Sistemas Baseados em Conhecimento (Sistemas Especialistas)

Profa. Dra. Sarajane Marques Peres
Escola de Artes, Ciências e Humanidades – Universidade de São Paulo (EACH-USP)
http://each.uspnet.usp.br/sarajane/

Sistemas Baseados em Conhecimento

- *"A Inteligência requer conhecimento"*
- Características do conhecimento humano
  - Volumoso
  - Impreciso
  - Dinâmico
  - Organizado por conteúdo

- Um sistema artificial deve ter:
  - Capacidade de generalização
  - Compreensão pelas pessoas que o fornecem
  - Facilmente modificado
  - Vastamente utilizado (impreciso)

Sistemas baseados em conhecimento

- O que é um sistema baseado em conhecimento?
- Humanos: resolvem problemas aplicando seus conhecimentos a um dado problema

Exemplo de um SBC

- West é criminoso ou não?
  - "A lei americana diz que é proibido vender armas a uma nação hostil. Cuba possui alguns mísseis, e todos eles foram vendidos pelo Capitão West, que é americano"

- Como você resolveria este problema de classificação?
  - **Linguagem**: você entende o que está escrito em português
  - **Conhecimento**: você sabe um pouco de geopolítica e armas
  - **inferência**: você é capaz de raciocinar usando este conhecimento descrito em português

Solucionando o caso do cap. West (linguagem

conhecimento prévio

A) Todo americano que vende uma arma a uma nação hostil é criminoso
B) Todo país em guerra com uma nação X é hostil a X
C) Todo país inimigo político de uma nação X é hostil a X
D) Todo míssil é um arma
E) Toda bomba é um arma
F) Cuba é uma nação
G) USA é uma nação
H) Cuba é inimigo político dos USA
I) Irã é inimigo político dos USA

conhecimento do problema

J) West é americano
K) Existem mísseis em cuba
L) Os mísseis de cuba foram vendidos por West

novo conhecimento

M) Cuba possui um míssel M1 - *de K*
N) M1 [e um míssil - de K
O) M1 é uma arma - *de D e N*
P) Cuba é hostil aos USA - *de F, G, H e C*
Q) M1 foi vendido a Cuba por West - *de L, M e N*
R) West é crimonoso - *de A, J, O, P e Q*

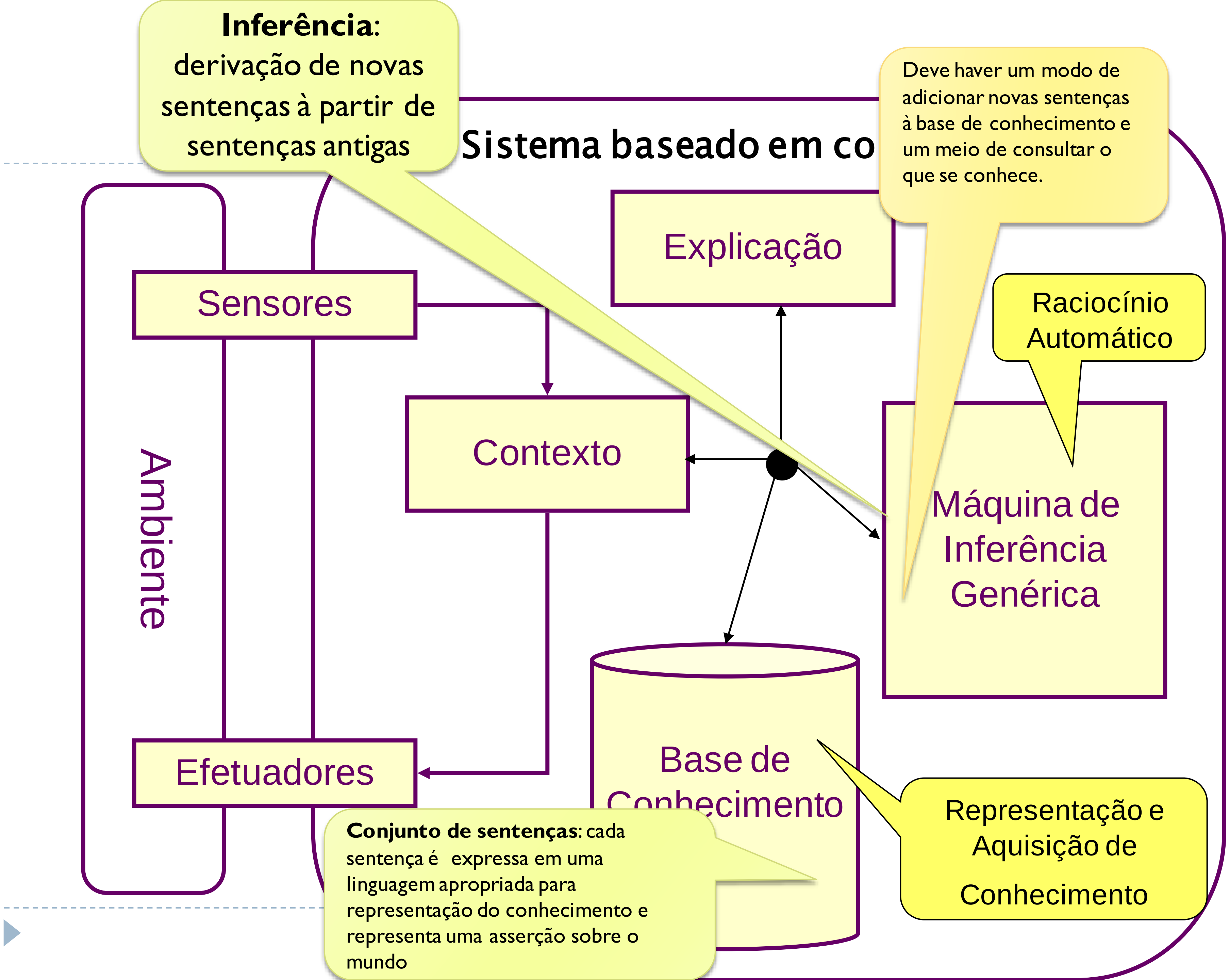
Como uma máquina poderia resolver este problema?

- Segundo a IA...
  - Identificar o **conhecimento** do domínio
  - Representá-lo em uma **linguagem** formal
  - Implementar um mecanismo de **inferência** para utilizá-lo
- The Knowledge Principle (Lenat & Feigenbaum)
  - If a program is to perform a complex task well, it must know a great deal about the world in which it operates
- Questões-chave
  - Como adquirir esse conhecimento?
  - Como representá-lo adequadamente?
  - Como raciocinar com ele correta e eficientemente?

Sistemas baseados em conhecimento

- São sistemas que
  - **raciocinam** sobre suas possíveis ações no mundo

- Conhecem:
  - o estado atual do mundo (propriedades relevantes)
  - como o mundo evolui
  - como identificar estados desejáveis do mundo
  - como avaliar o resultado das ações
  - conhecimento sobre conhecimento (meta-conhecimento)
  - etc.


Inferência: derivação de novas sentenças à partir de sentenças antigas
Sistema baseado em co
Deve haver um modo de adicionar novas sentenças à base de conhecimento e um meio de consultar o que se conhece.
Explicação
Sensores
Raciocínio Automático
Contexto
Ambiente
Máquina de Inferência Genérica
Base de Conhecimento
Efetuadores
Representação e Aquisição de Conhecimento
Conjunto de sentenças: cada sentença é expressa em uma linguagem apropriada para representação do conhecimento e representa uma asserção sobre o mundo

Sistemas baseados em conhecimento

- Dois componentes principais (separados):
  - Base de Conhecimento
  - Mecanismo de Inferência

- Base de Conhecimento:
  - contém o conhecimento do domínio do problema
    - representações de ações e acontecimentos do mundo
    - Cada representação: sentença
    - Sentenças: linguagens específicas
    - Formalismos de representação

Sistema baseado em conhecimento

- Mecanismo (máquina) de Inferência associado:
  - O processador de um SBC
    - responsável por *inferir*, a partir do conhecimento da base, novos fatos ou hipóteses intermediárias/temporárias
  - Progressivo X retroativo
  - Fluxo de busca e fluxo de posição
  - Processamento do Módulo Inferência: busca
    - Bases de conhecimento grandes: heurísticas

- Contexto

- Explicação
  - responsável pela explicação das conclusões apresentadas

Sistema baseado em conhecimento

- Principais diferenças de um SBC e os convencionais
  - Organização dos dados
  - SBCs: métodos que fazem busca em um espaço de possíveis soluções e fazem uso intensivo de heurísticas para tornar a busca efetiva
    - SCs: Algoritmos determinísticos para realizar suas funções
  - Separação do conhecimento e método de solução
    - Maior capacidade de explicação

Sistema (agente) baseado em conhecimento

- O programa se adapta a uma descrição no “nível de conhecimento” onde é especificado o que ele sabe e quais são suas metas para determinar o seu comportamento.

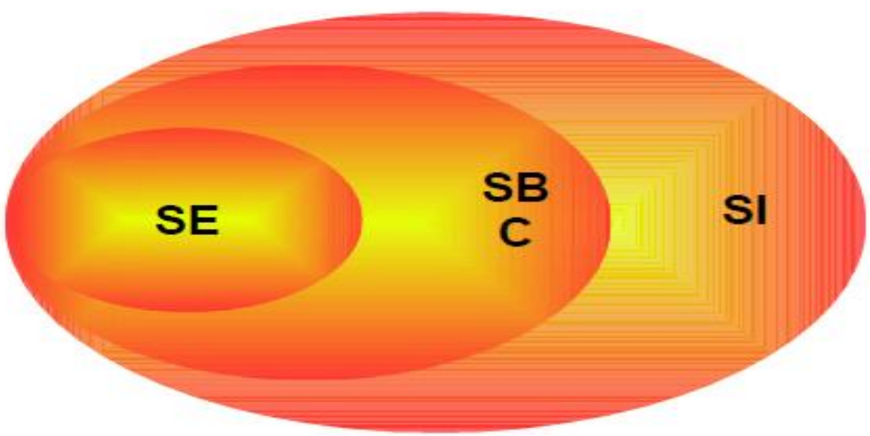
- Abordagem declarativa:
  - É possível construir um agente baseado em conhecimento informando o que ele precisa conhecer e especificando mecanismos que permitam que ele aprenda.

SI X SBC X SE

- Sistemas Inteligentes: exibem conhecimento inteligente

- Sistemas Baseados em Conhecimento: tornam explicito o conhecimento, além de separá-lo do sistema

- Sistemas Especialistas: aplicam conhecimento especializado na resolução de problemas difíceis do mundo real

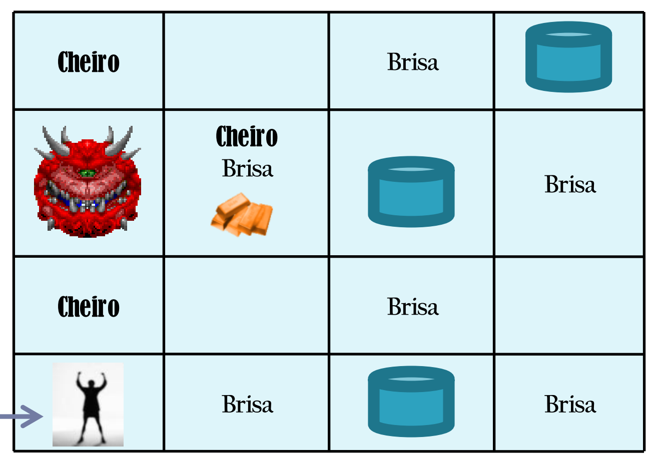


Mundo do Wumpus

Especificação para o Mundo do WUMPUS

- O mundo do WUMPUS:
  - é uma caverna que consiste em salas conectadas por passagens;
  - à espreita em algum lugar está o WUMPUS, um monstro que devora qualquer guerreiro que entrar em sua sala;
  - o WUMPUS pode ser atingido por um agente, mas o agente só tem uma flecha;
  - algumas salas contêm poços sem fundo nos quais cairá qualquer um que vagar por ela (com exceção do WUMPUS, que é muito grande para cair em um poço);
  - neste mundo é possível encontrar um monte de ouro;

Queremos construir
(implementar)
o jogador
(o agente inteligente) !!!!!



Representação típica do mundo de WUMPUS.

PEAS – Definição do ambiente

- PEAS - Performance, Environment, Actuators e Sensors.
  - Medida de desempenho:
    - +1.000: por pegar o ouro
    - −1.000: se cair em um poço ou for devorado pelo Wumpus
    - −1: para cada ação executada
    - −10: pelo uso da flecha
  - Ambiente:
    - Uma malha de 4X4.
    - O agente inicia em [1,1], voltado para a direita.
    - As posições do ouro ou do WUMPUS são escolhidas ao acaso.
    - Probabilidade de um quadrado ter um poço é de 0,2.
  - Atuadores:
    - O agente pode mover-se para frente, virar à esquerda ou direita, morre se cair no poço ou for pego pelo WUMPUS.
    - Mover-se para frente não terá efeito se houver uma parede.
    - A ação AGARRAR pode ser usada para levantar um objeto que está no mesmo quadrado em que se encontra o agente.
    - A ação ATIRAR pode ser usada para disparar uma flecha em linha reta diante do agente. A flecha continua até achar o WUMPUS ou uma parede. O agente só tem uma flecha.

PEAS – Definição do ambiente

- Sensores (5):
  - No quadrado contendo o WUMPUS ou nos quadrados diretamente adjacentes (não na diagonal) o agente perceberá um cheiro ruim.
  - Nos quadrados diretamente adjacentes a um poço, o agente perceberá uma brisa.
  - No quadrado onde está o ouro, o agente perceberá um resplendor.
  - Quando caminhar para uma parede, perceberá um impacto.
  - Quando o WUMPUS é morto, ele emite um **grito triste** que pode ser percebido em qualquer lugar na caverna.

Exemplo de percepção do agente

[Cheiro, Brisa, Nada, Nada, Nada]

O agente explora o ambiente

- A base de conhecimento inicial do agente contém as regras do ambiente; em particular ele sabe que está em [1,1] e que [1,1] é um quadrado seguro.

- A primeira percepção é [Nada, Nada, Nada, Nada, Nada], a partir da qual o agente conclui que seus quadrados vizinhos são seguros.

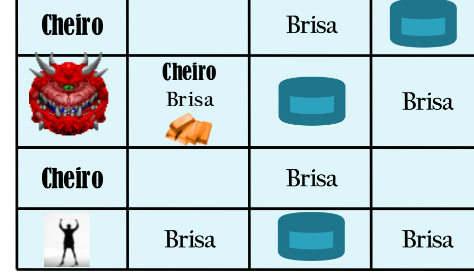
- A partir do fato de que não há nenhum cheiro ou brisa em [1,1], o agente pode deduzir que [1,2] e [2,1] estão livres dos perigos.

- Suponha que o agente decida se mover para frente até [2,1].

- O agente detecta uma brisa em [2,1], e assim deve haver um poço em um quadrado vizinho. O poço não pode estar em [1,1], de acordo com regras do jogo e, portanto, deve haver um poço em [2,2], [3,1] ou ambos.

- Neste momento, existe apenas um quadrado conhecido para onde o agente pode se mover com segurança e que ainda não foi visitado. Deste modo, o agente se voltará, retornará a [1,1] e depois prosseguirá para [1,2].

O agente explora o ambiente



- A nova percepção em [1,2] é: [Cheiro, Nada, Nada, Nada, Nada].

- O cheiro em [1,2] significa que deve haver um WUMPUS por perto. No entanto, ele não pode estar em [1,1], pelas regras do jogo, e não pode estar em [2,2] (ou o agente teria detectado um cheiro quando estava em [2,1])

- O agente então deduz que o WUMPUS está em [1,3].

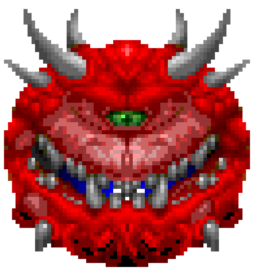
- Além disso, a falta de uma brisa em [1,2] implica que não existe poço em [2,2].

- Esta inferência é bastante difícil, porque combina o conhecimento obtido em diferentes instantes, em diferentes lugares e se baseia na falta de uma percepção para dar um passo crucial.

O agente explora o ambiente

- Em cada caso no qual o agente tira uma conclusão a partir das informações disponíveis, essa conclusão tem a garantia de ser correta se as informações disponíveis estiverem corretas.

- **Essa é uma propriedade fundamental do raciocínio lógico.**



Algumas palavras sobre Lógica

Lógica

- Sentenças são expressas de acordo com a SINTAXE da linguagem de representação. A sintaxe especifica todas as sentenças que são bem-formadas.

$x + y = 4$ → Bem formada

$x\ 4\ y + =$ → Mal formada

- O raciocínio envolve a geração e a manipulação dessas configurações.

- A SEMÂNTICA da linguagem define a verdade de cada sentença em relação a cada mundo possível.

$x + y = 4$ é VERDADE em um mundo no qual
$x = 2$ e $y = 2$,
mas é falsa no mundo onde
$x = 1$ e $y = 1$.

Raciocínio Lógico

- Envolve a relação de conseqüência lógica entre sentenças – a idéia de que uma sentença decorre logicamente de outra sentença.

- Em notação matemática, a relação é expressa como na forma:

$$\alpha \Vdash \beta$$

indicando que a sentença α tem como conseqüência lógica a sentença β.

- $\alpha \Vdash \beta$ se e somente se, em todo modelo (situação, contexto, mundo) no qual α é VERDADEIRO, β também é.

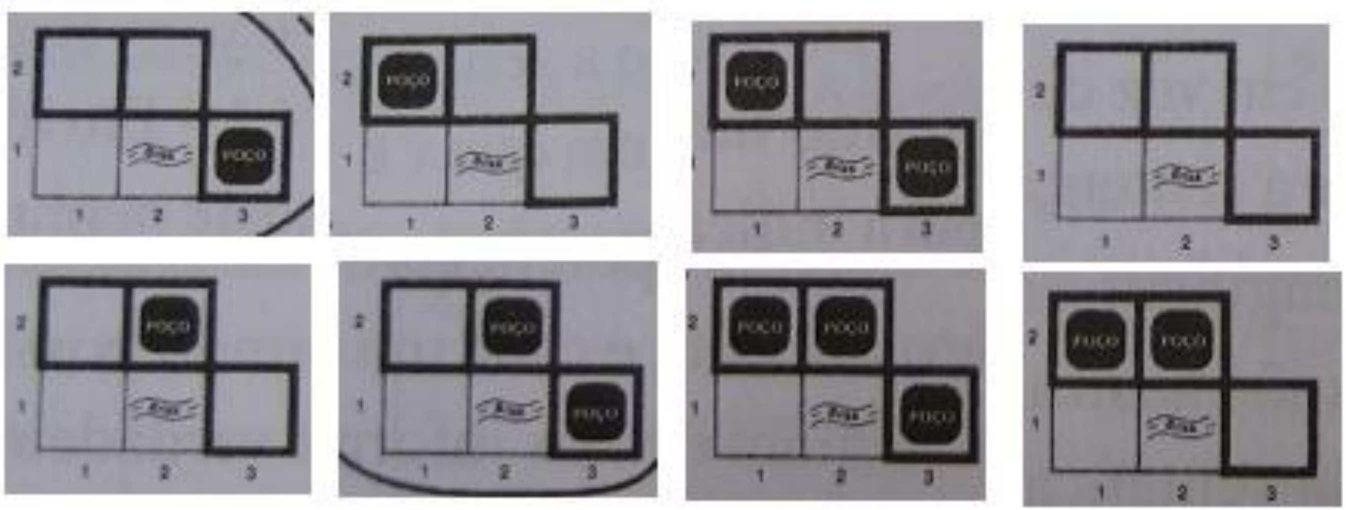
- Também podemos dizer: se α então β.

Contextualizando

- Considere a situação no mundo do WUMPUS: o agente detectou nada em [1,1] e uma brisa em [2,1]. Essas percepções, combinadas com o conhecimento que o agente tem das regras do mundo de WUMPUS constituem a **Base de Conhecimento (BC).**

- O agente está interessado (entre outras coisas) em saber se os quadrados adjacentes [1,2], [2,2] e [3,1] contêm poços.

- Cada um dos três quadrados pode ou não conter um poço e assim existem $2^3 = 8$ modelos possíveis.



Quadrados escuros: não temos informação

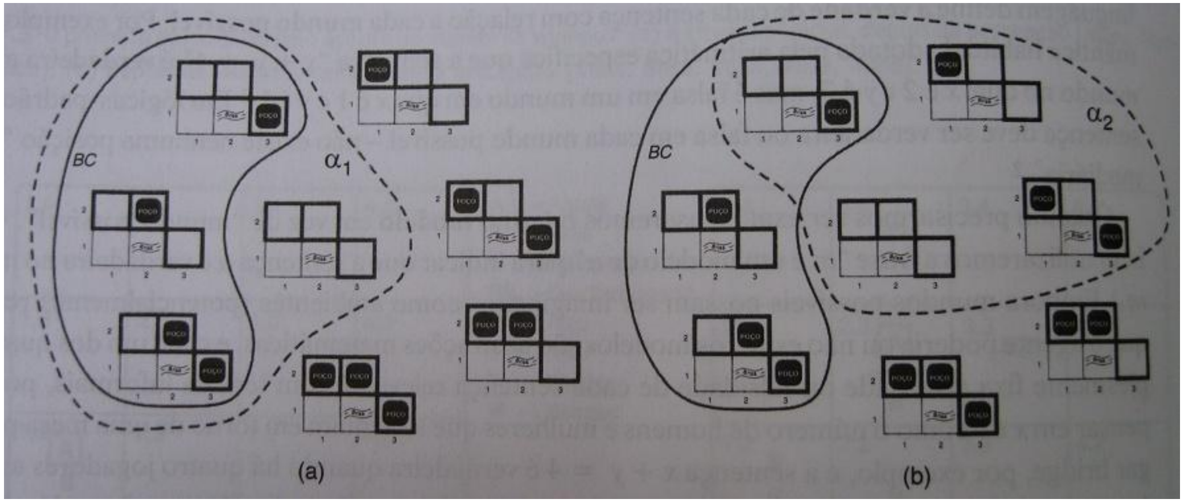
Quadrados claros: temos informação

- A BC é falsa em modelos que contradizem o que o agente sabe.

Contextualizando

- Considerando duas conclusões possíveis:

α1 = não existe nenhum poço em [1,2]
α2 = não existe nenhum poço em [2,2]



- Por inspeção, vê-se que:

Em todo modelo no qual BC é verdadeira, α1 também é verdadeira.

Conseqüentemente, **BC ⊩ α1** :não existe nenhum poço em [1,2].

Em alguns modelos nos quais BC é verdadeira, α2 é falsa.
O agente não pode concluir nada sobre haver poço em [2,2]

Inferência Lógica

- **A definição de conseqüência lógica pode ser aplicada para derivar conclusões – isto é, para conduzir a inferência lógica.**

- O algoritmo de inferência lógica ilustrado pelo exemplo dos slides anteriores é chamado "**verificação de modelo**".
  - Ele enumera todos os modelos possíveis para verificar que α é verdadeira em todos os modelos em que BC é verdadeira.

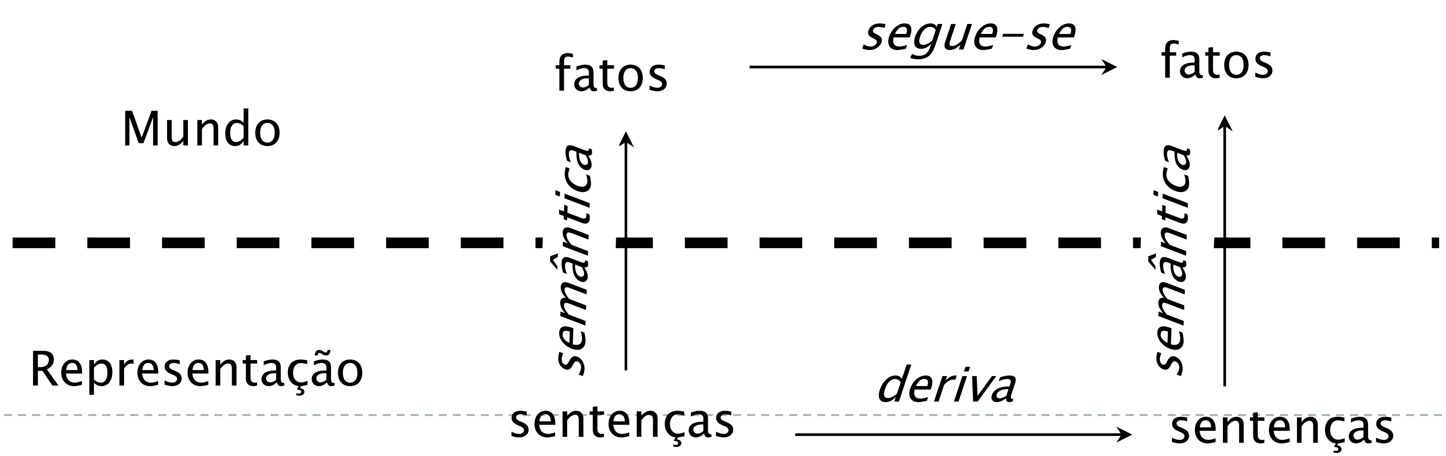
- Se um algoritmo de inferência *i* pode derivar α de BC, escrevemos:

$$BC \Vdash_i \alpha$$

que lemos como "α deriva de BC por *i*" ou "*i* deriva α de BC".

Inferência Lógica

- Um algoritmo de inferência que deriva apenas sentenças permitidas é **consistente,** ou seja, ele preserva a verdade.
  - Um procedimento de inferência não consistente inventa coisas à medida que prossegue.
- Um algoritmo de inferência será **completo** (apresenta **completude**) se puder derivar qualquer sentença permitida.
- Se BC é verdadeira no mundo real, qualquer sentença α derivada de BC por um procedimento de inferência consistente também será verdadeira no mundo real.



Um pouco sobre Lógica Proposicional (ou lógica booleana)

Sintaxe

- A sintaxe da lógica proposicional define as sentenças permitidas.
- As **sentenças atômicas** – os elementos sintáticos indivisíveis – consistem em um único **símbolo proposicional**.
- Cada símbolo proposicional representa uma proposição que pode ser VERDADEIRA ou FALSA.
- Existem dois símbolos proposicionais com significados fixos:
  - VERDADEIRO é a proposição sempre verdadeira
  - FALSO é a proposição sempre falsa.

Sintaxe

- As **sentenças complexas** são construídas a partir de sentenças mais simples com a utilização de **conectivos lógicos**.
  - ¬(não)
  - ∧ (e)
  - v (ou)
  - ⇒ (implica)
  - ⇔ (se e somente se)
- A ordem de precedência em lógica proposicional é: ¬ , ∧ , v , ⇒ e ⇔.

Sintaxe

- A precedência não resolve ambiguidades em sentenças como $A \wedge B \wedge C$, que poderiam ser lidas como $((A \wedge B) \wedge C)$ ou como $(A \wedge (B \wedge C))$.
- Tendo em vista que essas duas leituras têm o mesmo significado de acordo com a semântica, sentenças como essas são permitidas.
- Outro exemplo permitido: $A \vee B \vee C$.
- Sentenças como $A \Rightarrow B \Rightarrow C$ não são permitidas, porque as duas leituras tem significados diferentes; nesse caso, se deve fazer uso de parênteses.

Semântica

- A semântica define as regras para determinar a verdade de uma sentença com respeito a um modelo específico.

- Por exemplo: se as sentenças na base de conhecimento fizerem uso dos símbolos proposicionais $P_{1,2}$, $P_{2,2}$, $P_{3,1}$, então um modelo possível será (onde P = Poço):

$$m1 = \{P_{1,2} = \text{falsa}, P_{2,2} = \text{falsa}, P_{3,1} = \text{verdadeira}\}$$

Mundo do Wumpus

- A semântica da lógica proposicional deve especificar como calcular o valor verdade de qualquer sentença, dado um modelo (de maneira recursiva). Todas as sentenças são construídas a partir de sentenças atômicas e dos cinco conectivos.

- Assim precisamos especificar como calcular a verdade de sentenças atômicas e como calcular a verdade de sentenças formadas com cada um dos cinco conectivos

$m1 = \{P_{1,2} = \text{falsa}, P_{2,2} = \text{falsa}, P_{3,1} = \text{verdadeira}\}$

Semântica

- Para sentenças atômicas:
  - *Verdadeiro* é verdadeiro em todo modelo e *Falso* é falso em todo modelo.
  - O valor-verdade de todos os outros símbolos proposicionais deve ser especificado diretamente no modelo. Por exemplo, no modelo *m*1 $P_{1,2}$ é falsa.
- Para sentenças complexas, tem-se regras como:
  - Para qualquer sentença *s* e qualquer modelo *m*, a sentença $\neg s$ é verdadeira em *m* se e somente se *s* é falso em *m*.
- As regras para cada conectivo podem ser resumidas em uma tabela-verdade que especifica o valor verdade de uma sentença complexa para cada atribuição possível de valores-verdade a seus componentes.

$m1 = \{P_{1,2} = \text{falsa}, P_{2,2} = \text{falsa}, P_{3,1} = \text{verdadeira}\}$

Semântica

Tabela-verdade para os conectivos lógicos

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
falso	falso	verdadeiro	falso	falso	verdadeiro	verdadeiro
falso	verdadeiro	verdadeiro	falso	verdadeiro	verdadeiro	falso
verdadeiro	falso	falso	falso	verdadeiro	falso	falso
verdadeiro	verdadeiro	falso	verdadeiro	verdadeiro	verdadeiro	verdadeiro

A sentença $\neg P_{1,2} \wedge (P_{2,2} \vee P_{3,1})$, avaliada em m1, resulta em:

verdadeiro ^ (falso v verdadeiro) = verdadeiro ^ verdadeiro = verdadeiro

Uma base de conhecimento consiste em um conjunto de sentenças. Uma base de conhecimento lógica é uma conjunção dessas sentenças.

P = Parede
Q = Vira

Se Parede → vira!
Somente se Parede → vira!

Contextualizando

- As regras do mundo de WUMPUS são mais bem escritas usando-se $\Leftrightarrow$.
- Por exemplo:
  - Bi-condicional: um quadrado tem uma brisa SOMENTE SE um quadrado vizinho tem um poço.

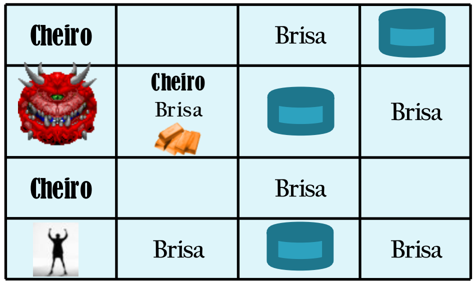
$$\mathbf{Brisa_{1,1} \Leftrightarrow (Poço_{1,2} \vee Poço_{2,1})}$$

- A implicação simples: um quadrado tem uma brisa ENTÃO um quadrado vizinho tem um poço

$$\mathbf{Brisa_{1,1} \Rightarrow (Poço_{1,2} \vee Poço_{2,1})}$$

é verdadeira no mundo de WUMPUS, mas é incompleta. Ela não elimina modelos em que $Brisa_{1,1}$ é falsa e $Poço_{1,2}$ é verdadeira.

Uma base de conhecimento simples



- **Lidando com poços** no mundo de WUMPUS.
- Vocabulário de símbolos proposicionais. Para cada *i, j* :
  - Seja $P_{i,j}$ verdadeira se existe um poço em [*i,j*].
  - Seja $B_{i,j}$ verdadeira se existe uma brisa em [*i,j*].
- A base de conhecimento inclui as sentenças a seguir, cada uma rotulada por conveniência:
  - Não há poço em [1,1]:

$$R1: \neg P_{1,1}.$$

  - Um quadrado tem uma brisa se e somente se existe um poço em um quadrado vizinho. Isso tem de ser declarado para cada quadrado; aqui estão os relevantes para o nosso exemplo.

$$R2: B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1}).$$
$$R3: B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$$

  - As sentenças precedentes são verdadeiras em todos os mundos de WUMPUS.
  - ... Já automatizando!
  - **Incluindo** as percepções de brisa para os dois primeiros quadrados visitados no mundo específico em que o agente se encontra (levando em consideração o nosso exemplo).

$$R4: \neg B_{1,1}$$
$$R5: B_{2,1}$$

Inferência

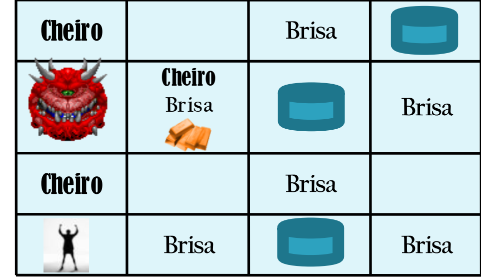
$R1: \neg P_{1,1}.$

$R2: B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1}).$

$R3: B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$

$R4: \neg B_{1,1}$

$R5: B_{2,1}$



- O objetivo da inferência lógica é decidir se $BC \Vdash \alpha$ para alguma sentença α.
- $P_{2,2}$ é permitida (é uma posição onde o agente pode ir com segurança)? e $P_{1,2}$?
- Algoritmo de verificação de modelos: enumere os modelos e verifique se α é verdadeira em todo modelo no qual BC é verdadeira.
  - Na lógica proposicional os modelos são atribuições de verdadeiro ou falso a todo símbolo proposicional.
  - No nosso problema com 7 símbolos, $2^7 = 128$ modelos são possíveis; em três deles BC é verdadeira.
  - $P_{2,2}$ é verdadeira em dois dos três modelos e falsa em um, assim não podemos dizer ainda se existe um poço em [2,2].

$B_{1,1}$	$B_{2,1}$	$P_{1,1}$	$P_{1,2}$	$P_{2,1}$	$P_{2,2}$	$P_{3,1}$	R_1	R_2	R_3	R_4	R_5	BC
falso	falso	falso	falso	falso	falso	falso	verdadeiro	verdadeiro	verdadeiro	verdadeiro	falso	falso
falso	falso	falso	falso	falso	falso	verdadeiro	verdadeiro	verdadeiro	falso	verdadeiro	falso	falso
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
falso	verdadeiro	falso	falso	falso	falso	falso	verdadeiro	verdadeiro	falso	verdadeiro	verdadeiro	falso
falso	verdadeiro	falso	falso	falso	falso	verdadeiro	verdadeiro	verdadeiro	verdadeiro	verdadeiro	verdadeiro	verdadeiro
falso	verdadeiro	falso	falso	falso	verdadeiro	falso	verdadeiro	verdadeiro	verdadeiro	verdadeiro	verdadeiro	verdadeiro
falso	verdadeiro	falso	falso	falso	verdadeiro	verdadeiro	verdadeiro	verdadeiro	verdadeiro	verdadeiro	verdadeiro	verdadeiro
falso	verdadeiro	falso	falso	verdadeiro	falso	falso	verdadeiro	falso	falso	verdadeiro	verdadeiro	falso
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
verdadeiro	verdadeiro	verdadeiro	verdadeiro	verdadeiro	verdadeiro	verdadeiro	falso	verdadeiro	verdadeiro	falso	verdadeiro	falso

Regras de Inferência

- **Regras de inferência**: Padrões de inferência que podem ser aplicados para derivar cadeias de conclusões que levam aos objetivos desejados.

- A mais conhecida: **MODUS PONENS**

$$\frac{\alpha \Rightarrow \beta, \quad \alpha}{\beta}$$

- Sempre que quaisquer sentenças da forma $\alpha \Rightarrow \beta$ e α são dadas, a sentença β pode ser deduzida.

- Se (WumpusAdiante ^ WumpusVivo) ⇒ Atirar e (WumpusAdiante ^ WumpusVivo) são dadas, então **Atirar** pode ser deduzida.

Regras de Inferência

- Modus Ponens: $$\frac{\cancel{\alpha} \Rightarrow \beta, \cancel{\alpha}}{\beta}$$
- E-eliminação: $$\frac{\alpha_1 \wedge \alpha_2 \wedge \ldots \wedge \alpha_n}{\alpha_i}$$
- E-introdução: $$\frac{\alpha_1, \alpha_2, \ldots, \alpha_n}{\alpha_1 \wedge \alpha_2 \wedge \ldots \wedge \alpha_n}$$
- Ou-introdução: $$\frac{\alpha_i}{\alpha_1 \vee \alpha_2 \vee \ldots \vee \alpha_n}$$
- Eliminação de dupla negação: $$\frac{\neg\neg\ \alpha}{\alpha}$$
- Resolução unidade: $$\frac{\alpha \vee \cancel{\beta}, \neg \cancel{\beta}}{\alpha}$$
- Resolução: $$\frac{\alpha \vee \cancel{\beta}, \neg \cancel{\beta} \vee \gamma}{\alpha \vee \gamma} \Leftrightarrow \frac{\neg \alpha \Rightarrow \cancel{\beta}, \cancel{\beta} \Rightarrow \gamma}{\neg \alpha \Rightarrow \gamma}$$

α_i / β diz que a sentença β pode ser deduzida da BC constituída pelos α_i

Regras de Inferência

- Todas as equivalência lógicas abaixo podem ser usadas como regras de inferência.

$(\alpha \wedge \beta)$	$\equiv$	$(\beta \wedge \alpha)$	comutatividade de $\wedge$
$(\alpha \vee \beta)$	$\equiv$	$(\beta \vee \alpha)$	comutatividade de $\vee$
$(\alpha \wedge \beta) \wedge \gamma)$	$\equiv$	$(\alpha \wedge (\beta \wedge \gamma)$	associatividade de $\wedge$
$(\alpha \vee \beta) \vee \gamma)$	$\equiv$	$(\alpha \vee (\beta \vee \gamma)$	associatividade de $\vee$
$\neg(\neg\alpha)$	$\equiv$	α	eliminação de negação dupla
$(\alpha \Rightarrow \beta)$	$\equiv$	$(\alpha \wedge (\neg\beta \Rightarrow \neg\alpha)$	contraposição
$(\alpha \Rightarrow \beta)$	$\equiv$	$(\neg\alpha \vee \beta)$	eliminação de implicação
$(\alpha \Leftrightarrow \beta)$	$\equiv$	$((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$	eliminação de bicondicional
$\neg(\alpha \wedge \beta)$	$\equiv$	$(\neg\alpha \vee \neg\beta)$	de Morgan
$\neg(\alpha \vee \beta)$	$\equiv$	$(\neg\alpha \wedge \neg\beta)$	de Morgan
$(\alpha \wedge (\beta \vee \gamma))$	$\equiv$	$((\alpha \wedge \beta) \vee (\alpha \wedge \gamma))$	distributividade de $\wedge$ sobre $\vee$
$(\alpha \vee (\beta \wedge \gamma))$	$\equiv$	$((\alpha \vee \beta) \wedge (\alpha \vee \gamma))$	distributividade de $\vee$ sobre $\wedge$

Regras de Inferência

- Exemplo:
  - Equivalência para eliminação de bicondicional gera as duas regras de inferência:

$$\frac{\alpha \Leftrightarrow \beta}{(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)}$$

$$\frac{(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)}{\alpha \Leftrightarrow \beta}$$

  - Estudando a aplicação de regras de inferência e equivalência no mundo de WUMPUS

Propriedade da monotonicidade

- A **monotocidade** afirma que o conjunto de sentenças permitidas só pode aumentar à medida que as informações são acrescentadas à base de conhecimento.

- Para quaisquer sentenças α e β, se

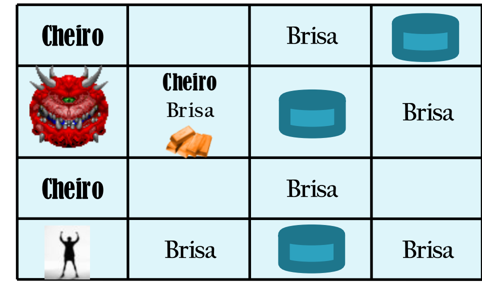
$$BC \Vdash \alpha \text{ então } BC \vee \beta \Vdash \alpha$$

- Suponha que a base de conhecimento contenha a asserção adicional β afirmando que existem exatamente oito poços no mundo. Esse conhecimento poderia ajudar o agente a tirar conclusões adicionais, mas não pode invalidar qualquer conclusão α já deduzida.

- OBS.: As lógicas **não-monotônicas**, que violam a propriedade de monotonicidade, captam uma propriedade comum do raciocínio humano: a **mudança de idéia**.

Exemplo: BC mundo de wumpus

- Seja Pij verdade se existe um poço em [i, j].
- Seja Bij verdade se há brisa em [i, j].
- Agente na posição [1,1]
  - **R1:** $\neg P_{11}$
  - **R2:** $\neg B_{11}$
  - "Poços causam brisas em quadrados adjacentes "
  - **R3:** $B_{11} \Leftrightarrow (P_{12} \vee P_{21})$
- Agente na posição [2,1]
  - **R4:** B_{21}
  - "Poços causam brisas em quadrados adjacentes "
  - **R5:** $B_{21} \Leftrightarrow (P_{11} \vee P_{22} \vee P_{31})$



- Aplicando a eliminação de bicondicional a R2.

$$R3: B_{1,1} \Leftrightarrow (P_{1,2} \lor P_{2,1})$$
$$R6: (B_{1,1} \Rightarrow (P_{1,2} \lor P_{2,1})) \land ((P_{1,2} \lor P_{2,1}) \Rightarrow B_{1,1})$$

- Aplicando a Eliminação-de-E a R6:

$$\frac{\alpha \land \beta}{\alpha} \qquad \frac{\alpha \land \beta}{\beta}$$

$$R7: (P_{1,2} \lor P_{2,1}) \Rightarrow B_{1,1}$$

- A equivalência lógica para contraposição e a Eliminação-de-E fornece:

$$R8: (\neg B_{1,1} \Rightarrow \neg ((P_{1,2} \lor P_{2,1}))$$

- Aplicando Modus Ponens com R8 e com a percepção R4:

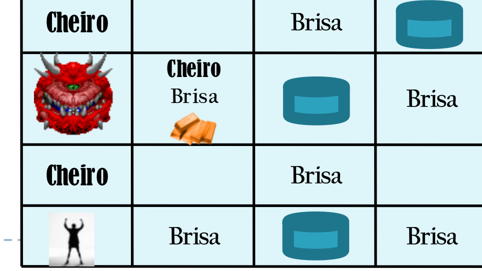
$$R4: \neg B_{1,1}$$
$$R9: \neg (P_{1,2} \lor P_{2,1})$$

- Aplicando De Morgan

$$R10: \neg P_{1,2} \land \neg P_{2,1})$$

Não existe poço em [1,2] e nem em [2,1].

Resolução



- Resolução: gera um algoritmo de inferência completo quando acoplada a qualquer algortimo de busca completo.
  - Algoritmo de busca completo: encontram qualquer meta acessível.
  - Se as regras de inferência forem inadequadas, uma meta pode não ser acessível.
- Regra da resolução no mundo de WUMPUS.
  - Considere:
    - O agente retorna de [2,1] para [1,1], e então vai para [1,2], onde percebe um cheiro, mas nenhuma brisa. Adicionamos tais fatos à base de conhecimento.

$$R11: B_{2,1}$$
$$R12: B_{2,1} \Leftrightarrow (P_{1,1} \lor P_{2,2} \lor P_{3,1})$$

    - Daqui é possível derivar que não existem poços em [2,2] nem em [1,1]. E com algum esforço derivacional chega-se à informação que existe poço em [1,1] ou [2,2] ou [3,1].

$$R13: \neg P_{2,2}$$
$$R14: \neg P_{1,1}$$
$$R15: P_{1,1} \lor P_{2,2} \lor P_{3,1}$$

Faça!!

Resolução

- Agora vem a primeira aplicação da regra de resolução:

o literal $\neg P_{2,2}$ em R13 *se resolve* com o literal $P_{2,2}$ em R15

- E tem-se

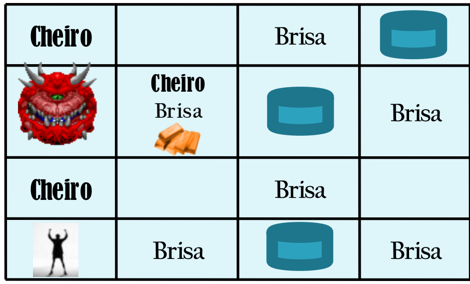
$$R16: P_{1,1} \vee P_{3,1}$$

- De mode semelhante:

o literal $\neg P_{1,1}$ em R1 *se resolve* com o literal $P_{1,1}$ em R16

- E tem-se

$$R17: P_{3,1}$$



Resolução

Satisfatibilidade é ligada à inferência via o seguinte:

$BC \models \alpha$ se e somente se $(BC \wedge \neg\alpha)$ é insatisfatível

Raciocinando por contraposição

Resolução

Forma Normal Conjuntiva -- *Conjunctive Normal Form (CNF)*

conjunção de disjunções de literais

E.g., $(A \lor \neg B) \land (B \lor \neg C \lor \neg D)$

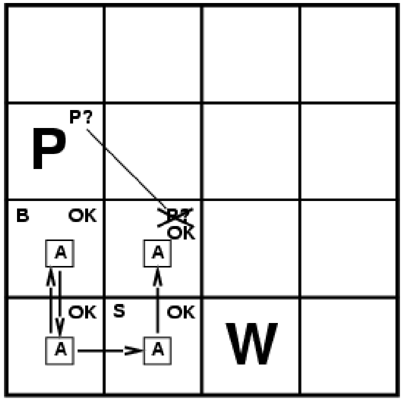
- Regra de inferência resolução (para CNF):

$$\frac{\ell_1 \lor \dots \lor \ell_k, \qquad m_1 \lor \dots \lor m_n}{\ell_1 \lor \dots \lor \ell_{i-1} \lor \ell_{i+1} \lor \dots \lor \ell_k \lor m_1 \lor \dots \lor m_{j-1} \lor m_{j+1} \lor \dots \lor m_n}$$

onde ℓ_i e m_j são literais complementares.

$$\frac{\ell_1 \lor \ell_2 \qquad \neg \ell_2 \lor \ell_3}{\ell_1 \lor \ell_3}$$

E.g., $\dfrac{P_{1,3} \lor P_{2,2}, \quad \neg P_{2,2}}{P_{1,3}}$



- correta e completa para lógica proposicional

Conversão para CNF

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

- Eliminar $\Leftrightarrow$, trocando $\alpha \Leftrightarrow \beta$ por $(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$.
  $(B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$

2. Eliminar $\Rightarrow$, trocando $\alpha \Rightarrow \beta$ por $\neg \alpha \vee \beta$.
   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg(P_{1,2} \vee P_{2,1}) \vee B_{1,1})$

3. Mover $\neg$ para dentro usando as leis de de Morgan e negação dupla:
   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge ((\neg P_{1,2} \vee \neg P_{2,1}) \vee B_{1,1})$

4. Aplicar a lei distributiva ($\wedge$ sobre $\vee$) e eliminar '(' ')':
   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg P_{1,2} \vee B_{1,1}) \wedge (\neg P_{2,1} \vee B_{1,1})$

Algoritmo de Resolução

- Primeiro a entrada é convertida em CNF.
- Em seguida a regra de resolução é aplicada às cláusulas restantes.
- Cada par que contém literais complementares é resolvido para gerar uma nova cláusula, que é adicionada ao conjunto..

Algoritmo de Resolução

- O processo continua até que:
  - não exista nenhuma cláusula nova a ser adicionada; nesse caso, não há consequência lógica
  - a cláusula vazia é derivada; assim, a consequência lógica é verificada.

Raciocinando por contraposição

Algoritmo da resolução

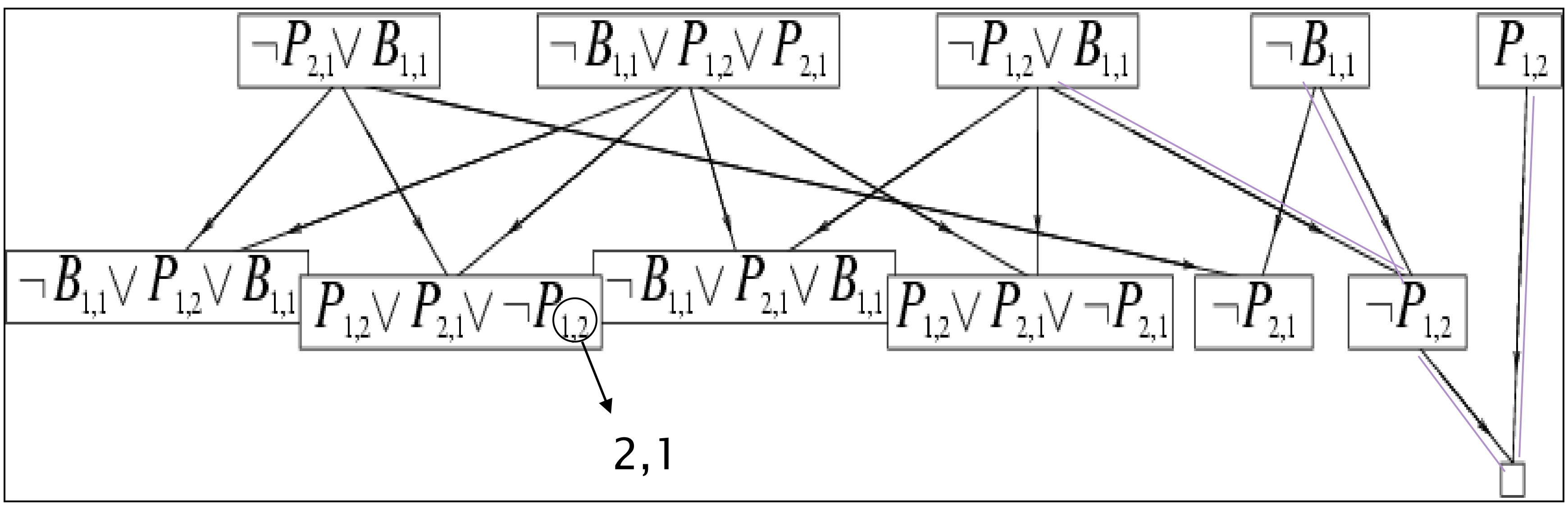
- Prova por contradição, i.e., para provar α em BC, mostrar que $KB \wedge \neg\alpha$ é insatisfatível

```
function PL-RESOLUTION(KB, α) returns true or false
    clauses ← the set of clauses in the CNF representation of KB ∧ ¬α
    new ← { }
    loop do
        for each Ci, Cj in clauses do
            resolvents ← PL-RESOLVE(Ci, Cj)
            if resolvents contains the empty clause then return true
            new ← new ∪ resolvents
        if new ⊆ clauses then return false
        clauses ← clauses ∪ new
```

Exemplo de resolução

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

- $((B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})) \wedge \neg B_{1,1}$



Cláusulas de Horn

- Bases de conhecimento reais muitas vezes só contêm cláusulas de uma espécie restrita, chamada **cláusula de Horn.**
  - Uma cláusula de Horn é uma disjunção de literais dos quais no *máximo um é positivo* .
  - Por exemplo:
    - a cláusula ($\neg L_{1,1}$ v $\neg$Brisa v $B_{1,1}$), onde $L_{1,1}$ significa que a posição do agente é [1,1], é uma cláusula de Horn, enquanto ($\neg B_{1,1}$ v $P_{1,2}$ v $P_{2,1}$) não é.
  - Importante por 3 razões:

  1. **Toda cláusula de Horn pode ser escrita como uma implicação cuja premissa é uma conjunção de literais positivos e a conclusão é um único literal positivo.**

($\neg L_{1,1}$ v $\neg$Brisa v $B_{1,1}$) pode ser escrita como $L_{1,1}$ ^ Brisa $\Rightarrow$ $B_{1,1}$)

Se o agente está em [1,1] e existe uma brisa, então [1,1] é arejado.

De Morgan e Eliminação da implicação
Tente: lembre-se que $L_{1,1}$ e Brisa devem ser considerados como α e $B_{1,1}$ como β.

Encadeamento para frente e para trás

2. A inferência com cláusulas de Horn pode ser feita através dos algoritmos de encadeamento para a frente e encadeamento para trás.

3. A decisão de conseqüência lógica com cláusulas de Horn pode ser feita em tempo linear em relação ao tamanho da base de conhecimento.

- **Encadeamento para frente (BC, q):** determina se um único símbolo proposicional *q* – a consulta – é permitido por uma base de conhecimento de cláusulas de Horn.
  - Ele começa a partir de fatos conhecidos na base de conhecimento. Se todas as premissas de uma implicação forem conhecidas, sua conclusão será acrescentada ao conjunto de fatos conhecidos.
- **Encadeamento para trás (BC, q)**: se a consulta q é reconhecida como verdadeira, não é necessário nenhum trabalho. Caso contrário, o algoritmo encontra as implicações da base de conhecimento que geram a conclusão q. Se for possível demonstrar que todas as premissas de uma dessas implicações são verdadeiras, então q é verdadeira.

Encadeamento pra frente e pra trás (Forward and backward chaining)

- Cláusula de Horn (resolução restrita)

  BC = conjunção de cláusulas de Horn

  - cláusula de Horn =
    - símbolo proposicional; ou
    - (conjunção de símbolos) $\Rightarrow$ símbolo

      (CORPO) $\Rightarrow$ CABEÇA

      (I.e., disjunção de literais nos quais no máximo um é positivo)
  - E.g., $C \wedge (B \Rightarrow A) \wedge (C \wedge D \Rightarrow B)$
- Modus Ponens (para Horn): completo para BC Horn

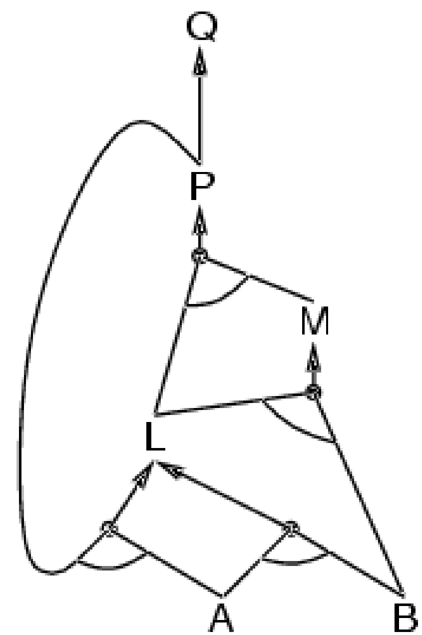
$$\frac{\alpha_1, \ldots, \alpha_n, \qquad \alpha_1 \wedge \ldots \wedge \alpha_n \Rightarrow \beta}{\beta}$$

- Podem ser usadas com forward chaining ou backward chaining.
- Algoritmos simples e de complexidade linear (em rel. ao tamanho da base de conhecimento) !

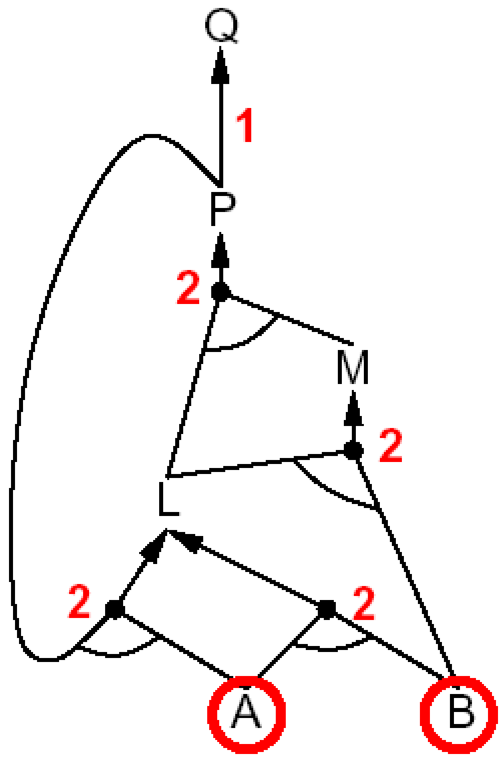
Forward chaining

- Começa a partir de fatos conhecidos (literais positivos) na base de conhecimento. Se todas as premissas de uma implicação forem verdade, sua conclusão será acrescentada ao conjunto de fatos conhecidos.

$$
\begin{aligned}
&P \Rightarrow Q \\
&L \wedge M \Rightarrow P \\
&B \wedge L \Rightarrow M \\
&A \wedge P \Rightarrow L \\
&A \wedge B \Rightarrow L \\
&A \\
&B
\end{aligned}
$$

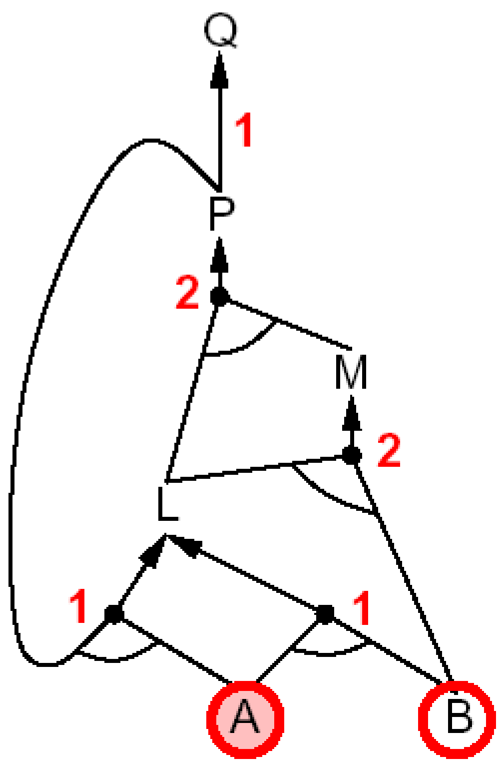


Forward chaining - Encadeamento Para Frente (exemplo)



$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$\boxed{A}$$
$$B$$

Encadeamento Para Frente (exemplo)



$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

$B \wedge L \Rightarrow M$

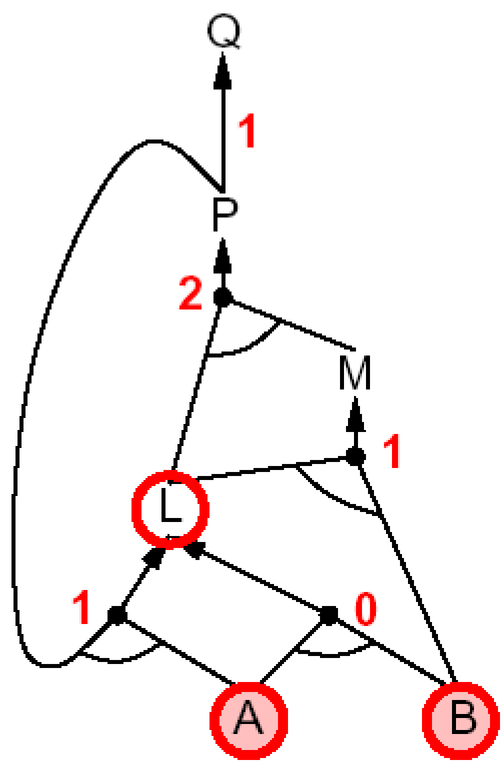
$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

$\boxed{B}$

Encadeamento Para Frente (exemplo)



$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

$B \wedge L \Rightarrow M$

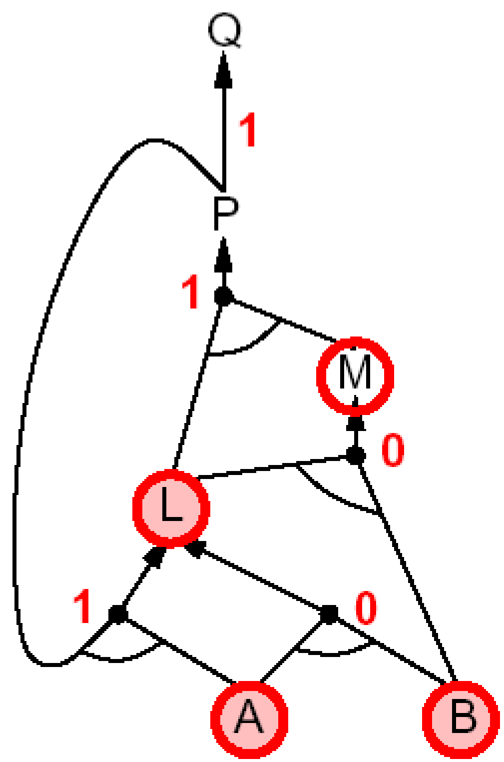
$A \wedge P \Rightarrow L$

$\boxed{A \wedge B \Rightarrow L}$

A

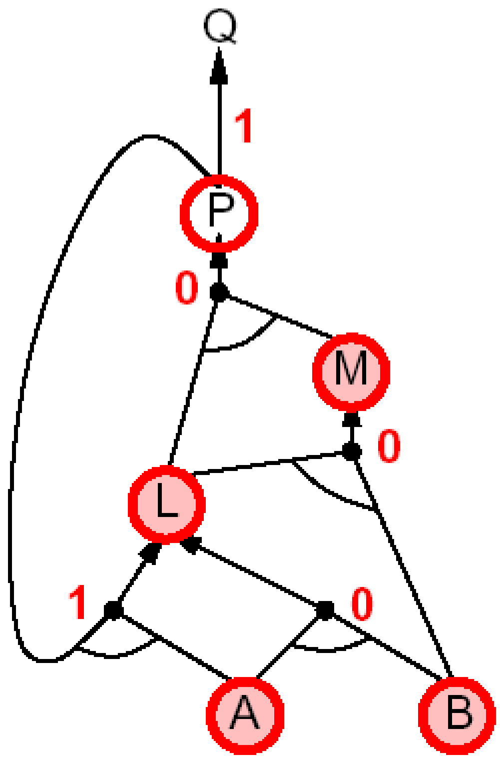
B

Encadeamento Para Frente (exemplo)



$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$\boxed{B \wedge L \Rightarrow M}$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$

Encadeamento Para Frente (exemplo)



$P \Rightarrow Q$

$\boxed{L \wedge M \Rightarrow P}$

$B \wedge L \Rightarrow M$

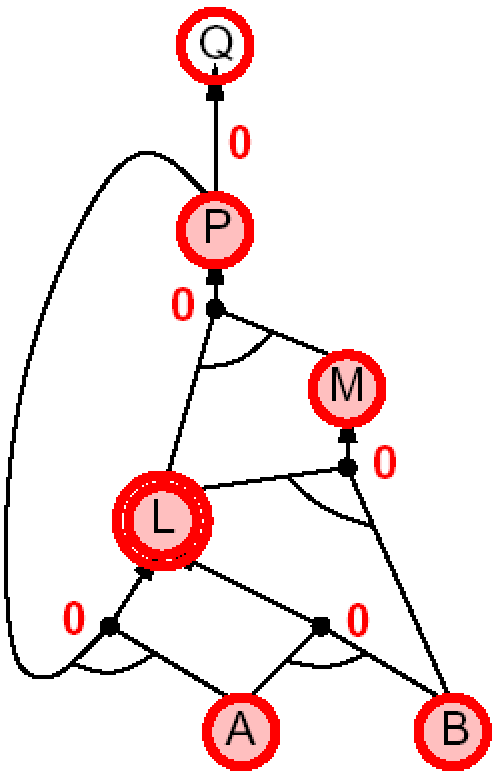
$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B

Encadeamento Para Frente (exemplo)



$\boxed{P \Rightarrow Q}$

$L \wedge M \Rightarrow P$

$B \wedge L \Rightarrow M$

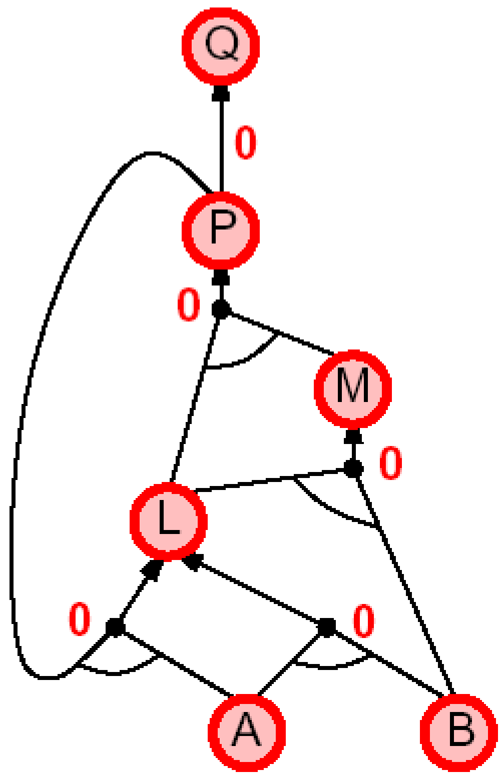
$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B

Encadeamento Para Frente (exemplo)



$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

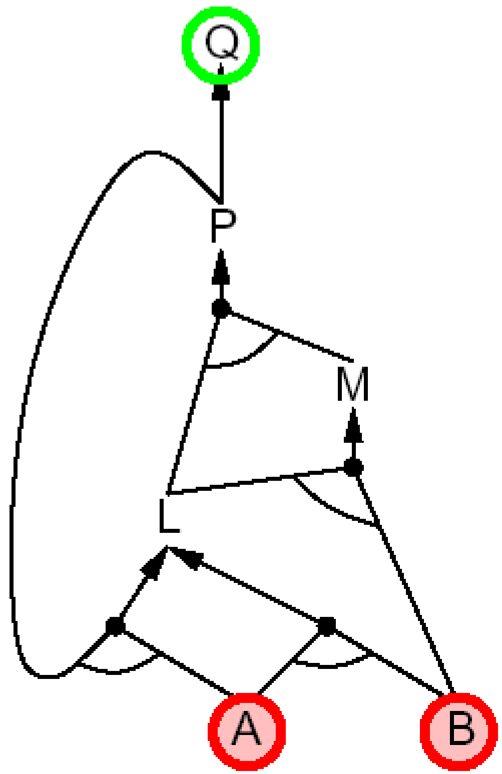
B

Backward chaining - Encadeamento Para Trás

- Funciona da pergunta *q* à base de conhecimento:
  - para provar *q* na BC,
    - verifique se *q* já faz parte de BC, ou
    - prove pela BC todas as premissas de alguma regra que conclua *q*

Evitar laços: verifique se os novos subgoals já foram provados ou já falharam!

Encadeamento Para Trás (exemplo)



$\boxed{P \Rightarrow Q}$

$L \wedge M \Rightarrow P$

$B \wedge L \Rightarrow M$

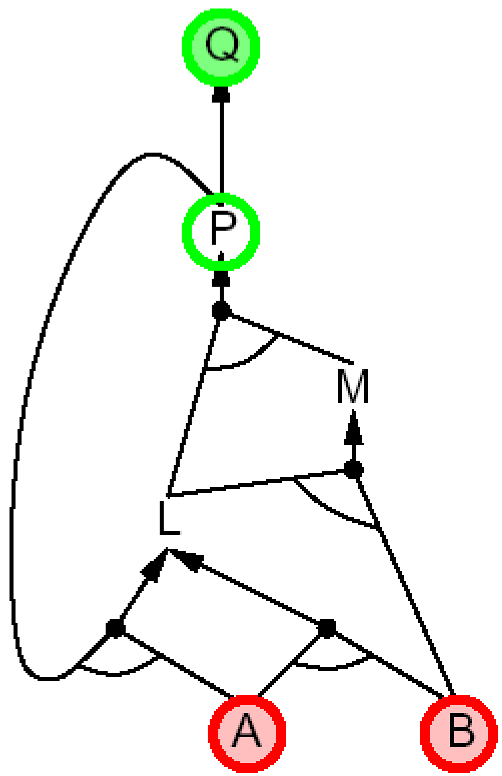
$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

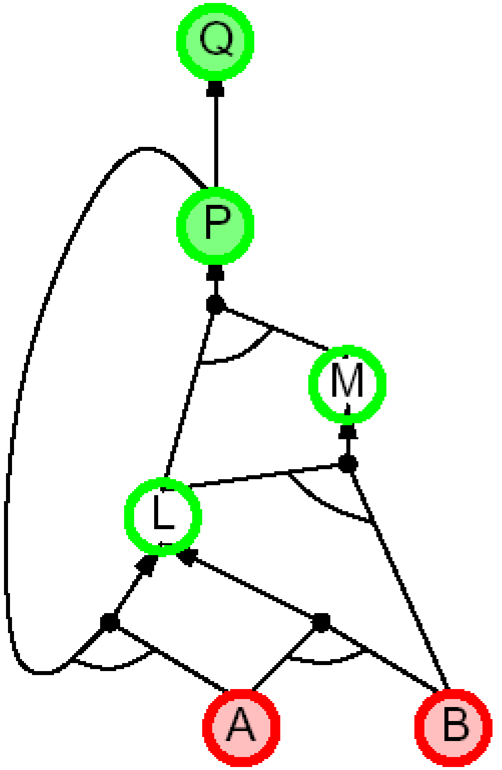
B

Encadeamento Para Trás (exemplo)



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

Encadeamento Para Trás (exemplo)



$P \Rightarrow Q$

$\boxed{L \wedge M \Rightarrow P}$

$B \wedge L \Rightarrow M$

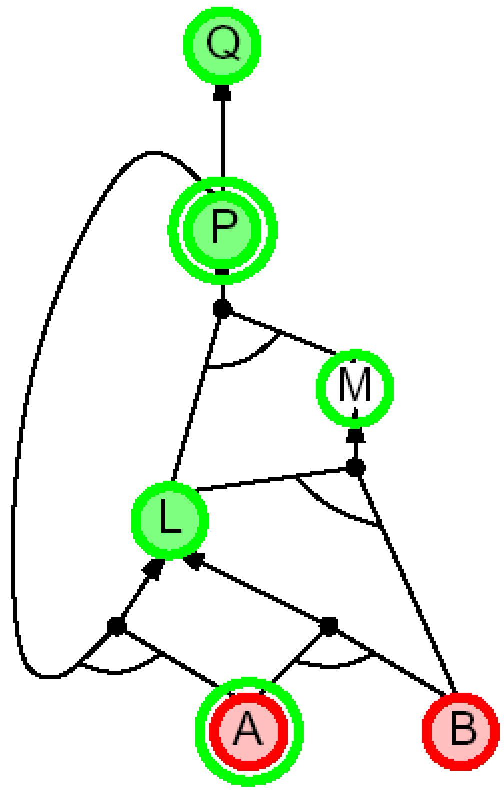
$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B

Encadeamento Para Trás (exemplo)



$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

$B \wedge L \Rightarrow M$

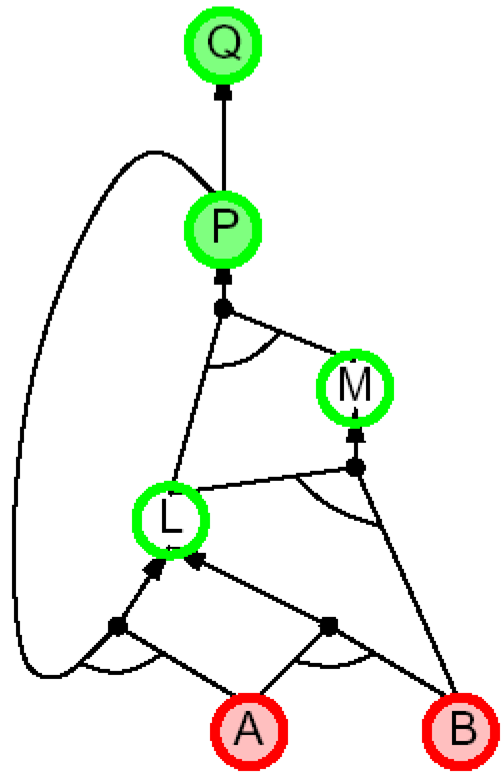
$\boxed{A \wedge P \Rightarrow L}$

$A \wedge B \Rightarrow L$

A

B

Encadeamento Para Trás (exemplo)



$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

$B \wedge L \Rightarrow M$

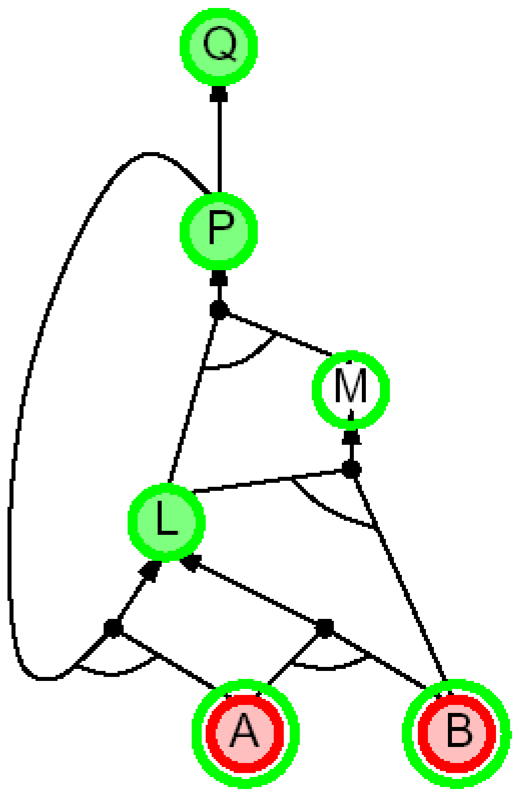
$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B

Encadeamento Para Trás (exemplo)



$P \Rightarrow Q$

$L \land M \Rightarrow P$

$B \land L \Rightarrow M$

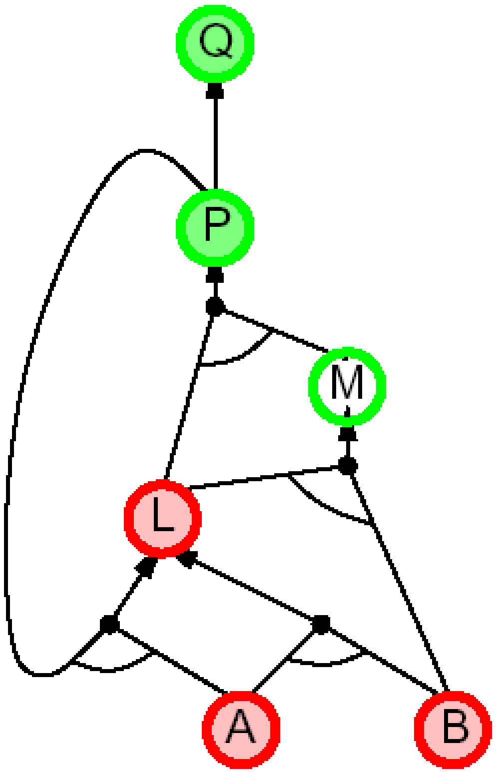
$A \land P \Rightarrow L$

$\boxed{A \land B \Rightarrow L}$

A

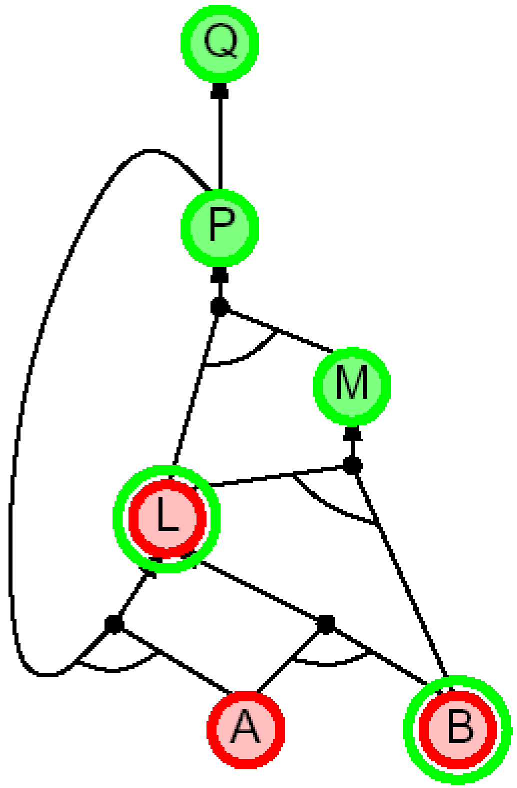
B

Encadeamento Para Trás (exemplo)



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

Encadeamento Para Trás (exemplo)



$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

$\boxed{B \wedge L \Rightarrow M}$

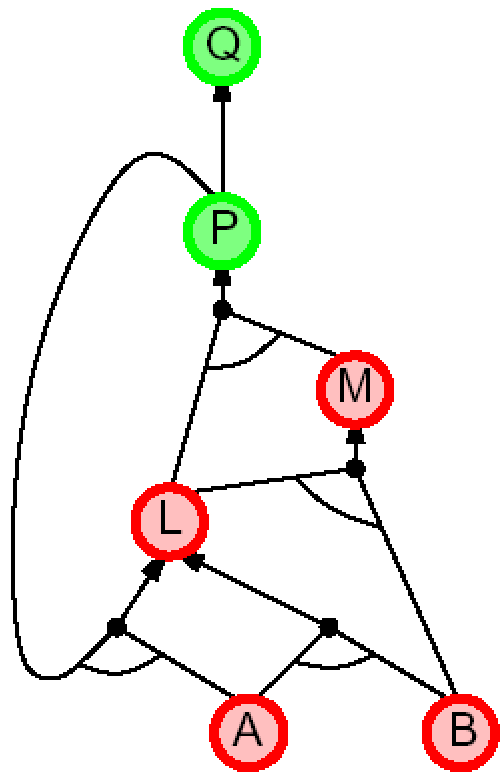
$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B

Encadeamento Para Trás (exemplo)



$P \Rightarrow Q$

$\boxed{L \wedge M \Rightarrow P}$

$B \wedge L \Rightarrow M$

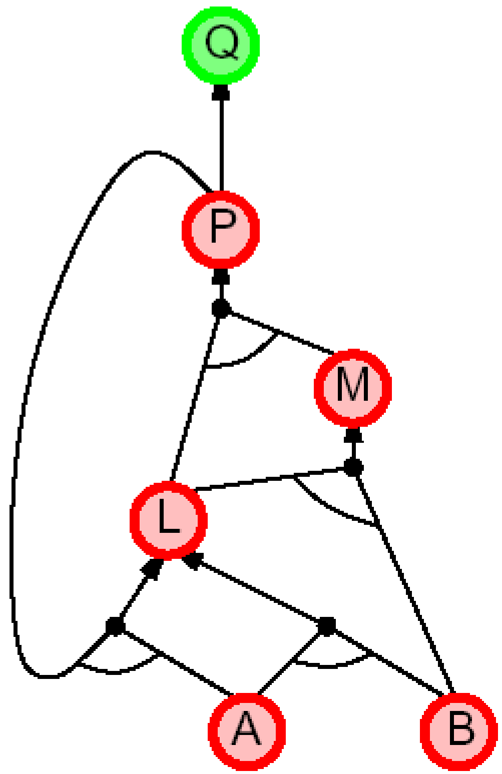
$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B

Encadeamento Para Trás (exemplo)



$\boxed{P \Rightarrow Q}$

$L \wedge M \Rightarrow P$

$B \wedge L \Rightarrow M$

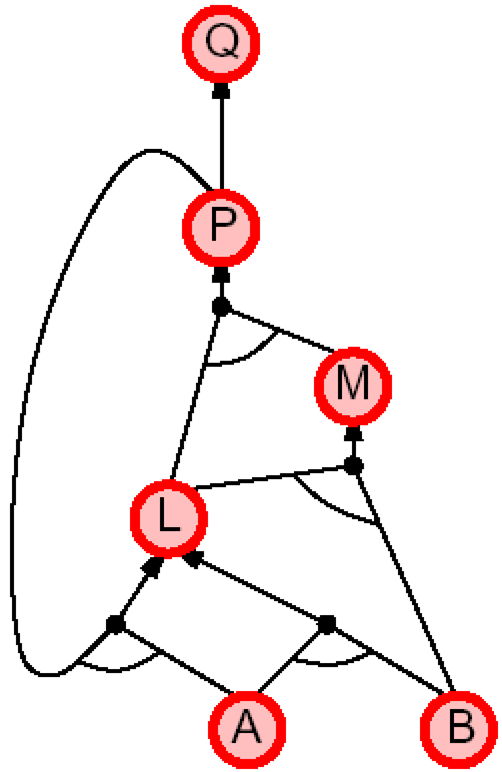
$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B

Encadeamento Para Trás (exemplo)



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

Forward vs. backward chaining

- ForwC é baseado no dados,
  - Pode ser usado para derivar conclusões a partir de percepções de entrada, sem uma consulta específica em mente;
  - Pode executar muito trabalho irrelevante para o objetivo;
  - Executa um trabalho extensivo;
- BackC é baseado no objetivo,
  - Apropriado para resolução de problemas;
  - Funciona em tempo linear
  - Complexidade de BackC pode ser muito menor do que linear em relação ao tamanho da base de conhecimento por que o processo só toca fatos relevantes para provar um objetivo.