Introduction to Artificial Intelligence

Week 1

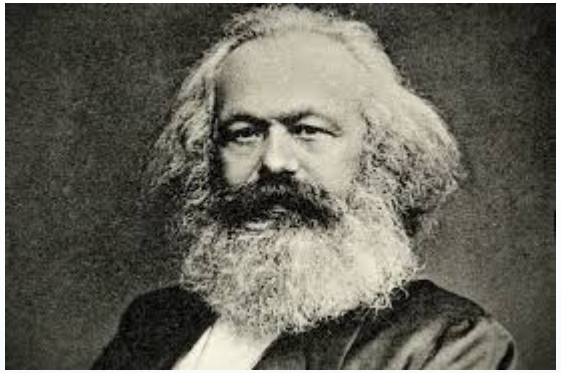
My Contact

- Email: m.makhmutov@innopolis.ru
  - Please preface all emails about the course with the subject [IntroAI] to allow for a quick response
- Office 418
  - Office hours – Thursday 4:20 – 6:00 PM
  - This is the BEST method for us to make contact and for you to receive a solution
- Please do not telegram, use it only in case of urgent questions
- FYI
  - I sleep at night
  - Usually I will ask for larger issues for you to come see me

TAs

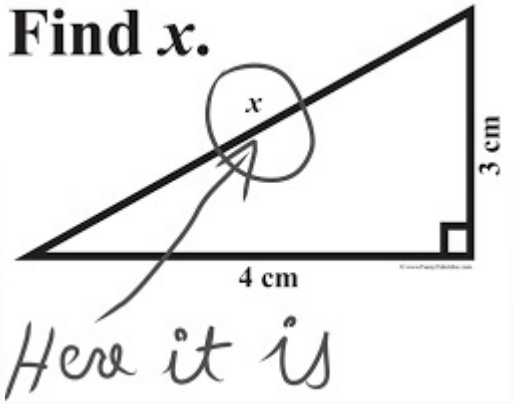
- Rufina Galieva (B20-01, B20-02) r.galieva@innopolis.university
- Dmitry Devitt (B20-03, B20-04) d.devitt@innopolis.university
- Munir Makhmutov (B20-05, B20-06) m.makhmutov@innopolis.ru

Marks Breakdown



- Lab Participation – 10%
  - No lab in first week and week of the midterm
  - Students are allowed to miss up to 2 labs without penalty to grade
    - I don't care if this is due to club reasons, illness, etc.
- Assignment 1 – 20%
- Midterm Test – 25%
- Assignment 2 – 20%
- Final – 25%
- Bonus – 5%
  - Given based on instructor's and TA's impression

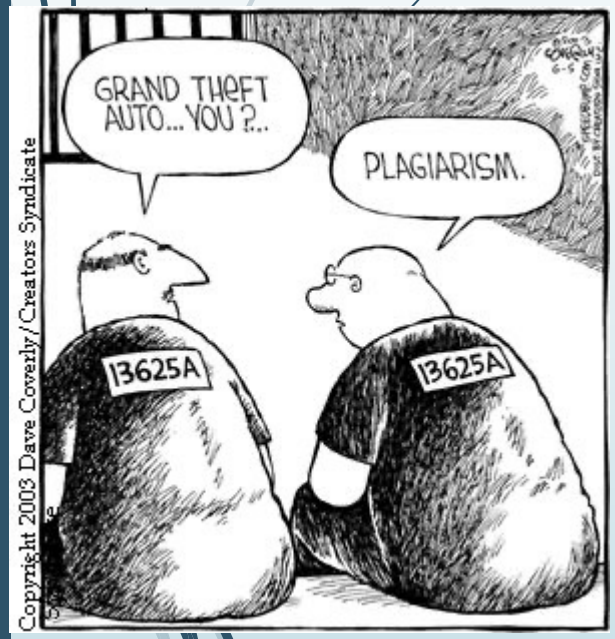
Course Procedures



- As a matter of course I will not accept late assignments or missed tests without prior legitimate excuses given in a reasonable period before the due date
- Not legitimate
  - I was working on a project/hackaton for X and didn't do the work for this class
  - I was sick due to my own hand (*i.e.* too much parties)
  - My computer broke two weeks ago and I couldn't work on the assignment
- There is a short time period between final exam and grades being locked – any requests beyond this period will be not heard
  - I do not regrade based on "but I need more marks to get the grade of X"
  - Legitimate reasons for regrade
    - Marks not added correctly
    - Mistake by the marker based in fact
      - note if we cannot understand what you wrote as you intended then we cannot regrade it
  - You may always request feedback or clarification
    - We want you to learn from any mistakes

Academic Misconduct

- **Academic misconduct** is any action or attempted action that may result in creating an unfair **academic** advantage for oneself or an unfair **academic** advantage or disadvantage for any other member or members of the **academic** community. – UC Berkley
- Some of academic misconduct cases:
  - Cheating – copying or communicating answers to an evaluation
  - Plagiarism – using someone else's work as your own
  - Work for hires – hiring out your course work to a third party
  - Submission of work for which credit has been obtained upon previously
  - Possession of unauthorised aids
  - Preventing any other students from accessing materials for an academic advantage
  - Falsification of records or Impersonation
  - Disruption to classrooms – *e.g.* pulling a fire alarm to get out of a test; failure to follow instructions during a test (*i.e.* pencils down)
  - Accomplices – Enabling or acting as a knowing or negligent party to any of the above



Academic Misconduct Policy at IU

- Innopolis policy (minimums)
  - First offence is 0 on the unit and note in record
  - Second offence is 0 in the course and note in record
  - Third is immediate expulsion
  - Even on a first offense on a major unit such as a final exam this may be enough to lead to an expulsion due to low grades
- Note – we reserve the right to increase penalties in egregious cases
- We reserve the right to use technical methods to detect plagiarism such as MOSS or Turn-it-in

What are you allowed to do

- Cite sources in essays, code, etc. when you use them
- Study for a test with other students in the class
- Ask another student to look at your code as there is a bug you can't figure out – just don't have them fix it for you though
- Demonstrate to another student the method of solving a problem conceptually – don't write their answer for them
- Ask for help from TAs and/or the Professor when you are having difficulties

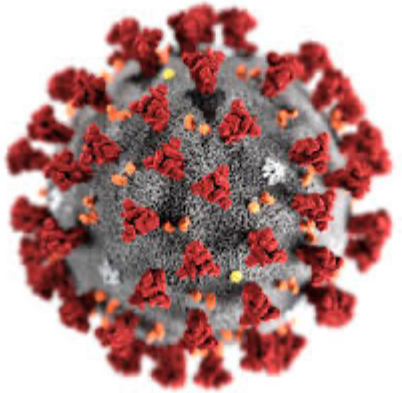
Course Materials

- Main Text
  - Artificial Intelligence a Modern Approach 3rd Edition – Russel and Norvig
- Secondary Text
  - Computational Intelligence For Modeling and Optimization - Ashlock

Tips for Success

- Read the syllabus
  - Lots of answers to many of life's (or at least the course administrations) questions are in it
- Read the book
- Attempt all questions
- Study a little each week – don't think you can just cram on the last day
- Form a regular study group and meet to discuss the assignments and tests
- Come to office hours after attempting the question and ideally have some plan of attack
  - If you're the TA/Professor what 'question' would you want to assist with more?
    - "I don't know how to do Question X!"
    - "I think Question X can be solved by Y method but I cannot see how to make step Z to occur - can you suggest how to accomplish this?"
- Try to find out where these techniques are being used for an issue/problem you care about

COVID Protocols



- Masks
  - Masks are required in all teaching spaces
  - It must cover mouth and nose
- Sanitation
  - Wash your hands
- Distance
  - 1.5m
  - Please file in and out of the room maintaining distance
- Myself and the TAs are empowered to dismiss from the room anyone who is not following these rules for the safety of the class
- Breaks
  - I think everyone gets tired wearing these masks
  - I suggest two 5-10 min breaks every half hour

Attendance

- You are expected to attend both labs and lectures
- We reserve the right to maintain attendance lists in lecture and forward them to the DoE
- The legitimate excuses of students due to actions or illness should be known by DoE
- If there is any individual issues – please contact myself and the DoE in an email

Agreements

Introduction

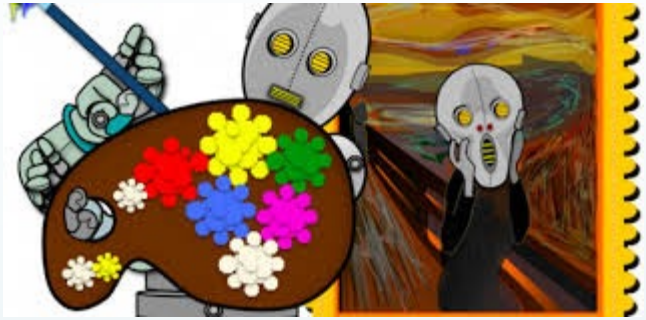
The introduction to the introduction to AI

Artificial Intelligence

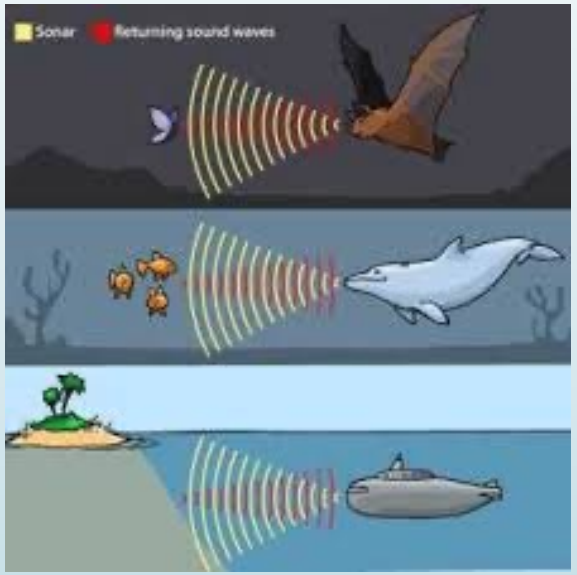
- Breaks down into four major definitions classes:
  - Thinking Humanly
  - Thinking Rationally
  - Acting Humanly
  - Acting Rationally
- Six major areas of concern (Russel & Norvig)
  - Natural Language Processing
  - Knowledge Representation
  - Automated Reasoning
  - Machine Learning
  - Computer Vision
  - Robotics

AI onwards

- Computational Creativity



- Biomimicry



Why should we care?

Catalogue of fears

Probability of computerisation of different occupations, 2013
(1 = certain)

Job	Probability
Recreational therapists	0.003
Dentists	0.004
Athletic trainers	0.007
Clergy	0.008
Chemical engineers	0.02
Editors	0.06
Firefighters	0.17
Actors	0.37
Health technologists	0.40
Economists	0.43
Commercial pilots	0.55
Machinists	0.65
Word processors and typists	0.81
Real-estate sales agents	0.86
Technical writers	0.89
Retail salespeople	0.92
Accountants and auditors	0.94
Telemarketers	0.99

Source: "The Future of Employment: How Susceptible are Jobs to Computerisation?", by C. Frey and M. Osborne (2013)

Economist.com

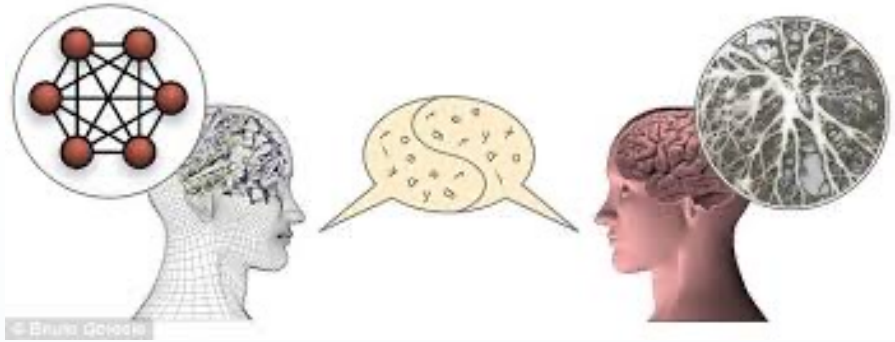
The future job situation:

- The Guardian, Sept. 13, 2016, "Robots will eliminate 6% of all US jobs by 2021, report says"
- BBC News, May 25, 2016, "Foxconn replaces '60,000 factory workers with robots'"
- The Economist, Jun. 25, 2016, "The impact on jobs: Automation and anxiety"

Why should we care?

- Forbes Article - Automation, COVID, And The Future Of Work
  - Oct 2020 by Adi Gaskell
  - Questions the automation of jobs
  - Jobs in companies where AI has been engaged have actually INCREASED 15%
    - Firms productivity which engaged with AI was increased
    - This lead the firms to engage with new locations and move into new markets
  - Young people are the most vulnerable to employment issues
    - Most younger people are in jobs which are consumer facing
      - Retail, Services, etc.
      - These are also more gig work or temporary employment opportunities

Why should we care?



- Via the Analysis of building something we learn more about it
- In order to understand our own process of thought
  - Build something which "thinks"
  - Explore what it means to be "rational"
  - What does it mean to be "intelligent"?
- Is there something beyond mechanical action to our brains
  - Free Will

Why should we care?

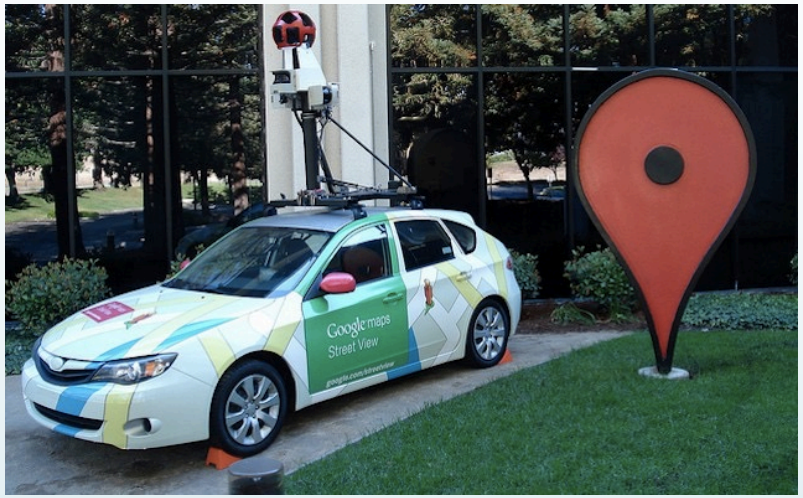
- Fear is defeated by knowledge
- If our future is going to be a terrifying world of the robots and AI
  - "I, for one, welcome our new overlords"
- The wars of the future will not be fought on the battlefield or at sea. They will be fought in space, or possibly on top of a very tall mountain. In either case, most of the actual fighting will be done by small robots. And as you go forth today remember always your duty is clear: To build and maintain those robots. - Simpsons

Intelligence is Situational

- Highly depends on cultural factors
- People of Indigenous tribes
  - Plants/Animals
  - Navigation of waters
- Methods of Transport
  - Horse
    - Do you know how to ride one
    - Do you know how to feed, water, and restaddle one
    - Do you know how to breed them
  - Cars
    - Do you know how to drive
    - Do you know how to refuel, put in washer fluid, do an oil change
    - Do you know how an engine works enough to make repairs or build one

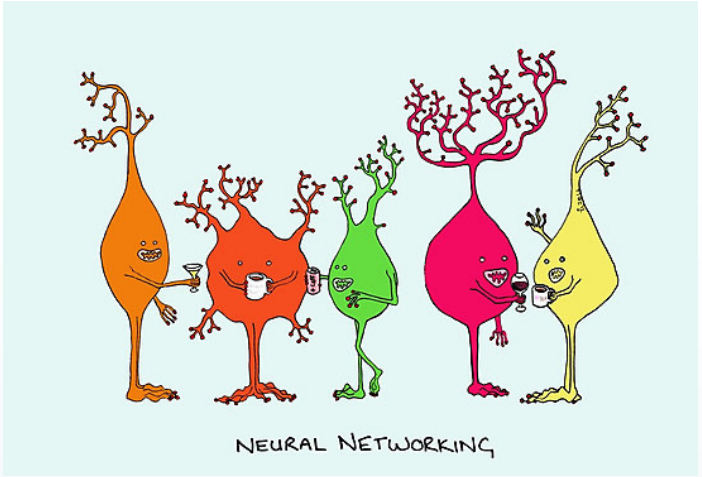
Self Driving Cars – Offloading Intelligence into Technology

- AI is taking the act of driving a car away from us
- Gradual Process
  - Powered Steering
  - Clutch assists
  - Automatic Transmissions
  - ABS
  - Four wheel skid prevention systems
  - Backup Cameras
  - Drive-by-wire
- What was once a skill required to live will become
  - Much like horses are to us now a skill which is reserved for those who are interested
  - Special teams of off-board drivers who will fix what cannot be done by AI now



Human Decision Making

- Decision Trees
- Game Trees
- Expert Systems
- Neural Networks



Bioinspired

- Humans are perhaps not only the one type of intelligence we mimic
- Swarms
  - Birds Flocking
  - Ant Colonies seeking food
  - Bees finding flowers
- Evolution and Breeding
  - Genetics
- Movement in Robotics

Embodied AIs

- A robot without a program is soulless
- An AI without a body is just a ghost
- AIs now have more physical (robot) or digital (agent) embodiment
  - Siri/Galaxy/Cortana – Digital Embodiment
  - Video Game Characters
- Humans love to make non-Human things have human characteristics
- Greek Mythos
  - Ovid's narrative poem Metamorphoses
  - Pygmalion (sculptor) fell in love with a statue he had carved
  - Comes to life due to inference of the gods (Aphrodite)
  - Daedalus (inventor) put voice to a statue using quicksilver (*i.e.* mercury)
  - Talos is an artificial man made of Bronze (Hephaestus)
  - Pandora made of clay (Hephaestus at request of Zeus)

Strong & Weak AI

- Strong AI
  - Human based design
  - Human based constructions
- Weak AI
  - Problem Solving for specific tasks
  - May or may not have human aspects
  - Majority of AI work is on this issue

Welcome to the course

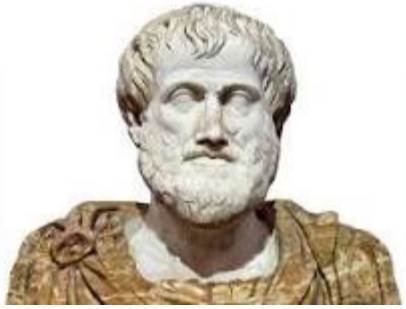
- Think about what products you use every day will be obsolete due to AI in the near future
  - Will this be good for humanity or bad – make a list of pros and cons
- What is Intelligence?

Introduction to Artificial Intelligence

Week 2

What is Intelligence?

Philosophy



- Epistemology – fr. Greek - from Greek ἐπιστήμη, epistēmē, meaning "knowledge", and λόγος, logos, meaning "logical discourse"
- Study of how we know what we know
- Many epistemologists see a separation of Truth/Belief/Justification that gives Knowledge
  - Truth is the objective reality
  - Belief is the subjective idea of what reality is
  - Justification is an explanation
  - Knowledge is True Belief with Justification (comes from Socrates)
  - Example: I had a coffee this morning to keep myself awake for class

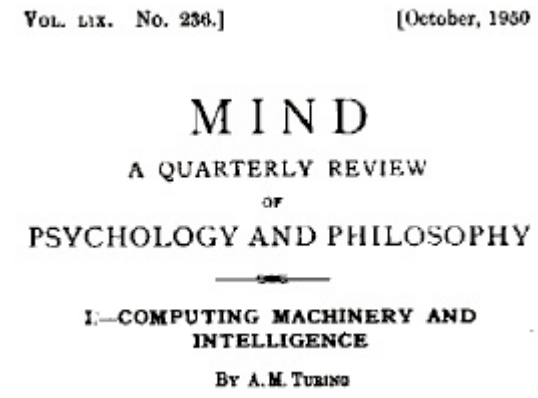
Gettier Cases

- Smith and Jones apply for a job
- Smith asks Jones for some change and knows he has 10 Rouble coins in his pocket
- Smith is informed that it is Jones who will be given the job
  - Jones has ten coins in his pocket, Jones will get the job
  - Smith now believes that the man with 10 coins in his pocket will get the job
- Assume there is a change of plan and Smith is selected; also unknown to Smith to him was that in his pocket are 10 coins
- So, the above propositions were True, were beliefs, were justified
- But Smith didn't KNOW!
  - Includes a justified false beliefs
  - A Justification must also be necessarily the cause

Alan Mathison Turing OBE FRS

- Born 23 June 1912 to an English civil servant for India
- Studied mathematics at King's College, Cambridge, and was elected a fellow
- Developed the Universal Turing Machine as part of the Church-Turing Thesis
- PhD from Princeton; offered a postdoctoral position by John von Neumann but instead returns to England to Cambridge
- Multiple Philosophical fights with Ludwig Wittgenstein
- Went on to work at Bletchley Park
  - Developed the Bombe decoder
  - *The Applications of Probability to Cryptography* and *Paper on Statistics of Repetitions* not released by UK Govt. for 70 years
- Developed and popularized the Turing Test
- Went on to studies of new computers, biology, and mathematics
- Turing was prosecuted in 1952, Public government apology 2009, full pardon in 2013
- Died 1954 of cyanide poisoning (Apple) – most likely at his own hand

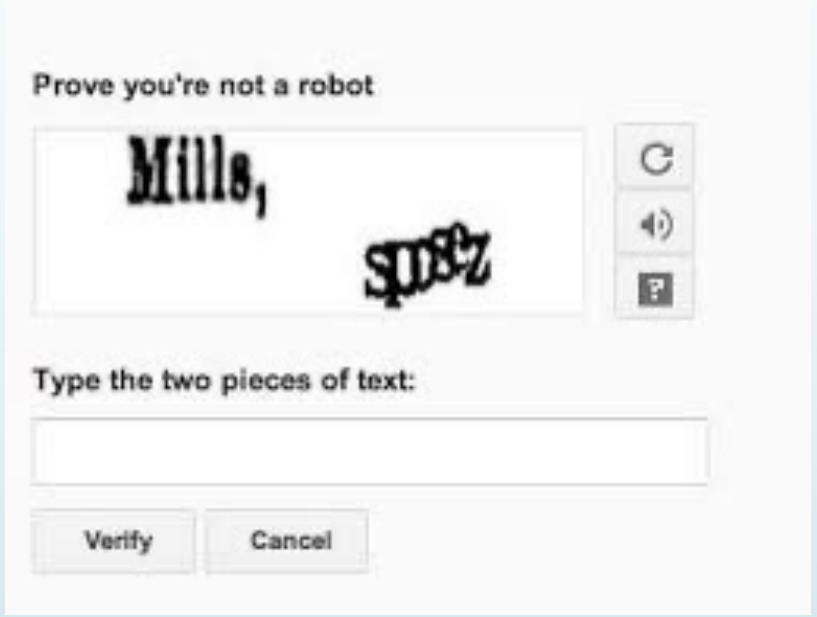
Turing Test



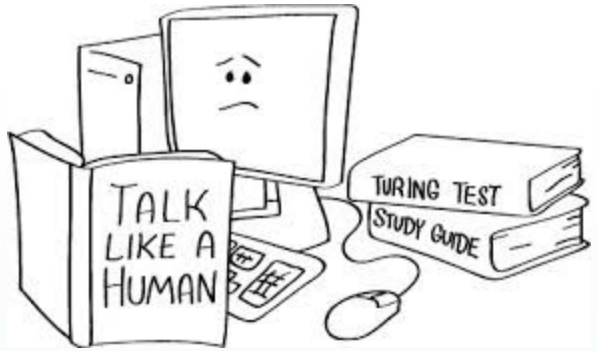
- Published in 1952
- Interrogator
- Human v. Computer
- Can the Interrogator detect the computer
- Inverse of this test used as a human detector

CAPTCHA aka completely automated public Turing test to tell computers and humans apart

- Utilized in a number of systems
  - Email
  - Online ordering
  - Account creation
- Cost effective
- Has been utilized in order to decode OCR issues



Computers do not think like us?



- Image Recognition
  - Segmentation
  - Detection
  - Conceptualization
  - Verbalization

Syntax v. Semantics

- Chair has a meaning conceptually and as an object

Chinese Room Argument

- Thought experiment
- Imagine yourself as someone who has no knowledge of the language of Chinese
  - Rather easy for some of us – but feel free to substitute any language you don't know
- You are locked in the room with a large book of instructions, a pile of paper, and two slots labeled in and out
- From the in slot comes a page of Chinese characters, the instructions tell you what to do and to write before putting it on the out slot

Does this mean you KNOW Chinese?

- You don't know what the characters mean – semantics
- You know actions to be produced on symbols – syntax
- You will be slow to begin with, but as you learn the rule system you before better and better at your actions
- This is how a computer makes actions but is it all it does?

Arguments against the Room

- You are not processing meaningless symbols
  - They have a meaning internally and a different external meaning
- This might be only part of the bigger system
  - Many Chinese rooms
  - Who writes the rule book?
    - Can the rules be changed?
    - Development of rules is now part of many AI

First Order and Prepositional Logic

Why do we care?

- Utilized in a number of AI systems
- Allows us to have a basic idea of ‘knowledge’
- Categories
- Binary predicates
  - Part of the set/Not part of the set

Operations

- For Every ∀
  - For Every X its value is 3; ∀x=3
- There exists ∃
  - There is an X which is 3; ∃x=3
- Conjunction ∧
  - X and Y; x ∧ y
- Disjunction v
  - X or Y; x v y
- Implication →
  - If x then y; x →y
- Biconditional ↔
  - X if and only Y; x ↔ y
- Such that (s.t.) or :

Example Prepositions

- REDHAIR(X) – X has red hair
- EVIL(X) – X is evil
- GLASSES(X) – X wears glasses
- FEMALE(X) – X is female

Example

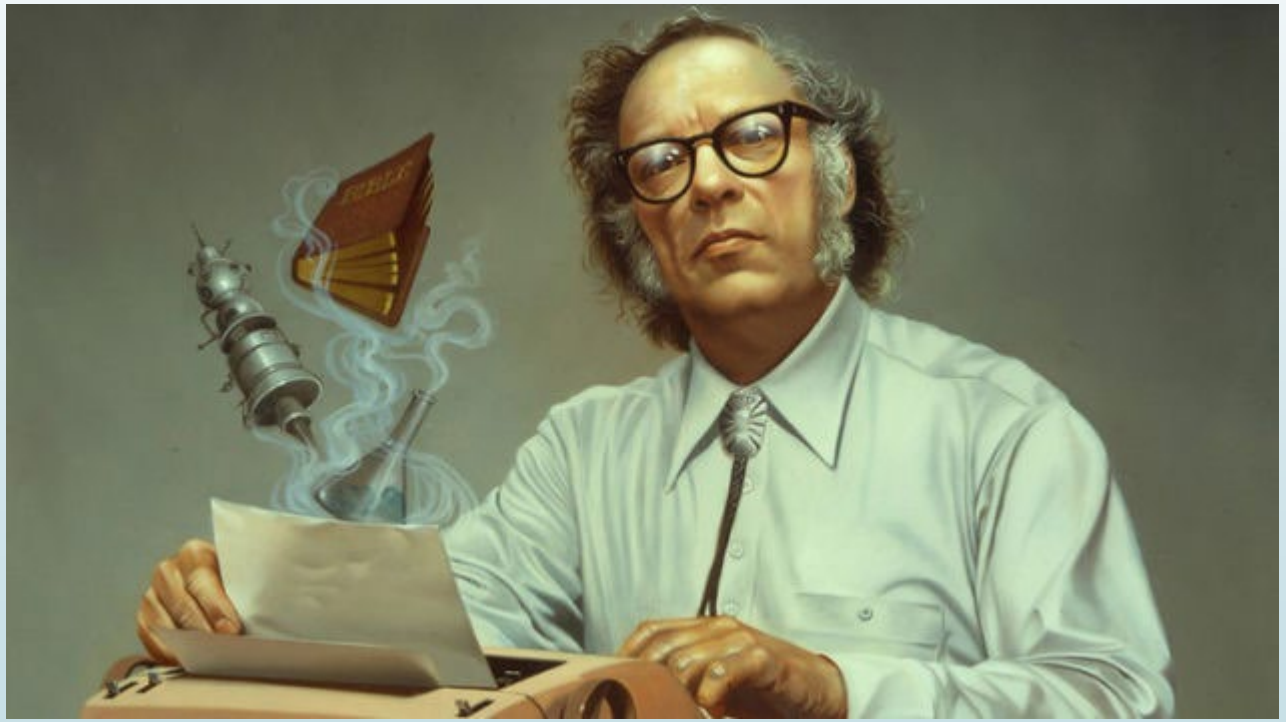
- All female redheads are evil
- $\forall X$, REDHAIR(X) $\wedge$ FEMALE(X) $\rightarrow$ EVIL(X)
- $\forall X$ FEMALE(X) $\rightarrow$ $\exists X$ REDHAIR(X) v EVIL(X)
- For all females there exists one who has red hair or who is evil

Introduction to Artificial Intelligence

What is Intelligence – Part 2

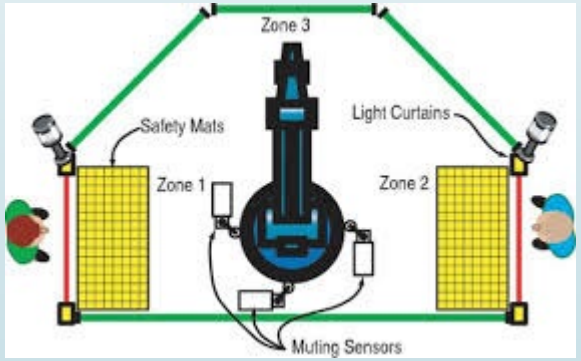
On robotics and ourselves

Isaac Asimov



“A robot may not injure a human being or, through inaction, allow a human being to come to harm.”

LAW ONE – DO NO HARM


Zone 3
Safety Mats
Light Curtains
Zone 1
Zone 2
Muting Sensors

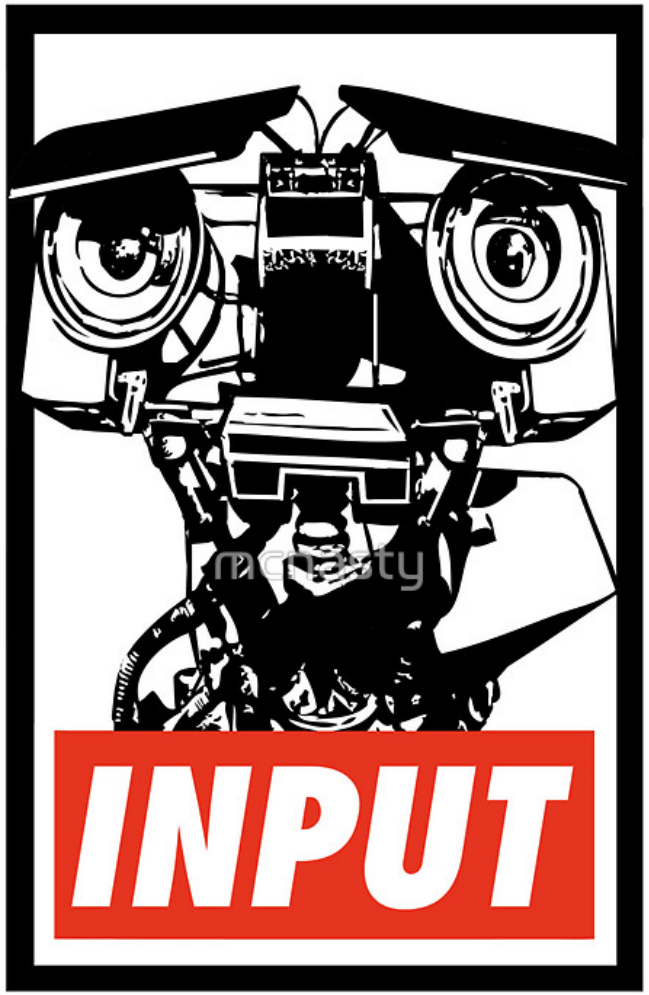
“A robot must obey the orders given to it by human beings, except where such orders would conflict with the First Law.”

LAW TWO – OBEY HUMANS


KEEP
CALM
AND
AVOID
SKYNET

“A robot must protect its own existence as long as such protection does not conflict with the First or Second Law.”

LAW THREE - SELF PRESERVATION


INPUT


KEEP
CALM
AND
NO
DISASSEMBLE

Robots that Kill

- Where does this place military and police drones
  - Police/Military are required to use force which is reasonable at the time for the action (police bound by constitutional laws and militaries by international law)
    - Dallas police utilized a robot with a grenade strapped on it to subdue a suspect who shot at officers
    - Numerous militaries with semi autonomous drones with missiles

Zeroth Law

- **A robot may not injure humanity, or, through inaction, allow humanity to come to harm.**
- **What does this say for ethical calculus?**

Introduction to Artificial Intelligence

Week 3

Agents

Mr. Anderson….

What is an Agent?

- An agent is anything which:
  - Lives in an environment
  - Can perceive the environment via its sensors
  - Can act upon the environment via its actuators
- Agents also have memory of the current and past perceptions of the environment
  - Percept – the current state
  - Percept Sequence – the complete history of Percepts

Humans as Agents

- Environments
  - Home
  - School
  - Work
- Sensors
  - Eyes
  - Ears
  - Tongue
  - Nose
  - Skin
- Actuators
  - Hands
  - Feet
  - Speech

Rationality

- What is rational at any given time depends upon:
  - The performance measure defines success
  - The agent's prior knowledge of the environment
  - The actions which an agent can perform
  - The agent's percept sequence to date
- Definition:
  - For Each possible percept sequence, a rational agent should select an action that is expected to maximize its performance measure, given the evidence provided by the percept sequence and whatever built-in knowledge the agent has.

Agents are rational, not necessarily omniscient

- A jester knocks over the king's wine
  - "What an idiot!"
  - Did the jester make a cartwheel and miss the landing?
  - Did the jester see it being poisoned and the king did not?
- Agents need to gather information before making a decision of an action
  - "Why did you spill my wine jester?!"
- Note that agents may perceive differently from each other
- Agents have a degree of autonomy
  - The King does not control every action of the jester

PEAS – Performance measure, Environment, Actuators, Sensors

- Agent Type
  - Jester
- Performance measure
  - The King is Happy
- Environment
  - The Court
- Actuators
  - Tumbling
  - Juggling
  - Joke Telling
- Sensors
  - Hearing Laughter
  - Seeing Smiles

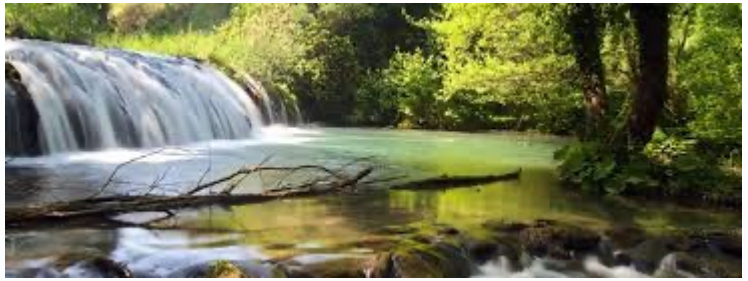
Properties of the Environment (1)

- Fully Observable v. Partially Observable
  - Can the agent see everything in the world via its sensors
  - Does it miss elements which are relevant to the decision making process
- Single v. Multiple Agent
  - Is there only one agent within the space
  - Does there need to be some form of competitive or cooperation between agents
  - How do the agents communicate

Properties of the Environment (2)

- Deterministic v. Stochastic
  - Can we determine the next state of the environment given the previous state and the action we are applying into it
  - We can be certain in environments which are fully observable and deterministic
- Episodic v. Sequential
  - Episodic environments allow for atomic actions not dependent upon those previous
  - Sequential actions are those in which the current decision has consequences on future actions

Properties of the Environment (3)



- Static v. Dynamic
  - Does the environment change as the agent is thinking about its actions
  - Semidynamic if the environment does not change as time passes by but the agent's performance score does
- Discrete v. Continuous
  - Are there states in the environment, or does the environment smoothly act over time
    - Chess – discrete movements
    - Jenga – continuous actions of physics on the blocks
- Known v. Unknown
  - Does the designer of the agent have full knowledge of the rules of the environment

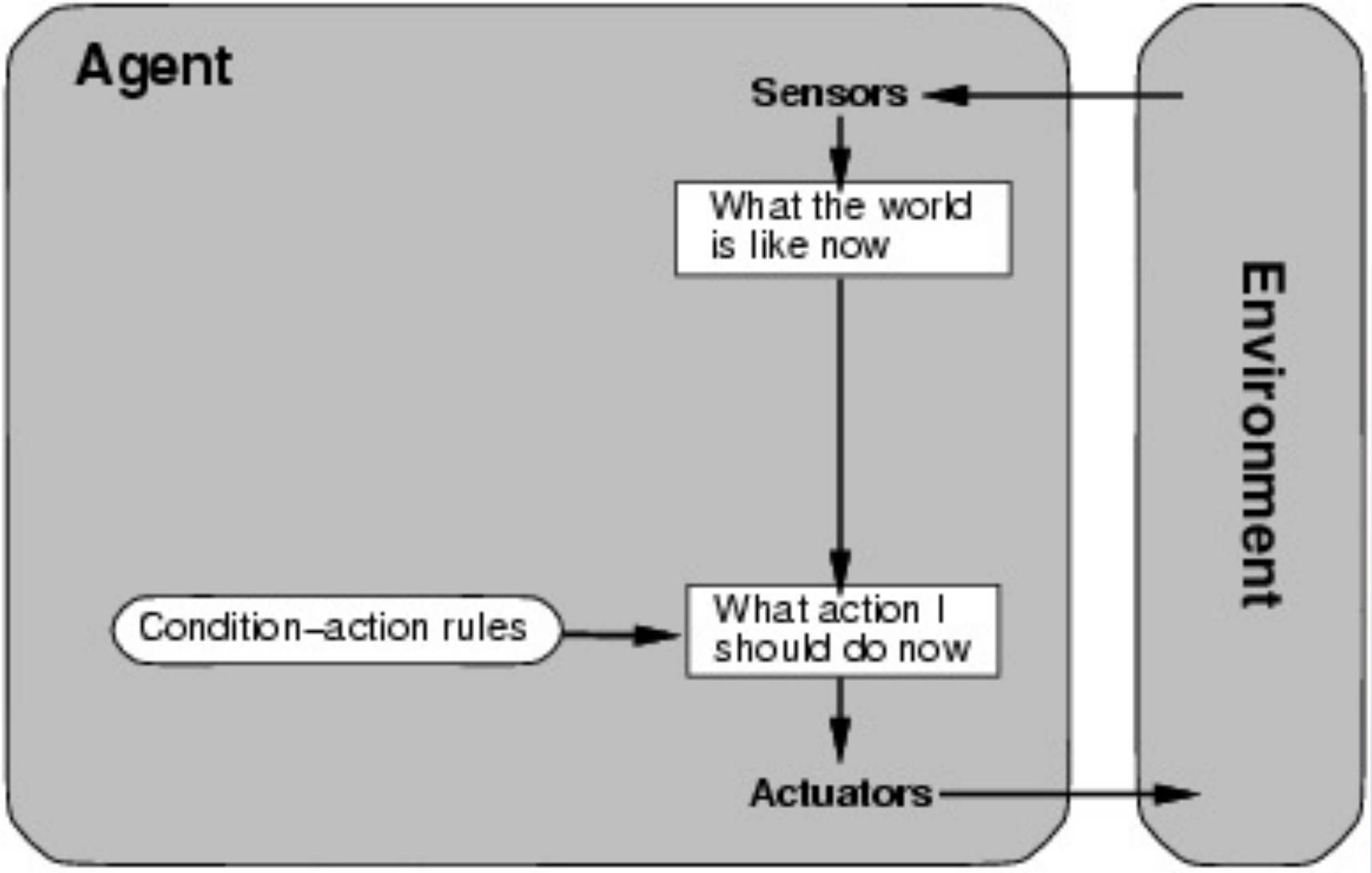
The worst case

- An environment which is partially observable, multiagent, stochastic, sequential, dynamic, continuous, and unknown.
- Ordering in a new restaurant in a new country
  - Can't see how they make the food
  - Have to deal with the waiter, other customers
  - Can't predict when the food will come or how exactly it will taste
  - Depending on what I order, I might order more or might order an ambulance
  - Maybe they don't have anymore what I would like to order
  - Time keeps on ticking
  - Don't know the cultural rules

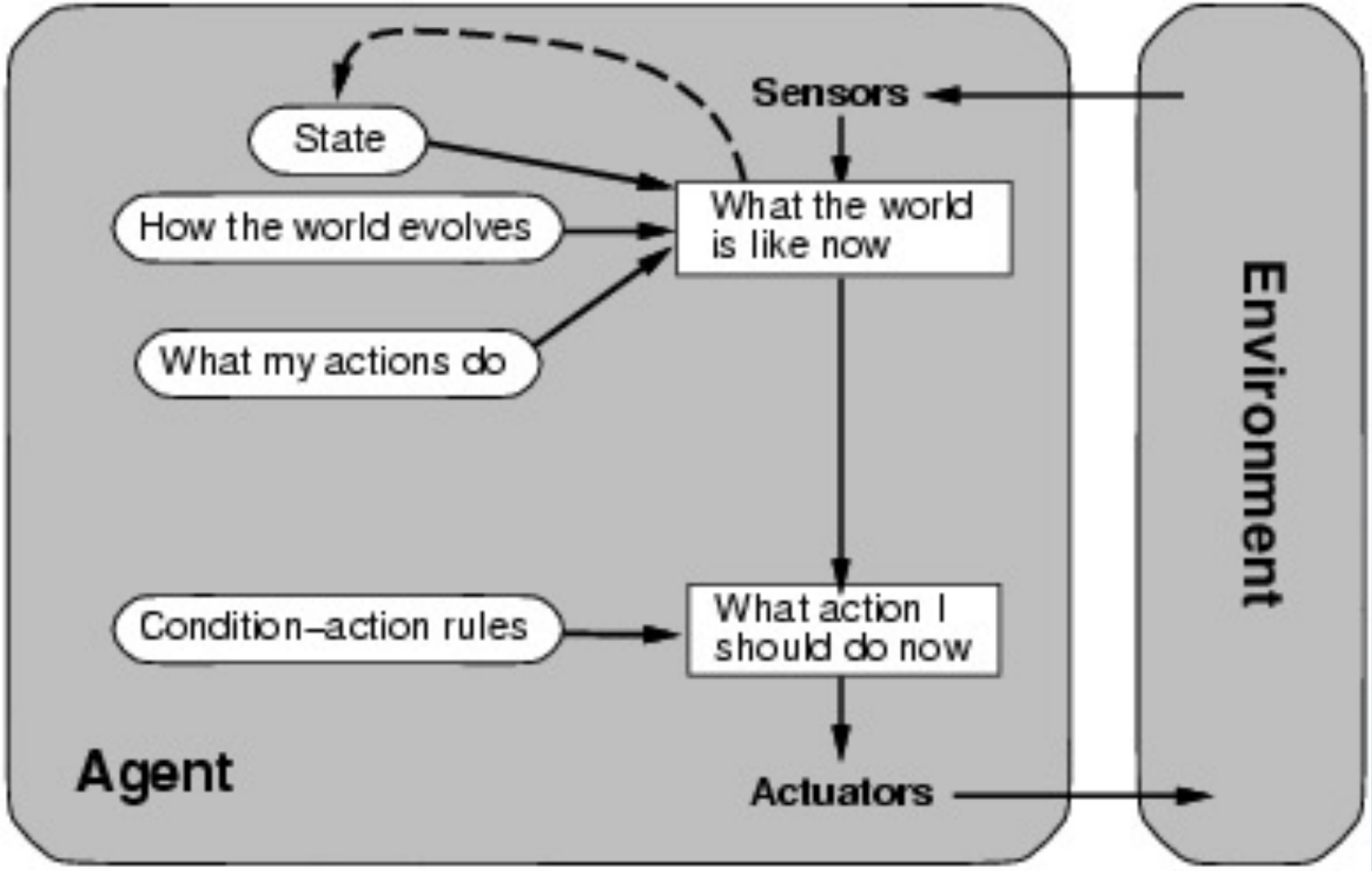
Testing an AI

- We do not just want one environment but an environment class
  - Run multiple simulations with many different changes to the environment
- Self-driving Car
  - Traffic level
  - Lights
  - Weather
- Card Game Player
  - Against many other desks
  - Other player agents

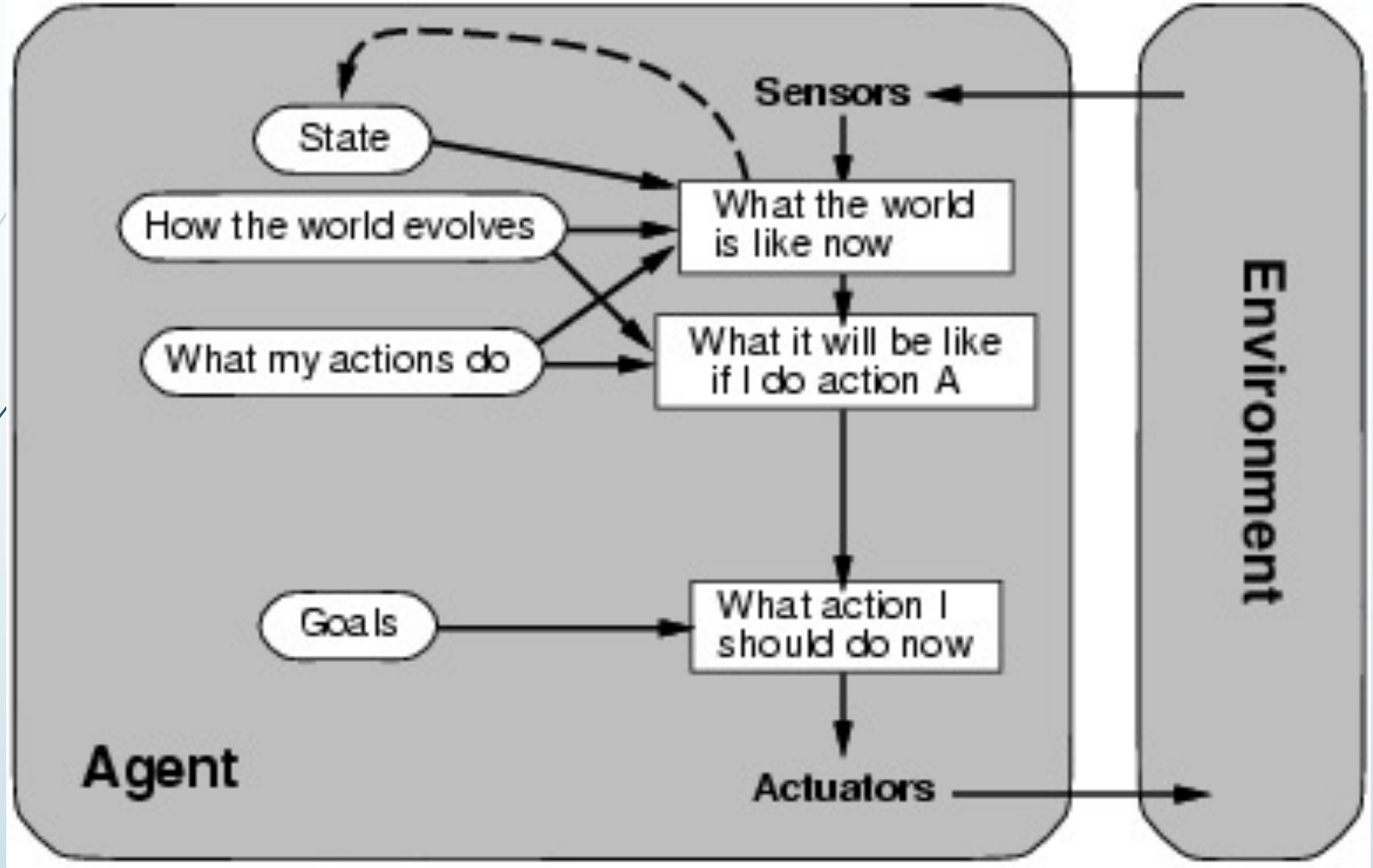
Simple reflex agents



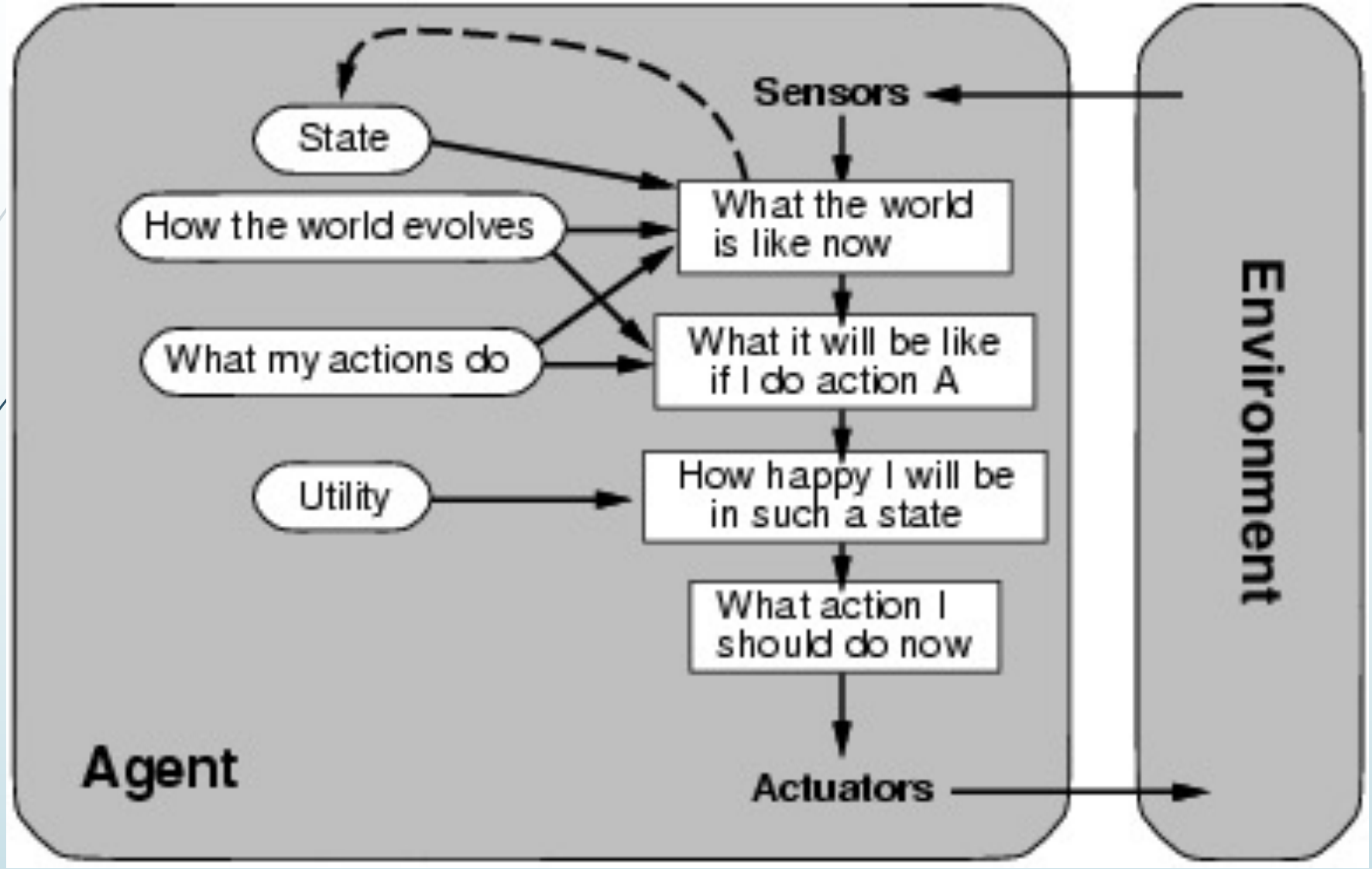
Model-based reflex agents



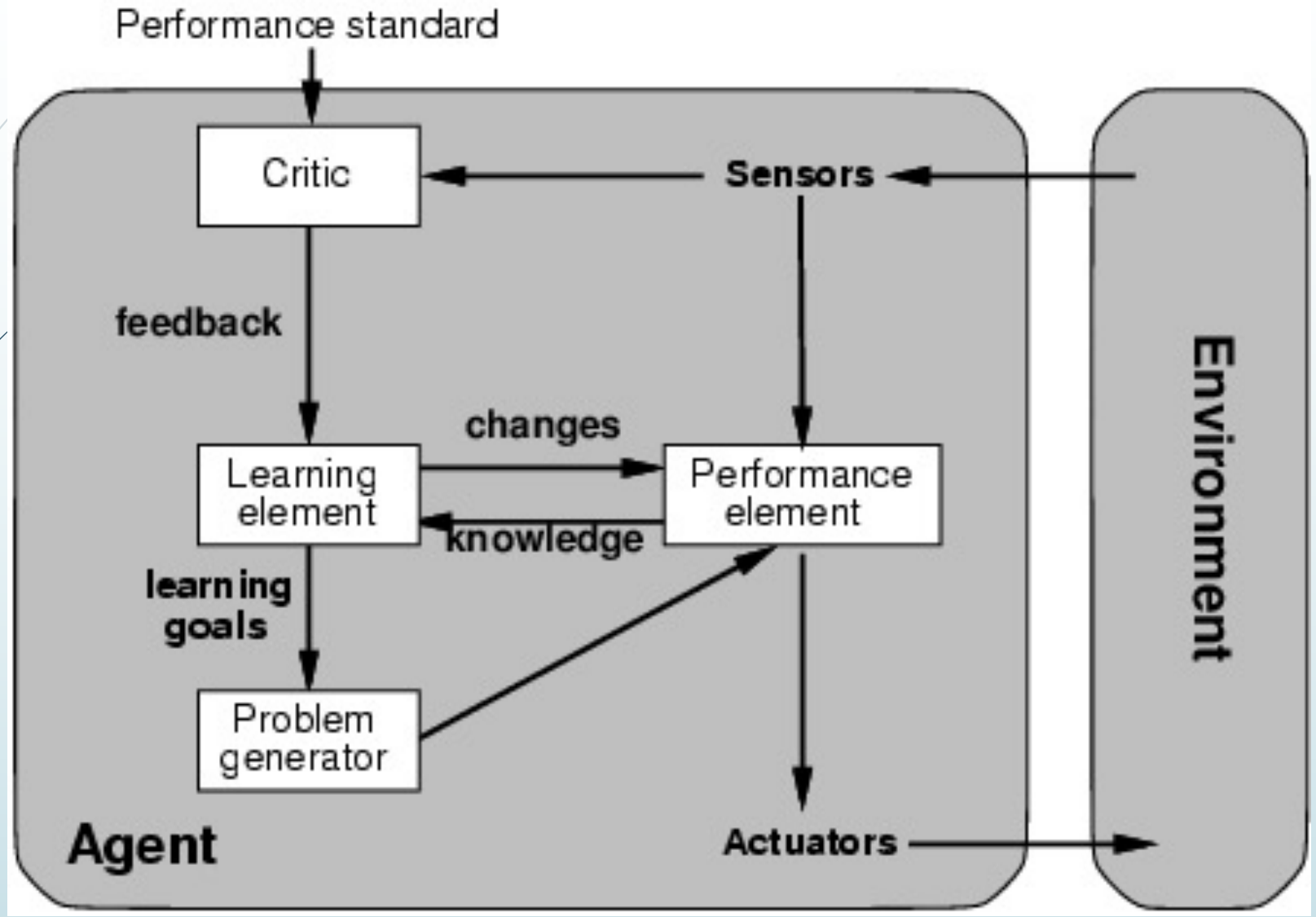
Goal-based agents



Utility-based agents



Learning agents



Introduction to Artificial Intelligence

Week 4

Learning by Searching

Your Car is Lost in a Carpark

- How would you find it?
- Walk at random
  - No guarantee you will find it
  - Could be lost forever
- Walk down each row
  - Guarantee you will find it
  - Might take a long time
- Walk to the place you think it was last, then expand your search
  - Guarantee you will find it
  - As long as your memory is good – on average it will be less than every row, worst case just as good
- Walk to the place you think it was last, hit the alarm button, then expand search
  - Guarantee you will find it
  - Alarm can be heard from a long way away, and if you press it to no effect the car is not local to you (radius of activation)

Search Space

- Many problems define a set of actions on an environment which can be seen as a search space of states
  - We are at one point in the environment with a destination, there is a list of action states which will bring us to the destination
- Each search type is a method of navigation of the space
- We will rarely want to go to the same point in the search space
  - Represents a cycle in the search, which are not useful in terms of state spaces
- Solution – most searches will avoid this by using tree structures
  - Each node in the tree is a state within the search
  - Search Trees, Game Trees

Types of Searches

Uninformed Search

- Have no sense of the problem domain
- Generally applicable in all cases

Informed Search

- Use a heuristic function developed for the domain
- Applicable in their own domain

Goal-Based Agent

- Four Steps in the Agent:

1. Define Goals
   - What are the states which are considered to be satisfying the goal?
2. Problem Formulation
   - What actions are available to move towards the goal?
   - What states are available to move towards the goal?
3. Search
   - Determine the pathway of states in order to meet with the goal
4. Execute
   - Move though the series of states

Example: Painting Robot (simplified)

1. Goal
   - All show surfaces of the part are covered in 1mm of paint
2. Problem Formulation
   - A sequence of points about the part for the robot arm to move into in order to ensure a dispersal about the entire part
3. Search
   - Examine all pathways about the part, taking into account the movements on each part of the arm
   - A pathway is good as long as it covered the part with 1mm of paint
4. Execute
   - The arm moves though the sequence of positions

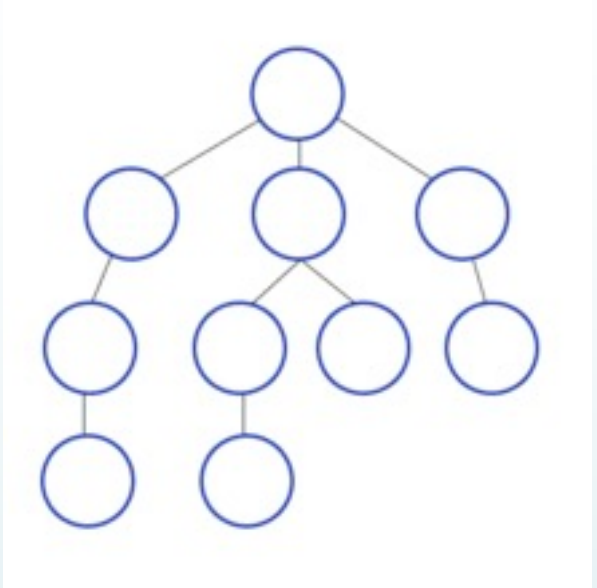
State Space of the Robot

- State space is defined as the location of all moving elements on the robot
  - Relative positioning on all joints from a home position
  - Might also include the rate of spray on the nozzle – from 0 to full
- Transitions of states could be broken down into a sequence of movements in the joints on each rotational axis (hence a 6-axis robot)
  - Move(Elbow, X degrees)

N Queens and Incremental Goals

- Place N queens onto an NxN chess board such that no queen attacks any other queen; a more complicated problem is a Costas Array
- States
  - Place a queen down
- Goal
  - No queen attacks any other queen
  - Hard sometimes to see a good pathway to the goal
- Need a measure in order to see a good pathway
  - Minimize number of queens attacking

Tree Search

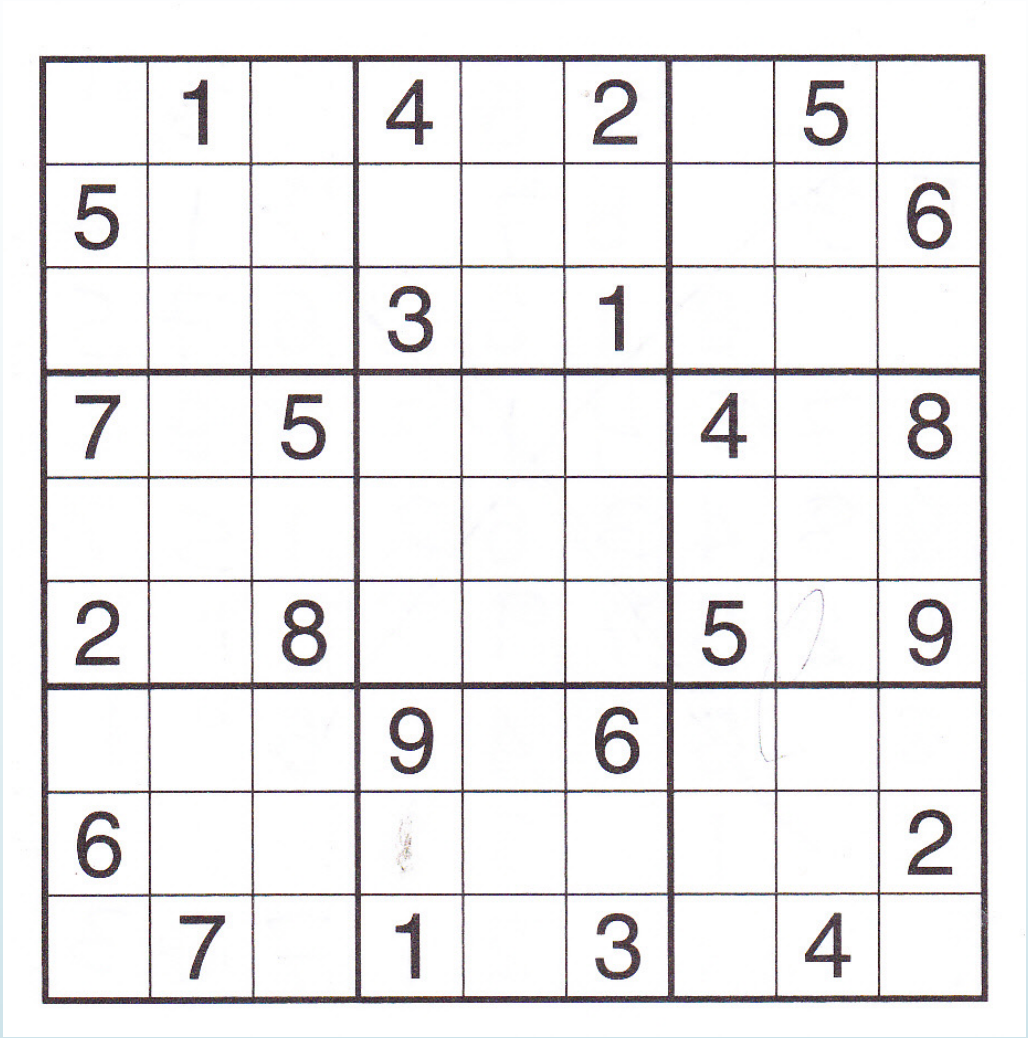


- Recursive definition of a tree search as DFS
- SearchTree(State, Move, visited list)
  - Apply Move to the New State
  - If (New State == goal) return TRUE
  - Else
    - If (neighbouring states are empty) return FALSE
    - For each of the MOVE on neighbouring states not in visited list
      - If SearchTree(Neighbouring states, MOVE, visited list + State) return TRUE;

Backtracking Searches

- Move towards the goal, if you reach a position which is going to be less successful – stop and move back up the tree (backtrack, retrace your steps)
- Recursive definition of a depth-first search of a backtrack:
- SearchTreeBacktrack(State, Move, visited list)
  - Apply Move to the New State
  - If (New State == goal) return TRUE
  - If (New State causes an invalid path to goal/or costs too much) return FALSE
  - Else
    - If (neighbouring states are empty) return FALSE
    - For each of the MOVE on neighbouring states not in visited list
      - If SearchTreeBacktrack(Neighbouring states, MOVE, visited list + State) return TRUE;

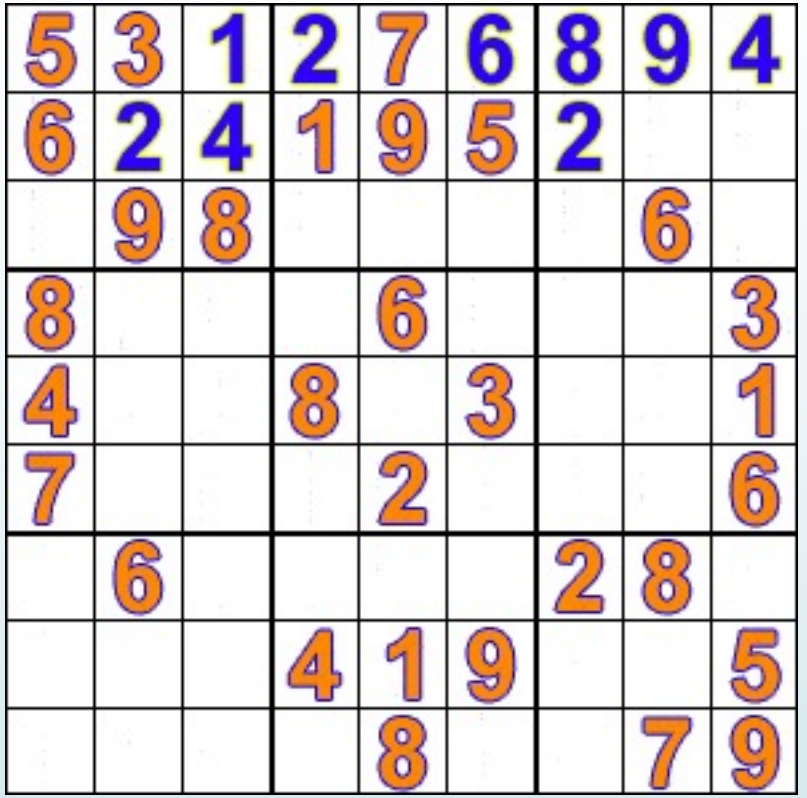
Sudoku



	1		4		2		5	
5								6
			3		1			
7		5				4		8
2		8				5		9
			9		6			
6								2
	7		1		3		4	

Backtrack on Sudoku

- isValid
  - Check if we have not violated the rules
    - 1-9 in each row
    - 1-9 in each column
    - 1-9 in each box
- Choice at each empty box 1-9
- Multiple Solutions?



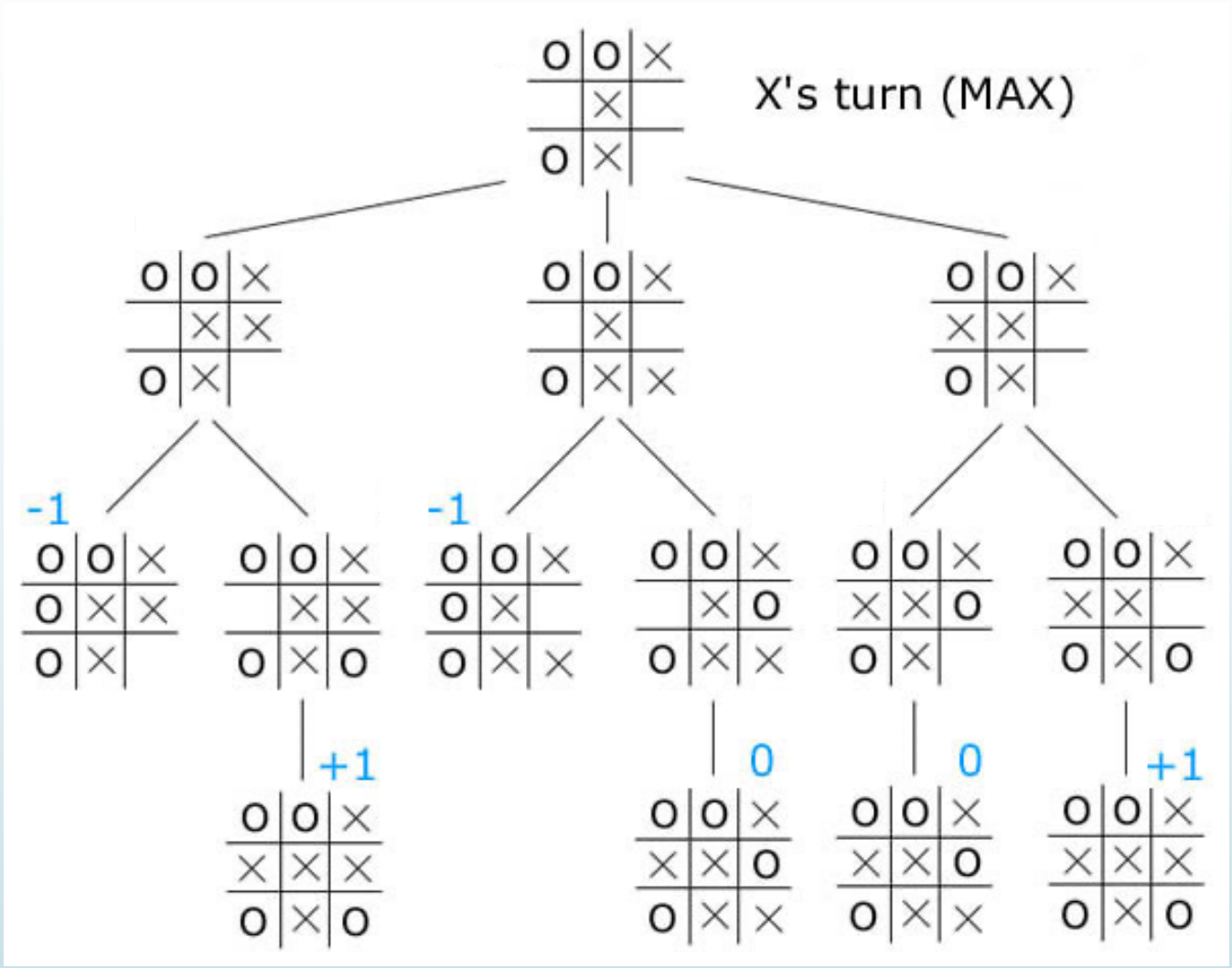
Heuristic Search

- Word comes from the Greek for discovery
- Heuristics use an educated guess on where to move the search next – rules-based movement about a search space
- Not necessarily deterministic – may have stochastic generation methods
- Trial and Error
  - Experimental and Experiential
  - Does not guarantee the most optimal solution
- Examples
  - Greedy and Hill Climbers are heuristics

Minimax

- Opponent goal is to reduce winnings (MINI-Move) and player goal is to increase winnings (MAX-Move)
- Game with a scoring system or win lose condition which can be evaluated
  - GO
  - Chess
- Game Tree
  - Selection of plays in a turn-based game can be reduced to looking at a pathway on a tree
  - Used as part of the theoretical foundations of GAME THEORY

Minimax Example

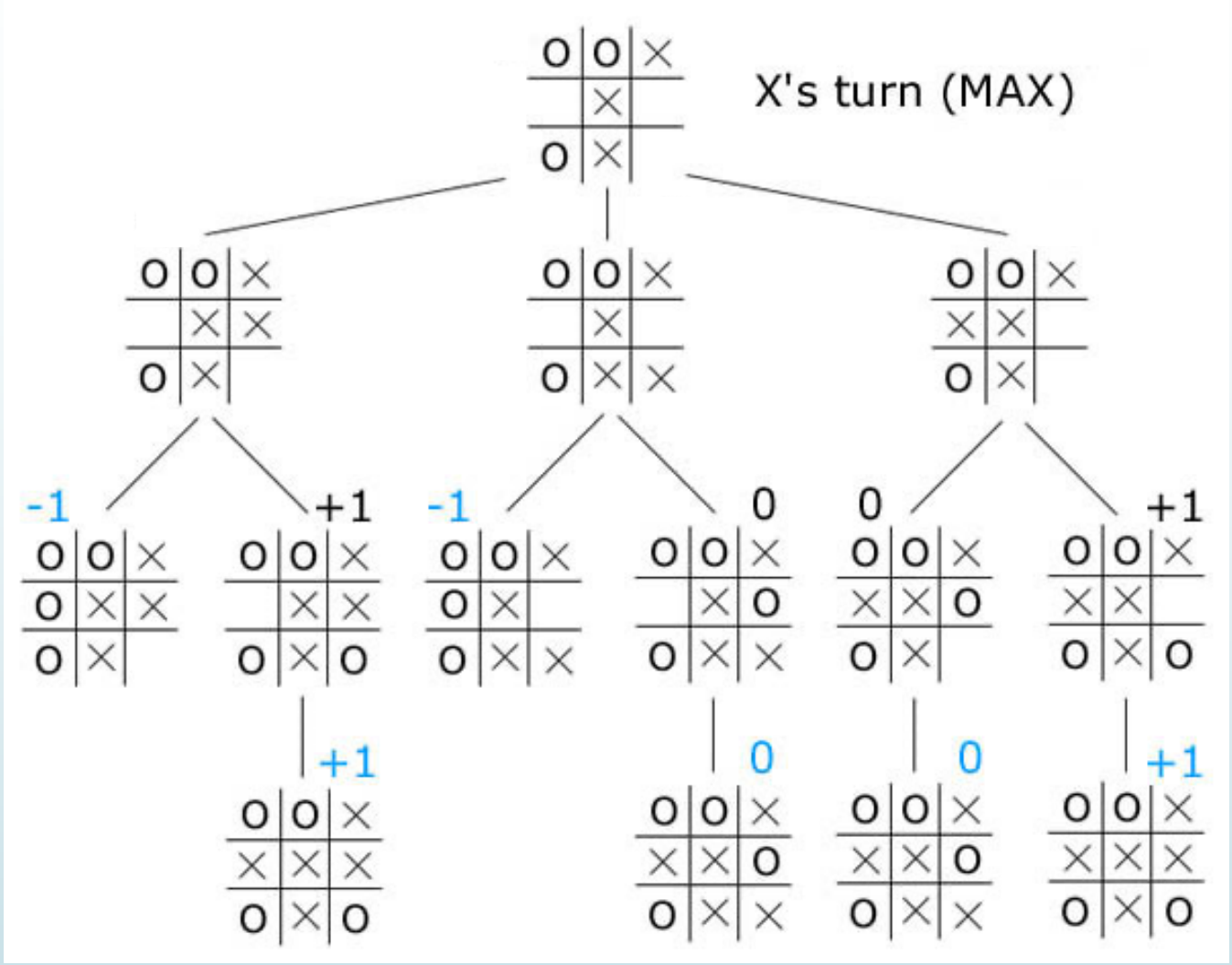


X's turn (MAX)

0's turn (MIN)

X's turn (MAX)

Minimax Example

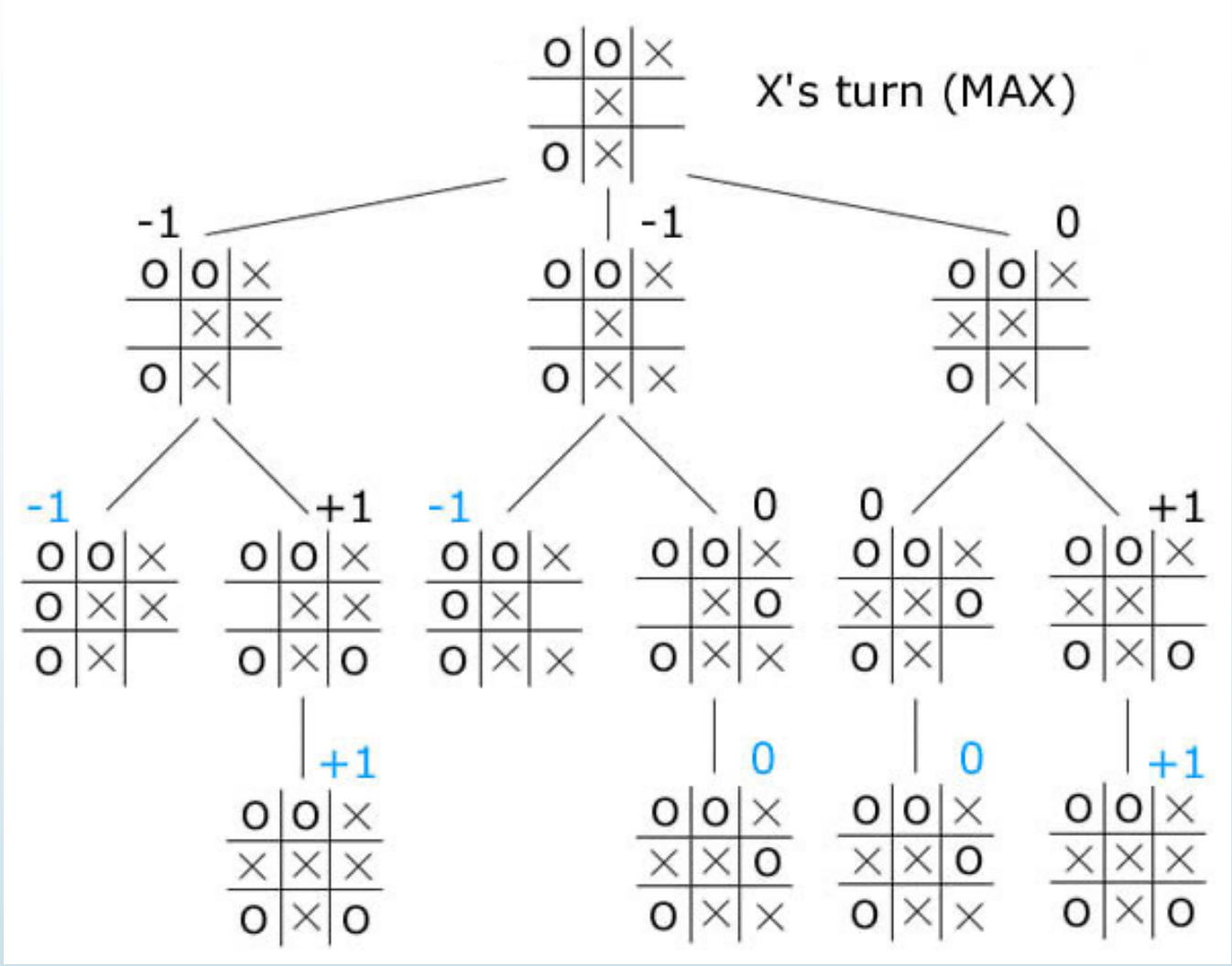


X's turn (MAX)

O's turn (MIN)

X's turn (MAX)

Minimax Example

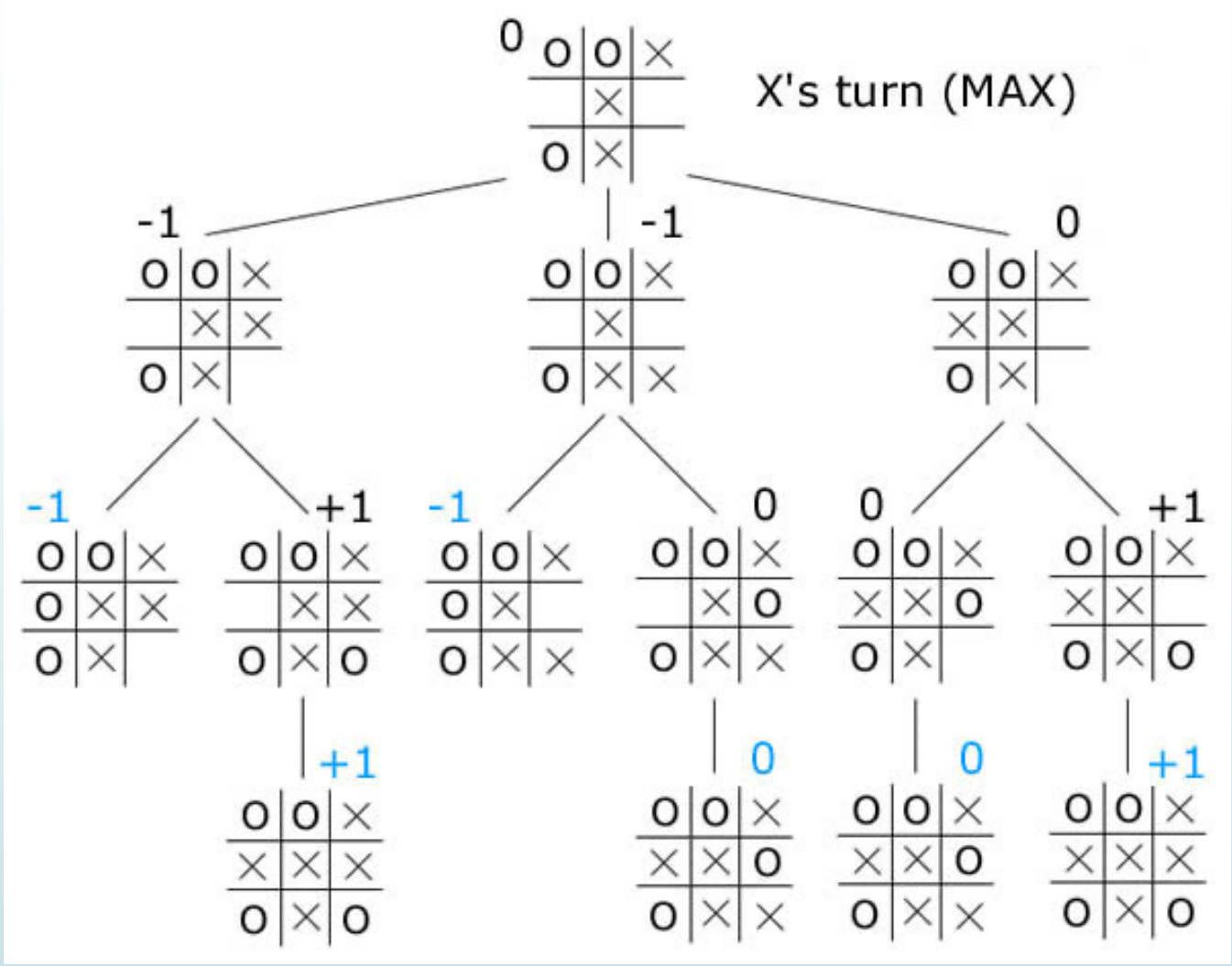


X's turn (MAX)

O's turn (MIN)

X's turn (MAX)

Minimax Example



X's turn (MAX)

O's turn (MIN)

X's turn (MAX)

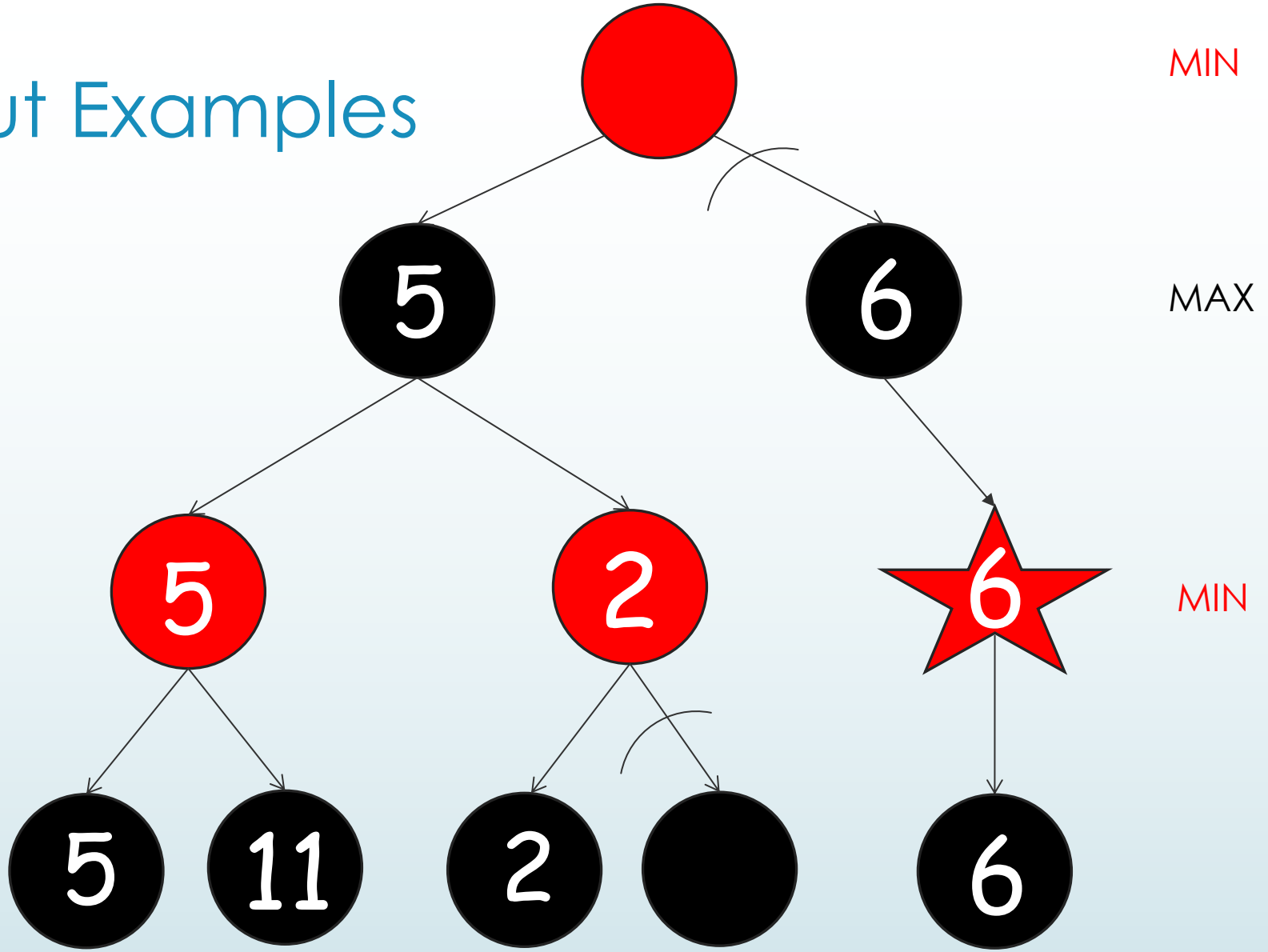
Backtracking

- Simulate the entire game
- Assume rational/perfect play from the opponent
- Theoretically there is a perfect game play method if we search the entire game
- Solved Games
  - Tic Tac Toe
  - Checkers
  - Two player heads-up limit poker – Recent discovery by researchers in University of Alberta
- Unsolved
  - Chess
  - Go

Backtracking with Alpha Beta Cuts

- Heuristic search based on backtracking
- Used in situations where there is a value to the solution at each point to allow for an evaluation
- Applied for Game Trees
- Alpha Cut
  - Maximum lower bound (Max Plays - Black)
- Beta Cut
  - Minimum upper bound (Min Play - Red)

Cut Examples



Introduction to Artificial Intelligence

Week 5

Learning by Searching

Types of Searches

Uninformed Search

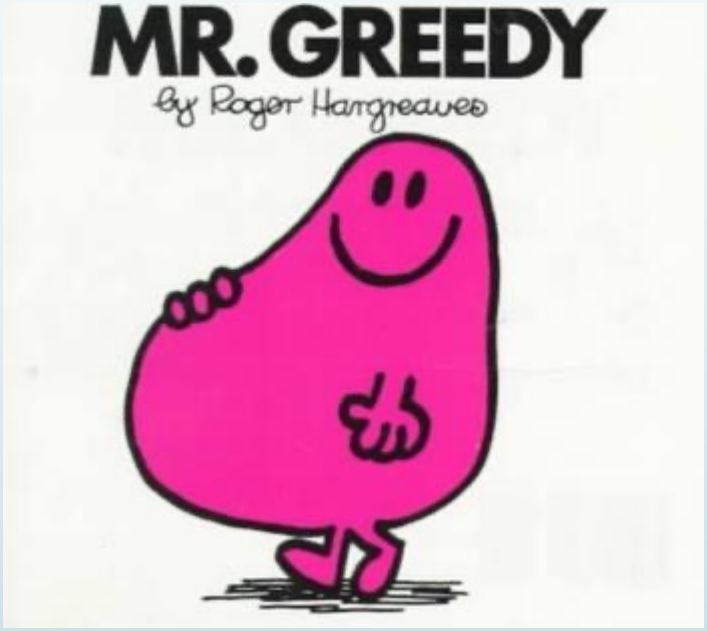
- Have no sense of the problem domain
- Generally applicable in all cases

Informed Search

- Use a heuristic function developed for the domain
- Applicable in their own domain

Greedy Approaches

- Generate and Test Methods primarily
- Current best solutions are held
- When I find something better – I get rid of what I had
- Always wanting more
- Always takes the
  Locally Optimal choice

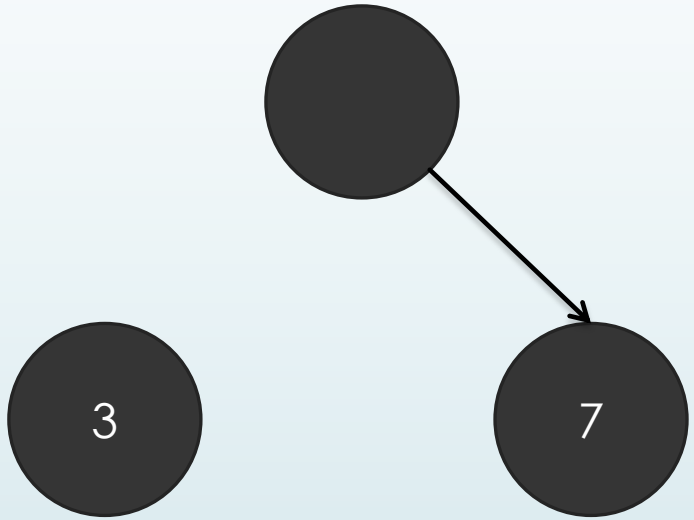


Greedy Algorithm Basics

- Start with a search space from which a candidate solution will be taken from
- There is a number of decisions to be made in sequence to find a solution, at each step make a choice which maximizes the outcome
- Done

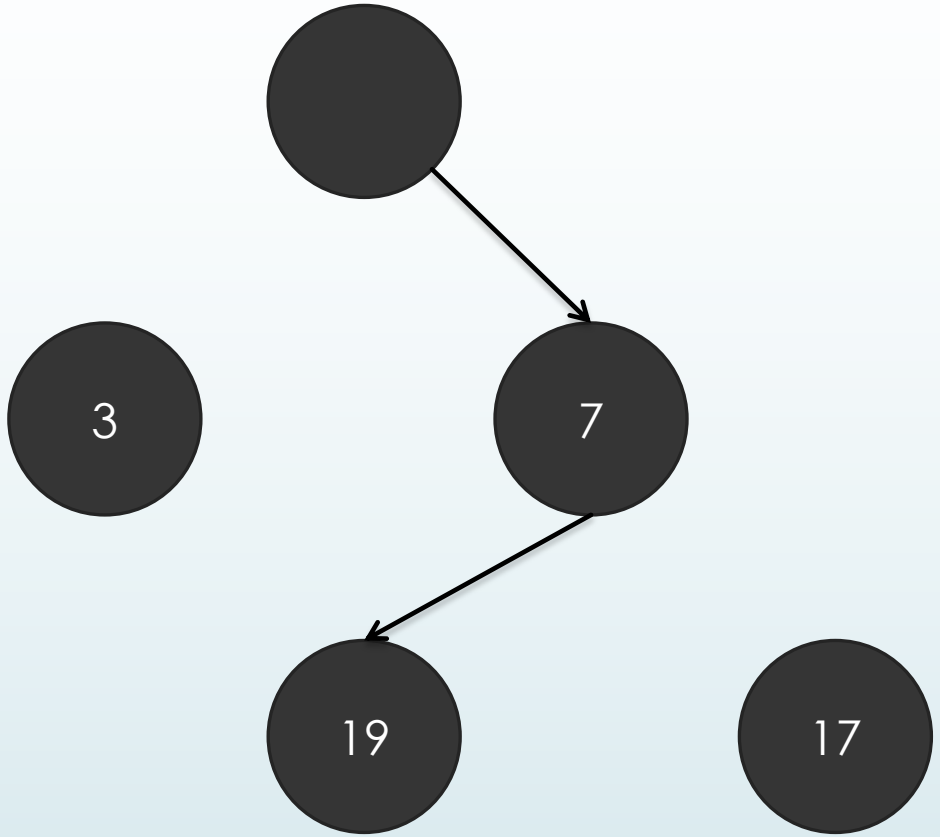
Example – Game Tree

- Two players are making moves in a game, each move gives a certain number of points to the player – goal is to make the most points



- In the first move our greedy player will take 7 points

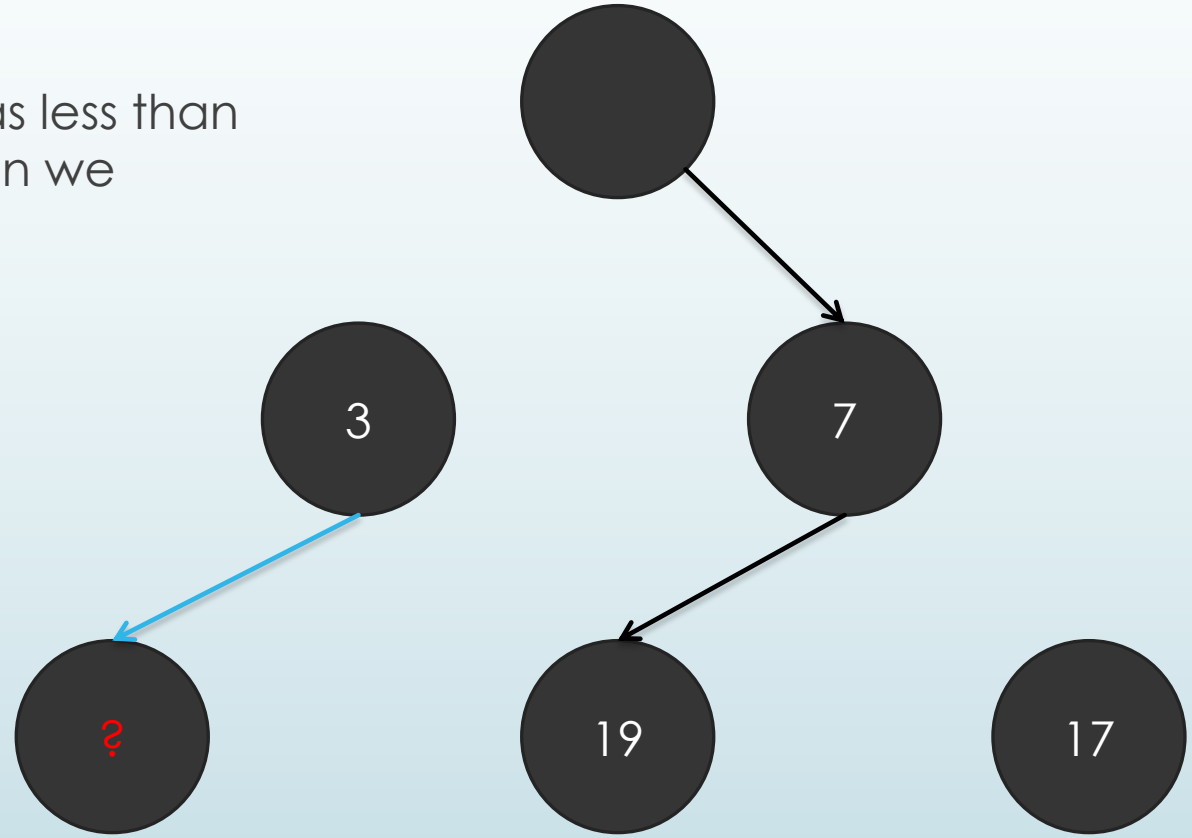
Example – Game Tree



- In the second move move our greedy player will take 19 points, for a total of 26 points

Was Greedy Good

- Maybe – Depends on the Game being played
- If the ? Position was less than or equal to 23, then we have maximized the score
- If ? was greater, then we have failed



What Ensures That Greed Will Be Rewarded

- Sub-problem optimization
  - Also called optimal substructures
  - The optimal solution to the problem contains optimal sub-problems
- Greedy Choice Property
  - We never need to consider the play until now and can move from this point only by making the current best decision. We never need to review the past moves in order to make a better decision
  - In our example, if we have played the optimal game until now, then the current point is also part of the optimal value on this move

Activities Not Clashing Problem

- We have a set of activities each with a start and end time
- We want to find the maximum number of activities that a single person can complete in the time allotted

Example

- One Operator on one machine
- Assume all parts give the same profit margin, but we have a choice of which car to build now
- Times are dependent on assembly

Greedy Algorithm

- Sort the activities by their end time from smallest to largest
- Start with Set of Activities *S = {1}*, and finish time *f = f[1]*
- For all the other activities *X* in order *I*
  - If s[x] >= f
    - S = S U X
    - f = f[x]

Why it Works?

- We order *S* from smallest to largest finish time, so *s[1]* finishes first
- Let *A* be an optimal subset of *S* and ordered by increasing finish time which does not contain *S[1]*, let *S[k]* be its first member

Why it Works? Cont…

- Then we can construct a set $B = (\{A \setminus S[k]\} \cup S[1])$, as $f[1] < f[k]$, then the activities are disjoint, note that $|B| = |A|$, therefore both are optimal
- Once this choice is made of $s[1]$, then the remaining disjoint parts of the set reduce into a smaller instance of the same problem
- We always make the right greedy choice by picking the first finishing task – note their might be other optimal solutions – we have at least shown one construction

Expand on the Problem

- However, not every part costs the same profit margin
- We are interested in profitability – not in number produced
- How do we maximize our profits?

Greed is No Longer Good

- The greedy solution is based on the number of completed parts solely
- If producing a smaller number of parts has a higher profit, or if there are multiple maximum part solutions with different profits, we no longer have the optimal solution
- We no longer have the greedy choice property!

Generate and Test

- Greedy is a constructive solution method – we construct the solution one part at a time
- Method to produce a solution may also generate the entire solution in one step; non-constructive
- Generate the problem solution
- Test the solution against the problem, or some representative amount of the problem

Hill Climbers

- Single Searching point
- Search locally for changes to the actions
- Have a representation of a solution string
- Have a test of the representation about the space
- Each solution string maps to an outcome
  - An objective function
  - A fitness on the problem

Climbing Hill in Our Example

- Representation
  - String of actions which we can take
- Objective
  - Maximization of profit value based on the actions
  - No string of actions is allowed to have an overlap
- Local Change
  - Add and/or remove some number of items from the string
- Climbing the Hill
  - Replace my position if the change is better

Hill Climbing Algorithm

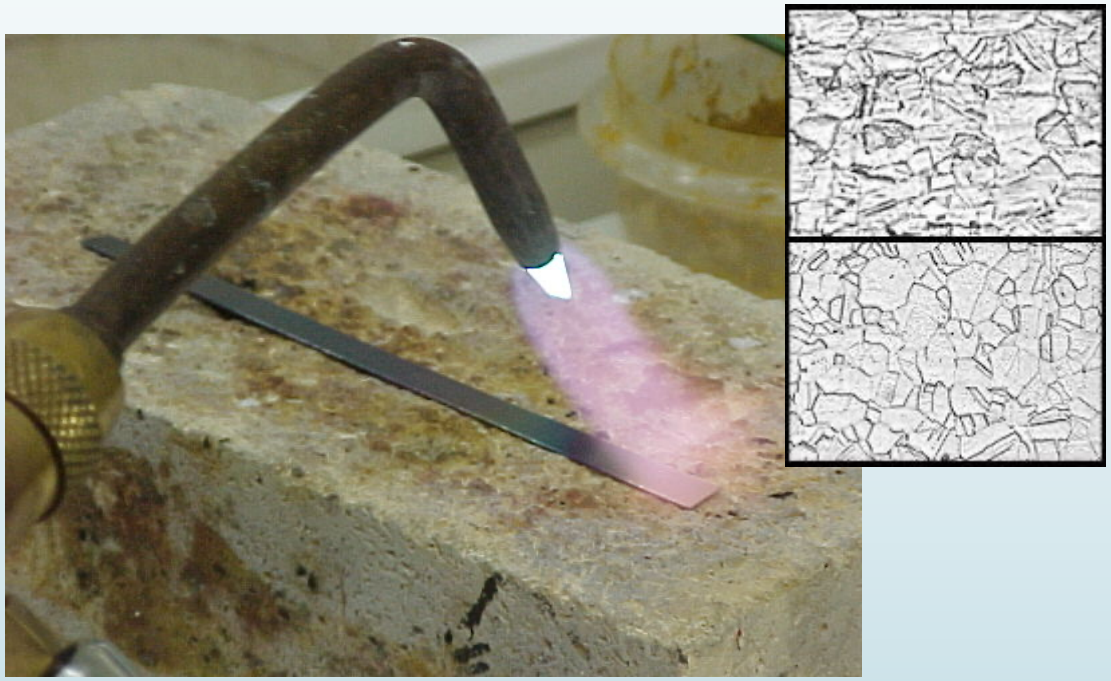
```
function HILL-CLIMBING(problem) returns a solution state
   inputs: problem, a problem
   static: current, a node
           next, a node

   current ← MAKE-NODE(INITIAL-STATE[problem])
   loop do
      next ← a highest-valued successor of current
      if VALUE[next] < VALUE[current] then return current
      current ← next
   end
```

Issues and Work About

- Local Maximums
  - Stuck in suboptimal places if our value is on a localized hill
    - Change the mutation
    - Change the representation
    - Random restart
- Multiple Optimums
  - Multiple Good Solutions
    - Random Restart
  - Large areas of equal solutions
    - Change the mutation
    - Random Restart
- Jumping over areas of the space
  - Selected Mutation hops over the space

Simulated Annealing

- Based on the properties of Metal working
- Metal is heated and cooled in order for it to be made workable



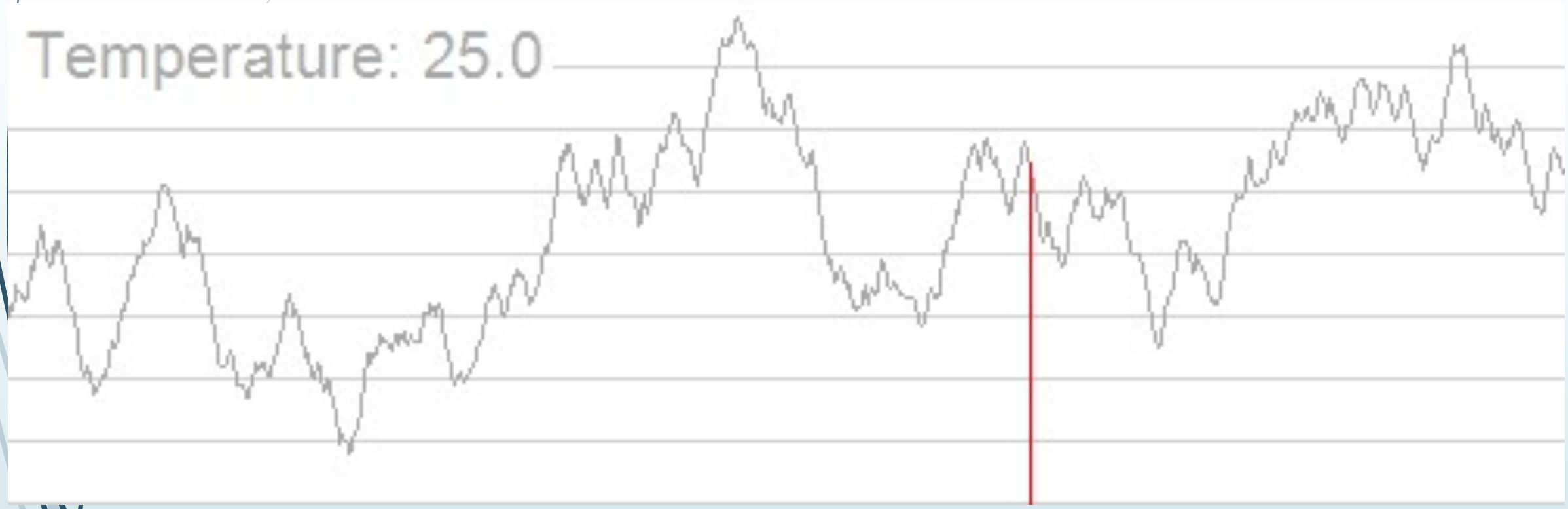
Simulation of This Process

- Set down a number of search points
- Points move dependent upon the temperature applied
- They move about the space based on a temperature which is slowly reducing over time
  - This allows for large movement at the beginning of the algorithm and reduces into smaller and smaller movements as time goes on
- Two Types of controls
  - Temperature controls size of step
  - Temperature controls the acceptance of a step

Simulated Annealing – Step Size

- Start at an initial temperature value (usually something hot) and create a random initial solution
- Until temperature is cool
  - Make a change to the current solution (neighbour) based on the temperature
    - Current location + Rand_Distribution(temperature)
  - Decrease the temperature

Temperature Controls the Step Size



Simulated Annealing

- Start at an initial temperature value (usually something hot) and create a random initial solution
- Until temperature is cool
  - Make a small change to the current solution (neighbour)
  - Decide if we move to this neighbour solution
    - Neighbour solution is better or
    - Rand_Distribution() < temp
  - Decrease the temperature

Exploring and Exploiting

- Exploration – the want for a search method to find new areas of the search space which have not been visited yet
- Exploitation – the want for a search method to locally search about the best known areas in the space in order to refine these current solutions
- All search algorithms will have some mix of both properties
- Note that if we only have a limited number of objective tests available, then we have a zero-sum game – we can explore, or we can search with a test

Exploration or Exploitation?

- We create a simulated annealing which has a reheat if we are not seeing a 10% improvement in the solution

Exploration

- The reheat increases the step size
- The reheat increases the probability we accept worst solutions

Exploration or Exploitation?

- After the application of a search we look at the least significant digit and test from 0-9 taking the best final solution

Exploitation

- Looking at the least significant digit we are looking in a close locality to the presvious, making the assumption that close parameter settings make for close outputs

Parallel

- All of these methods are trivially parallel
- We can place one of these searching units onto a different ‘core’ processing unit without issue
- Hill Climbers and Simulated Annealing use the idea of a solution neighbourhood, so we can divide the space

A* Search

- Fast well-established algorithm for game AI
  - Pathways
  - Movement of NPC
  - RTS movement
    - Usually one member of a flock
- Works in areas where we do not know the connectivity
  - Makes it very suitable for PCG – because we do not need nodes defined by the generator – but can still understand the measurements
- Can take into account multiple different connection topologies and styles of movement
- Large number of iterative improvements and problem domain specific implementations with heuristics
- $f(n) = g(n) + h(n)$
  - $g(n)$ is the path from the initial position to the current node
  - $h(n)$ is the path from the current node to the goal position

A* Algorithm

- A node contains a location, a cost, and a parent
- Initialize a set CLOSED to be null
- Initialize a priority queue based on cost of a node called OPEN to be the starting location, set cost = 0
- While OPEN is non-empty do
  - Current <- Pop OPEN.top
  - For all neighbouring locations X
  - If not (Blocked or Closed)
    - If not in OPEN
      - X.cost = current.cost+1
        X.parent = current
      - Push X to OPEN

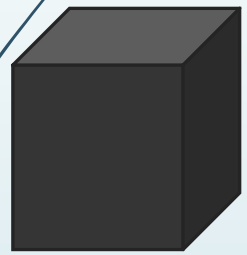
A* Algorithm

- Else if X is in OPEN
  - If current.cost+1< Open.x.cost // we found a better path
  - Open.x.cost = current.cost+1
  - Open.x.parent = current
- END FOR ALL X
- Push Current to CLOSED

A* for PATH FINDING

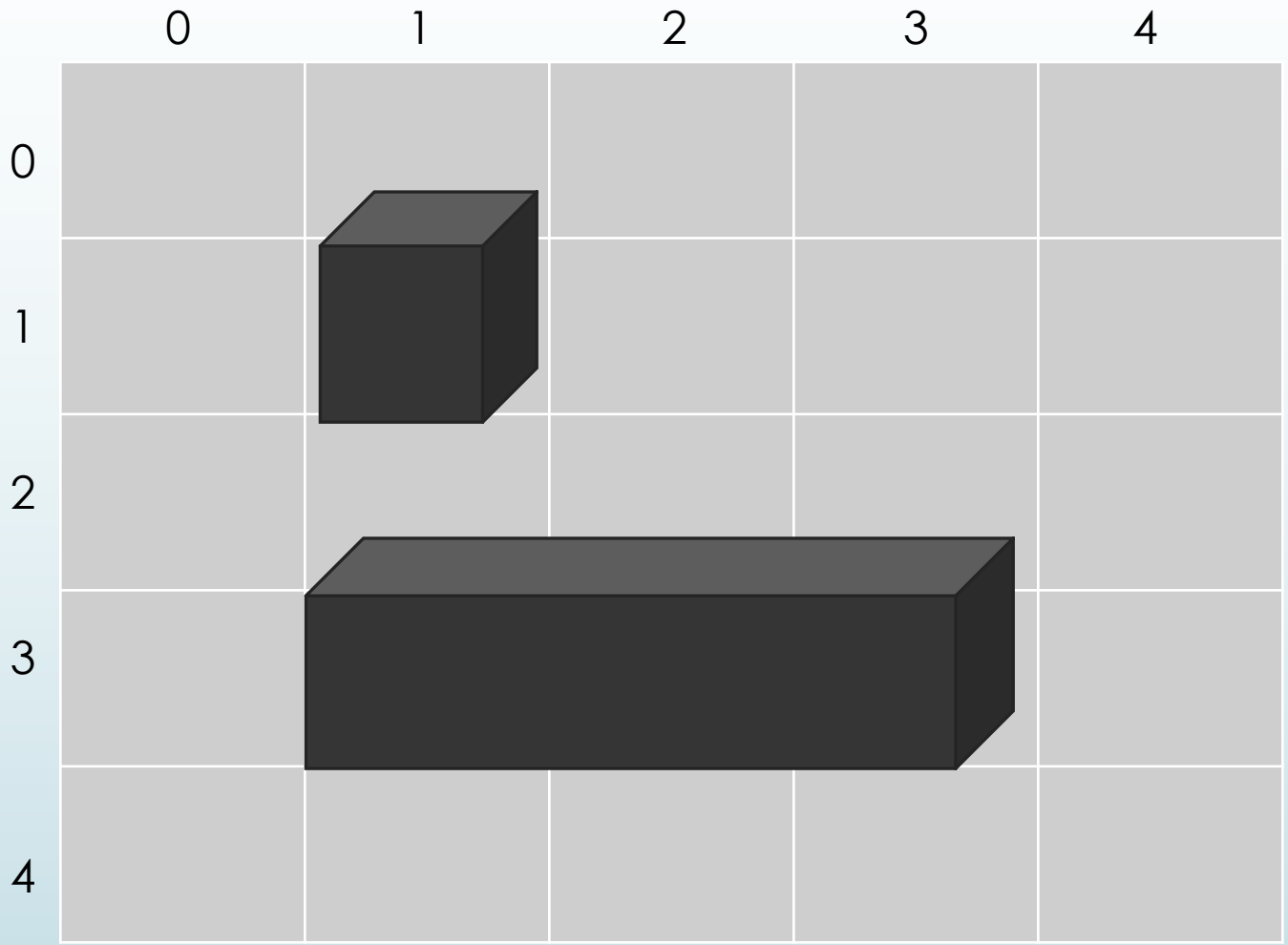
- REPORT_PATH (START, FINISH)
  - STACK PATH initialized to NULL
  - RUN A* (START)
  - Find FINISH in CLOSED
  - Current <- FINISH
  - While Current not START
    - PUSH Current to PATH
    - Current <- Current.parent
  - PRINT START
  - While STACK not NULL
    - PRINT POP PATH

World

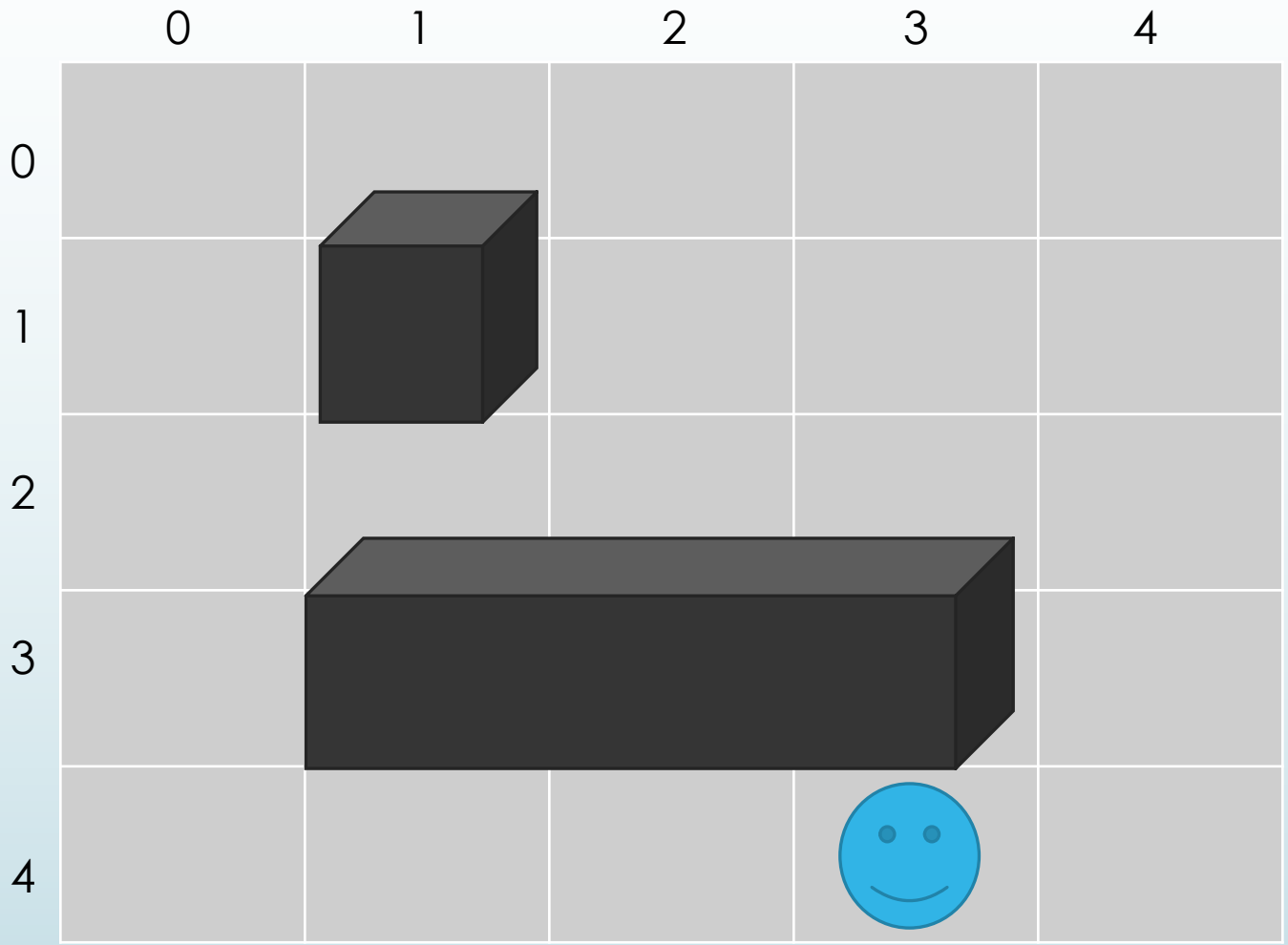


	0	1	2	3	4
0					
1					
2					
3					
4					

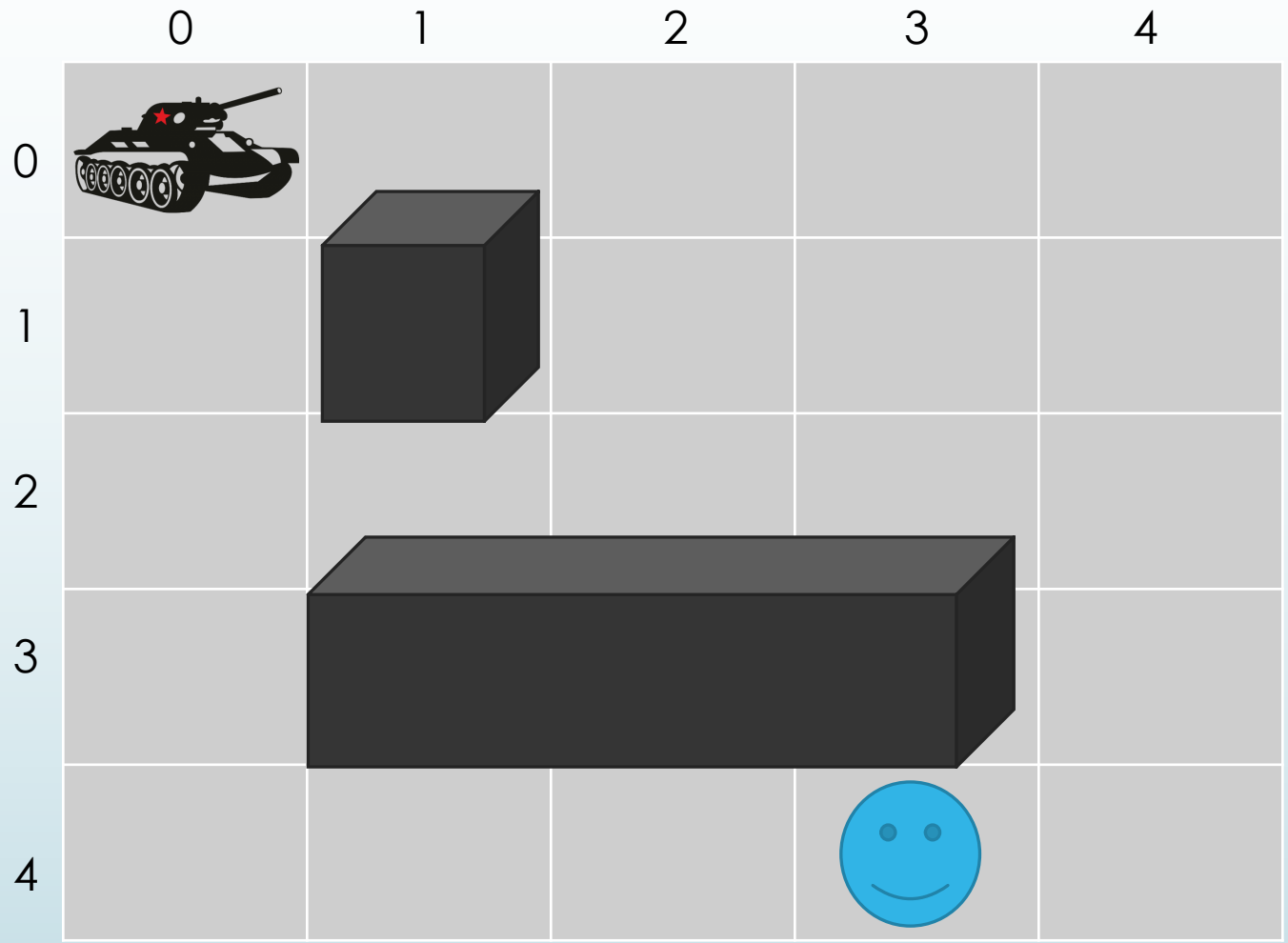
World With Obstacles



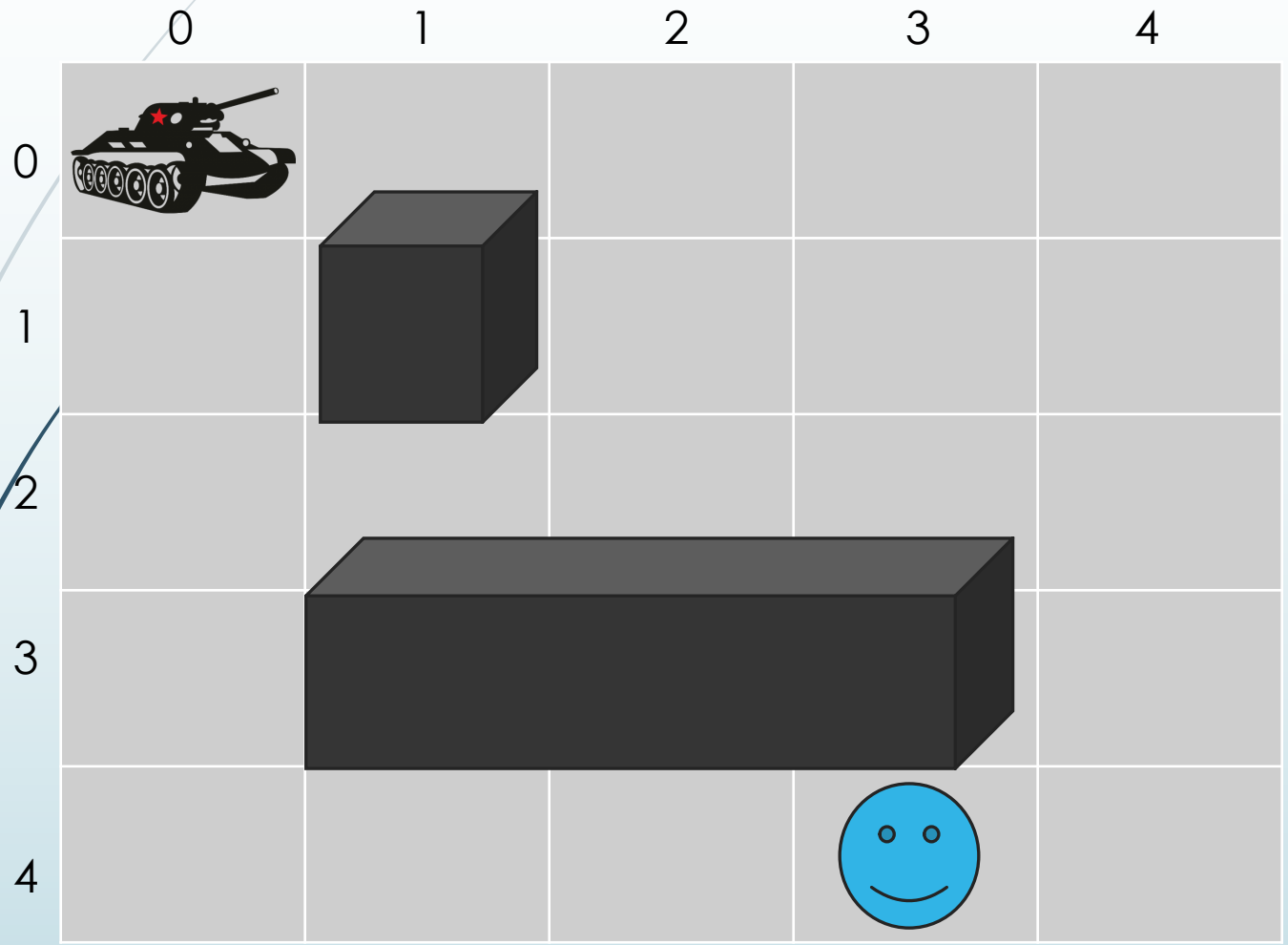
World With Obstacles and Goal



World With Obstacles, Goal and Start Position



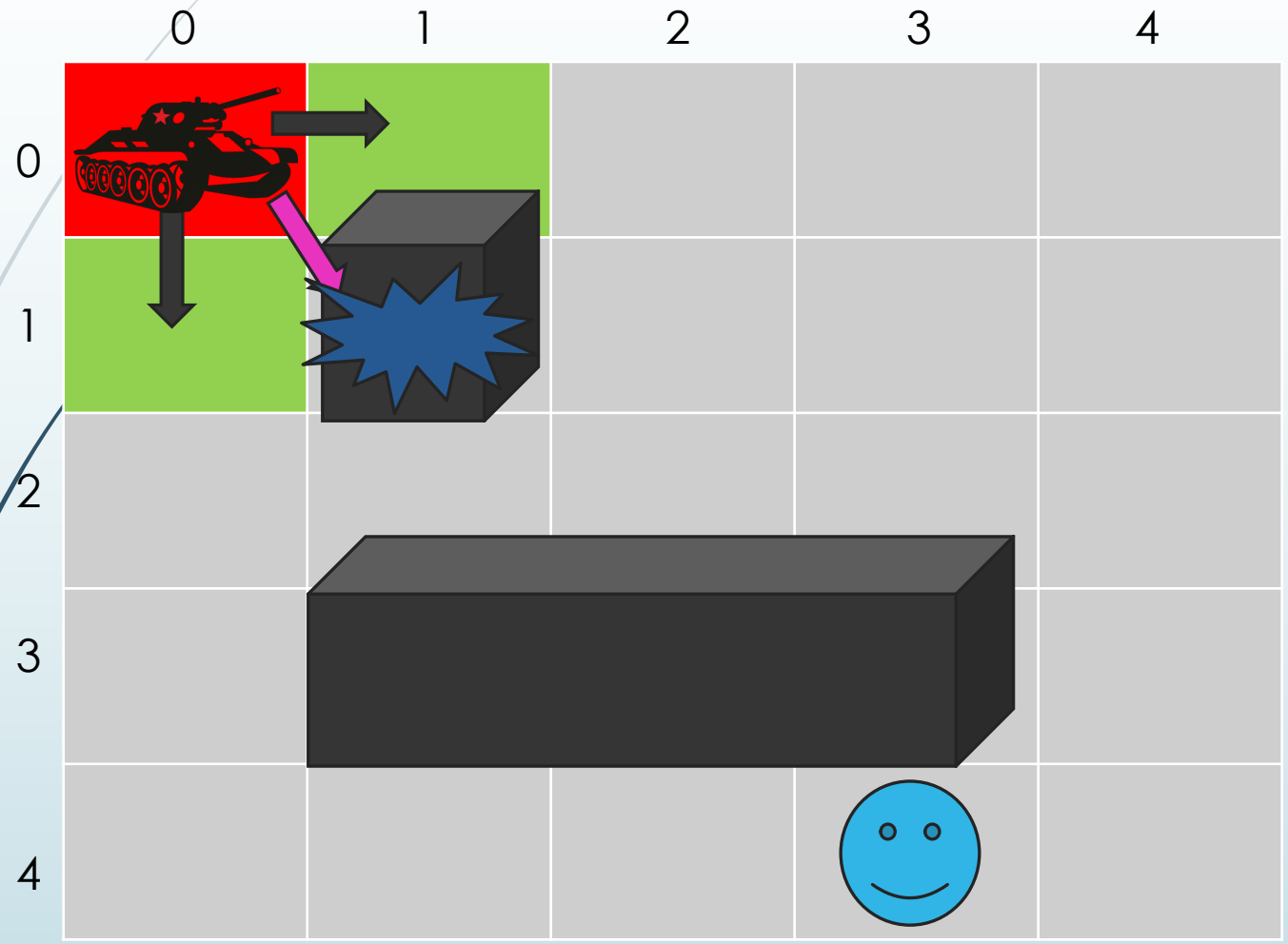
Add Initial Point to Open List and Assign Cost



CLOSED:
NULL

OPEN:
(0,0)-0

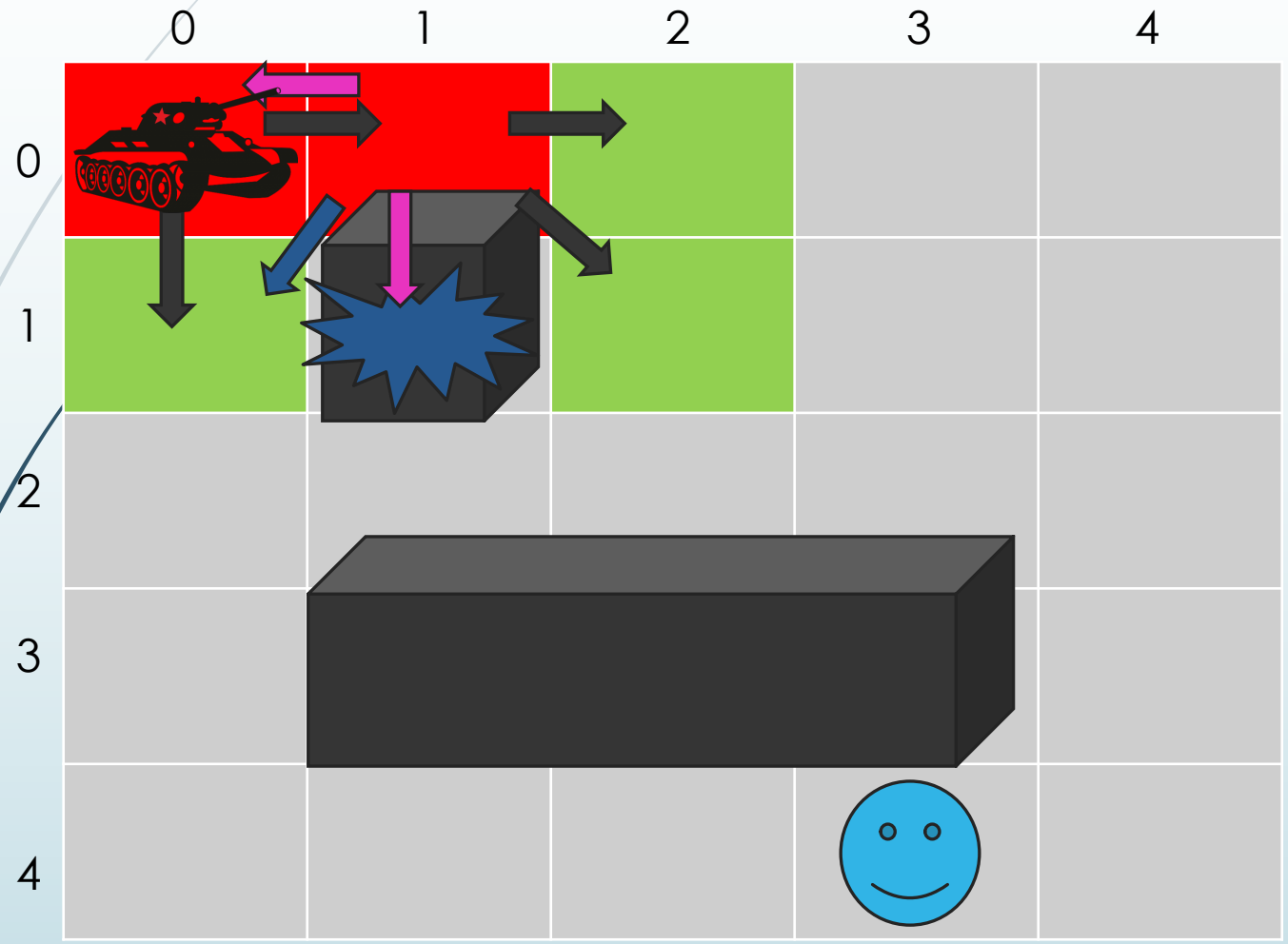
Move Lowest Cost to Closed and Search Neighbours



CLOSED:
(0,0)-0-NEW

OPEN:
(1,0)-1-NEW
(0,1)-1-NEW
(1,1)-COLLISION!

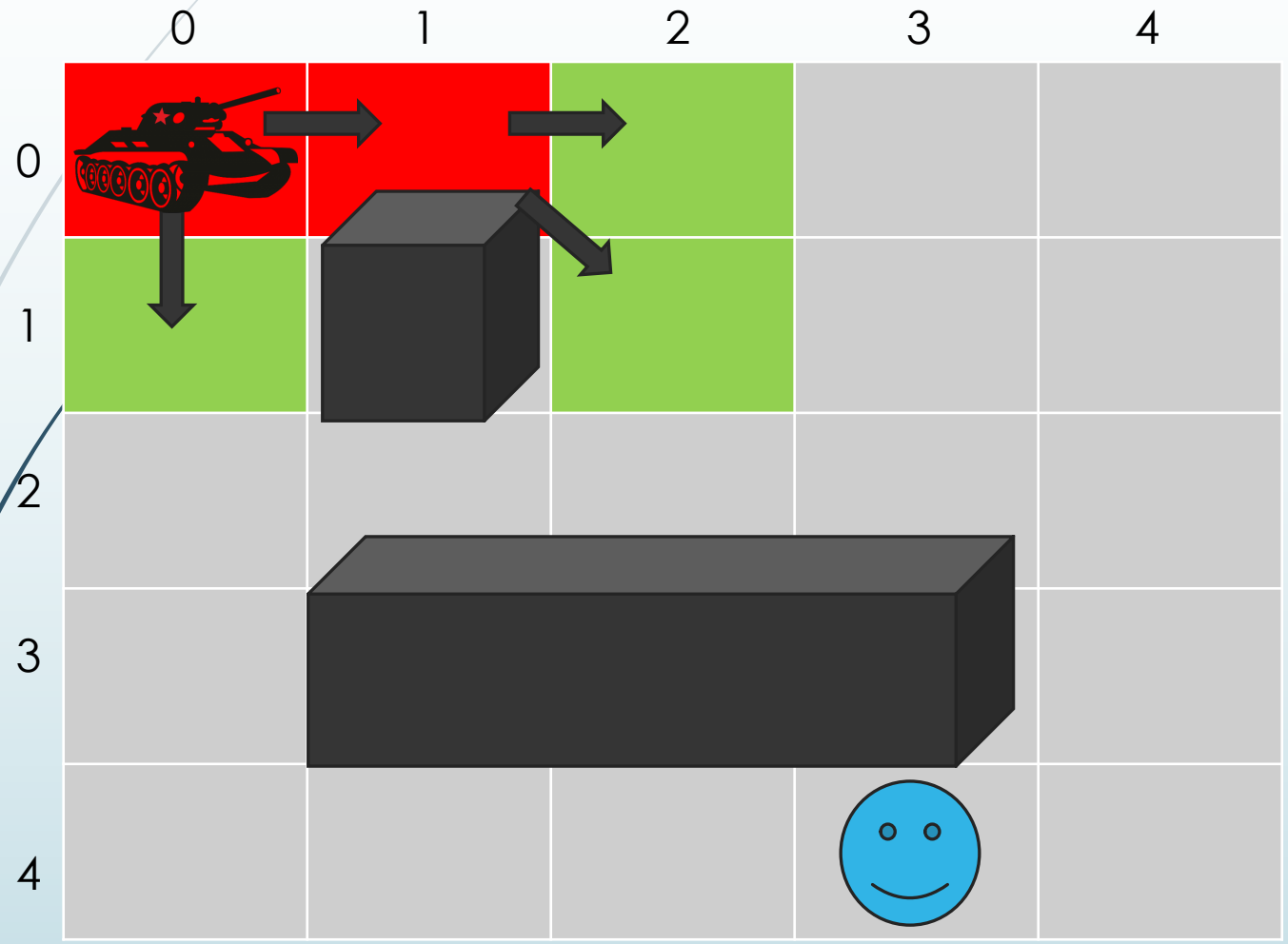
Move Lowest Cost to Closed and Search Neighbours



CLOSED:
(0,0)-0
(1,0)-1-NEW

OPEN:
(0,1)-1
(2,0)-2-NEW
(2,1)-2-NEW
(1,1)-COLLISION!
(0,1)-2-Cost is greater!
(0,0)-CLOSED!

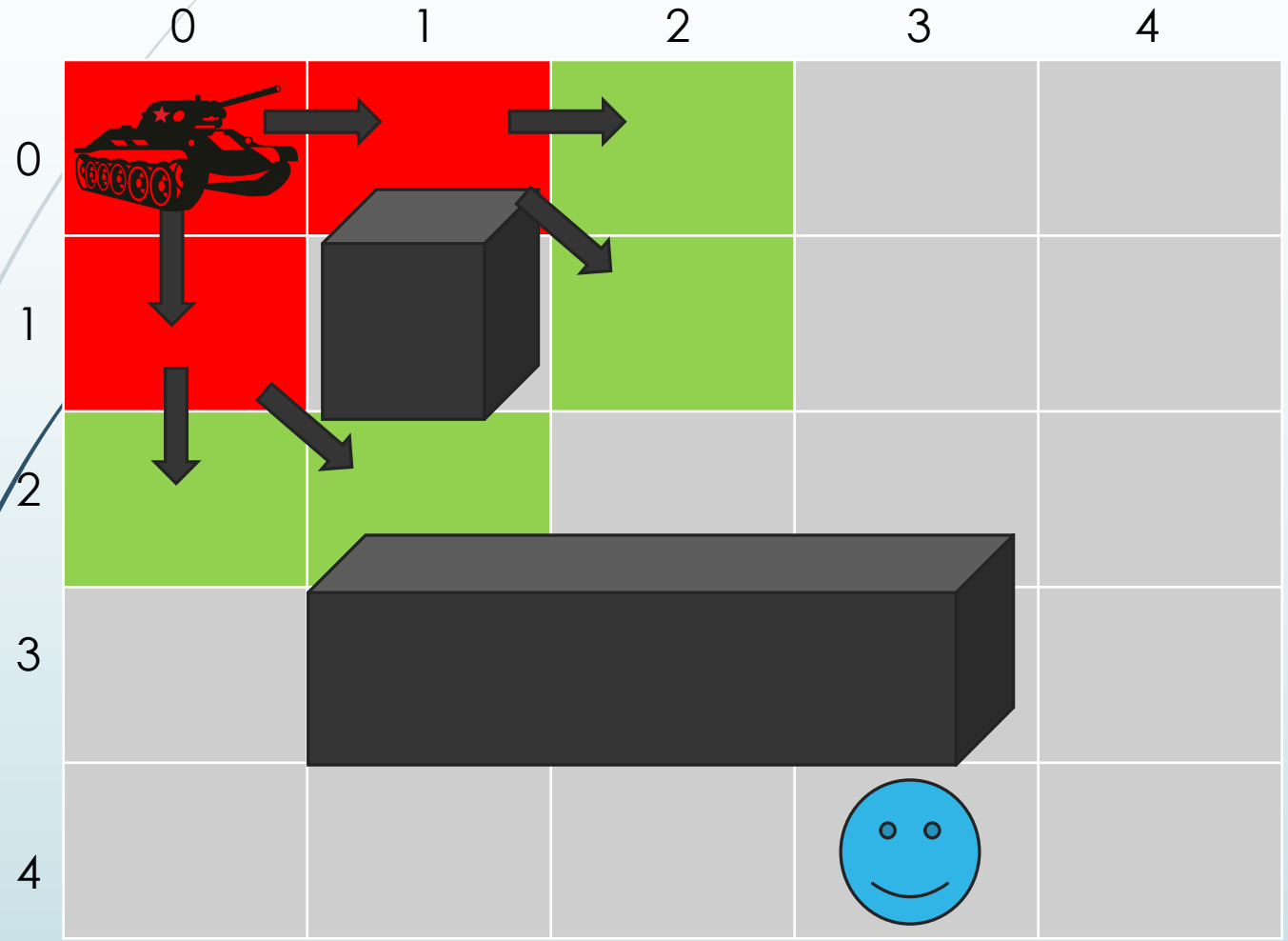
Open and Closed after Two Steps



CLOSED:
(0,0)-0
(1,0)-1

OPEN:
(0,1)-1
(2,0)-2
(2,1)-2

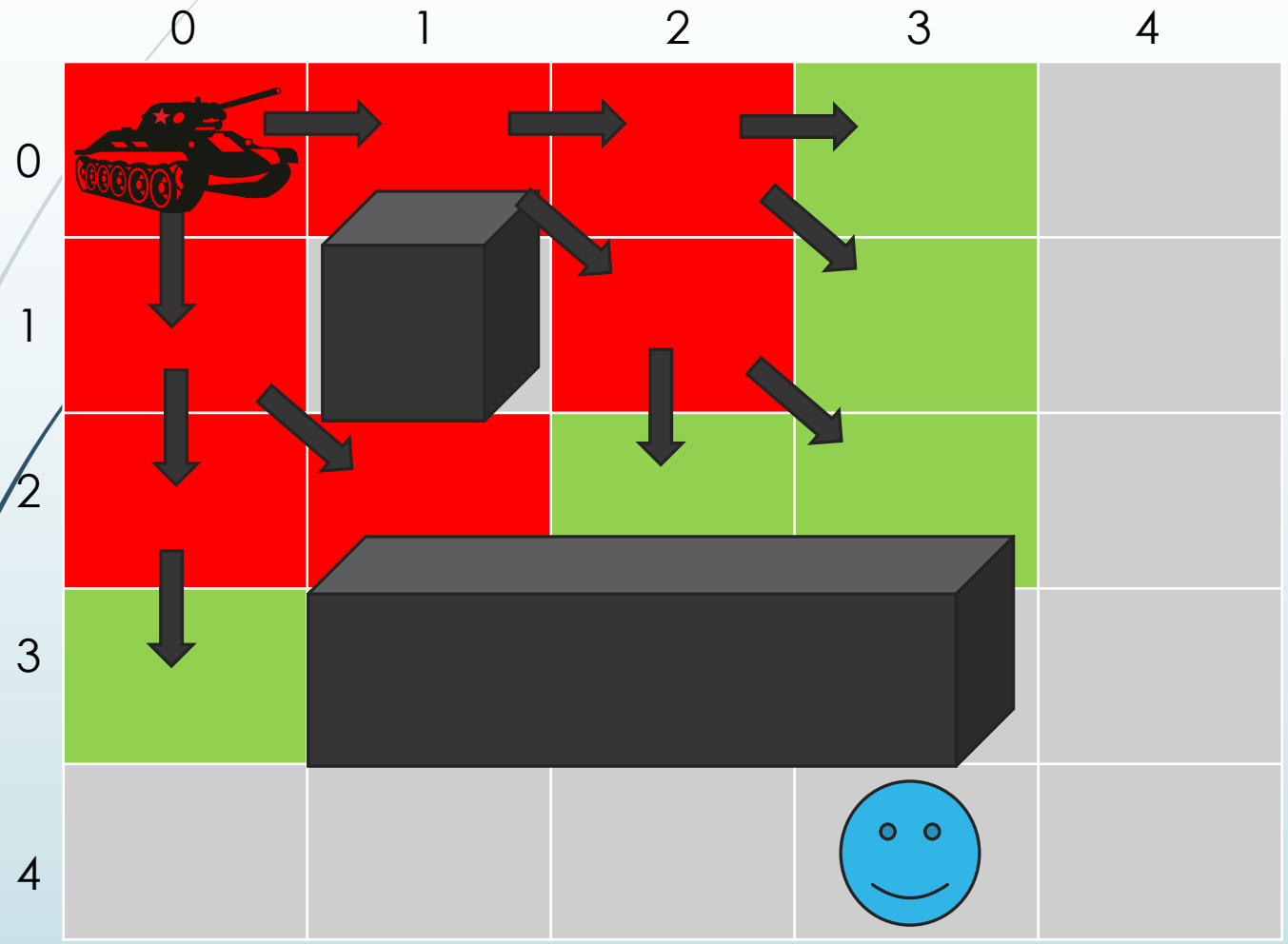
Move Lowest Cost to Closed and Search Neighbours (Only Valid)



CLOSED:
(0,0)-0
(1,0)-1
(0,1)-1-NEW

OPEN:
(2,0)-2
(2,1)-2
(0,2)-2-NEW
(1,2)-2-NEW

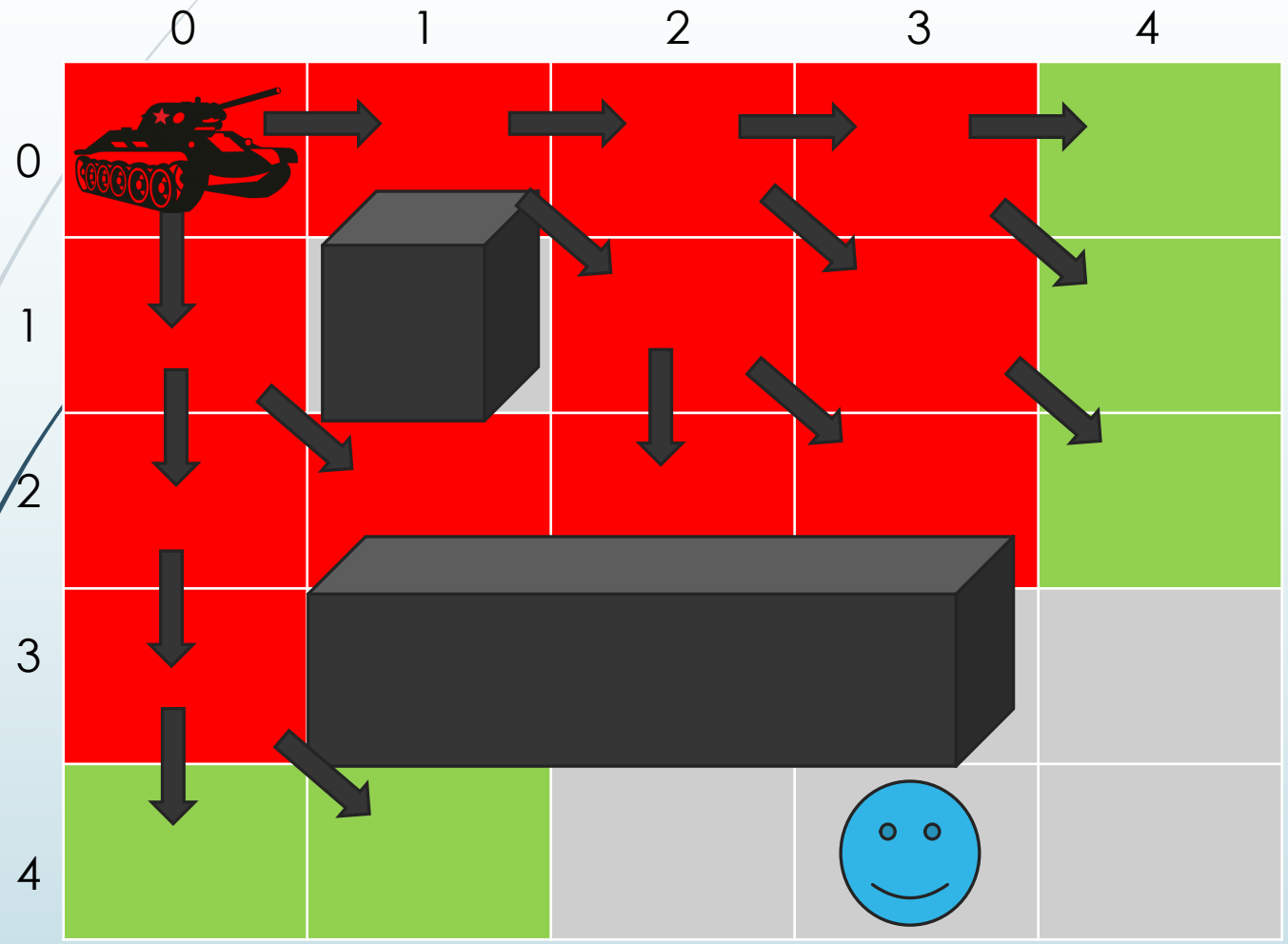
After All Distance 2 Locations Checked



CLOSED:
(0,0)-0, (1,0)-1, (0,1)-1, (2,0)-2, (2,1)-2, (0,2)-2, (1,2)-2

OPEN:
(3,0)-3, (3,1)-3, (3,2)-3, (2,2)-3, (0,3)-3

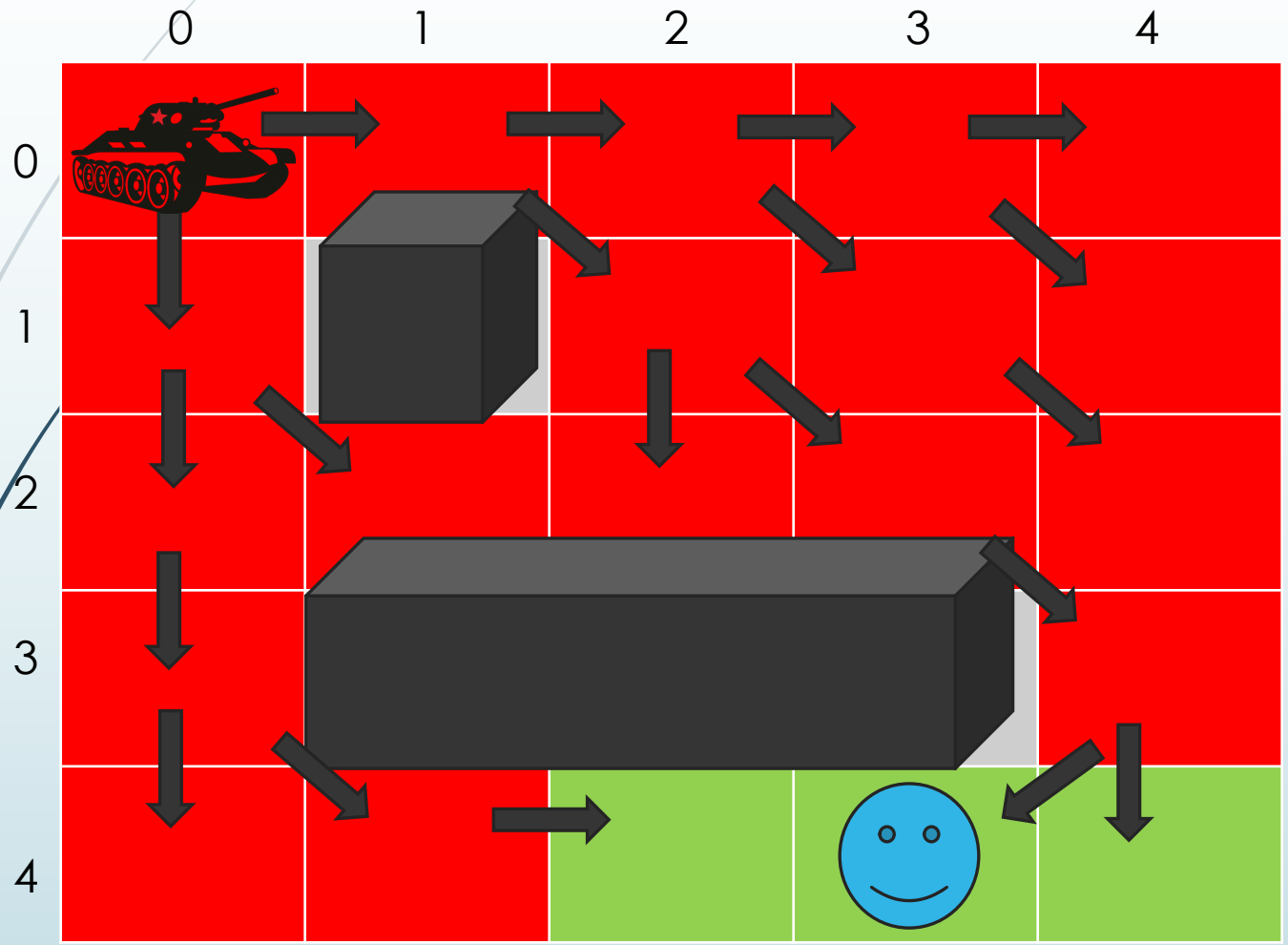
After All Distance 3 Locations Checked



CLOSED:
(0,0)-0, (1,0)-1, (0,1)-1, (2,0)-2, (2,1)-2, (0,2)-2, (1,2)-2, (3,0)-3, (3,1)-3, (3,2)-3, (2,2)-3, (0,3)-3

OPEN:
(4,0)-4, (4,1)-4, (4,2)-4, (0,4)-4, (1,4)-4

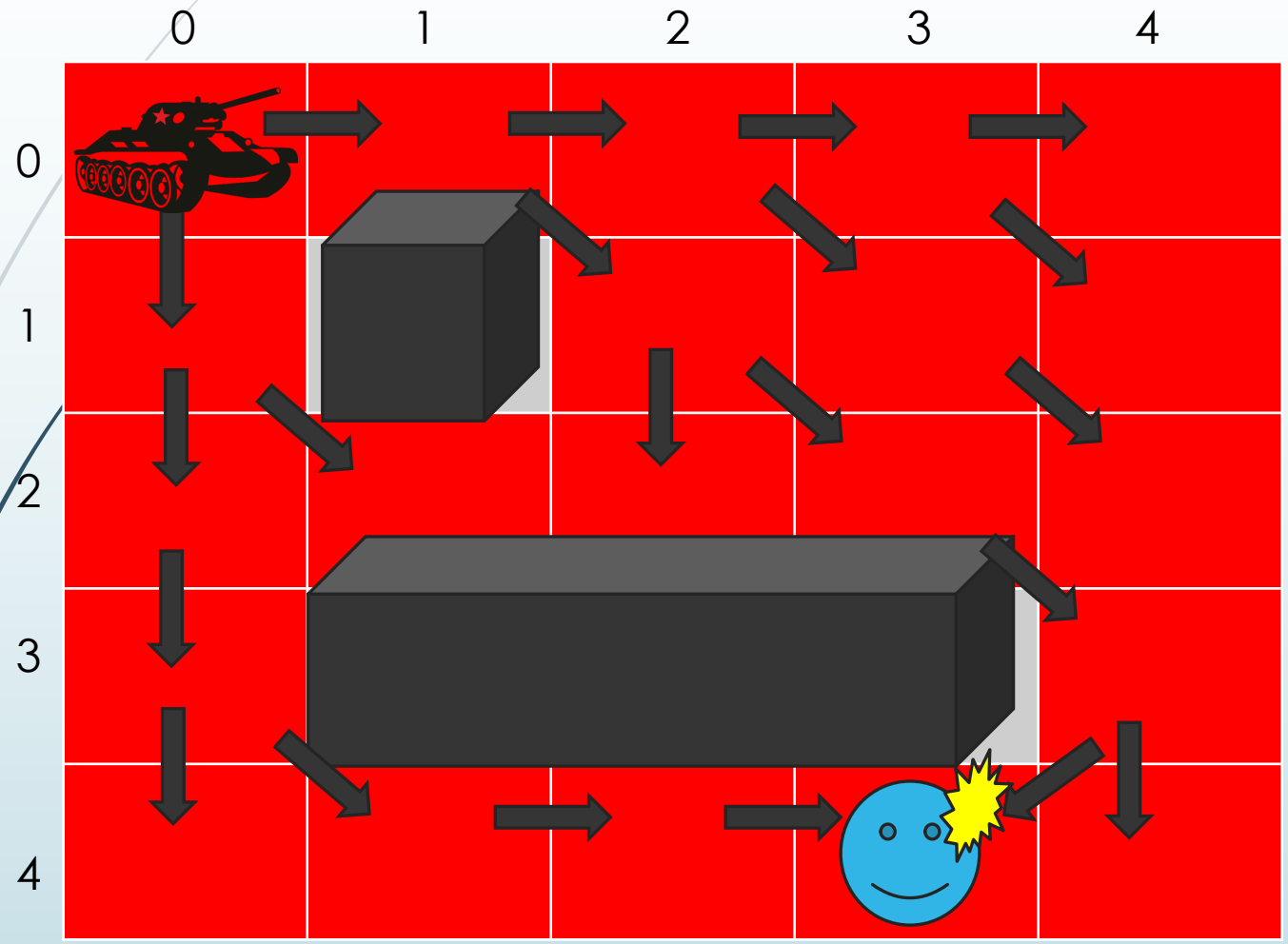
After All Distance 4 Locations Checked



CLOSED:
(0,0)-0, (1,0)-1, (0,1)-1, (2,0)-2, (2,1)-2, (0,2)-2, (1,2)-2, (3,0)-3, (3,1)-3, (3,2)-3, (2,2)-3, (0,3)-3, (4,0)-4, (4,1)-4, (4,2)-4, (0,4)-4, (1,4)-4, (4,3)-4

OPEN:
(2,4)-5, (3,4)-5, (4,4)-5

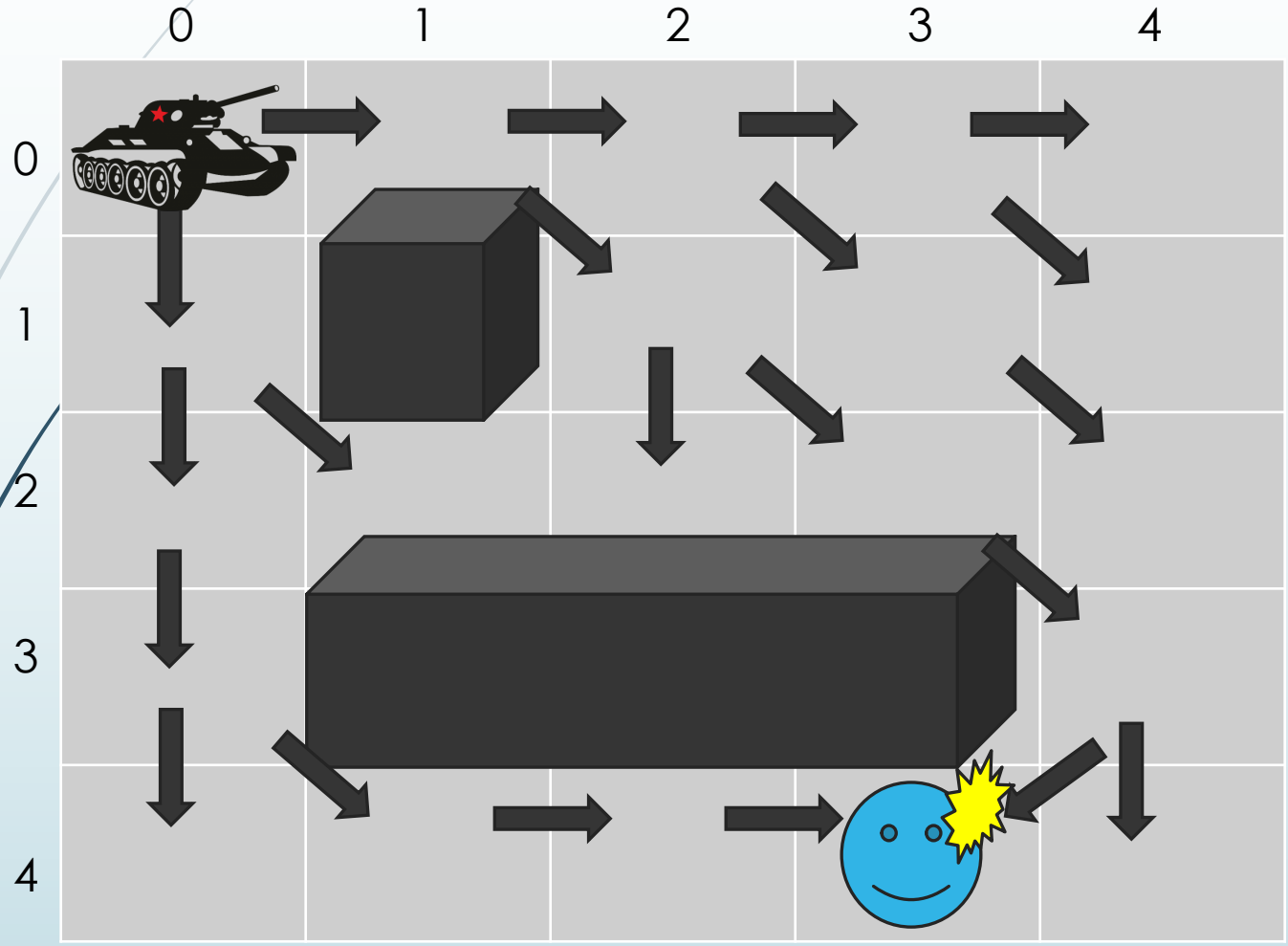
After All Distance 5 Locations Checked



CLOSED:
(0,0)-0, (1,0)-1, (0,1)-1, (2,0)-2, (2,1)-2, (0,2)-2, (1,2)-2, (3,0)-3, (3,1)-3, (3,2)-3, (2,2)-3, (0,3)-3, (4,0)-4, (4,1)-4, (4,2)-4, (0,4)-4, (1,4)-4, (4,3)-4 (2,4)-5, (4,4)-5, (3,4)-5

OPEN:
NULL

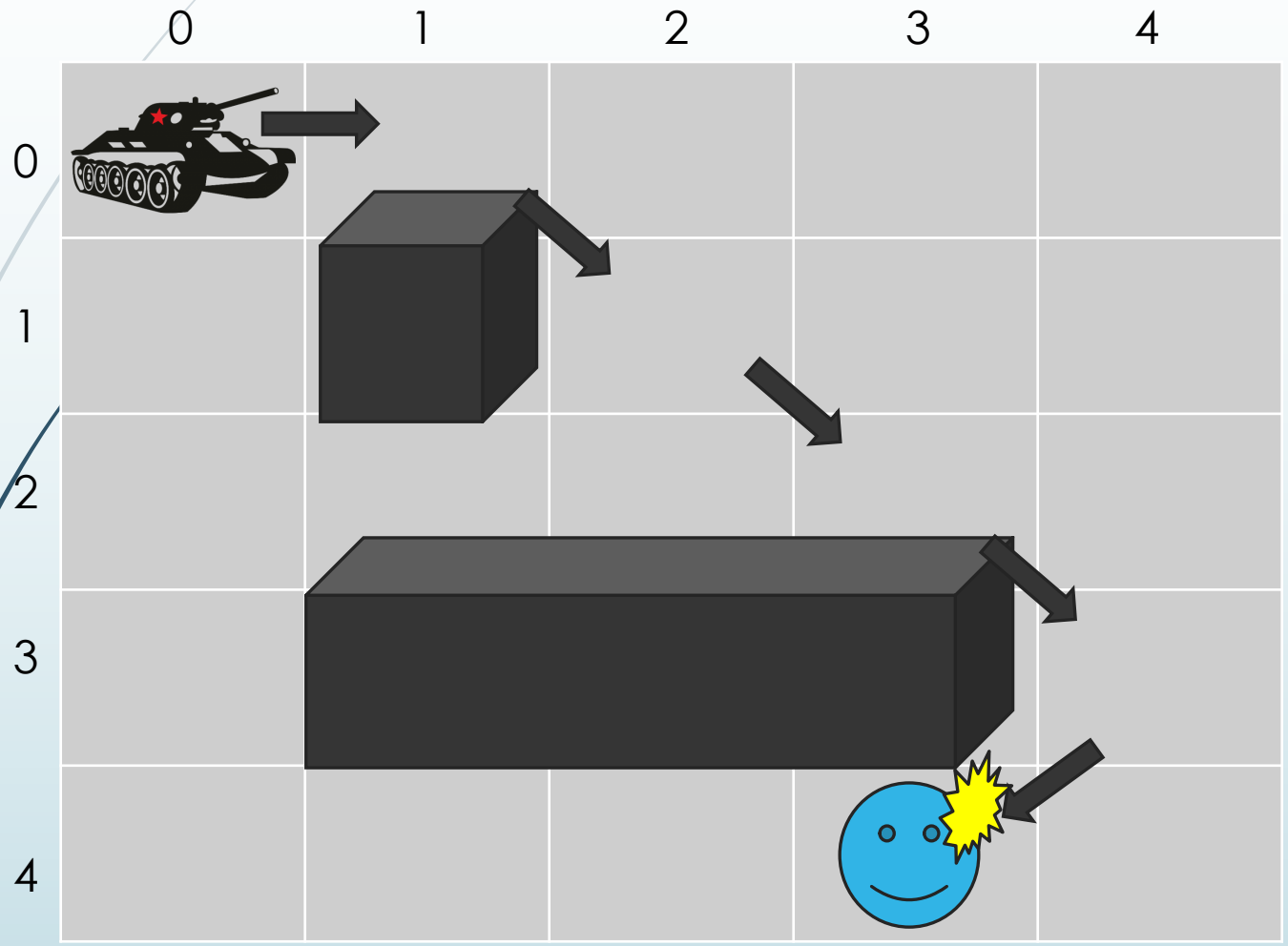
Finished



CLOSED:
(0,0)-0, (1,0)-1, (0,1)-1, (2,0)-2, (2,1)-2, (0,2)-2, (1,2)-2, (3,0)-3, (3,1)-3, (3,2)-3, (2,2)-3, (0,3)-3, (4,0)-4, (4,1)-4, (4,2)-4, (0,4)-4, (1,4)-4, (4,3)-4 (2,4)-5, (4,4)-5, (3,4)-5

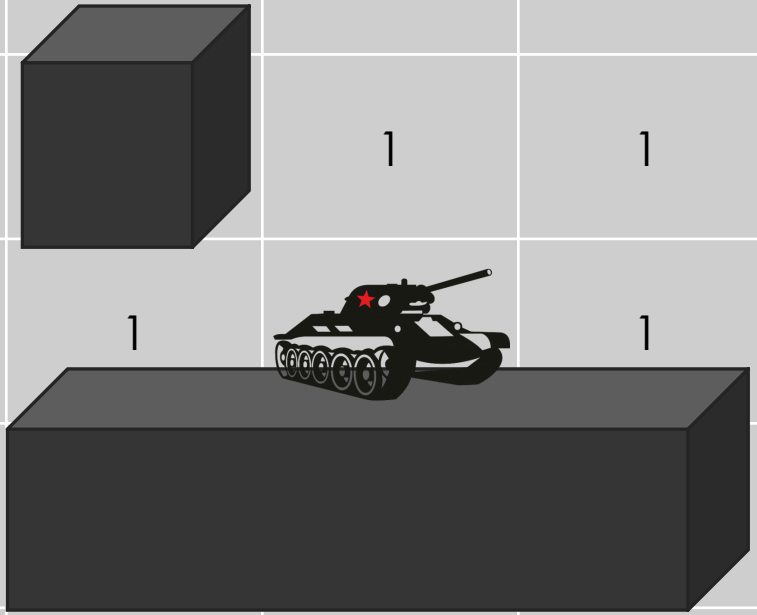
OPEN:
EMPTY

A* Path



World With Movement Values

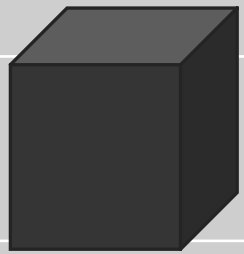
	0	1	2	3	4
0	3	2	2	2	2
1	2		1	1	2
2	2	1		1	2
3	2				2
4	3	4	5		3

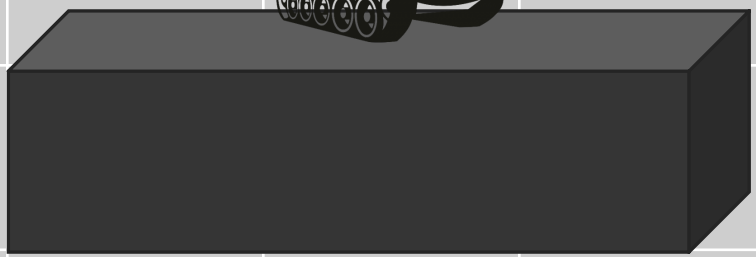


These are g(n) values

World With Movement Values with Manhattan Heuristic

	0	1	2	3	4
0	3+7	2+6	2+5	2+4	2+5
1	2+6		1+4	1+3	2+4
2	2+5	1+4		1+2	2+3
3	2+4				2+2
4	3+3	4+2	5+1		3+1

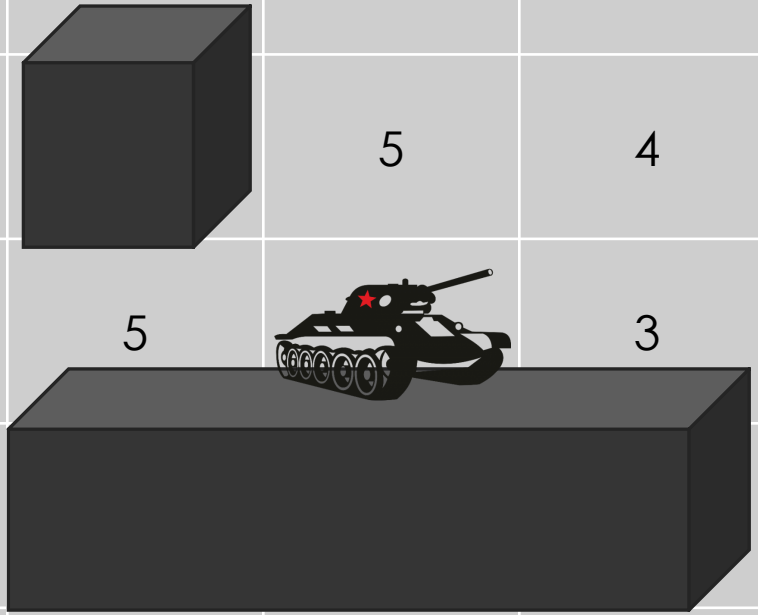




These are g(n) + h(n) values

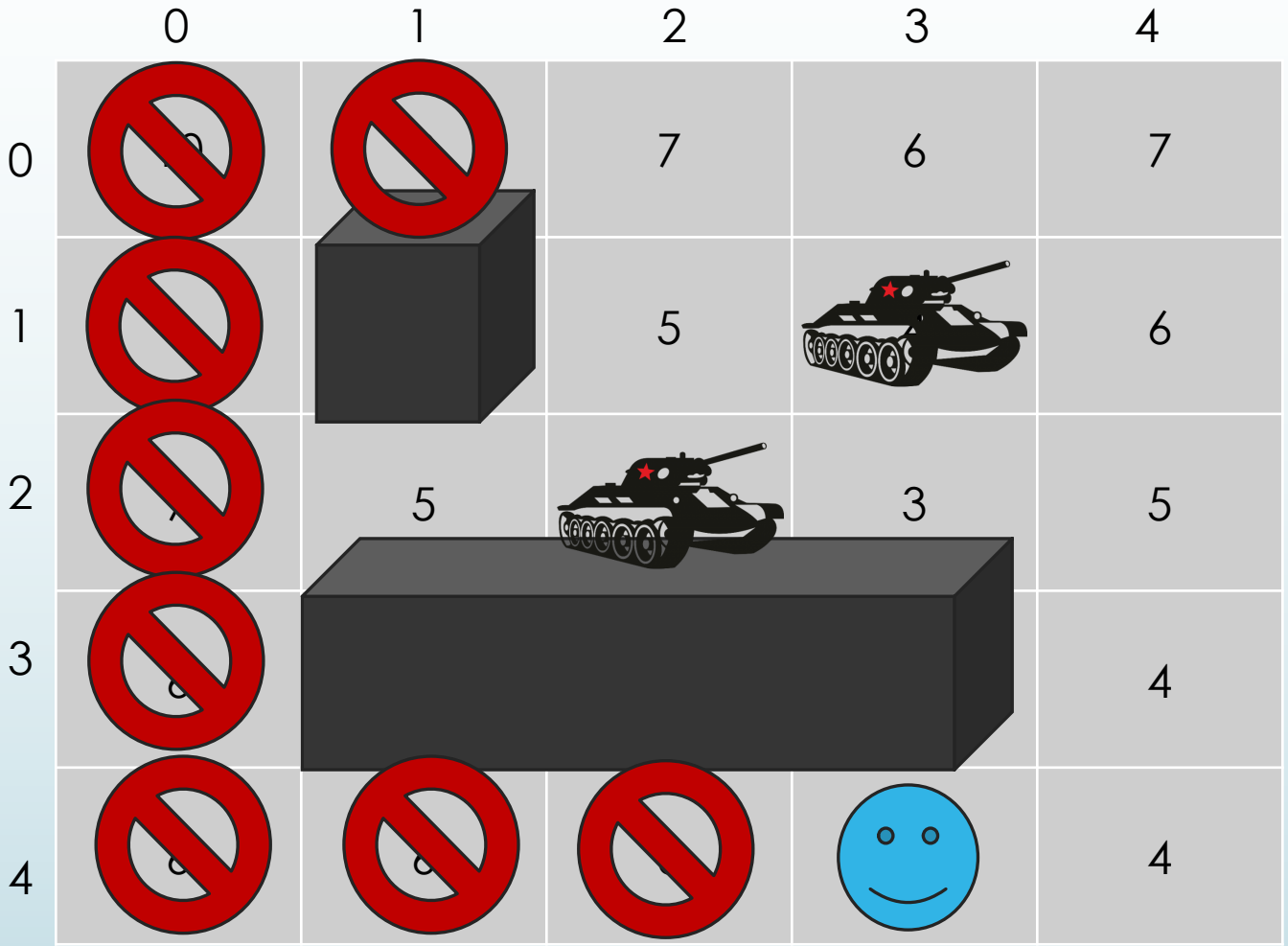
World With Movement Values with Manhattan Heuristic

	0	1	2	3	4
0	10	8	7	6	7
1	8		5	4	6
2	7	5		3	5
3	6				4
4	6	6	6		4



These are f(n) values

World With Movement Values with Manhattan Heuristic – Worst Case



Adding the Heuristic to A*

- If we only look at the number of current steps, then A* degrades into a basic breadth first search
- By adding to the cost the current straight line distance from the start and end point
  - Open.x.cost = current.cost+1+Heuristic();
  - We push the tank towards the right path faster

Improvements A*

- Dealing with other units
  - Treat stopped units as unpassable
  - Treat moving units nearby as unpassable
  - More complexity – prediction of their movement for avoidance – query their path
- Cost of movement
  - Taken into account in the cost function
- Non-grid setting
- Open World
  - Enforce a grid or graph
  - Develop pathways explicitly for longer movements as per before

Tabu Search

- Glover 1980s
- Idea coming from a social taboo – never return to where I once was
  - Heuristic – were I have seen is not where I want to go
- Search is conducted by single point changes in the candidate solution with a list of previous places that the search was at, a Tabu List
- The tabus are feature sets of the parameters
- Searches locally, then places the locality on the tabu areas of the search – forcing it to move away

Tabu Loop

- Set a current position, set best to this position, set a tabu list to null
- While not at a stopping condition
  - Create a candidate list from neighbours
  - Check if the candidates are on the tabu list. If they are, remove them
  - Find the best candidate
  - Move to this as current position
  - Test the best candidate against the best
    - If it is the new best – replace
    - Create a tabu by looking at the differences in the features between this new best and the previous best
    - Produce a tabu restriction based on this difference
    - Add this tabu to the tabu list
    - Check tabu list length and remove old tabus

Example - Tabu

- String of 5 parameters – containing numbers 0-9
- Tabu List is *,*,3,*,*
- New Tabus are created by adding one point where the best strings matched
- Best is 0,3,3,2,1 scoring 14
- Local neighbourhood is +/-1 from each parameter
- Generate the new list
  - 0,2,3,2,0 - 17
  - 0,3,4,2,1 - 18
  - 0,4,4,2,0 - 20

Compare and Add Tabu

- 0,3,3,2,1
- 0,4,4,2,0
- 0,*,*,2,*
- Add the tabu of 0,*,*,*,* or *,*,*,2,*
- New Tabu list of *,*,3,*,* and (0,*,*,*,* or *,*,*,2,*)

Issues with Too Many Tabus

- Small sets of tabus might make it impossible to move
- In our example with 3 tabus all happening about the same value, it would lock movement completely (lock upper and lower move, then move between and make that tabu)
- Rationalize the Tabus – Costly
- Only allow a small enough number of Tabus – where is this necessary value? Is it too low? How often will it happen?
- Aspiration criteria – Accept Tabu moves if they SIGNIFICANTLY increase fitness