CPT302 W3

Agent Architectures

An agent architecture is a software design for an agent (代理是一个计算机系统，实际上就是软件)，其结构的 top-level 被分解成 perception - state - decision - action 的循环。

An agent architecture defines:

- key data structures
- operations on data structures
- control flow between operations

Types of Agents

Symbolic Reasoning Agents：符号推理代理，它通过 symbolic and logical representations 来显式地做出决定 (把环境等抽象成符号，企图让代理理解世界，比如表示“桌子上放着 A”: on{A, table})。

Reactive Agents：反应代理，代理直接对环境做出反应，而不是像 Symbolic Reasoning Agents 那样进行逻辑推理 (Symbolic Reasoning 可解释性强，但效果差，因此有了 Reactive Agents)。

Hybrid Agents：混合代理，Hybrid architectures 试图结合symbolic architectures 和 reactive architectures 的精华。

Deductive Reasoning Agents

演绎推理代理是 Symbolic Reasoning Agents 的一种，这是构建 AI 系统的传统方法 (symbolic AI)。

- Symbolic representation of environment and behavior (用符号表示环境和行为)
- Syntactic manipulation (句法操作) of symbolic representation

Symbolic representation -> logical formulae (用逻辑公式表示)

Syntactic manipulation -> logical deduction (theorem proving，根据逻辑公式进行逻辑推理)

Transduction Problem

在 Symbolic Reasoning Agents 中，我们需要把环境和行为表示成符号。

Transduction Problem 就是讨论如何将现实世界翻译成准确、充分的由符号描述的问题，并使该描述有用。

Representation / Reasoning Problem

如何用符号表示有关复杂现实世界的实体和过程的信息的问题，以及如何让代理及时使用这些信息进行推理，以使结果有用。

Agents as Theorem Provers

代理理论 (Theory of agency) φ：一些理论，解释了智能代理应该如何表现以优化某些性能。理论 φ 被视为可执行的说明 (executable specification)，可以直接执行以产生代理的行为。

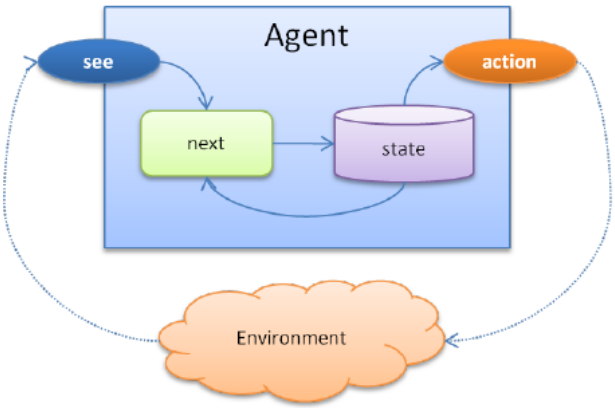
Deliberate agents: simple model of logic-based agents

- Internal state assumed to be a database of formulae (predicate logic，谓词逻辑，看 THE SYNTAX OF PREDICATE LOGIC)，我们使用 predicate logic 来表示当前环境的状态
- 类似于人类的 beliefs - Internal state 可能包括不正确（过时、无效）的信息

Let:

- L be the set of formulae of classical first-order logic (一阶逻辑，也叫一阶谓词逻辑)
- $D = 2^L$ 是 L 数据库的集合 (即所有可能状态的组合)
  - $DB_1, DB_2, \ldots$ 是 D 中的元素
  - DB 表示代理的 internal state (当前环境的状态)
- ρ 是一组演绎规则 (deduction rules)，用于模拟代理的决策过程
- $DB \vdash_\rho \varphi$ 意为：公式 φ 可以仅用演绎规则 ρ 就从数据库 DB 中证明 (在当前状态 DB 下，根据演绎规则 ρ，可以证明操作 φ 是最优的)

Action Selection as Theorem Proving



- Perception function *see*
- Next state function *next*
- Action-selection function *action*

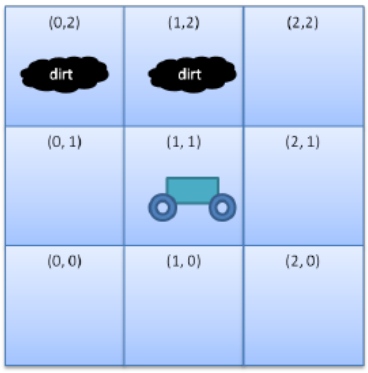
以上面的 agent 为例，我们需要找到当前环境状态下的最优操作。

foreach $\alpha \in Ac$ **do**
  if $DB \vdash_\rho Do(\alpha)$ **then**
    return α;
  end
end
foreach $\alpha \in Ac$ **do**
  if $DB \nvdash_\rho \neg Do(\alpha)$ **then**
    return α;
  end
end

注：上图第一个循环希望找到最优的操作；当没有找到最优的操作时，我们希望找到可能执行的操作，因此第二个循环的意思是“如果不执行 α 不能被证明是最优的 (说明执行 α 是可能的)”。

Example - Vacuum World

- A small robotic agent that cleans up a room divided into a grid of equally sized squares
- Agent is equipped with a dirt sensor and a vacuum cleaner
- Agent always has a definite orientation (north, south, east, west)
- Agent can move foreward one step and turn right 90º
- For simplicity we assume the room is 3×3 and agent always starts in square 0,0 facing north



接下来我们用谓词逻辑来表示环境和行为：

- Possible percepts
  $Per = \{dirt, null\}$
- Domain predicates describing internal state
  $In(x,y)$ – agent is at (x,y)
  $Dirt(x,y)$ – there is dirt at (x,y)
  $Facing(d)$ – agent is facing direction d
- Possible actions
  $Act = \{forward, suck, turn\}$

注：Per 代表对环境的感知，有两种情况：dirt 和 null。

Deduction rules

1. $In(x,y) \wedge Dirt(x,y) \longrightarrow Do(suck)$
2. $In(0,0) \wedge Facing(north) \wedge \neg Dirt(0,0) \longrightarrow Do(forward)$
3. $In(0,1) \wedge Facing(north) \wedge \neg Dirt(0,1) \longrightarrow Do(forward)$
4. $In(0,2) \wedge Facing(north) \wedge \neg Dirt(0,2) \longrightarrow Do(turn)$
5. $In(0,2) \wedge Facing(east) \wedge \neg Dirt(0,2) \longrightarrow Do(forward)$
6. $\cdots$

Problems with Deductive Reasoning

- 如何将摄像机的输入转换为 {dirt, null}，以及如何表示动态环境的属性
- Decision making 假设环境是静态的：calculative rationality
    - Decision making process suggests an action that was optimal when the process started
    - 如果决策是即时的，那么我们可以解决这个问题
- 通过定理证明做出决策是复杂的（它可能永远不会终止）