COMP3211 Tutorial 2: Simple Agents

Fengming ZHU

Feb. 19&22, 2024

Department of CSE
HKUST

Outline

Overview

Production System

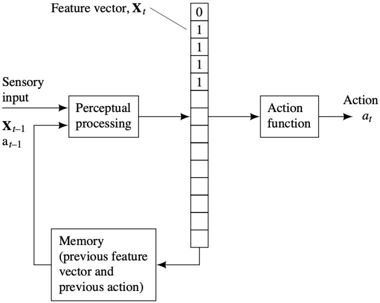
- Boundary-Following Agents
- Capabilities and Limitations

Genetic Programming

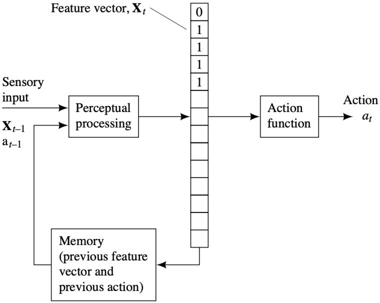
- Biological description
- Application in optimization

Overview

Simple Agents



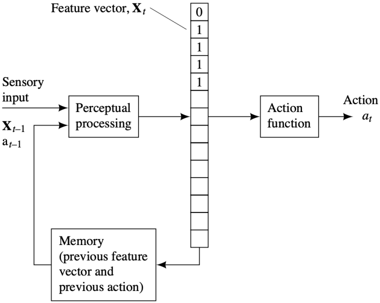
Simple Agents



In terms of memory:

- Stimulus-response (reactive) agents,
- State machines,

Simple Agents



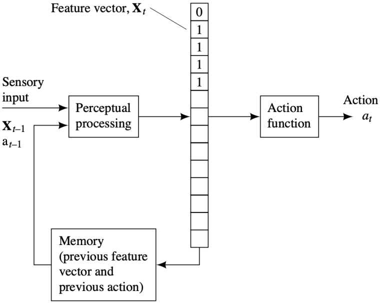
In terms of memory:

- Stimulus-response (reactive) agents,
- State machines,

In terms of action function:

- TLUs,
- Production systems

Simple Agents



In terms of memory:

- Stimulus-response (reactive) agents,
- State machines,

In terms of action function:

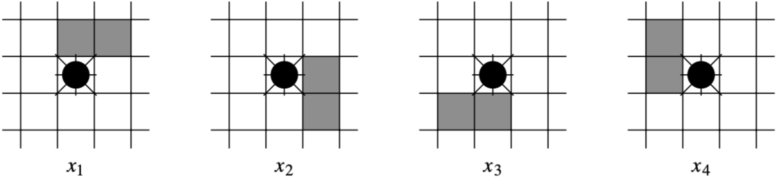
- TLUs,
- Production systems

In terms of optimization:

- Error correction,
- Genetic programming

Production System

Boundary-Following Agents



$x_4\overline{x_1} \rightarrow north,$

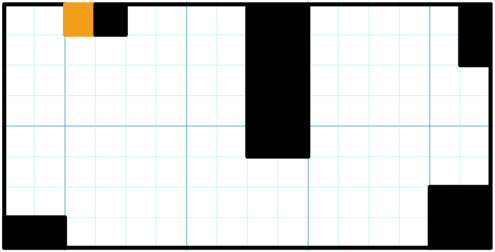
$x_3\overline{x_4} \rightarrow west,$

$x_2\overline{x_3} \rightarrow south,$

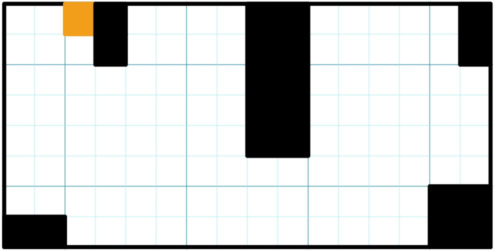
$x_1\overline{x_2} \rightarrow east,$

$1 \rightarrow north.$

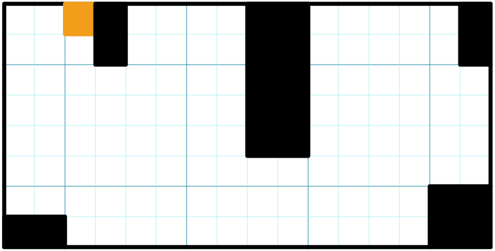
Example 1:
Can you find a production system by which the agent can reach the goal from any initial position?



Example 2:
Can you find a production system by which the agent can reach the goal from any initial position?



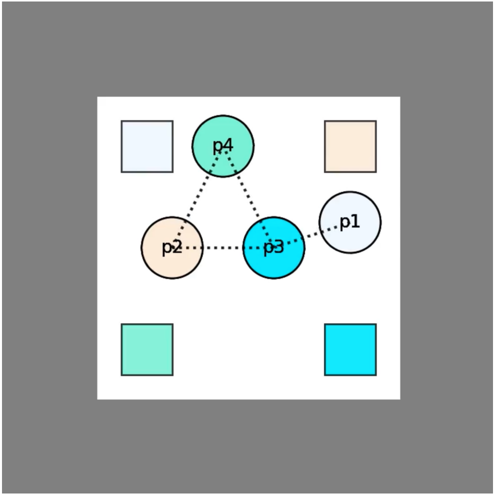
Example 2:
Can you find a production system by which the agent can reach the goal from any initial position?



How many past sensory readings should the agent remember?

Excercise

How about multi-agents:



Genetic Programming

Biological description

Genetic process:

- Large enough population,

Biological description

Genetic process:

- Large enough population,
- Survival of the fittest,

Biological description

Genetic process:

- Large enough population,
- Survival of the fittest,
- Copy and Crossover,

Biological description

Genetic process:

- Large enough population,
- Survival of the fittest,
- Copy and Crossover,
- Mutation

Application in optimization

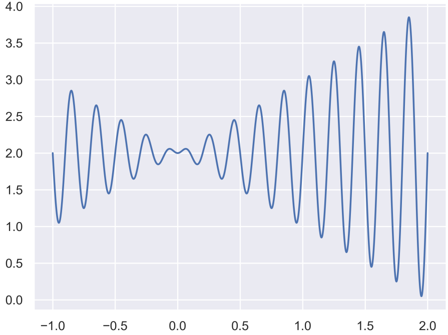
Example 3:
Find nice optima in the interval $[-1, 2]$ for the following function:

- $f(x) = x\sin(10\pi x) + 2$

Application in optimization

Example 3:
Find nice optima in the interval $[-1, 2]$ for the following function:

- $f(x) = x \sin(10\pi x) + 2$
- Quite complicated...



Example

Find nice optima in the interval $[-1, 2]$ for the following function:

- $f(x) = x\sin(10\pi x) + 2$

Example

Find nice optima in the interval $[-1, 2]$ for the following function:

- $f(x) = x\sin(10\pi x) + 2$

By genetic programming:

- Initialization: randomly generate "lots of"

$$(x, f(x)), \text{representing (individual, fitness)}$$

Example

Find nice optima in the interval $[-1, 2]$ for the following function:

- $f(x) = x \sin(10\pi x) + 2$

By genetic programming:

- Initialization: randomly generate "lots of"

$$(x, f(x)), \text{representing (individual, fitness)}$$

- Repeat for enough rounds:

Example

Find nice optima in the interval $[-1, 2]$ for the following function:

- $f(x) = x\sin(10\pi x) + 2$

By genetic programming:

- Initialization: randomly generate "lots of"

$$(x, f(x)), \text{representing (individual, fitness)}$$

- Repeat for enough rounds:
  - Crossover: $x_{child} \leftarrow \lambda x_{father} + (1-\lambda)x_{mother}$

Example

Find nice optima in the interval $[-1, 2]$ for the following function:

- $f(x) = x \sin(10\pi x) + 2$

By genetic programming:

- Initialization: randomly generate "lots of"

$$(x, f(x)), \text{representing (individual, fitness)}$$

- Repeat for enough rounds:
  - Crossover: $x_{child} \leftarrow \lambda x_{father} + (1 - \lambda) x_{mother}$
  - Mutation: $x \leftarrow \texttt{Normal}(x, 0.5)$, for 10% of individuals (x's)

Example

Find nice optima in the interval $[-1, 2]$ for the following function:

- $f(x) = x \sin(10\pi x) + 2$

By genetic programming:

- Initialization: randomly generate "lots of"

$$(x, f(x)), \text{representing (individual, fitness)}$$

- Repeat for enough rounds:
  - Crossover: $x_{child} \leftarrow \lambda x_{father} + (1 - \lambda) x_{mother}$
  - Mutation: $x \leftarrow \texttt{Normal}(x, 0.5)$, for 10% of individuals (x's)
  - Selection: choose top 2/3 fittest individuals (x's)

Example

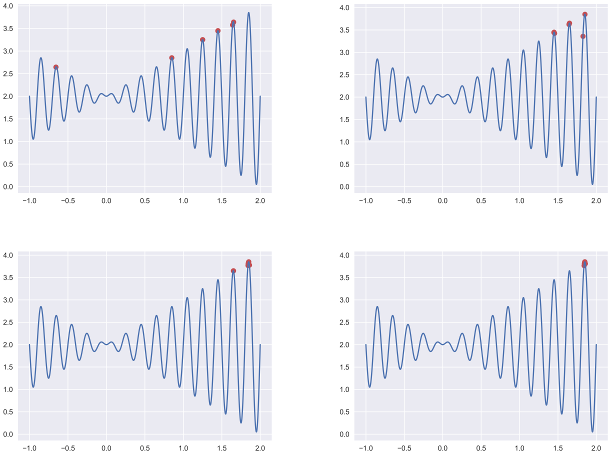


Figure 1: Larger population and more generations

Compared to GD

In terms of optimization, why we still need gradient descent to train a TLU/neural network?

- Genetic programming: zero-order information
- GD (error-correction as a special case): first-order derivatives
- Newton's method: second order derivatives (computing inverse of Hessian matrix is hard)

Thanks!