Artificial Intelligence

Lecture 9 – Reasoning about Actions

Outline

- Reasoning about the effects of actions
- Situation calculus
- Example: Blocks world
- The frame problem
- Qualification and ramification problems

Knowledge representation & reasoning

- In previous lectures, we have used propositional and predicate calculus to reason about a *static world*
- Propositions and predicates describe properties and relations of a single state of affairs
- Sentences are true or false relative to this particular state of affairs
- Truth assignment/interpretation is *fixed* – if anything were to change, the truth assignment/interpretation would no longer reflect the new state of affairs

Change

- *Representing how things change* is one of the most important areas in knowledge representation
  - to learn we need to represent what happened in the past
  - to plan we need to represent hypothetical future states
- *Reasoning about change* is one of the most important kinds of reasoning – critical for selecting *actions*

Reasoning about Change

- When reasoning about how actions change the world, we need to consider:
  - when an action is *applicable* – the qualification problem
  - what the action *changes* – the ramification problem
  - what the action *does not change* – the representational and inferential frame problems

Problem-Solving Representations

- AI problem-solving is defined in terms of goals, states and operators
- State representations are *global* – they contain all the information about a particular state, e.g., a complete board position in chess
- Applying an operator transforms a (complete) state description into a new state description
- Works well when the state descriptions are compact, and the global consequences of an action are easy to specify, e.g., noughts and crosses

Localised Representations

- In more complex domains, actions typically have local effects
- For example, moving an object from *A* to *B* does not usually change any ot its other properties, such as its size, weight, colour etc.
- Not necessary to represent the effect of an action on the whole state
- Focus instead on just those aspects of a state which are relevant to performing an action and which are affected by it
- When representing and reasoning about action and change we therefore use partial descriptions of states and localised descriptions of actions

Situation Calculus

- World is represented as a series of *situations* or 'snapshots'
- Predicates and functions whose truth values vary from situation to situation are called *fluents*
- Each fluent is extended with an additional situation argument, e.g., *At*(*brian*, *C3*, s_0), *At*(*brian*, *office*, s_1)
- Situations are simply constants – names for a particular state of the world
- All sentcences which are true at a given point in time (situation) have the same situation argument

Example: Blocks World

- The blocks world domain consists of
  - a table, set of cubic blocks and a robot arm
  - each block is either on the table, stacked on top of another block or held by the arm
  - the arm can pick up a block and move it to another position either on the table or on top of another block
  - the arm can only pick up one block at a time, so it cannot pick up a block which has another block on top

Example: Representing the Blocks World

- Blocks are represented by constants: *a*, *b*, *c*, ... etc.
- States are described using the following fluents:

On(*x*, *y*, *s*)	block *x* is on block *y* in situation *s*
OnTable(*x*, *s*)	block *x* is on the table in situation *s*
Clear(*x*, *s*)	there is no bock on top of block *x* in situation *s*
Holding(*x*, *s*)	the arm is holding block *x* in situation *s*
ArmEmpty(*s*)	the arm is not holding any block in situation *s*

- We also need some axionms specifying the *domain theory*

Example: Reasoning about the Blocks World

$\forall x, \forall s\ (OnTable\ (x, s) \leftrightarrow \neg\exists y\ On(x, y, s) \wedge \neg Holding(x, s))$

- for all blocks *x* and situations *s*, *x* is on the table in *s*, if and only if there is no block *y* which *x* is on in *s* and the arm is not holding *x* in *s*

$\forall x, \forall s\ (\exists y\ On(x, y, s) \leftrightarrow \neg OnTable(x, s) \wedge \neg Holding(x, s))$

- for all blocks *x* and situations *s*, *x* is on a block *y* in *s*, if and only if *x* is not on the table and the arm is not holding *x* in *s*

$\forall x, \forall s\ (Holding(x, s)) \leftrightarrow \neg\exists y\ On(x, y, s) \wedge \neg OnTable(x, s))$

- for all blocks *x* and situations *s*, the arm is holding *x* in *s*, if and only if there is no block *y* which *x* is on and *x* is not an the table in *s*

Example: Reasoning about the Blocks World

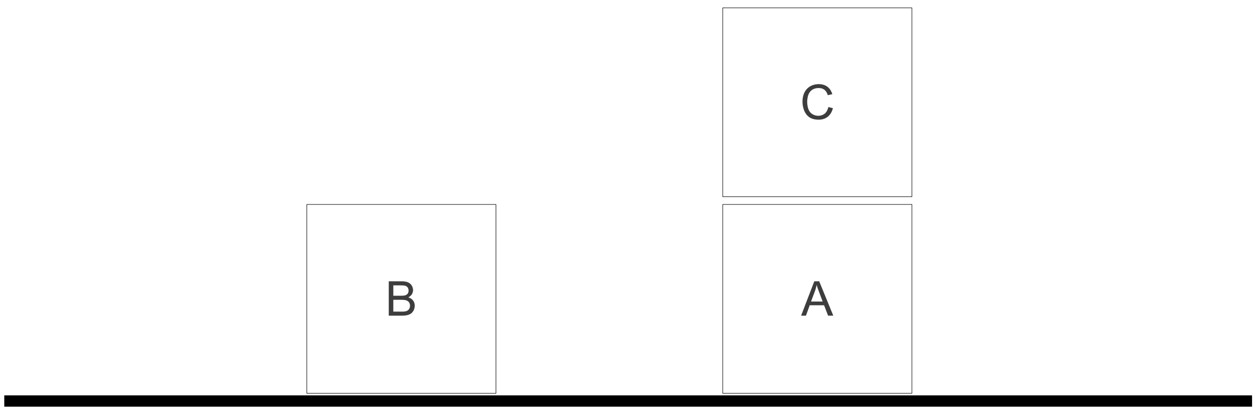
$\forall x, \forall s\ (Clear\ (x, s)\ \leftrightarrow \neg\exists y\ On(y, x, s))$

- for all blocks *x* and situations *s*, *x* is clear in *s*, if and only if there is no block *y* which is on *x* in *s*

$\forall s\ (ArmEmpty(s))\ \leftrightarrow \neg\exists x\ Holding(x, s))$

- for all situations *s*, the arm is empty in *s*, if and only if there is no block *x* which the arm is holding in *s*

Example: Blocks World



- initial state

$$On(c, a, s_0) \wedge OnTable(a, s_0) \wedge OnTable(b, s_0) \wedge ArmEmpty(s_0) \wedge Clear(b, s_0) \wedge Clear(c, s_0)$$

Example: Blocks World

- From the initial state

$$On(c, a, s_0) \wedge OnTable(a, s_0) \wedge OnTable(b, s_0) \wedge$$
$$ArmEmpty(s_0) \wedge Clear(b, s_0) \wedge Clear(c, s_0)$$

- we can derive

$$\neg OnTable(c, s_0),\ \ \neg\exists y\ On(a, y, s_0),\ \ \neg\exists y\ On(b, y, s_0),$$

$$\neg Clear(a, s_0),\ \neg\exists x\ Holding(x, s_0)$$

- However this doesn't tell us what is true if we perform an action in s_0

Representing Actions

- Actions are represented by *terms* which name the action and specify its parameters, e.g.,
  - the term *pickup(x)* denotes the action of picking up a block *x*
  - the term *drop* denotes the action of droping the block held by the arm
- Note that actions are terms not formulas – they denote the action performed, not true or false

Results of Actions

- An action performed in a given situation *s* results in a *new situation*
- The formula *result(a, s)* is used to denote the situation which results fro performing the action *a* in the situation *s*, e.g.,
  - the term *result(pickup(x), s)* denotes the situation which results from picking up the block *x* in situation *s*
- Each action is decribed by two axioms: a *possibility axiom* and an *effects axiom*

Possibility and Effects Axioms

- A possibilty (precondition) axiom says when it is possible to execute an action
- Possibility axioms have the form *Precondition* → *Poss(a, s)*, where *Poss(a, s)* means that it is possible to execute action *a* in situation *s*
- An effects (postcondition) axiom says what happens when which is possible is executed
- Effects axioms have the form *Poss(a, s)* → *Changes that result from the action*

Example Possibility Axioms

$\forall x, \forall s\ (ArmEmpty\ (s) \land Clear(x, s) \rightarrow Poss(pickup(x), s))$

- the action *pickup(x)* is possible in situation *s*, if the arm is empty and block *x* is clear in *s*

$\forall x, \forall s\ (Holding(x, s)) \rightarrow Poss(drop, s))$

- the action *drop* is possible in situation *s*, if the arm is holding block *x* in *s*

Example Effects Axioms

$\forall x, \forall y, \forall s\ (Poss(pickup(x), s) \wedge On(x, y, s) \rightarrow$

$\qquad Holding(x, result(pickup(x), s)) \wedge Clear(y, result(pickup(x), s))$

- the effects of performing a *pickup(x)* action which is possible in situation *s* are the arm is holding block *x* and if *x* was on block *y*, then *y* is clear in the situation resulting from the pickup action

$\forall x, \forall y, \forall s\ (Poss(pickup(x), s) \wedge OnTable(x, s) \rightarrow$

$\qquad Holding(x, result(pickup(x), s)))$

$\forall x, \forall y, \forall s\ (Poss(drop, s) \wedge Holding(x, s) \rightarrow$

$\quad ArmEmpty(result(drop, s)) \wedge OnTable(x, result(drop, s)))$

- the effects of performing a drop action which is possible in situation *s* are that the arm is empty and *x* is on the table in the situation resulting from the drop action

Frame Axioms

- Effects axioms are not sufficient to keep track of which formulas are true in a given situation
- We also need to explicitly state which parts of the world are not changed by an action
- Axioms which describe which parts of the world are *not* changed by an action are called *frame axioms*
- Together, the effect and frame axioms provide a complete description of how the world evolves in response to actions

Example Frame Axioms

$\forall a, \forall x, \forall s\ (\neg Holding(x, s) \wedge a \neq pickup(x) \rightarrow$

$\neg Holding(x, result(a, s)))$

- for all actions *a*, blocks *x* and situations *s*, if the arm is not holding *x* in *s* and *a* is not the action of picking up *x*, then the arm will not be holding *x* in the situation which results from performing the action *a*

$\forall a, \forall x, \forall s\ (Holding(x, s) \wedge a \neq drop \rightarrow Holding(x, result(a, s)))$

- for all actions *a*, blocks *x* and situations *s*, if the arm is holding *x* in *s* and *a* is not the drop action, then the arm will still be holding *x* in the situation which results from performing the action *a*

Exercise: Situation Calculus

- Use the possibility, effects and frame axioms (and the domain theory) to show that picking up block c and dropping it on the table results in all three blocks being on the table, i.e. that

$$OnTable(a, result(drop, result(pickup(c), s_0))) \wedge$$

$$OnTable(b, result(drop, result(pickup(c), s_0))) \wedge$$

$$OnTable(c, result(drop, result(pickup(c), s_0)))$$

is true

Exercise 1: Solution (frame axioms)

$$\forall a, \forall x, \forall s\, (OnTable(x, s) \wedge a \neq pickup(x) \rightarrow$$
$$OnTable(x, result(a, s)))$$

- for all actions *a*, blocks *x* and situations *s*, if *x* is on the table in *s* and *a* is not the action of picking up *x*, then *x* is still in the table in the situation which results from performing the action *a*
- we also need a unique name assumption, e.g.:

$$\forall a, \forall y, \forall s\, (x \neq y \rightarrow pickup(x) \neq pickup(y))$$

Exercise 1: Solution

- For block *a*, given $OnTable(a, s_0)$ we can show

$$OnTable(a, result(pickup(c), s_0))$$

- Using modus ponens and the frame axiom

$$\forall act, \forall x, \forall s\ (OnTable(x, s) \wedge act \neq pickup(x) \rightarrow OnTable(x, result(act, s)))$$

- Since $pickup(a) \neq pickup(c)$ using modus ponens and the frame axiom, we can then show

$$OnTable(a, result(drop, result(pickup(c), s_0)))$$

- Since $pickup(a) \neq drop$
- Derivation for block *b* is similar

Exercise 1: Solution

- For block *c*: from the initial state and the possibility axiom

  $\forall x, \forall s\ (ArmEmpty\ (s) \land Clear(x, s) \rightarrow Poss(pickup(x), s))$
- we have that it is possible to perform a *pickup* action on *c* $Poss(pickup(c), s_0))$
- From the effects axiom

  $\forall x, \forall y, \forall s\ (Poss(pickup(x), s) \land On(x, y, s) \rightarrow$

  $Holding(x, result(pickup(x), s)) \land Clear(y, result(pickup(x), s))$
- we can derive the effect of actually picking up block *c*, i.e.:

  $Holding(c, result(pickup(x), s_0)) \land Clear(a, result(pickup(x), s_0))$

Exercise 1: Solution

- from *Holding*$(c, result(pickup(x), s_0))$ and the possibility axiom

  $\forall x, \forall s\ (Holding(x, s)) \rightarrow Poss(drop, s))$

- we have that $Poss(drop, result(pickup(c), s_0))$ and using the effects axiom

  $\forall x, \forall y, \forall s\ (Poss(drop, s) \wedge Holding(x, s) \rightarrow$

  $ArmEmpty(result(drop, s)) \wedge OnTable(x, result(drop, s)))$

- we can show

  $OnTable(c, result(drop, result(pickup(c), s_0)))$

Exercise 2: Situation Calculus 2

- Write possibility, effects and frame axioms for the action *put(x, y)* which puts the block *x* on the block *y*
- For *put(x, y)* to be possible, *x* must be held by the arm, and *y* must be clear
- The effects of *put(x, y)* are that the arm is empty and that *x* is on *y*
- The frame axioms should state that, apart from the effects of a *put(x, y)* listed above, nothing else changes

Exercise 2: Solution

$\forall x, \forall y, \forall s\ (Holding(x, s) \land Clear(y, s) \rightarrow Poss(put(x, y), s))$

- the action *put(x, y)* is possible in situation *s*, if the arm is holding block *x* and *y* is clear in *s*

$\forall x, \forall y, \forall s\ (Poss(put(x, y), s) \rightarrow$

$ArmEmpty(result(put(x, y), s))) \land On(x, y, result(put(x, y), s)))$

- The effects of performing a *put(x, y)* action which is possible in situation *s* are that the arm is empty and block *x* is on block *y* in the situation resulting from the put action

Exercise 2: Solution

$\forall a, \forall x, \forall y, \forall s\ (Holding(x, s) \wedge (a \neq drop \vee a \neq put(x, y))$
$\quad \rightarrow Holding(x, result(a, s)))$

- for all actions *a*, blocks *x* and situations *s*, if the arm is holding *x* in *s* and *a* is not a drop action or a put action, then the arm will still be holding *x* in the situation which results from performing the action *a*
- Note: this *replaces* the previous frame axiom for *Holding*

The Frame Problem(s)

- The (typically very large) number of frame axioms required to reason about actions in a domain gives rise to the *representaional frame problem*
- The *inferential frame problem* refers to the need to raeson explicitly about things that don't change
- When reasoning about sequences of actions, each propeerty of interest must be (re)derived for each new situation, even if the property hasn't changed
- Since each action usually changes only a few facts about a situation this is very inefficient

The Qualification Problem

- In general, it is difficult to specify *precisely* the situations in which an action will have the specified (intended) results
- For example, it may not be possible to perform a *pickup* action if the block is slippery or glued to the table
- If these "side conditions" are left out of the effect and frame axioms, we may derive false beliefs about the consequences of executing and action
- How to qualify the "normal" effects of an action in "abnormal" circumstances is the *qualification problem*

The Ramification Problem

- In addition to the *explicit* consequences specified in their definitions, actions also have *implicit* consequences
- For example, picking up a box also picks up all of the objects in the box (if any), and if I take the box somewhere, I also take its contents etc.
- The *ramification problem* can be seen as the derivation of the *ultimate* effects of an action
- May involve additional simple inferences (if the box is in the living room, then all the objects in the box are in the living room), reasoning about cause and effect (naive physics) and other kinds of consequences
- Hard to know when to stop...

Example

- Gavrillo Princip shoots Archduke Franz Ferdinand
- Shooting Franz Ferdinand causes him to die
- The death of Franz Ferdinand leads to the Austro-Hungarian Empire declaring war on Serbia
- Which leads to the first World War
- Which leads to the Treaty of Versailles ...