Knowledge Representation and Reasoning

Lecture 3: Query Language

dr Anna Maria Radzikowska
Warsaw University of Technology
Faculty of Mathematics and Information Science
E–mail: annrad@mini.pw.edu.pl

Project is co-financed by European Union within European Social Fund
Statements:

- **value statement:** $\alpha$ after $A_1, \ldots, A_n$
- **observation statement:** observable $\alpha$ after $A_1, \ldots, A_n$
- **effect statement:** $A$ causes $\alpha$ if $\pi$
- **release statement:** $A$ releases $f$ if $\pi$
- **constraint statement:** always $\alpha$. 
Action Language $\mathcal{AR}$ – recall...

Statements:

- **value statement**: $\alpha$ after $A_1, \ldots, A_n$
- **observation statement**: observable $\alpha$ after $A_1, \ldots, A_n$
- **effect statement**: $A$ causes $\alpha$ if $\pi$
- **release statement**: $A$ releases $f$ if $\pi$
- **constraint statement**: always $\alpha$.

A **structure** for a language $\mathcal{L}$ of the class $\mathcal{AR}$ is a triple $S = (\Sigma, \sigma_0, Res)$, where

- $\Sigma$ is a set of states
- $\sigma_0 \in \Sigma$ is the initial state
- $Res : Ac \times \Sigma \rightarrow 2^\Sigma$ is a transition function.
A structure $S = (\Sigma, \sigma_0, Res)$ is a model of an action domain $D$ iff

1. (M.1) $\Sigma$ is the set of all states for $D$
2. (M.2) every value statement and every observation statement is true in $S$
3. (M.3) for any $A \in Ac$ and for any $\sigma \in \Sigma$, $Res(A, \sigma)$ is the set of all states $\sigma' \in \Sigma$ for which the sets $New(A, \sigma, \sigma')$ are minimal (wrt set inclusion).

Recall that $New(A, \sigma, \sigma')$ is the set of literals $\overline{f}$ that hold in $\sigma'$ and

- $f$ is inertial and $\sigma(f) \neq \sigma'(f)$, or
- there is a statement $A$ releases $f$ if $\pi$ in $D$ such that $\sigma \models \pi$.  

Example 3.1: Two switches

Recall:

There are two switches, which can be in the position on or off. If both switches are in the same position, the light is on, otherwise it is off. Pressing any switch changes its position.

Representation in $\mathcal{KR}$:

\begin{align*}
\text{noninertial} & \text{ light;} \\
\text{initially} & \text{ switch}_1 \land \text{ switch}_2; \\
\text{always} & \text{ light } \equiv (\text{ switch}_1 \equiv \text{ switch}_2); \\
T\text{OGGLE1} & \text{ causes } \neg\text{ switch}_1 \text{ if switch}_1; \\
T\text{OGGLE1} & \text{ causes } \text{ switch}_1 \text{ if } \neg\text{ switch}_1; \\
T\text{OGGLE2} & \text{ causes } \neg\text{ switch}_2 \text{ if switch}_2; \\
T\text{OGGLE2} & \text{ causes } \text{ switch}_2 \text{ if } \neg\text{ switch}_2.
\end{align*}
Example 3.1: Two switches (cont.)

Here we have \( \Sigma = \{\sigma_0, \sigma_1, \sigma_2, \sigma_3\} \), where

\[
\begin{align*}
\sigma_0 &= \{\text{switch}_1, \text{switch}_2, \text{light}\} \\
\sigma_1 &= \{\neg \text{switch}_1, \neg \text{switch}_2, \text{light}\} \\
\sigma_2 &= \{\text{switch}_1, \neg \text{switch}_2, \neg \text{light}\} \\
\sigma_3 &= \{\neg \text{switch}_1, \text{switch}_2, \neg \text{light}\}.
\end{align*}
\]

Assume that all fluents are inertial. Then we have:

\[
\begin{align*}
\text{Res}_0(\text{TOGGLE1}, \sigma_0) &= \{\sigma_1, \sigma_3\} \\
\text{New}(\text{TOGGLE1}, \sigma_0, \sigma_1) &= \{\neg \text{switch}_1, \neg \text{switch}_2\} \\
\text{New}(\text{TOGGLE1}, \sigma_0, \sigma_3) &= \{\neg \text{switch}_1, \neg \text{light}\} \\
\text{Res}(\text{TOGGLE1}, \sigma_0) &= \{\sigma_1, \sigma_3\}.
\end{align*}
\]

So \text{TOGGLE1} in \( \sigma_0 \) is non–deterministic, but the resulting state \( \sigma_1 \) is counterintuitive!
Example 3.1: Two switches (cont.)

\[ \sigma_0 = \{ \text{switch}_1, \text{switch}_2, \text{light} \} \]
\[ \sigma_1 = \{ \neg \text{switch}_1, \neg \text{switch}_2, \text{light} \} \]
\[ \sigma_2 = \{ \text{switch}_1, \neg \text{switch}_2, \neg \text{light} \} \]
\[ \sigma_3 = \{ \neg \text{switch}_1, \text{switch}_2, \neg \text{light} \}. \]

Similarly,

\[ \text{Res}_0(\text{TOGGLE}_1, \sigma_1) = \{ \sigma_0, \sigma_2 \} \]
\[ \ast \quad \text{New}(\text{TOGGLE}_1, \sigma_1, \sigma_0) = \{ \text{switch}_1, \text{switch}_2 \} \]
\[ \ast \quad \text{New}(\text{TOGGLE}_1, \sigma_1, \sigma_2) = \{ \text{switch}_1, \neg \text{light} \} \]
\[ \text{Res}(\text{TOGGLE}_1, \sigma_1) = \{ \sigma_0, \sigma_2 \}. \]
Example 3.1: Two switches (cont.)

\[
\begin{align*}
\sigma_0 &= \{ \text{switch}_1, \text{switch}_2, \text{light} \} \\
\sigma_1 &= \{ \neg\text{switch}_1, \neg\text{switch}_2, \text{light} \} \\
\sigma_2 &= \{ \text{switch}_1, \neg\text{switch}_2, \neg\text{light} \} \\
\sigma_3 &= \{ \neg\text{switch}_1, \text{switch}_2, \neg\text{light} \}.
\end{align*}
\]

Similarly,

\[
\begin{align*}
Res_0(\text{TOGGLE}_1, \sigma_1) &= \{ \sigma_0, \sigma_2 \} \\
Res(\text{TOGGLE}_1, \sigma_1) &= \{ \sigma_0, \sigma_2 \}.
\end{align*}
\]

Note that only the resulting state \(\sigma_2\) coincides with our intuition.
Example 3.1: Two switches (cont.)

\[
\begin{align*}
\sigma_0 &= \{\text{switch}_1, \text{switch}_2, \text{light}\} & \sigma_2 &= \{\text{switch}_1, \neg\text{switch}_2, \neg\text{light}\} \\
\sigma_1 &= \{\neg\text{switch}_1, \neg\text{switch}_2, \text{light}\} & \sigma_3 &= \{\neg\text{switch}_1, \text{switch}_2, \neg\text{light}\}.
\end{align*}
\]

Now let the fluent \textit{light} be \textbf{noninertial}. Then

\[
\begin{align*}
\text{Res}_0(\text{TOGGLE}_1, \sigma_0) &= \{\sigma_1, \sigma_3\} \\
\text{New}(\text{TOGGLE}_1, \sigma_0, \sigma_1) &= \{\neg\text{switch}_1, \neg\text{switch}_2\} \\
&\ast \quad \text{New}(\text{TOGGLE}_1, \sigma_0, \sigma_3) = \{\neg\text{switch}_1\} \\
\text{Res}(\text{TOGGLE}_1, \sigma_0) &= \{\sigma_3\}.
\end{align*}
\]

Although \(\sigma_0(\text{light}) \neq \sigma_3(\text{light})\), we have \(\text{light} \notin \mathcal{F}_I\) (noninertial), so \(\neg\text{light} \notin \text{New}(\text{TOGGLE}_1, \sigma_0, \sigma_3)\).
Example 3.1: Two switches (cont.)

\[
\begin{align*}
\sigma_0 &= \{ \text{switch}_1, \text{switch}_2, \text{light} \} \\
\sigma_1 &= \{ \neg \text{switch}_1, \neg \text{switch}_2, \text{light} \} \\
\sigma_2 &= \{ \text{switch}_1, \neg \text{switch}_2, \neg \text{light} \} \\
\sigma_3 &= \{ \neg \text{switch}_1, \text{switch}_2, \neg \text{light} \}.
\end{align*}
\]

Also,

\[
\begin{align*}
Res_0(\text{TOGGLE}_1, \sigma_1) &= \{ \sigma_0, \sigma_2 \} \\
New(\text{TOGGLE}_1, \sigma_1, \sigma_0) &= \{ \text{switch}_1, \text{switch}_2 \} \\
* \quad New(, \sigma_1, \sigma_2) &= \{ \text{switch}_1 \} \\
Res(\text{TOGGLE}_1, \sigma_1) &= \{ \sigma_2 \}.
\end{align*}
\]
Example 3.1: Two switches (cont.)

- $switch_1, switch_2, light$
- $\lnot switch_1, \lnot switch_2, \lnot light$

Toggle1

- $switch_1, \lnot switch_2, \lnot light$
- $\lnot switch_1, \lnot switch_2, light$

Toggle2

Toggle1

Toggle2
Example 3.2: Buying a paper

Consider an action \( \text{Buy} \) which causes that an agent has a paper \( A \) (\( \text{has}A \)) or a paper \( B \) (\( \text{has}B \)).

\( \text{Buy} \) causes \( \text{has}A \lor \text{has}B \).

Put

\[
\begin{align*}
\sigma_1 &= \{ \neg \text{has}A, \neg \text{has}B \} \\
\sigma_2 &= \{ \neg \text{has}A, \text{has}B \} \\
\sigma_3 &= \{ \text{has}A, \neg \text{has}B \} \\
\sigma_4 &= \{ \text{has}A, \text{has}B \}.
\end{align*}
\]

Then

\[
\begin{align*}
\text{Res}_0(\text{Buy}, \sigma_1) &= \{ \sigma_2, \sigma_3, \sigma_4 \} \\
\text{New}(\text{Buy}, \sigma_1, \sigma_2) &= \{ \text{has}B \} \\
\text{New}(\text{Buy}, \sigma_1, \sigma_3) &= \{ \text{has}A \} \\
\text{New}(\text{Buy}, \sigma_1, \sigma_4) &= \{ \text{has}A, \text{has}B \} \\
\text{Res}(\text{Buy}, \sigma_1) &= \{ \sigma_2, \sigma_3 \}
\end{align*}
\]
Example 3.2: Buying a paper (cont.)

Problem:

*How to obtain the effect of this action such that an agent has both papers?*
Example 3.2: Buying a paper (cont.)

Problem:

How to obtain the effect of this action such that an agent has both papers?

A possible solution:

\[
\text{BUY } \text{causes } \text{hasA } \lor \text{hasB}; \\
\text{BUY } \text{releases } \text{hasA if } \neg \text{hasA}; \\
\text{BUY } \text{releases } \text{hasB if } \neg \text{hasB};
\]
Example 3.2: Buying a paper (cont.)

\[ \sigma_1 = \{ \neg \text{hasA}, \neg \text{hasB} \} \]
\[ \sigma_2 = \{ \neg \text{hasA}, \text{hasB} \} \]
\[ \sigma_3 = \{ \text{hasA}, \neg \text{hasB} \} \]
\[ \sigma_4 = \{ \text{hasA}, \text{hasB} \} \].

So we get:

\[ \text{Res}_0(\text{Buy, } \sigma_1) = \{ \sigma_2, \sigma_3, \sigma_4 \} \]
\[ \times \text{New}(\text{Buy, } \sigma_1, \sigma_2) = \{ \neg \text{hasA}, \text{hasB} \} \]
\[ \times \text{New}(\text{Buy, } \sigma_1, \sigma_3) = \{ \text{hasA}, \neg \text{hasB} \} \]
\[ \times \text{New}(\text{Buy, } \sigma_1, \sigma_4) = \{ \text{hasA}, \text{hasB} \} \]

\[ \text{Res}(\text{Buy, } \sigma_1) = \{ \sigma_2, \sigma_3, \sigma_4 \}. \]
Queries
Queries in $\mathcal{AR}$

By a *query* of $\mathcal{AR}$ we mean any expression of the form:

- **value queries:**

  - *necessary* $\alpha$ *after* $A_1, \ldots, A_n$ *from* $\pi$
  - *possibly* $\alpha$ *after* $A_1, \ldots, A_n$ *from* $\pi$

The 1st query states that $\alpha$ *ALWAYS* holds after performing the sequence $A_1, \ldots, A_n$ of actions from any state satisfying $\pi$.

The 2nd query says that $\alpha$ *SOMETIMES* holds after executing $A_1, \ldots, A_n$ from any state satisfying $\pi$.

When the option *from* $\pi$ is omitted, these queries refer to the initial state.
**Basic queries (cont.)**

- **executability queries**:

  \[
  \text{necessary executable } A_1, \ldots, A_n \text{ from } \pi \\
  \text{possibly executable } A_1, \ldots, A_n \text{ from } \pi
  \]

  Intuitively, the first query states that the sequence \((A_1, \ldots, A_n)\) is *always* executable from any state satisfying \(\pi\), while the second one says that \((A_1, \ldots, A_n)\) *may be* executed from any state where \(\pi\) holds.

  Again, if the phrase *from \(\pi\)* is omitted, then the initial state is to be taken into account.
Basic queries (cont.)

- **accessibility query:**

  \[ \text{necessary accessible } \gamma \text{ from } \pi \]

  \[ \text{possibly accessible } \gamma \text{ from } \pi \]

  The first query states that the goal $\gamma$ is always achieved from any state satisfying $\pi$, whereas the second query says that $\gamma$ may be achieved from any state where $\pi$ holds.

  If the phrase \textit{from} $\pi$ is omitted, then it refers to the initial state.
Basic queries (cont.)

- **accessibility query:**

  - necessary accessible $\gamma$ from $\pi$
  - possibly accessible $\gamma$ from $\pi$

The first query states that the goal $\gamma$ is always achieved from any state satisfying $\pi$, whereas the second query says that $\gamma$ may be achieved from any state where $\pi$ holds.

If the phrase *from* $\pi$ is omitted, then it refers to the initial state.

**Underlying intuition:**

There exists a sequence $(A_1, \ldots, A_n)$, $n \geq 0$, of actions, executable from any state satisfying $\pi$ (resp. the initial state), which leads to states satisfying $\gamma$. Various sequences of actions, starting in various initial states, may lead to various states, where $\gamma$ holds.
**goal queries**: 

\[
necessary \text{ goal } \gamma \text{ from } \pi \]
\[
necessary \text{ goal } \gamma \text{ from } \pi .
\]

Intuitively, the first query returns the sequence \((A_1, \ldots, A_n)\) of actions, executable from any state satisfying \(\pi\) (if such a sequence exists at all), which always leads to states where the goal condition \(\gamma\) holds. Similarly, the second query returns the sequence \((A_1, \ldots, A_n)\) of actions, executable from any state satisfying \(\pi\) (if such a sequence exists at all), which may lead to a state satisfying the goal condition \(\gamma\).

When the goal is to be achieved from the initial state, the phrase *from \(\pi\)* will be omitted.
Satisfiability

Let $D$ be an action domain and let $Q$ be a query of $\mathcal{AR}$. We say that $Q$ is a consequence of $D$, in symbols $D \models Q$, iff

- if $Q$ is a value query of the form $\textit{necessary } \alpha \textit{ after } A_1, \ldots, A_n \textit{ from } \pi$,

then $D \models Q$ iff for every model $S = (\Sigma, \sigma_0, \text{Res})$ of $D$, for every state $\sigma \in \Sigma$ such that $\sigma \models \pi$, and for every mapping $\Psi_S$, if $\Psi_S((A_1, \ldots, A_n), \sigma)$ is defined, then $\Psi_S((A_1, \ldots, A_n), \sigma) \models \alpha$
Let $D$ be an action domain and let $Q$ be a query of $\mathcal{AR}$. We say that $Q$ is a consequence of $D$, in symbols $D \models Q$, iff

- if $Q$ is a value query of the form

  $$\text{necessary } \alpha \text{ after } A_1, \ldots, A_n \text{ from } \pi,$$

  then $D \models Q$ iff for every model $S = (\Sigma, \sigma_0, Res)$ of $D$, for every state $\sigma \in \Sigma$ such that $\sigma \models \pi$, and for every mapping $\Psi_S$, if $\Psi_S((A_1, \ldots, A_n), \sigma)$ is defined, then $\Psi_S((A_1, \ldots, A_n), \sigma) \models \alpha$

- if $Q$ is a value query

  $$\text{necessary } \alpha \text{ after } A_1, \ldots, A_n,$$

  then $D \models Q$ iff for every model $S = (\Sigma, \sigma_0, Res)$ of $D$, and for every mapping $\Psi_S$, if $\Psi_S((A_1, \ldots, A_n), \sigma)$ is defined, then $\Psi_S((A_1, \ldots, A_n), \sigma_0) \models \alpha$. 
if $Q$ is a value query of the form

$$\textit{possibly } \alpha \textit{ after } A_1, \ldots, A_n \textit{ from } \pi,$$

then $D \models Q$ iff for every model $S = (\Sigma, \sigma_0, Res)$ of $D$, for every state $\sigma \in \Sigma$, if $\sigma \models \pi$, then there exists a mapping $\Psi_S$ such that $\Psi_S((A_1, \ldots, A_n), \sigma)$ is defined and $\Psi_S((A_1, \ldots, A_n), \sigma) \models \alpha$. 
Satisfiability (cont.)

- if $Q$ is a value query of the form

$$\textit{possibly } \alpha \textit{ after } A_1, \ldots, A_n \textit{ from } \pi,$$

then $D \models Q$ iff for every model $S = (\Sigma, \sigma_0, \text{Res})$ of $D$, for every state $\sigma \in \Sigma$, if $\sigma \models \pi$, then there exists a mapping $\Psi_S$ such that $\Psi_S((A_1, \ldots, A_n), \sigma)$ is defined and $\Psi_S((A_1, \ldots, A_n), \sigma) \models \alpha$.

- if $Q$ is a value query of the form

$$\textit{possibly } \alpha \textit{ after } A_1, \ldots, A_n,$$

then $D \models Q$ iff for every model $S = (\Sigma, \sigma_0, \text{Res})$ of $D$, there exists a mapping $\Psi_S$ such that $\Psi_S((A_1, \ldots, A_n), \sigma)$ is defined and $\Psi_S((A_1, \ldots, A_n), \sigma_0) \models \alpha$. 
Example 3.3: Tossing a coin

Let an action domain $D$ be as follows:

- **initially** heads

  Toss *releases* heads.

Then

\[
D \models \text{possibly } \neg \text{heads after Toss}
\]

\[
D \models \text{possibly } \text{heads after Toss}
\]

\[
D \not\models \text{necessary } \neg \text{head after Toss}
\]

\[
D \not\models \text{necessary } \text{head after Toss}
\]
if $Q$ is an executability query of the form

\[
\text{necessary executable } A_1, \ldots, A_n \text{ from } \pi,
\]

then $D \models Q$ iff for every model $S = (\Sigma, \sigma_0, Res)$ of $D$, for every state $\sigma \in \Sigma$ such that $\sigma \models \pi$, and for every mapping $\Psi_S$, $\Psi_S((A_1, \ldots, A_n), \sigma)$ is defined.
if $Q$ is an executability query of the form

$$\textit{necessary executable } A_1, \ldots, A_n \textit{ from } \pi,$$

then $D \models Q$ iff for every model $S = (\Sigma, \sigma_0, Res)$ of $D$, for every state $\sigma \in \Sigma$ such that $\sigma \models \pi$, and for every mapping $\Psi_S$, $\Psi_S((A_1, \ldots, A_n), \sigma)$ is defined.

if $Q$ is an executability query of the form

$$\textit{possibly executable } A_1, \ldots, A_n \textit{ from } \pi,$$

then $D \models Q$ iff for every model $S = (\Sigma, \sigma_0, Res)$ of $D$, for every state $\sigma \in \Sigma$ such that $\sigma \models \pi$, there is some mapping $\Psi_S$ such that $\Psi_S((A_1, \ldots, A_n), \sigma)$ is defined.
Satisfiability (cont.)

- if $Q$ is an executability query of the form

  \[ \text{necessary executable } A_1, \ldots, A_n, \]

  then $D \models Q$ iff for every model $S = (\Sigma, \sigma_0, \text{Res})$ of $D$, and for every mapping $\Psi_S$, $\Psi_S((A_1, \ldots, A_n), \sigma)$ is defined.
Satisfiability (cont.)

- if $Q$ is an executability query of the form
  
  $$\text{necessary executable } A_1, \ldots, A_n,$$
  
  then $D \models Q$ iff for every model $S = (\Sigma, \sigma_0, Res)$ of $D$, and for every mapping $\Psi_S$, $\Psi_S((A_1, \ldots, A_n), \sigma)$ is defined.

- if $Q$ is an executability query of the form
  
  $$\text{possibly executable } A_1, \ldots, A_n,$$
  
  then $D \models Q$ iff for every model $S = (\Sigma, \sigma_0, Res)$ of $D$, there is some mapping $\Psi_S$ such that $\Psi_S((A_1, \ldots, A_n), \sigma)$ is defined.
if $Q$ is an accessibility query of the form

\[ \text{necessary accessible } \gamma \text{ from } \pi, \]

then $D \models Q$ iff for every model $S = (\Sigma, \sigma_0, Res)$ of $D$ and for every state $\sigma \in \Sigma$, if $\sigma \models \pi$, then there exists a sequence $(A_1, \ldots, A_n)$ of actions, $n \geq 0$, such that for every mapping $\Psi_S$, $\Psi_S((A_1, \ldots, A_n), \sigma)$ is defined and $\Psi_S((A_1, \ldots, A_n), \sigma) \models \gamma$. 
Satisfiability (cont.)

if $Q$ is an accessibility query of the form

$\text{necessary accessible } \gamma \text{ from } \pi$,

then $D \models Q$ iff for every model $S = (\Sigma, \sigma_0, Res)$ of $D$ and for every state $\sigma \in \Sigma$, if $\sigma \models \pi$, then there exists a sequence $(A_1, \ldots, A_n)$ of actions, $n \geq 0$, such that for every mapping $\Psi_S$, $\Psi_S((A_1, \ldots, A_n), \sigma)$ is defined and $\Psi_S((A_1, \ldots, A_n), \sigma) \models \gamma$.

if $Q$ is an accessibility query of the form

$\text{necessary accessible } \gamma \text{ from } \pi$,

then $D \models Q$ iff for every model $S = (\Sigma, \sigma_0, Res)$ of $D$ and for every state $\sigma \in \Sigma$, if $\sigma \models \pi$, then there exists a sequence $(A_1, \ldots, A_n)$ of actions, $n \geq 0$, such that for some mapping $\Psi_S$, $\Psi_S((A_1, \ldots, A_n), \sigma)$ is defined and $\Psi_S((A_1, \ldots, A_n), \sigma) \models \gamma$. 
if $Q$ is an accessibility query of the form

\[ \text{necessary accessible } \gamma, \]

then $D \models Q$ iff for every model $S = (\Sigma, \sigma_0, Res)$ of $D$ there exists a sequence $(A_1, \ldots, A_n)$ of actions, $n \geq 0$, such that for every mapping $\Psi_S$, $\Psi_S((A_1, \ldots, A_n), \sigma_0)$ is defined and $\Psi_S((A_1, \ldots, A_n), \sigma_0) \models \gamma$. 
Satisfiability (cont.)

- if $Q$ is an accessibility query of the form
  
  \textit{necessary accessible} $\gamma$,

  then $D \models Q$ iff for every model $S = (\Sigma, \sigma_0, Res)$ of $D$ there exists a sequence $(A_1, \ldots, A_n)$ of actions, $n \geq 0$, such that for every mapping $\Psi_S$, $\Psi_S((A_1, \ldots, A_n), \sigma_0)$ is defined and $\Psi_S((A_1, \ldots, A_n), \sigma_0) \models \gamma$.

- if $Q$ is an accessibility query of the form
  
  \textit{possibly accessible} $\gamma$,

  then $D \models Q$ iff for every model $S = (\Sigma, \sigma_0, Res)$ of $D$ there exists a sequence $(A_1, \ldots, A_n)$ of actions, $n \geq 0$, such that for some mapping $\Psi_S$, $\Psi_S((A_1, \ldots, A_n), \sigma_0)$ is defined and $\Psi_S((A_1, \ldots, A_n), \sigma_0) \models \gamma$. 
Consider an action domain:

A *causes* $p \lor q$;

B *causes* $\neg r$;

*impossible B if* $\neg q$. 
Example 3.4

Consider an action domain:

- \( A \) causes \( p \lor q \);
- \( B \) causes \( \neg r \);
- \textit{impossible} \( B \) if \( \neg q \).

[Diagram showing the relationships between actions A and B and their effects on p, q, and r.]
Example 3.4 (cont.)
Example 3.4 (cont.)

\[ D \not\models \textit{necessary accessible} \lnot r \textit{ from } \lnot p \]

\[ D \models \textit{possibly accessible} \lnot r \textit{ from } \lnot p. \]
Example 3.4 (cont.)

\[ \overline{D} \not \models \text{necessary accessible } \overline{\overline{r}} \text{ from } \overline{\overline{p}} \]

\[ \overline{D} \models \text{possibly accessible } \overline{\overline{r}} \text{ from } \overline{\overline{p}}. \]

\textit{Why?}
Example 3.4 (cont.)

\[D \nvDash necessary accessible \ \neg r \ from \ \neg p\]
\[D \vDash possibly accessible \ \neg r \ from \ \neg p.\]

**WHY?**

Because performing the sequence \((A, B)\) in the state \((\neg p, \neg q, r)\) may (but not need) lead to the state \((\neg p, q, \neg r)\).
Satisfiability (cont.)

if $Q$ is a goal query of the form

$necessary\ goal\ \gamma\ from\ \pi$

then $D \models Q$ iff there exists a sequence $(A_1, \ldots, A_n)$ of actions, $n \geq 0$, such that for every model $S = (\Sigma, \sigma_0, Res)$ of $D$, if $\sigma \models \pi$ for any $\sigma \in \Sigma$, then for every mapping $\Psi_S((A_1, \ldots, A_n), \sigma)$ $\Psi_S((A_1, \ldots, A_n), \sigma)$ is defined and $\Psi_S((A_1, \ldots, A_n), \sigma) \models \gamma$. 
Satisfiability (cont.)

- if $Q$ is a goal query of the form
  
  **necessery goal** $\gamma$ from $\pi$

  then $D \models Q$ iff there exists a sequence $(A_1, \ldots, A_n)$ of actions, $n \geq 0$, such that for every model $S = (\Sigma, \sigma_0, Res)$ of $D$, if $\sigma \models \pi$ for any $\sigma \in \Sigma$, then for every mapping $\Psi_S((A_1, \ldots, A_n), \sigma)$ $\Psi_S((A_1, \ldots, A_n), \sigma)$ is defined and $\Psi_S((A_1, \ldots, A_n), \sigma) \models \gamma$.

- if $Q$ is of the form
  
  **possibly goal** $\gamma$ from $\pi$

  then $D \models Q$ iff there exists a sequence $(A_1, \ldots, A_n)$ of actions such that for every model $S = (\Sigma, \sigma_0, Res)$ of $D$, if $\sigma \models \pi$ for any $\sigma \in \Sigma$, then there is a mapping $\Psi_S((A_1, \ldots, A_n), \sigma)$ such that $\Psi_S((A_1, \ldots, A_n), \sigma)$ is defined and $\Psi_S((A_1, \ldots, A_n), \sigma) \models \gamma$. 
if \( Q \) is a goal query of the form

\[ \text{necessary goal } \gamma \text{ from } \pi \]

then \( D \models Q \) iff there exists a sequence \((A_1, \ldots, A_n)\) of actions, \( n \geq 0 \), such that for every model \( S = (\Sigma, \sigma_0, Res) \) of \( D \), if \( \sigma \models \pi \) for any \( \sigma \in \Sigma \), then for every mapping \( \Psi_S((A_1, \ldots, A_n), \sigma) \) is defined and \( \Psi_S((A_1, \ldots, A_n), \sigma) \models \gamma \).

if \( Q \) is of the form

\[ \text{possibly goal } \gamma \text{ from } \pi \]

then \( D \models Q \) iff there exists a sequence \((A_1, \ldots, A_n)\) of actions such that for every model \( S = (\Sigma, \sigma_0, Res) \) of \( D \), if \( \sigma \models \pi \) for any \( \sigma \in \Sigma \), then there is a mapping \( \Psi_S((A_1, \ldots, A_n), \sigma) \) such that \( \Psi_S((A_1, \ldots, A_n), \sigma) \) is defined and \( \Psi_S((A_1, \ldots, A_n), \sigma) \models \gamma \).

If \( D \models Q \), then the sequence \((A_1, \ldots, A_n)\) is called an answer for the query \( Q \) wrt \( D \).
if $Q$ is a goal query of the form

*necessary goal* $\gamma$

then $D \models Q$ iff there exists a sequence $(A_1, \ldots, A_n)$ of actions such that for every model $S = (\Sigma, \sigma_0, Res)$ of $D$ and for every mapping $\Psi_S$, $\Psi_S((A_1, \ldots, A_n), \sigma_0)$ is defined and $\Psi_S((A_1, \ldots, A_n), \sigma_0) \models \gamma$. 
if $Q$ is a goal query of the form

**necessary goal** $\gamma$

then $D \models Q$ iff there exists a sequence $(A_1, \ldots, A_n)$ of actions such that for every model $S = (\Sigma, \sigma_0, Res)$ of $D$ and for every mapping $\Psi_S$, $\Psi_S((A_1, \ldots, A_n), \sigma_0)$ is defined and $\Psi_S((A_1, \ldots, A_n), \sigma_0) \models \gamma$.

if $Q$ is a goal query of the form

**possibly goal** $\gamma$

then $D \models Q$ iff for every model $S = (\Sigma, \sigma_0, Res)$ of $D$ there is a mapping $\Psi_S$ such that $\Psi_S((A_1, \ldots, A_n), \sigma_0)$ is defined and $\Psi_S((A_1, \ldots, A_n), \sigma_0) \models \gamma$. 
Satisfiability (cont.)

- if $Q$ is a goal query of the form
  
  \textit{necessary goal } $\gamma$

  then $D \models Q$ iff there exists a sequence $(A_1, \ldots, A_n)$ of actions such that for every model $S = (\Sigma, \sigma_0, \text{Res})$ of $D$ and for every mapping $\Psi_S$, $\Psi_S((A_1, \ldots, A_n), \sigma_0)$ is defined and $\Psi_S((A_1, \ldots, A_n), \sigma_0) \models \gamma$.

- if $Q$ is a goal query of the form
  
  \textit{possibly goal } $\gamma$

  then $D \models Q$ iff for every model $S = (\Sigma, \sigma_0, \text{Res})$ of $D$ there is a mapping $\Psi_S$ such that $\Psi_S((A_1, \ldots, A_n), \sigma_0)$ is defined and $\Psi_S((A_1, \ldots, A_n), \sigma_0) \models \gamma$.

As before, the sequence $(A_1, \ldots, A_n)$, if it exists, is called an \textit{answer for the query } $Q$ \textit{wrt } $D$. 
Thank you for your attention!