Introduction to intelligent systems

Symbolic AI

Mikkel N. Schmidt

Technical University of Denmark,
DTU Compute, Department of Applied Mathematics and Computer Science.

Overview

1. Logic
2. Expert systems
3. Search
4. Demo: Lunar Lander and 2048
5. Tasks

Feedback group

- Lucas Rieneck Gottfried Pedersen
- Benjamin Banks
- Tobias Emde Ralsted Jensen
- Solvej Amélie Brun Sønderbæk

Learning objectives

I Symbolic AI.

I Forward and backward chaining.

I Monte carlo search.

II Boolean logic.

II Rule-based systems and expert systems.

I Understand the concepts and definitions, and know their application. Reason about the concepts in the context of an example. Use correct technical terminology.

II As above plus: Read, manipulate, and work with technical definitions and expressions (mathematical and Python code). Carry out practical computations. Interpret and evaluate results.

Logic

Boolean algebra notation

- Variables

True	$x = T$	$x = 1$
False	$x = F$	$x = 0$

- Operators

and	$x \wedge y$	$x \cdot y$
or	$x \vee y$	$x + y$
not	$\neg x$	$\overline{x}$

Boolean algebra

Commutative law	$a + b = b + a$
	$a \cdot b = b \cdot a$
Associative law	$a + (b + c) = (a + b) + c$
	$a \cdot (b \cdot c) = (a \cdot b) \cdot c$
Distributive law	$(a \cdot b) + (a \cdot c) = a \cdot (b + c)$
	$(a + b) \cdot (a + c) = a + (b \cdot c)$
Double negative law	$\overline{\overline{a}} = a$
Identities	$a + 0 = a, \qquad a + 1 = 1$
	$a \cdot 1 = a, \qquad a \cdot 0 = 0$
Complement law	$a + \overline{a} = 1$
	$a \cdot \overline{a} = 0$
Absorbtion laws	$a + a \cdot b = a$
	$a + \overline{a} \cdot b = a + b$
DeMorgan's law	$\overline{a \cdot b} = \overline{a} + \overline{b}$
	$\overline{a + b} = \overline{a} \cdot \overline{b}$

Boolean function

$$f(x_1, x_2, \ldots, x_n)$$

- Inputs: n Boolean values
- Output: A Boolean value

Will I eat chocolate?

The following Boolean function governs when I eat chocolate:

$$f(a, b, c) = \overline{a} \cdot b + b \cdot \overline{c} + b \cdot c + a \cdot \overline{b} \cdot \overline{c}$$

f=I'll eat chocolate a=I'm hungry b=I'm tired c=I have proper food

Question 1 (easy) Will I eat chocolate if im hungry, not tired, and have proper food? I.e., what is the value of $f(1, 0, 1)$?

Question 2 (difficult) Simplify the expression for $f(a, b, c)$ as much as possible.

Will I eat chocolate?

The following Boolean function governs when I eat chocolate:

$$f(a, b, c) = \overline{a} \cdot b + b \cdot \overline{c} + b \cdot c + a \cdot \overline{b} \cdot \overline{c}$$

f=I'll eat chocolate a=I'm hungry b=I'm tired c=I have proper food

Question 1 (easy) Will I eat chocolate if im hungry, not tired, and have proper food? I.e., what is the value of $f(1, 0, 1)$?

Question 2 (difficult) Simplify the expression for $f(a, b, c)$ as much as possible.

Solution

1. $f(1, 0, 1) = \overline{1} \cdot 0 + 0 \cdot \overline{1} + 0 \cdot 1 + 1 \cdot \overline{0} \cdot \overline{1} = 0 \cdot 0 + 0 \cdot 0 + 0 \cdot 1 + 1 \cdot 1 \cdot 0 = 0$
2. $f(a, b, c) = b \cdot (\overline{a} + \overline{c} + c) + a \cdot \overline{b} \cdot \overline{c} = b + a \cdot \overline{b} \cdot \overline{c} = b + a \cdot \overline{c}$

Expert systems

Expert systems

- A collection of facts and rules

  Facts Observable / inferred variables

  Rules If-statements

  if <condition>

  then <consequent>

- Rules are created by experts

Example: Will I eat chocolate?

if tired
then eat chocolate

if hungry and not have proper food
then eat chocolate

if have proper food
then dont eat chocolate

if tired and not hungry
then dont eat chocolate

Conflicts and undefined cases

Constructing rule that cover all cases without conflict is difficult

Conflict
- Two or more rules fire with conflicting consequents
- Solution: Prioritize rules

Undefined cases
- No rules fire to generate the consequent we are interested in
- Solution: Add more rules or define defaults

Conflicts and undefined cases

Can you identify a conflict and an undefined case in the example?

if tired
then eat chocolate

if hungry and not have proper food
then eat chocolate

if have proper food
then dont eat chocolate

if tired and not hungry
then dont eat chocolate

Conflicts and undefined cases

Can you identify a conflict and an undefined case in the example?

if tired
then eat chocolate

if hungry and not have proper food
then eat chocolate

if have proper food
then dont eat chocolate

if tired and not hungry
then dont eat chocolate

Solution

- Conflict: Rule 1 and 4 fire if *tired* and not *hungry*.
- Undefined: No rule fires if not *tired*, not *hungry*, and not *have proper food*.

Forward and backward chaining

Forward chaining Reasoning from the data

1. Apply the first rule that fires
2. The fired rule generates new facts, influencing which rules may fire
3. Repeat from 1, making sure each rule fires only once

Backward chaining Reasoning to prove an outcome

1. Find the rule(s) that could generate the fact we want to prove
2. Check if the conditions of the rule(s) are true
3. If unknown, find the rule(s) that could generate the facts needed for the condition
4. Continue until the fact is proven or we have failed our attempt

Forward and backward chaining

Consider the example on the right (it has no conflicts)

if *happy*=True or *birthday*=True
then *sing*=True

if *birthday*=True and *diet*=False
then *cake*=True

if *cake*=True and (*sing*=True or *diet*=True)
then *problem*=True

1. What can you infer from the fact *birthday*=True?
   (Use forward chaining)
2. Can you prove that *problem*=True from *sing*=True and *diet*=False?
   (Use backward chaining)

Forward and backward chaining

Consider the example on the right (it has no conflicts)

if *happy*=True or *birthday*=True
then *sing*=True

if *birthday*=True and *diet*=False
then *cake*=True

if *cake*=True and (*sing*=True or *diet*=True)
then *problem*=True

1. What can you infer from the fact *birthday*=True? (Use forward chaining)
2. Can you prove that *problem*=True from *sing*=True and *diet*=False? (Use backward chaining)

Solution

1. *sing*=True follows from rule 1. Nothing more can be inferred
2. No it cannot be proven. Only rule 3 could generate *problem*=True but we need *cake*=True. This could only be generated from rule 2 we then need *birthday*=True which we do not have.

Search

AI by search

Search as an AI strategy

- Observables: State of the environment
- Action: Change environment state
- Goal: Take actions to reach a certain desirable state

Monte Carlo search

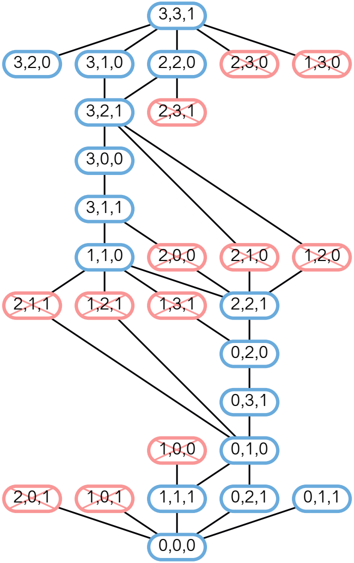
Exact search is often not feasible

- Large state space: Search all paths not possible
- Approximate search: Find next step that likely leads to a solution
- Requires measure of reward to rank states/partial paths

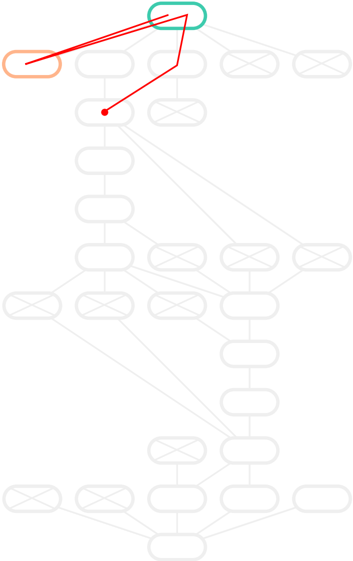
Monte Carlo search

1. Explore N random paths from current state
2. Score the explored paths by their reward
3. Take the first step that leads to the highest average score
4. Repeat

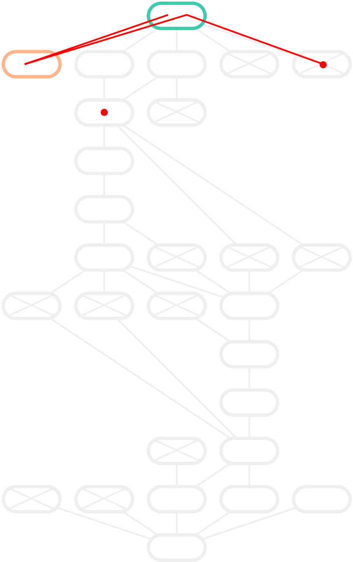
Monte Carlo search



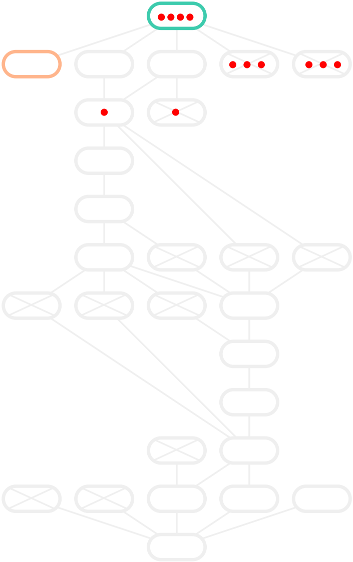
Monte Carlo search



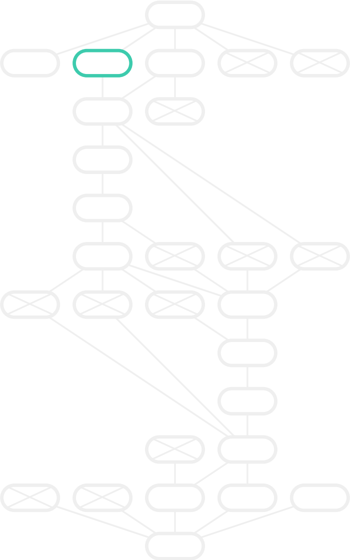
Monte Carlo search



Monte Carlo search



Monte Carlo search



Demo: Lunar Lander and 2048

Tasks

Tasks for today

Tasks today

- Start working on Lab Report 2: Read description on DTU Learn
- Today's feedback group
  - Lucas Rieneck Gottfried Pedersen
  - Benjamin Banks
  - Tobias Emde Ralsted Jensen
  - Solvej Amélie Brun Sønderbæk

Lab report hand in

- Lab 2: Symbolic AI (Deadline: Thursday 28 September 20:00)

Next time

- Read the notes "Data representation" + Solve all problems