

# 第 9 章 智能体与多智能体系统

---



# 第9章 智能体与多智能体系统

- 随着计算机网络和信息技术的发展，智能体技术得到广泛应用。多智能体不仅具备自身的问题求解能力和行为目标，而且能够相互协作，来达到共同的整体目标。因此，能够解决现实中广泛存在的复杂的大规模问题。
- 本章在介绍智能体与多智能体系统概念的基础上，简要介绍多智能体系统中的通信、协作、协调等基本技术。

# 第9章 智能体与多智能体系统

- 9.1 智能体的概念与结构
- 9.2 多智能体系统的概念与结构
- 9.3 多智能体系统的通信
- 9.4 多智能体系统的协调
- 9.5 多智能体系统的协作
- 9.6 多智能体系统的协商
- 9.7 AI Agent

# 第9章 智能体与多智能体系统

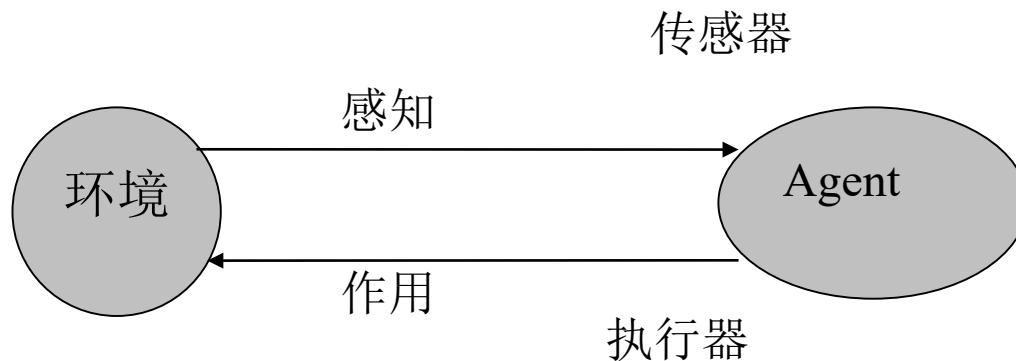
- ✓ 9.1 智能体的概念与结构
- 9.2 多智能体系统的概念与结构
- 9.3 多智能体系统的通信
- 9.4 多智能体系统的协调
- 9.5 多智能体系统的协作
- 9.6 多智能体系统的协商
- 9.7 AI Agent

# 9.1 智能体的概念与结构

## ● 9.1.1 智能体的概念

■ Agent可以看做是一个程序或者一个实体，它嵌入在环境中，通过传感器(sensors)感知环境，通过效应器(effectors)自治地作用于环境并满足设计要求。

■ Agent与环境的交互作用



# 9.1 智能体的概念与结构

## ● 9.1.2 智能体的特性

**(1) 自主性:** Agent具有独立的局部于自身的知识和知识处理方法，能够根据其内部状态和感知到的环境信息自主决定和控制自身的状态和行为。

**(2) 反应性:** Agent能够感知、影响环境。Agent的行为是为了实现自身内在的目标，在某些情况下，Agent能够采取主动的行为，改变周围的环境，以实现自身的目标。

**(3) 社会性:** 很多Agent同时存在，形成多智能体系统，模拟社会性的群体。Agent具有和外部环境中其它Agent相互协作的能力，在遇到冲突时能够通过协商来解决问题。

**(4) 进化性:** Agent应该能够在交互过程中逐步适应环境，自主学习，自主进化。

# 9.1 智能体的概念与结构

## ● 9.1.3 智能体的结构

Agent结构接收传感器的输入，然后运行Agent程序，并把执行的结果传送到效应器进行动作。

Agent系统的结构直接影响到系统的性能。

Agent、体系结构和程序之间的关系：

$$\text{Agent} = \text{体系结构} + \text{程序}$$

# 9.1 智能体的概念与结构

## ● 9.1.3 智能体的结构

**Agent**结构需要解决以下问题：

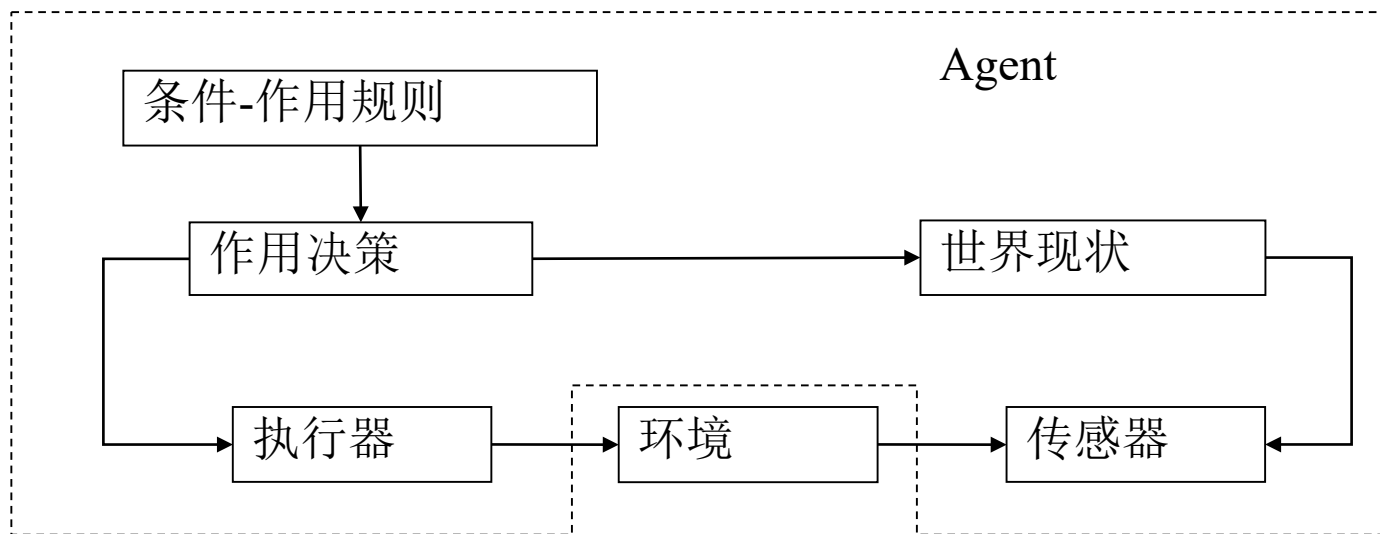
- (1) 由哪些模块组成；
- (2) 这些模块之间如何交互信息；
- (3) 感知的信息如何影响它的行为和内部状态；
- (4) 如何将这些模块用软件或硬件的方式组合起来形成一个有机的整体。

单个Agent的结构按属性可以分为：

反应式体系结构、慎思式体系结构和混合式体系结构。

# 9.1 智能体的概念与结构

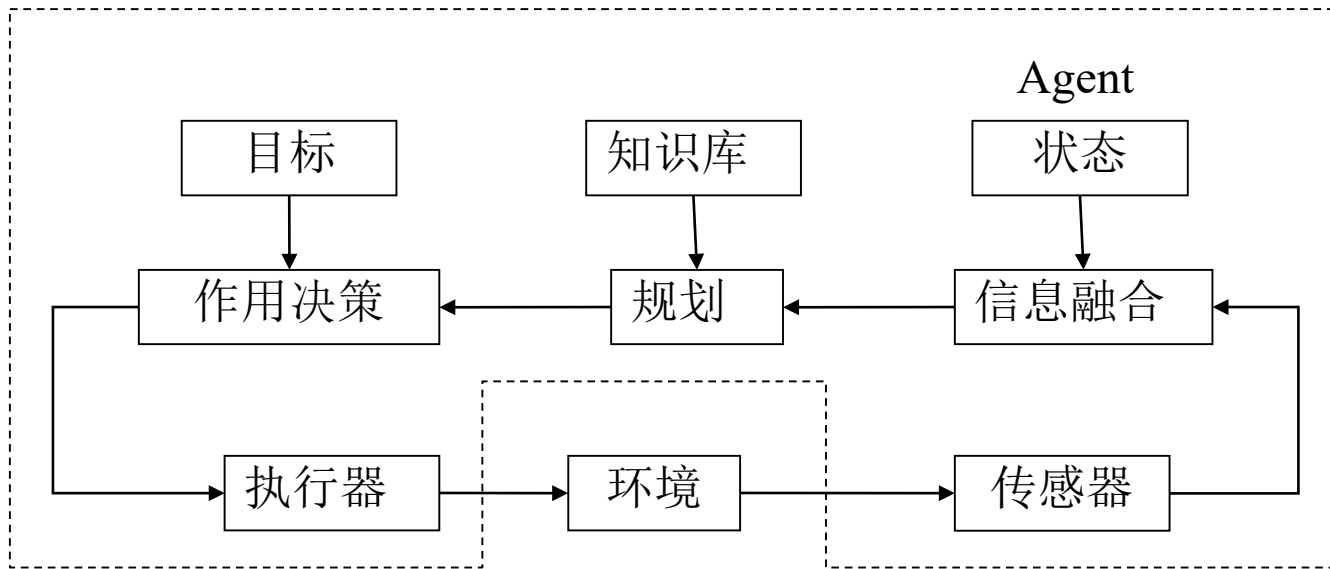
- 9.1.4 反应式Agent
- 反应式Agent是一种具备对当时处境的实时反应能力的Agent。



# 9.1 智能体的概念与结构

- 9.1.5 慎思式Agent

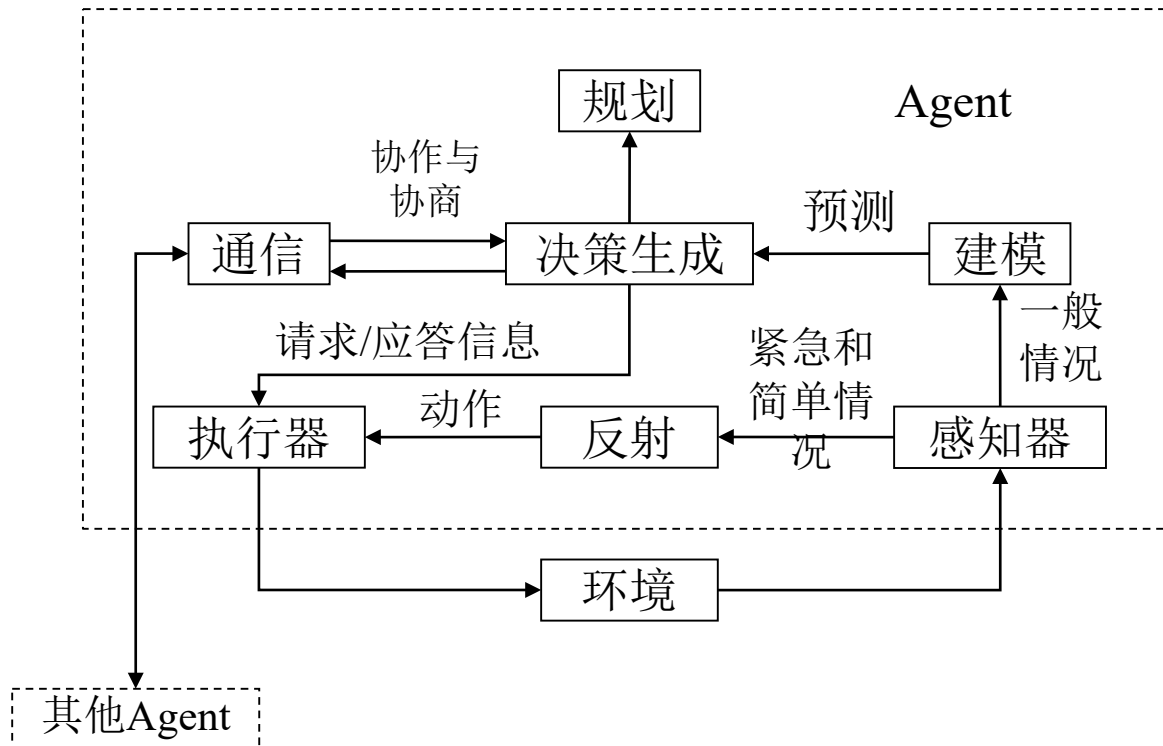
- 慎思式Agent是一种基于知识的系统，包括环境描述和丰富的智能行为的逻辑推理能力。



# 9.1 智能体的概念与结构

## ● 9.1.6 复合式Agent

- 复合式Agent是在一个Agent内组合多种相对独立和并行执行的智能形态，其结构包括感知、动作、反应、建模、规划、通信和决策等模块。



# 9.1 智能体的概念与结构

## ● 9.1.7 Agent的应用

- (1) 电信。利用Agent的特性解决复杂系统和网络管理方面的任务，包括负载均衡、故障预测、问题分析和信息综合等。
- (2) 兴趣匹配。Agent更多应用于商业网站向用户提供建议。
- (3) 用户助理。用Agent协助用户更好地完成特定的任务。
- (4) 组织结构。由多个Agent构造一个类似于人类组织的系统，不同的Agent代表着系统内的不同角色，通过这些Agent之间的通信和协作来完成具体的任务。目前主要应用于电子商务。
- (5) 智能信息检索。Agent可以通过利用相关知识检索一些特定信息。

# 9.1 智能体的概念与结构

(6) 决策支持系统。Agent能够监控系统的一些关键信息，在系统可能出现问题的时候，警告相应的操作员，并在数据挖掘技术和决策支持模型的协助下，为复杂的决策提供有效的支持。

(7) 移动计算。Agent能为移动应用提供自然有效且稳定的离线计算模式，在移动用户再次连上网络之后再把结果反馈给用户。

(8) 远程教育。Agent作为虚拟教师、虚拟学习伙伴、虚拟实验设备、虚拟图书管理员等，实现虚拟的教学、练习和实验环节等。

(9) 数字娱乐。如在个性化的节目中插入点播服务；在游戏、动画中进行更加人性化的角色设计。

# 第9章 智能体与多智能体系统

- 9.1 智能体的概念与结构
- ✓ 9.2 多智能体系统的概念与结构
- 9.3 多智能体系统的通信
- 9.4 多智能体系统的协调
- 9.5 多智能体系统的协作
- 9.6 多智能体系统的协商
- 9.7 AI Agent

## 9.2 多智能体系统的概念与结构

### 9.2.1 多智能体系统的特点

- MAS中每个智能体具有独立性和自主性。
- MAS支持分布式应用，具有良好的模块性。
- MAS按面向对象的方法构造多层次、多元化的智能体。
- MAS是一个协调式的系统，也是一个集成系统。
- 在MAS中，智能体之间相互通讯，彼此协调，并行地求解问题，提高了问题求解效率。
- 同一个MAS中各个智能体可以是异构的。
- 在MAS中，不同领域的专家系统、同一领域不同的专家系统可以协作求解单一专家系统难以解决的问题。

## 9.2 多智能体系统的概念与结构

### 9.2.2 多智能体系统的基本类型

□ **BDI模型**：是一个概念和逻辑上的理论模型，是研究Agent理性和推理机制的基础。

□ **协商模型**：Agent的协作行为一般通过协商而产生。合同网协议就是协商模型的典型代表，主要解决任务分配、资源冲突和知识冲突等问题。

□ **协作规划模型**：用于制订其协调一致的问题规划。

□ **自协调模型**：随环境变化自适应地调整行为。

## 9.2 多智能体系统的概念与结构

### 9.2.3 多智能体系统的体系结构

- **网络结构**: **Agent**之间都是直接通信的，通信和状态知识都是固定的。
- **联盟结构**: 若干相距较近的**Agent**通过一个叫做协助者的**Agent**来进行交互，而远程**Agent**之间的交互和消息发送是由局部**Agent**群体的协助者**Agent**协作完成的。
- **黑板结构**: 黑板结构中的局部**Agent**把信息存放在可存取的黑板上，实现局部数据共享。

# 第9章 智能体与多智能体系统

- 9.1 智能体的概念与结构
- 9.2 多智能体系统的概念与结构
- ✓ 9.3 多智能体系统的通信
- 9.4 多智能体系统的协调
- 9.5 多智能体系统的协作
- 9.6 多智能体系统的协商
- 9.7 AI Agent

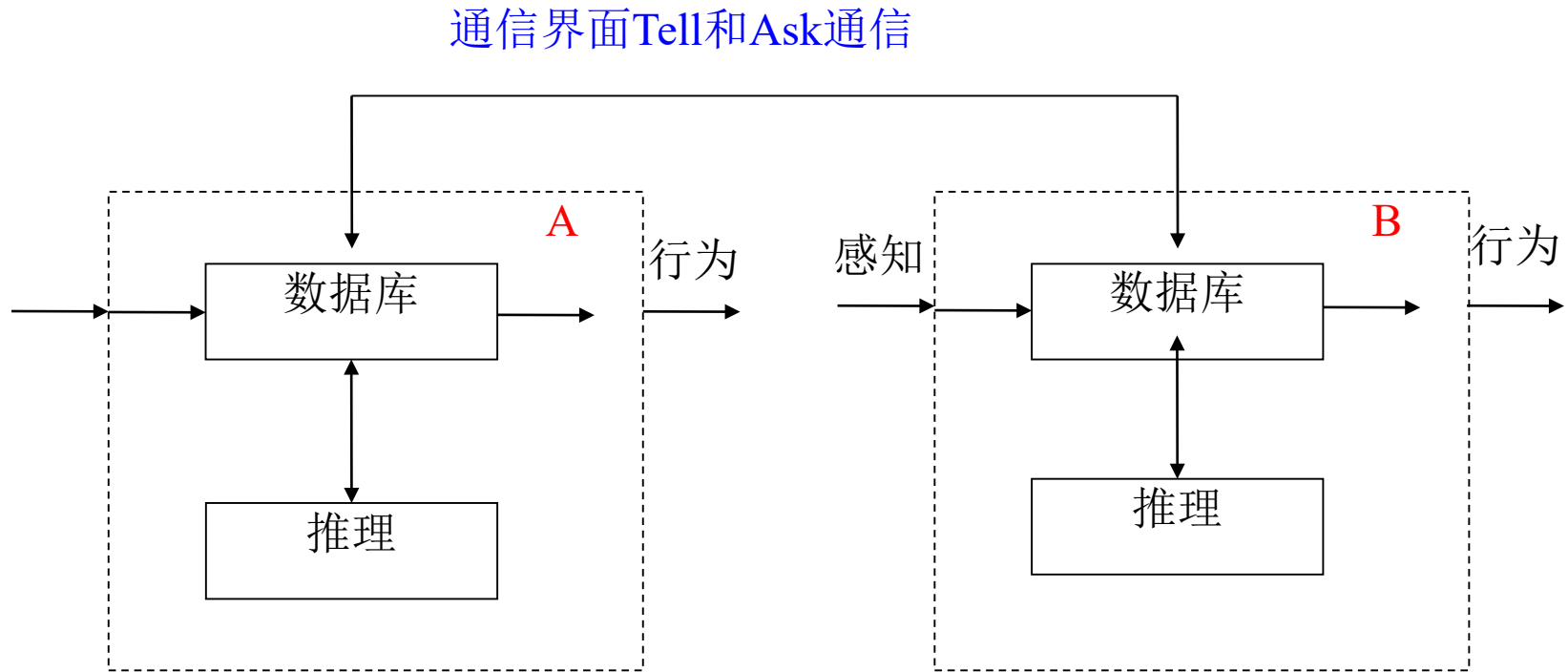
## 9.3 多智能体系统的通信

两个**Agent**之间的通信过程如下：

- 发送方将自己的思想翻译成通信所用语言的格式；
- 发送方将语言格式加载到通信传播媒体，如声音、文字和图像；
- 传播载体到达接收方；
- 接收方读取载体中的语言代码；
- 接收方在思维空间中按其格式将语言代码翻译为思想，从而熟悉发送方的意识状态。

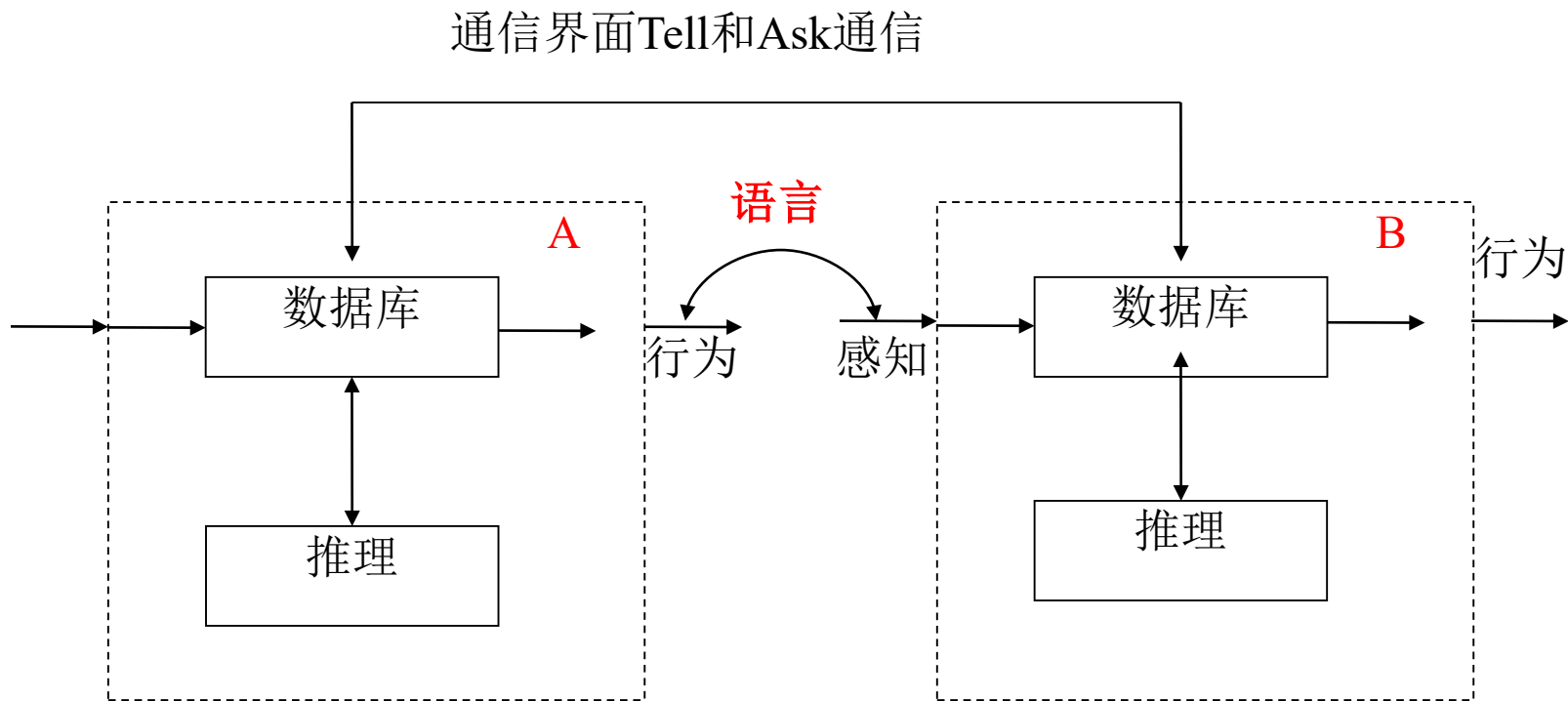
## 9.3.1 智能体通信的类型

□ 使用**Tell**和**Ask**通信：**Agent**分享共同内部表示语言，并通过界面**Tell**和**Ask**直接访问相互的知识库。



## 9.3.1 智能体通信的类型

□ **使用形式语言通信**：大多数Agent的通信是通过语言而不是通过直接访问知识库而实现的。



## 9.3.2 Agent通信的方式

### 1. 黑板系统

□ 黑板系统的模型有三个主要组成部分：

□ ① 知识源，即Agent，是作为求解问题的独立单元，具有不同的专门知识，独立完成特定的任务。

□ ② 黑板，即公共工作区，为知识源提供信息和数据，同时，供知识源进行修改。

□ ③ 监控机制。根据黑板当前的问题求解状态，以及各知识源的不同求解能力，对其进行监控，使之能适时相应黑板变化，及时进行问题求解。

## 9.3.2 Agent通信的方式

### 2. 消息/对话系统

有两种方式来实现**Agent**间的消息传递：

□① 直接通信方式：每个**Agent**必须知道消息在什么时候发送到什么地方，系统中有哪些**Agent**可以合作，这些**Agent**各具备什么样的能力等。这要求系统中的每个**Agent**都拥有其他**Agent**的信息。

□② 中介的通信：在基于中介的消息传送中，若干相距较近的**Agent**通过通信服务器来进行交互和消息发送，而远程**Agent**之间的交互是由局部群体中的通信服务器协作完成的。

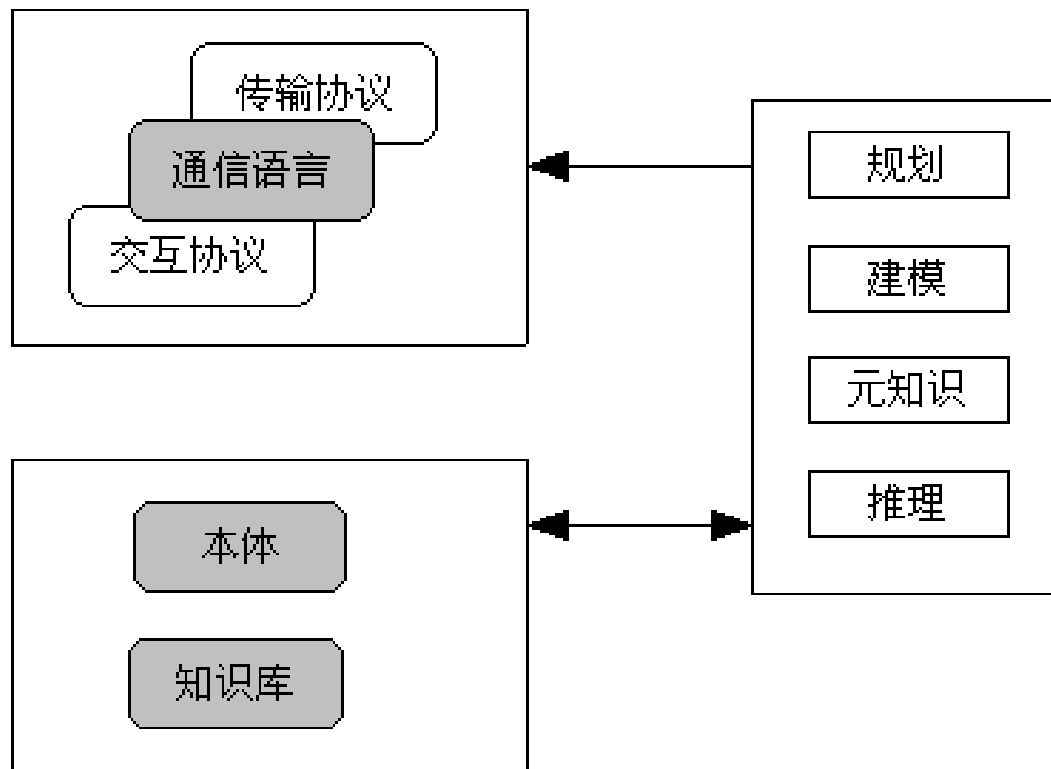
## 9.3.3 智能体通信语言

### 1. 知识交换格式语言KIF

- 主要是基于谓词逻辑，可以作为描述专家系统、数据库、多智能体的知识表示工具。
- KIF负责将一种语言翻译成另一种语言，或者为两种异构智能体的知识表达提供语义共享。
- 可共享重用知识则是一个词汇表，它可以使可共享知识库的内容更容易被理解，同时也为特定的领域提供开发工具和方法。
- 外部接口主要设计软件智能体运行时能够共享知识和信息的通信高层语言KQML，即知识查询操纵语言。KQML是自主的异步智能体之间共享知识和实现协作问题求解的通信语言。
- KQML既是一种消息格式，也是支持实时智能体之间知识共享的消息处理协议，实现基于知识的异构系统之间互操作和集成。

## 9.3.3 智能体通信语言

- 在实际应用中，软件实体之间的互操作和知识共享是应用程序作为软件智能体的重要基础，甚至是必不可少的先决条件。
- 互操作软件智能体的抽象模型。



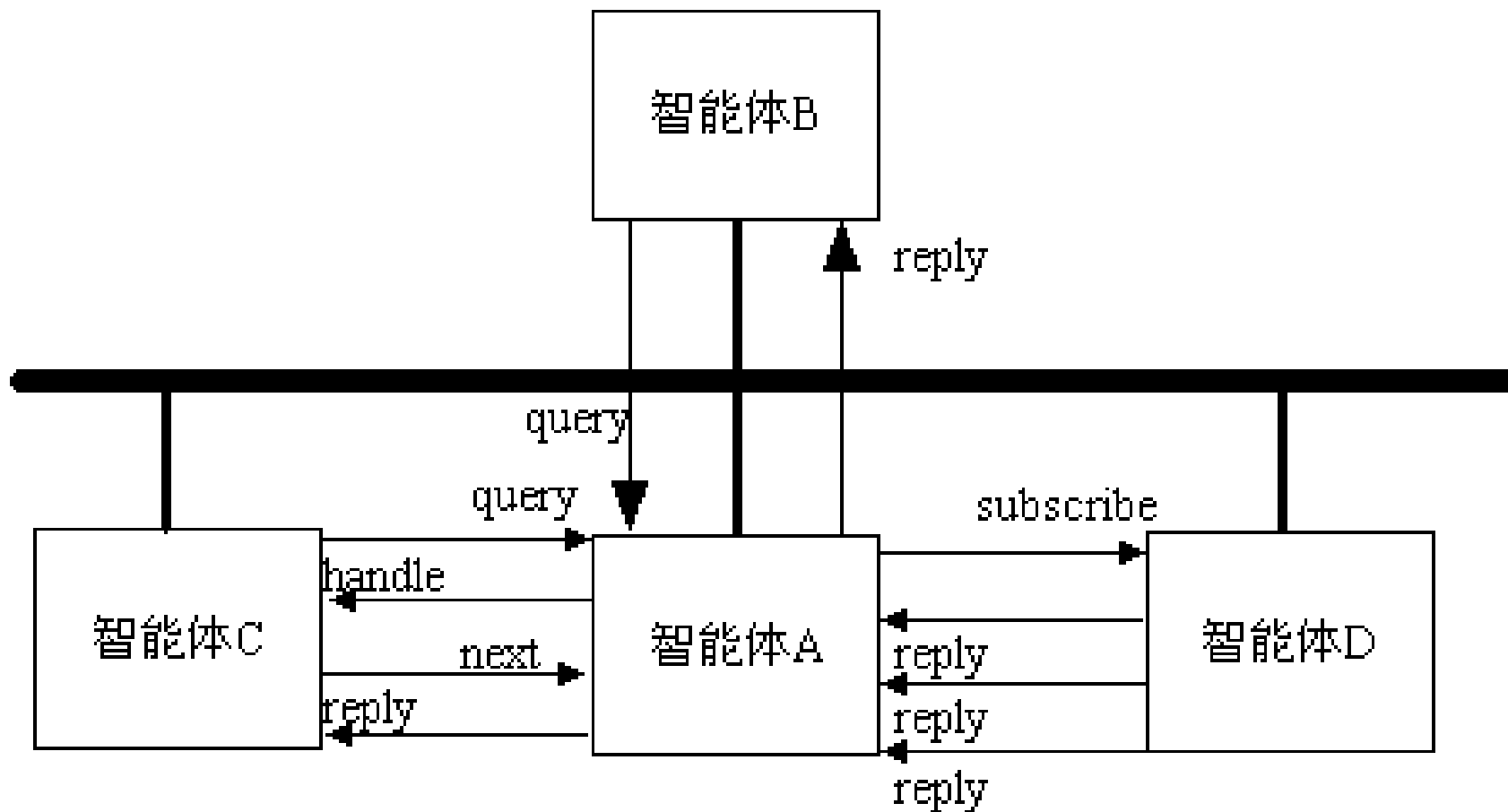
## 9.3.3 智能体通信语言

### 2. 知识查询操纵语言KQML

- **KQML**为多智能体通信定义了一套消息表达机制和消息传递格式，构建了一种标准通用框架。
- **KQML**分为3层：内容层、消息层和通信层。
- 内容层使用应用程序本身的表达语言来传送消息的实际内容。
- 通信层主要负责对消息的某些特性进行编码，这些特性描述了底层通信参数，例如发送者和接收者的标识符。
- 消息层是整个**KQML**语言的核心。将一条消息从一个应用程序传送到另一个应用程序时，消息层完成对所传送信息的封装。消息层的一个最基本功能是识别传输消息发送时所使用的协议，并且给消息发送者提供一个附加在内容上的述行语或原语。**KQML**语言在内容不可知的情况下实现对消息的分析、路由和正确的传送。

## 9.3.3 智能体通信语言

□ KQML 语言中有多种在进程之间进行信息交换的协议。



# 第9章 智能体与多智能体系统

- 9.1 智能体的概念与结构
- 9.2 多智能体系统的概念与结构
- 9.3 多智能体系统的通信
- ✓ 9.4 多智能体系统的协调
- 9.5 多智能体系统的协作
- 9.6 多智能体系统的协商
- 9.7 AI Agent

## 9.4 多智能体系统的协调

- 协调和协作是**MAS**研究的核心问题之一，是一个系统智能水平的重要体现。
- 协调是一组**Agent**完成集体活动时相互作用的性质。
- 协作是非对抗的**Agent**之间保持行为协调的特例。
- **MAS**中的协调是指多个**Agent**为了一致和谐的方式工作而进行交互的过程，避免**Agent**之间的死锁或活锁。
- 死锁指多个**Agent**无法进行各自的下一步动作；活锁指多个**Agent**不断工作却无任何进展。

## 9.4 多智能体系统的协调

当前主要有以下四种协调方法。

- (1) 基于集中规划的协调：将具备其他**Agent**的知识、能力和环境资源知识的**Agent**可作为**主控Agent**，对该**MAS**的目标进行分解，对任务进行规划，并指示或建议其他**Agent**执行相关任务。特别适用于环境和任务相对固定、动态行为集可预计和需要集中监控的情况。
- (2) 基于协商的协调：通过协商来实现任务的分配。协商是**Agent**间交换信息、讨论和达成共识的方式。

## 9.4 多智能体系统的协调

- (3) 基于对策论的协调：有通信协调和无通信协调两类。无通信协调是在没有通信的情况下，**Agent**根据对方及自身效益模型，按照对策论选择适当行为，**Agent**至多也只能达到协调的平衡解。在有通信协调中则可得到协作解。
- (4) 基于社会规划的协调：以每个**Agent**必须遵循的社会规则、过滤策略、标准和惯例为基础的协调方法。这些规则对于**Agent**的行为加以限制，过滤某些有冲突的意图和行为，保证其他**Agent**必须的行为方式。

# 第9章 智能体与多智能体系统

- 9.1 智能体的概念与结构
- 9.2 多智能体系统的概念与结构
- 9.3 多智能体系统的通信
- 9.4 多智能体系统的协调
- ✓ 9.5 多智能体系统的协作
- 9.6 多智能体系统的协商
- 9.7 AI Agent

# 9.5 多智能体系统的协作

## 9.5.1 多智能体的协作类型

- (1) 完全协作型：系统中的智能体围绕一个共同的全局目标全力以赴地协作，各个智能体没有自己的局部目标。
- (2) 协作型：系统中的智能体具有一个共同的全局目标，同时各个智能体还有与全局目标一致的局部目标。
- (3) 自私型：系统中不存在共同的全局目标，各智能体都为自己的局部目标工作，而且目标之间可能存在冲突。
- (4) 完全自私型：系统中不存在共同的目标，各智能体都为自己的局部目标工作，并且不考虑任何协作行为。
- (5) 协作与自私共存型：系统中既存在共同的全局目标，某些智能体也可能还具有与全局目标无直接联系的局部目标。

## 9.5 多智能体系统的协作

多智能体系统的协作过程一般分为六个阶段：

- (1) 产生协作需求，即确定协作目标；
- (2) 协作规划，求解合理的协作结构；
- (3) 寻求协作伙伴；
- (4) 选择协作方案，即根据协作竞争者反推最佳的协作方案；
- (5) 按协作或交互协议进行协作以实现所确定的目标；
- (6) 结果评估，即判断协作的效果并为以后的协作提供可供参考的经验和教训。

# 9.5 多智能体系统的协作

## 9.5.2 合同网协作方法

- 在所有的协作方法中，合同网是最著名并且应用最广泛的一种协作方法。是由**Smith**于**1980**年提出的。
- 基本思想:人们在商务过程中用于管理商品和服务的合同机制。
- 在合同网方法中，所有智能体分为两种角色：管理者和工作者。
- 在合同网协作方法中，智能体的角色在协作过程中的角色可以变化：任何智能体通过发布任务通知书而成为管理者；相反，任何智能体通过应答任务通知书而成为工作者。

# 9.5 多智能体系统的协作

## 9.5.3 黑板模型协作方法

黑板模型由三个基本模块构成：

- **知识源**：应用领域根据求解问题专门知识的不同划分成若干相互独立的专家，这些专家称为知识源（即智能体）。每一知识源独立完成一个特定领域的任务。
- **黑板**：共享的问题求解工作空间。主要存放知识源所需要的信息和求解过程中的解状态数据，如初始数据、部分解、替换解、最终解等。在问题求解过程中，知识源不断地修改黑板。知识源之间的通信和交互智能通过黑板进行。
- **监控机制**：根据黑板上的问题求解状态和各知识源的求解技能，依据某种控制策略，动态地选择和激活合适的知识源，使知识源能实时地响应黑板的变化。

# 9.5 多智能体系统的协作

## 9.5.4 市场机制协作方法

- **基本思想**: 针对分布式资源分配的待定问题，建立相应的计算经济，使智能体之间通过最少的直接通信来协调多个智能体之间的活动。
- 系统中只存在两种类型的智能体：生产者和消费者。智能体以各种价格对商品进行投标，但所有的商品交换都以当前市场价格进行，每一智能体通过投标以便获得最大利益和效用。
- 在开放的市场环境中，智能体应该可以自由地选择自己的贸易策略，其行为不一定合乎常规。市场机制假定智能体所给予的偏好是与智能体获得行动结果的知识相一致，因此智能体推理行为就是最大化它自身的偏好。

# 第9章 智能体与多智能体系统

- 9.1 智能体的概念与结构
- 9.2 多智能体系统的概念与结构
- 9.3 多智能体系统的通信
- 9.4 多智能体系统的协调
- 9.5 多智能体系统的协作
- ✓ 9.6 多智能体系统的协商
- 9.7 AI Agent

# 9.6 多智能体系统的协商

## 1. 协商协议

**Multi-Agent**协商概括为协商协议、协商策略和协商处理。

□ 协商协议用于处理协商过程中协商之间的交互和作用，是交易双方交互的规则。主要研究的内容是**Agent**通信语言的定义、表示、处理和语义解释。

□ 协商协议的形式化表示通常有三种方法：巴科斯范式表示、有限自动机表示和语义表示。

□ 巴科斯范式表示具有简洁、明了的特点，是最常用的方法。

□ 常用的协商协议：根据协商对象的数量分为一对一、一对多、多对多的协议；根据协商的顺序分为轮流出价、同时出价协商协议；根据协商议题的数量分为单属性和多属性协商等。

# 9.6 多智能体系统的协商

## 2. 协商策略

协商策略是**Agent**选择协商协议和通信消息的策略。

□ 协商策略分为：单方让步策略、竞争型策略、协作型策略、破坏协商策略和拖延协商策略。主要是竞争型和协作型策略。

□ 竞争型策略是指协商参与者在协商过程中表现出竞争行为，使协商结果向有利于自身利益方向发展的协商对策。合同网协调模型、劳资协商、基于对策论的协商过程等都属于此类。

□ 协作型策略是指协商各方都从整体利益出发，在协商过程中互相合作，采取的协商对策有利于互相能接受的协商结果。

□ 策略选择的通用方法是，依据影响协商的多方面因素，给出适宜的策略选择函数。

# 9.6 多智能体系统的协商

## 3. 协商处理

- 协商处理是对单个协商方及协商系统、协商行为的描述及分析，包括协商算法和系统分析两部分内容。
- 协商算法用于描述**Agent**在协商过程中的行为，包括通信、决策、规划和知识库操作等。
- 系统分析的任务是分析和评价**Agent**协商的行为和性能，回答协商过程中的问题求解质量、算法效率，以及系统的公平性和死锁等问题。
- 协商协议主要处理协商过程中**Agent**之间的交互。协商策略主要设计**Agent**内的决策和控制过程。而协商处理则侧重于对单个**Agent**和多个**Agent**协商社会的整体协商行为的描述和分析。

# 第9章 智能体与多智能体系统

- 9.1 智能体的概念与结构
- 9.2 多智能体系统的概念与结构
- 9.3 多智能体系统的通信
- 9.4 多智能体系统的协调
- 9.5 多智能体系统的协作
- 9.6 多智能体系统的协商
- ✓ 9.7 AI Agent

# AI Agent

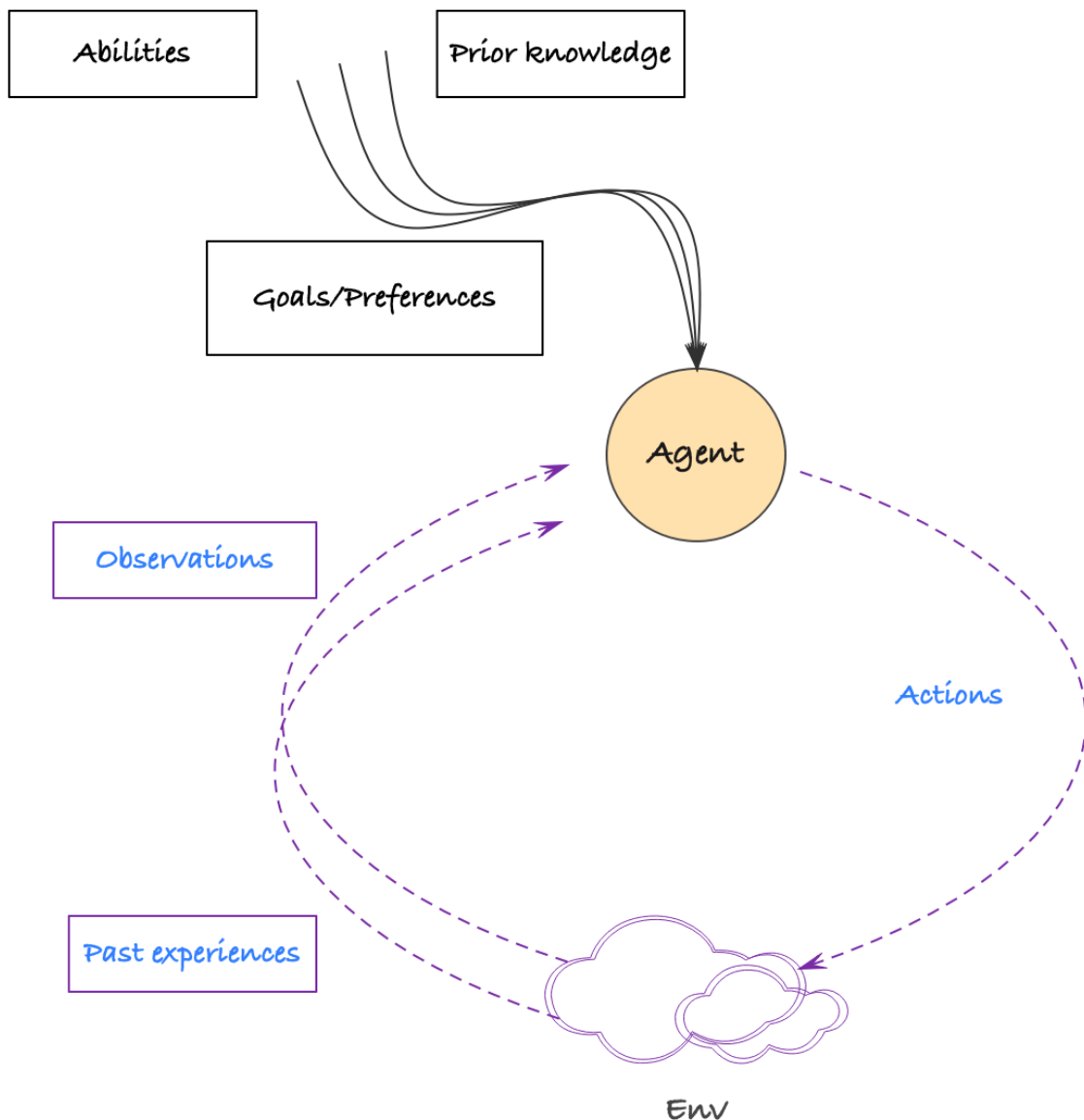
**AI Agent**（人工智能代理）是一种能够感知环境、进行决策和执行动作的智能实体。

	LLM
	LLM + Planning
	LLM + Planning + Memory
	Agents (LLM + Planning + Memory + Tools) <small>知乎 @产品经理大群</small>

# AI Agent

AI Agents 是一种软件程序，旨在与其环境交互，感知接收到的数据，并根据该数据采取行动以实现特定目标。

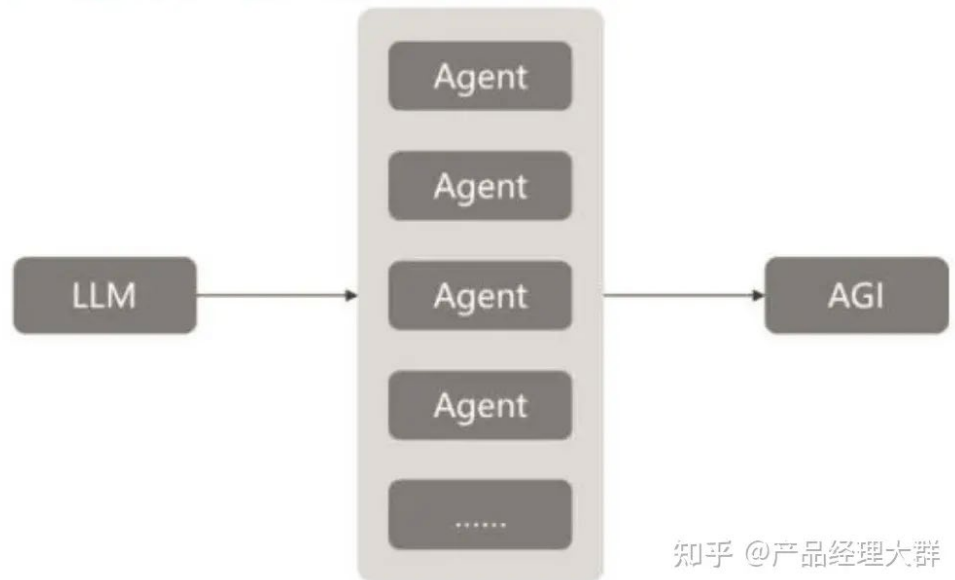
自主AI Agents是一种先进的软件程序，可以在没有人类控制的情况下独立运行。可以自主思考、行动和学习，无需人类不断输入。可以适应新情况，从经验中学习，并利用自己的内部系统做出决策。



# AI Agent

□ AI Agent也可以称为“智能体”，也可理解为“智能业务助理”，旨在大模型技术驱动下，让人们以自然语言为交互方式高自动化地执行和处理专业或繁复的工作任务，从而极大程度释放人员精力。

□ Agent其实基本就等于“大模型 + 插件 + 执行流程 / 思维链”，分别会对应控制端(Brain / 大脑)、感知端(Preception)、执行端(Action)环节



- LLM的浪潮推动了AI Agent相关研究快速发展，AI Agent是当前通往AGI的主要探索路线。
- LLM庞大的训练数据集中包含了大量人类行为数据，为模拟类人的交互打下了坚实基础；随着模型规模不断增大，LLM涌现出了上下文学习能力、推理能力、思维链等类似人类思考方式的多种能力。
- 将LLM作为AI Agent的核心大脑，可以实现以往难以实现的将复杂问题拆解成可实现的子任务、类人的自然语言交互等能力。
- 由于大模型仍存在大量的问题如幻觉、上下文容量限制等，通过让大模型借助一个或多个Agent的能力，构建成为具备自主思考决策和执行能力的智能体，成为了当前通往AGI的主要研究方向。

# Agent的出现

- 让**通用**大模型学习行业知识和行业语料成为**行业**大模型，再进一步学习业务知识和专业领域工具演进为**场景**大模型，是通用**AI**深入业务场景，承担更复杂任务的关键路径。这一过程让大模型的持续进化最终以**AI Agent**的产品形态，开始了对业务的流程及其管理和服务模式的重构与优化。
- **OpenAI**联合创始人安德烈·卡帕斯认为，对于通用人工智能（**AGI**）的应用，将广泛采用**AI Agent**的产品形式开展业务，而来自于各行业领域的开发人员和创业者们，将比通用大模型的开发商们更具有对**AI Agent**的开发优势。**AI Agent**，也被越来越多的人认可为是数字经济时代深刻赋能各行各业的高效生产力工具。
- 不同于传统的人工智能，**AI Agent**具备通过独立思考、调用工具去逐步完成给定目标的能力。
- **AI Agent**和**LLM**的区别在于，**LLM**与人类之间的交互是基于**prompt**实现的，用户**prompt**是否清晰明确会影响大模型回答的效果。而**AI Agent**的工作仅需给定一个目标，能够针对目标独立思考并做出行动。**AI Agent**则可以通过和环境进行交互，感知信息并做出对应的思考和行动。

# AI Agent的演化

- **Symbolic Agents:**在人工智能研究的早期阶段，最主要的方法是符号人工智能，采用逻辑规则和符号表示来封装知识和促进推理过程。主要关注两个问题：转换问题和表示/推理问题。拥有明确的、可解释的推理框架，而且由于其符号性质，表现出高度的表达能力。这种方法的典型例子是基于知识的专家系统。然而，**Symbolic Agent**在处理不确定性和大规模现实世界问题时面临着局限性。
- **Reactive Agents:**主要关注Agent与其Environment之间的交互，强调快速和实时响应。优先考虑直接将输入输出进行映射，而不是复杂的推理和符号操作。**Reactive Agent**通常需要较少的计算资源，从而能做出更快的反应，但可能缺乏复杂的高层决策和规划能力。
- **RL-based Agents:**主要关注点是如何让Agent通过与环境的交互进行学习，使其在特定任务中获得最大的累积奖励。最初，主要基于强化学习算法，如策略搜索和价值函数优化，**Q-learning**和**SARSA**就是一个例子。随着深度学习的兴起，出现了深度神经网络与强化学习的整合，即深度强化学习。这使得Agent可以从高维输入中学习复杂的策略，从而取得了众多重大成就，如**AlphaGo**和**DQN**。这种方法的优势在于能让Agent在未知环境中自主学习，而无需明确的人工干预。这使得它能广泛应用于从游戏到机器人控制等一系列领域。然而，强化学习也面临着一些挑战，包括训练时间长、采样效率低以及稳定性问题，尤其是在复杂的真实世界环境中应用时。

# AI Agent的演化

## Agent with transfer learning and meta learning

- 训练强化学习Agent需要大量样本和较长的训练时间，而且缺乏泛化能力。
- 因此，引入迁移学习来加速Agent对新任务的学习。迁移学习减轻了新任务培训的负担，促进了知识在不同任务间的共享和迁移，提高了学习效率、绩效和泛化能力。
- 此外，AI Agent也引入了元学习。元学习的重点是学习如何学习，使Agent能从少量样本中迅速推断出新任务的最优策略。这样的Agent在面对新任务时，可以利用已获得的一般知识和策略迅速调整其学习方法，从而减少对大量样本的依赖。
- 然而，当源任务和目标任务之间存在显著差异时，迁移学习的效果可能达不到预期，并可能出现负迁移。此外，元学习需要大量的预训练和大量样本，因此很难建立通用的学习策略。

# AI Agent的演化

## LLM-based Agent

- 由于LLM已经展示出令人印象深刻的新兴能力，LLM开始被用来构建AI Agent。
- LLM作为Agent的大脑或控制器的主要组成部分，并通过多模态感知和工具利用等策略来扩展其感知和行动空间。
- 通过思维链（CoT）和问题分解等技术，LLM-based Agent表现出与Symbolic Agent相当的推理和规划能力。还可以通过从反馈中学习和执行新的行动，获得与环境互动的能力。同样，LLM在大规模语料库中进行预训练，并显示出少量泛化的能力，从而实现任务间的无缝转移，而无需更新参数。
- LLM-based Agent由于具有自然语言理解和生成能力，可以无缝互动，从而促进多个Agent之间的协作和竞争。

# AI Agent能力

## 1、感知和理解能力

AI Agents 能够感知环境中的信息，例如图像、声音、文本等，并理解这些信息的含义和上下文。这涉及到使用传感器、计算机视觉、语音识别、自然语言处理等技术来实现对输入数据的解析和理解。

## 2、决策和规划能力

AI Agents 能够基于感知到的信息和存储的知识做出决策并制定行动计划，可以使用逻辑推理、统计分析、规划算法或机器学习技术来评估不同行动的可能结果和潜在风险，并选择最佳的行动策略。

## 3、自主学习和适应能力

AI Agents 具有学习和改进自身的能力，可以从经验中学习，并根据反馈信息不断改进自己的性能和表现。代理可以使用监督学习、增强学习、迁移学习等技术来获取新知识、调整决策制定和行动执行过程，以提高自身的智能水平和适应能力。

# AI Agent能力

## 4、交互和沟通能力

AI Agents 能够与人类或其他代理进行交互和沟通，可以理解自然语言指令、生成自然语言回应，并使用语音、文字或其他形式与用户进行有效的交流。这涉及到自然语言处理、对话系统、语音合成等技术的应用。

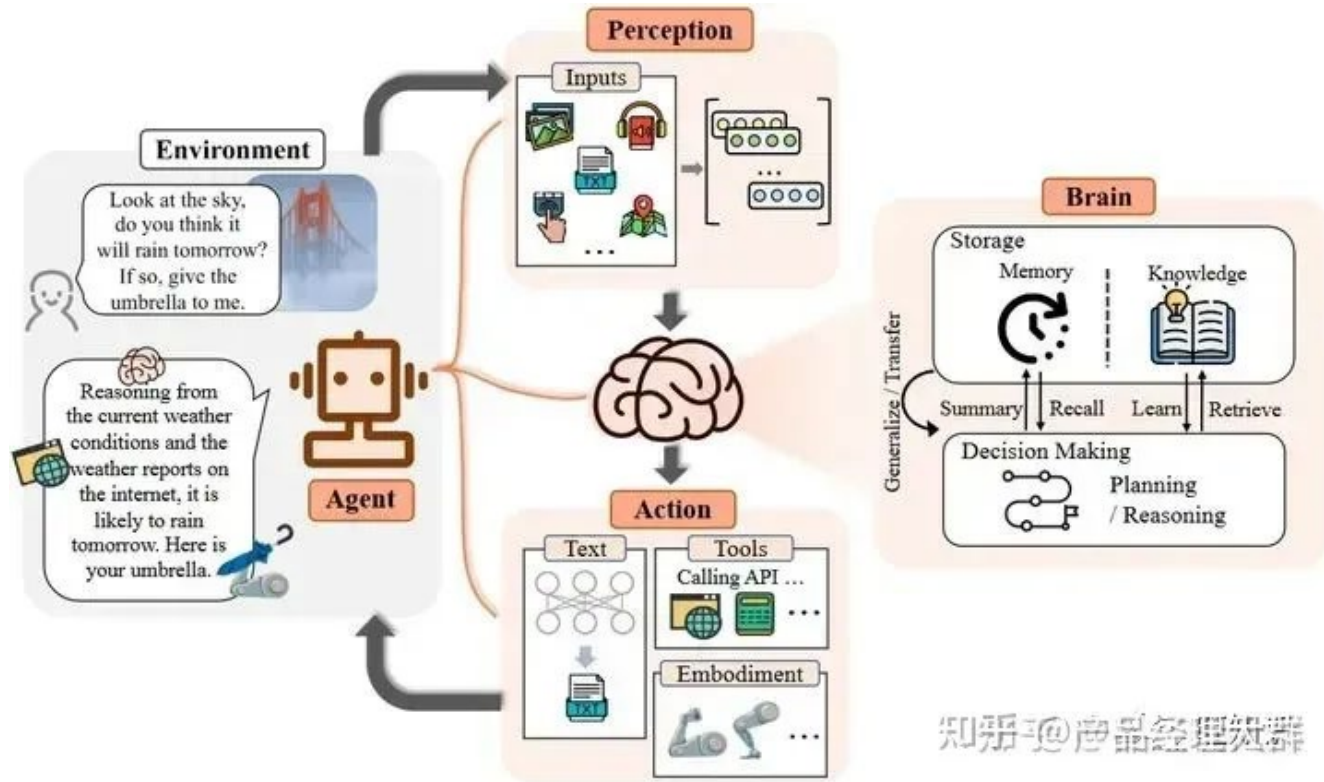
## 5、知识表示和存储能力

AI Agents 能够组织和存储知识，并有效地检索和利用这些知识来支持决策和行动，可以使用符号逻辑、图形模型、向量表示等方式来表示和存储知识，并使用数据库或其他数据结构来实现高效的知识管理。

## 6、情境感知和灵活性

AI Agents 具有对不同情境的感知能力，并可以根据情境的变化做出相应调整，可以识别环境中的变化、适应新的任务要求，并灵活地调整决策和行动策略，以适应不同的场景和需求。

# AI Agent构建



LLM-based Agent: 由大脑、感知、行动三个部分组成。

- 感知Perception: 负责感知和处理来自外部环境的多模态信息;
- 大脑Brain: 作为控制器, 基于LLM, 承担记忆、思考和决策等基本任务;
- 行动Action: 根据计划执行对应的任务, 比如调用第三方api, 从工具集(tools)选中合适的tool执行任务。

# AI Agent构建

一个基于大模型的AI Agent系统可以拆分为大模型、规划、记忆与工具使用四个组件部分。

AI Agent可能会成为新时代的开端，其基础架构可以简单划分为：

**Agent = LLM + 规划技能 + 记忆 + 工具使用**

其中LLM扮演了Agent的“大脑”，在这个系统中提供推理、规划等能力。

# AI Agent构建-大脑

大脑主要由一个**LLM**组成，不仅存储知识和记忆，还承担着**信息处理和决策**等功能，并可以呈现推理和规划的过程，能很好地应对未知任务。

**运行机制：**在接收感知模块处理的信息后，大脑模块首先转向存储，在知识中检索并从记忆中回忆。这些结果有助于**Agent**制定计划、进行推理和做出明智的决定。同时，它还可以更新常识和领域知识等知识，以备将来使用。**LLM-based Agent**还可以利用其固有的概括和迁移能力来适应陌生场景。

- 自然语言交互
- 知识
- 记忆
- 推理和规划
- 可迁移性和通用性

# AI Agent-感知

感知模块的核心目的是将Agent的感知空间从纯文字领域扩展到包括文字、听觉和视觉模式在内的多模态领域。

- 文本输入
- 视觉输入
- 听觉输入
- 其他输入

# AI Agent-行动

人类在感知环境后，大脑会对感知到的信息进行整合、分析和推理，并做出决策。随后，他们利用神经系统控制自己的身体，做出适应环境或创造性的行动，如交谈、躲避障碍或生火。

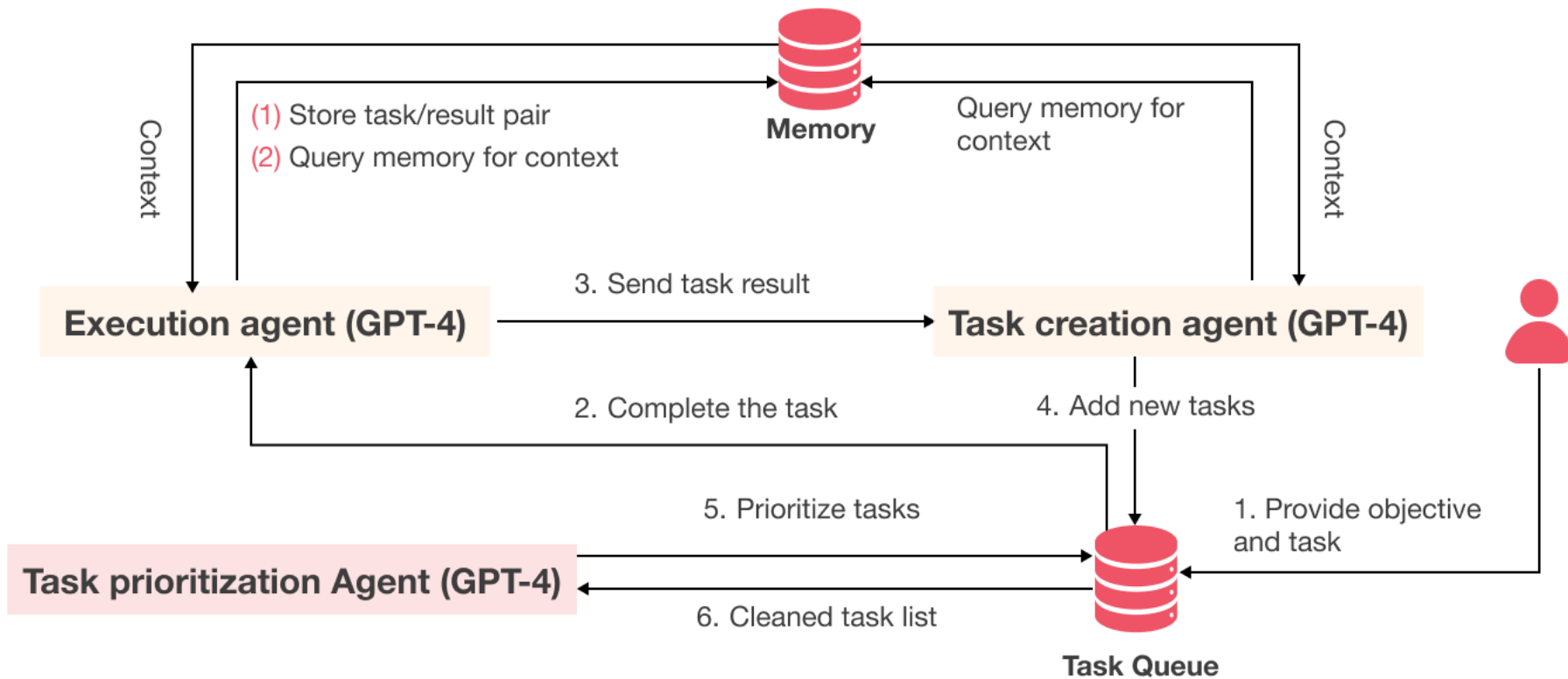
当一个Agent拥有类似大脑的结构，具备知识、记忆、推理、规划和概括能力以及多模态感知能力时，它也有望拥有类似人类的各种行动来应对周围环境。

在Agent的构建过程中，行动模块接收大脑模块发送的行动序列，并执行与环境互动的行动。

- 文本输出
- 工具使用
- 具身行动

# AI Agent-工作过程

## Autonomous AI agent workflow



# AI Agent-工作过程

针对 **AI Agents** 具体实现过程，主要涉及以下活动，具体：

## ➤ 环境感知

**Agent**通过传感器或其他数据源感知环境。传感器包括视觉传感器（如相机）、听觉传感器（如麦克风）、物理传感器（如触摸传感器）等。这些传感器帮助代理获取环境中的信息，例如图像、声音、位置等。

## ➤ 知识表示

**AI Agents** 使用适当的知识表示方法来组织和存储从环境中获取的信息。这些信息可能包括先验知识、学习到的模式或规则。常见的知识表示方法包括符号逻辑、图形模型、向量表示（如词嵌入）等。通过有效的知识表示，**AI Agents** 能够更好地理解和利用环境中的信息。

## ➤ 决策制定

基于感知到的环境信息和存储的知识，**AI Agents** 使用决策制定机制来生成适当的行动。这可能涉及使用逻辑推理、统计分析、规划算法或机器学习技术来评估不同行动的可能结果和潜在风险。决策制定过程旨在使代理能够选择最佳行动以实现其目标。

# AI Agent-工作过程

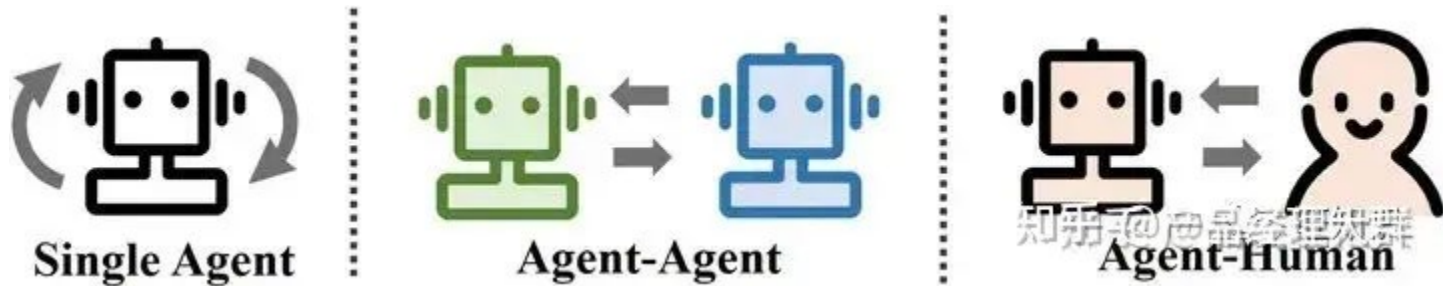
## ➤ 计划和执行行动

**Agents** 制定计划或一系列步骤来实现其目标。一旦决策制定完成，**AI Agents** 将执行行动并与环境进行交互。这可能涉及控制执行器（如机器人的电机）、发送指令（如语音助手的语音合成）或与其他代理进行通信。执行行动后，**Agents** 会观察执行结果，并将其用作反馈以调整下一步的决策。

## ➤ 反馈与改进

在完成上述的执行行动后，**AI Agents** 通过与环境的交互获得反馈。这些反馈可以来自环境中的直接观测结果，也可以来自人类用户或其他代理的指令和评估。**Agents** 使用这些反馈来学习和改进自己的行为。这可能包括使用监督学习、强化学习或迁移学习等技术来调整决策制定和行动执行过程，以提高代理的性能和适应能力。

# AI Agent-实践



三种应用场景：单个代理部署、多代理交互、人与代理交互。

- **Single Agent**拥有多种能力，在各种应用方向上都能表现出出色的任务解决能力。比如将**Agent**作为一个**AI助手**，类似目前**GPT**的问答操作；
- **Agent-Agent**就是多个**Agent**可以相互交互，它们可以通过合作或对抗性互动取得进步。比如**chatdev**，定义了多个角色，实现了一个虚拟的开发公司；
- **Human-Agent**就是加入人工提示或者反馈，根据人工信息，**Agent**可以调整任务，实现更好的完成任务。

# AI Agent-应用

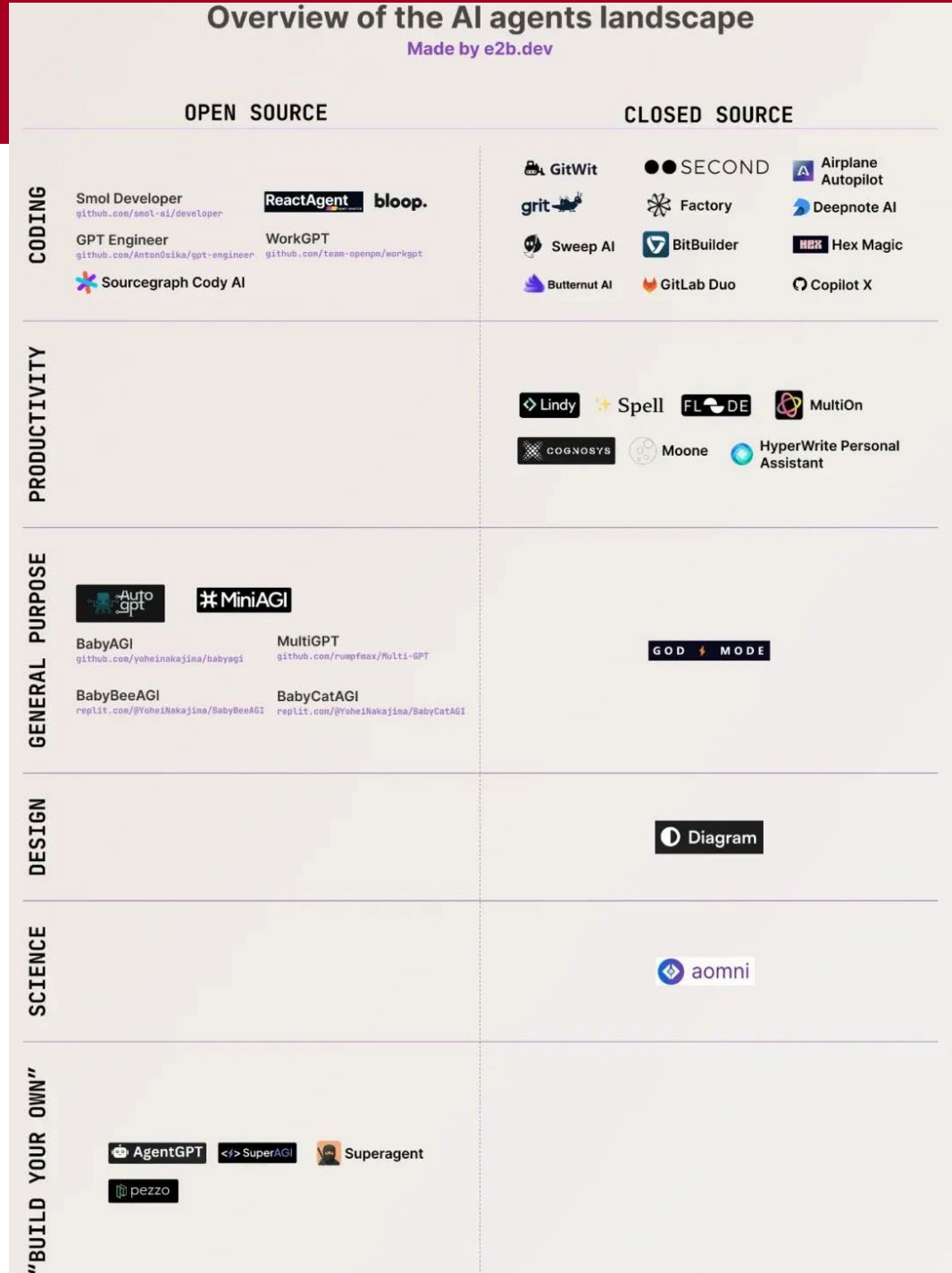


Agent能实现以下目标:

- 帮助用户从日常任务和重复劳动中解脱出来，从而减轻人类的工作压力，提高任务解决效率。
- 不再需要用户提供明确的低级指令。相反，Agent可以独立分析、规划和解决问题。
- 在解放用户双手的同时，Agent也解放了用户的大脑，使其能够从事探索性和创新性工作。

# AI Agent-领域

对 50 多个 AI Agents 进行了概览，包括了各种类型的代理，涉及的领域包括自然语言处理、图像识别、语音识别、智能助理等。



# AI Agent- 示例AgentGPT

主要包含3个部分：

- **Reasoning and Planning: 推理和规划。** 如果定义一个目标，只是简单的输入给 LLM 模型，只能得到一个粗略的答案。如果使用“prompt engineering”（提示工程），则可以将目标分解成多个更容易理解的步骤，并用思维链（CoT）提示的方法进行反思。
- **Memory: 记忆。** 分为短期记忆和长期记忆。
  - 短期记忆是根据上下文学习，受限于 LLM 的 token 长度；
  - 长期记忆是当执行一个复杂任务时，需要考虑到历史任务的情况，当代理的任务长时间运行时，超过 token 长度时，就会过期的记忆。
  - AgentGPT 采用了向量数据库存储历史信息特征向量。
- **Tools: 工具集。** LLM 只支持文本的输出，对于复杂任务，比如定个机票，就无法完成。解决方法就结合“prompt engineering”，定义一系列工具集，通过 prompt 和工具功能的描述，agent 就可以根据任务调用相应的工具。比如，定义一个“search”工具，功能描述是用“Google Search”的 api 搜索内容，描述中还可以包含 api 调用的输入输出以及内容格式等。

# AgentGPT-Prompt Engineering

Prompts的重点就是设计合适的模板，不同的LLM对prompt的格式可能不同。常用方法如下：

➤ zero-shot方法(只提供prompt，不提供示例)

```
Prompt:  
Classify the text into neutral, negative or positive.  
Text: I think the vacation is okay.  
Sentiment:  
  
Output: Neutral
```

➤ One-shot, Two-shot, N-shot等方法，即提供1、2或N个示例，以提高模型的准确性

```
Prompt:  
Classify the text into neutral, negative or positive.  
Text: I think the vacation is okay.  
Sentiment: Neutral  
Text: The vacation is amusing.  
Sentiment:  
  
Output: Positive
```

# AgentGPT- Chain-of-Thought (CoT) Prompting

通过增加逻辑推理步骤，实现比较复杂的任务。

## Standard Prompting

### Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

### Model Output

A: The answer is 27. ❌

## Chain-of-Thought Prompting

### Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls.  $5 + 6 = 11$ . The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

### Model Output

A: The cafeteria had 23 apples originally. They used 20 to make lunch. So they had  $23 - 20 = 3$ . They bought 6 more apples, so they have  $3 + 6 = 9$ . The answer is 9. ✓

知乎 @yeyan1

# AgentGPT- Chain-of-Thought (CoT) Prompting

## **Prompt:**

The odd numbers in this group add up to an even number: 4, 8, 9, 15, 12, 2, 1.

A: Adding all the odd number (9, 15, 1) gives 25. The answer is False.

The odd numbers in this group add up to an even number: 17, 10, 19, 4, 8, 12, 24.

A: Adding all the odd number (17, 19) gives 36. The answer is True.

The odd numbers in this group add up to an even number: 16, 11, 14, 4, 8, 13, 24.

A: Adding all the odd number (11, 13) gives 24. The answer is True.

The odd numbers in this group add up to an even number: 17, 9, 10, 12, 13, 4, 2.

A: Adding all the odd number (17, 9, 13) gives 39. The answer is False.

The odd numbers in this group add up to an even number: 15, 32, 5, 13, 82, 7, 1.

## **Output:**

A: Adding all the odd number (15, 5, 13, 7, 1) gives 41. The answer is False.

# AgentGPT-ReAct (Reasoning + Action)

ReAct技术将推理和执行合并，可以动态推理和执行，通过不断的thought和action，获取最终的答案。

**Thought 1:** I need to search Apple Remote and find the program it was originally designed to interact with.

**Act 1:** Search[Apple Remote]

**Obs 1:** The Apple Remote is a remote control introduced in October 2005 by Apple... originally designed to control the Front Row media center program...

**Thought 2:** Apple Remote was originally designed to control the Front Row media center program. I need to search Front Row next and find what other device can control it.

**Act 2:** Search[Front Row]

**Obs 2:** Could not find [Front Row]. Similar: ['Front Row Seat to Earth' , 'Front Row Motorsports', 'Front Row (software)', ...]

**Thought 3:** Front Row is not found. I need to search Front Row (software).

**Act 3:** Search [Front Row (software)]

**Obs 3:** Front Row is a discontinued media center software ...

**Thought 4:** Front Row (software) is controlled by an Apple Remote or the keyboard function keys. So the answer is keyboard function keys.

**Act 4:** Finish [keyboard function keys]

# AgentGPT -Plan and Solve

也是一种思维链方法，要求模型理解问题，提取变量和相关值，并逐步指定计划。

**Q:** Grace weight 125 pounds. Alex weights 2 pounds less than 4 times what Grace weights.

**A:** Let's first understand the problem and devise a plan to solve the problem. Then, let's carry out the plan and solve the problem step by step.

LLM →

**Plan:** We can use algebra to solve this problem. We can set up an equation with the variable  $x$  representing Alex's weight and solve for  $x$ . Then we can add the two weights together to get the combined weight.

**Step 1:** Set up an equation. We know that weights 2 pounds less than 4 times what Grace weights. We can represent this as:

Alex's weight =  $4 * \text{Grace's weight} - 2$

**Step 2:** Solve for  $x$ . We know that Grace's weight is 125 pounds. We can substitute this into the equation to solve for Alex's weight:

Alex's weight =  $4 * 125 - 2$

**Step 3:** Calculate the combined weight. We can add the two weights together to get the combined weight:

Combined weight =  $125 + (4 * 125 - 2)$ ;

**Step 4:** Answer. The combined weight of Grace and Alex is 497 pounds. ❌

**Q:** Grace weight 125 pounds. Alex weights 2 pounds less than 4 times what Grace weights.

**A:** Let's first understand the problem, extract relevant variables and their corresponding numerals, and devise a plan. Then, let's carry out the plan, calculate intermediate results (pay attention to calculation and common sense), solve the problem step by step, and show the answer.

LLM →

**Variables:**

Grace: 125 pounds, Alex:  $x$  pounds

**Plan:**

We can use the given information to calculate the weight of Alex.

**Calculation:**

Alex weights 2 pounds less than 4 times what Grace weights.

4 times what Grace weights =  $4 * 125 = 500$

Alex weight =  $500 - 2 = 498$  pounds

**Answer:**

Combined weight of Grace and Alex =  $125 + 498 = 623$  pounds ✓

# AgentGPT - Memory

agent执行多次时，会忘记之前的操作，AgentGPT用Weaviate向量数据库解决这个问题。向量数据库能够存储任务执行的历史记录作为memory，agent就在任务循环中访问memory的数据。

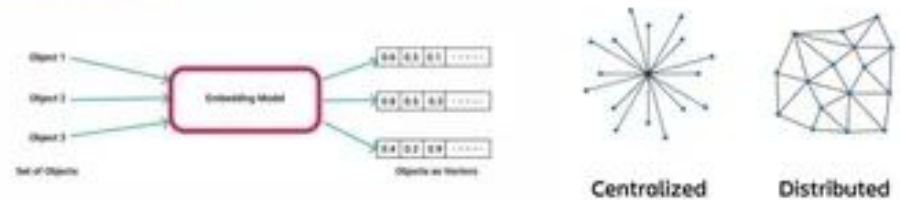
## 向量数据库的部分核心技术

### Embedding 技术：

**针对问题：**文本、图像、音频等非结构数据存储问题。  
**解决方法：**利用Embedding技术将高维度的数据（例如文字、图片、音频）映射到低维度空间，即把图片、声音和文字转化为向量来表示，将这些向量存储起来就构成向量数据库。实现Embedding过程的方法包括神经网络、LSH（局部敏感哈希算法）等。

### 向量索引技术：

**针对问题：**向量数据维度很高，直接进行全量扫描或者基于树结构的索引会导致效率低下或者内存爆炸。  
**解决方法：**采用近似搜索算法来加速向量的检索，通常利用向量之间的距离或者相似度来检索出与查询向量相近的K个向量，距离度量包括欧式距离、余弦、内积、海明距离，向量索引技术包括 k-d tree (k-dimensional tree), PQ (乘积量化), HNSW (可导航小世界网络) 等。



### 分布式系统架构：

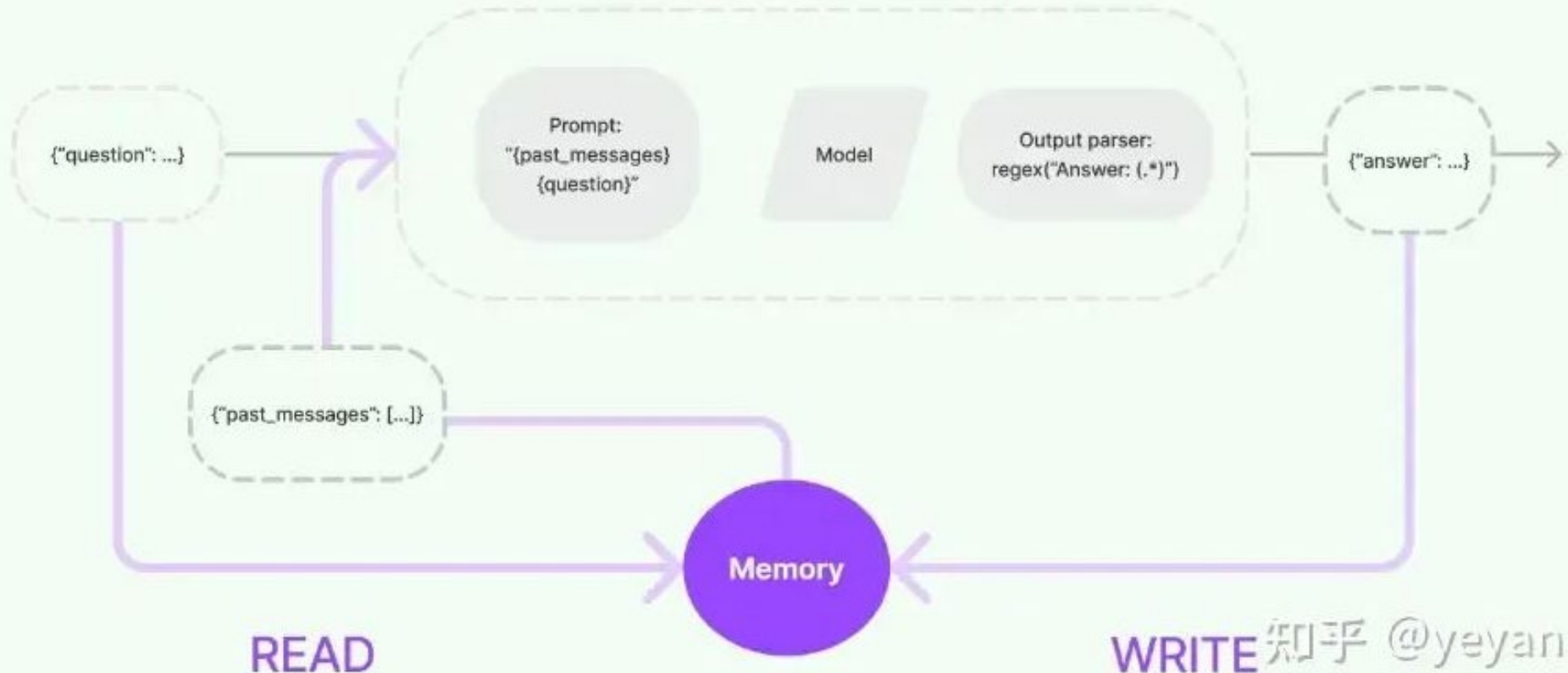
**针对问题：**向量数据规模庞大，单机无法满足存储、计算需求。  
**解决方法：**使用分布式系统。分布式系统是计算机程序的集合，这些程序利用多个节点的计算资源来实现共同的目标，节点通常代表独立的物理硬件设备，但也可代表单独的软件进程或其他递归封装的系统。

### 硬件加速技术：

**针对问题：**向量数据计算密集，单纯依靠CPU的计算能力难以满足实时性和并发性的要求。  
**解决方法：**利用专用硬件来加速向量检索。这些硬件包括GPU, FPGA, AI芯片等，用于提供更高的浮点运算能力和并行处理能力。

# AgentGPT - Memory

用户输入之后，在执行核心逻辑之前，先从memory中读取之前的信息，作为past\_messages，修改用户输入，然后再一起送入LLM模型，得到当前任务的输出。在得到最终任务的结果前，将输入输出先存入memory，在之后的任务中可以参考这部分任务的信息。





**THE END**

