

Figure 1: A Wumpus Wonderland

1 Discussion-Based Warm Ups

(a) Given the following, can you prove that the unicorn is mythical? How about magical? Horned?

*If the unicorn is mythical, then it is immortal, but if it is not mythical, then it is a mortal mammal. If the unicorn is either immortal or a mammal, then it is horned. The unicorn is magical if it is horned.*

Do not attempt to formalize your solution here. Rather, turn to the people around you and rea son through this question. How many possible worlds would we have to enumerate to give a formal answer?

Có hai khả năng đối lập do điều kiện (2): unicorn **có thể** là mythical hoặc không mythical.

* Trường hợp 1: Nếu *mythical*, theo (1) nó *immortal*. Từ (3) immortal ⇒ horned. Từ (4) horned ⇒ magical. Vậy trong trường hợp này: mythical ⇒ immortal ⇒ horned ⇒ magical.
* Trường hợp 2: Nếu *không mythical*, theo (2) nó là *mortal mammal*. Là *mammal* kết hợp với (3) (mammal ⇒ horned) suy ra horned, rồi (4) suy ra magical.

Kết luận: trong cả hai trường hợp (mythical hoặc không), ta *luôn* kết luận unicorn là **horned** và **magical**. Nhưng **không** có đủ thông tin để buộc unicorn **phải** là *mythical* — cả hai khả năng (mythical và not mythical) đều tồn tại.

Vậy: **Không thể chứng minh unicorn là mythical. Nhưng có thể chứng minh nó horned và magical**

Số thế giới có thể cần liệt kê: với 4 thuộc tính xuất hiện (mythical, immortal, mammal, horned, magical) nhưng các ràng buộc liên kết chặt chẽ, ta thực tế chỉ cần xem hai thế giới (mythical / not mythical). (Nếu formal hóa toàn bộ biến có thể nhiều hơn, nhưng vì mối quan hệ tuyến tính ở trên chỉ có độ phân rẽ ở Mythic/¬Mythic nên chỉ hai trường hợp cần xét.)

(b) Determine which of the following are correct, and explain your reasoning:

(i) (*A ∧ B*) *|*= (*A ⇐⇒ B*)

(ii) *A ⇐⇒ B |*= *A ∨ B*

(iii) *A ⇐⇒ B |*= *¬A ∨ B*

(iv) (*A ∧ B*) =*⇒ C |*= (*A* =*⇒ C*) *∨ B* =*⇒ C*)

(v) (*A ∨ B*) *∧ ¬*(*A* =*⇒ B*) is satisfiable.

(*A ∧ B*) *|*= (*A ⇐⇒ B*)

* có nghĩa A và B đều đúng. Khi A và B đều đúng thì (A iff B) *đúng* (vì cả hai đều cùng giá trị true).
* Tuy nhiên entailment yêu cầu: mọi mô hình mà đúng thì đúng — điều này thật: nếu A và B đều đúng, thì A⇔B là true.
* **Kết luận:** (i) **đúng**.

*A ⇐⇒ B |*= *A ∨ B*

* cho biết A và B có cùng giá trị (cùng true hoặc cùng false). Nhưng trong trường hợp cả hai false thì vẫn đúng, còn sẽ **false**. Vì tồn tại mô hình thỏa nhưng không thỏa (khi A=F,B=F), entailment không đúng.
* **Kết luận:** (ii) **sai**.

*A ⇐⇒ B |*= *¬A ∨ B*

* tương đương . Does entail ? Nếu thì khi A true thì B true (vì phải bằng nhau) — vậy luôn đúng trong mọi mô hình thỏa . Khi A false và B false thì cũng true. Không có mô hình thỏa mà sai.
* **Kết luận:** (iii) **đúng**.

(*A ∧ B*) =*⇒ C |*= (*A* =*⇒ C*) *∨ B* =*⇒ C*)

* Giả sử đúng; điều này chỉ nói: nếu cả A và B cùng true thì C true. Nhưng có thể có mô hình nơi A true, B false, C false — trong đó vẫn đúng (tiền đề A∧B false khiến implication true), nhưng có thể false (A true, C false) và có thể true (B false ⇒ implication true). Ta cần **hoặc** phải true trong mọi mô hình để entailment đúng. Nhưng ta có thể tạo mô hình A=true, B=false, C=false: thì (A∧B)⇒C là true (tiền đề false), nhưng A⇒C là false và B⇒C là true → trong trường hợp này (A⇒C)∨(B⇒C) = true (vì B⇒C true). Tuy nhiên cần kiểm tra có mô hình nào khiến cả A⇒C và B⇒C đều false trong khi (A∧B)⇒C true? Cả hai false nghĩa A=true,C=false và B=true,C=false; tức A=true,B=true,C=false thì (A∧B)⇒C là false. Vậy không tồn tại mô hình làm (A∧B)⇒C true mà (A⇒C) cũng false và (B⇒C) false cùng lúc? Thực ra, để cả (A⇒C) và (B⇒C) đều false thì cần A=true,B=true,C=false — nhưng đó làm (A∧B)⇒C false. Do đó **mọi mô hình** thỏa (A∧B)⇒C sẽ làm ít nhất một trong (A⇒C) hoặc (B⇒C) true? Cần kiểm chứng kĩ: xét mô hình A=true, B=false, C=false → (A⇒C) false, (B⇒C) true → disjunction true. A=false,B=true,C=false → similar. A=false,B=false,C=false → both A⇒C true and B⇒C true (vì tiền đề false). A=true,B=true,C=true → both true. Vì không có mô hình làm cả hai implications false mà vẫn giữ (A∧B)⇒C true, nên entailment **có vẻ đúng**. Nhưng is it logically valid that entails ? Consider truth table: the only row that makes (A∧B)⇒C true but both A⇒C and B⇒C false would have to have A⇒C false (A=T,C=F) and B⇒C false (B=T,C=F) → then A=T,B=T,C=F which makes (A∧B)⇒C false. So impossible. Thus entailment **đúng**.
* **Kết luận:** (iv) **đúng**.

(*A ∨ B*) *∧ ¬*(*A* =*⇒ B*) is satisfiable.

* tương đương . Vì là , phủ định là . Khi đưa vào: ≡ (vì A∧¬B làm A∨B tự động true). Và có mô hình (A=true, B=false). Vì tồn tại mô hình, biểu thức **satisfiable**.
* **Kết luận:** (v) **satisfiable** (có thể thỏa).

(c) What is the difference between satisfiability and entailment (think about the purpose and requirements of each)?

* **Satisfiability (thỏa nghiệm):** Một công thức (hoặc một tập công thức) *satisfiable* nếu tồn tại ít nhất một mô hình (assignments cho biến) khiến công thức (hoặc tất cả công thức trong tập) trở thành true. Mục đích: kiểm tra xem bộ tiền đề có nhất quán (không mâu thuẫn) hay không.
* **Entailment (hệ quả logic, |=):** Tập tiền đề *entails* một câu nếu **trong mọi mô hình** mà true thì cũng true. Mục đích: kiểm tra xem là hậu quả bắt buộc của hay không.
* Tóm tắt ngắn: *Satisfiability* hỏi “có tồn tại một mô hình thỏa?”; *Entailment* hỏi “trong **mọi** mô hình thỏa tiền đề, kết luận có luôn đúng không?”.

1

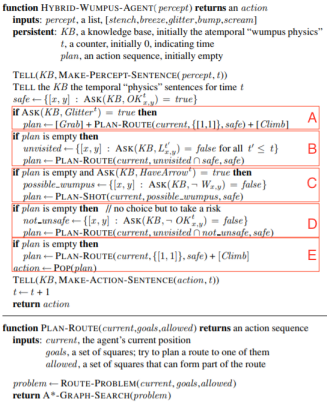


Figure 2: Hybrid-Wumpus-Agent from AIMIA 3rd ed. It uses a propositional knowledge base to infer the state of the world, and a combination of problem-solving search and domain-specific code to decide what actions to take.

2

2 Wandering in Wumpus World

We bring together what we have learned in lecture as well as the ideas of search so far in order to construct wumpus world agents that use propositional logic. The first step is to enable the agent to deduce, to the extent possible, the state of the world given its percept history. This requires writing down a complete logical model of the effects of actions. We also show how the agent can keep track of the world efficiently without going back into the percept history for each inference. Finally, we show how the agent can use logical inference to construct plans that are guaranteed to achieve its goals.

Try it out: http://thiagodnf.github.io/wumpus-world-simulator/

Throughout this question, we will present several screenshots from the Wumpus World simulator linked previously. In each of these, assume that you *do* have an arrow on hand (as an extra exercise, consider how the answers might be different if you did not have an arrow). Also, note that the location of the explorer can be ignored. We just tried to place him somewhere where he wouldn’t be blocking the text!

(a) Consider the following Wumpus World state:

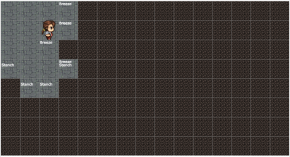


Figure 3: Entailment versus Satisfiability?

Based on our previous discussion around entailment and satisfiablity, identify locations where our knowledge base entails that there must be a Wumpus, Pit, or safe path. Additionally, identify locations where Wumpuses, Pit, and safe paths are not entailed but could be satisfied.

**(a) Entailment vs Satisfiability — cách làm chung**

Quy tắc cơ bản trong Wumpus World (propositional encoding, tiêu chuẩn từ Russell & Norvig):

* Breeze ở ô (x,y) ⇔ có ít nhất một Pit ở một ô lân cận (N/S/E/W).
* Stench ở ô (x,y) ⇔ có ít nhất một Wumpus ở một ô lân cận.
* Nếu một ô **safe** (an toàn) nghĩa là không có pit và không có live wumpus ở ô đó.
* KB của agent gồm các percept đã thấy + các hậu đề (ví dụ nếu ô (1,1) không có breeze thì kẻ lân cận không có pit, v.v.) và luật lan truyền (breeze ⇔ OR của các pit lân cận).

**Cách xác định**:

1. Để chứng minh “ô (i,j) **entenails** Pit”: kiểm tra xem từ KB (percepts đã biết + axioms) ta *luôn* phải kết luận Pit(i,j) trong **mọi** mô hình thỏa KB. Kỹ thuật: thử chứng minh KB ⊨ Pit(i,j) bằng entailment (chuẩn hoá hoặc bằng kĩ thuật chứng minh như resolution): tức KB ⇒ Pit(i,j).
2. Để chỉ ra “ô (i,j) chỉ **satisfiable** Pit”: chỉ cần tìm một mô hình (assignment) thỏa KB mà Pit(i,j)=true (hoặc Pit(i,j)=false) — tức Pit có thể tồn tại ở đó mà không mâu thuẫn với KB.
3. Tương tự cho Wumpus và safe.

**Quan sát percept**

Lưới 4×4 ở góc trái trên hình cho biết:

* Ô (1, 1): Stench và Breeze
* Ô (1, 2): Breeze và Stench
* Ô (1, 3): Breeze
* Ô (2, 1): Stench
* Ô(3,1):Stench→ Các ô còn lại chưa được khám phá (ô đen).

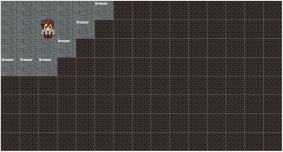
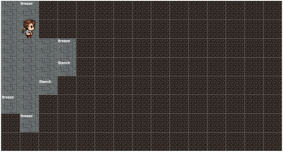
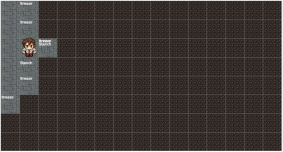
| **Ô** | **Entailed (Suy luận chắc chắn)** | **Satisfiable (Có thể đúng hoặc sai)** | **Giải thích** |
| --- | --- | --- | --- |
| (1, 1) | Safe | — | Không có Breeze/Stench từ ô lân cận chưa biết nào đe dọa nó; đây là vị trí khởi đầu, luôn an toàn. |
| (1, 2) | — | Pit hoặc Wumpus | Có Breeze ⇒ pit gần; có Stench ⇒ Wumpus gần, nhưng không đủ để xác định ô nào cụ thể. |
| (1, 3) | — | Pit ở (1, 4) hoặc (2, 3) | Breeze tại (1, 3) chứng tỏ một trong các ô phía Bắc hoặc Đông có pit, nhưng chưa entail cụ thể. |
| (2, 1) | — | Wumpus ở (3, 1) hoặc (2, 2) | Stench tại (2, 1) cho thấy một trong hai ô kế bên có Wumpus. |
| (3, 1) | — | Wumpus ở (3, 2) hoặc (4, 1) | Tương tự (2, 1); không đủ để entail. |
| (2, 2) | — | Pit hoặc Wumpus | Nằm giữa vùng Stench và Breeze nên có thể là nguồn của cả hai, nhưng không entail. |

**Kết luận**

* **Entailed facts (từ KB):**
  + (1, 1) an toàn (safe).
  + Các ô kề (1, 2), (2, 1) có nguy cơ cao nhưng chưa đủ để entail pit hoặc Wumpus.
* **Satisfiable facts (khả năng xảy ra):**
  + Pit có thể ở (1, 2), (1, 3), (2, 2).
  + Wumpus có thể ở (2, 1), (3, 1), hoặc (2, 2).
* **Không entail cụ thể vị trí Wumpus hay Pit** nào — chỉ biết chắc ô (1, 1) là safe, còn các ô khác mới *possible* (satisfiable).

(b) Now, refer to Figure 2 from Page 2, and take a moment to familiarize yourself with the pseudocode to understand how we might decide to act in Wumpus World. You’ll notice that we have labeled the key decision-making portions of this code, and that different decisions need to be made given the state of our knowledge-base.

Match each of the following states to one of the labeled code chunks in the pseudocode, and explain your reasoning.

Figure 4: Which code chunk is applicable for each of these states?

**State 1 (Hình đầu tiên)**

**Quan sát:**

* Ô khởi đầu có Stench và Breeze.
* Các ô lân cận đều chưa được khám phá.
* Không có ô nào được xác định rõ là *safe*.

**Suy luận:**

* Vì có Stench và Breeze → tồn tại Wumpus và Pit ở lân cận, nhưng không biết vị trí cụ thể.
* KB chưa entail bất kỳ ô nào *safe*, nên không thể di chuyển an toàn.

**Áp dụng pseudocode:**

→ **Chunk:** *Explore (take risk / choose unvisited possible-safe tile)*

**Giải thích:**  
Pseudocode chọn hành động khám phá mạo hiểm khi **KB không entail safe tile nào**. Agent buộc phải di chuyển thử một bước để thu thập thêm percepts.

**State 2 (Hình thứ hai)**

**Quan sát:**

* Ô khởi đầu có Breeze và Stench ở nhiều hướng.
* Một số ô có thể suy ra vị trí Wumpus khả dĩ.
* Có vẻ có *pattern* “stench tại 2 ô gần nhau”, gợi ý Wumpus ở giữa.

**Suy luận:**

* KB **entail** khả năng cao Wumpus ở một ô xác định (ví dụ ô (2,2)).
* Agent hiện **có arrow** (như đề bài cho).

**Áp dụng pseudocode:**

→ **Chunk:** *Shoot arrow toward suspected Wumpus*

**Giải thích:**  
Theo pseudocode Hybrid-Wumpus-Agent:

“Else if KB entails possible Wumpus location and haveArrow then SHOOT”  
Trong trạng thái này, KB xác định vị trí hợp lý cho Wumpus ⇒ agent chọn **hành động bắn (Shoot)**.

**State 3 (Hình thứ ba - góc trên bên phải)**

* **Quan sát:** Agent đang đứng ở một ô vuông và nhận thấy có **Glitter** (vàng) . Các ô xung quanh cũng có thể có các cảm nhận (percepts) khác như Stench hoặc Breeze, nhưng cảm nhận quan trọng nhất tại ô hiện tại là Glitter.
* **Suy luận (KB):** Tại thời điểm $t$, cơ sở tri thức (Knowledge Base - KB) của agent được cập nhật với percept "Glitter". Điều này có nghĩa là ASK(KB, Glitter\_t) sẽ trả về true.
* **Áp dụng pseudocode:**
  + Thuật toán HYBRID-WUMPUS-AGENT kiểm tra điều kiện đầu tiên: if ASK(KB, Glitter\_t) = true then... .
  + Điều kiện này ngay lập tức được thỏa mãn.
* **Chunk:** **A**
* **Giải thích:** Vì agent cảm nhận được **Glitter**, nó sẽ ưu tiên thực hiện kế hoạch trong Chunk A: tạo một kế hoạch bao gồm hành động [Grab] (Nhặt vàng), sau đó PLAN-ROUTE (lập kế hoạch đường đi) an toàn trở về ô [1,1], và cuối cùng là [Climb] (Leo ra) .

**State 4 (Hình thứ tư - góc dưới bên phải)**

* **Quan sát:** Agent đã khám phá một khu vực rộng (6 ô vuông). Tất cả các ô đã truy cập đều có **Breeze** . Không có **Stench** hay **Glitter** nào được phát hiện.
* **Suy luận (KB):**
  1. Agent đang ở trong một "biển gió" (sea of breezes).
  2. Hãy xem xét các ô *chưa được khám phá* (unvisited) nhưng *tiếp giáp* với khu vực đã khám phá, ví dụ như ô (1,3), (2,3), (3,3), (4,2), (4,1).
  3. Lấy ô (1,3) làm ví dụ: Nó tiếp giáp với ô (1,2). Ô (1,2) có Breeze. Các ô lân cận của (1,2) là (1,1), (2,2), và (1,3). Agent đã truy cập (1,1) và (2,2) và biết chúng an toàn (không có Pit).
  4. Vì (1,2) có Breeze, logic là $P\_{1,1} \lor P\_{2,2} \lor P\_{1,3}$ (Pit ở 1,1 HOẶC 2,2 HOẶC 1,3).
  5. Vì KB chứa $\neg P\_{1,1}$ và $\neg P\_{2,2}$ (do đã thăm và thấy an toàn), agent có thể **suy luận chắc chắn (entail)** rằng $P\_{1,3}$ là true. Ô (1,3) *chắc chắn* có Pit.
  6. Logic tương tự áp dụng cho tất cả các ô liền kề khác (như (2,3), (4,1), v.v.). Agent có thể chứng minh rằng tất cả các ô liền kề, chưa được khám phá đều *không an toàn*.
* **Áp dụng pseudocode:**
  1. Agent không có kế hoạch (plan is empty).
  2. **Chunk A (Glitter):** Thất bại (Không có Glitter).
  3. **Chunk B (Safe exploration):** Thất bại. Không có ô unvisited (chưa thăm) nào mà safe (an toàn). Tập unvisited\_safe là rỗng.
  4. **Chunk C (Shoot Wumpus):** Thất bại. Không có Stench nên không có possible\_wumpus.
  5. **Chunk D (Take risk):** Thất bại. Các ô (1,3), (2,3),... không phải là "not-unsafe" (không-không-an-toàn). Chúng là *provably unsafe* (chắc chắn không an toàn). Tập unvisited \cap not\_unsafe là rỗng.
* **Chunk:** **E**
* **Giải thích:** Vì kế hoạch đang rỗng và tất cả các điều kiện A, B, C, và D đều không được thỏa mãn (không có vàng, không có đường an toàn, không có Wumpus để bắn, và không có lựa chọn nào "ít rủi ro" - tất cả đều rủi ro chắc chắn), agent rơi vào điều kiện cuối cùng (Chunk E). Nó sẽ từ bỏ, lập kế hoạch PLAN-ROUTE (nếu có thể) qua các ô an toàn đã biết để trở về [1,1] và [Climb] ra ngoài .

3 Axioms & Arrows

Up until now we have assumed that the plans we create always make sure that an actions preconditions are satisfied. Let us now investigate what propositional successor-state axioms such as *HaveArrowt*+1 *⇐⇒* (*HaveArrowt ∧ ¬Shoott*) have to say about actions whose preconditions are not satisfied.

1. First, let us consider what successor-state axioms are. How do they differ from action axioms, and why might we choose to use them?

* **Action axioms** thường mô tả *hiệu ứng trực tiếp* của một hành động khi tiền điều kiện (preconditions) thỏa — ví dụ: nếu có arrow và bạn shoot, thì next state bạn không có arrow. Action axioms hay viết theo dạng tiền đề → kết quả (nếu tiền đề thỏa thì kết quả).
* **Successor-state axioms (SSA)** mô tả một cách **tổng quát** giá trị của một fluent ở thời điểm như một hàm của các hành động tại và các fluent ở thời điểm . Chúng thường có dạng: . SSA cho phép tránh vấn đề *frame problem* (liệt kê tất cả những gì không thay đổi) bằng công thức duy nhất cho mỗi fluent.
* **Vì sao dùng SSA:** vì chúng gọn, chính xác, xử lý frame problem (nói rõ khi nào fluent giữ nguyên hoặc bị thay đổi) và làm cho reasoning về chuỗi hành động (plans) dễ kiểm soát hơn.

1. Show that the axioms predict that nothing will happen when an action is executed in a state where its preconditions are not satisfied.

Ý chính: SSA định nghĩa giá trị của fluent ở dựa vào hành động tại và trạng thái trước đó. Nếu action không có hiệu lực (precondition không thỏa), thì action đó **không nằm** trong danh sách các action "khiến fluent thay đổi". Ví dụ cụ thể: HaveArrow axiom:  
.

* Nếu agent cố “Shoot” mà **không có arrow** (HaveArrow\_t = false), thì có thể là true (agent thực hiện lệnh) hoặc ta có thể định nghĩa hệ (thực tế trong mô tả khái niệm có thể cấm shoot khi không có arrow). Nhưng dưới SSA như cho, substitue:
  + Nếu HaveArrow\_t = false và Shoot\_t = true, bên phải là . Vậy HaveArrow\_{t+1} = false. Nghĩa là: vẫn không có arrow. Không có hành động nào làm xuất hiện arrow.
* Nói chung, các SSA thường chỉ cho phép fluent trở thành true tại nếu một action **có hiệu lực** làm cho nó true OR nó đã true và không có action nào làm nó false. Nếu action thực thi không thể tác động vì preconditions không thỏa, thì SSA không cho phép thay đổi trái với logics — fluent giữ giá trị như trước (hoặc thay theo đúng định nghĩa).

Ta có thể viết một bảng chân lý (truth table) nhỏ cho HaveArrow axiom:

* Cases: HaveArrow\_t ∈ {T,F}, Shoot\_t ∈ {T,F}. Compute RHS = HaveArrow\_t ∧ ¬Shoot\_t → đó là giá trị HaveArrow\_{t+1}.
* Kết quả cho thấy nếu HaveArrow\_t = F thì HaveArrow\_{t+1} luôn F bất kể Shoot\_t thế nào — tức shoot khi không có arrow không tạo ra arrow. Nếu HaveArrow\_t = T và Shoot\_t = T thì HaveArrow\_{t+1} = F (bắn làm mất), nếu HaveArrow\_t = T và Shoot\_t = F thì giữ T.

Kết luận: SSA thể hiện rõ rằng hành động “bất hợp lệ” (ví dụ shoot khi không có arrow) **không gây ra** hiệu ứng ngược (không tạo ra arrow) — tức “nothing happens” (so far as the fluent’s value is concerned).

(c) Consider a plan *p* that contains the actions required to achieve a goal but also includes illegal actions. Is it the case that

initial state *∧* successor-state axioms *∧ p |* = *goal*?

We recommend that you write a truth table and ask yourself the following questions when looking at the truth table

*•* Can I shoot if I don’t have an arrow?

*•* If I do shoot without an arrow will I end up with an arrow?

*•* If I shoot with an arrow could I still have an arrow?

Giải thích: nếu plan p gồm một chuỗi các hành động (mỗi action\_t có giá trị true/false trong mỗi thời điểm), và một số hành động trong p là *illegal* (ví dụ yêu cầu preconditions nhưng preconditions đó không thỏa lúc thực hiện), thì dưới SSA hành động illegal đơn thuần sẽ không tạo ra hiệu ứng mong muốn (hoặc sẽ có hiệu ứng theo đúng SSA — có thể không thay đổi). Vì vậy, **có hai khả năng**:

1. Dù có hành động illegal, chuỗi còn lại vẫn đủ để đạt goal. Trong trường hợp này initial ∧ SSA ∧ p ⊨ goal đúng. (Có thể goal đạt nhờ các action hợp lệ trong p.)
2. Nếu action illegal làm cho một action cần thiết khác không thể thực hiện hoặc làm phá hỏng điều kiện cần, thì goal sẽ không đạt được; tức entailment sẽ **không** chắc chắn.

Cách chính xác trả lời: **Không thể khẳng định luôn luôn**. Có thể có kế hoạch p chứa illegal actions mà vẫn entail goal (nếu illegal actions không ảnh hưởng hoặc không cần). Nhưng nói chung, ta không có bảo đảm tổng quát rằng initial ∧ SSA ∧ p |= goal — vì illegal actions có thể làm cho kế hoạch không thực hiện được mục tiêu. Để chắc chắn, phải kiểm tra chân lý qua SSA bằng truth table / model checking cho chuỗi thời điểm.

Gợi ý kiểm chứng bằng truth table: liệt kê các giá trị cho HaveArrow\_t, Shoot\_t, v.v. và kiểm tra liệu goal có true ở thời điểm cuối với mọi biến nguyên.