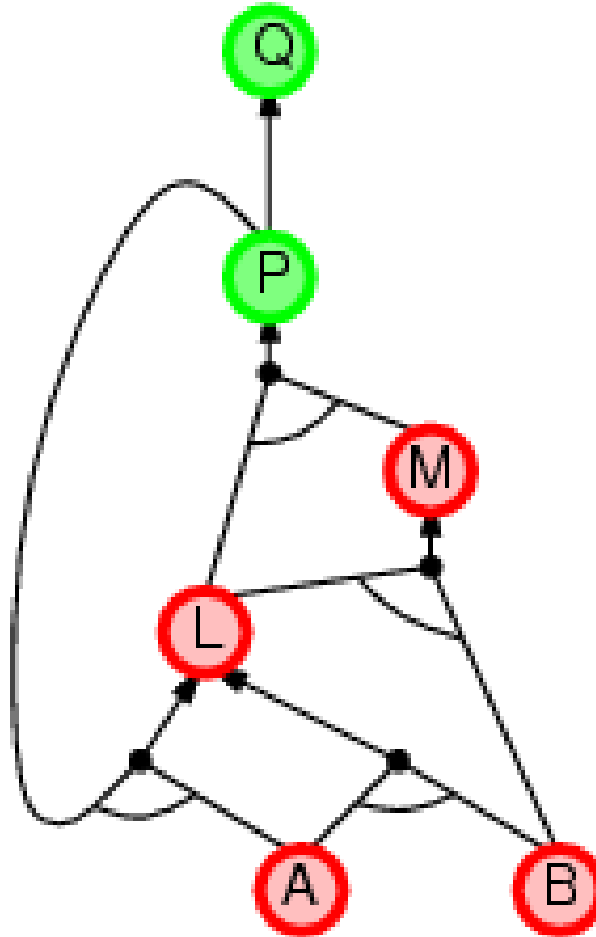
# Пример за верижење наназад



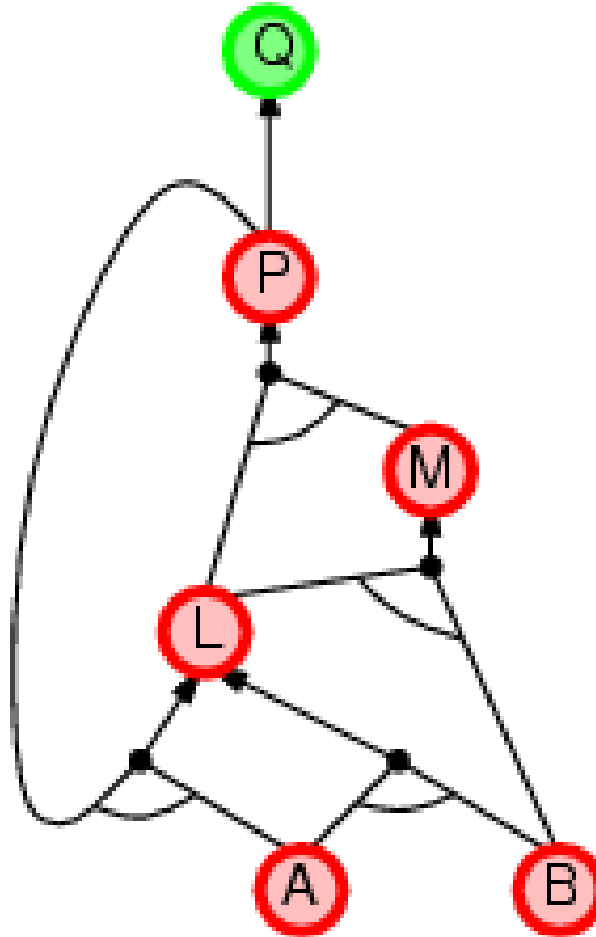
# Пример за верижење наназад



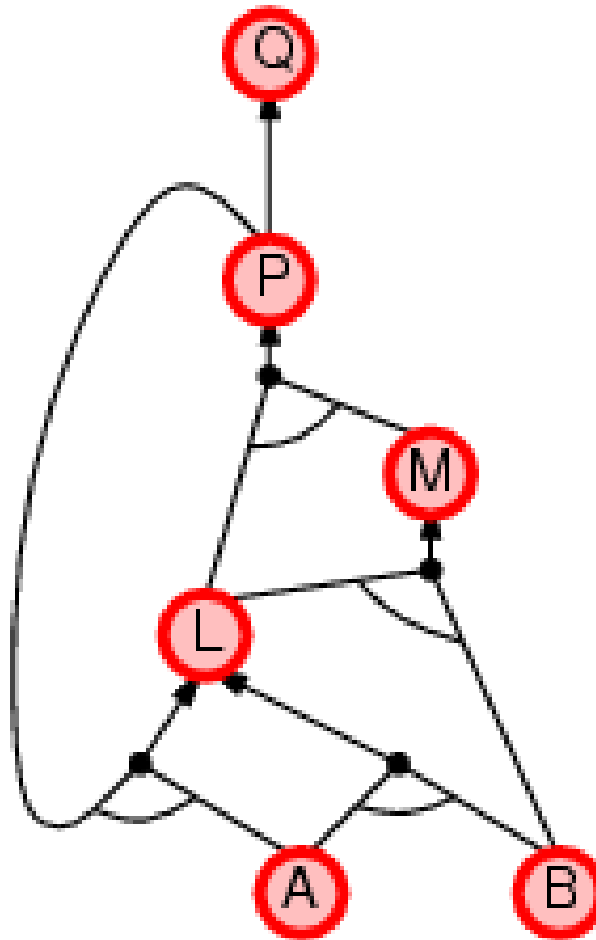
# Пример за верижење наназад



# Пример за верижење наназад



# Пример за верижење наназад





# Резолуција

- Кон множеството изрази да се додаде спротивниот факт од фактот што треба да се докаже
- Цел: да се елиминираат исказите сè додека не се дојде до противречност
- Ако се дојде до противречност значи дека спротивниот факт не може да биде точен, односно значи дека самиот факт е точен
- Се постигнува полесно ако сите искази се преведат во КНФ (конјунктивна нормална форма)



# Што е КНФ (CNF)?

- КНФ = конјунктивна нормална форма  
(CNF: conjunctive normal form)
- CNF е конјункција на литерали:
  - $(A \vee \neg B) \wedge (B \vee \neg C \vee \neg D)$
- Потсетување:
  - Дисјунктивна нормална форма  
(DNF: disjunctive normal form)
  - DNF е дисјункција на литерали:
    - $(A \wedge \neg B) \vee (B \wedge \neg C \wedge \neg D)$

# Како се доаѓа до КНФ (CNF)?

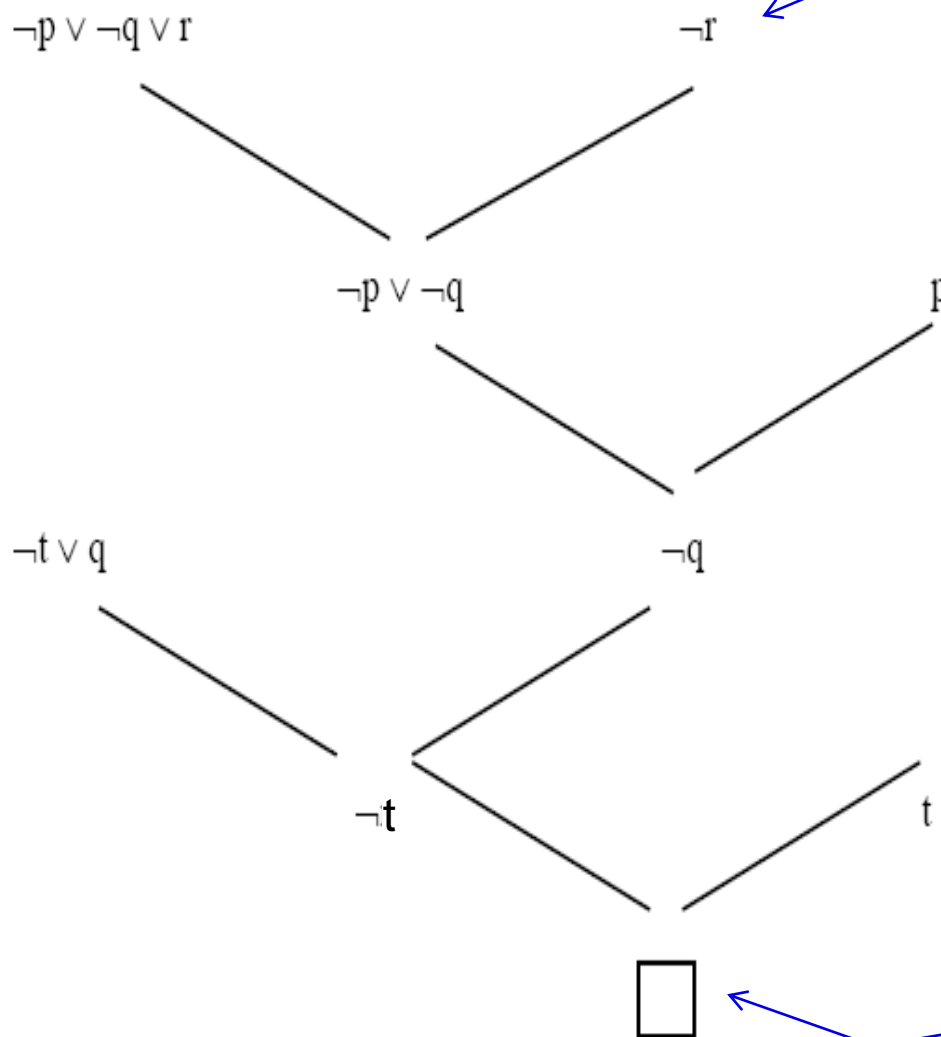
1. Елиминирај ја еквиваленцијата  $p \Leftrightarrow q$  ,  
заменувајќи ја со конјункција од  
импликации  $(p \Rightarrow q) \wedge (q \Rightarrow p)$ .
2. Елиминирај ја импликацијата  $p \Rightarrow q$  ,  
заменувајќи ја со  $\neg p \vee q$ .
3. Извлечи ја негацијата  $\neg$  што е можно  
понанапред користејќи ги законите на  
Де Морган и двојната негација.
4. Примени го дистрибутивниот закон.



# Пример 1: Дали е точно $r$ ?

аксиоми	условни форми
$p$	$p$
$(p \wedge q) \Rightarrow r$	$\neg p \vee \neg q \vee r$
$(s \vee t) \Rightarrow q$	$\neg s \vee q$
$t \Rightarrow q$	$\neg t \vee q$
$t$	$t$

# Одделување на делот од дрвото кое го дава одговорот



Пробуваме да  
додаме негација  
од  $r$  во базата на  
знаење

На крај добиваме  
празен став  
(противречност),  
па заклучуваме  
дека  $r$  е точно

# Резиме за расудувањата

	ОД	И ОД	ЗАКЛУЧИ
Верижење напред	$p$	$(\text{or } (\text{not } p) \ q)$	$q$
Верижење назад	$(\text{not } q)$	$(\text{or } q \ (\text{not } p))$	$(\text{not } p)$
Резолуција	$(\text{not } m)$	$(\text{or } m \ n)$	$n$

операторите се дадени во т.н. prefix нотација која се користи во LISP  
(or m n) значи “m ИЛИ n”



# Прашања?

# Ограничувања на исказното сметање

- Атомите немаат внатрешна структура.
- Не може да се употребуваат променливи
- Со тоа, обемот на светот е ограничен
- Не е во состојба да ги воспостави општите релации меѓу објектите
- Не може да ги прикаже односите кои се менуваат во просторот и времето

# Предикатно сметање

- Компромисно решение за дел од недостатоците на исказното сметање
- Логика од прв ред (first-order logic или first-order predicate calculus)
- Се однесува на објекти (objects) кои исполнуваат одредени услови (propositions)
- Користи симболи кои претставуваат компоненти, терми и дсф-и.

# Предикатното сметање е јазик за претставување на знаењето

## ■ Синтакса:

- ☐ Компоненти
- ☐ Терми
- ☐ Добро создадени формули (дсф)

## ■ Семантика:

- ☐ Светови
- ☐ Интерпретации
- ☐ Модели

## ■ Квантификатори

# Компоненти

- Множество **објекти**: Петре, Птици, Луѓе, Равенки
- Множество **функции** со различна “кратност” или “арност”\*: сопственик $\text{Na}^2$ , растојание $\text{Меѓу}^2$
- Множество **релации** (или предикати): Родител $\text{Na}^3$ , Почеток $\text{Na}^1$ , Еднаков $\text{Na}^2$
- Исказните сврзници  $\wedge$ ,  $\vee$ ,  $\neg$ , и  $\Rightarrow$ , како и заградите и запирката

\* под “арност” се подразбира на колку објекти се однесува функцијата или релацијата. На пример, негацијата е унарна, растојанието и татковството бинарни, родителството тернарно ....

на македонски: еднократна, двократна, трократна, ..., па оттука “кратност”

# Конвенции 1

- Според Russel и Norvig:
  - објектите се именките или подметите
  - релации се глаголите или прироците
  - функциите се релациите кои за дадена влезна вредност добиваат само една вредност

# Конвенции 2

- Според Nilsson, трите компоненти се именуваат како константи. Тој сугерира:
  - имињата на објектните константи да започнуваат со голема буква или со цифра
  - имињата на релациските константи да започнуваат со мала буква, а кратноста да се наведе како степен
  - имињата на функциските константи да започнуваат со голема буква, а кратноста да се наведе како степен

# Терми

- Објектот (објектната константа) е терм
- Терми се логичките изрази кои се однесуваат на некој објект.
- Тоа значи дека еден комплексен терм се добива како  $n$ -кратна ( $n$ -арна) функција во која по името во загради се наведуваат сите терми кои се нејзини аргументи.
- Термот без променливи се вика основен терм (ground term)
- Пр. СинНа(Чарлс, Елизабета),  
ИмаРодители(Хари, Чарлс, Дијана)

# Добро создадени формули (дсф)

- Атоми: релациска константа по која во загради следуваат термите одделени со запирка
- Исказни дсф-и: секој израз добиен на начинот на кој се добиваат дсф-ите во исказното сметање



# Светови

- Светот има бесконечен број објекти, наречени единици (individuals)
- Бројот на функциите над објектите е неограничен
- Бројот на релациите над единиците е неограничен

# Интерпретација

- Интерпретација е начинот на кој или како:
  - објектните константи се пресликуваат во објектите од светот
  - релациските константи (предикатите) се пресликуваат во релациите од светот
  - функциските симболи се пресликуваат во функциите од светот
  
- Пример: Што значи  $\forall x \exists y \text{ Saka}(x,y)$ ?
  1. “Секој сака некого.”
  2. “Секој е сакан од некого.” би било  $\forall y \exists x \text{ Saka}(x,y)$ .



# Квантификатори

- Универзален квантификатор  $\forall$ 
  - Сите птици летаат
  - $\forall x \text{ Ptica}(x) \Rightarrow \text{Leta}(x)$
- Егзистенцијален квантификатор  $\exists$ 
  - Кралицата има круна на главата
  - $\exists x \text{ Kruna}(x) \wedge \text{NaGlava}(x, \text{Kralica})$
- Вгнездени квантификатори
  - Секој сака некого.
  - $\forall x \exists y \text{ Saka}(x, y)$

# Не заборавете:

- Ако постои барем една замена за  $x$  за која  $Q(x)$  има вредност  $T$ , тогаш  $\exists x Q(x)$  има вредност  $T$
- $\forall x P(x) \Leftrightarrow \forall y P(y)$

## Законите на Де Морган:

- $\neg(\forall x P(x)) \Leftrightarrow \exists x \neg P(x)$
- $\neg(\exists x Q(x)) \Leftrightarrow \forall x \neg Q(x)$
- $\forall x P(x) \Leftrightarrow \neg(\exists x \neg P(x))$
- $\exists x Q(x) \Leftrightarrow \neg(\forall x \neg Q(x))$

# Важат и овие својства:

- Редоследот на ист квантификатор не е важен:
- $\forall x \forall y$  е исто што и  $\forall y \forall x$
- $\exists x \exists y$  е исто што и  $\exists y \exists x$
- Ама редоследот на различните квантификатори е важен
- Пример:
- $\exists x \forall y \text{ Saka}(x,y)$     Постои некој што ги сака сите.
- $\forall y \exists x \text{ Saka}(x,y)$     Секој има некој што го сака.

# Кој одговор е точен?

Сите виолинисти се музикални.

- Одговор 1:

$$\forall x \text{ Violinist}(x) \wedge \text{Muzikalni}(x)$$

- Одговор 2:

$$\forall x \text{ Violinist}(x) \Rightarrow \text{Muzikalni}(x)$$

- Одговорот 1 значи:

Сите се виолинисти и се музикални.

# Кој одговор е точен?

Постои математичар кој е музикален.

- Одговор 1:

$$\exists x \text{ Mathematicar}(x) \wedge \text{Muzikalni}(x)$$

- Одговор 2:

$$\exists x \text{ Mathematicar}(x) \Rightarrow \text{Muzikalni}(x)$$

- Одговорот 2 ќе важи за било кој музикален, независно од тоа дали е математичар или не.

# Значи, запомнете:

- По правило:
- **единствениот** сврзник со универзалниот квантификатор  $\forall$  е импликацијата  $\Rightarrow$
- **единствениот** сврзник со егзистенцијалниот квантификатор  $\exists$  е конјункцијата  $\wedge$
  
- Вообичаени грешки се:
- со универзалниот квантификатор  $\forall$  да се користи конјункцијата  $\wedge$
- со егзистенцијалниот квантификатор  $\exists$  да се користи импликацијата  $\Rightarrow$

# Обидете се да ги преведете речениците:

1. Не постои немузикален музичар.
2. Барем еден виолинист е немузикален.
3. Сите пингвини се немузикални.

при што во вашата база не постои предикат за немузикален, туку постои само предикатот музикален.

# Еднаквост на термите

- $term_1 = term_2$  се еднакви во некоја интерпретација на светот ако  $term_1$  и  $term_2$  се однесуваат на ист објект
- На пример:

BratSestra изразено преку Roditel:

$$\forall x, y \text{ BratSestra}(x, y) \Leftrightarrow$$

$$[\neg(x = y) \wedge \exists m, t \neg(m = t) \wedge \text{Roditel}(m, x) \wedge \text{Roditel}(t, x) \wedge \text{Roditel}(m, y) \wedge \text{Roditel}(t, y)]$$

или во превод: секои двајца  $x$  и  $y$ , такви што  $x \neq y$ , имаат барем една заедничка мајка  $m$  и барем еден заеднички татко  $t$  и притоа,  $m \neq t$ .

# СЛИЧНОСТИ

## ■ И Sestri се BratSestra:

$$\forall x,y \text{ Sestri}(x,y) \Leftrightarrow \text{BratSestra}(x,y) \wedge \text{Zensko}(x) \wedge \text{Zensko}(y)$$

Во овој случај, Sestri го повлекува BratSestra!

## ■ Мајката е женскиот родител:

$$\forall m,c \text{ Majka}(c) = m \Leftrightarrow (\text{Zensko}(m) \wedge \text{Roditel}(m,c))$$

## ■ BratSestra е симетрично:

$$\forall x,y \text{ BratSestra}(x,y) \Leftrightarrow \text{BratSestra}(y,x)$$

## ■ Симетрично е и Sestri

$$\forall x,y \text{ Sestri}(x,y) \Leftrightarrow \text{Sestri}(y,x)$$

# Како да се расудува во предикатното сметање?

- Користејќи ги истите расудувања како и во исказното сметање
- Предуслов за резолуцијата:
  - Унификација
  - Сколемизација (метода измислена од Thoralf Skolem)

# Што е целта на унификацијата?

- Унификацијата ги заменува променливите:
  - ☐ со конкретни дозволени вредности или
  - ☐ со други недоделени променливи и ги споредува

# Унификација

- Унификацијата е процес во кој се бара најсоодветната замена со која два логички изрази стануваат еднакви.
- $\text{Unify}(\alpha, \beta) = \theta$  ако  $\alpha\theta = \beta\theta$ , т.е.  
ако две реченици имаат нешто што ги обединува (унифицира, *unify*), тогаш тоа нешто е обединувач (*unifier*).
- Ако унификацијата како резултат даде повеќе обединувачи, тогаш се бара најопштиот MGU  
Most General Unifier

# Алгоритамот за унификација според Расел и Норвиг

```
function UNIFY( $x, y, \theta$ ) returns a substitution to make  $x$  and  $y$  identical
inputs:  $x$ , a variable, constant, list, or compound
          $y$ , a variable, constant, list, or compound
          $\theta$ , the substitution built up so far

if  $\theta = \text{failure}$  then return failure
else if  $x = y$  then return  $\theta$ 
else if VARIABLE?( $x$ ) then return UNIFY-VAR( $x, y, \theta$ )
else if VARIABLE?( $y$ ) then return UNIFY-VAR( $y, x, \theta$ )
else if COMPOUND?( $x$ ) and COMPOUND?( $y$ ) then
    return UNIFY(ARGS[ $x$ ], ARGS[ $y$ ], UNIFY(OP[ $x$ ], OP[ $y$ ],  $\theta$ ))
else if LIST?( $x$ ) and LIST?( $y$ ) then
    return UNIFY(REST[ $x$ ], REST[ $y$ ], UNIFY(FIRST[ $x$ ], FIRST[ $y$ ],  $\theta$ ))
else return failure
```

# Унификација на променливите според Расел и Норвиг

```
function UNIFY-VAR( $var, x, \theta$ ) returns a substitution
  inputs:  $var$ , a variable
           $x$ , any expression
           $\theta$ , the substitution built up so far

  if  $\{var/val\} \in \theta$  then return UNIFY( $val, x, \theta$ )
  else if  $\{x/val\} \in \theta$  then return UNIFY( $var, val, \theta$ )
  else if OCCUR-CHECK?( $var, x$ ) then return failure
  else return add  $\{var/x\}$  to  $\theta$ 
```



# Резиме за унификацијата

- Унификацијата на два изрази се стреми да најде такви замени со кои двата литерала ќе станат еднакви.
- Таа прво тргнува од претпоставката дека множеството  $\theta$  е празно, а потоа се вршат замените се додека не ги исцрпат сите литерали во двата изрази.
- Најопштиот унификатор е оној унификатор кој ги опфаќа сите унификатори на двата изрази.
- Променлива не може да се замени со терм кој ја користи точно таа променлива (на пр.  $x/f(x)$ ).



# Секојдневни примери

1.  $\text{Roditeli}(x, \text{tatko}(x), \text{majka}(\text{Petre}))$
2.  $\text{Roditeli}(\text{Petre}, \text{tatko}(\text{Petre}), y)$
3.  $\text{Roditeli}(\text{Petre}, \text{tatko}(y), z)$
4.  $\text{Roditeli}(\text{Ivana}, \text{tatko}(y), z)$

1. и 2.

$$\Theta = \{x/\text{Petre}, y/\text{majka}(\text{Petre})\}$$

1. и 3.

$$\Theta = \{x/\text{Petre}, y/\text{Petre}, z/\text{majka}(\text{Petre})\}$$

1. и 4.

$$\Theta = \emptyset$$

# Обопштен модус поненс (Generalized Modus Ponens GMP)

Од

$$p_1', p_2', \dots, p_n', (p_1 \wedge p_2 \wedge \dots \wedge p_n \Rightarrow q)$$

при што (за секое  $i$ ) важи дека:  $p_i' \theta = p_i \theta$

следува:  $q \theta$

# CNF за предикатното сметање

1. Елиминирање на импликациите
2. Извлекување на негациите што е можно понапред
3. Стандардизирање на променливите, т.е. преименување на сите едноимени променливи со нови
4. Сколемизација
5. Исфрлување на универзалните квантификатори
6. Дистрибуција на дисјункциите и конјункциите

# Што е целта на сколемизацијата?

- Сколемизацијата ги заменува сите егзистенцијални квантификатори со константи или со универзално квантифицирани функции
- На таков начин, предикатното сметање во спрега со унификацијата се сведува на исказното сметање.

# Како се врши сколемизацијата?

- Секој терм кој е егзистенцијално квантифициран се заменува со именувана функција која зависи од постоечките терми.
- На пример,

$$\forall x [\exists y \text{ Animal}(y) \wedge \neg \text{Loves}(x,y)] \vee [\exists z \text{ Loves}(z,x)]$$

станува

$$\forall x [\text{Animal}(F(x)) \wedge \neg \text{Loves}(x,F(x))] \vee \text{Loves}(G(x),x)$$

# Важна забелешка

- Внимавајте многу на негациите, затоа што тие го менуваат опсегот на квантификаторите !!!

# Расудувањето на дело

## ФАКТИ:

- Сите птици се животни.
- Сите животни јадат.
- Некои птици летаат.
- Некои птици се малечки.
- Канаринците се птици.
- Коки е канаринец.

## ДА СЕ ОДРЕДИ ВИСТИНИТОСТА НА:

- Канаринците летаат.
- Канаринците се животни.
- Коки е птица.
- Коки лета.
- Коки е животно.
- Коки е малечок.

# Превод од македонски во јазикот на логиката од прв ред

1.  $\forall x \text{ Птица}(x) \Rightarrow \text{Животно}(x)$
2.  $\forall x \text{ Животно}(x) \Rightarrow \text{Јаде}(x)$
3.  $\exists x \text{ Птица}(x) \wedge \text{Лета}(x)$
4.  $\exists x \text{ Птица}(x) \wedge \text{Мало}(x)$
5.  $\forall x \text{ Канаринец}(x) \Rightarrow \text{Птица}(x)$
6.  $\text{Канаринец}(\text{Коки})$

# Замена на егзистенцијалните квантификатори со функции

1.  $\forall x \text{ Птица}(x) \Rightarrow \text{Животно}(x)$
2.  $\forall x \text{ Животно}(x) \Rightarrow \text{Јаде}(x)$
3.  $\text{Птица}(\text{Летачи}(x)) \wedge \text{Лета}(\text{Летачи}(x))$
4.  $\text{Птица}(\text{Малечки}(x)) \wedge \text{Мало}(\text{Малечки}(x))$
5.  $\forall x \text{ Канаринец}(x) \Rightarrow \text{Птица}(x)$
6.  $\text{Канаринец}(\text{Коки})$

# Верижење наанапред

## “Коки е животно.”

- Канаринец(Коки)  
од  $\forall x \text{ Канаринец}(x) \Rightarrow \text{Птица}(x)$ ,  $x/\text{Коки}$ ,  
значи резултатот е:  $\text{Птица}(\text{Коки})$
- Птица(Коки)  
од  $\forall x \text{ Птица}(x) \Rightarrow \text{Животно}(x)$ ,  $x/\text{Коки}$
- Животно(Коки)    Т

# Расудувањето во врска со точноста на Коки е животно е пребарување



# Верижење нанапред “Коки лета.”

- Канаринец(Коки)  
од  $\forall x \text{ Канаринец}(x) \Rightarrow \text{Птица}(x), x/\text{Коки}$
- Птица(Коки)
- $\exists x \text{ Птица}(x) \wedge \text{Лета}(x)$  преминува во:  
 $\text{Птица}(\text{Летачи}(x)) \wedge \text{Лета}(\text{Летачи}(x))$   
но, променлива **не може** да се замени со  
терм кој ја користи точно таа променлива,  
значи дека не важи замената  
 $\text{Коки}/\text{Летачи}(\text{Коки})!$
- Заклучок: Постојната база на знаење **не  
може да одлучи** дали Коки лета.

# Верижење наназад “Коки е ЖИВОТНО.”

- G: Животно(Коки)  
од Птица( $x$ )  $\Rightarrow$  Животно( $x$ ),  $x$ /Коки,  
се генерира потцелта:
- G1: Птица(Коки)  
од Канаринец( $x$ )  $\Rightarrow$  Птица( $x$ ),  $x$ /Коки,  
се генерира следната потцел:
- G2: Канаринец(Коки)
- Оваа потцел постои во базата, значи Животно(Коки) е вистина.

# Сколемизација (1)

1.  $\neg \text{Птица}(x_1) \vee \text{Животно}(x_1)$
2.  $\neg \text{Животно}(x_2) \vee \text{Јаде}(x_2)$
3.  $\text{Птица}(\text{Летачи}(x_3)) \wedge \text{Лета}(\text{Летачи}(x_3))$
4.  $\text{Птица}(\text{Малечки}(x_4)) \wedge$   
 $\text{Мало}(\text{Малечки}(x_4))$
5.  $\neg \text{Канаринец}(x_5) \vee \text{Птица}(x_5)$
6.  $\text{Канаринец}(\text{Коки})$

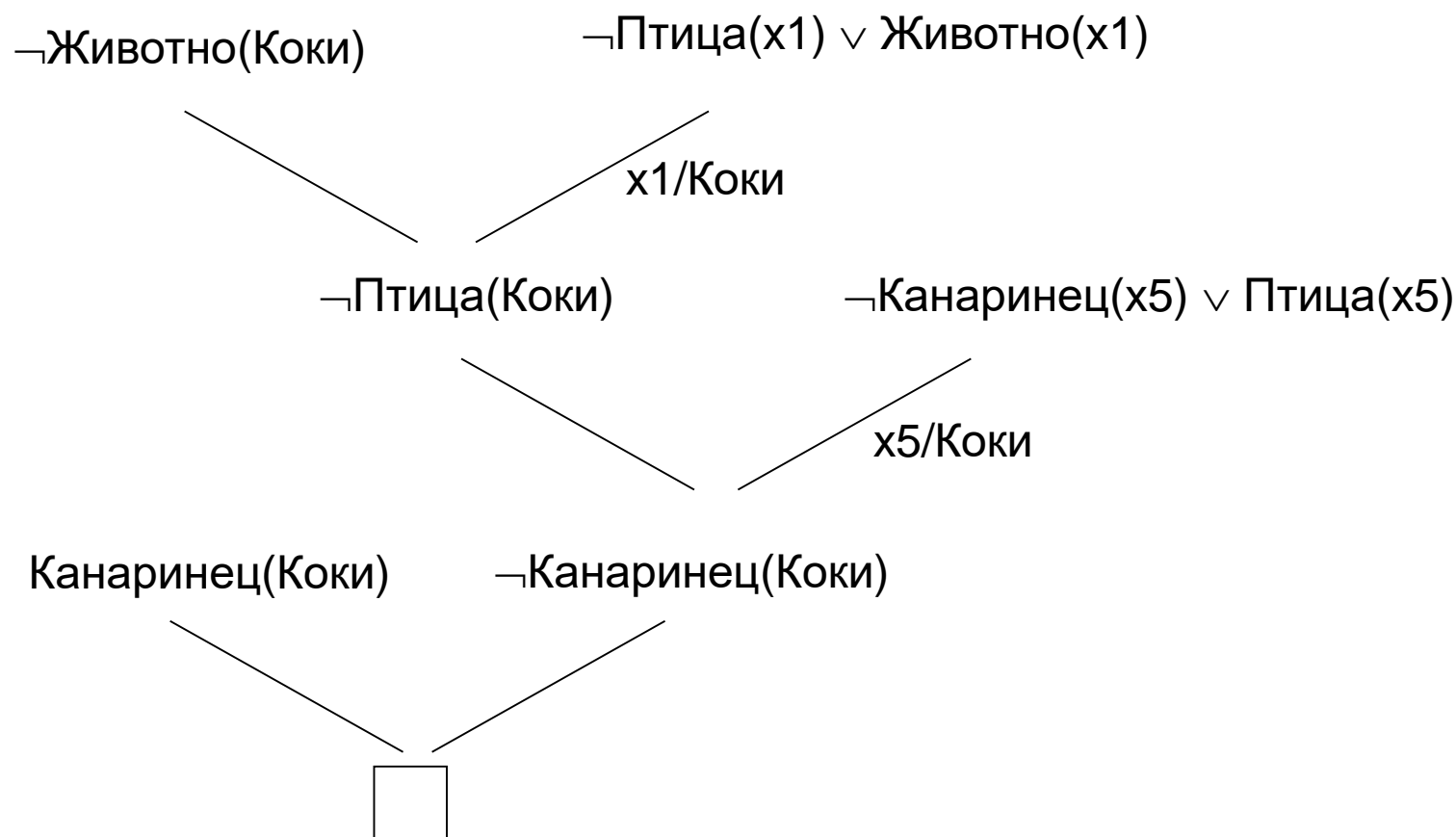


# Сколемизација (2)

1.  $\neg \text{Птица}(x_1) \vee \text{Животно}(x_1)$
2.  $\neg \text{Животно}(x_2) \vee \text{Јаде}(x_2)$
3.  $\text{Птица}(\text{Летачи}(x_3))$
4.  $\text{Лета}(\text{Летачи}(x_3))$
5.  $\text{Птица}(\text{Малечки}(x_4))$
6.  $\text{Мало}(\text{Малечки}(x_4))$
7.  $\neg \text{Канаринец}(x_5) \vee \text{Птица}(x_5)$
8.  $\text{Канаринец}(\text{Коки})$

# Резолуција за “Коки е животно.”

- Претпоставка: Коки не е животно.



# Примена кај едноставен агент базиран врз логика

```
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
```

- Агентот мора да биде способен да:
  - Претставува состојби, акции итн.
  - Вклучува нови перцепти во својата претстава за светот
  - Да ја ажурира внатрешната претстава за светот
  - Донесува заклучоци за некои скриени својства на светот
  - Донесува заклучоци за соодветните акции

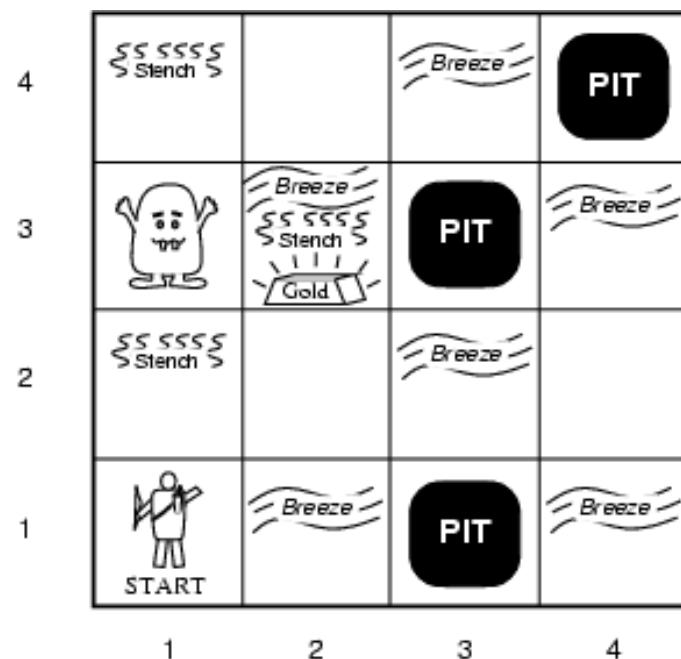
# Опис преку PEAS на светот на апото (Wumpus World)

## Мерки за изведбата (Performance measure)

- ☐ злато+1000, смрт -1000
- ☐ -1 за секој чекор, -10 ако се употреби стрелата

## Околина (Environment)

- ☐ Поле соседно до апото е смрдливо
- ☐ Поле соседно до дупка (pit) е ветровито
- ☐ Златото сјае само во истото поле
- ☐ Со стрела може да се убие апото само ако си свртен кон него (стрелата продолжува сè до сид)
- ☐ Имаш само една стрела
- ☐ Златото може да се земе само ако си во истото поле
- ☐ Златото може да го оставиш само во истото поле



## Актуатори (Actuators):

Сврти лево, Сврти десно, Оди напред, Земи, Пушти, Стрелај

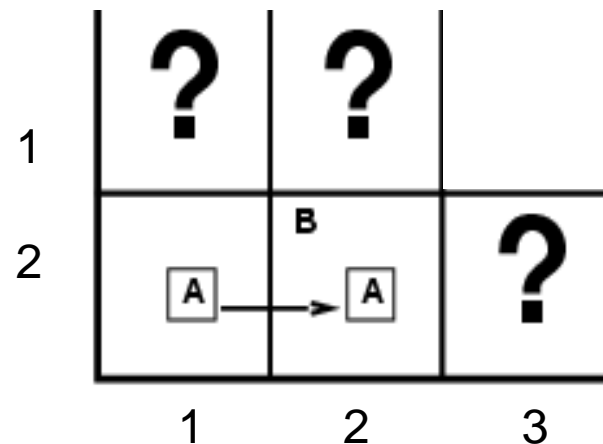
## Сензори (Sensors):

Смрдеа (Stench), Ветре (Breeze), Сјај (Glitter), Судар (Bump), Врискање (Scream)

повеќе на  
стр. 236 – 240  
од книгата

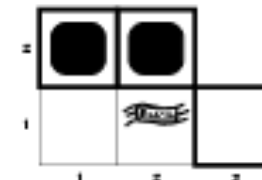
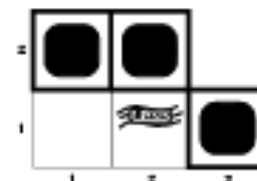
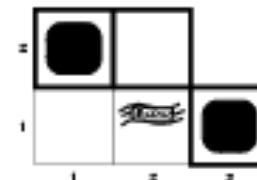
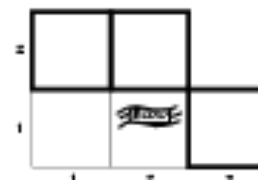
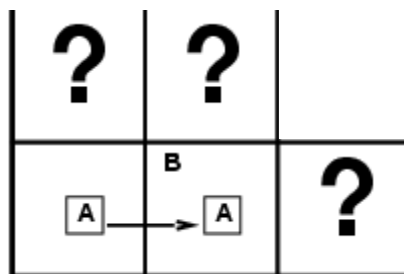
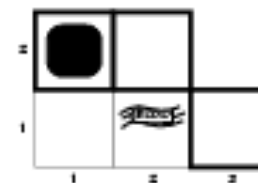
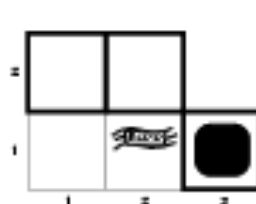
# Повлекување во светот на апото (Entailment in the wumpus world)

- Да разгледаме можни модели на  $KB$  во еден упростен свет на играта каде што постојат само дупки (pits) и други некои ограничувања
- Ова е состојбата по откривањето дека нема ништо во  $[1,1]$ , па се придвижува на десно и потоа открива ветре (breeze) во  $[2,1]$

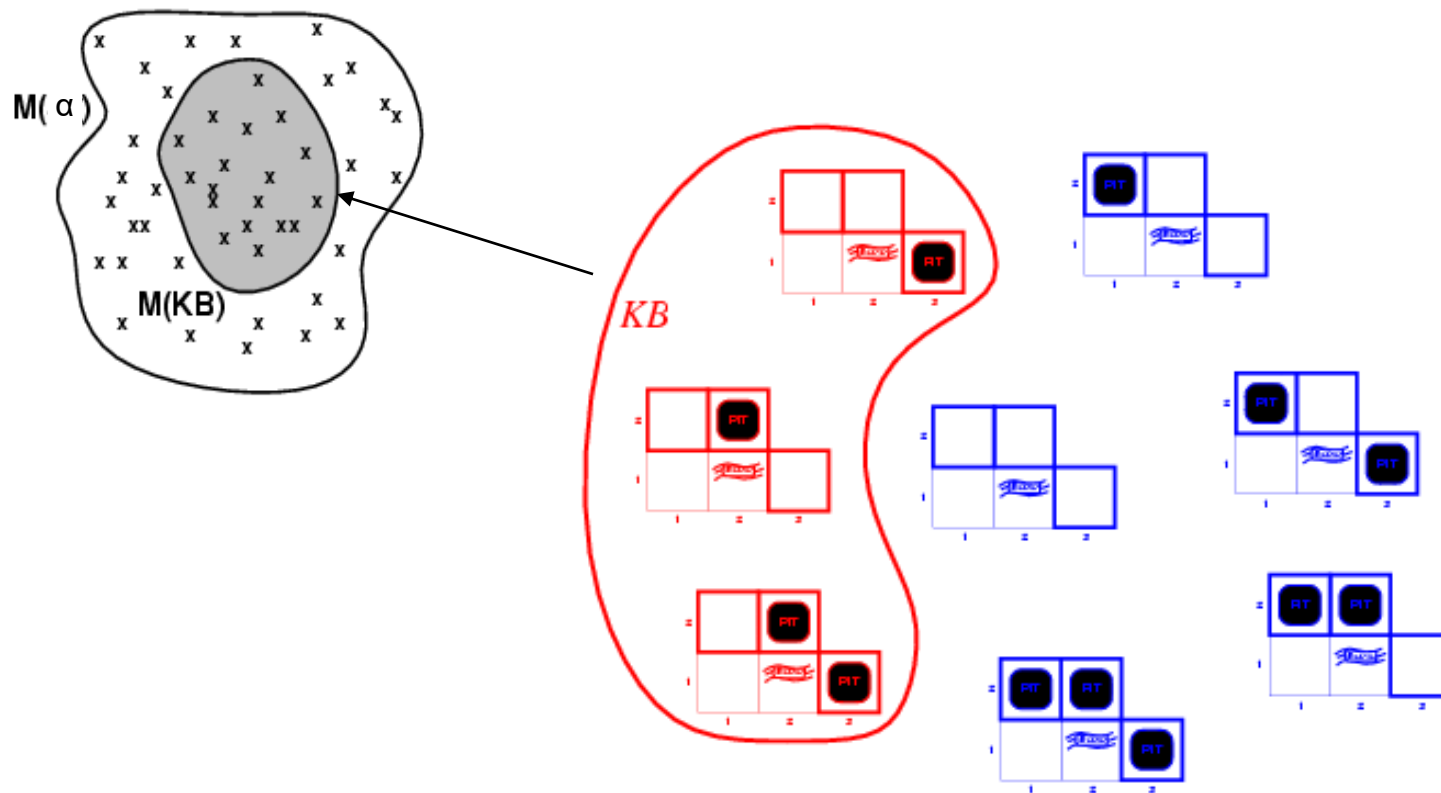


# Модели на апото (Wumpus)

Сите можни модели во овој упростен свет на играта со апото.



# Модели на апото (Wumpus)

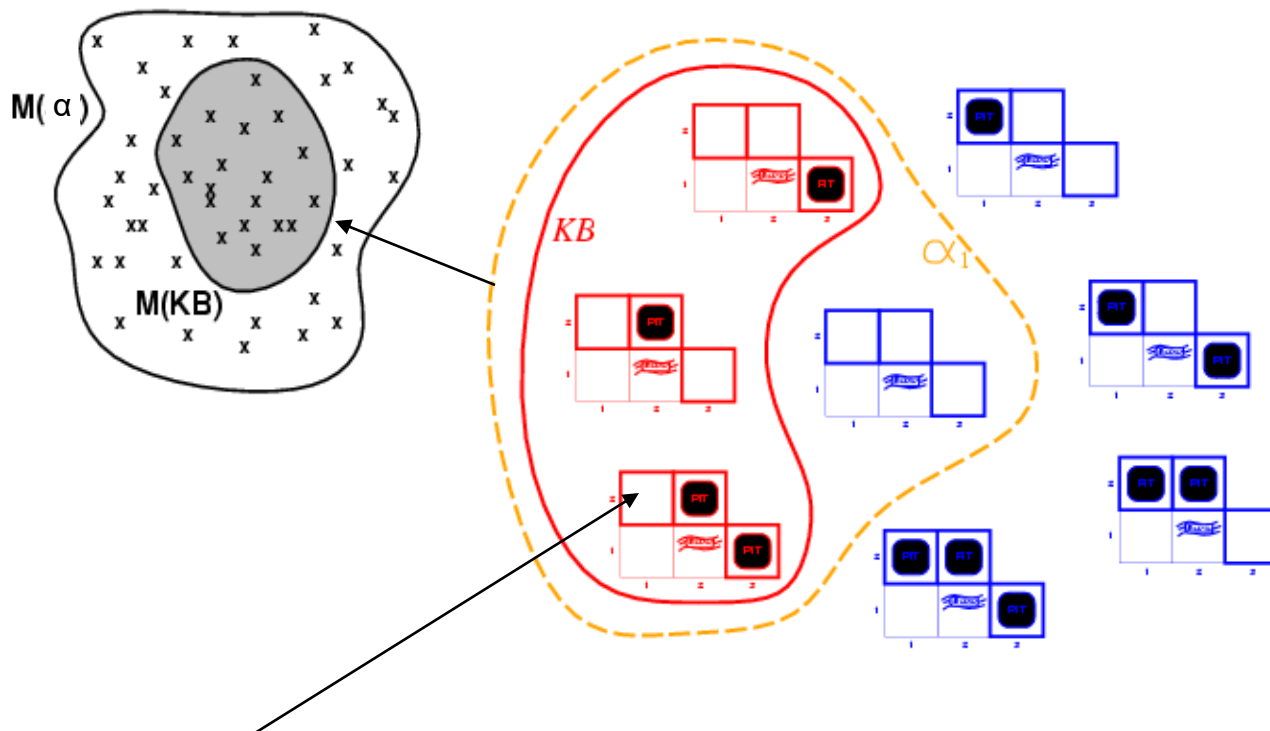


- $KB$  = сите можни светови на апото (wumpus-worlds) кои се доследни (consistent) со набљудувањата и со “физиката” на светот на апото.

# Донесување на заклучоци

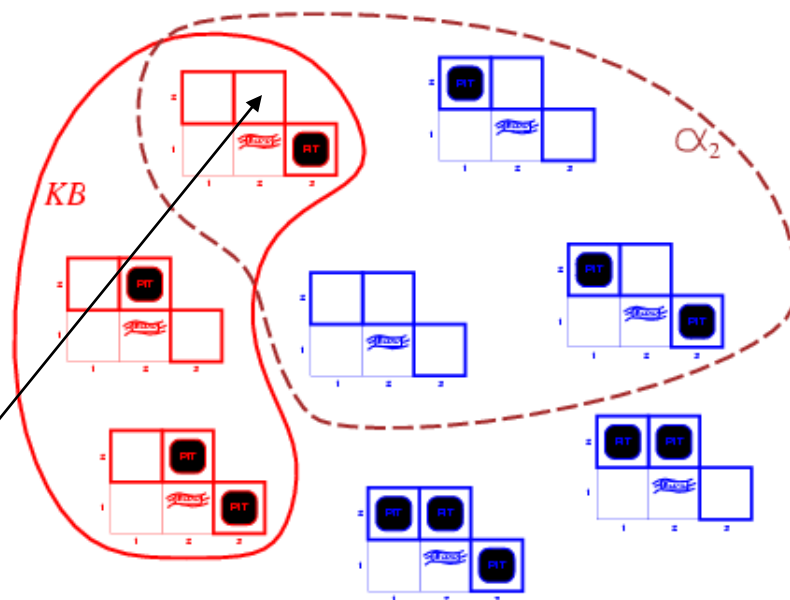
- Да разгледаме 2 можни заклучока за дадена KB
  - $\alpha_1 = "[1,2] \text{ е безбедно}"$
  - $\alpha_2 = "[2,2] \text{ е безбедно}"$
- Една можна постапка за донесување на заклучок
  - Почни од KB
  - Проверка на моделот
    - Провери дали KB  $\models \alpha$  со тоа што ќе провериш дали во сите можни модели каде што KB е точно и  $\alpha$  е исто така точно
- Коментари:
  - Проверката на модели ги набројува сите можни светови
    - Може да работи само на конечни домени, ќе има проблем со експоненцијалниот раст на можни модели

# Модели на апото (Wumpus)



$\alpha_1 = "[1,2] \text{ е безбедно}]", KB \models \alpha_1$ , докажано со проверка на моделот и може да се придружи на KB за во идните проверки

# Модели на апото (Wumpus)



$\alpha_2 = "[2,2] \text{ е безбедно}, KB \neq \alpha_2$

- Постојат некои модели повлечени во KB каде  $\alpha_2$  е неточно.
- Па  $\alpha_2$  не може да се придружи на KB за во следните чекори на донесување заклучоци

# Агенти со носење заклучоци во видео игра со апото (wumpus)

Дефинирање на агент во светот на апото со користење на предикатна логика:

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

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

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

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

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

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

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

...

$$L_{x,y} \wedge \textit{FacingRight} \wedge \textit{Forward} \Rightarrow L_{x+1,y}$$

...

$\Rightarrow$  64 различни симболи за исказите, 155 предикатни искази

# Примена на логичките агенти

- Успешно се применуваат кај експертни системи за:
  - Логичка проверка на софтвери за отстранување на грешки
  - Проектирање на сложени логички кола
  - Дијагностика

# Користена литература

- Artificial Intelligence, A Modern Approach  
3rd edition, Russel and Norvig



# Прашања?