Introduction to Artificial Intelligence

cuu duong than cong . com

Chapter 3: Knowledge Representation and Reasoning
(2) Propositional Logic (cont)

cuu duong than cong . com

Nguyễn Hải Minh, Ph.D
nhminh@fit.hcmus.edu.vn

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Outline

- Horn Clauses
- Forward and Backward Chaining
- DPLL Algorithm

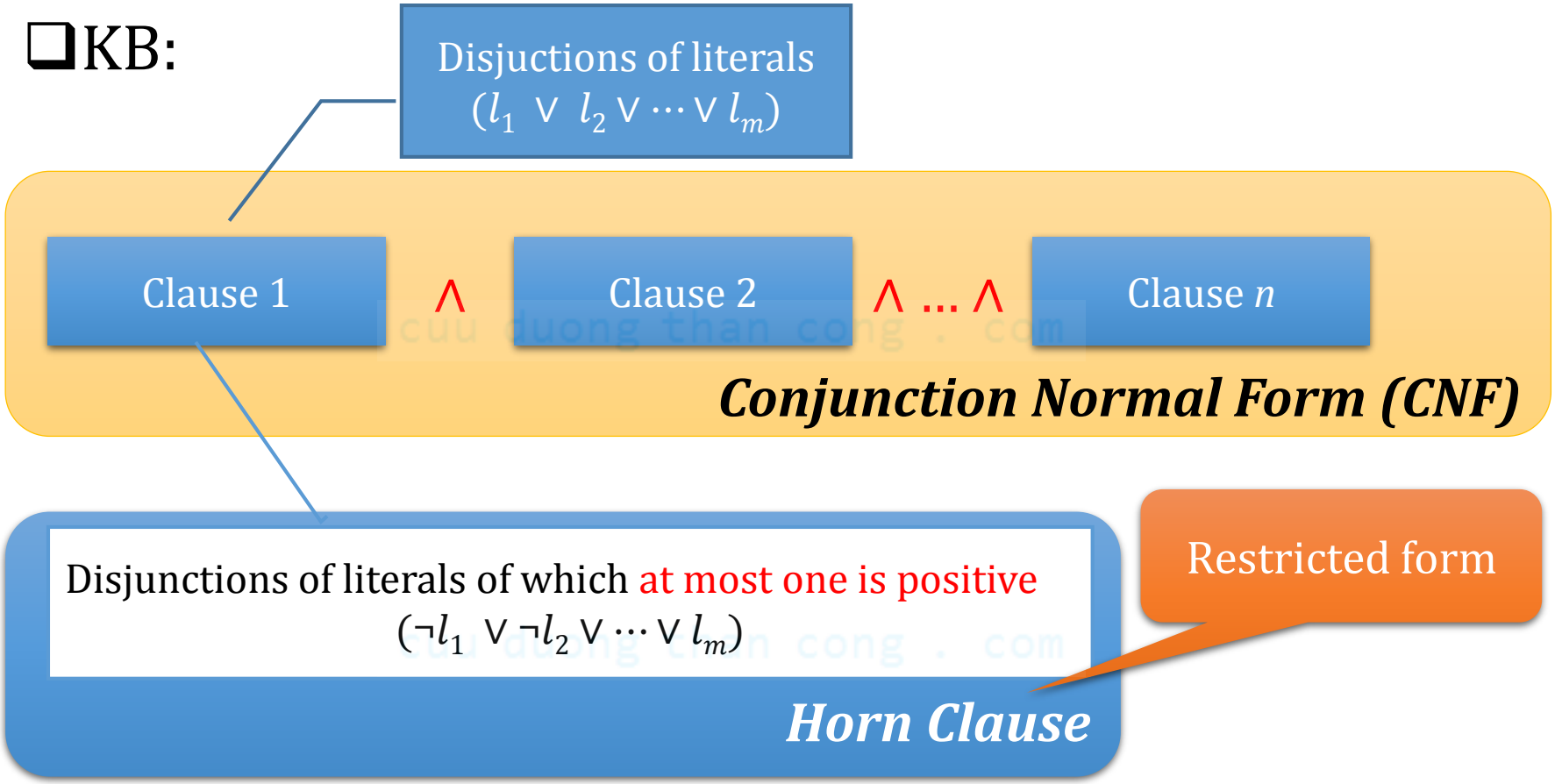
cuu duong than cong . com

cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Horn Clauses

❑KB:



E.g., $\neg B_{1,2} \vee \neg B_{2,1} \vee P_{2,2}$ $\Leftrightarrow$ $B_{1,2} \wedge B_{2,1} \Rightarrow P_{2,2}$

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Horn Clauses

- Modus Ponens for Horn Form:

$$\frac{(B \Rightarrow A), B}{A}$$

- More general version of the rule:

$$\frac{(B_1 \wedge B_2 \wedge \ldots \wedge B_k \Rightarrow A), \quad B_1, B_2, \ldots, Bk}{A}$$

- Why do we need Horn Clauses?
  - Horn clauses are closed under resolution
  - In the implication form, the sentence is easier to understand
  - Can be used with forward chaining or backward chaining.
  - These algorithms are very natural and run in linear time.

CuuDuongThanCong.com https://fb.com/tailieudientucntt

A grammar for CNF, Horn Clauses

$$
\begin{aligned}
CNFSentence &\rightarrow Clause_1 \wedge \cdots \wedge Clause_n \\
Clause &\rightarrow Literal_1 \vee \cdots \vee Literal_m \\
Literal &\rightarrow Symbol \mid \neg Symbol \\
Symbol &\rightarrow P \mid Q \mid R \mid \ldots \\
HornClauseForm &\rightarrow DefiniteClauseForm \mid GoalClauseForm \\
DefiniteClauseForm &\rightarrow (Symbol_1 \wedge \cdots \wedge Symbol_l) \Rightarrow Symbol \\
GoalClauseForm &\rightarrow (Symbol_1 \wedge \cdots \wedge Symbol_l) \Rightarrow False
\end{aligned}
$$

cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Forward chaining

- Idea:
  - fire any rule whose premises are satisfied in the *KB*,
  - add its conclusion to the *KB*, until query is found

query →

$P \Rightarrow Q$

KB →

$L \wedge M \Rightarrow P$

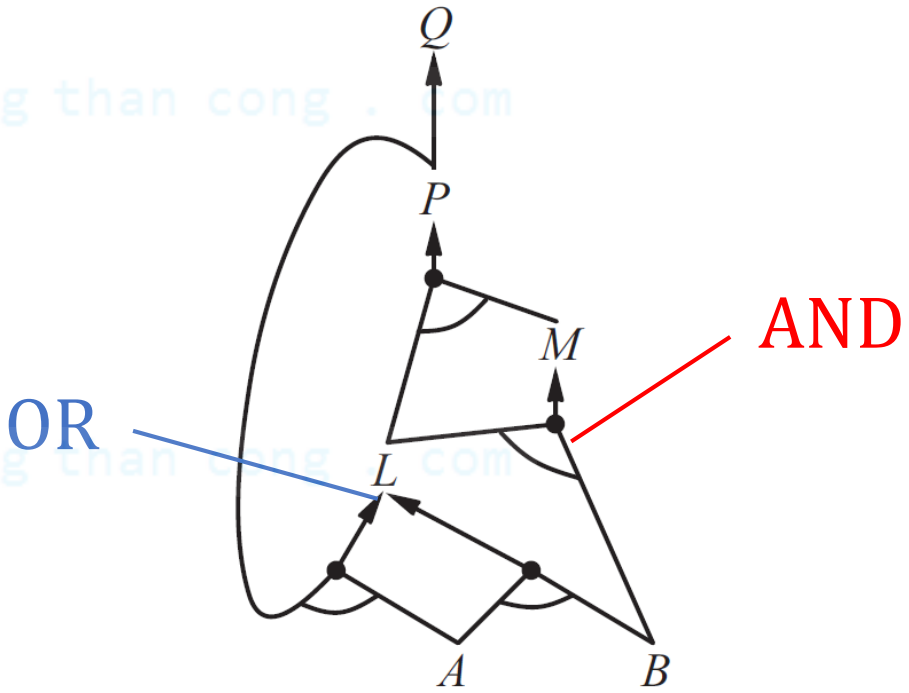
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Forward chaining algorithm

function PL-FC-ENTAILS?(KB, q) **returns** *true* or *false*

inputs: KB, the knowledge base, a set of propositional definite clauses
q, the query, a proposition symbol

$count \leftarrow$ a table, where $count[c]$ is the number of symbols in c's premise

$inferred \leftarrow$ a table, where $inferred[s]$ is initially *false* for all symbols

$agenda \leftarrow$ a queue of symbols, initially symbols known to be true in KB

while $agenda$ is not empty **do**
  $p \leftarrow$ POP($agenda$)
  if $p = q$ **then return** *true*
  if $inferred[p] = false$ **then**
    $inferred[p] \leftarrow true$
    for each clause c in KB where p is in c.PREMISE **do**
      decrement $count[c]$
      if $count[c] = 0$ **then** add c.CONCLUSION to $agenda$

return *false*

cuu duong than cong . com

- Forward chaining is sound and complete for Horn KB

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Forward chaining example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

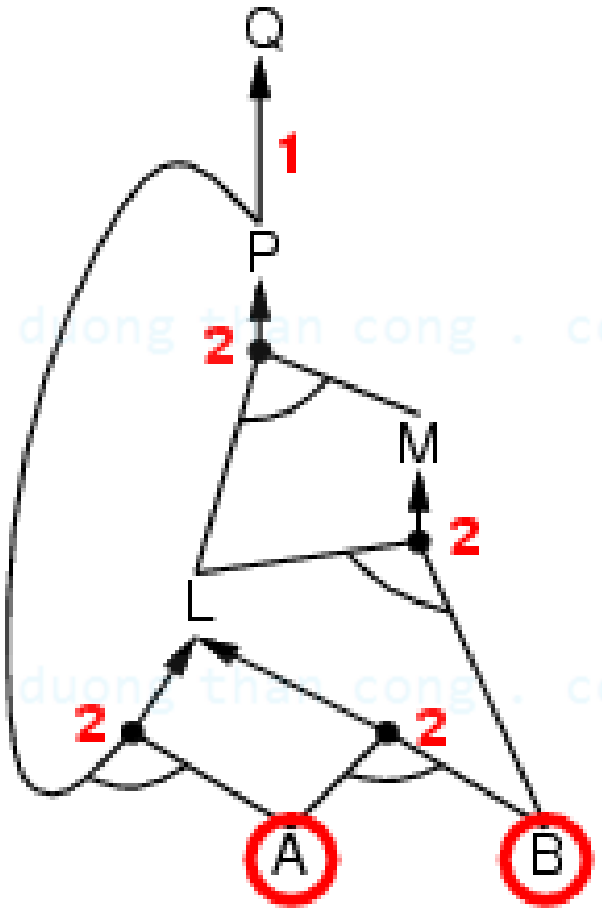
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Forward chaining example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

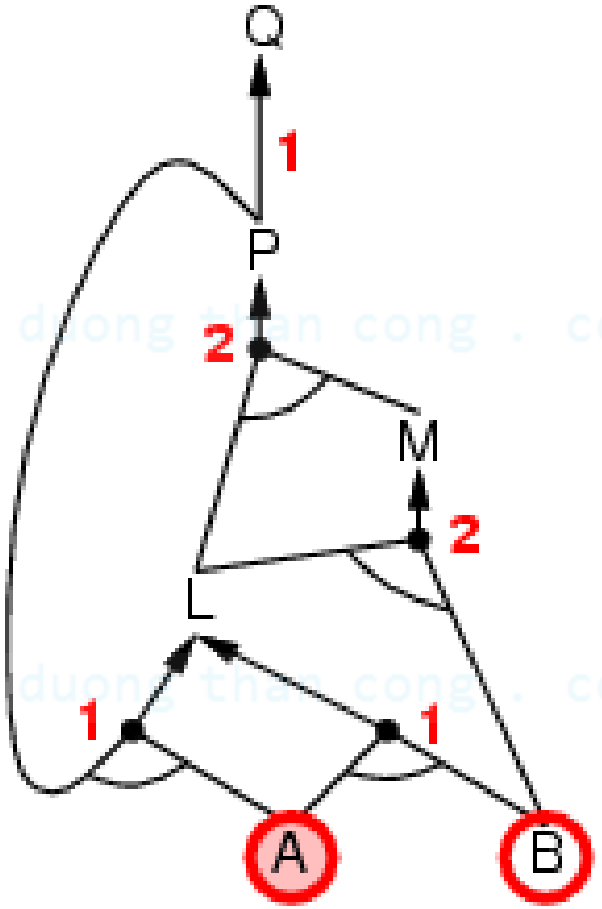
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Forward chaining example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

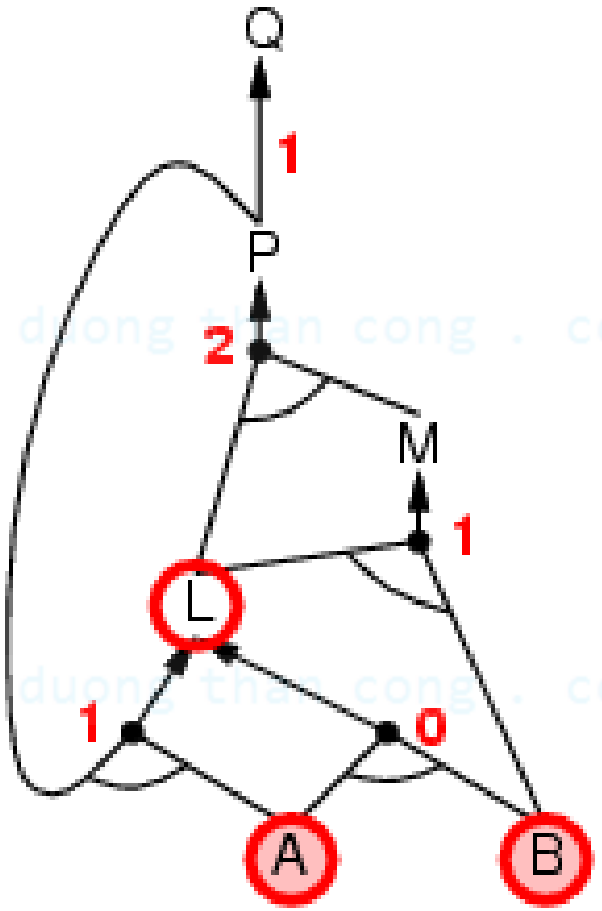
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Forward chaining example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

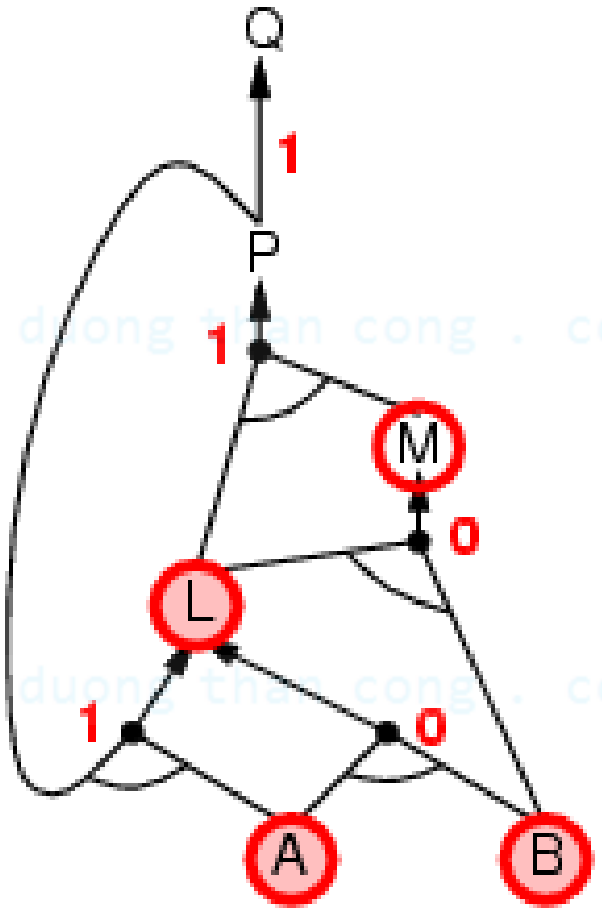
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Forward chaining example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

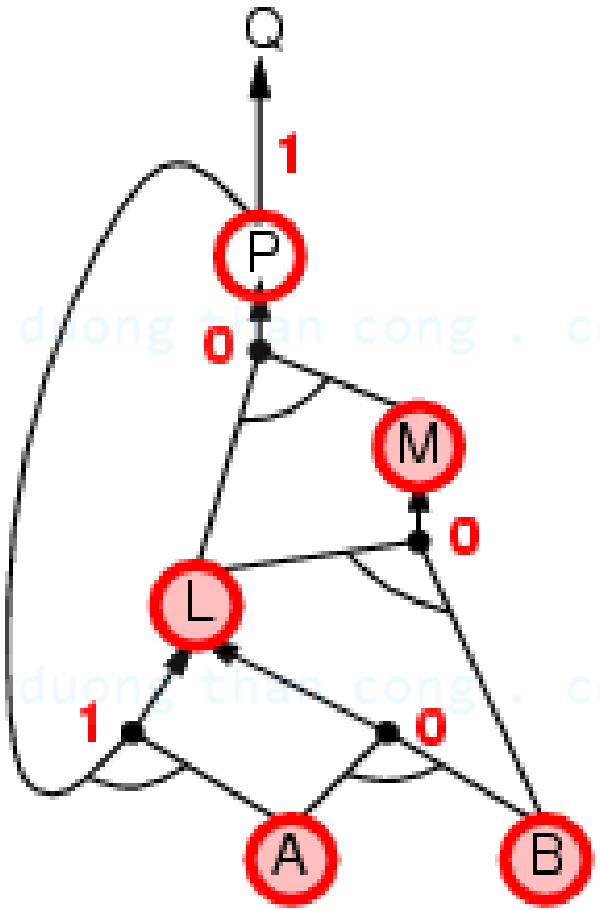
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Forward chaining example

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

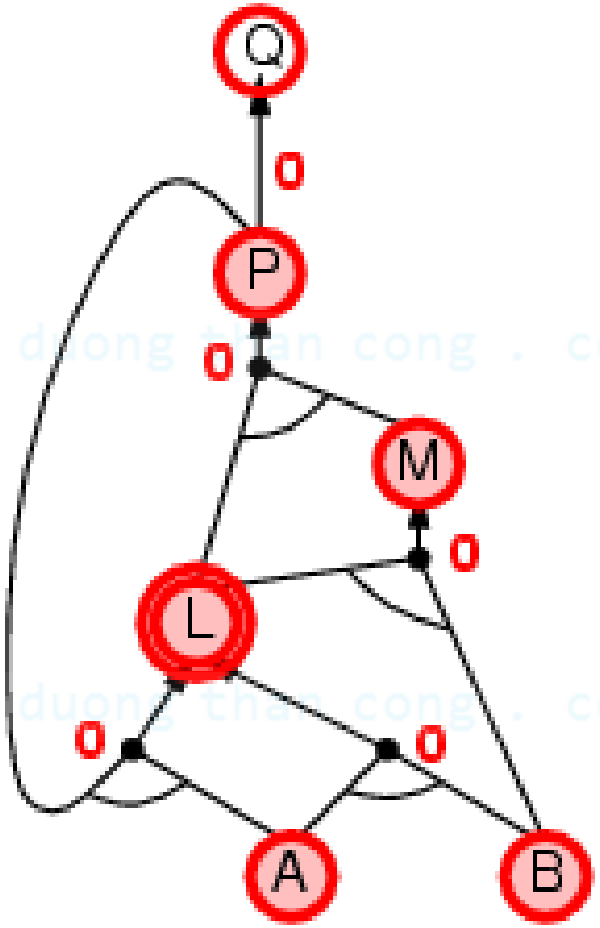
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

$$A$$

$$B$$



cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Forward chaining example

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

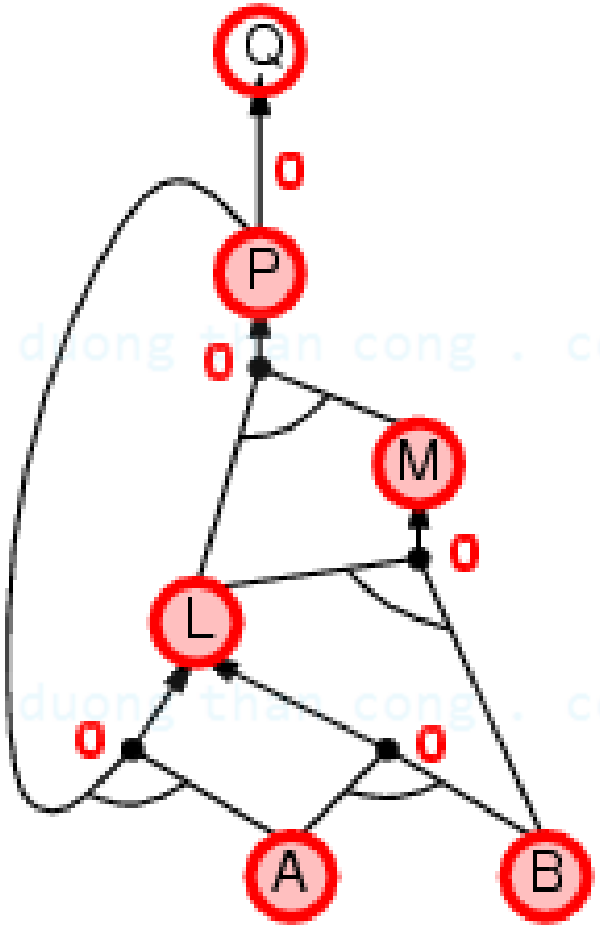
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

$$A$$

$$B$$



cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Forward chaining example

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

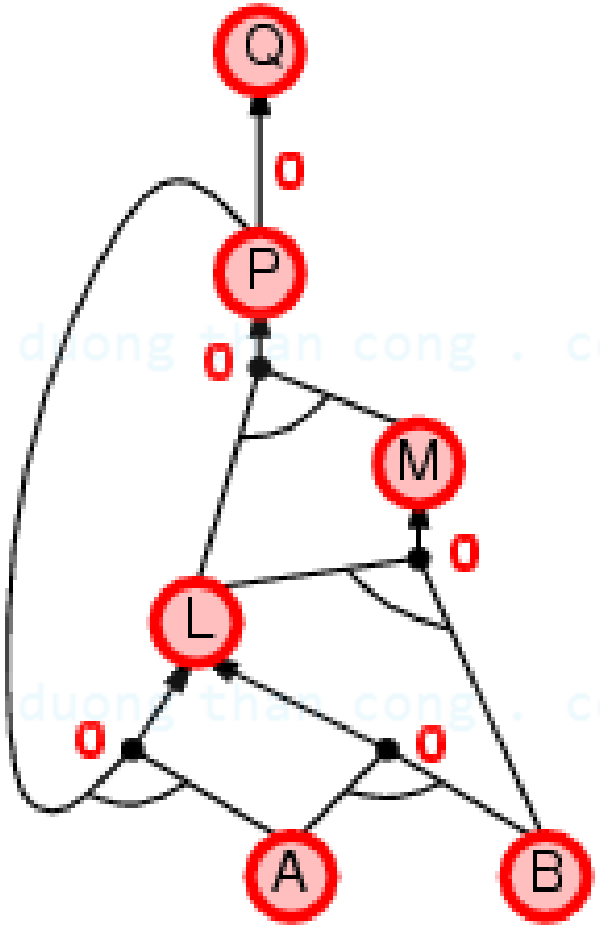
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

$$A$$

$$B$$



cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Backward chaining

- Idea: work backwards from the query q:
  - to prove q by BC,
    - check if q is known already, or
    - prove by BC all premises of some rule concluding q
- Avoid loops: check if new subgoal is already on the goal stack
- Avoid repeated work: check if new subgoal
  - has already been proved true, or
  - has already failed

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Backward chaining example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

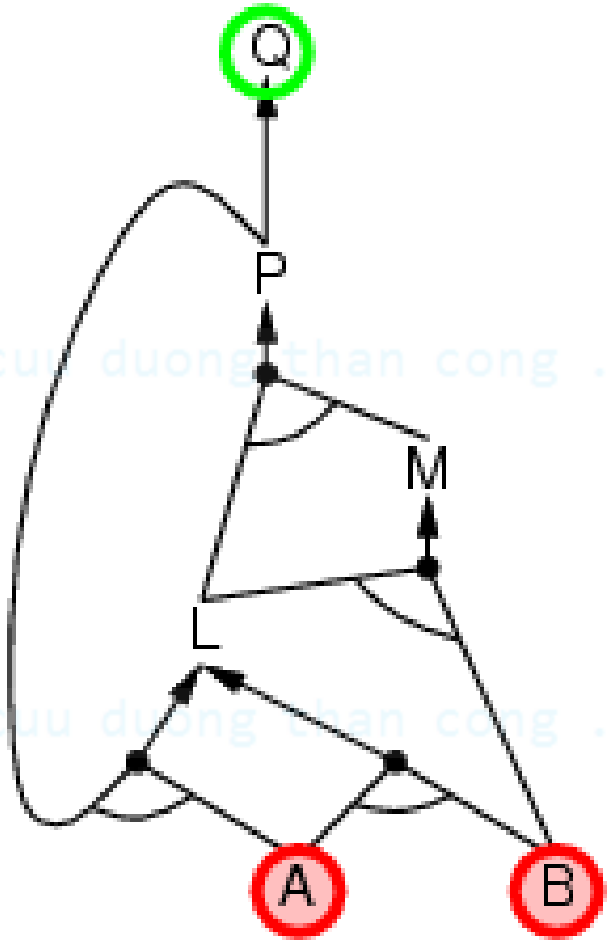
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Backward chaining example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

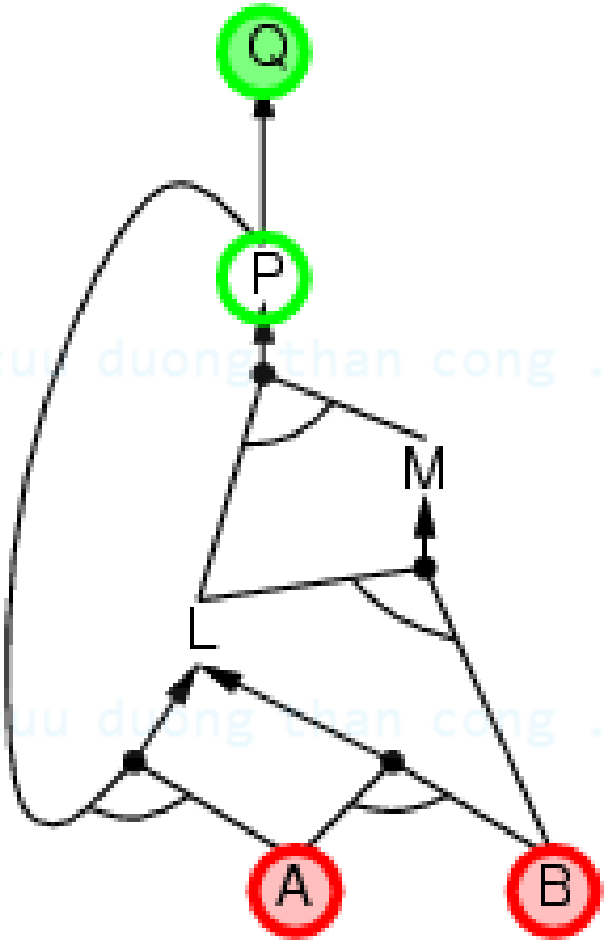
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Backward chaining example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

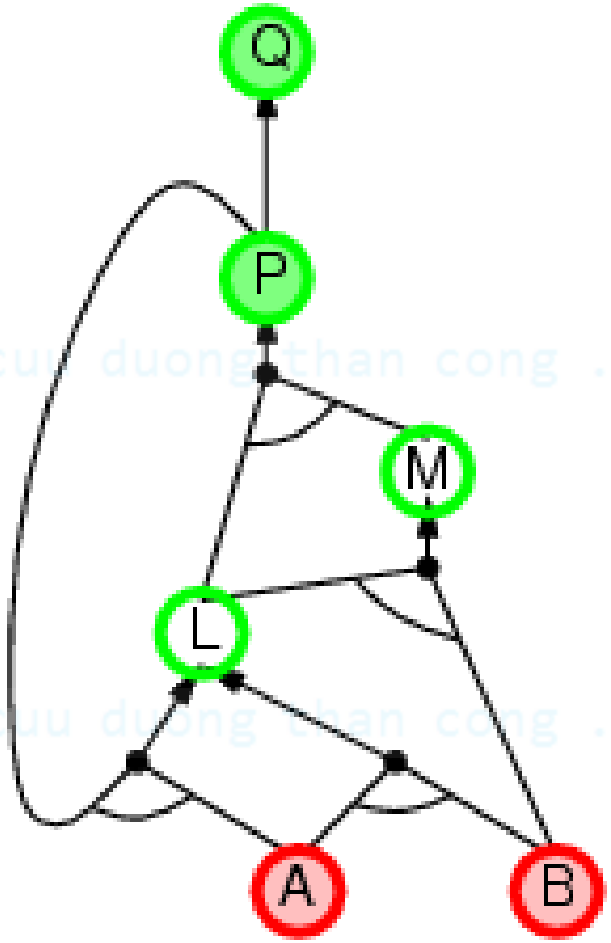
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Backward chaining example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

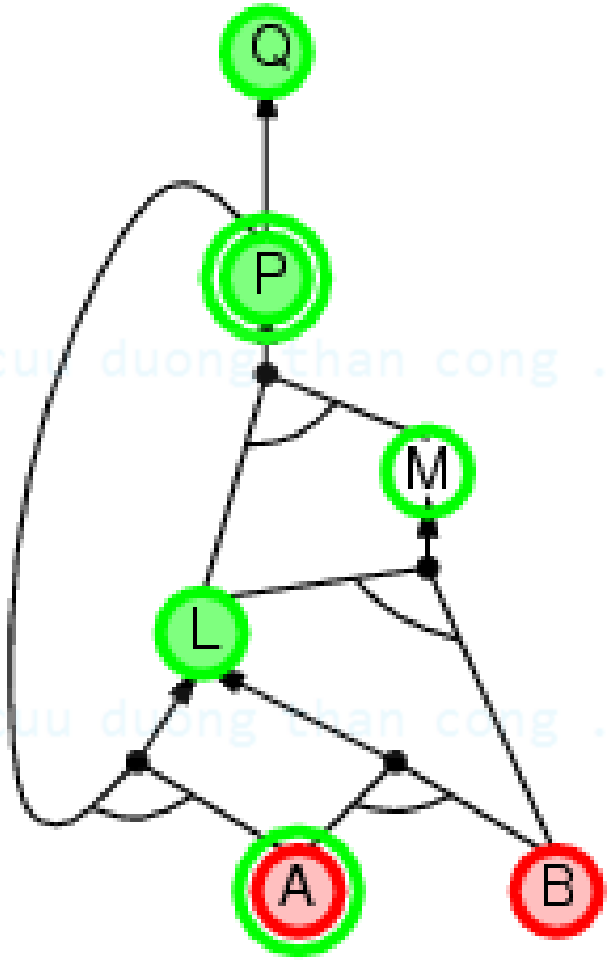
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Backward chaining example

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

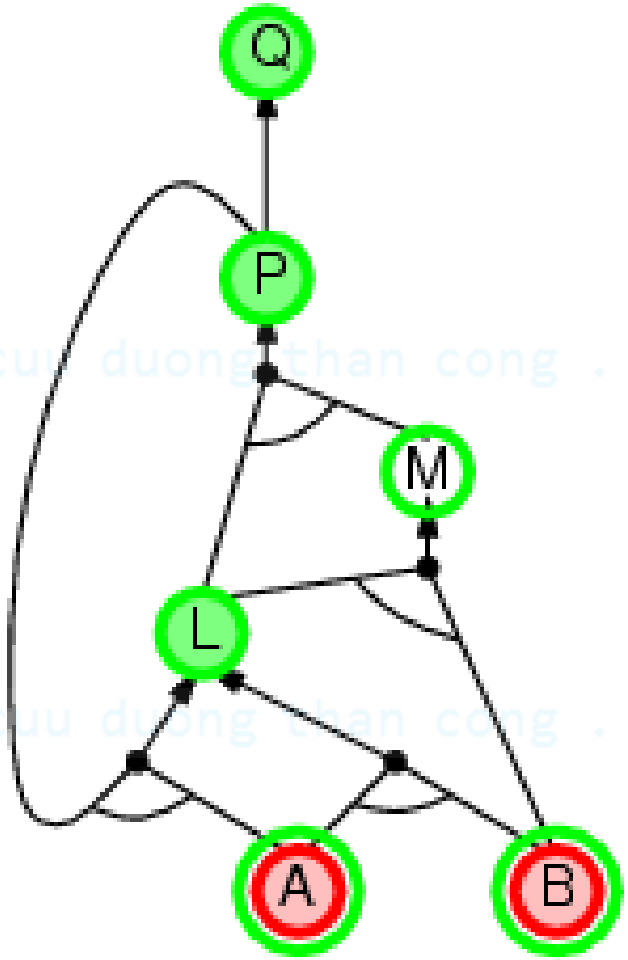
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

$$A$$

$$B$$



cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Backward chaining example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

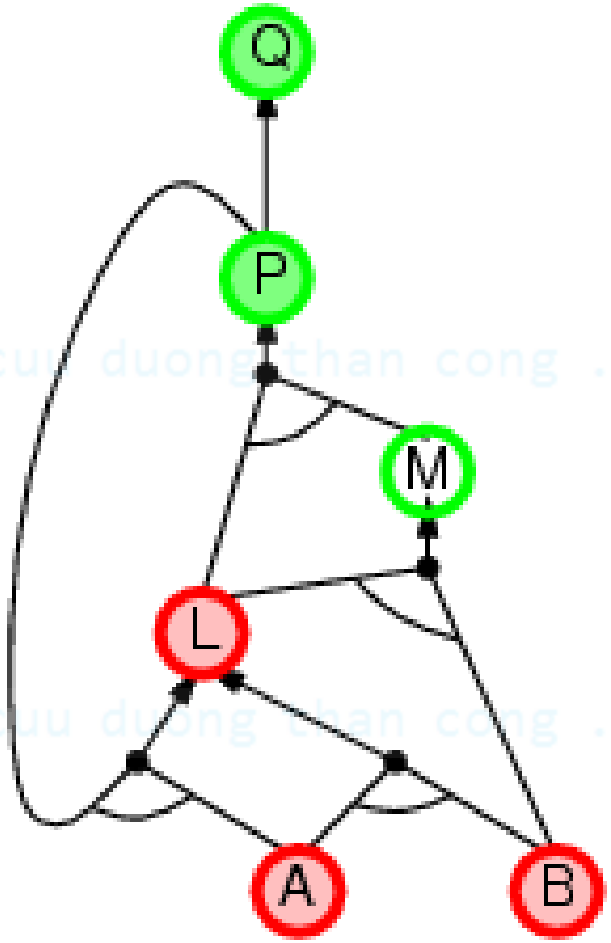
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Backward chaining example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

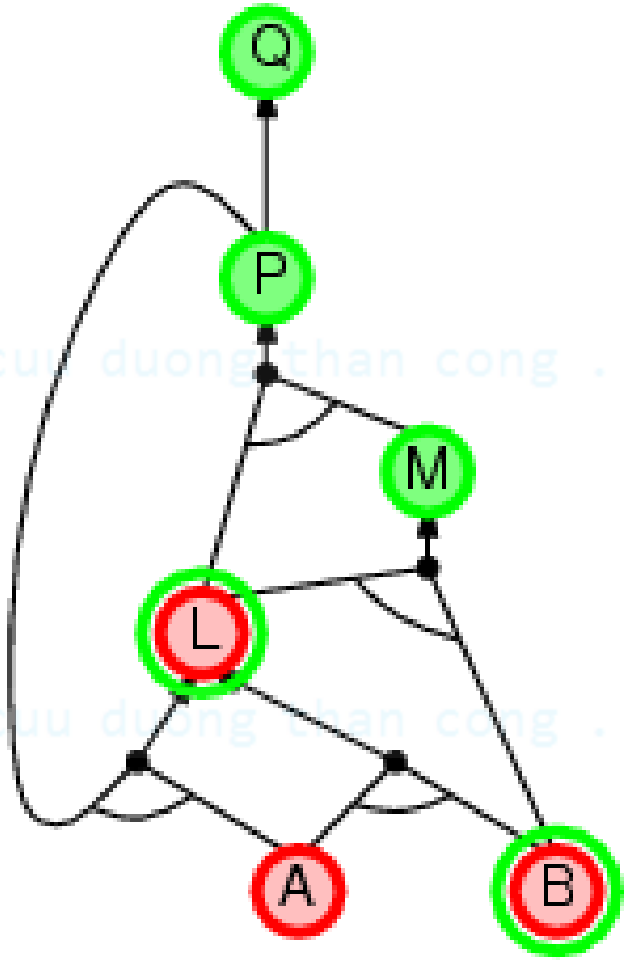
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Backward chaining example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

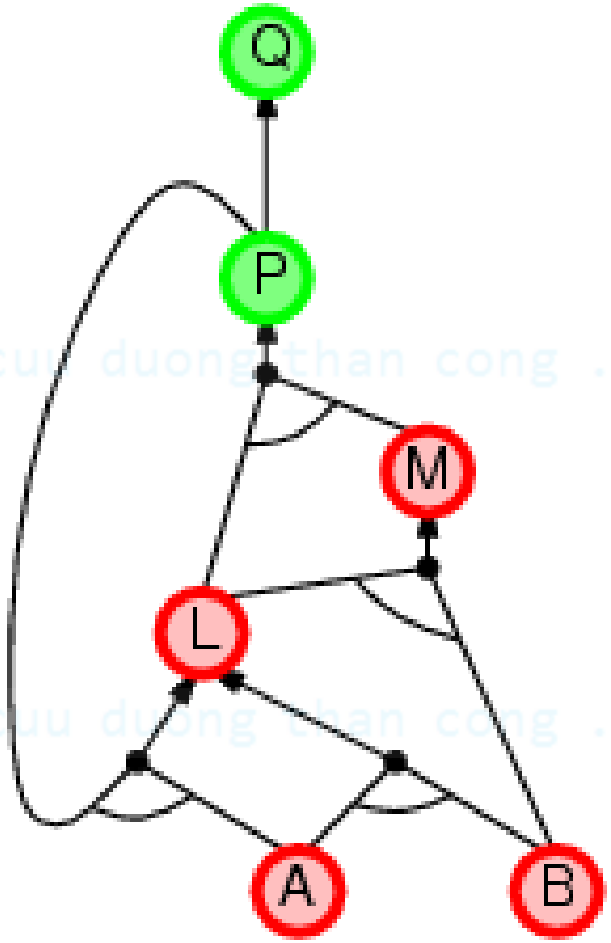
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Backward chaining example

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

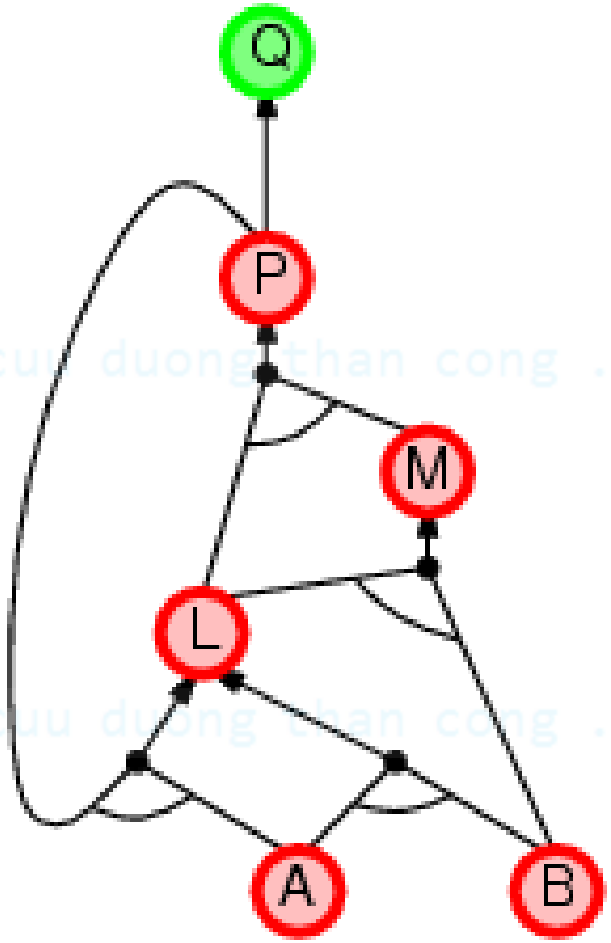
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

$$A$$

$$B$$



cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Backward chaining example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

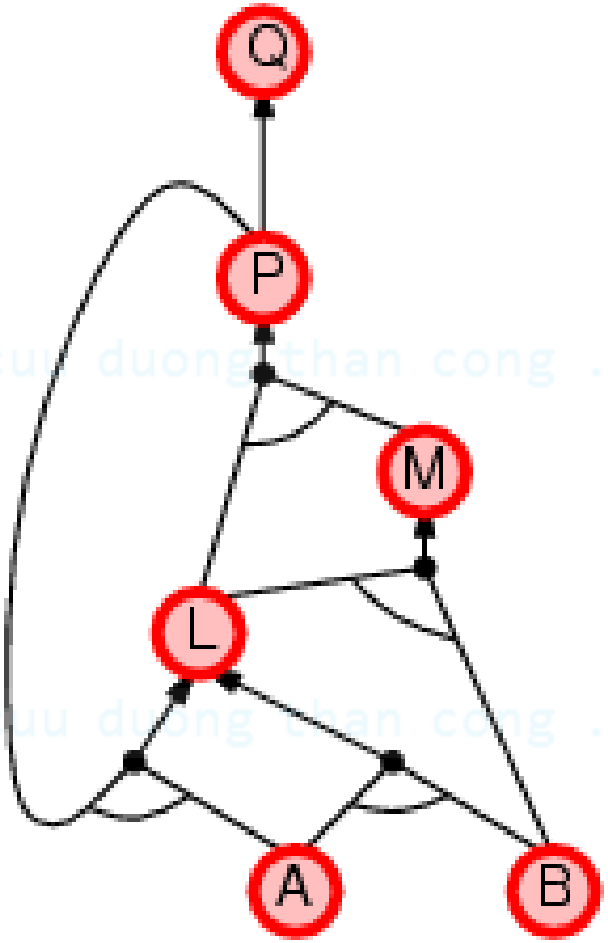
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Forward vs. backward chaining

- Forward chaining:
  - FC is data-driven, automatic, unconscious processing,
    - e.g., object recognition, routine decisions
  - May do lots of work that is irrelevant to the goal
- Backward chaining:
  - BC is goal-driven, appropriate for problem-solving,
    - e.g., Where are my keys? How do I get into a PhD program?
  - Complexity of BC can be much less than linear in size of KB

cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Efficient propositional inference

- The SAT problem (checking satisfiability)
  - Testing if KB |= α
  - This can be done by testing *Un*satisfiability of KB ∧ ¬α

- Two families of efficient algorithms for propositional inference:

1. Complete backtracking search algorithms
   - **DPLL** algorithm (*Davis, Putnam, Logemann, Loveland*)
2. Incomplete local search algorithms (hill-climbing)
   - **`WalkSAT`** algorithm

CuuDuongThanCong.com https://fb.com/tailieudientucntt

The DPLL algorithm

- Often called the *Davis-Putnam algorithm* (1960)
- Determine if an input propositional logic sentence (in CNF) is satisfiable
  - A recursive, depth-first enumeration of possible models.
- Improvements over truth table enumeration:
  1. Early termination
  2. Pure symbol heuristic
  3. Unit clause heuristic

CuuDuongThanCong.com https://fb.com/tailieudientucntt

The DPLL algorithm

1. Early termination
   - A clause is true if *any* literal is true.
   - A sentence is false if *any* clause is false.

cuu duong than cong . com

❑ Example:
   - (A ∨ B) ∧ (A ∨ C) is true if A is true, regardless B and C

cuu duong than cong . com

→ Avoid examination of entire subtrees in the search space

CuuDuongThanCong.com https://fb.com/tailieudientucntt

The DPLL algorithm

2. Pure symbol heuristic

- Pure symbol: always appears with the same "sign" in all clauses.
- e.g., In the three clauses $(A \vee \neg B)$, $(\neg B \vee \neg C)$, $(C \vee A)$, A and B are pure, C is impure.

→*Make a pure symbol literal true*

→*Doing so can never make a clause false*

cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt

The DPLL algorithm

3. Unit clause heuristic

- Unit clause: only one literal in the clause
- The only literal in a unit clause must be true

❑ Example:

- If the model contains B = true then $(\neg B \vee \neg C)$ simplifies to $\neg C$, which is a unit clause
- C must be false (so that $\neg C$ = true)
- Then A must be true (so that $C \vee A$ is true)

→ *Unit propagation*

cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt

The DPLL algorithm

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*
  } 1. Early Termination
  P, *value* ← 2 FIND-PURE-SYMBOL(*symbols*, *clauses*, *model*)
  if *P* is non-null **then return** DPLL(*clauses*, *symbols* – *P*, *model* ∪ {*P*=*value*})
  P, *value* ← 3 FIND-UNIT-CLAUSE(*clauses*, *model*)
  if *P* is non-null **then return** DPLL(*clauses*, *symbols* – *P*, *model* ∪ {*P*=*value*})
  P ← FIRST(*symbols*); *rest* ← REST(*symbols*)
  return DPLL(*clauses*, *rest*, *model* ∪ {*P*=*true*}) **or**
    DPLL(*clauses*, *rest*, *model* ∪ {*P*=*false*}))

CuuDuongThanCong.com https://fb.com/tailieudientucntt

The `WalkSAT` algorithm

- Incomplete, local search algorithm
- Evaluation function: The min-conflict heuristic of minimizing the number of unsatisfied clauses
- Balance between greediness and randomness

cuu duong than cong . com

cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt

The `WalkSAT` algorithm

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, typically around 0.5
max_flips, number of flips allowed before giving up

$model \leftarrow$ a random assignment of *true*/*false* to the symbols in $clauses$
for $i = 1$ **to** max_flips **do**
if $model$ satisfies $clauses$ **then return** $model$
$clause \leftarrow$ 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*

cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt

The `WalkSAT` algorithm

- When WalkSAT returns a model
  - The input sentence is Satisfiable
- When it returns false:
  - The sentence is unsatisfiable OR
  - We need to give it more time

→ *Most useful when we expect a solution to exist*

cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Inference-based agents in the Wumpus world

A wumpus-world agent using propositional logic:

$\neg P_{1,1}$

$\neg W_{1,1}$

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

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

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

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

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

...

$\Rightarrow$ 64 distinct proposition symbols, 155 sentences

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Expressiveness limitation of propositional logic

- KB contains "physics" sentences for every single square
- For every time t and every location $[x,y]$,

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

- Rapid proliferation of clauses

cuu duong than cong . com

CuuDuongThanCong.com https://fb.com/tailieudientucntt

Summary

- Logical agents apply inference to a knowledge base to derive new information and make decisions
- Basic concepts of logic:
  - syntax: formal structure of sentences
  - semantics: truth of sentences wrt models
  - entailment: necessary truth of one sentence given another
  - inference: deriving sentences from other sentences
  - soundness: derivations produce only entailed sentences
  - completeness: derivations can produce all entailed sentences
- Wumpus world requires the ability to represent partial and negated information, reason by cases, etc.
- Resolution is complete for propositional logic
  Forward, backward chaining are linear-time, complete for Horn clauses
- Propositional logic lacks expressive power

CuuDuongThanCong.com https://fb.com/tailieudientucntt