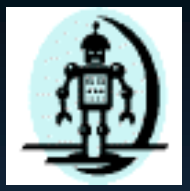
Sztuczna Inteligencja
Reprezentacja wiedzy V
Agenci

Włodzisław Duch
Katedra Informatyki Stosowanej UMK
Google: Włodzisław Duch

Strona wykładów

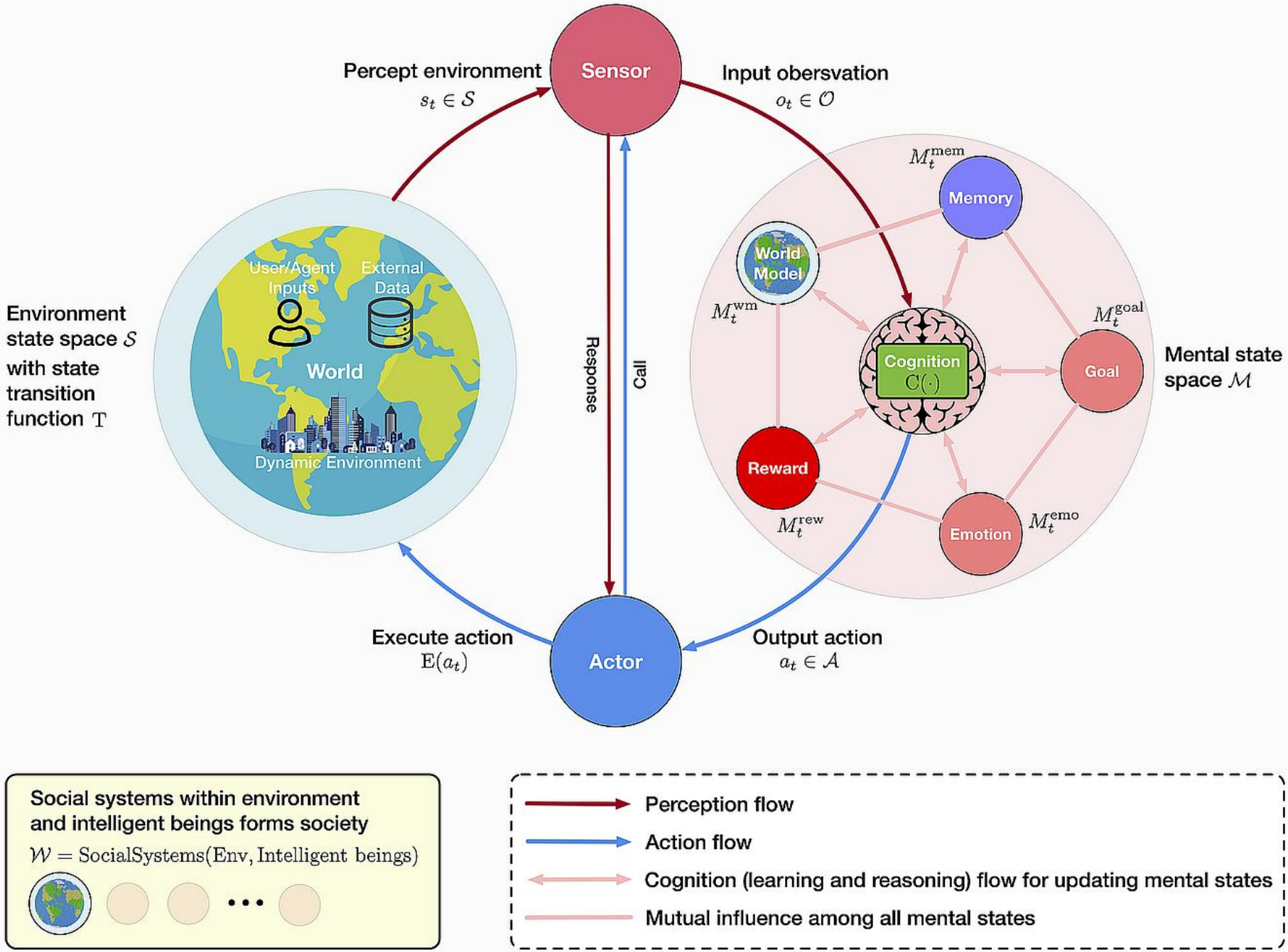
3 prawa robotyki



3 prawa robotyki Issaca Asimova
(z Podręcznika Robotyki, wyd. 56, AD 2058)

- Robot nie może skrzywdzić człowieka swoim działaniem, jak też nie może go skrzywdzić przez zaniechanie działania, które mógłby podjąć.
- Robot musi zawsze wykonywać polecenia człowieka, poza przypadkami, w których zabrania mu tego Pierwsze Prawo.
- Robot musi chronić siebie przed uszkodzeniami we wszystkich przypadkach, w których nie zabraniu mu tego Pierwsze lub Drugie prawo.

Taki robot musi mieć naprawdę głębokie zrozumienie świata …


Percept environment
$s_t \in \mathcal{S}$
Sensor
Input obersvation
$o_t \in \mathcal{O}$
Environment state space $\mathcal{S}$ with state transition function T
User/Agent Inputs
External Data
World
Dynamic Environment
Response
Call
M_t^{mem}
Memory
World Model
M_t^{wm}
Cognition
$\mathrm{C}(\cdot)$
M_t^{goal}
Goal
Mental state space $\mathcal{M}$
Reward
M_t^{rew}
Emotion
M_t^{emo}
Execute action
$\mathrm{E}(a_t)$
Actor
Output action
$a_t \in \mathcal{A}$
Social systems within environment and intelligent beings forms society
$\mathcal{W} = \mathrm{SocialSystems}(\mathrm{Env}, \mathrm{Intelligent\ beings})$
Perception flow
Action flow
Cognition (learning and reasoning) flow for updating mental states
Mutual influence among all mental states

Agenci

Agent programowy to:

- Każdy system, który odbiera informacje z otoczenia i reaguje na te informacje.
- Cel: stworzyć agentów, którzy są samowystarczalni, zdolni do wykonywania wyspecjalizowanych funkcji.

Człowiek:

Sensory: oczy, uszy, skóra, nos, język ...

Efektory: ręce, nogi, język, usta ...

Robot:

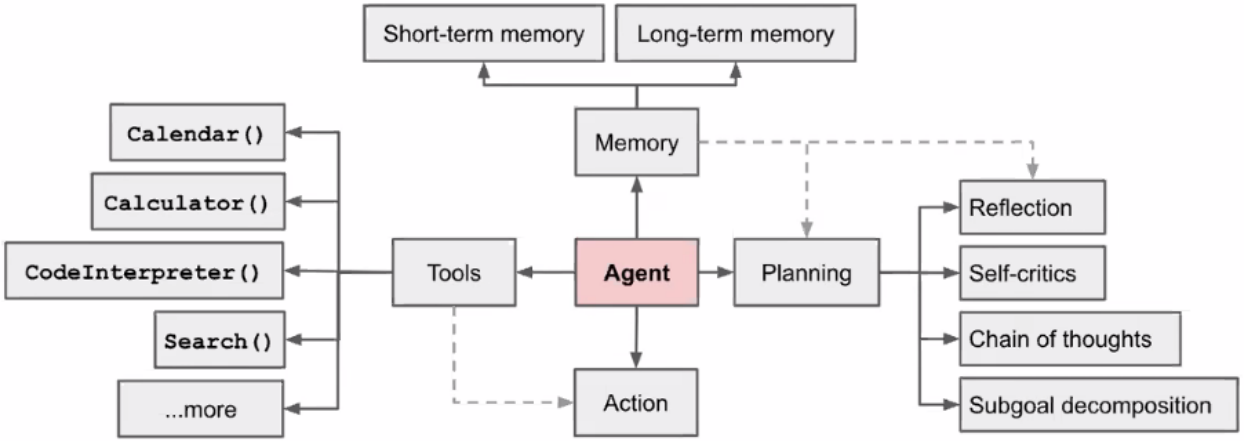
Sensory: kamery, czujniki podczerwieni, sonary, mikrofony ...

Efektory: koła, motory, manipulatory ...

Programy:

Sensory wirtualne: ciągi bitów, zawierające informacje dochodzące ze środowiska ...

Efektory wirtualne: ciągi bitów, przesyłane do środowiska...



Agent – program który potrafi postrzegać swoje otoczenie i na tej podstawie podejmować decyzje i wykonywać akcje prowadzące do osiągnięcia postawionego przed nim celu

Lilian Weng/ OpenAI, 2023

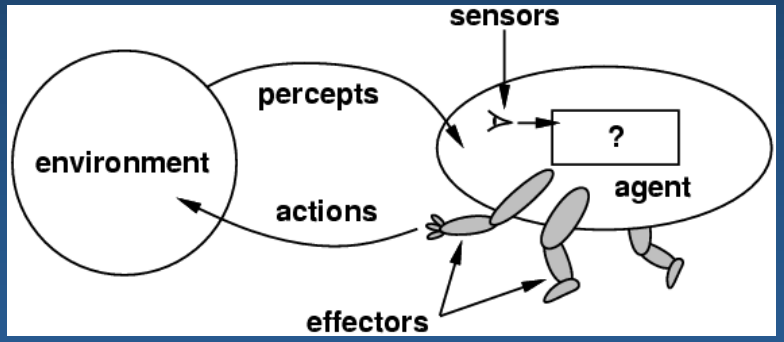
- **Carl Hewitt**, model aktorów sformułowany w *Viewing Control Structures as Patterns of Passing Messages* Journal of Artificial Intelligence, 1977
- „The purpose of this paper is to discuss some organizational aspects of programs using the actor model of computation. In this paper we present an approach to modelling intelligence in terms of a society of communicating knowledge-based problem-solving experts. In turn each of the experts can be viewed as a society that can be further decomposed in the same way until the primitive actors of the system are reached. We are investigating the nature of the communication mechanisms needed for effective problem-solving by a society of experts and the conventions of discourse that make this possible."

- Model aktorów
  - Aktor jest podstawową jednostką przetwarzania współbieżnego
  - Aktor posiada stan i zachowanie
  - Aktorzy komunikują się wyłączne poprzez komunikaty przekazywane asynchronicznie
  - Aktorzy nie współdzielą swoich stanów
  - Aktor w odpowiedzi na komunikat może: przesłać komunikat do innego aktora, utworzyć innego aktora, zmienić swoje zachowanie przy odbiorze kolejnego komunikatu
  - Niedostępność aktora nie ma wpływu na reszte systemu

Działanie agenta

Idealny racjonalny agent powinien:

- Posiadać miarę oceny swojego działania z punktu widzenia stawianych przed nim celów.
- Wykorzystywać informację zawartą w nadchodzących postrzeganych danych tak, by optymalizować tą miarę w oparciu o dostępną wiedzę.



Idea: Marvin Minsky, Society of mind. Simon & Schuster 1986
Umysł = to co robi mózg.
Inteligencja: wiele prostych procesów, które ze sobą współpracują.

Jak?

Działanie racjonalne – oparte na rozumowaniu.

Mając różne możliwości działania, należy wybrać (wyszukać) najlepszą z punktu widzenia celu.

Obiektywne miary jakości działania biorą pod uwagę.

- Szansa na sukces?
- Czas potrzebny do rozwiązania problemu?
- Koszt tego rozwiązania?

Automatyczny kierowca:

Osiągnięcie celu podróży, przestrzeganie przepisów i bezpieczeństwo jazdy, wybieranie najkrótszej drogi.

Agent wyszukujący sklepy i porównujący ceny:

Czy znalazł wszystkie oferty? Jak szybko? Czy uwzględnił wszystkie preferencje użytkownika?

Wiedza i działanie

Działanie racjonalne wymaga wiedzy:

- Wiedza agenta jest zawsze ograniczona, np. z powodu ograniczeń percepcji lub braku doświadczenia, co czasami nie pozwala na znalezienie najprostszego rozwiązania.

- Zachowanie zależy od relacji pomiędzy postrzeganym (senso) a wykonywanym (motorycznych).
- Relacje w prostych przypadkach mogą być zapisane w tabeli.
- Relacje mogą być dane w postaci funkcji.

Agent powinien działać autonomicznie:

- Autonomia wymaga zmiany zachowania w zależności od napływających danych.
- Nabywanie doświadczenia wymaga uczenia się agenta.
- W najprostszym razie uczenie się jest zmianą programu przez twórcę agenta.

Struktura

Agent = architektura systemu + program

- Program: zbiór algorytmów określających relacje pomiędzy spostrzeżeniami a działaniami agenta.
- Architektura:
  - Sprzęt i środowisko programowe.
  - Dostęp do danych zewnętrznych (percepcji).
  - Kontrola wykonywania programu.
  - Dostęp do efektorów.

Agent scharakteryzowany jest przez:

- Postrzeżenia.
- Akcje.
- Cele.
- Środowisko, w którym działa.

Przykłady agentów

Agent	Postrzeżenia	Akcje	Cele	Środowisko
Robot przemysłowy	Piksele z kamery	Ruch manipulatora	Sortowanie części	Taśma montażowa
Kontroler rafinerii	Ciśnienie, temp, par. chemiczne	Regulacja zaworów	Maks. wydajności	Rafineria
Analizator obrazów	Piksele z kamer	Segmentacja obrazu	Prawidłowa segmentacja	Satelita, skrzyżowanie
Diagnostyka medyczna	Testy medyczne	Pytania, testy, terapia	Poprawa zdrowia	Szpital
Automatyczny kierowca	Czujniki, kamery	Kierowanie, wybór drogi	Bezpieczny przejazd	Ulica, ruch drogowy
Agent sklepowy	Ceny, oferty, sklepy.	Negocjacje, wyszukiwanie	Prezentacja, porównanie	Internet

RoboCup i wielu agentów

Roboty grają w piłkę nożną.

Postrzeżenia ?

Akcje ?

Cele ?

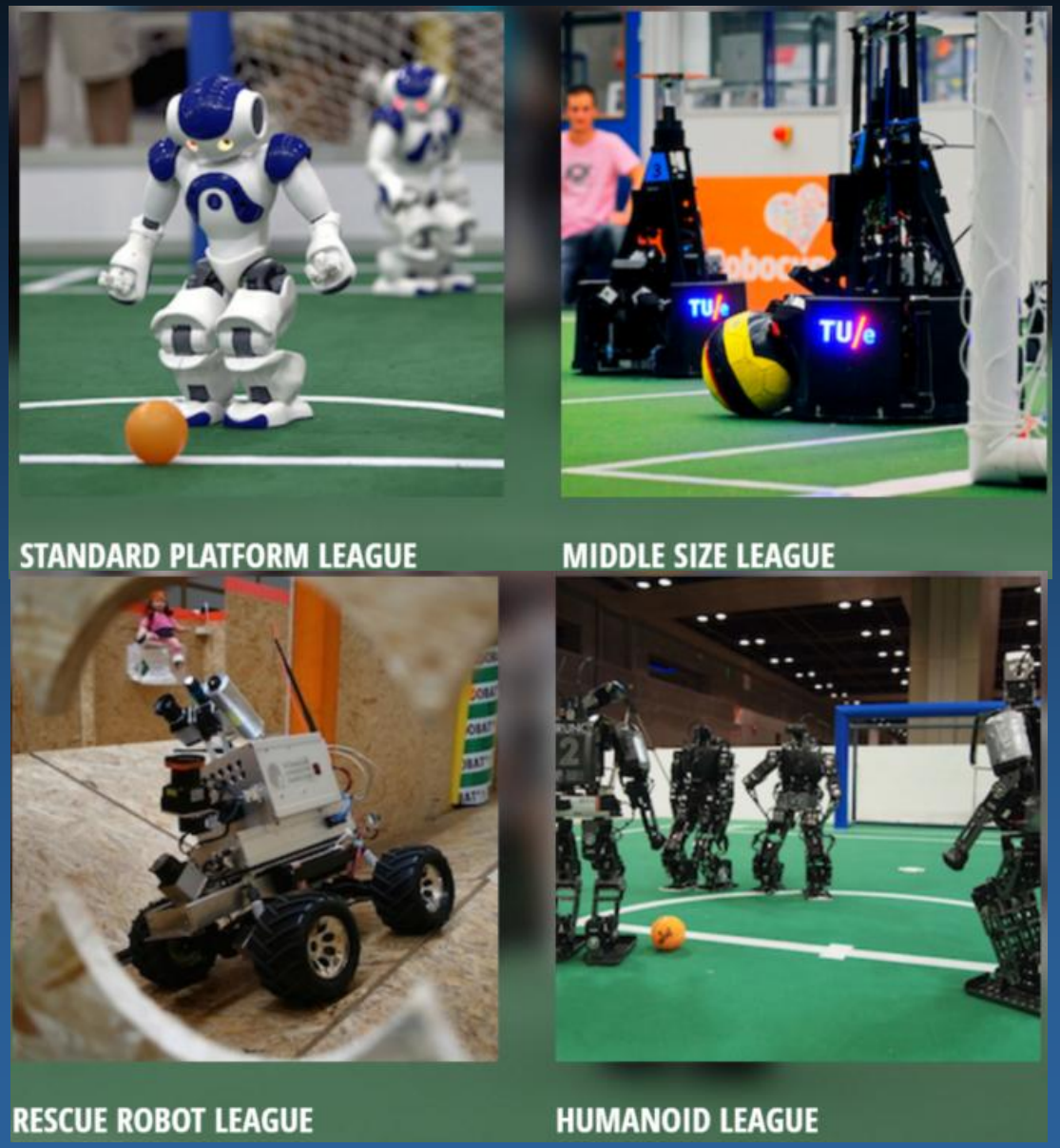
Środowisko ?

Drużyna: system wieloagentowy.

Wiele zastosowań.

Cel RoboCup: do 2050 r humanoidalne roboty mają wygrać z mistrzami świata.

Artemis zwycięzca Robocup



Typy agentów

Agenci mogą być prostymi lub złożonymi systemami.

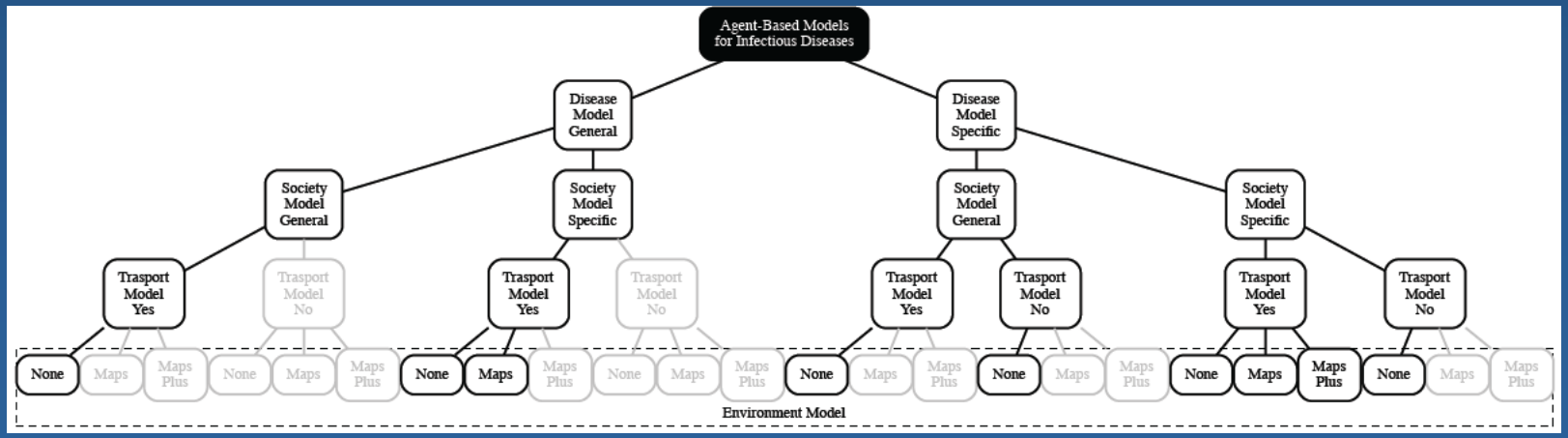
- Agent wykorzystujący proste odruchy, nie ma pamięci, wykorzystuje reguły definiujące działanie, zwykle korzysta z systemu produkcyjnego (regułowego).
- Agent oparty na tabeli spostrzeżeń/działań, wykorzystuje duże tabele by wybrać następne działanie w zależności od sytuacji; tabela służy tu jako pamięć i umożliwia modelowanie przejść pomiędzy stanami wewnętrznymi.
- Agent oparty na odruchach z pamięcią stanu wewnętrznego, np. agent Markowa pamiętający swój poprzedni stan; tylko stan bieżący ma wpływ na podjętą decyzję.
- Agenci posiadający cele wykorzystują informację o bieżącym stanie i posiadający reprezentację możliwych stanów, ocenianych z punktu widzenia przyjętych celów.
- Agenci maksymalizujący funkcje użyteczności.
- Rise of AI Agents (YouTube)

Agent Tablicowy

Agenci posługujący się tablicami korelacji:

```
function TABLE-DRIVEN-AGENT(percept) returns action
  static: percepts, a sequence, initially empty
          table, a table, indexed by percept sequences, initially fully specified

  append percept to the end of percepts
  action ← LOOKUP(percepts, table)
  return action
```

- Agent grający w szachy musiałby mieć tablicę o rozmiarze rzędu 35^{120}.
- Trudno jest zbudować tabelę wszystkich możliwych powiązań.
- Agent nie jest autonomiczny, bo przy zmianie środowiska (np. reguł gry) nie potrafi sobie poradzić, wszystkie akcje trzeba z góry przewidzieć.
- Liczne modele epidemologiczne wykorzystują technologię wieloagentową.



Agent z prostymi refleksami

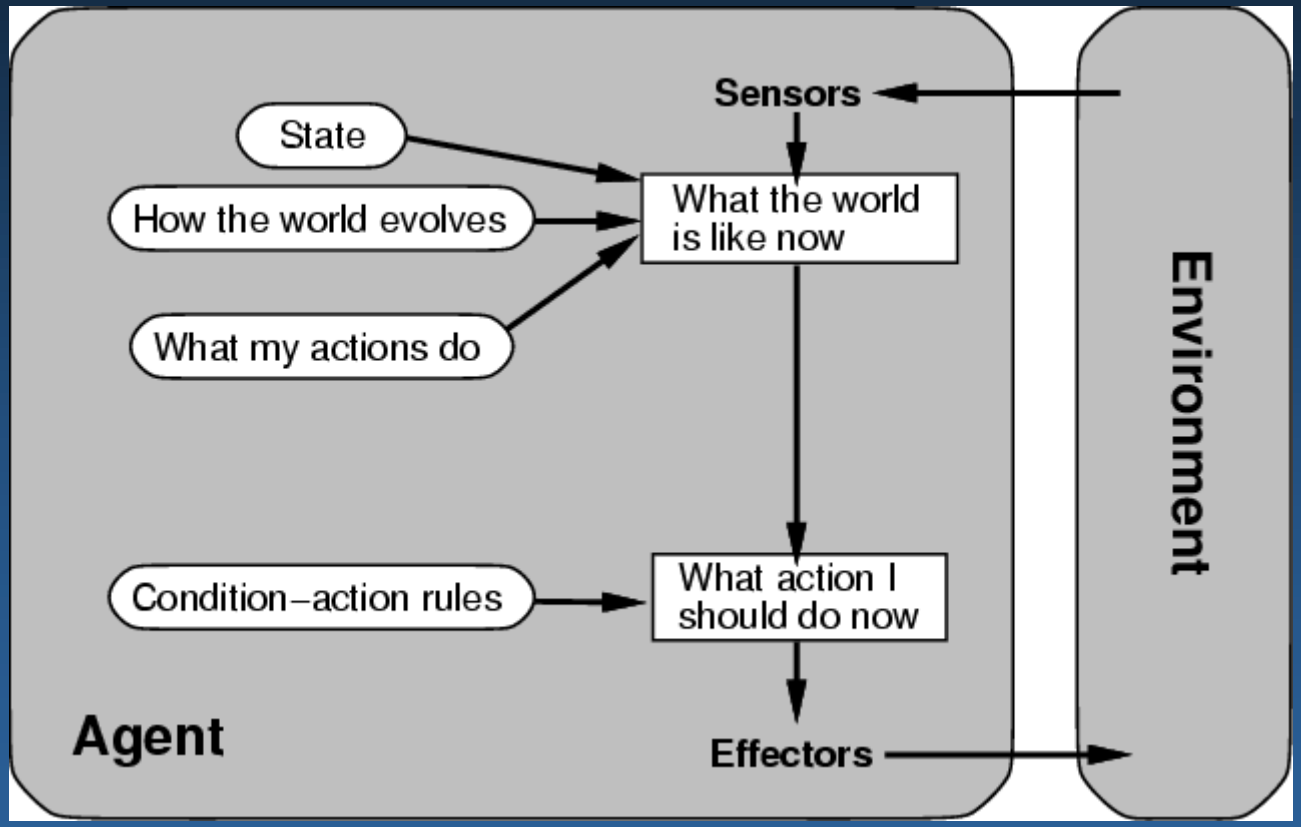
Agenci określający stan i stosujący regułę dla tego stanu:

```
function SIMPLE-REFLEX-AGENT(percept) returns action
    static: rules, a set of condition-action rules

    state ← INTERPRET-INPUT(percept)
    rule ← RULE-MATCH(state, rules)
    action ← RULE-ACTION[rule]
    return action
```

- Reguły typu: warunek-akcja, np.
  - Jeśli światło(czerwone) to przyspieszaj
  - Jeśli światło(zielone) to hamuj
- Decyzje podejmowane tylko w oparciu o spostrzeżenie, rozważana jest tylko jedna decyzja.
- Zmiana pasa w czasie jazdy wymaga pamięci stanu poprzedniego: czy widać coś we wstecznym lusterku.
- Pamięć jest czasem niezbędna.

Agent ze stanami wewnętrznymi

Agent ma model środowiska i rezultatów swojego działania.



Agent ze stanami wewnętrznymi

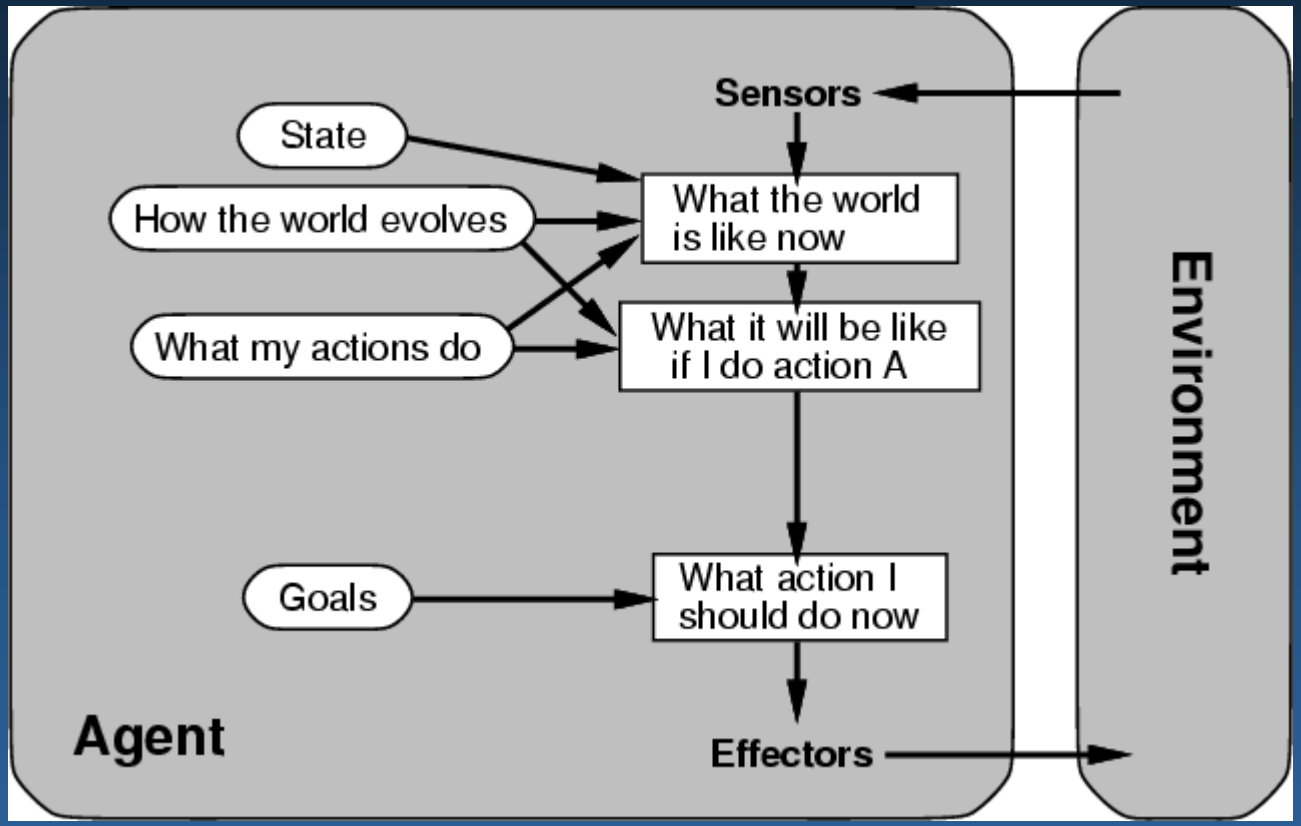
Agenci zmieniający swoje stany wewnętrzne na skutek percepcji.

```
function REFLEX-AGENT-WITH-STATE(percept) returns action
  static: state, a description of the current world state
          rules, a set of condition-action rules

  state ← UPDATE-STATE(state, percept)
  rule ← RULE-MATCH(state, rules)
  action ← RULE-ACTION[rule]
  state ← UPDATE-STATE(state, action)
  return action
```

- Stany wewnętrzne pozwalają na zróżnicowanie reakcji przy tych samych stanach percepcyjnych, w zależności od przeszłego działania agenta.
- Wymaga modelu świata, reprezentacji jego zmian i wpływu własnych działań na stan świata.
- Ograniczenie się tylko do poprzedniego stanu znacznie upraszcza budowę agenta (łańcuchy Markowa).

Agent z celami

Agenci reprezentujący stany wewnętrzne będące modelem środowiska i stany pożądane.



Agent z celami

Cel: opis sytuacji pożądanej.

Ogólny: osiągnięcie przewagi w grze.

Szczegółowy: dotarcie do planowanego miejsca.

Ostatni stan nie wystarczy, konieczny jest opis stanu oparty na reprezentacji skutków wirtualnych akcji.

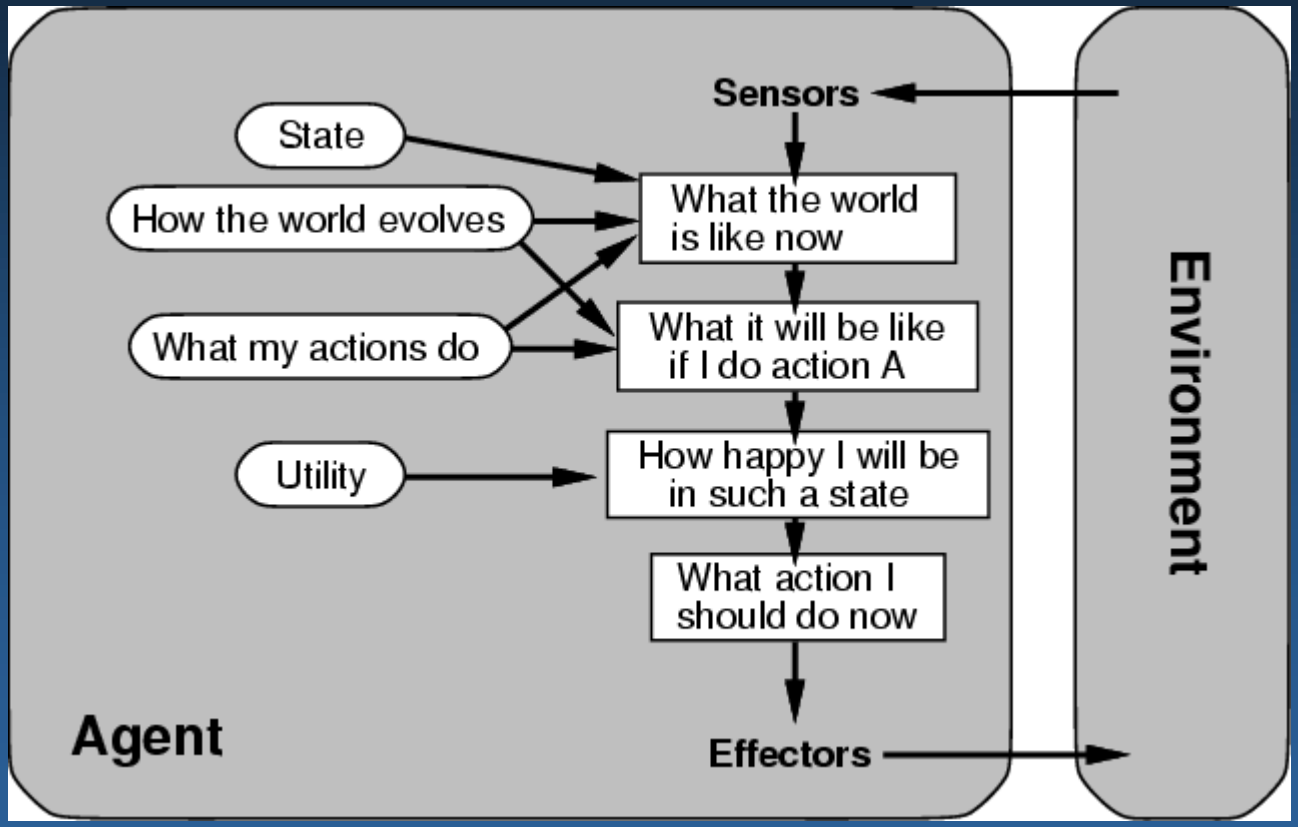
Stan wewnętrzny: pozwala śledzić zmiany stanu świata, nie tylko bezpośrednio spostrzegane zmiany.

Rozważanych jest wiele decyzji: co się stanie jeśli ...

Podstawą działania jest planowanie i szukanie.

Agent z f. użyteczności

Stany oceniane za pomocą f. użyteczności.



Agent z f. użyteczności

Preferowane stany wewnętrzne mają wyższe wartości funkcji użyteczności – modeluje to ukierunkowanie emocjonalne.

Np. należy wybrać szybsze, bezpieczniejsze lub pewniejsze środki transportu.

F. użyteczności f: stan => U(stan)

F. użyteczności umożliwia:

- rozstrzyganie pomiędzy sprzecznymi celami.
- szansą na sukces a wagą celu.

Szukanie – związane z celami, działania bezpośrednie.

Użyteczność – związana z osiąganiem dalekosiężnych celów (np. wygranej lub zdobyciu przewagi w grach).

Agent ds. zakupów

Agent krążący po sklepie powinien orientować się w przestrzeni:

- agent refleksyjny – ma szczegółową mapę (przy zmianie położeń ma kłopoty).
- agent celowy – sam tworzy mapę i wykorzystuje ją do rozumowania, dostosowuje się do zmian, np. wyprzedaży.

Wybieranie przedmiotów: musi rozpoznawać przedmioty.

- agent refleksyjny – zbiera przedmioty dobrze wyglądające.
- agent celowy – posługuje się lista zakupów, ocenia wybierane przedmioty. Może planować, np. jeśli nie znajduje poszukiwanego przedmiotu planuje inne sposoby jego zakupu.
- agent z f. użyteczności – uwzględnia jakość i cenę.

Przykłady agentów

Działa wiele agentów programowych:

- donoszące o błędach w programie – czuwają i przy wykryciu błędu zbierają informację i poszukują właściwego odbiorcy.
- agent pocztowy – dostarcza pocztę, poszukując odbiorcy.
- agent WWW – poszukuje informacji w sieci korzystając z różnych mechanizmów wyszukiwawczych, grafów wiedzy.
- agent bibliograficzny – czuwa nad aktualizacją bibliograficznych linków.
- agent bioinformatyczny – śledzi pojawianie się nowych informacji i nowych baz informatycznych, dokonuje konwersji formatów danych.
- agent prowadzący samochód: 27.06.2011 Nevada zalegalizowała samochody z automatycznym kierowcą (Google cars).
- Robot Proces Automation (RPA), system regułowy automatyzacji schematycznych procesów, pozwala agentom na wykonywanie powtarzalnych czynności.
- Inteligentni agenci LLM – planują, rozwiązują problemy, działają sprawczo.

Agenci na co dzień

Microsoft Agent technology (od 1997!), używający animowanych postaci:

- powiadamia i czyta pocztę elektroniczną;
- zbiera i odczytuje informacje z giełdy, serwisów pogodowych i wybranych stron WWW, np. ostatnie wiadomości, ostrzeżenia przed wirusami;
- dostarcza pozdrowienia i wiadomości odczytujące je lub odśpiewując;
- przypomina o zdarzeniach z kalendarza: spotkaniach, rocznicach etc ;
- opowiada kawały, ściągając je z internetu; konwersja formatów danych;
- znajduje interesujące strony WWW i je poleca.

Obecnie zastąpiła to Cortana i MS Bing, ale bez awatara, bez polskiej wersji (4/2022). Nie ma też polskich wersji Alexy czy Siri i innych agentów.

Telegram Bots pozwala na przesyłanie wiadomości używając botów.

Lista polskich botów.

BotPrize new competition: czy gram z komputerem czy człowiekiem?

Świat agentów

Każda duża firma IT tworzy swoich agentów, głównie chatboty.

CyberBuddy korzysta z old Microsoft Agent Talking Buddy (tylko historia).

Dodaj swojemu botowi inteligencji z MS Cognitive Services
Microsoft Azure Bot Service

Alexa Prize Competition 3.5 M$ nagrody, "advancing Conversational AI"

Ultra Hal Assistant – Hal, agent od Zabaware; podtrzymuje dialogi, pamięta.

Long list of ChatBots narzędzia do tworzenia swoich agentów.

Zbuduj sobie bota: Pandora chatbot. albo Telegram Bot

Polskie boty (stare) https://chatbot.pl/ Bots.ai (Warta)

Desperados Bot pierwsza loteria w Polsce prowadzona przez chatbota.

BotPrize nagroda 1.000.000 $.

- Digital Twin Applications i technologia cyfrowych bliźniaków MS Azure.

WD: Świat Bytów Wirtualnych, wykład (6 godzin) z demonstracjami
Program (PDF), prezentacja część 1 (PPT) oraz część 2 (PPT) (2005).

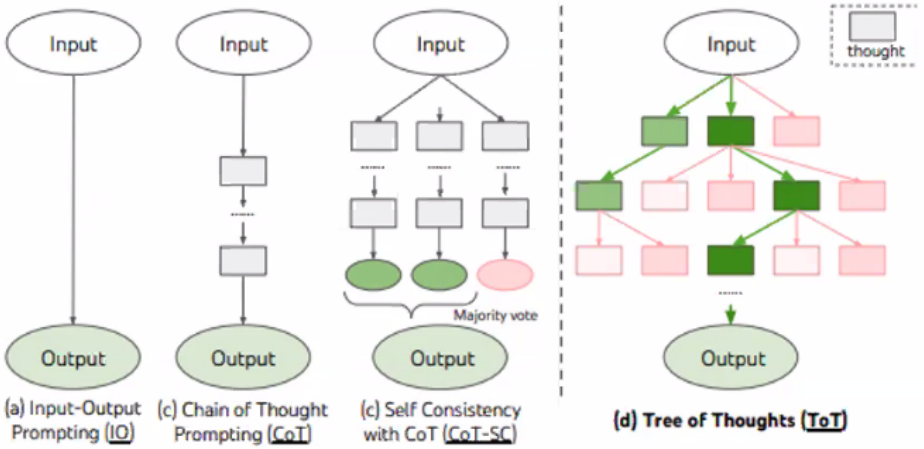
Architektura rozwiązań agentowych - pryncypia

Asynchroniczna komunikacja	Zdarzeniowość	Skalowalność	Rozproszenie
Modularność	Rozszerzalność	Obserwowalność	Interoperacyjność
Bezpieczeństwo	Wytłumaczalność	LLM na żądanie	Optymalizacja

Planowanie - strategie agentów

Planowanie

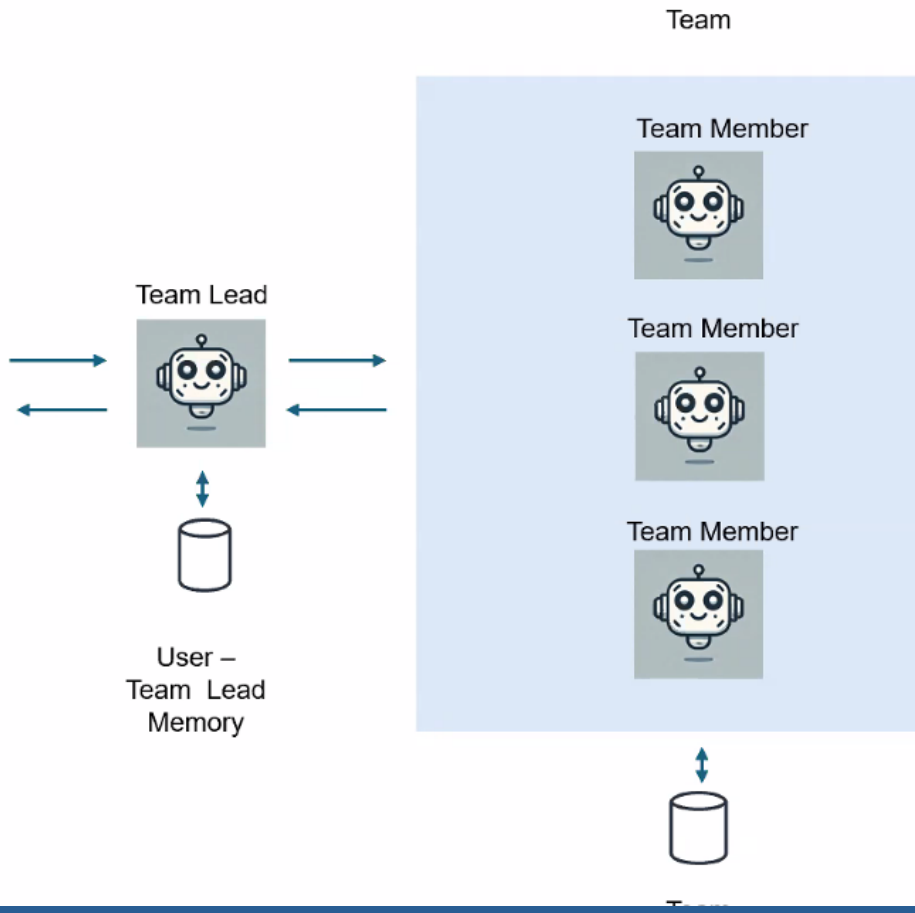
- Chain-of-Tought Prompting, Jason Wei, 2023
- Tree of Thoughts, Shunyu Yao, 2023
- LLM Planning, Bo Liu, 2023
- ReAct: Synergizing Reasoning and Acting in Language Models, Shunyu Yao, 2023 – Though, Action, Observation loop



Agenci

Agent jako narzędzie

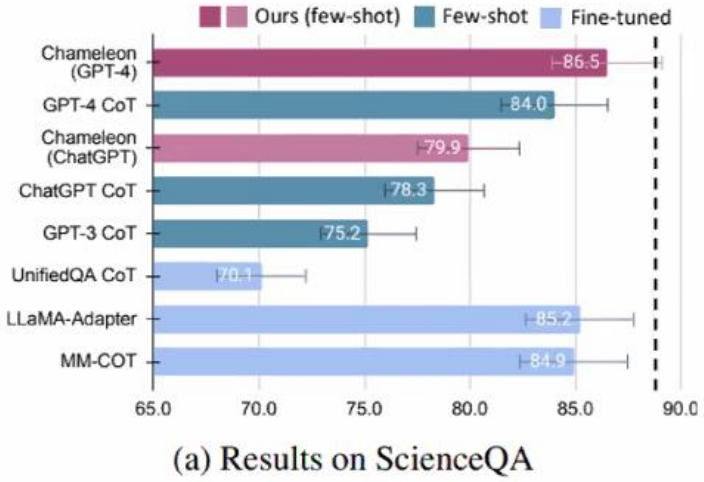
- Agent może wywołać innego agenta w scenariuszu team leader – team member
- Wzorzec agent jako narzędzie pozwala na dynamiczną delegację zadań do wyspecjalizowanych agentów, zrównoleglenie ich działania
- Agenci pracujący w zespole mogą współdzielić pamięć, historię rozmowy



Narzędzia agentów

Tool use (= function calling)

- Chameleon: Plug-and-Play Compositional Reasoning with Large Language Models, Pan Lu, 2023



(a) Results on ScienceQA

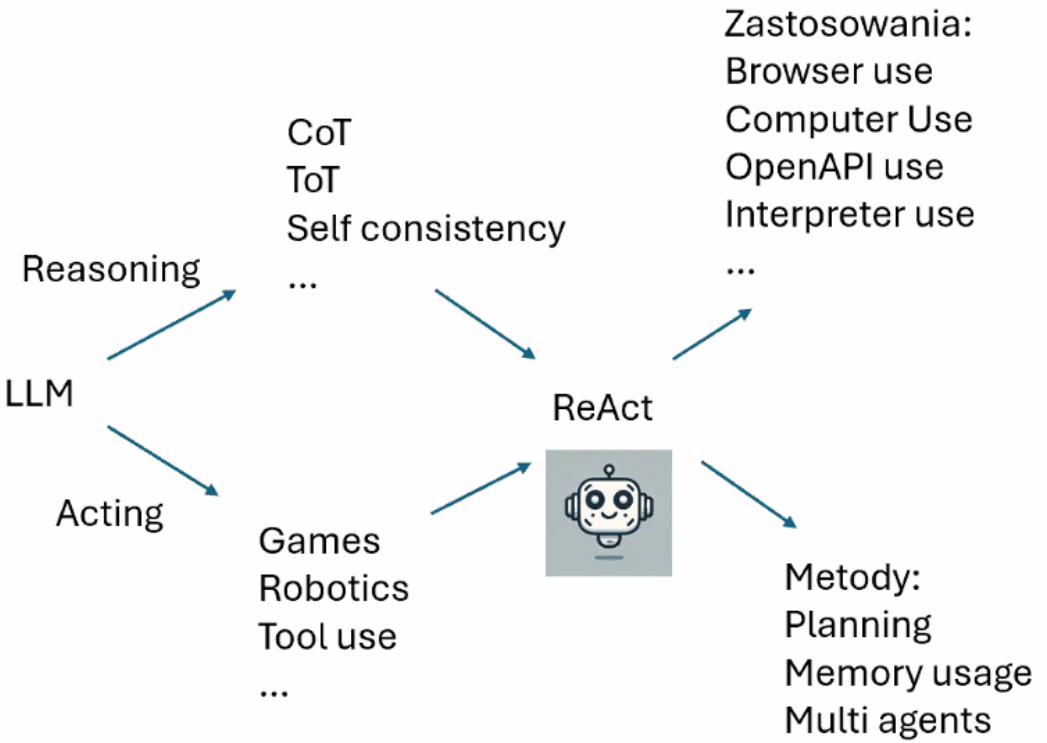
- Tool – funkcja o określonej nazwie, opisie i parametrach
- Agent posiada rejestr funkcji dla niego udostępnionych
- Funkcja może być przekazana w wywołaniu API modelu LLM w postaci JSON, który to format jest rozpoznawalny przez model
- Agent execution loop

Agenci

ReAct agent

ReAct: Synergizing Reasoning and Acting in Language Models, Shunyu Yao, 2023

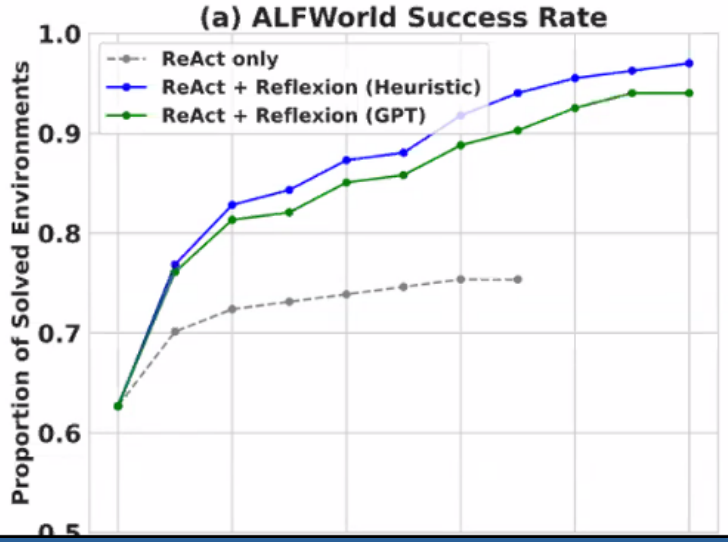
„ In this paper, we explore the use of LLMs to generate both reasoning traces and task-specific actions in an interleaved manner, allowing for greater synergy between the two: reasoning traces help the model induce, track, and update action plans as well as handle exceptions, while actions allow it to interface with and gather additional information from external sources such as knowledge bases or environments.”



Agenci

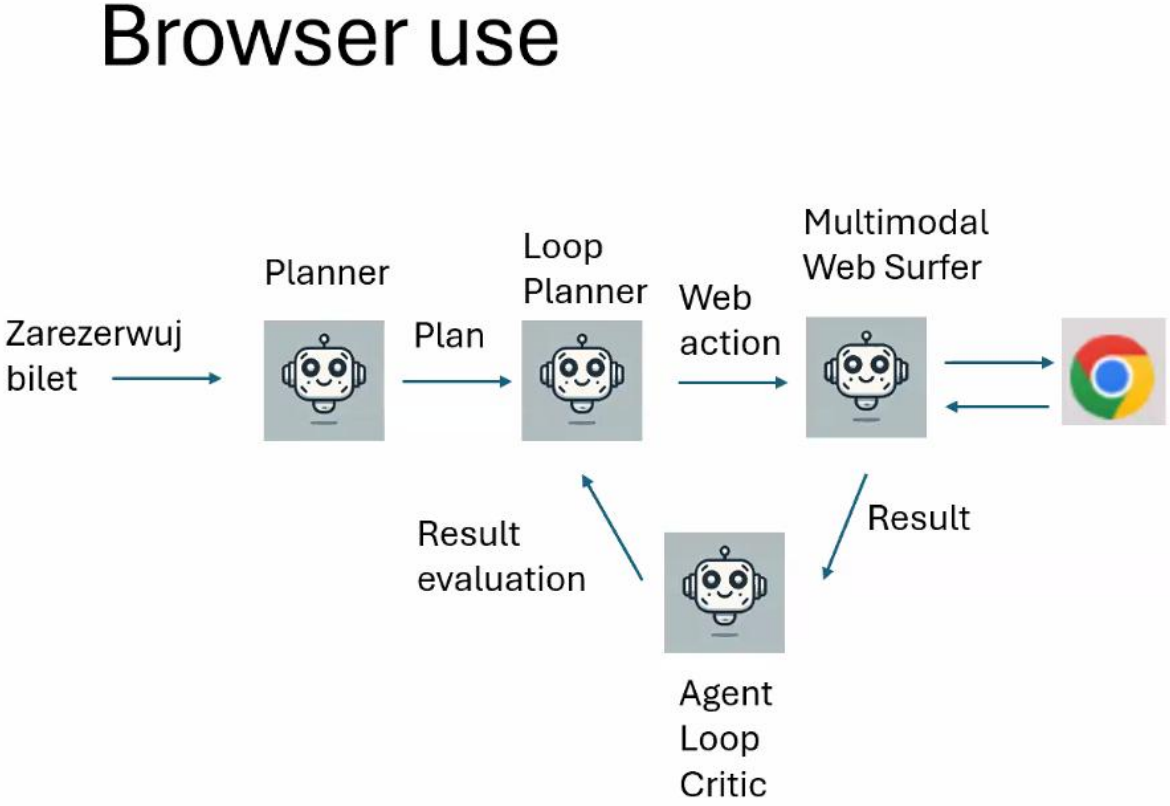
Reflection - agent uczący się na własnych błędach

- Reflexion: Language Agents with Verbal Reinforcement Learning, Noah Shinn, 2023



- Algorytm wykorzystuje werbalne uczenie ze wzmocnieniem (verbal RL)
- Agent za pomocą LLM generuje odpowiedzi i akcje
- Funkcja ewaluacyjna ocenia wynik akcji podejmowanych przez agenta (sygnał zwrotny)
- Mechanizm refleksji na podstawie sygnału zwrotnego generuje za pomocą LLM werbalną refleksję
- W kolejnych iteracjach podczas generowania przez agenta akcji kontekst jest wzbogacany o gromadzone w pamięci refleksje

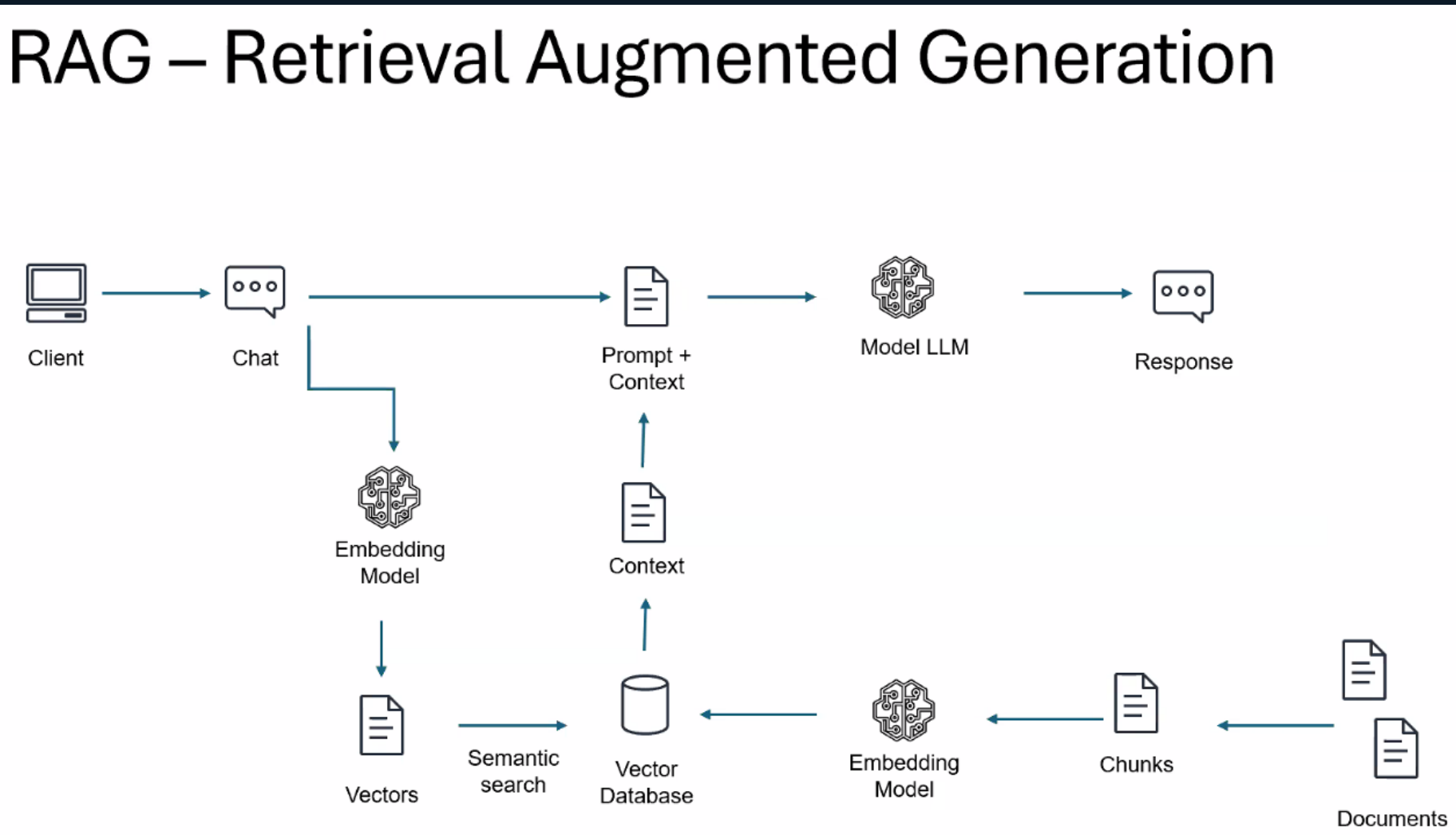
Sterowanie przeglądarką




```python
# Initialize the model client
model_client = OpenAIChatCompletionClient(model="gpt-4o")

# Create the MultimodalWebSurfer agent
web_surfer = MultimodalWebSurfer(
    "web_surfer",
    model_client,
    headless=False,
    start_page="https://www.google.com/travel/flights",
    debug_dir="./debug_logs",
    to_save_screenshots=True
)

# Create the team with proper sequence
team = RoundRobinGroupChat(
    [
        initial_planner,     # First breaks down the task
        agent_loop_planner, # Manages the loop
        web_surfer,          # Performs web actions
        agent_loop_critic,  # Evaluates results
    ],
    termination_condition=TextMentionTermination("exit")
)

# Run the team on the task
await Console(team.run_stream(task=initial_task))
```

Implementacja z użyciem Microsoft Autogen 0.4, interakcja z przeglądarką za pomocą PlayWright

RAG


RAG – Retrieval Augmented Generation
Client
Chat
Prompt + Context
Model LLM
Response
Embedding Model
Context
Vectors
Semantic search
Vector Database
Embedding Model
Chunks
Documents

Agenci przejmują komputer

Computer use

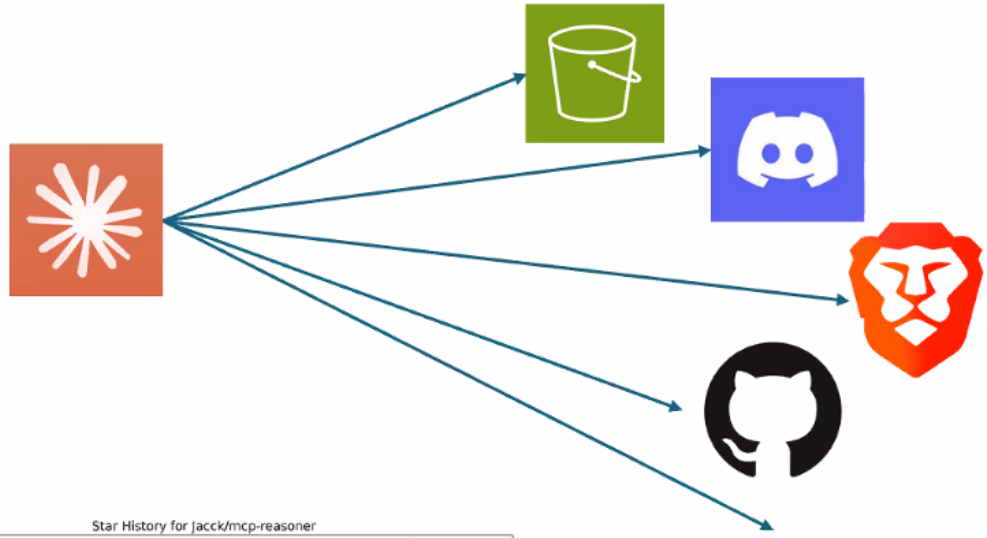
- Open Interpreter pozwala agentowi LLM uruchamiać lokalnie w terminalu komendy systemu operacyjnego, programy
- Anthropic Computer Use, 10.2024
- Microsoft Computer Use, 11.03.2025

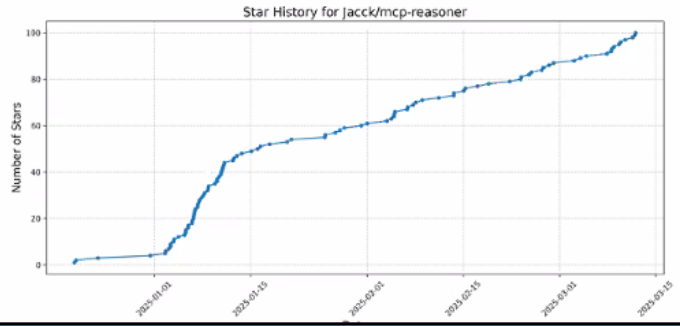
```
▌ Model set to gpt-4-turbo

Open Interpreter will require approval before running code.

Use interpreter -y to bypass this.

> open firefox

  To open Firefox on your Windows machine,
  we can use a PowerShell command that executes the program.

  Plan
   1 Use PowerShell to execute a command that opens Firefox.

  Let's start by executing the code.
  Would you like to run this code? (y/n)
  Start-Process firefox
  Copyright (C) Microsoft Corporation. All rights reserved.
  PS C:\Users\user>
  PS C:\Users\user> try {
  >> Write-Output ""
  >> Start-Process firefox
  >> } catch {
  >>     Write-Error $_
  >> }
  >>
  PS C:\Users\user> Write-Output ""

  Firefox has been successfully opened on your machine.
```

OpenAI Operator, Anthropic's Claude Computer Use API, Google Gemini 2.0 computer use agents.

Wzorce interakcji agentów – rozszerzony LLM

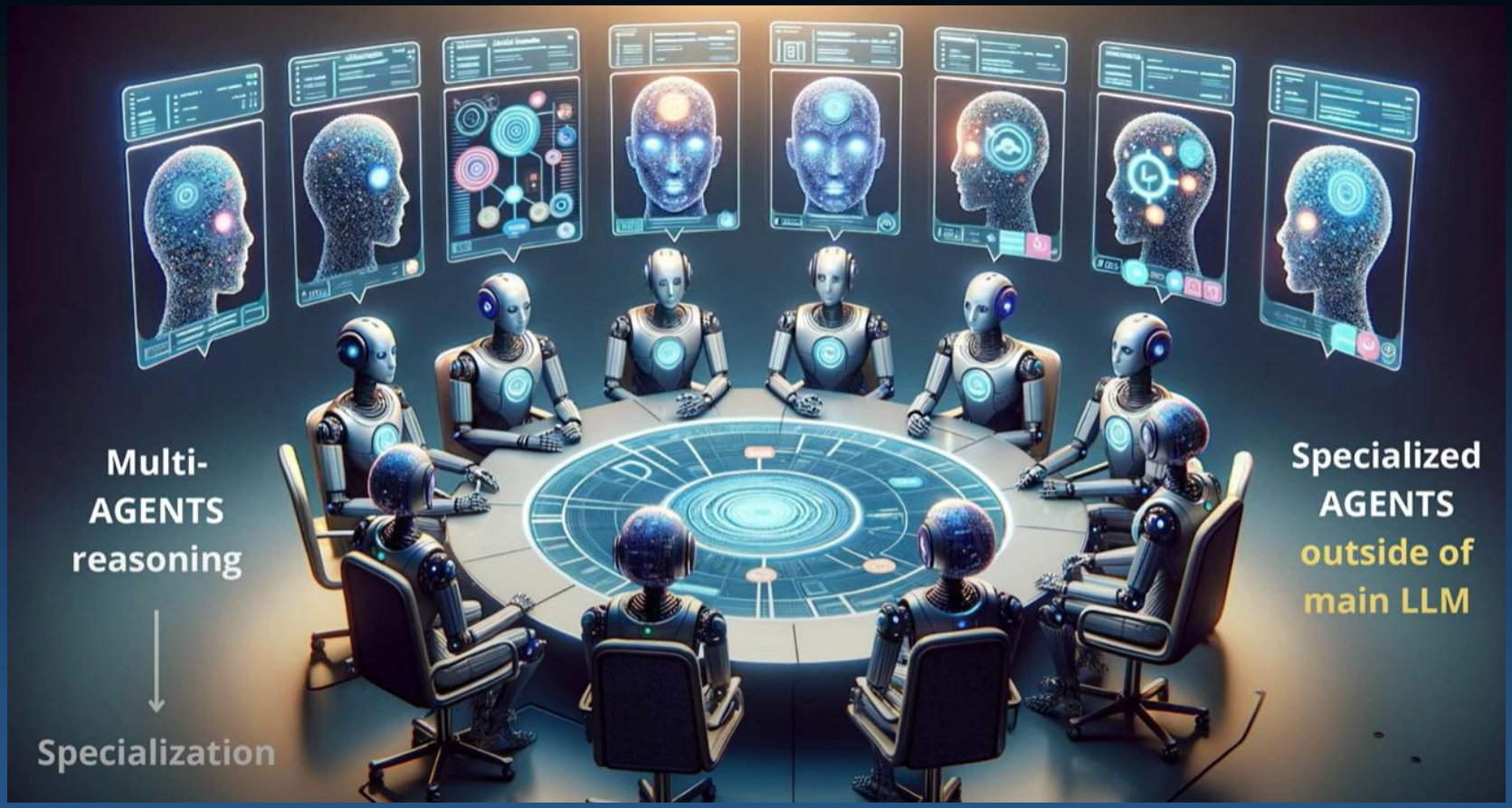
- Anthropic, „Building effective agents”, 19 grudnia 2024
- LLM rozszerzony o podstawowe narzędzia
- Model Context Protocol (MCP) to protokół standaryzujący dostarczanie kontekstu do LLM
- Claude Desktop z MCP jest agentem
- Claude Desktop może używać inny LLM





mcp-reasoner

100* na github



Tysiąc agentów działających jednocześnie? Lindy.ai integruje się z wieloma aplikacjami. Sztuczne społeczeństwa agentów mogą rozwiązywać problemy, rozwijać osobowości, tworzyć subkultury, symulować zachowania społeczne.

Takata, R., Masumori, A., & Ikegami, T. (11/2024). *Spontaneous Emergence of Agent Individuality through Social Interactions in LLM-Based Communities*.

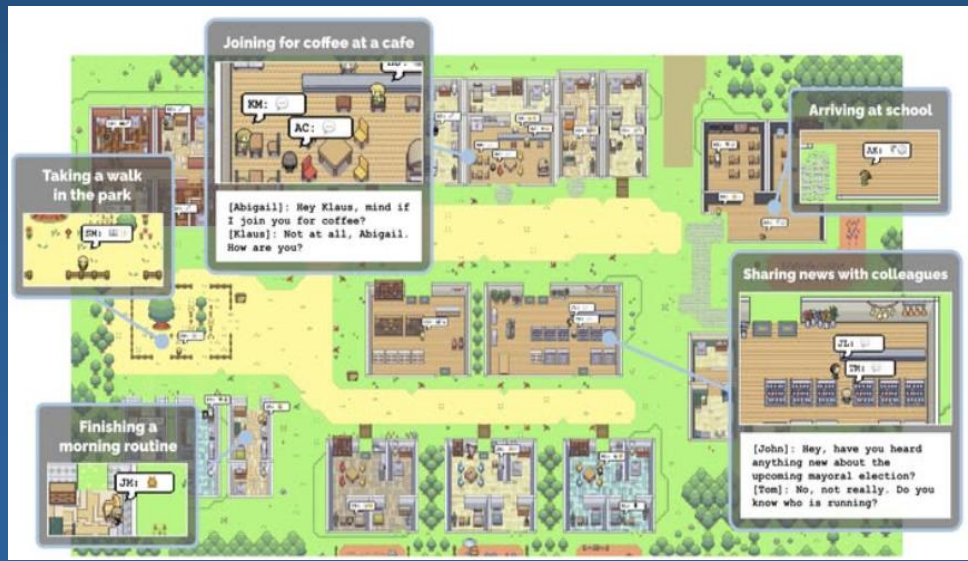
Simulacrum 1052 botów

Park, J. S. I inn. (2024). *Generative Agent Simulations of 1,000 People*.

Architektura agentów symuluje postawy i zachowania 1052 prawdziwych osób. Przeprowadzono 2-godzinne wywiady z ludźmi.
Agenci byli poinstruowani by naśladowali daną osobę w oparciu o dane z tych wywiadów. Ankiety, inwentarze, behawioralne gry ekonomiczne zostały wykorzystane do pomiaru, jak dobrze agenci odtwarzają postawy i zachowania poszczególnych osób.

Agenci: Dokładne przewidywali cechy osobowości po dwóch tygodniach.

Pomyślnie odtworzono 5 wyników eksperymentów z zakresu nauk społecznych.
Współczynnik dopasowania w odpowiedziach General Social Survey w porównaniu z ludzką spójnością był na poziomie 85%.
Agenci replikujący ludzkie zachowania mają wiele zastosowań, w tym w symulakrum szpitala.



Symulakrum szpitalne

Wirtualny szpital agentów (Li et al, 5/2024) symuluje cały proces leczenia choroby, z autonomicznymi agentami w roli pacjentów, pielęgniarek, laborantów i lekarzy, pracujących w różnych oddziałach.

Agent lekarz uczy się rozpoznawać i leczyć chorobę.

Lekarze gromadzą doświadczenia zarówno z udanych, jak i nieudanych przypadków. Objawy i wynik testów dotyczą rzeczywistych przypadków medycznych.

Po obsłużeniu ~ 10 000 pacjentów agent lekarz osiągnął dokładność 93% dla zbioru MedQA chorób układu oddechowego, SOTA!

W Metaversum wirtualni lekarze mogą z powodzeniem zastąpić „wujka Google”.

Reprezentacja wiedzy: przykładowe pytania

- Co reprezentują elementy sieci semantycznej?
- Jak prowadzi się rozumowanie dla sieci semantycznych?
- Jak reprezentowane są słowa w mózgu? Jak to się ma do sieci semantycznych?
- W jaki sposób można przewidzieć aktywację mózgu dla nowych pojęć?
- Na czym polega kreatywność przy wymyślaniu nowych słów?
- Co to jest system produkcyjny? Z czego się składa i jak działa?
- Jakie są zalety i wady systemów produkcyjnych?
- Podać przykład ramy dla pojęcia XXX.
- Rodzaje agentów.
- Czy agent typu X wystarczy by zrobić Y?
- Co mogą zrobić duże modele językowe jako podstawa dla agentów.
- Jakie są możliwości systemów wieloagentowych?