Introduction

Professor Marie Roch
Chapters 1 & 2, Russell & Norvig

Intelligence per Merriam Webster

1 a : 1) the ability to learn or understand or to deal with new or trying situations: REASON; also : the skilled use of reason
    2) the ability to apply knowledge to manipulate one's environment or to think abstractly as measured by objective criteria (as tests)

b Christian Science : the basic eternal quality of divine Mind

c : mental acuteness: SHREWWDNESS

2 a : an intelligent entity; especially : ANGEL

b : intelligent minds or mind <cosmic intelligence>

3 : the act of understanding: COMPREHENSION
What does it mean for a machine to be intelligent?

Turing Test – Can a human tell that they are interacting with a computer?

Variant of this competition occurs today with Loebner Prize (very restricted Turing test for chatbots)

Artificial intelligence

• Turing test - acting intelligent is sufficient
• Cognitive approach – Model the thinking of humans or other animals
• Laws of thought – Set of logic rules drawback: difficult to model informal/general knowledge
• Rational agent – Logical inference is good, but not always applicable.
When will machines be intelligent?
(Armstrong, et al. 2014)

Eliza (Weizenbaum, 1966)

young woman: Men are all alike.
eliza: In what way?
young woman: They’re always bugging us about something specific or other.
eliza: Can you think of a specific example?
young woman: Well, my boyfriend made me come here.
eliza: Your boyfriend made you come here?
young woman: He says I’m depressed much of the time.
eliza: I am sorry to hear you are depressed.
young woman: It’s true. I’m unhappy.
eliza: Do you think coming here will help you not to be unhappy?
young woman: I need some help, that much seems certain.
eliza: What would it mean to you if you got some help?
young woman: Perhaps I could learn to get along with my mother.
Simple rules ---> disappointment

(X me Y) ---> (X you Y)
(I remember X) ---> (Why do you remember X just now?)
(My {family-member} is Y) ---> (Who else in your family is Y?)
(X {family-member} Y) ---> (Tell me more about your family)

Areas contributing to AI

• Philosophy
• Mathematics
• Neuroscience
• Psychology
• Linguistics
• Computer science

• Many application areas contribute as well (e.g. economics)
Intelligent agents

• Sensors provide perceptual input (percepts) of environment

• Agent makes decisions

• Actions carried out through actuators that may affect the environment

Intelligent agents

• Softbot – Software only agent
  • Available data are percepts
  • Examples: web-based reputation monitoring, game opponent

SHALL WE PLAY A GAME?
Task Environments

In what environment will the agent be operating?

- fully vs. partially observable
  - partially observable → uncertain state

- Rules are
  - known: Outcome (or outcome probabilities) are known
  - unknown – Outcomes must be learned

Task environments

- single- vs. multi- agent

- multiagent issues
  - cooperative vs. competitive
  - communication
  - randomization to prevent predictability
Task environments

• What happens when an agent acts?
  • deterministic – we know next state
  • stochastic
    • nondeterministic factors may influence (stochastic → probabilities) leading to an uncertain state

• Decisions are
  • episodic – Next decision only depends on state
  • sequential – Next decision dependent on previous ones

Task environments

• State can be
  • static – does not change while agent is deciding next action
  • dynamic – Environment constantly changing
  • semidynamic – Environment static, but performance is time dependent

See p. 45 Figure 2.6 for example task environments
Quick and dirty Python 3.x

• About the language
  • Interpreted high level language
  • Reasonably simple to learn
  • Rich set of libraries

• For details, see texts in syllabus or
  www.learnpython.org or www.diveintopython.net

• Python comment
  # comment from hash character to end of line

Python data types

• Numbers: 42.8 or 9

• Strings: single or double quote delimited
  ‘hi there’ “Four score and seven years ago…”

• Dictionaries: Python’s hash table
  quotes = dict() # new dictionary
  quotes[“Lincoln”] = “Four score and seven years ago…”
  OR
  quotes = {“Lincoln”: “Four…”,
            “Roosevelt”: “The only thing we have to fear…”}
Python data types

- Sequences
  - Lists ['“Four”, “score”, “and”']
  - tuples (“Four”, “score”, “and”)
- Difference between tuple and list
  - List – can grow or shrink
  - Tuple – Fixed number of elements
    - Faster
    - Can be used as hash table indices

Python data types

- None – special type for null object
- Booleans: True, False
- Variables are untyped
**Python Expressions**

- **assignment:** count = 0
- **list membership:** value in [4, 3, 2, 1]
- **indexing 0 to N-1:** listvar[4], tuplevar[2]
- **slices:**
  - listvar[0:N] → items 0 to N-1
  - listvar[:N] → items 0 to N-1
  - listvar[3:] → items 3 to end
  - listvar[0:2:N] → even items at 0, 2, 4, ...
  - listvar[-4:-1] → 4\(^{th}\) to the last to 2\(^{nd}\) to the last
- **logical operators:** and, or, not

---

**Python expressions**

- **comparison operators:** < > >= <= !=
- **basic math operators:** + - / *
- **exponentation:** x ** 3 # x cubed
- **bitwise operators:** & | ~ and ^ (xor)
Python control structures

• Use indentation to denote blocks
• Conditional execution
  if expression:
    statement(s)
  elif expression:
    statements(s)
  else:
    statement(s)

Python control structure

• Iteration
  done = False
  while not done:
    statements(s)
    done = expression

  for x in range(10):  # 0 to 9
    print(x)
    print("x={}".format(x))

  alter iteration behavior with break and continue (usual semantics)
Python functions

```python
def foobar(formal1, formal2, formal3=None):
    "foobardoesn’t do much"  # doc string
    # Use """" multi-line text """" for long doc strings
    statement(s)
    return value
```

- formal3 defaults to None if not supplied

- Variable scope rules
  local, enclosing function, global, builtin names

Python objects

```python
class Board:
    "Grid board class"
    def __init__(self, rows, cols):  # constructor
        "construct a board with specified rows and cols"
        self.rows = rows
        self.cols = cols
        # list comprehension example
        self.board = [[None for c in range(cols)] for r in range(rows)]
    def place(self, row, col, item):
        "place an item at position row, col"
        self.board[row][col] = item
    def get(self, row, col):
        "get an item from position row, col"
        return self.board[row][col]
```
Python objects

• Create:  \( b = \text{Board}(8,8) \)
• \( b.\text{place}(2, 7, \text{‘black-king’}) \)
• \( b.\text{get}(2,7) \)
  “black-king”

Python

• Integrated development environments
  • Eclipse with PyDev ← what I use
  • Komodo (ActiveState)
  • Pycharm
  • others (see Python.org)

• Versions of Python
  • Python.org – stock Python
  • PyCharm
  • Anaconda – bundles with lots of libraries and and ID
Agent structure

• An agent’s architecture consists of
  • data structures
  • code

• Simplest agent: table driven

```java
function table-driven-agent(percept) returns action
    persistent: percepts (sequence, empty at first)
    table of actions indexed by
    percept sequence

    percepts.append(percept)
    return lookup(percepts, table)
```

Simple-reflex agents

• Ignores percept history, uses the current one
• Productions (aka conditions-action) decide action, e.g.
  - person waving → wave
  - person smiling → smile
  - person swinging hammer towards me → duck!
Model-based reflex agents

• Add internal state
• New percepts update the state
• Productions based on percept and state

Goal-based agents

• Agent works to achieve a specific state
• Usually requires: Search and Planning
Utility-based

• Based on utility theory: The idea of how useful or happy something makes you.
• Decisions are made to maximize the expected utility.

Learning agents

• Decision rules are adjusted based on performance over time
Learning agents

• Learning element uses knowledge from performance element to modify rules

Learning agents

• Critic monitors the environment to see how the system is doing and keeps learning element from going off track.
Learning agents

• Critic monitors the environment
  • Provides utility of the current percepts
  • Allows learning element to learn useful goals

Problem generator

• Uses learning element’s goals to suggest experiments
• Experiments may lead to the learning element improving the performance element