CS4386 AI Game Programming

Lecture 01
Introduction

Semester B, 2020-2021
Department of Computer Science
City University of Hong Kong

Teaching Patterns

- Lecture (2 hours per week)
  - Explain concept and general knowledge of Game AI and game programming
- Tutorial (1 hour per week)
  - Present open-ended questions for encouraging students to exercise critical thinking
  - Provide hand-on practices on game AI and game programming to deepen students understanding on the related subjects

Course Materials

- All materials related to this course will be posted on the Canvas course page https://canvas.cityu.edu.hk/
- It is your own responsibility to check for updated information and announcements on the Canvas course page
- In addition to the Canvas course page, sometimes the instructors may send emails to students for issues related to this course, so check your email regularly and reply promptly in case we ask you to do something

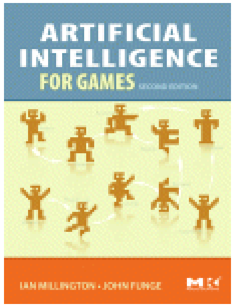
Textbook and Reference

Textbook

- Artificial Intelligence for Games (Second Edition), Ian Millington and John Funge, Morgan Kaufmann, 2009

  E-book free access inside CityU:

  https://www.sciencedirect.com/book/9780123747310/artificial-intelligence-for-games

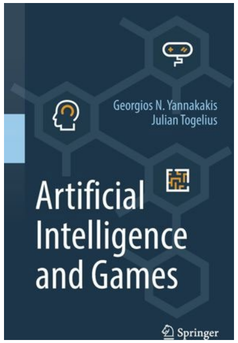

ARTIFICIAL INTELLIGENCE FOR GAMES
SECOND EDITION
IAN MILLINGTON · JOHN FUNGE
MK

Reference

- Artificial Intelligence and Games, Georgios N. Yannakakis and Julian Togelius, Springer, 2018

  E-book free access inside CityU:

  https://link.springer.com/book/10.1007/978-3-319-63519-4



- Online resources

Assessment Patterns

- Course work (50%)
  - Assignments (35%)
  - Quizzes (15%) [to be conducted during tutorials and selected lectures]
- Examination (50%)

- Passing Criteria

  Necessary condition: at least 30% of the maximum mark for the examination must be obtained

Tutorial

- Hands-on programming exercises or practice problems will be given in the weekly 1-hour tutorial session
- You should only attend your registered tutorial session
- You will be asked to complete some online quizzes related to the tutorial, which will be counted in the course work marks.

Requirements

- Basic programming skill
- Ability to debug with patience
- Diligence

Programming Clinic

- Online Zoom consultation sessions from the programming clinic will be arranged every week, from week 4 to week 13 in MMW1411
- Dedicated student helper will stand-by in the laboratory and help those walk-in students to solve problems regarding general programming (Java/C++/C...), web programming (JavaScript/HTML...), data structures as well as fundamentals of computing (digital circuit, binary number...)
- Time table will be available at http://courses.cs.cityu.edu.hk/clinic/

How Can You Become More Effective Learner?

- Take initiative in your learning
- Attend and participate in all classes (both lecture & tutorial)
- Submit assignments and online quizzes on time
- Don't be afraid of asking questions (inside or outside classroom)
- Use online resources as much as possible
- Refrain from academic dishonest behaviour ⚠

How Much Time You Should Spend on This Course?

- According to University Guide, 1 credit unit = 40 to 50 hours (including lecture, tutorial/lab, self study)
- This course has 3 credit units, which translates to 120 to 150 hours per semester or 10 hours per week
- As you spend 3 hours on lecture and tutorial, you should spend about 7 hours self-study and practice

Academic Honesty

http://www.cityu.edu.hk/provost/academic_honesty/rules_on_academic_honesty.htm

- Academic honesty is central to the conduct of academic work. Students are expected to present their own work, give proper acknowledgement of other's work, and honestly report findings obtained
- **Academic dishonesty** is regarded as a **serious academic offence** in the University. Any related offence may lead to **disciplinary action with a penalty including without limitation**, expulsion from the University, debarment from re-admission, deprivation of an academic award already conferred or revocation of a certification granted

Academic Dishonesty

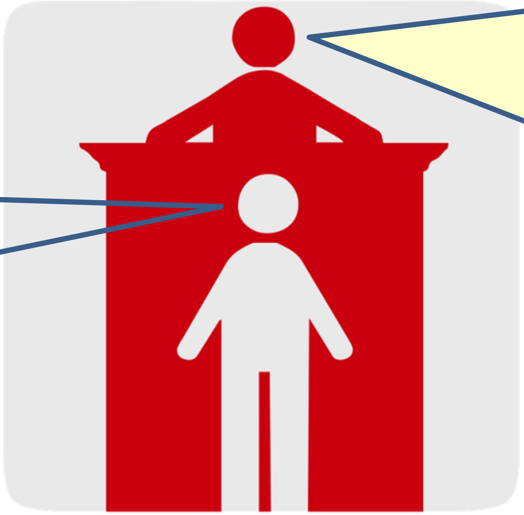
http://www.cityu.edu.hk/provost/academic_honesty/rules_on_academic_honesty.htm

- Academic dishonesty includes but is not restricted to the following behaviors:
  - Plagiarism, e.g., the failure to properly acknowledge the use of another person’s work or submission for assessment material that is not the Student's own work;
  - Misrepresentation of a piece of group work as the Student's own individual work;
  - Collusion, i.e., allowing another person to gain advantage by copying one's work;
  - Unauthorized access to an examination/test paper;
  - Possession/use of unauthorized material in assessment;
  - Unauthorized communication during assessment;
  - Use of fabricated data claimed to be obtained by experimental work, or data copied or obtained by unfair means;
  - Impersonating another Student at a test or an examination or allowing oneself to be impersonated.

Selected Useful Tips to follow Principles of Academic Honesty

http://www6.cityu.edu.hk/ah/useful_tips.htm

- Start early and manage your time (Tip 1)
- Do your own work (Tip 7)
- Do not give a ‘reference’ copy of your work to someone (Tip 8)
- Talk about academic honesty with your classmates (Tip 9)
- Ignorance is no excuse (Tip 11)

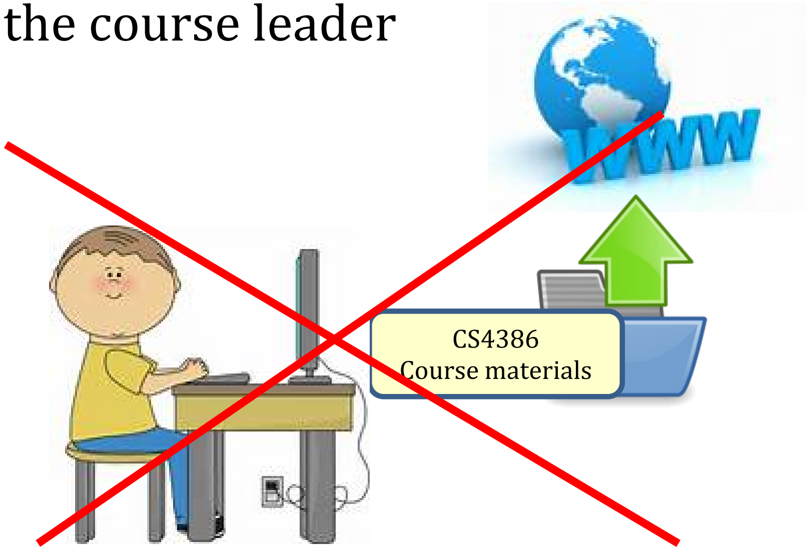


Your Honor, nobody has ever told me what I did was wrong. Also, I have seen others doing the same thing so I should not be guilty!

You are not a child any more and you have the intelligence to enter university so you should have known better what is right and wrong. Just because you see other people doing bad things does not provide justifications for you to do the same. I therefore find you guilty!

DO NOT redistribute course materials

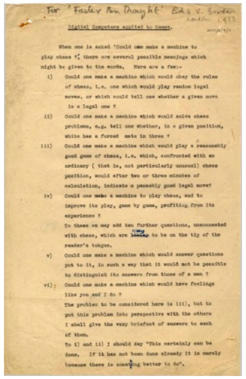
- You do not own the copyright of the course materials even though you can download them from Canvas course page
- So you SHOULD NOT redistribute any of the course materials to anyone else who are not taking this course, e.g., you SHOULD NOT upload any course materials to websites such as Course Hero
- If you are in doubt, first check with the course leader (howard@cityu.edu.hk)



History of AI and Games

Alan Turing's Chess Program

- Alan Turing (re)invented the Minimax algorithm and used it to play chess
- The program was created in 1950 and written on paper
- Alan M. Turing. Digital computers applied to games. Faster than thought, 101, 1953.



For 'Faster than Thought' Ed. B.V. Bowden London 1953

AMT/B/7/1

Digital Computers applied to Games.

When one is asked 'Could one make a machine to play chess ?', there are several possible meanings which might be given to the words. Here are a few:-

i) Could one make a machine which would obey the rules of chess, i.e. one which would play random legal moves, or which could tell one whether a given move is a legal one ?

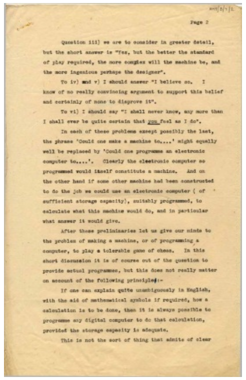
ii) Could one make a machine which would solve chess problems, e.g. tell one whether, in a given position, white has a forced mate in three ?

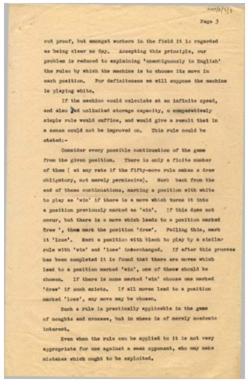
iii) Could one make a machine which would play a reasonably good game of chess, i.e. which, confronted with an ordinary (that is, not particularly unusual) chess position, would after two or three minutes of calculation, indicate a passably good legal move?

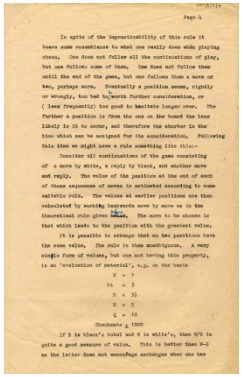
iv) Could one make a machine to play chess, and to improve its play, game by game, profiting from its experience ?

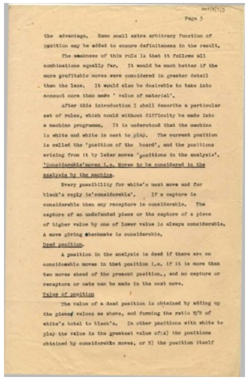
To these we may add two further questions, unconnected with chess, which are ~~likely~~ likely to be on the tip of the reader's tongue.

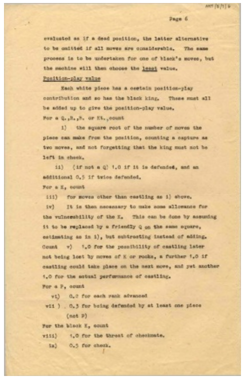
v) Could one make a machine which would answer questions

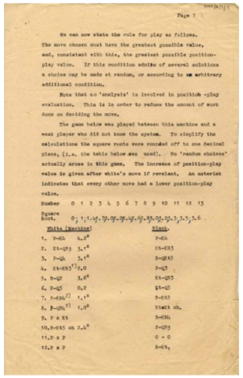


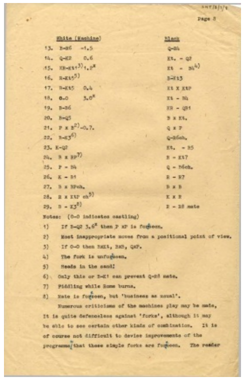


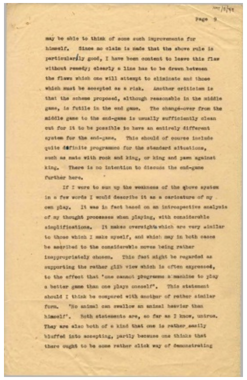


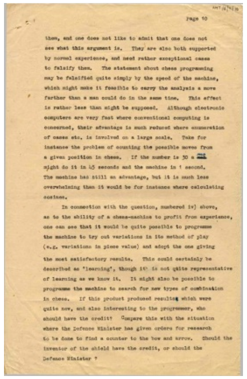






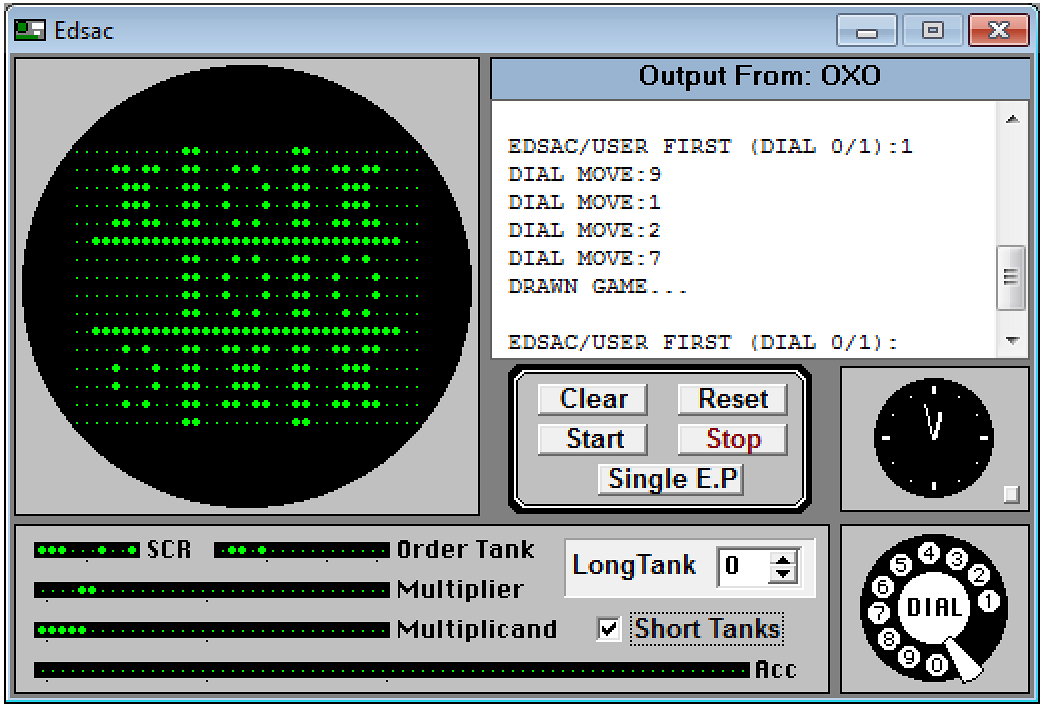






Tic-Tac-Toe game by A. S. Douglas

- The first software to master a game is a digital version of the Tic-Tc-Toe game, also known as Noughts and Crosses (or OXO)
- Programmed by A. S. Douglas in 1952 as part of his doctoral dissertation at Cambridge for EDSAC (Electronic Delay Storage Automatic Calculator)

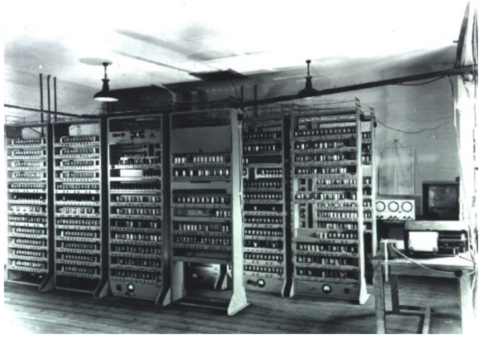


```
EDSAC/USER FIRST (DIAL 0/1):1
DIAL MOVE:9
DIAL MOVE:1
DIAL MOVE:2
DIAL MOVE:7
DRAWN GAME...

EDSAC/USER FIRST (DIAL 0/1):
```

LongTank 0

☑ Short Tanks



https://en.wikipedia.org/wiki/OXO

http://www.computerspacefan.com/Noughts%20and%20Crosses%20Sim.htm
(Noughts and Crosses running on an EDSAC simulator)

Arthur Samuel's Checker Program

- Arthur Samuel invented the form of machine learning that is now called reinforcement learning using a program that learned to play Checkers by playing against itself
- It was demonstrated to the public on TV in 1956

https://www.ibm.com/ibm/history/ibm100/us/en/icons/ibm700series/impacts/

Trying a Checkers game online

Early Research on Game-Playing AI

- Classic board games such as Checkers and Chess are the focus
- These games have great complexity arising from simple rules which had challenged the best human minds for hundreds or even thousands of years

Checkers

- After over 3 decades of research on tree search, the Chinook Checkers player managed to beat the World Checkers Champion Marion Tinsley in 1994
- The Checkers game was solved in 2007

http://afflictor.com/tag/marion-tinsley/

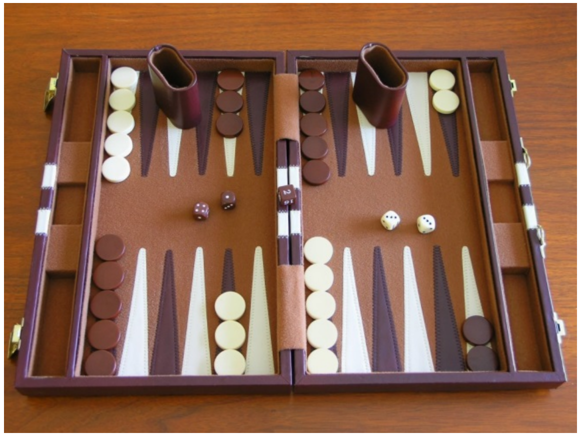
Chess

- IBM's Deep Blue, consisted of a Minimax algorithm with numerous Chess-specific modifications and a very highly tuned board evaluation function running on a custom supercomputer
- Deep Blue famously won against the grandmaster of Chess, Garry Kasparov, in 1997
- It is now possible to download public domain software that will play better than any human player when running a regular laptop

https://www.youtube.com/watch?v=NJarxpYyoFI

Backgammon

- Gerald Tesauro developed the backgammon software named TD-Gammon in 1992
- TD-Gammon employs an artificial neural network which is trained via temporal difference learning by playing backgammon against itself a few million times
- TD-Gammon managed to play backgammon at a level of a top human backgammon player



https://www.youtube.com/watch?v=Eaa7KrIc4ys
(How to play Backgammon)

Jeopardy!

- Watson competed on the TV game Jeopardy! And won $1 million against former winners of the game

https://www.youtube.com/watch?v=P18EdAKuC1U
(AI vs human players)

https://www.youtube.com/watch?v=EBVhE5Ws7q4
(Sample episode with human players)

Go

- Go has a branching factor of about 250 and a vast search space larger than that of Chess
- Lee Sedol, a 9-dan professional Go player, lost a 5-game match in 2016 against Google DeepMind's AlphaGo software which featured a deep reinforcement learning approach
- Ke Jie, the world's number 1 ranking player, lost a 3-game Go match against AlphaGo in 23-27 May 2017

https://www.youtube.com/watch?v=rOL6QJdAlm8
(AlphaGo beats Lee Sedol in final match)

https://www.youtube.com/watch?v=OlqRIWAkWZo
(Ke Jie: AlphaGo is almost like the God of Go)

Extending to Video Games

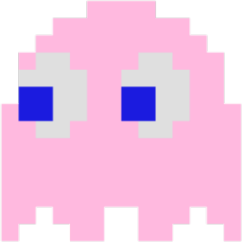
- Classic board games have discrete turn-based mechanics and the full state of the game is visible to both players
- In the last decade and half, a research community has worked on applying AI to video games
- A large part of the research focuses on developing AI for playing games, either as effectively as possible, or in the style of humans (or a particular human)
- A notable milestone in video game AI playing was reached in 2014 when algorithms developed by Google DeepMind learned to play several console video games on a super-human skill level just from the raw pixel inputs

Ms Pac-Man

- This game was practically solved in 2017 by the Microsoft Maluuba team using a hybrid reward architecture reinforcement learning technique

Blinky

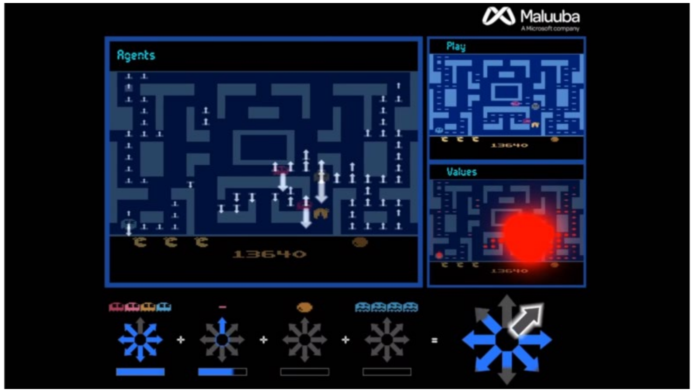
Inky



Pinky

Sue

https://www.giantbomb.com/ms-pac-man/3030-6332/characters/



https://www.youtube.com/watch?v=zQyWMHFjewU (Divide and conquer: How Microsoft researchers used AI to master Ms. Pac-Man)

Diablo III

- Certain video games created some of their content algorithmically during runtime, rather than having it designed by humans
- Diablo III has random scripted events and randomized environments which aims to increase replay value of the game

https://us.diablo3.com/en/game/guide/gameplay/world

Nevermind

- Nevermind can use biofeedback sensor to track the emotional changes of the player and adapt the game accordingly

https://www.youtube.com/watch?v=r7jbxn8g14g
(Nevermind (Biofeedback))

Why Games for AI?

Games are Hard and Interesting Problems

- Complexity and interestingness of games as a problem makes them desirable for AI
- Games are hard because their finite state spaces are often vast, and the goodness of any game state is hard to assess properly
- Many games are NP-hard, meaning that the worst-case complexity of solving them is very high
- A typical game in Go has about 10^{170} states
- According to a study by Usunier 2016, a typical game in Starcraft has at least $10^{1,685}$ possible states, which require about 700 bytes to represent its search space
- As a comparison, the number of protons in the observable universe is only about 10^{80}, which require about 34 bytes to represent

Rich Human-Computer Interaction

- Computer games are dynamic media by definition and offer one of the richest forms of human-computer interaction (HCI)
- The available options for the player are linked to the game action space and the complexity associated with it
- In addition to traditional keyboard, mouse, and tablet-like haptic, other modalities such as new game controllers or devices that capture heart rate variability, body movement, speech may be used
- Interaction between the player and the game is of key importance for AI research as it gives algorithms access to rich player experience stimuli and player emotional manifestations

Games are Popular

- Since video games were introduced, games have generated global market revenue in the order of $100 billion
- Games have become popular because
  - Games enhance a user's intrinsic motivation and engagement by offering interactivity capacities with a virtual environment
  - Games can be played using various forms of devices (PC, mobile phone, console, etc.)
  - Games shape our social and cultural values at large (e.g., Pokemon Go)
  - Experts (i.e., professional players) for many board and digital games that are world-ranked according to their gameplay performance have participated regularly in competitions against AI algorithms
- As games become more popular, grow in quantity, and become more complex, new AI solutions are constantly required to meet the new technological challenges

There are Challenges for All AI Areas

- Games challenge all core areas of AI
  - Signal processing (from multiple modalities of fast-paced interaction and spatial-temporal nature of the signals)
  - Machine learning (Go and video games)
  - Tree search (Checkers and Chess)
  - Knowledge representation and reasoning, natural language processing (Jeopardy!)
  - Planning and navigation (Starcraft)

Games Best Realize Long-Term Goals of AI

- Social and Emotional Intelligence
  - a user under gaming conditions, more than any other form of human-computer interaction, is generally open to affective-based alterations of the interaction and influences of his/her emotional state
- Computational Creativity
  - the fusion of the numerous and highly diverse creative domains, visual art, sound design, graphic design, interaction design, narrative, etc., within a single software application that makes games the ideal arena for studying the potential of software to autonomously generate creative outcomes
- General Intelligence
  - the capacity of AI to play unseen games well, i.e., general game playing, has seen a number of advancements in recent years

Why AI for Games?

AI Plays and Improves Your Game

- AI plays games with two core objectives in mind: play well and/or play believably (or human-like, or interestingly)
- AI that plays well as a player character focuses on optimizing the performance of play, which is important for automatic game testing and for evaluation of game design as a whole
- AI that plays well as a non-player character can empower dynamic difficulty adjustment and automatic game balancing mechanisms that enhance the experience for the player

More Content, Better Content

- AI can be applied in content generation
- Memory consumption should be limited so content can typically be compressed by keeping it “unexpanded” until needed
- If new content can be generated with sufficient variety, quality and quantity, then it may become possible to create truly endless games with ultimate replay value
- When content generation is associated with aspects of play we can expect personalized and adaptive play to emerge via modification of content

Reference

- Artificial Intelligence and Games
  - Chapter 1