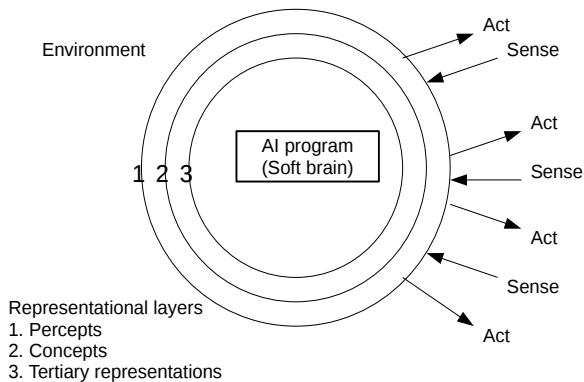


Artificial agent



Artificial agent schematic

Dimensions of variation of task environment

- ▶ Fully observable env. vs partially observable env.
- ▶ Deterministic vs stochastic ($((s, a) \rightarrow s_1$, s_1 completely determined by s and a - except for other agents). Called strategic - if deterministic except for other agents.
- ▶ Episodic (indp. sense-act chunks) vs sequential (curr. action affects future)
- ▶ Dynamic (env. can change - even durin sense-act cycle) vs static. Called semi-dynamic if environment does not change with time but agents performance does.
- ▶ Discrete vs continuous - nature of time applied to state, env., actions.
- ▶ Single agent vs multiple agents.

Examples: Dimensions of variation

Task Env	Obs(F/P)	Det.(D/S)	Epis(E/S)	Static(S/D)	Disc.(D/C)	No. of Ags(1/M)
Crossword	F	D	S	S	D	1
Tournament chess	F	Str.	S	Semi	D	M
Self driving car	P	S	S	D	C	M
Med. diagnosis	P	S	S	D	C	1
Part picking	P	S	E	D	C	1
Image analysis	F	D	E	Semi	D	1
Plant controller	P	S	S	D	C	1
Tutor	P	S	S	D	D	M
Chem. struc.	P	S	S	S	D	1

AI is agent design

AI: science of agent design in different task environments.

Formally, construct function $f : S \times P \times K \rightarrow A$, where S is set of states, P is set of percepts, K is the knowledge base (history), A set of actions.

f is the *findAction* function which takes as argument: current state, percepts (from sensing), knowledge (past history, other information) and outputs an action.

Agent types

Agents can be classified into some broad agent types.

- ▶ Reactive or reflex agents. Modelled by table driven representations.
- ▶ Rule driven agents. Modelled by situation-action rules.
- ▶ Model driven agents. Constructs models of env. and effect of actions on env. and autonomous changes in env.
- ▶ Goal driven agents. Have models and also goals. Typically, one goal.
- ▶ Utility based agents. Generalization of goal driven agents. Optimize utility in the presence of multiple conflicting goals using trade-offs.
- ▶ Learning agents. f can keep changing based on changing dependencies. Even when dependencies are assumed constant they are not known a priori so must be learnt from data.