Week2 Embedded Agents

Agents are embedded in an environment, meaning that an agent affects and is affected by the environment. An agent experiences its environment through sensors an acts on its environment through effectors.

2.1 Math revision

- Power Set: The power set of a set A is the set of all subsets of A. It is written as 2^A
  - $A = \{x \mid 0 < x < 7 \text{ and } x \text{ is odd}\}$. A={1,3,5} hence 2^A={∅,{1},{3},{5},{1,3},{1,5},{3,5},{1,3,5}}

2.2 Accessible and inaccessible environments

- An accessible environment is one in which the agent can obtain complete, accurate, up-to-date information about the environment's state
- Most moderately complex environments (including, for example, the everyday physical world and the Internet) are inaccessible.

Example:

- Accessible: Chess
- Inacessible: Stock market

2.3 Deterministic and non-deterministic environments

- A deterministic environment is one in which every action has a single guaranteed effect — there is no uncertainty about the state that will result from performing an action.
- The physical world can to all intents and purposes be regarded as non-deterministic.

2.4 Static and dynamic environments

- A static environment is one in which the only changes to the environment are those caused by actions made by the agent.对环境的唯一更改是由agent执行的操作引起的更改。
- A dynamic environment is one that has other processes and/or agents operating on it, and so changes in ways beyond the agent's control. 不仅仅是有一个agent在决策，其他agents在决策时会互相影响

2.5 Formal specification of an embedded agent

$Env =< E, e_0, \pi >$

2.6 Utility Functions

Utility functions allow an agent to understand how "good" an outcome is.

Week3:Deductive,reactive and hybrid reasoning agents

- Deductive reasoning agents, which have an explicitly model of the world represented in logical formulas and reason using logical deduction.
    - Concurrent MetateM, a specific language for specifying deductive reasoning agents.
- Reactive agents, which do not have an explicit model of the world that they reason with, but instead their behaviour is driven by a set of "stimulus -> action" rules.
    - The subsumption architecture, probably the most well known reactive agent architecture.
- Hybrid agents, which combine a purely reactive layer with higher-level reasoning layers.
    - We'll look at TouringMachines and InteRRaP as two examples of these.

3.1 Deductive Agents

3.1.1 Concurrent MetateM

$\circ\phi$	ϕ true in next state
$\circledcirc\phi$	ϕ true in previous state
$\Diamond\phi$	ϕ true now or at some point in future
$\Box\phi$	ϕ true now and at all points in future
$\blacklozenge\phi$	ϕ true sometime in the past
$\blacksquare\phi$	ϕ true at every point in the past
$\phi\mathcal{U}\psi$	ψ true at some point in the future and ϕ true at all points until then
$\phi\mathcal{S}\psi$	ψ true at some point in the past and ϕ true at all points since then
$\phi\mathcal{W}\psi$	ϕ true at all points in the future unless ψ becomes true
$\phi\mathcal{Z}\psi$	like $\mathcal{S}$ but ψ may never have become true

$\Diamond\varphi$	未来某个时刻有φ
$\Box\varphi$	未来总是有φ
$\blacklozenge\varphi$	过去某个时刻有φ
$\blacksquare\varphi$	过去总是有φ
$\varphi\mathcal{U}\ \psi$	直到 ψ 以前 φ 为真
$\varphi\mathcal{S}\ \psi$	从ψ以来 φ 一直为真
$\varphi\mathcal{W}\ \psi$	除非 ψ 为真 φ 才为真
$\varphi\mathcal{Z}\ \psi$	除非 ψ 为真 φ 才为真

E.g:

$student(you)\ \upsilon\ graduate(you)$ -- at some point in the future you will graduate, until then you will be a student. 直到毕业前你一直是学生

$title(me, dr)\ S\ awarded(me,phd)$ -- at some point in the past I was awarded a PhD and my title has been Dr ever since then. 从授予我PHD学位以后，我一直都是Dr

$\neg friends(me, you)\ W\ apologise(you, me)$ -- we will never be friends unless you apologise to me.

$title(you, dr)\ Z\ awarded(you, phd)$ -- if it's true that you've been awarded a PhD in the past, then your title has been Dr ever since then.

3.2 Reactive Agents

不再使用像MetaM中那样的符号来表示行为，只是通过响应环境而产生的各种简单行为的相互作用产生智能行为。

3.2.1 subsumption architecture

subsumption architecture - the most famous reactive agent architecture

把行为组织成等级层次结构，等级层级结构中的低层可以抑制高层，层次越低优先度越高，**其主要思想就是层次越高表示的越抽象的行为**举例来说，有人希望移动机器，人有躲避障碍的行为。这显然要给躲避障碍物赋予一个高的优先级。

举例：火星车Steels' experiments with the Mars explorer system

局限性：

- 由于纯反应式 Agent 按照局部信息（即关于Agent当前状态的信息）做出央策，很难想像这种决策方法能考虑非局部信息
- 必须经过费力费时才能创造出纯反应式 Agent
- Agents可用的行为越多，理解或预测它们的行为就越困难，因为不同层之间的相互作用的动态变得太复杂而无法理解。

3.3 Hybrid Agents

Week4: Practical Reasoning Agents

2.1 Delibration and Means-ends reasoning

- **Deliberation**: deciding what state of affairs the agent wants to achieve; the results of deliberation are referred to as the agent's **intentions**. (**weighing up** the options available to me and **deciding** which ones to commit to as intentions.)
- **Means-ends reasoning**: deciding how to achieve these states of affairs. (Once I've decided on my intention, I need to work out what I'm going to do to try to achieve this.)
- **Intentions**: We have seen that the output of the agent's deliberation are **intentions**. If an agent has an intention X this means is intends to (try) to bring about the state of affairs X.

2.2 Resource bounded reasoning and calculative rationality

calculative rationality

E.g:

Is the following statement true or false?

An agent with the property of calculative rationality that is situated in a static environment is guaranteed to always make optimal decisions.

TRUE

If an agent has the property of calculative rationality, the decisions it makes are guaranteed to be optimal according to the information it had available when it started making its decision. If the environment is static, this means that the only way the world can change is if the agent itself performs some action. This means that the world is guaranteed not to change during the time the agent is deciding what to do, and so a decision that is optimal according to the information available when the decision making process started will still be optimal when the agent finishes making its decision.

2.3 BDI agents

BDI stands for **Beliefs, Desires, Intentions**

Based on an agent's beliefs, it has a set of desires. Once an agent commits to trying to achieve one of its desires, this desire becomes an intention.

E.g: If my workload eases then I might decide it is feasible to visit my family and commit to this as an intention. Once I have committed to this as an intention, we expect (as discussed previously) that: I will invest some effort in trying to achieve this;

2.3.1 BDI V1

1. $B := B_0;$
2. $I := I_0;$
3. *while true do*
4. $\quad$ *get next percept* $p;$
5. $\quad B := brf(B, p);$
6. $\quad D := options(B, I);$
7. $\quad I := filter(B, D, I);$
8. $\quad \pi := plan(B, I);$
9. $\quad execute(\pi);$
10. *end while*

- When an agent receives a new percept from the environment, it updates its beliefs to take this new information into account using its belief revision function: $brf()$.
- $options()$, which takes as input the agent's current beliefs and intentions and returns a set of desires;
- $filter()$ function, which takes as input the agent's current beliefs, desires and intentions and returns the "best" options to commit to as its new intentions.
- $plan()$ Function, which takes the agent's current beliefs and intentions, and returns a plan for the agent to follow in order to try to achieve its intentions.
- Means-ends reasoning is concerned with working out what the agent is going to do to try to achieve its intentions. This is what the $plan()$ function does: determines a plan the agent is going to employ to try to achieve its intentions.
- **Deliberation is a two step process**. First the agent identifies its options (line 6) and then the agent filters these to select its intentions (line 7). Deliberation therefore takes place over lines 6 and 7.

2.3.2 BDI V2

```
1.  B := B_0;
2.  I := I_0;
3.  while true do
4.      get next percept p;
5.      B := brf(B,p);
6.      D := options(B,I);
7.      I := filter(B,D,I);
8.      π := plan(B,I);
9.      while not empty(π) do
10.         a := head(π);
11.         execute(a);
12.         π := tail(π);
13.         get next percept p;
14.         B := brf(B,p);
15.         if not sound(π,I,B) then
16.             π := plan(B,I);
17.         end if
18.     end while
19. end while
```

Version 2 of our agent control loop allows the agent to execute one action from its plan at a time, pausing to observe the environment after each action and reconsider whether its plan is still appropriate based on its new beliefs.

- $empty(\pi)$ is true if and only if the plan π is empty.
- $head(\pi)$ returns the first action of the plan π.
- $tail(\pi)$ returns the remainder of the plan π once the first action of π has been removed.
- $sound(\pi, I, B)$ is true if and only if, according to the agent's beliefs B, the agent can expect the plan π to achieve its intentions I.

2.3.3 BDI V3

```
1.  B := B_0;
2.  I := I_0;
3.  while true do
4.      get next percept p;
5.      B := brf(B,p);
6.      D := options(B,I);
7.      I := filter(B,D,I);
8.      π := plan(B,I);
9.      while not (empty(π) or succeded(I,B) or impossible(I,B)) do
10.         a := head(π);
11.         execute(a);
12.         π := tail(π);
13.         get next percept p;
14.         B := brf(B,p);
15.         D := options(B,I);
16.         I := filter(B,D,I);
17.         if not sound(π,I,B) then
18.             π := plan(B,I);
19.         end if
20.     end while
21. end while
```

In version 2 of the control loop, the only time the agent ever stops to consider its intentions is when it has completed executing its plan. We will now adapt the control loop so that the agent is able to adjust its intentions if it is appropriate to do so. For this we need to introduce the following predicates.

- $succeeded(I, B)$ is true if and only if, based on its beliefs B, the agent believes it has achieved its intentions I.
- $impossible(I, B)$ is true if and only if, based on its beliefs B, the agent believes it is impossible to achieve its intentions I

If the agent believes that its intentions are either impossible or already achieved, there is no point in the agent persisting in trying to achieve those intentions.

Agents using version 3 of the BDI control loop stop and deliberate again in order to reconsider their intentions after every action they execute (lines 15-16 of control loop version 3).

2.3.4 BDI V4 (Meta-level)

慎思的过程需要花费大量的时间，当一个agent在慎思时，周围的环境也在改变，可能会提出没有关系的新形成的意图

```
1.   B := B_0;
2.   I := I_0;
3.   while true do
4.       get next percept p;
5.       B := brf(B,p);
6.       D := options(B,I);
7.       I := filter(B,D,I);
8.       π := plan(B,I);
9.       while not (empty(π) or succeded(I,B) or impossible(I,B)) do
10.          a := head(π);
11.          execute(a);
12.          π := tail(π);
13.          get next percept p;
14.          B := brf(B,p);
15.          if reconsider(I,B) then
16.              D := options(B,I);
17.              I := filter(B,D,I);
18.          end if
19.          if not sound(π,I,B) then
20.              π := plan(B,I);
21.          end if
22.      end while
23.  end while
```

- $reconsider(I, B)$ is true if and only if, given its beliefs B, the agent believes it is appropriate to reconsider its intentions I.

There is an important implicit assumption here: the cost of determining whether **reconsider(I,B) is true must be less than the cost of the actual deliberation** process itself (i.e., performing functions options and filters). If this were not the case, then the agent might as well go ahead and do the deliberation, since this would be less costly than trying to decide whether it should deliberate or not.

E.g:

The following table summarises the possible interactions between deliberation and the meta-level control (via $reconsider(I, B)$).

Situation number	Did *reconsider(I,B)* return true?	Did reconsidering cause the agent to change its intentions?	Would the agent have changed its intentions if it had reconsidered them?
1	No	n/a	No
2	No	n/a	Yes
3	Yes	No	n/a
4	Yes	Yes	n/a

So in situation 1, the agent choose not to deliberate, and if it had chosen to deliberate then it would not have changed its intentions.

In situation 2, the agent choose not to deliberate, and if it had chosen to deliberate then it would have changed its intentions.

In situation 3, the agent chose to deliberate, and did not change its intentions.

In situation 4, the agent chose to deliberate, and did change its intentions.

In which of the situations is the agent's meta-control system, i.e., $reconsider(I, B)$, behaving optimally?

agents慎思之后就会改变intention，因为deliberation是两部分组成（ **weighing up** the options available to me and **deciding** which ones to commit to as intentions）。当且仅当Agent选择镇思的时候，使Agent 改变意图，两数reconsider (...)的行为是最优的。对于 Agent 选择了慎思过程，但没有改交意图时，则在慎思过程中花费的努力就是浪费。同样，如果 Agent应该改变意图，但是没有能改变失败了，则花费在实现意图上的努力也是浪费的。因此1和4最优

Week5: communication, coordination and cooperation

5.1 Communication

Communication acts:

- The **locutionary act**: this is the act of making the utterance.
- The **llocutionary act**: the conveying of the speaker's intentions to the hearer.
- The **perlocution**: the effect achieved by the speech act.

For example, if I utter the speech act "Please make me some tea":

- The locutionary act is me saying "Please make me some tea".
- The illocutionary act is that I have requested you to make me some tea.
- The perlocution depends on how you respond to my request. I hope that you will make me some tea, but you might not.

5.1.1 KQML

Autonomous heterogeneous agents in a multi-agent system require a shared **agent communication language** (ACL) so they can communicate with one another. KQML属于ACL的一种

5.1.2 FIPA

在FIPA ACL中，把发送agent必须要满足的条件叫可行性前提（Feasibility Precondition），把Agent希望通过执行通信行动后所取得的效果称为理性效果（Rational Effect）

the intended effect of each performative is defined (the rational effect): the receiver of a message understands that this is what the sender wishes to achieve.

FIPA ACL. 语义还规定了发送方认为应当是真的条件.考虑一个通信行动$\langle i, inform(j, \alpha)\rangle$，在这里，Inform 的内容是命题$\alpha$，它的含义是 Agent i通知 Agent j 命题α为真。 根据 FIPA ACL. 的语义,inform 的Feasibility Precondition 就是Agent i认为命题α为真,并且相信 Agentj没有关于α的任何知识。Inform 的Rational Effect就是 Agent i 想让 Agent j相信命题α为真.

E.g:

If we see an agent send a FIPA request message then we can be sure that the feasibility precondition is true? FALSE!!!

解释：The feasibility precondition is given in terms of the sender's mental state (i.e., the sender must believe that the receiver can perform the action, and they must not believe that the receiver already intends to perform the action). Agents are autonomous, independent entities and we cannot (in general) access their private mental state. This means that we can't be sure, when an agent sends a request message, that the relevant feasibility precondition holds. For example, if the sender is a malicious agent, it may in fact believe that the receiver cannot perform the action, but nevertheless makes the request as a way of disrupting the receiver (by taking up its attention with the message). What this means in general is that we cannot be sure that agents are conforming to the semantics of the FIPA ACL.

5.1.3 Social commitments

By giving a clear semantics for an ACL, we aim to make the agents' reasoning process easier. If an agent ag1 is conforming to the semantics of the FIPA ACL, then this means that an agent ag2 receiving a message from ag1ag1 can know something about ag1's mental state. But we have just seen that we cannot be sure if an agent is conforming to the FIPA ACL semantics, exactly because they are defined in terms of the agent's mental state. This is a challenge for ACLs that is not yet resolved. One approach to address this is to define ACL semantics based on **social commitments.**

5.2 Cooperation

Cooperative Distributed Problem Solving (CDPS)

Cooperative Distributed Problem Solving involves these steps:

- Problem decomposition: dividing the problem into smaller tasks for distribution among agents.
- Subproblem allocation: determining which subproblems to allocate to which agents.
- Subproblem solution: where the agents solve the individual subproblems that they have been allocated.
- Subproblem synthesis: where solutions to individual subproblems are integrated into an overall solution.

5.2.1 Contract Net protocol

which addresses the second stage of Cooperative Distributed Problem Solving (CDPS): sub-problem allocation.

在合同网协议中，所有Agent 可以归纳为两种角色：管理者和承包商。其中，管理者的职责包括：对每一个待求解任务建立 任务通知书，将任务通知书发送给有关的承包商 Agent ；接收并评估来自承包商的投标；从投标中选择最合适的承包商，与之建立 合同；监督任务的完成，并综合结果。承包商的职责包括：接收相关任务通知书：根据自己的能力判断是否接受任务，不接受发送拒标通知，否则发送投标通知：如果投标被接受，按合同执行分配给自己的任务，向管理者发送求解结果。

在合同网协作方法中，不需要预先规定 Agent 的角色，任何Agent通过发布任务通知而成为管理者；任何 Agent通过应答任务通知而成为承包商。系统中的每一待求解任务，由承担该任务的Agent 负责完成。当该 Agent 无法独立完成该任务时，它就将履行管理者职责，为该任务发送任务通知书：然后从返回的投标中选择‘最合适”的 Agent，将任务分配给此Agent，建立相应的合同。

5.3 Coordinate

在合作中，要讨论的主要问题就是协调问题。协调问题是指如何管理 Agent动作之间的内部依赖关系。如果 Agent参加的动作中存在以任何方式的相互作用，那么一定的协调机制是必需的。两个动作之间是如何进行相互作用的?可以考感下面现实世界中的例子：

- 你和我都打算商开房间，并且我们各自向门走去，而门只允许我们一个人过去。我有礼貌地允许你先离开。在这个例子中，我们的活动需要协调，因为这里只有一个资源（门），而我们都需要使用该资源，但是在同一时间只允许一个人使用。
- 我打算提交一份助学金申请书，但是为了做这件事，需要你的签字。在这种情况下，我提交助学金申请书的动作依赖于你签字的动作 --除非你的动作完成，我不能完成我的动作。换句话说，我的动作依赖于你的动作。
- 我从网络上得到一份文件的软持贝。我知道你也对这份报告感火趣，这样我主动复印 一份报告，并给你一份拷贝。在这种情况下，我们的动作不需要严格的协调-—因为这份报告在网上可以免费获得，你也可能下载并打印。但是通过我主动的打印，节约了你的时间，进而直觉上提高了你的效用。

广义上动作之间的依赖关系分为消极和积极

Negative relationships

- **Resource induced**. This is where there are insufficient resources to allow both actions to occur simultaneously. For example, if we're both trying to walk through the same door, or if two agents both need access to the same printer.
- **Incompatibility**. This is where the actions require mutually exclusive states. For example, you may need to sit an exam, which requires that your phone is switched off. And your partner may be booking

you both a holiday, which requires your phone to be switched on so that they can contact you to confirm the dates.

Positive relationships

- **Requested**. This is where an agent explicitly asks for help with an action. For example, via the Contract Net Protocol.
- **Action equality**. This is where the agents recognise that they are both planning to perform the same action, and so one can perform it and save the other agent the work. For example, two students may realise that they are both planning to post the same question to the KEATS forum, and so agree that only one of them will do this. 我们都打算执行一个动作，并且相互知道，这样的话我们中的一个能独立去完成任务，从而节省另一个人的努力
- **Consequence**. This is where an action that one agent is planning has a side effect of achieving one of another agent's goals. For example, if I plan to drive the car I share with my partner tomorrow, I may need to first fill it up with petrol. My partner may have the goal that the car has some petrol in it, and so I am achieving their goal as a side effect of my action to drive the car. 我计划中的动作的执行可以导致你的某个目标的完成，这是我动作的副作用，这样就解除了你去完成该目标的需要
- **Favour**. This is similar to the consequence relationship. The action that one agent is planning has a side effect of contributing to another agent's goals, for example by achieving some precondition for an action in the other agent's plan. My flatmate may have a goal to cook a nice dinner, and the plan for this requires that milk is available. If I'm planning to do the weekly shopping, which involves buying milk, this helps my flatmate to achieve their goal. 我计划中的部分有这样的副作用，其执行会对你目标的实现产生贡献，也许是使目标的实现变得更容易（例如，通过实现你动作的一个前提条件）

我们设计的系统希望利用积极作用，消除消极作用

One approach agents can use to coordinate their activities is **multi-agent planning**.

- **Centralised planning for distributed agents**

  In this approach, there is a coordinating agent who has all the information about each individual agent's private goals and available actions. The coordinating agent uses this information to find an effective global plan for all of the agents.

- **Partial global planning**

  Here, the agents plan for themselves but by sharing this information they also build up a (partial) view of the global plan of all agents, which they can use to improve their own plans. They then share their revised partial view of the global plan with the other agents, allowing them to improve their own plans, who then share their revised partial view of the global plan, and so on.

- **Centralised merging of individual plans**

  In this approach, the agents first plan for themselves (according to the information they have about their own goals and actions) and then share their local plans with a coordinating agent. This coordinating agent then must merge these plans to create an efficient global plan for all the agents.

5.3.1 Normative multi-agent systems

In the multi-agent planning approach, we saw how this can be achieved by the agents sharing information about their actions and goals (either directly with each other, or with a centralised coordinating agent), but often agents may not want to share this private information with other agents in the system.

An agent can also have expectations about the behaviour of other agents because of knowledge it has about the system they are working in. An agent can expect certain behaviours because it is *normal* (i.e., a conventional way to act) or because it is *normative* (i.e., it is an expected standard set by the system). This is the area of **normative multi-agent systems**.

Norms can be **prescriptive** (that is, standards set by the system, so normative). Some examples:

- In the UK, you must drive on the left (unless you are on a one way street or on private property).
- At King's, you must not plagiarise, cheat or collude on assessments.

Or norms can be **conventions** (that is, normal expected behaviours). Some examples:

- If it's your friend's birthday, you should buy them a present.
- People in the UK typically have dinner around 7pm.
- If you use an escalator in the UK, you should either stand on the right or walk up it on the left.

5.3.1.1 Prescriptive Norms

< *target, deontic-component, context, content, punishment* >

where

target defines the agents to whom the norm applies;

deontic-component denotes the type of norm, either obligation, prohibition or permission;

context defines when the norm applies;

content is the behaviour that is constrained by the norm;

punishment is what happens if an agent gets caught violating the norm.

- An **obligation** norm says that "in the given *context*, an agent of the *target* type is obliged to perform (i.e., they must perform) the *content* behaviour, and if they get caught not doing so they will receive the *punishment*."

For example the norm

< *all-drivers, obligation, at-red-light, stop, 3-points-on-licence* >

says that all drivers must stop when they're at a red light, and if they're caught not doing so they will get 3 points on their licence.

- A **prohibition** norm says that "in the given *context*, an agent of the *target* type is prohibited from performing (i.e., they must not perform) the *content* behaviour, and if they get caught doing so they will receive the *punishment*."

For example the norm

< *all-drivers, prohibited, on-public-UK-road, drive-on-right, 6-points-on-licence* >

says that all drivers must not drive on the right when they're on a public road in the UK, and if they're caught doing so they will get 6 points on their licence.

- **Permission** norms are slightly different. These do not say that a *target* agent must or must not do something, but rather that it is permitted to do the *content* behaviour in the given *context*. Permission norms are typically used to identify exceptions to prohibitions and obligations, and as such do not normally have a *punishment* associated with them.

For example the norm

< *all-drivers, permitted, on-one-way-UK-street, drive-on-right, __* >

says that all drivers are permitted to drive on the right if they are on a one way street in the UK.

Another example, the norm

< *ambulance-drivers, permitted, at-red-light AND on-way-to-emergency, go, __* >

says that ambulance drivers are permitted to drive through red lights as long as they are on their way to an emergency.

An agent's reasoning about what to do is affected by its knowledge of the norms in the system:

An agent knows that if it is caught violating a prescriptive norm, this will lead to a punishment (i.e., some decrease in its utility).

If an agent violates a convention this will not cause direct punishment, but it may indirectly cause some decrease in utility by hindering the agent's coordination with other agents in the system.

So an agent can choose to violate a norm, but in doing so it should weigh up any potential negative consequences of this.

Consider the following prescriptive norm:

< *all-drivers, prohibited, on-UK-highway, drive-over-70-mph, fine-£50* >

This says that all drivers on UK highways are prohibited from driving over 70 miles per hour, and if they get caught violating this norm they will get a £50 fine.

Let's look together now at some different reasons a driver on a UK highway might have for disobeying this norm:

< *all-drivers, prohibited, on-UK-highway, drive-over-70-mph, fine-£50* >

Ignorance: the agent may not be aware of the norm. For example, if the driver doesn't normally drive in the UK and they haven't seen any of the road signs.

Lack of understanding: the agent may not understand what the norm means. For example, if the driver does not know what counts as a UK highway.

Punishment is too low: if the punishment is too low, then the agent has little incentive to obey the norm (there will be little impact on their utility if they get caught). For example, if the driver is a millionaire they may not care whether they get a £50 fine or not.

Probability of getting caught is too low: the agent may be willing to risk the punishment if they believe there is little chance that they will get caught. For example, if the driver believes they can out run any police.

Another goal may be more important: if obeying the norm will mean that the agent cannot achieve an important goal, the loss in utility from not achieving this goal may be greater than any punishment from disobeying the norm. For example, if the driver's partner is about to give birth and the driver is racing to get to the hospital to meet them, it may be so important to the driver that they are with their partner for the birth that they are willing to take the penalty if they get caught disobeying the norm.

There may be no choice: an agent may not have the capability to comply with the norm. For example, if the driver's accelerator pedal gets stuck!

These reasons hint at some of the challenges involved in designing appropriate prescriptive norms to govern a system:

Agents need to know the norms. There must be an efficient way of communicating the norms to all the agents in the system. This becomes more complicated if the system is open (meaning that that agents can freely choose whether to join or leave the system at any time), as it becomes harder to ensure that the agents that are currently part of the system are all aware of the norms.

Agents need to understand the norms. Think back to the discussions earlier on in this topic about agent communication, and how the agents need a shared understanding of the meaning of the terms that they exchange. The agents need a shared understanding of the norms in the system and of the terms that are used to define them.

Monitoring of agents' adherence to the norms needs to be effective. There is a balance to be struck here between the amount of effort involved in monitoring the agents' behaviours, and the likelihood that any agents disobeying a norm will be caught.

Punishments need to be suitable. Punishments need to be sufficient that they will dis-incentivise agents from disobeying the norms, but shouldn't be so great that they completely curtail the agents' autonomy. If the agents (or the agent designers) view the punishments as unreasonably severe, they may decide that the system does not allow them sufficient individual choice to disobey norms when they feel it is appropriate to do so.

The other key challenge is **how to design a suitable set of prescriptive norms that will encourage the types of behaviour that are desired from the system.** This is similar to the challenges faced by the designer of a reactive agent, where it can be very hard to come up with an appropriate set of individual behaviours that, when working together in a subsumption architecture, will produce the desired behaviour. It can be very hard to understand how all the different possible interactions between the norms and the different agents in the system will impact on the overall behaviour of the system.

While prescriptive norms are designed for a system, conventions *emerge* from the behaviours of the agents in the system. Typically, conventions emerge over time due to agents copying each others' behaviours. For example, imagine you visit a country you've not been to before, and when you go on an escalator for the first time you notice that everyone seems to be either standing on the left or walking on the right. You are likely to adopt the same behaviour, and so will be better coordinated with the other people using the escalator.

That finishes our discussion of different mechanisms agents can use to help coordinate their behaviour. The last thing we will cover in this topic is trust and reputation.

5.4 Trust and reputation

该信任哪个agent呢

There are two main ways an agent can determine how reliable it believes another agent to be: based on its own experience (trust) and based on the experience of others (reputation).

Trust. An agent can determine its measure of trust in another agent according to its past experience with that agent. if I've used two different delivery companies in the past, one of which consistently delivered later than promised and the other of which was always on time, I will trust the company which is always on time more than I trust the one that is always late.

Reputation. This is derived from the experience of other agents. Did the agent do a good job for others in the past? For example, if a company receives consistently negative reviews we would view that company as having a poor reputation. Or if agent Ag1 tells Ag3 about their bad experiences with Ag2, this may affect Ag3's view of Ag2's reputation.

5.4.1 FIRE Model

how an agent can calculate its trust of another agent?

The **FIRE model** is a model of trust and reputation that agents can use to help them decide who they might delegate tasks to

FIRE uses four different sources of information to judge another agent's reliability. Let us refer to the agent who is judging another agent's reliability as the *evaluator*, and the agent whose reliability is being judged as the *target* agent. The four sources of information that the evaluator can use are:

Direct experience. The evaluator can use the knowledge it has of its previous interactions with the target agent. Trust derived from direct experience is called **Interaction Trust**.

Witness information. The evaluator can use information about other agents' experiences of interacting with the target agent (these other agents are the *witnesses*). Trust derived from witness information is called **Witness Reputation**.

Role-based rules. This is where the evaluator can have certain expectations about the target agent because of some role that the target holds. For example, if the evaluator and the target agent are both part of the same organisation, the evaluator may be inclined to trust the target agent. Trust derived from role-based rules is called **Role-based Trust**.

Third-party references provided by the target agent. This is similar to witness information, except the target agent itself collects the references and can provide these to the evaluator. Trust derived from third-party references is called **Certified Reputation**.

E.g:

An agent, Ag0, determines who to delegate a task to using a simplified FIRE trust model, in which it uses only interaction trust to compare agents, only considers a single quality-of-service term, and uses a recency function such that a rating of an interaction on one day is given twice the weight of an interaction from the day before. Agents Ag1, Ag2 and Ag3 offer to perform the task.

- 10 days ago, Ag0 delegated to Ag1 with rating 0.5 and to agent Ag3 with rating 0.8.
- 8 days ago, Ag0 delegated to Ag1 with rating 0.9 and delegated to Ag2 with rating 0.6.
- 7 days ago, Ag0 delegated to Ag2 with rating 0.3.

Which of the following statements are true?

解析：

Ag0's trust in Ag1

Ag0 has had two prior interactions with Ag1: 10 days ago with a rating of 0.5; and 8 days ago with a rating of 0.9. We need to find the weighted average of these two ratings, where the weights applied to the ratings are such that the rating of an interaction on one day is given twice the weight of an interaction from the day before. Since the interaction rated 0.5 came two days before the interaction rated 0.9, we need to weight the 0.9 rating at 4 times the 0.5 rating. (If the rating of an interaction on one day is given twice the weight of an interaction from the day before, then the rating of an interaction must be given four times the weight of an interaction from two days ago.) Ag0's trust in Ag1 is calculated as the weighted average of the ratings (i.e., the sum of the weighted ratings divided by the sum of the weights):

((1 x 0.5) + (4 x 0.9))/(1+4) = 4.1/5 = 0.82 （最近一天是前一天权重的两倍）

Ag0's trust in Ag2:

Ag0 has had two prior interactions with Ag2: 8 days ago with a rating of 0.6; and 7 days ago with a rating of 0.3. We need to find the weighted average of these two ratings, where the weights applied to the ratings are such that the rating of an interaction on one day is given twice the weight of an interaction from the day before. Since the interaction rated 0.6 came one day before the interaction rated 0.3, we need to weight the 0.3 rating at 2 times the 0.6 rating. Ag0's trust in Ag2 is calculated as the weighted average of the ratings (i.e., the sum of the weighted ratings divided by the sum of the weights):

((1 x 0.6) + (2 x 0.3))/(1+2) = 1.2/3 = 0.4

Ag0's trust in Ag3:

Ag0 has only had one prior interaction with Ag3: 10 days ago with a rating of 0.8. As there is only one interaction to consider, the weighted average of its rating is straightforward, it's simply equal to the rating. So Ag0's trust in Ag3 is 0.8.

Since Ag0 trusts Ag1 more than it does either Ag2 or Ag3, it selects Ag1 to delegate the task to.

Wee6: Game theory, and negotiation

6.1 Game Theory

6.1.1 Dominant strategy

优势策略：a strategy s_i is dominant for agent i if, no matter what the other agents do, agent i can't do any better than play s_i.

A **rational** agent will play a dominant strategy(if one exists).

		Player 2	
		Split	Steal
Player 1	Split	5k 5k	10k 0
	Steal	0 10k	0 0

Is there a dominant strategy for player 1?? **Steal**

如果P2选择Split，那对于P1而言最好的选择就是Steal，因为能获得10k；如果P2选择了Steal，对于P1而言选Split和Steal都一样，因为都只能获得0；综上所述，dominant strategy for P1 is **Steal**

E.g:

Consider the following pay-off matrix.

		Player 2	
		S1	S2
Player 1	S3	-1 -1	2 1
	S4	3 2	-1 -1

Which of the following statements is true?

Your answer :

There is no dominant strategy for Player 1.

Response:

Correct!

If Player 2 plays S1, then the best thing Player 1 can do is S4 (for utility of 2).

If player 2 plays S2, then the best thing Player 1 can do is S3 (for utility of 1).

Thus there is no dominant strategy for Player 1 in this situation (since there is no strategy that is the best response to both of Player 2's strategies).

6.1.2 Dominanted strategy

劣势策略，我们希望删除劣势策略

A rational agent will never play a dominated strategy

6.1.3 Nash Equilibrium

双方都不能改变策略从中获益(better off)

P1和P2都选择Steal，是Nash Equilibrium。因为假设P2不改变选择，P1选择了Split，依然还是获益0；假设P1不改变选择，P2选择了Split，依然获益是0，因此是Nash Equilibrium。

P1选择Split，P2选择Steal，是Nash Equilibrium。因为假设P2不改变选择，P1选择了Steal，依然还是获益0；假设P1不改变选择，P2选择了Split，获益减少到5K，没有从中获益，因此是Nash Equilibrium。

技巧：横着找出每一行第二个数字最大的，在底下划条线；竖着找出每一列第一个数字最大的，在底下画一条线，最后找出有两条线的组合就是纳什均衡。

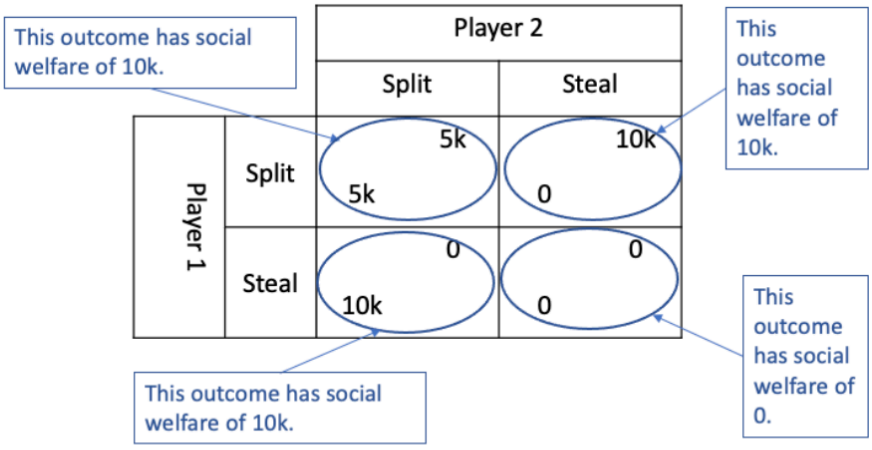
6.1.4 Pareto Optimal

如果**不存在**其他的协商结局使至少一个Agent 更好而没有使其他 Agent 更差，则称这个协商结局具有Pareto 效率。所以如果协商结局没有达到Pareto 效率则存在另一个协商结局使得至少有一个Agent 更满意，同时使其他所有的Agent 至少满意

1. 达到如果再改变就会使一方受益，另一方受损的最优临界点
2. 两者同时达到最好，没有比这个状态更好的

P1 Steal，P2 Steal就不是Pareto Optimal，因为P2可以选择Split来拿到10K的收益，但同时P1的收益没受到损害

6.1.5 Maximising social welfare



There are three outcomes that maximise social welfare in the Golden Balls example: Player 1 splits and Player 2 steals; Player 1 steals and Player 2 splits; both split.

6.2 Negotiation

6.2.1 Task Oriented Domains (TOD)

面向任务领域协商

Initial encounter:

< {shop, cook, washUp}, {getVideo, tidy, buyWine} >

where

c({shop, cook, washUp}) = 12

c({getVideo, tidy, buyWine}) = 10

Possible deal:

δ = < {cook, tidy, washUp}, {shop, getVideo, buyWine} >

where

c({cook, tidy, washUp}) = 9

c({shop, getVideo, buyWine}) = 6

在Initial encounter中，Agent1初始被分配的是shop, cook, washUp; Agent2初始被分配的是getVideo,tidy,buyWine

其中Agent1被分配任务的cost是12，同理Agent2是10；

在新分配的任务中，Agent1不再需要做shop而需要做tidy，新deal是cook, tidy,washUp;同理，Agent2的新deal是shop，getVideo,buyWine。

所以Agent1的Utility是12-9=3；Agent2的是10-6=4；

因此Agent1做新的deal可以节省3的cost，Agent2做新的deal可以节省4的cost；新deal比初试deal要更合适。

```
Utility = a  - b

a = the cost of what was original allocated to the agents in the initial encounter
b = the cost of agent has been allocated in the new deal

一项交易的Utility代表Agent从该交易中获得收益的多少
如果Utility是负数，则证明代理执行最初分配的任务比执行交易中分配的任务更好；如果是正数，则证明执行新任务比较好
```

- conflict deal: If agreement is not reached then we say that the result is the **conflict deal**. It is assumed that the conflict deal is the worst outcome。**对于conflict deal的utility是负的**
- **Individual rational**: A deal δ is said to be **individual rational** if neither agent prefers the conflict deal to δ.
- **negotiation set**：由交易组成，这些交易(具有个体理性，并且(i)是pareto 最优的。第一个约束背后的直观含义是指,提出对某一 Agent来说比冲突交易更不好的交易是没有意义的(因为这样的话 Agent 宁愿冲突)。第

二个条件背后的直观含义是, 提出一个建议时，如果还有另外的建议对某一 Agent 更好而没有其他Agent 受到损失，则这个建议是没有意义的.

6.2.2 Monotonic Concession Protoco

- 协商进行多轮
- 在第一轮协商中，两个agent同时从negotiation set中提出一项交易
- 如果两个Agent提出的交易分别是δ_1和δ_2,使得或者有$utility_1(\delta_2) >= utility_1(\delta_1)$，或者$utility_1(\delta_1) >= utility_1(\delta_2)$，则达成一致。**即如果有一个Agent发现另一个agent提出的交易至少与自己提出的一样好或者比自己提出的更好。**
- 如果不能达成一致，则协商继续进行另一轮，同时提出建议。在第u+1轮，不允许Agent提出比第u论建议对其他Agent更差的建议
- 如果在某一轮u>0，没有agent做出让步，则协商以conflict deal结束

6.2.3 Zeuthen strategy

决定了参与者在使用Monotonic Concession Protoco是应该如何工作

- Agent的第一个建议应该是什么？ 应该是Agent的第一个建议应该是他最希望的交易
- 在给定的一轮协议中，谁应该让步?
- 如果一个Agent让步，他应该让步多少?

在给定的一轮协议中，谁应该让步?

度量 Agent对冲突风险的意愿。直观上，如果Agent 当前建议的效用与冲突交易的效用差别小，则它更愿意骨冲突的风险。相反，如果一个Agent 当前的建议与冲突交易的差别大，则冲突时，这个Agent会遭受更大的损失，因此它更不愿意胃冲突风险--更愿意让步。

Let's assume that δi is the deal currently proposed by Agi and δj is the deal currently proposed by Agj. Formally, $Agi's$ willingness to risk the conflict deal is equal to

$risk = \frac{utility_i(\delta_i) - utility_i(\delta_j)}{utility_i(\delta_i)}$

分子定义为i当前建议的效用与j的建议对于i的效用的差;分母定义为 Agent i当前建议的效用。在达成一致以前。risk 的值在0~1之同。risk 的值超大（越接近1），表示i由冲交遭受的损失越小，因此更愿意冒冲突风险。反之，risk的值小（越接近0），表示由冲突遣受的损失越大，因此更不愿意冒冲突风险。

Week7: Making Group Decision

7.1 Plurality Voting Protocol

Only takes into account the top preference of each voter

E.g:

Ω={ω1,ω2,ω3}

40 agents have the preference ω1>ω2>ω3

30 agents have the preference ω2>ω3>ω1

30 agents have the preference ω3>ω2>ω1

Under the plurality protocol:

- ω1 gets 40 points.
- ω2 gets 30 points.
- ω3 gets 30 points.

7.2 Condorcet Winner

如果在成对的多数竞赛中，有一个结果击败了所有其他结果，那么它就是孔多塞赢家

Ω={ω1,ω2,ω3}

40 agents have the preference ω1>ω2>ω3

30 agents have the preference ω2>ω3>ω1

30 agents have the preference ω3>ω2>ω1

Pairwise majority contests

ω1 vs ω2: 40 prefer ω1, 60 prefer ω2, ω2 wins

ω1 vs ω3: 40 prefer ω1, 60 prefer ω3, ω3 wins

ω2 vs ω3: 70 prefer ω2, 30 prefer ω3, ω2 wins

So here, ω2 is the Condorcet winner, as it beats both ω1 and ω3 in pairwise majority contests, but under plurality voting, ω1wins.

7.3 Linear sequential majority elections

- There are a series of rounds.
- A pair of outcomes face each other in a pairwise majority election, the winner goes on to the next round.
- The order in which the outcomes face each other is determined by an agenda.

Example: If the agenda is ω4, ω2, ω3, ω1 then the first election is between ω4 and ω2, the winner of that goes onto an election with ω3, and the winner of that goes onto an election with ω1.

E.g1:

Let Ω={A,B,C}. Consider the following voter profile.

3 voters have the preference $A>B>C$

2 voters have the preference $B>C>A$

1 voter has the preference $B>A>C$

1 voter has the preference $C>A>B$

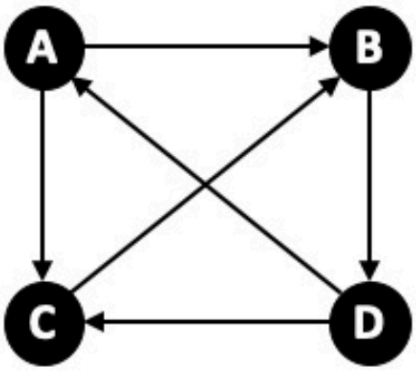
The vote is carried out under the linear sequential majority election protocol with the agenda A,C,B. Which candidate is selected as the winner?

解析

First round: *A* vs *C*. 4 voters prefer A to C; 3 voters prefer C to A. A goes through to the next round.

Second round: *A* vs *B*. 4 voters prefer A to B. 3 voters prefer B to A. A wins.

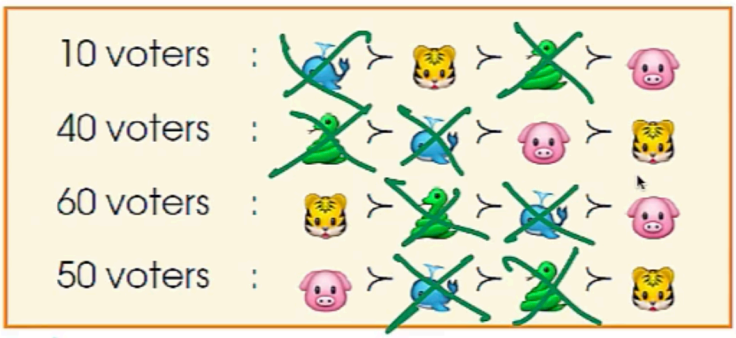
E.g2:

> Consider the following majority graph.
>
> 
>
>
> Which of the following statements are true? (Select all that apply. There is more than one correct answer and you will not be able to move on until you have identified them all.)

- It is possible to fix the agenda so that A wins under the linear sequential election voting protocol.
  - 因为C, D, B, A (notice there is a path from A to B to D to C)；
  - 或D, B, C, A (notice there is a path from A to C to B to D)
- It is possible to fix the agenda so that B wins under the linear sequential election voting protocol.
  - C, A, D, B (notice there is a path from B to D to A to C)
- It is possible to fix the agenda so that C wins under the linear sequential election voting protocol.
  - A, D, B, C (notice there is a path from C to B to D to A)
- It is possible to fix the agenda so that D wins under the linear sequential election voting protocol.
  - B, A, C, D (D->A->C->B)

7.4 Instant Runoff Voting Protocol

Example



Round1. ~~White~~ : 10. Snake: 40
Tiger: 60. Pig: 50

Round2. ~~Snake~~: 40. Tiger: 70.
Pig: 50,

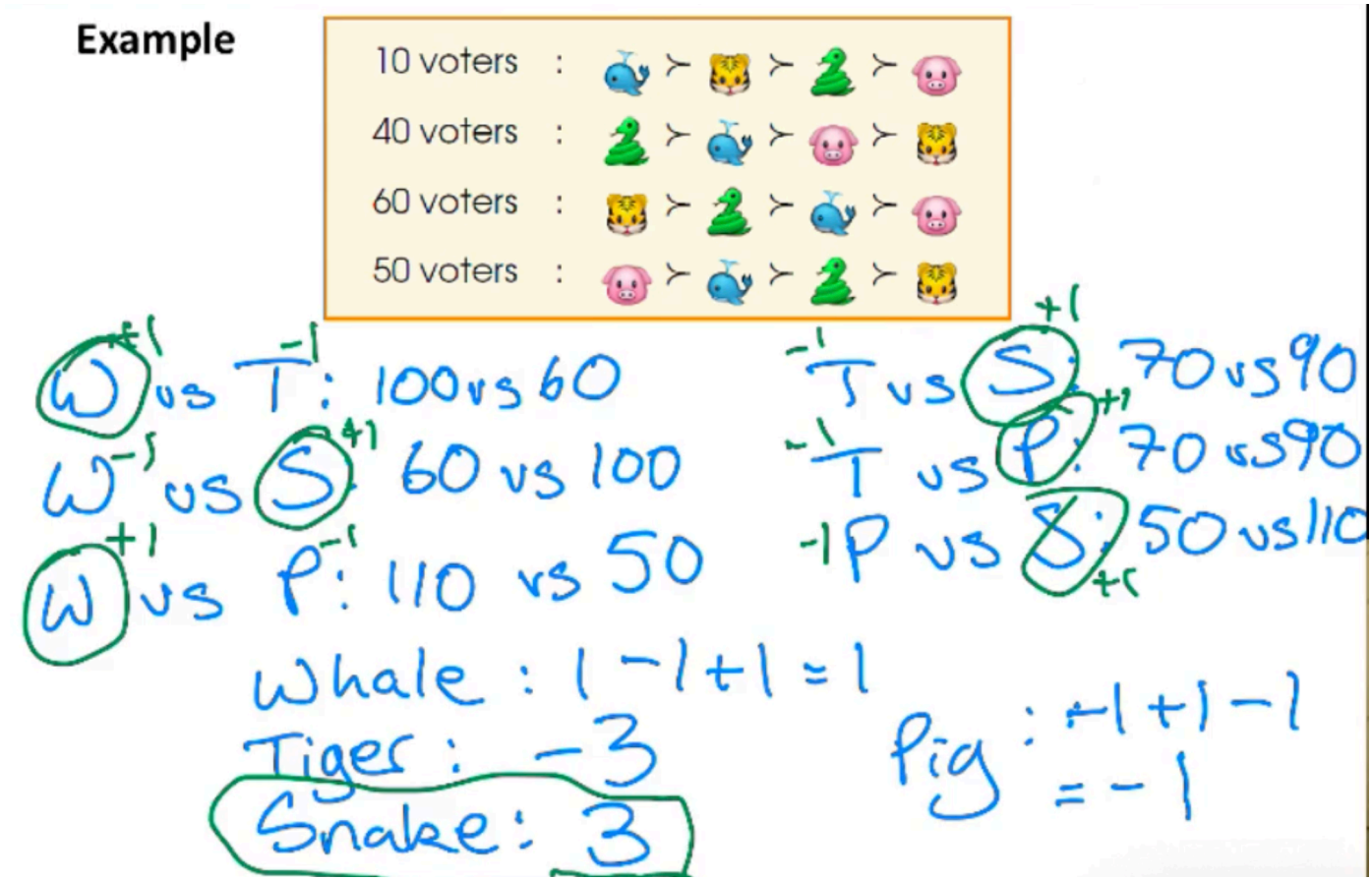
Round3. Tiger: ~~7~~0. Pig: 90

7.4 Copeland rule voting protocol

Under the Copeland rule, all candidates face each other in pairwise majority elections. Each candidate receives:

- 1 point for every pairwise majority election that they win;
- -1 point for every pairwise majority election that they lose;
- 0 points for every pairwise majority election they draw at.

The candidate with the most points is the winner.

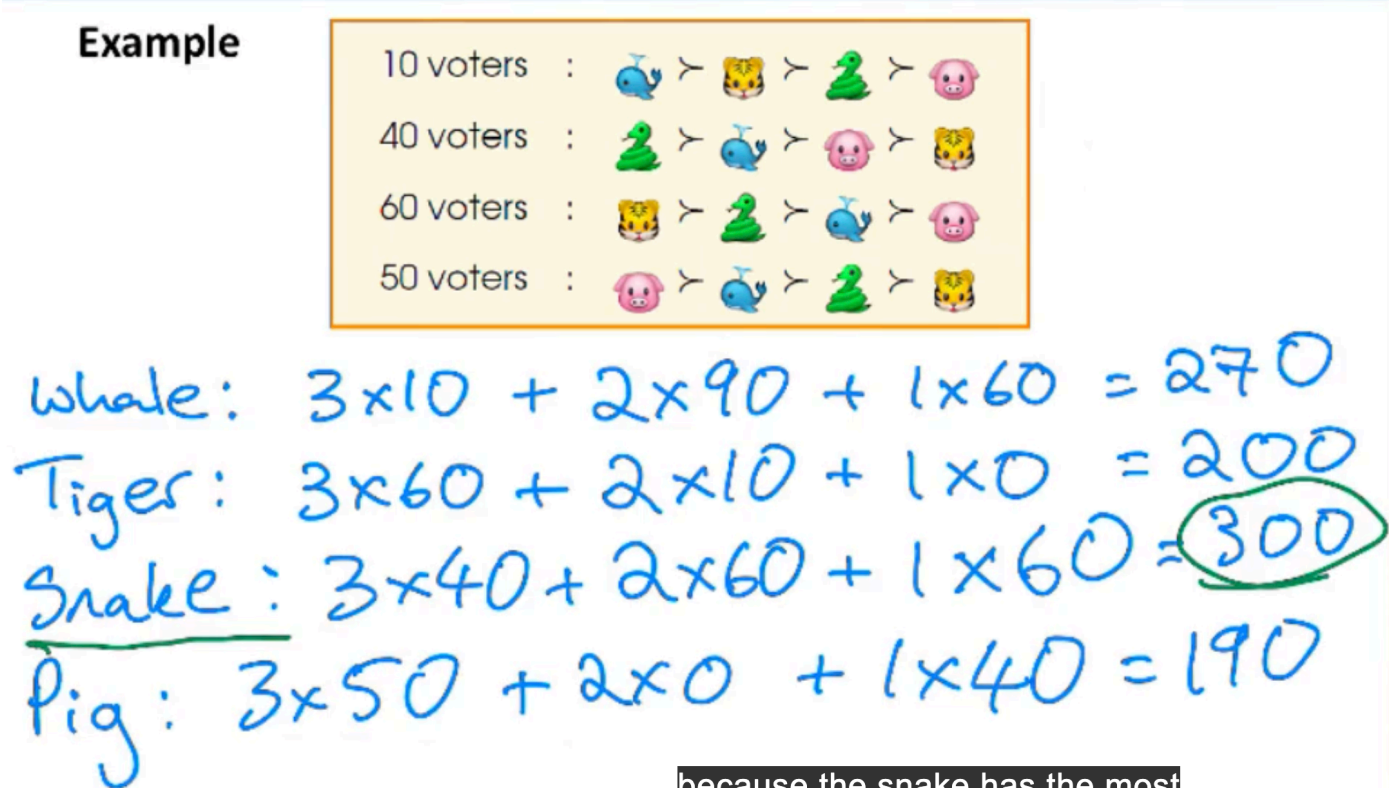


7.5 Borda count voting protocol

Under the Borda count protocol, where m is the number of candidates, each candidate receives:

- $(m-1)$points for each voter that lists them as their first choice;
- $(m-2)$ points for each voter that lists them as their second choice;
- and so on.

The candidate with the most points is the winner.



7.3 Strategy-proofness

防策略(Strategy-proofness)通常被认为是一个非常理想的属性,它要求投票者不能从谎报他们的真实偏好中获益,进而可以抑制社会选择中的策略投票,促使投票者都投出自己的真实选票,使选举结果能够体现人们的真实意愿.

A voting protocol is said to be strategy proof if and only if it is strategy proof for all voters. In other words no one has any incentive to misrepresent their preferences.

- **Dictatorships**: 独裁是指整个社会的总体偏好排序会只由一个人的偏好排序决定，这个人有可能是隐藏在复杂的投票机制下的，不一定是班长那种显而易见的独裁者。If a voting protocol is a dictatorship then it ignores the preferences of all voters except for those of the dictator *i*. **不存在这样的选民，使得他的选择一定为最后结果。*存在某个选举人，只要他认为任意两个候选人的关系是 A>B ，那么即使其它选举人都认为 A<B ，选举结果也一定是 A>B 。**
- **Surjective**: for every candidate there is some profile of voter preferences such that the candidate will win.

  To put this another way, every candidate has a chance of winning.

  This is a desirable property. Imagine we have a voting protocol that is not surjective, this would mean that there is some candidate which could not win -- even if every voter placed this candidate as their top choice.
- **Resolute**: there is always a unique winner (i.e., no ties).

  This is often a desirable property, since applications often require a unique winner.

  Note, the voting protocols we have looked at here can easily be made resolute by adding the extra condition that in the case of a tie a winner will be selected at random.

7.4 Gibbard-Satterthwaite Theorem

If we have 3 or more candidates, any **resolute** voting protocol that is **surjective** and **strategy proof** is a **dictatorship**.

It says that it is not possible to design a voting protocol that is resolute, surjective, strategy proof and not a dictatorship. The voting protocols we have looked at are resolute (assuming ties are resolved randomly), surjective, and not dictatorships, therefore they cannot be strategy proof.

7.5 Weekly Pareto

A voting protocol is said to be weakly Pareto if and only if:

- if there are candidates ωi and ωj such that **all** voters prefer ωiωi to ωj, then ωj will not be selected as the winner.

In other words, if every voter prefers ωi to ωj, the outcome will also reflect this. Hopefully it is clear that this is a desirable property.如果，选举中的每一个代理人比较A和B时都更喜欢A，那么B不可能是最终的选举结果

设“欢喜”表示“（比原来）处境严格变好”，“忧愁”表示“（比原来）处境严格变差”。 资源配置变化后，“有人欢喜，无人忧愁”相对于“无人忧愁”是“更强的”帕累托改善，相对于“所有人欢喜”是“更弱的”帕累托改善。 若无论资源配置如何变化，都不可能“有人欢喜，无人忧愁”，则该状态是“强帕累托最优”。 若无论资源配置如何变化，都不可能“所有人欢喜”，则该状态是“弱帕累托最优”。

7.6 Arrow's Theorem

帕累托最优（Pareto） 对于全部的N个排列顺序，如果A都在B之前，那么最后的结果A一定在B之前。

无关因素独立性（Independence of Irrelevant Alternatives (IIA)） 两个人的相关顺序不变的话，其他参与者的相对位置发生了变化，那么这两个人的相对位置也不会发生变化。 投票问题（voting theory）中，假如有四个候选人，即A、B、C和D，如果大多数民众（即超过一半）一致认为A优于C，那么B和D的相对位置发生了变化，也不会影响大多数民众的偏好上A优于C。

非独裁（Nondictatorship） 不存在这样的选民，使得他的选择一定为最后结果。

没有任何一种选举方式可以同时满足以上三个条件。对于具有两名以上候选人的选举，任何满足帕累托效率和IIA要求的社会福利职能都是独裁的——Arrow’s公理 It says that it is not possible to design a voting protocol that is weakly Pareto, independent of irrelevant alternatives and not a dictatorship.

We've now seen two impossibility results (Arrow's Theorem and the Gibbard-Satterthwaite Theorem) that tell us it is impossible to design democratic (non-dictatorship) voting protocols that have intuitively desirable properties.

7.7 Single Peak Preferences and Median voter rule

Gibbard-Satterthwaite定理和Arrow's Theorem都假设选民可以给予他们喜欢的候选人的任何偏好（即，他们可以以任何顺序列出它们）。如果我们限制允许选民提出的偏好命令，那么这些负面结果就不成立。

单峰偏好理论(Single Peak Preferences)是限定每个选民的偏好只能有一个峰值。

所谓单峰偏好，是指选民在一组按某种标准排列的备选方案中，有一个最为偏好的选择，而从这个方案向任何方面的偏离，选民的偏好程度或效用都是递减的。

如果在一个多数决策的模型中，个人偏好都是单峰的，则反映中间投票人意愿的那种政策会最终获胜，因为选择该政策会使一个团体的福利损失最小。

Week8: Allocating Scarce Resources

8.1 English Auction

优势策略（dominate strategy）：Agent相继以略高于当前叫价的价格叫价，直到叫价到达他们当前的估价。bid the smallest amount possible above the current bid until the current bid price reaches your true valuation of the item being auctioned

E.g:

Consider an English auction with two bidders, ag1ag1 and ag2ag2, who are each playing their dominant strategy.

We have

- v1 = 7.5
- v2 = 300

where vi is agent agi's true valuation of the item.

The minimum bid increment allowed is 1, and the auctioneer starts the bidding at 5.

How much will the auctioneer receive? (In other words, how much will the winning bidder end up paying for the item?) Assume that ag1 is always faster to bid than ag2 (that is, if both agents are willing to make a bid of b, ag1 will always be the agent who bids b).

Answer: 8

一旦到了7，Ag1就会停止拍卖，因为下一次拍价就是8了超过了Ag1的预期7.5；由于还没达到Ag2的期望，所以Ag2出价8，无人竞价，最终以8结束。

winner's valuation v_w

expected revenue $b + \epsilon$

2nd highest valuation b

winner's valuation=300; 2nd highest valuation=7.5;

8.1.1 Shill bids

虚假出价：A shill bid is a bid placed by the auctioneer, or by agents in collusion with the auctioneer, with the aim of increasing the price paid by the winner.

8.2 Dutch Auction

递减拍卖

8.2.1 Risk Averse

不愿意长时间等待，减少风险，减少收益

Risk Averse不愿意长时间的等待，因此很早就成交了，500w开始，490w就结束了，对于拍卖方来说，收益大。

8.2.2 Risk Seeking

愿意长时间等待时机，增大风险，增高收益

Risk Seeking愿意长时间的等待，因此很晚才成交，500w开始，50w才结束，对于拍卖方而言，收益少。

8.3 First Price Sealed Bid Auction

拍卖只有一轮，卖给出价信封中最高的那位买家

优势策略（dominate strategy）：低于真正价格叫价

8.4 Vickrey Auction

拍卖只有一轮，卖给出价信封中最高的那位买家，但只需要用第二高的价格购买即可

优势策略（dominate strategy）：**出真实的估价来叫价**

如果出高于估价的价格来叫价，有可能会拍到，但是会付出更多的价格购买，即使买到也会损失

如果出低于估价的价格来叫价，即便拍卖到了藏品，也要付出第二高的价格来购买，出低价没有得到相应的回报

注意，Vickrey 拍卖会使反社会行为成为可能。假设你希望得到某物品并且自己估价为90美元，但是，你知道有其他 Agent也想得到它并且估价为 100 美元。因为给出真正有效的优势策略，你不可能有比90美元更好的出价；你的对手出价100 美元并且得到了这个物品，但是只要支付90 美元。对这样的结果你可能不会高兴：你可能想要“惩行”一下获胜的对手，应该怎么做呢？假设你出价99 美元而不是90 美元，你仍然会把这个物品输给你的对手-但是他要比你以完全真正的叫价多支付，美元。当然，要做到这一点，你必须对对手的如何叫价非常自信-你不想出价99 美元而发现你的对手只出价95 美元，你要以比你的估价高5美元留下这个物品。这种行为会在商业环境下出现，其中一个公司没有办法与另一个公司直接竞争，但是可以利用它们的见解来迫使对手破产

8.5 Combinatorial Auctions

组合拍卖

如果有多个资源，并且多个资源之间可能存在依赖关系，那么对于青睐这些资源的用户而言，购买多个相互关联的资源的性价比可能要优于单独的购买一个资源，对于平台而言的话，也可能会利益最大化。那么，如何分配这一系列相互关联的资源就是一个问题了，**组合拍卖**就诞生了！

Suppose we have the following situation:

$\mathcal{Z} = \{z_1, z_2, z_3\}$ (i.e., the items up for auction are z_1, z_2 and z_3)

$Ag = \{ag_1, ag_2\}$ (i.e., there are two bidders: ag_1 and ag_2)

$v_1(\emptyset) = 0$ (i.e., $\emptyset$ is worth 0 to ag_1)

$v_1(\{z_1\}) = 1$ (i.e., $\{z_1\}$ is worth 1 to ag_1)

$v_1(\{z_2\}) = 3$ (i.e., $\{z_2\}$ is worth 3 to ag_1)

$v_1(\{z_3\}) = 2$ (i.e., $\{z_3\}$ is worth 2 to ag_1)

$v_1(\{z_1, z_2\}) = 3$ (i.e., $\{z_1, z_2\}$ is worth 3 to ag_1)

$v_1(\{z_1, z_3\}) = 5$ (i.e., $\{z_1, z_3\}$ is worth 5 to ag_1)

$v_1(\{z_2, z_3\}) = 6$ (i.e., $\{z_2, z_3\}$ is worth 6 to ag_1)

$v_1(\{z_1, z_2, z_3\}) = 6$ (i.e., $\{z_1, z_2, z_3\}$ is worth 6 to ag_1)

$v_2(\emptyset) = 0$ (i.e., $\emptyset$ is worth 0 to ag_2)

$v_2(\{z_1\}) = 5$ (i.e., $\{z_1\}$ is worth 5 to ag_2)

$v_2(\{z_2\}) = 1$ (i.e., $\{z_2\}$ is worth 1 to ag_2)

$v_2(\{z_3\}) = 1$ (i.e., $\{z_3\}$ is worth 1 to ag_2)

$v_2(\{z_1, z_2\}) = 5$ (i.e., $\{z_1, z_2\}$ is worth 5 to ag_2)

$v_2(\{z_1, z_3\}) = 5$ (i.e., $\{z_1, z_3\}$ is worth 5 to ag_2)

$v_2(\{z_2, z_3\}) = 1$ (i.e., $\{z_2, z_3\}$ is worth 1 to ag_2)

$v_2(\{z_1, z_2, z_3\}) = 5$ (i.e., $\{z_1, z_2, z_3\}$ is worth 5 to ag_2)

Which of the following allocations produces the highest social welfare?

The allocation ({z2,z3},{z1}) gives the highest social welfare of:

v1({z2,z3})+v2({z1})=6+5=11

8.6 Vickrey-Clarke-Groves mechanism(VCG)

- each agent submits a valuation function to the auctioneer (step 1 below),
- the auctioneer works out the allocation that will maximise the social welfare (step 2 below) and makes this allocation (step 3 below),
- the amount each agent must pay is calculated as in step 4 below.

VCG mechanism

1. Each agent $ag_i \in Ag$ simultaneously declares their valuation function $\hat{v}_i$. (This is denoted $\hat{v}_i$ to indicate that they may choose to declare something different to their true valuation v_i.)
2. The auctioneer computes the allocation $\langle Z_1^*, \ldots, Z_n^* \rangle$ which maximises social utility, according to the valuation functions $\langle \hat{v}_1, \ldots, \hat{v}_n \rangle$ declared in step 1.
3. The goods are allocated according to $\langle Z_1^*, \ldots, Z_n^* \rangle$.
4. Each agent ag_i pays the amount p_i to the auctioneer, where p_i is computed as follows.
   - Let $\langle Z_1', \ldots, Z_n' \rangle$ denote the allocation that would have been chosen at step 2 had agent ag_i declared (in step 1) the valuation $\hat{v}_i(Z) = 0$ for all $Z \in \mathcal{Z}$ and every other agent apart from ag_i had declared the same valuation as the did in step 1.
   - Then $p_i = \underbrace{\sum_{j \in Ag: j \neq i} \hat{v}_j(Z_j')}_{\text{Social welfare without } i} - \underbrace{\sum_{j \in Ag: j \neq i} \hat{v}_j(Z_j^*)}_{\text{Total value to other agents}}$.

P_i解释：Agi参加前- Agi参加后的social warfare

Social welfare without i: 如果i直接不参与竞拍，即只有其他代理参与的情况下，拍卖者根据social welfare最大化给出的最优分配，算得没有i的情况下的和

Toal value to other agents：所有代理参与到竞拍中，拍卖者根据social welfare最大化给出的最优分配。**分配后将i的报价忽略（设为0）之后计算的价值和**

E.g:

Suppose we have two goods a and b and two agents ag1ag1 and ag2ag2 who can each obtain at most one item. The agents declare the following true valuation functions:

$\hat{v}$ 1({a})=v1({a})=4

$\hat{v}$ 1({b})=v1({b})=5

$\hat{v}$ 2({a})=v2({a})=7

$\hat{v}$ 2({b})=v2({b})=9

- Which allocation of goods is made?

  最大化social welfare原则来讲，方案({a},{b}) 最佳，a卖给ag1，b卖给ag2

- How much does each agent pay to the mechanism?

假设Ag1没参与，则应分配({},{b})来满足最大化sw; 假设Ag2没参与，则应分配({b},{})来满足最大化sw

Ag1 should pay = Ag1不参与竞拍的最大sw之和 - 都参与竞拍并忽略Ag1的最大sw之和 = ({},{b}) - ({a},{b})忽略{a} = 9 - 9 =0；

Ag2 should pay = Ag2不参与竞拍的最大sw之和 - 都参与竞拍并忽略Ag2的最大sw之和 = ({b},{}) - ({a},{b})忽略{b} = 5 - 4 =1；

- What utility does each agent get?

  Ag1 utility = 最佳方案({a},{b}) 中Ag1应付的 - Ag1 应付的 = 4 - 0 = 0；

  Ag2 utility = 最佳方案({a},{b}) 中Ag2应付的 - Ag2 应付的 = 9 - 1 = 8；

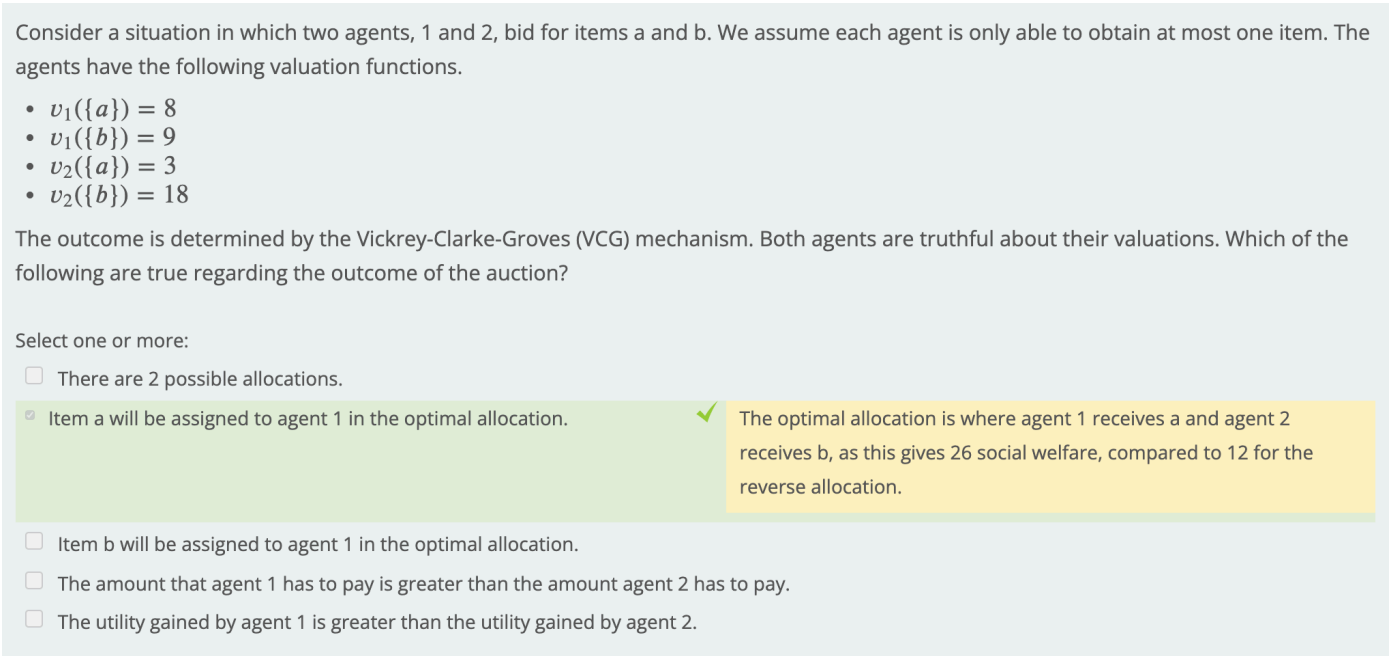
- Can agent ag1 improve its utility by lying and declaring valuation $\hat{v}$ 1≠v1? Hint: think about the possible cases.

Case 1: ({a},{b})计算utility = 4

Case 2: ({b},{a})计算utility = 3

所以不可能

Like with the Vickrey auctions, the **dominant strategy** for the VCG mechanism is to submit your true valuation at step 1. This is a desirable property, since it means that the bidders do not need to expend any effort in reasoning about what valuation to submit, they can do no better than to submit their true valuation.



a和b只能分配给一个人，{a}{b}

Week9: Reasoning with Arguments

An argument has a claim and a support. (Sometimes we refer to the claim as the argument's conclusion, and sometimes we refer to the support as the argument's premises.) The support of an argument provides reasons to believe in the claim.

- Claim：结论
- Support: 前提
- Support is equal to the claim called **atomic arguments**, 比如$\langle\{a\},a\rangle$，support:{a}; Claim: a
- When the claim of an argument A contradicts an argument B we say that A **attacks** B.
    - **Rebuttal**: this is when the claim of an argument ax contradicts the claim of an argument ay, and so leads to a symmetric attack (each argument attacks the other).
    - **Undercut**: this is when the claim of an argument ax contradicts (some part of) the support of an argument ay, leading to a one way attack from ax to ay.
    - 论据之间的攻击性，可以帮助我们决定哪些论证的子集是合理的，可以集体接受的

Let Δ be a database of logic formulae. An **argument** constructed from Δ is a tuple $\langle\Phi,\phi\rangle$ such that:

- $\Phi\subseteq\Delta$, 论点必须来自数据库
- $\Phi\nvdash\bot$,
- $\Phi\vdash\phi$, and （支持必须包含主张）
- there is no $\Phi'\subsetneq\Phi$ such that $\Phi'\vdash\phi$.

What this definition says is that:

- the support of the argument must come from your database,
- the support must be consistent,
- the support must entail the claim, and
- the support is a minimal set (under set inclusion) that entails the claim.

For an argument $argx=\langle\Phi,\phi\rangle$, we often refer to the claim of $argx$ as $claim(argx)=\phi$, and refer to the support of $argx$ as $support(argx)=\Phi$.

9.1 Attacks Between Arguments

Two types of attacks: rebuttals and undercuts

Let $\langle\Phi1,\phi1\rangle$ and $\langle\Phi2,\phi2\rangle$ be arguments.

- $\langle\Phi1,\phi1\rangle$ **rebuts反驳** $\langle\Phi2,\phi2\rangle$ if and only if $\phi1\equiv\neg\phi2$
- $\langle\Phi1,\phi1\rangle$ **undercuts削弱** $\langle\Phi2,\phi2\rangle$ if and only if there is some $\phi\in\Phi2$ such that $\phi1\equiv\neg\phi$

Consider the following arguments.

$arg_1 = \langle \{a, b, a \wedge b \rightarrow c\}, c \rangle$

$arg_2 = \langle \{d, d \rightarrow \neg b\}, \neg b \rangle$

$arg_3 = \langle \{g, g \rightarrow \neg c\}, \neg c \rangle$

Which of the following statements captures all of the attacks that exist between these arguments?

Your answer:

arg_1 rebuts arg_3, arg_3 rebuts arg_1 and arg_2 undercuts arg_1.

Response:
Correct.

$claim(arg_1) \equiv \neg claim(arg_3)$, and so arg_1 rebuts arg_3.

$claim(arg_3) \equiv \neg claim(arg_1)$, and so arg_3 rebuts arg_1.

$claim(arg_2) \equiv \neg b$ and $b \in support(arg_1)$, and so arg_2 undercuts arg_1.

9.2 Argument Evaluation

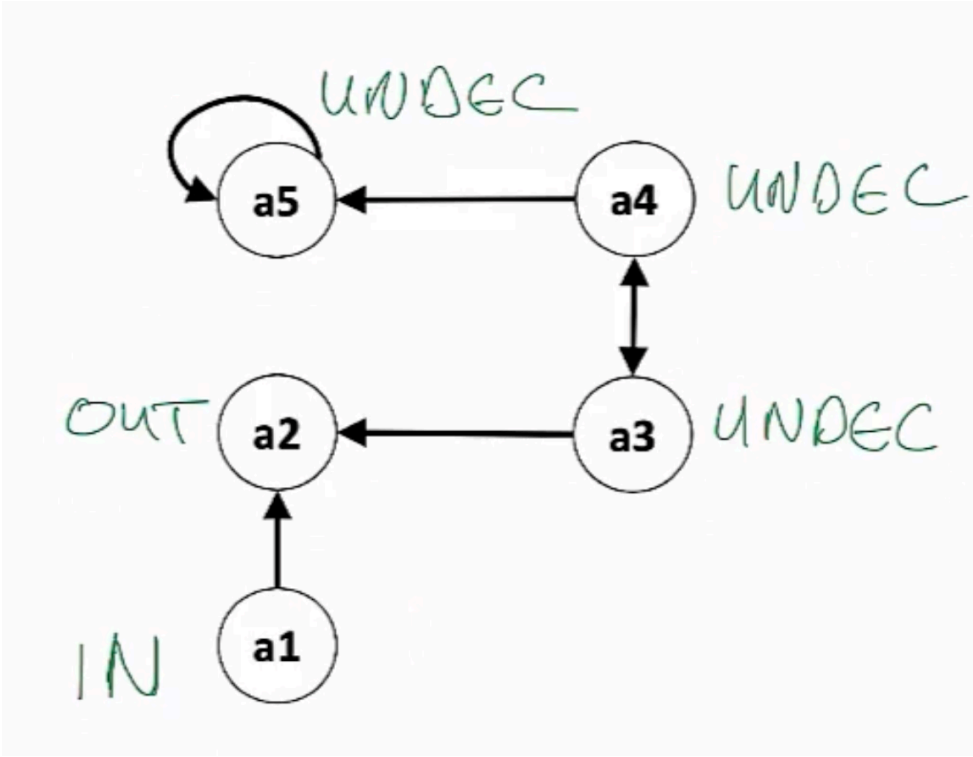
9.2.1 abstract argumentation graph

- 当且仅当一个论点的**所有**攻击者都被标记为OUT时，该论点才被标记为IN。1️⃣
- 当且仅当一个论据**至少有一个**攻击者被标记为IN时，该论据才被标记为OUT。2️⃣
  - 论证h没有攻击者，因此显然是可接受的（IN）
  - 因为h是可接受的，并且h攻击a，因此a是不可接受的论证，称为OUT
  - 同样，因为h是可接受的，并且h攻击p，因此p是不可接受的OUT
  - 因为p是不可接受的OUT，它是q的唯一攻击者，因此q是可接受的IN
- 满足1️⃣和2️⃣的被称为COMPLETE LABELING

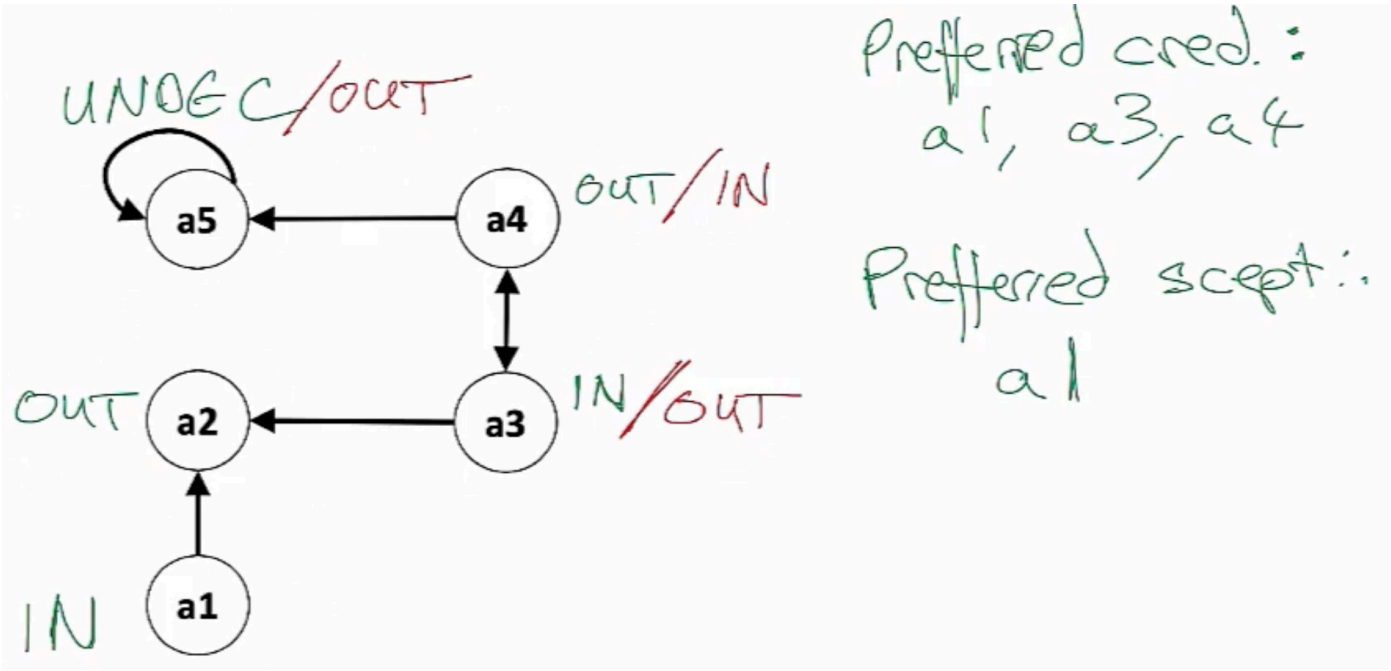
Different **argumentation semantics** place different constraints on which of the complete labellings are valid.

- Under the **complete semantics**, any complete labelling is valid. 任何complete labelling都是complete semantics，包括grounded semantics和preferred semantics也是complete semantics
- Under the **grounded semantics**, only the complete labelling that maximises (with reference to set inclusion) the arguments that are labelled as UNDEC is valid. (Note, it is guaranteed that there is always exactly one such labelling.) 让UNDEC近可能多标，IN仅可能的少
  - An argument is **acceptable under the grounded semantics** if and only if it is labelled as IN under the grounded semantics.
- Under the **preferred semantics**, any complete labelling that maximises (with reference to set inclusion) the arguments that are labelled as IN are valid. (Note, there may be more than one such labelling.) 让IN近可能的多标
  - Credulous可信的：出现过一次IN就可接受
  - Sceptical怀疑的：全为IN才可接受

E.g:



Grounded Accp: a1



9.2.2 Why argue?

论证支持所谓的非单调性原理（non-monotonic reasoning）,数据库知识的增长可能**增加结论**，也可能**撤回结论**。经典推理只支持单调推理，即随着知识库的增长，不会撤回结论。论证也允许Agent在获得新知识时，改变他们对事物的立场（经典推理不允许）

Week10: Argumentation-based Dialogues

When reasoning with argumentation, as an agent gains more knowledge it is possible that it may [retract] conclusions as well as add them. This is called [non-monotonic] reasoning and means that the agent can change its position on things. 当用论证推理时，随着代理人获得更多的知识，它有可能[收回]结论，也可能增加结论。这被称为[非单调性]推理，意味着代理人可以改变其对事物的立场。

We can use subjective [preferences] to resolve two way (symmetric) attacks in argumentation.

Argumentative principles are familiar to humans. This makes argumentation well-suited to supporting [explainable] AI.

An argument dialogue [protocol] defines the space of permissible dialogues; an agent uses its private [strategy] to determine which permissible move to make at any point.

A dialogue of type [inquiry] allows the agents to jointly reason.

A dialogue of type [deliberation] is appropriate when the agents need to reach a decision about what to do.

An agent can use its opponent model to strategise in argument dialogues. An opponent model is typically [uncertain] since agents' internal states are private.