

AI Planning for Autonomy

1. Short Introduction to Automated (AI) Planning

Nir Lipovetzky



THE UNIVERSITY OF
MELBOURNE

Semester 2, 2020
Copyright, University of Melbourne

Intro: Nir Lipovetzky

I've completed my PhD on 2012 in Barcelona.

My main research interests are in the area of Automated planning, specifically:

- Planning technology (algorithms [LAPKT](#), and tools [planning.domains](#), [planimation](#), etc.)
- Planning for Autonomy in industry
- Bounded General AI (Atari) \neq AGI
- Integration of Planning, Learning and Control

I've been at Melbourne Uni since 2012. You can find me at ~~DMD building, office 6.19~~
Zoom/Piazza/E-mail.

Contact: nir.lipovetzky@unimelb.edu.au

Love Futbol, but not FC. Barcelona ;)

Contact: adrianp@unimelb.edu.au

Great Knowledge on COMP90054 topics.

Head Tutor: Guang Hu

- Tutors: former top-students of COMP90054, Phd Students.

You can find Guang at ~~DMD building, office 6.13~~ Zoom/Piazza/E-mail.

Contact: ghu1@student.unimelb.edu.au

Outline of the Course

- 1 Introduction to AI and (AI) Planning
- 2 Classical Planning as Heuristic Search and Width-Based Search
- 3 Beyond Classical Planning:
 - *Factored-Model-Free, Non Determinism, Uncertainty, Soft goals, Plan Recognition, Epistemic (social) Planning, Path-Planning, Control*
- 4 Reinforcement Learning: Learning through Experience
- 5 Multi agent Planning
- 6 Hot/Latest exciting discussions on AI Ethics

Acknowledgment:

→ Slides based on earlier courses by Hector Geffner, Joerg Hoffmann, and Carmel Domshlak

Things you Should Know to Enjoy the Course

- 1 Algorithms such as Dynamic Programming
- 2 Basic Set Theory and Propositional Logic
- 3 Probabilistic Theory such as Conditional Probabilities
- 4 Python, start this week doing a Tutorial to refresh your knowledge

and importantly, you need to [stay up to date](#) reviewing and understanding the content, as most lectures build up on previous knowledge

If you don't understand...have a question...

↪ use [Piazza](#): COMP90054 StackOverflow, be active in answering questions!

Weekly Lectures:

- [Short videos](#) covering the equivalent of 1h background knowledge (instead of Thursday's Lecture)
- [Live Zoom lecture](#): Q&A, polls, and worked examples

→ [Zoom Tutorials](#) starting on week 2

→ [Online Review Quiz](#) starting on week 2

Assignments:

- [Individual early project](#),
- [optional](#) Assignment for extra credit
- [Final group project](#) in the form of a competition

Code released by Berkeley University, used widely for teaching purposes across best universities around the globe

We are in contact with current competitions run at RMIT, and other universities internationally

→ **Hall of Fame (best 8 teams each year will compete with best teams 2016-present, forever!)**

<https://sites.google.com/view/pacman-capture-hall-fame/>

Do not use online solutions, we are going to find out and it's not worth it!

It's an **open-ended project**, set your own goals

- **AI Concepts:** What are we actually talking about?
 - Clarify what the (modern) research field of AI does, and does not, try to do.
- **AI History:** How did this come about?
 - Just a little background to illustrate how we came from 'classical AI' to 'modern AI'.
- **AI Today:** What is the landscape of techniques and applications?
 - Rough overview, and some examples.

What is Intelligence?

Quiz! can choose multiple answers

- Ability to think . . . ?
- Simulating the brain . . . ?
- Creativity . . . ?
- Ability to learn . . . ?
- Being good at maths . . . ?
- Playing good Chess . . . ?

What is *Artificial* Intelligence?

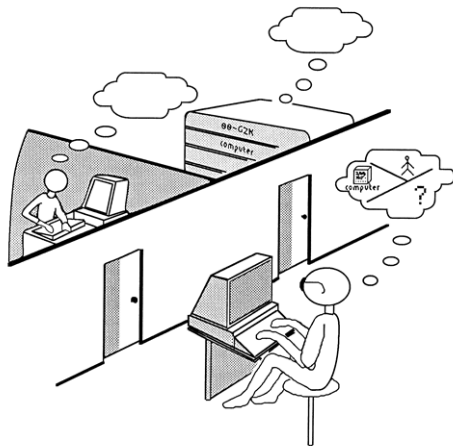
An engineering standpoint

The science of making machines do things that would require intelligence if done by humans. (M. Minsky)

Is this an operational definition? Hmm...

- How do we know what **human** activities require intelligence?
- BTW, what is human intelligence?

Acting Humanly with intelligence: Turing Test



- Not reproducible... only a proof of concept?

What is *Artificial* Intelligence?

Another perspective please

The exciting new effort to make computers think. Machines with minds, in the full and literal sense. (J. Haugeland)

Same problems as with Minsky's definition:

- what is **thinking**?
- what is **mind**?

What is *Artificial* Intelligence?

A Rational perspective

The branch of computer science that is concerned with the automation of intelligent behavior. (Luger and Stubblefield)

- **Intelligent behavior:** make 'good' (rational) action choices
- Are humans 'rational' agents?

What is *Artificial* Intelligence?

A Rational perspective

The branch of computer science that is concerned with the automation of intelligent behavior. (Luger and Stubblefield)

- **Intelligent behavior:** make 'good' (rational) action choices
- **Are humans 'rational' agents?** We often make **mistakes**; we are not all chess grandmasters **even though we may know all the rules of chess**. More about human systematic errors (*Thinking, fast and slow - Kahneman*)”

Agents:

- Perceive the environment through **sensors** (→**percepts**).
- Act upon the environment through **actuators** (→**actions**).

→ **Examples**? Humans, animals, robots, software agents (softbots), . . .

Rational Agents . . . do ‘the right thing’!

→ Any idea what that means, ‘**do the right thing**’?

Agents:

- Perceive the environment through **sensors** (→**percepts**).
- Act upon the environment through **actuators** (→**actions**).

→ **Examples**? Humans, animals, robots, software agents (softbots), . . .

Rational Agents . . . do ‘the right thing’!

→ Any idea what that means, ‘**do the right thing**’? Rational agents select their actions so as to maximize a **performance measure**.

→ **Q**: What’s the performance measure of an autonomous vacuum cleaner?

Agents:

- Perceive the environment through **sensors** (→**percepts**).
- Act upon the environment through **actuators** (→**actions**).

→ **Examples**? Humans, animals, robots, software agents (softbots), . . .

Rational Agents . . . do ‘the right thing’!

→ Any idea what that means, ‘**do the right thing**’? Rational agents select their actions so as to maximize a **performance measure**.

→ **Q**: What’s the performance measure of an autonomous vacuum cleaner? m^2 per hour, Level of cleanliness, Energy usage, . . .

→ What if the vacuum cleaner’s sensors are not good enough? [click: Robot Revolution news](#) ”

... TRY to do 'the right thing'!

→ The hypothetical best case ('the right thing') is often unattainable.

→ The agent might not be able to perceive all relevant information. (Is there dirt under this bed?)

Rationality vs. Omniscience:

- An **omniscient agent** knows everything about the environment, and knows the actual effects of its actions.
- A **rational agent** just makes the best of what it has at its disposal, maximizing expected performance given its percepts and knowledge.

→ **Example?** I check the traffic before crossing the street. As I cross, I am hit by a meteorite. Was I lacking rationality?

Mapping your input to the best possible output:

Performance measure × Percepts × Knowledge → Action

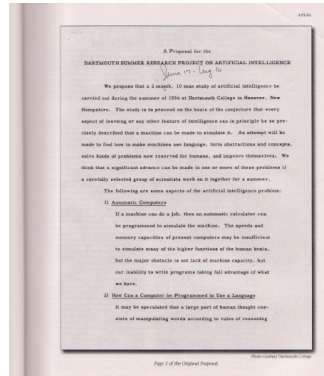
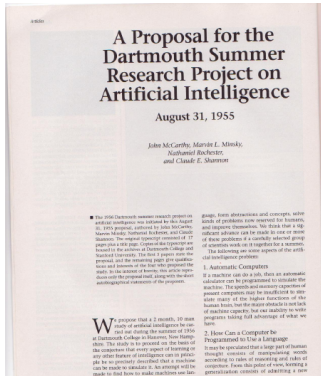
What Does AI Do?

→ Artificial intelligence as an idea can be roughly classified along the dimensions **thinking vs. acting** and **humanly vs. rationally**.

	Humanly	Rationally
Thinking	Systems that think like humans (Cognitive Science)	Systems that think rationally (Logics: Knowledge and Deduction)
Acting	Systems that act like humans (Turing Test)	Systems that act rationally (How to make good action choices)

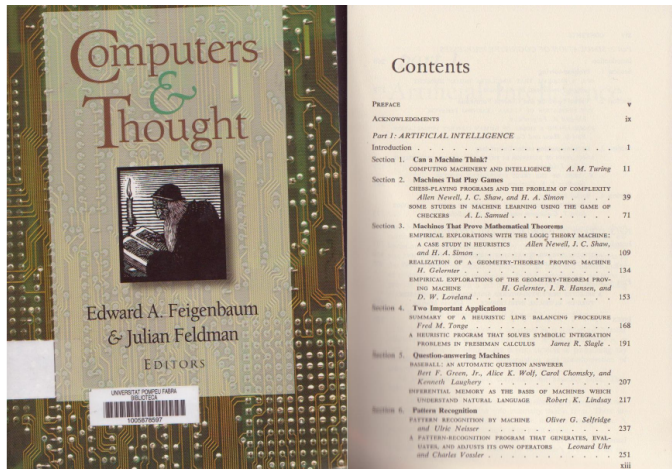
→ A central aspect of intelligence (and one possible way to define it) is the **ability to act successfully** in the world

Origins of AI: Darmouth 1956



The proposal (for the meeting) is to proceed on the basis of the conjecture that every aspect of . . . intelligence can in principle be so precisely described that a machine can be made to simulate it

Computers and Thought 1963



An early collection of AI papers and programs for playing chess and checkers, proving theorems in logic and geometry, planning, etc.

Many of the key AI contributions in 60's, 70's, and early 80's had to do with [programming](#) and the [representation of knowledge](#) in programs:

- Lisp (Functional Programming)
- Prolog (Logic Programming)
- Rule-based Programming
- 'Expert Systems' Shells and Architectures

For writing an AI dissertation in the 60's, 70's and 80's, it was common to:

- pick up a task and domain X
- analyze/introspect/find out how task is solved
- capture this reasoning in a program

The dissertation was then

- a **theory** about X (scientific discovery, circuit analysis, computational humor, story understanding, etc), and
- a **program** implementing the theory, **tested** over a few examples.

Many great ideas came out of this work . . . but there was a problem . . .

→ Theories expressed as programs cannot be proved wrong: when a program fails, it can always be blamed on 'missing knowledge'

Three approaches to this problem

- narrow the domain (expert systems)
 - problem: lack of generality
- accept the program is just an illustration, a demo
 - problem: limited scientific value
- fill up the missing knowledge (intuition, commonsense)
 - problem: not successful so far

→ The knowledge-based approach reached an **impasse** in the 80's, a time also of debates and controversies:

- **Good Old Fashioned AI** is 'rule application' but intelligence is not (Haugeland)

Many criticisms of mainstream AI partially valid then; less valid now.

Formalization of AI techniques and increased use of mathematics. Recent issues of AIJ, JAIR, AAAI or IJCAI shows papers on:

- 1 SAT and Constraints
- 2 Search and Planning
- 3 Probabilistic Reasoning
- 4 Probabilistic Planning
- 5 Inference in First-Order Logic
- 6 Machine Learning
- 7 Natural Language
- 8 Vision and Robotics
- 9 Multi-Agent Systems

→ Areas 1 to 4 often deemed about techniques, but more accurate to regard them as **models and solvers**.

$$Problem \implies \boxed{Solver} \implies Solution$$

Example:

- **Problem:** The age of John is 3 times the age of Peter. In 10 years, it will be only 2 times. How old are John and Peter?
- **Expressed as:** $J = 3P$; $J + 10 = 2(P + 10)$
- **Solver:** Gauss-Jordan (Variable Elimination)
- **Solution:** $P = 10$; $J = 30$

Solver is **general** as deals with any problem expressed as an instance of **model**
Linear Equations Model, however, is **tractable**, AI models are not . . .

$$Problem \implies \boxed{Solver} \implies Solution$$

- The basic models and tasks include
 - **Constraint Satisfaction/SAT:** find state that satisfies constraints
 - **Planning Problems:** find action sequence that produces desired state
 - **Planning with Feedback:** find strategy for producing desired state
- Solvers for these models are **general**; not tailored to specific instances
- All of these models are **intractable**, and some extremely powerful (POMDPs)
- The challenge is mainly computational: **how to scale up**
- For this, solvers must **recognize and exploit structure** of the problems
- Methodology is **empirical**: benchmarks and competitions

- **SAT**: determine if there is a **truth assignment** that satisfies a set of clauses

$$x \vee \neg y \vee z \vee \neg w \vee \dots \quad (1)$$

- Problem is NP-Complete, which in practice means worst-case behavior of SAT algorithms is **exponential** in number of variables ($2^{100} = 10^{30}$)
- Yet current SAT solvers manage to solve problems with **thousands of variables** and clauses, and used widely (circuit design, verification, planning, etc)
- **Constraint Satisfaction Problems (CSPs)** generalize SAT by accommodating non-boolean variables as well, and constraints that are not clauses
- Key is **efficient (poly-time) inference** in every node of search tree: **unit resolution, conflict-based learning**, ...
- Many other ideas **logically possible**, but **do not work** (don't scale up): pure search, pure inference, etc.

Classical Planning Model

- Planning is the **model-based approach** to autonomous behavior,
- A system can be in one of many **states**
- States assign **values** to a set of **variables**
- **Actions** change the values of certain variables
- **Basic task**: find **action sequence** to drive **initial** state into **goal** state

$$Model \implies \boxed{Planner} \implies Action\ Sequence$$

- **Complexity**: NP-hard; i.e., exponential in number of vars in **worst case**
- **Planner** is generic; it should work on any domain no matter what variables are about

Why do we need such an AI?

- **Cheess**: 2 player zero-sum game
- **Music/Speech Recognition**
- **Recommender systems**
- **Medical Diagnosis**: decision support systems
- **Self-driven car**
- **Playing Atari Games** [Deep Learning](#)
- ...

Why do we need such AI Planning?

Settings where greater autonomy required:

- **Space Exploration:** (RAX) first artificial intelligence control system to control a spacecraft without human supervision (1998)
- **Business Process Management**
- **First Person Shooters & Games:** classical planners playing Atari Games
- **Interactive Storytelling**
- **Network Security**
- **Logistics/Transportation/Manufacturing:** Multi-model Transportation, forest fire fighting, PARC printer
- **Wherehouse Automation:** Multi-Agent Path Finding, Post China, Amazon
- Automation of Industrial Operations (Schlumberger)
- Self Driving Cars ...

Find out more at [ICAPS in Action \(right panel\)](#)

Summary: AI and Automated Problem Solving

- A **research agenda** that has emerged in last 20 years: **solvers** for a range of **intractable models**
- **Solvers** unlike other programs are **general** as they do not target individual problems but families of problems (**models**)
- The challenge is **computational**: how to scale up
- Sheer **size of problem** shouldn't be impediment to meaningful solution
- **Structure** of given problem must recognized and **exploited**
- Lots of room for **ideas** but methodology **empirical**
- Consistent **progress**
 - effective inference methods (derivation of h, conflict-learning)
 - islands of tractability (treewidth methods and relaxations)
 - transformations (compiling away incomplete info, extended goals, . . .)