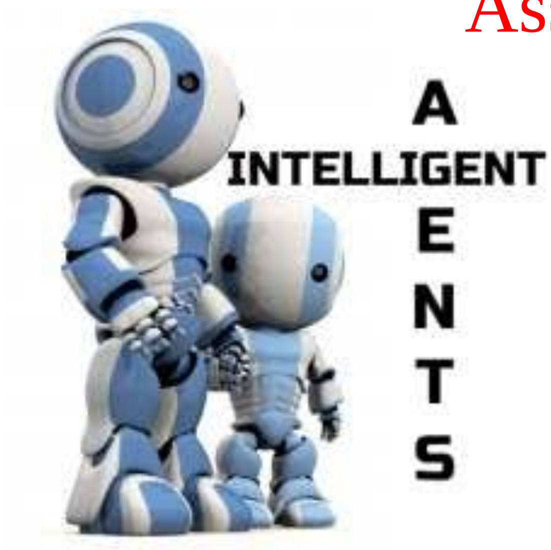
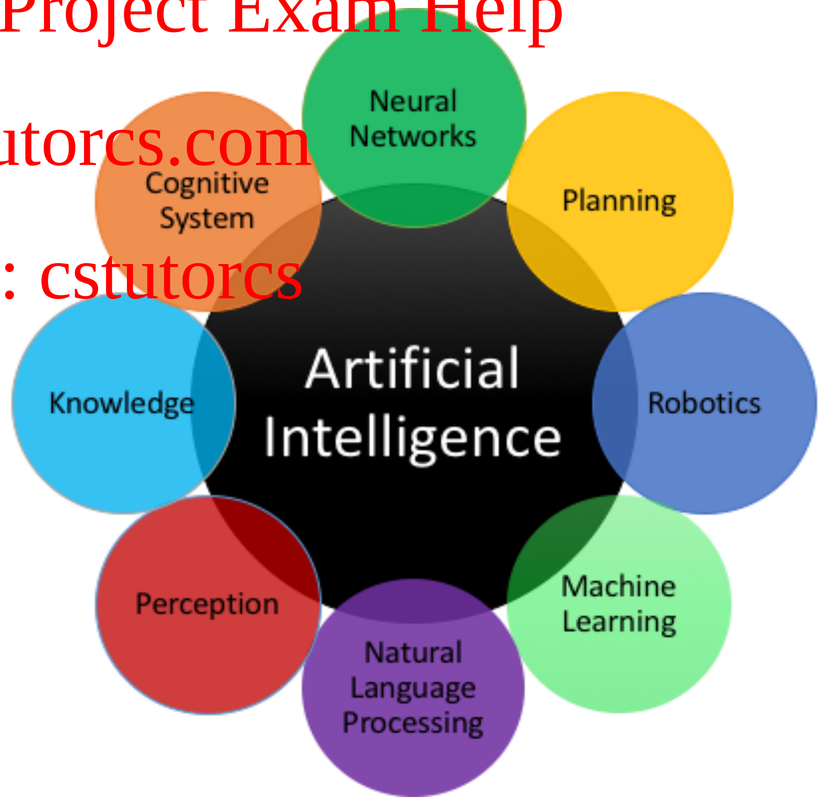
Introduction to AI: Part II

Alessandra Russo, Mark Law

{a.russo, mark.law}@imperial.ac.uk

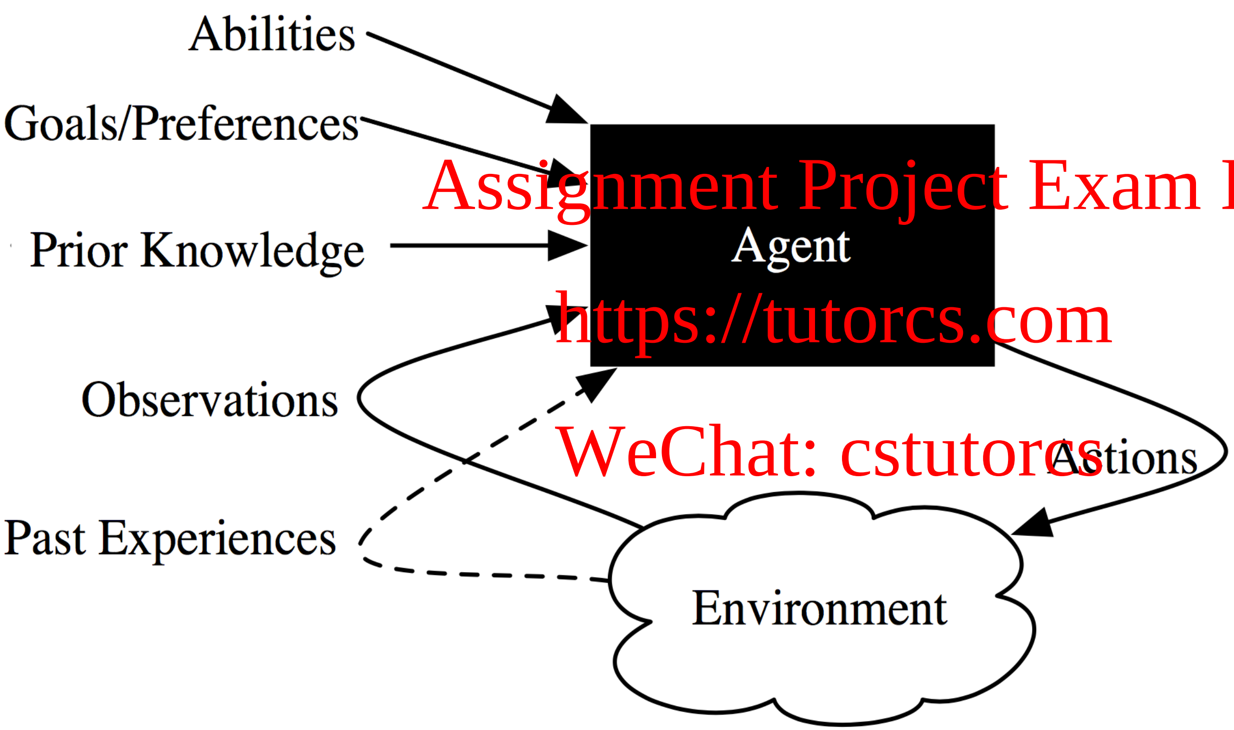




A brief history

20th century	Understanding computation. Several models of computation (e.g. Turing machine)
1950	First applications of computers were AI programs ➢ Program that learns to play checkers ➢ Logic Theorist that discovers proofs in propositional logic ➢ Perceptron (first work on formal neurons), by Rosenblatt
1970 - 80	Complex knowledge representations (McCharty and Hayes) ➢ How to represent the knowledge needed to solve a problem. ➢ Chat-80, a Q&A system to answer geographical questions.
1970 - 88	Domain specific expert systems ➢ Formal language for AI reasoning (Prolog)
1990 - 2000	Sub-disciplines of AI (e.g. perception, probabilistic and decision-theoretic, reasoning, planning).
2000 -	Machine learning, vision, robotics,…

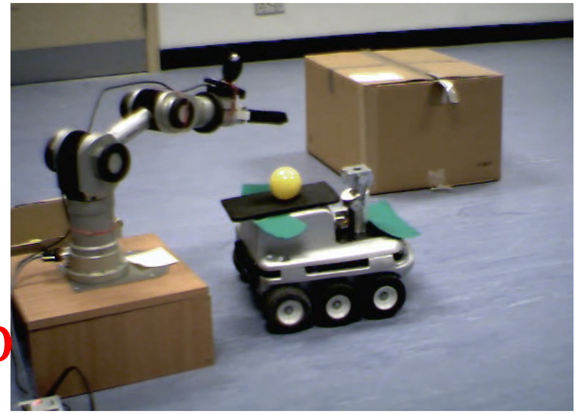
Intelligent Agent

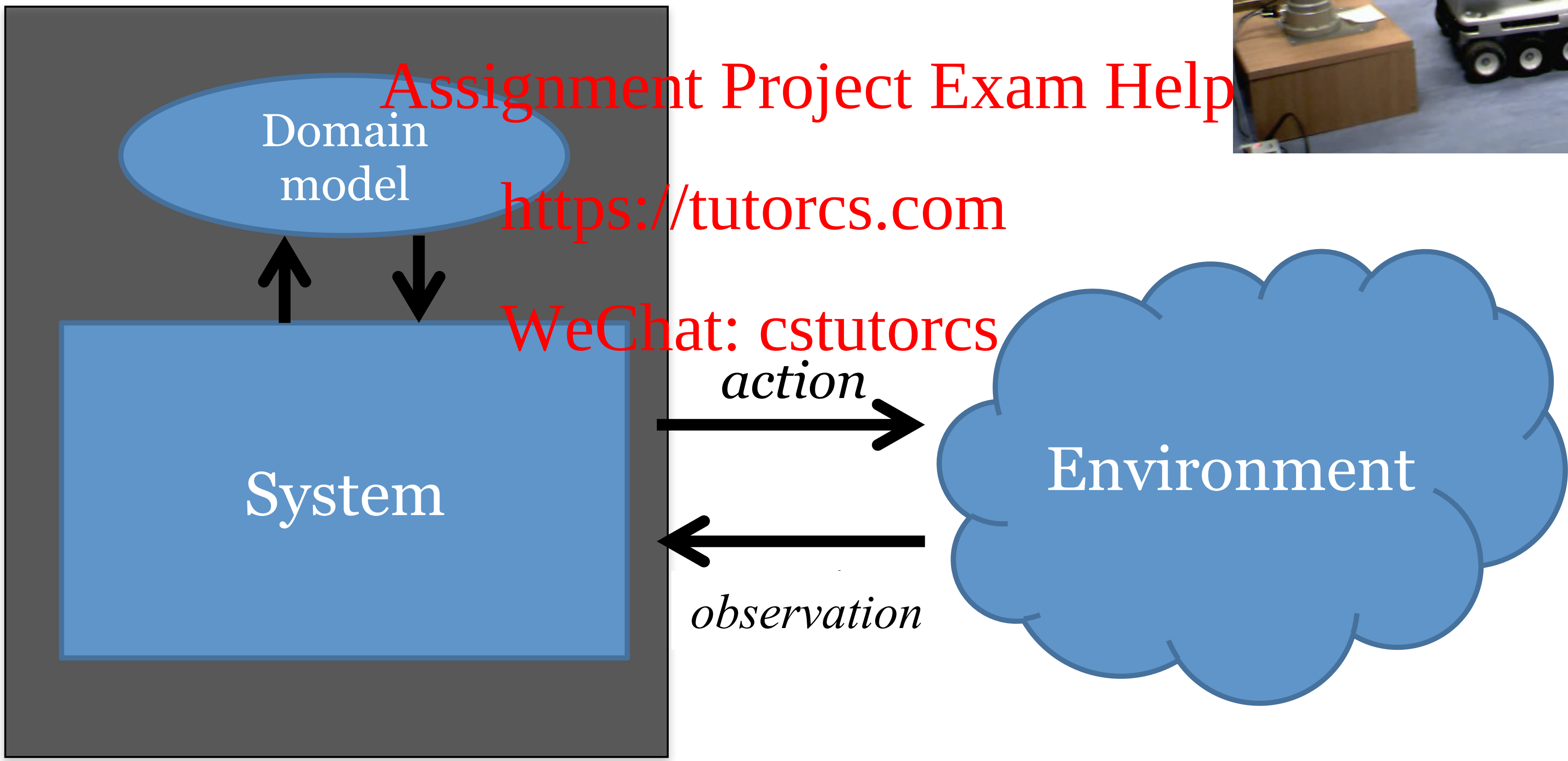


- Robot agents
- Digital assistant
- Software agent
- …..

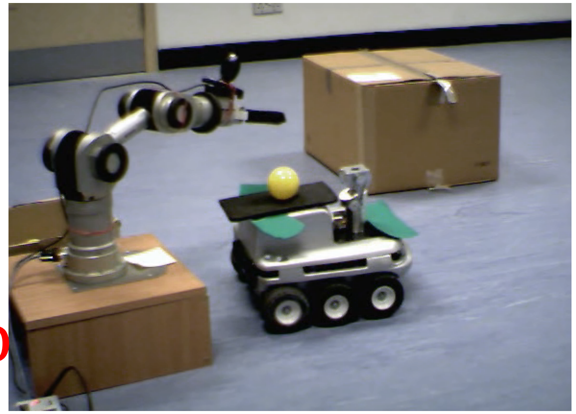
A Robot Agent

Reasoning and Planning

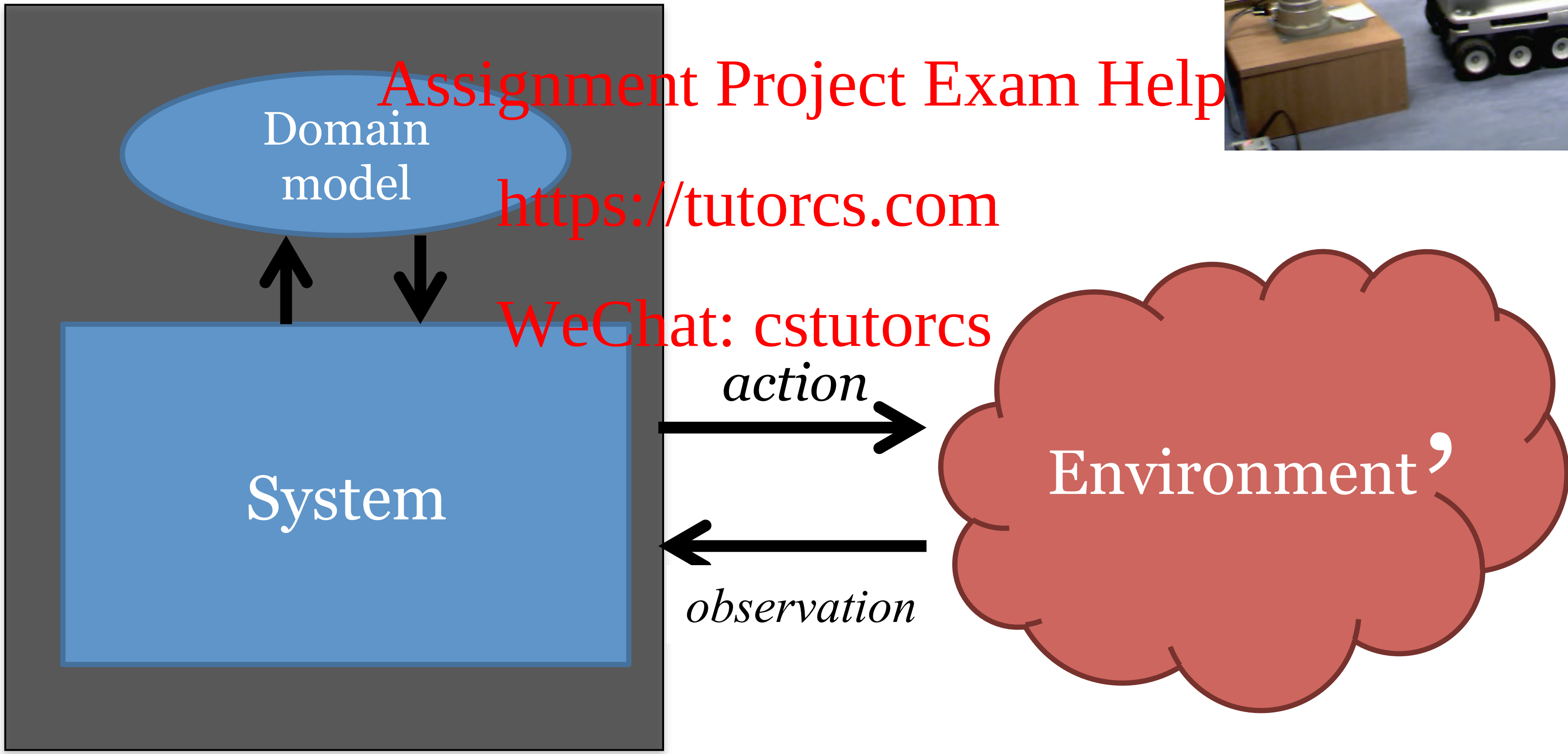




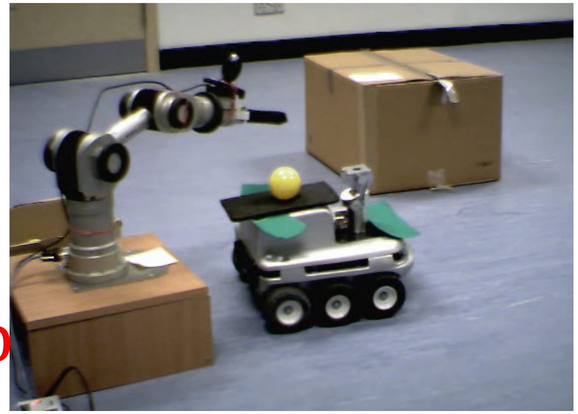
A Robot Agent



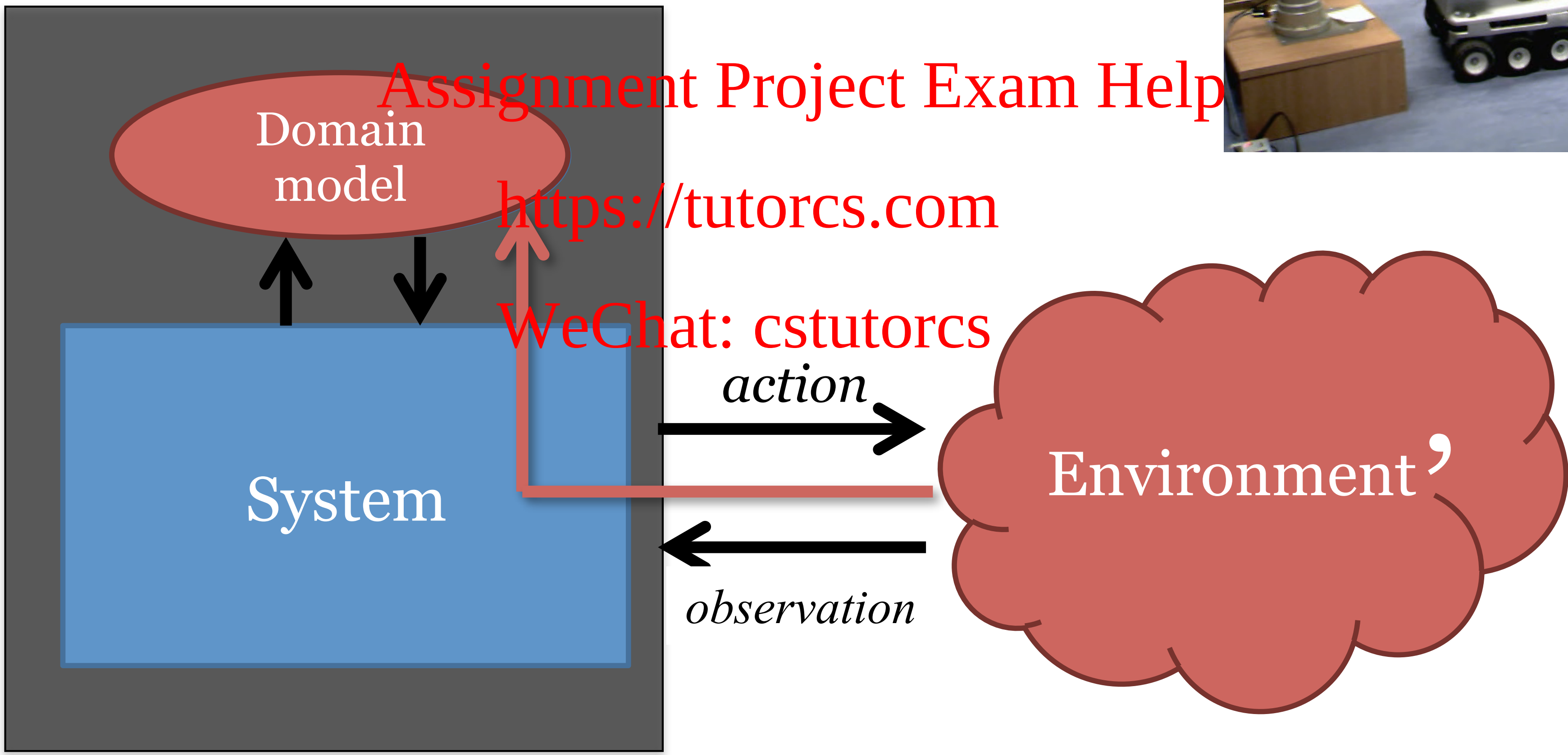
Exogenous Events



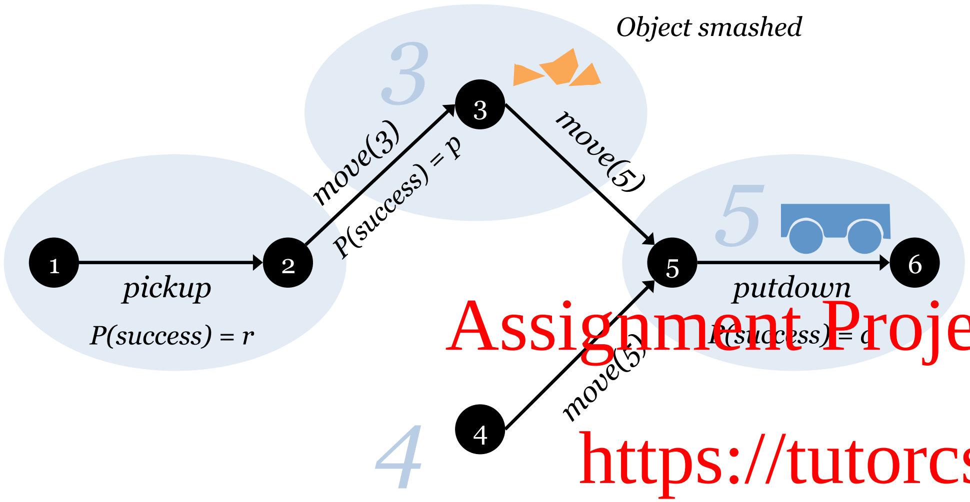
A Robot Agent



Learning new knowledge



A Robot Agent



Past experience

Execution traces

```
holdsAt(at(loc1), 0).
do(pickup, 0).

holdsAt(at(loc1), 1).
holdsAt(holdingObj, 1).
do(move(loc1, loc3), 1).

holdsAt(at(loc3), 2).
holdsAt(holdingObj, 2).
do(move(loc3, loc5), 2).

holdsAt(at(loc5), 3).
holdsAt(holdingObj, 3).
do(putdown, 3).
```

Prior Knowledge

Domain model

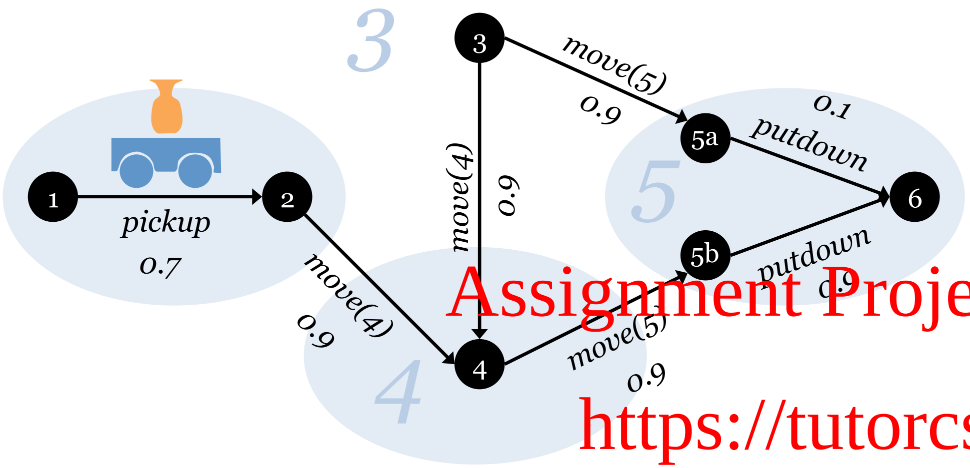
```
possible(pickup, T) :-
        not holdsAt(holdingObject, T),
        holdsAt(at(loc1), T).
possible(putdown, T) :-
        holdsAt(holdingObject, T),
        holdsAt(at(loc5), T).
possible(move(L1, L2), T) :-
        holdsAt(at(L1), T),
        connected(L1, L2).
...
initiates(pickup, holdingObject, T).
terminates(putdown, holdingObject, T).
initiates(move(L1, L2), at(L2), T).
terminates(move(L1, L2), at(L1), T).
```

+

Learned knowledge

```
r1:0.7 : succeeds(pickup, T).
r2:0.9 : succeeds(move(L1, L2), T) :-
               holdsAt(at(L1), T),
               connected(L1, L2),
               L2 != loc3.
r3:0.9 : succeeds(putdown, T) :-
               not happened(move(loc2, loc3), T-2).
r4:0.1 : succeeds(putdown, T) :-
               happened(move(loc2, loc3), T-2).
```

A Robot Agent



Past experience

Execution traces

```
holdsAt(at(loc1), 0).
do(pickup, 0).

holdsAt(at(loc1), 1).
holdsAt(holdingObj, 1).
do(move(loc1, loc3), 1).

holdsAt(at(loc3), 2).
holdsAt(holdingObj, 2).
do(move(loc3, loc5), 2).

holdsAt(at(loc5), 3).
holdsAt(holdingObj, 3).
do(putdown, 3).
```

Prior Knowledge

Domain model

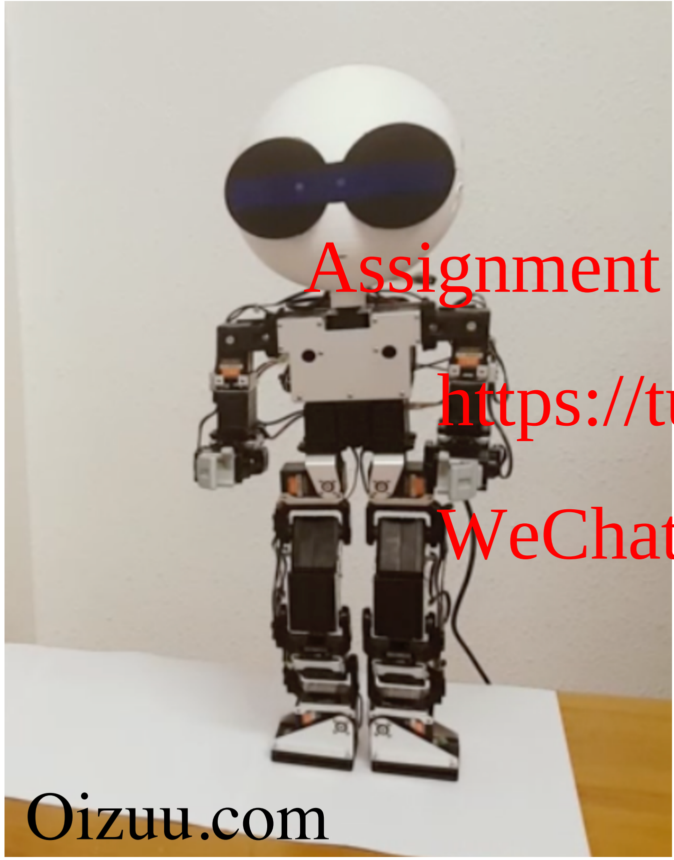
```
possible(pickup, T) :-
        not holdsAt(holdingObject, T),
        holdsAt(at(loc1), T).
possible(putdown, T) :-
        holdsAt(holdingObject, T),
        holdsAt(at(loc5), T).
possible(move(L1, L2), T) :-
        holdsAt(at(L1), T),
        connected(L1, L2).
...
initiates(pickup, holdingObject, T).
terminates(putdown, holdingObject, T).
initiates(move(L1, L2), at(L2), T).
terminates(move(L1, L2), at(L1), T).
```

+

Learned knowledge

```
r1:0.7 : succeeds(pickup, T).
r2:0.9 : succeeds(move(L1, L2), T) :-
              holdsAt(at(L1), T),
              connected(L1, L2),
              L2 != loc3.
r3:0.9 : succeeds(putdown, T) :-
              not happened(move(loc2, loc3), T-2).
r4:0.1 : succeeds(putdown, T) :-
              happened(move(loc2, loc3), T-2).
```

Reasoning and learning through human-robot dialogue



Machine learning has recently been able to support highly accurate NLP

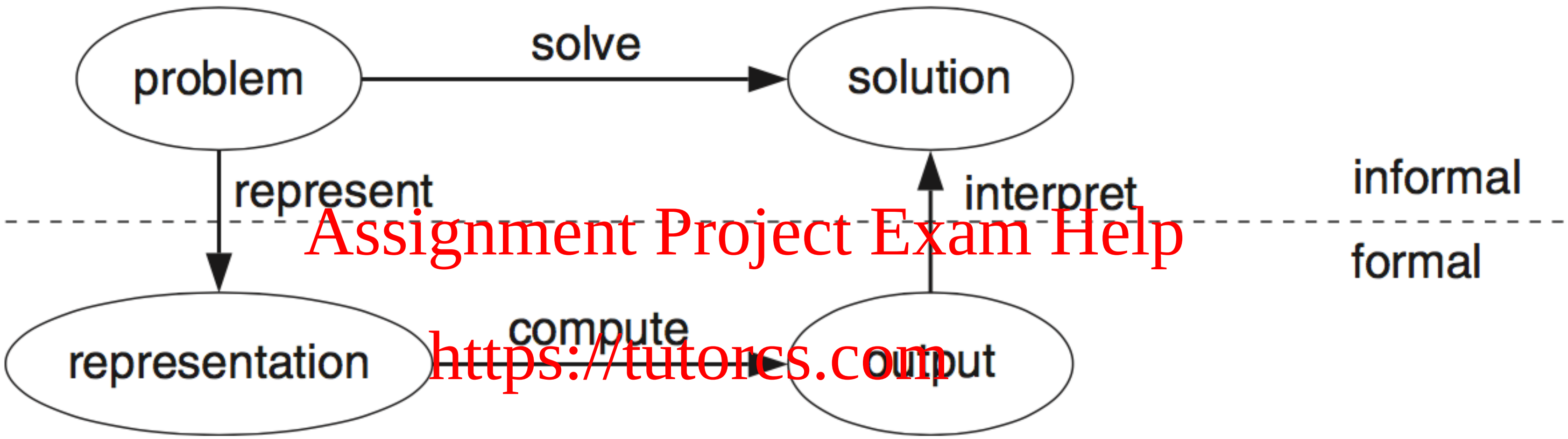
- SyntaxNet (from Google)
- Core-NLP (from Stanford)

But, limited in extracting common-sense and domain expert knowledge,

Symbolic reasoning and symbolic learning support deeper semantic understanding

https://1drv.ms/v/s!Aq-g0J2JpSjPox7CC5YSCvXLYNgl

Representation in problem solving



- Representation schema: form of knowledge used in an agent
- Representation: internal representation of knowledge
- Knowledge base: representation of all the knowledge that is stored in an agent

How should a representation be?

We are interested in representations that are:

- Expressive enough to captures knowledge needed to solve a problem.

- Close to the problem that need to be solve: declarative, compact and easy to maintain.

- Amenable to efficient computations, and able to trade off accuracy and computation time.
- Can be automatically acquired from people, past experience and data, i.e. learnable!

What should a solution be?

Given an informal description of a problem, what is a solution?

Typically four classes of solutions:

- Optimal solutions: robot travelling minimal distance

- Satisficing solution: good enough to deliver some items

- Approximately optimal solutions: robot travelling distance that is close enough to the optimal distance.
- Probable solutions: something that is likely to be a solution

From problem to representation

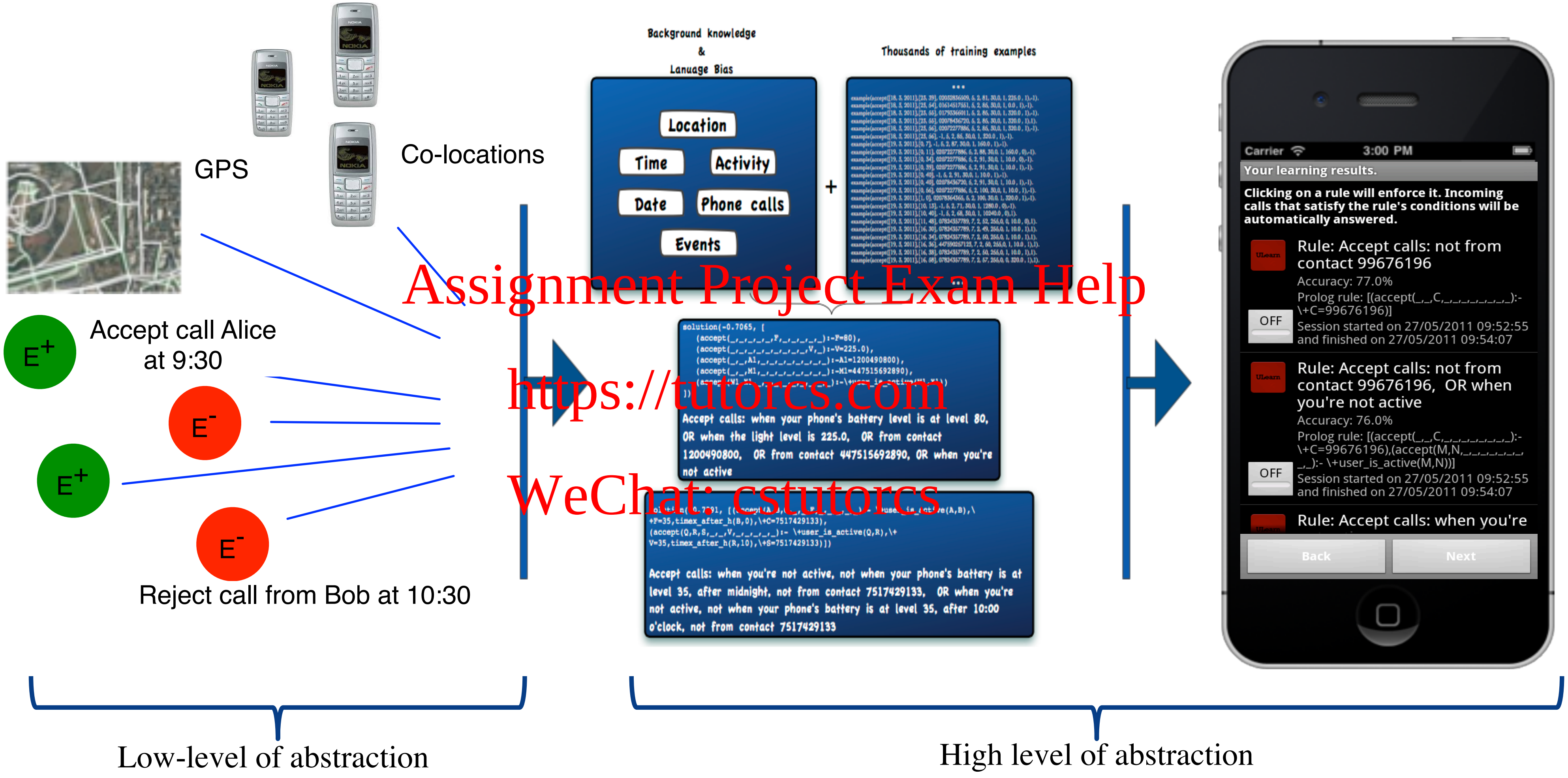
Given the type of solutions we want to compute, how do we represent the problem?

- What level of abstraction of the problem to represent?

- What individuals and relations in the world we need to represent?

- How can an agent represent the knowledge?
- How can an agent acquire the information from data, sensing, experience, or other agents?

Choosing level of abstraction



Model the problem with multiple levels of abstraction.

Reasoning

Reasoning, process by which an agent manipulates information to search through the space of possibilities to determine how to complete its task.

- Offline computation: done by the agent before it has to act. It uses background knowledge and data.

- Online computation: done by the agent between observing the environment and acting in the environment. It uses both background knowledge and observations to decide what to do.

Three forms of reasoning:

Deductive **Abductive** **Inductive**

Different levels of complexity

Models of the environment:

- States
- Features
- Relational descriptions: individuals/objects and relations

Uncertainty:

- Sensing uncertainty
- Effect uncertainty

Preferences:

- Trade-off between the desirability of various outcomes. Ordinal and cardinal preferences.

Number of agents:

- Single agent
- Multiple agents (adversarial versus cooperative agents).

Objectives

- Modelling a problem
  - Different representations and semantics (e.g. (non-)monotonicity and constraints)
  - Different forms of reasoning . (e.g. deductive and abductive inference)
  - Abductive reasoning

- Answer Set Programming
  - Language and semantics (e.g. (non-)determinism, preferences)
  - Bottom-up reasoning

- Some Typical AI Problems
  - Planning (e.g. abductive planning, Sat-Planning, ASP)
  - Diagnosis
  - Problem solving in ASP.
- Sat Solving
  - How to define a SAT problem
  - Algorithms
  - Applications

Reading Material

- *Prolog Programming for Artificial Intelligence,* Ivan Bratko Pearson 2012.
- *Artificial Intelligence, Foundations of Computational Agents,* David Poole and Alan Mackworth

- *Answer Set Solver, Clingo.* https://potassco.org

- *Knowledge Representation, Reasoning, and the Design of Intelligent Agents* – Michael Gelfond & Yulia Gelfond Kahl Januray 2014
- Some research papers

Slides and notes, complemented with information given during lectures and tutorials.

Main Topics

Keys aspects of AI and computational agents

- Deductive, Abductive and Inductive Reasoning
- Abductive Reasoning
  - Top-down approach
    - » Algorithm and use of integrity constraints
    - » Semantics, soundness and completeness properties
    - » Reasoning about events and Goal-directed planning
  - Bottom-up approach
    - » Weak constraints and notion of best explanation
    - » Semantics, soundness and completeness properties
- Answer Set Programming and Stable Model Semantics
  - » Language, Syntax and Semantics
  - » Non-deterministic rules, and optimisation statements
- SAT Solving