Classical AI

SUMMER
SPRINGBOARD
Look Inward. Go Upward.

“Classical” approaches to AI

- Graph Search
- Knowledge Representation
- Evolutionary Programming

SUMMER SPRINGBOARD
Look Inward. Go Upward.

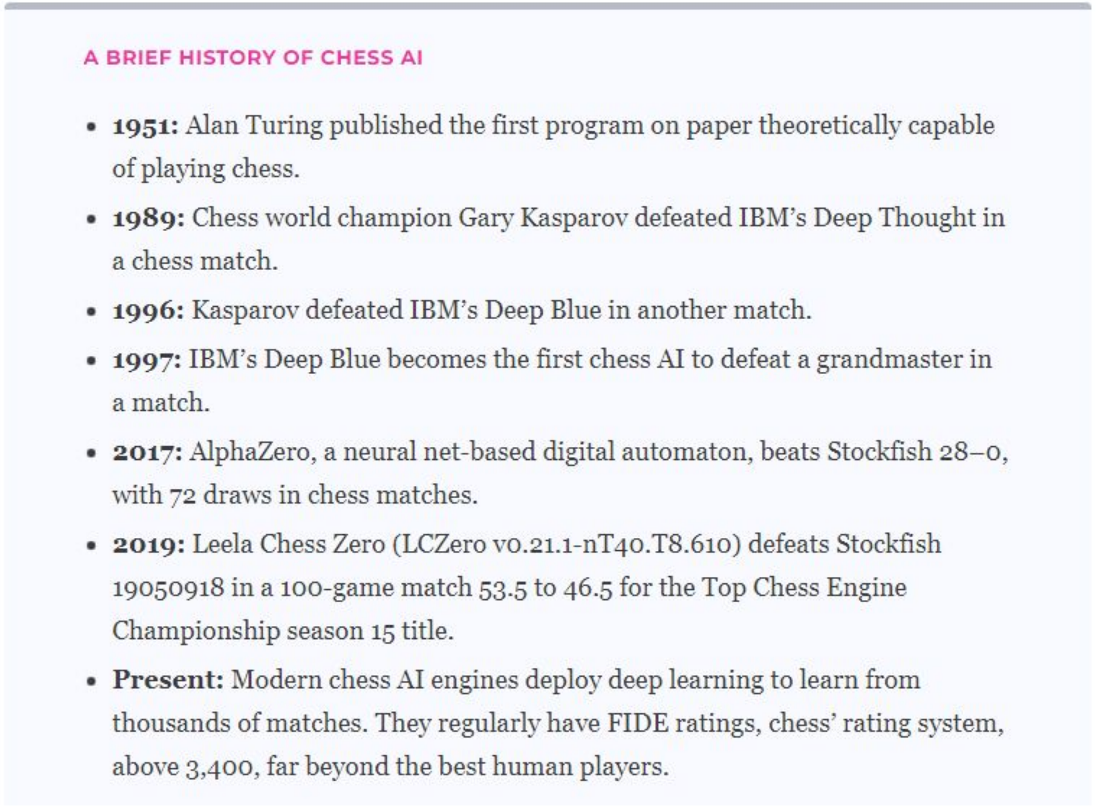
“Classical” approaches to AI

- Graph Search
- Knowledge Representation
- Evolutionary Programming

How to play chess (until 2017)

Summer Springboard

SUMMER
SPRINGBOARD
Look Inward. Go Upward.



A BRIEF HISTORY OF CHESS AI

- **1951:** Alan Turing published the first program on paper theoretically capable of playing chess.
- **1989:** Chess world champion Gary Kasparov defeated IBM's Deep Thought in a chess match.
- **1996:** Kasparov defeated IBM's Deep Blue in another match.
- **1997:** IBM's Deep Blue becomes the first chess AI to defeat a grandmaster in a match.
- **2017:** AlphaZero, a neural net-based digital automaton, beats Stockfish 28–0, with 72 draws in chess matches.
- **2019:** Leela Chess Zero (LCZero v0.21.1-nT40.T8.610) defeats Stockfish 19050918 in a 100-game match 53.5 to 46.5 for the Top Chess Engine Championship season 15 title.
- **Present:** Modern chess AI engines deploy deep learning to learn from thousands of matches. They regularly have FIDE ratings, chess' rating system, above 3,400, far beyond the best human players.

How did it know to do that?

Tic Tac Toe: [Jogo da Velha - Minimax (golb.us)]

How did it do that?

SUMMER SPRINGBOARD
Look Inward. Go Upward.

Game Theory

Game theory is the study of mathematical models of strategic interactions among rational agents. Modern game theory began with the idea of mixed-strategy equilibria in two-person zero-sum game and its proof by John von Neumann (1928).

?

A game is <u>zero-sum</u> if the players win and lose by the same amount

Ex:

Poker: If I win $50, you lose $50

Tic-Tac-Toe: If I win (+1), you lose (-1)

Counter-example:

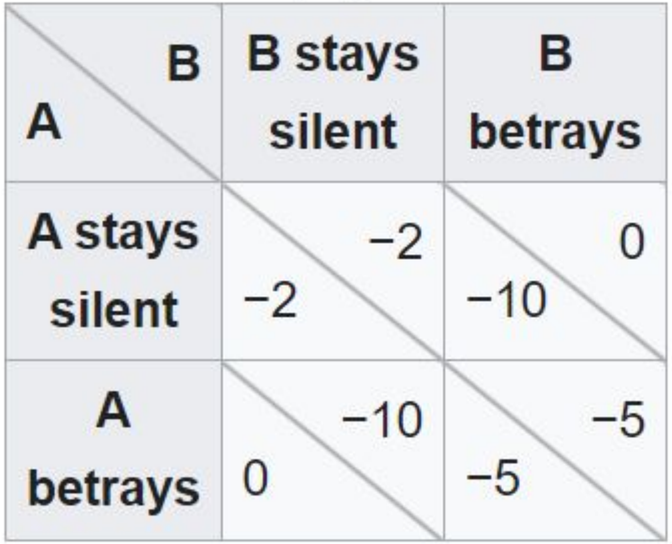
Games with scores: As I write this, the Celtics just beat the 76ers 121-87

Cooperative games: <u>Pandemic</u>

Two really important game theory examples

The Prisoner’s Dilemma*

- **Two members of a criminal gang, Alice and Bob, are arrested**
- Each prisoner is in solitary confinement with **no means of communication with their partner**
- The police do not have the evidence for a conviction.
- If **both stay silent** they will **both be sentenced to two years**
- If **one confesses they will go free**, but their **partner will serve ten years**
- If **both confess** they will **each serve 5 years**



B / A	B stays silent	B betrays
A stays silent	−2 −2	0 −10
A betrays	−10 0	−5 −5

What would you do?
What would you do if you had to do it more than once?

* The Prisoner’s Dilemma is a non-zero sum game

MMER
NGBOARD
vard. Go Upward.

There is an objectively correct answer

- Always stay silent
- Stay silent until you are betrayed
  - Co-operate until they prove themselves untrustworthy
- Always betray
- Betray until other person stays silent
  - Assume they are untrustworthy until proven otherwise
- Stochastic
  - e.g. flip a coin (that may not be 50/50)
- Stochastic with memory
  - Make a random decision informed by what has happened so far

There is an objectively correct answer

- ~~Always stay silent~~
- Stay silent until you are betrayed
  - Co-operate until they prove themselves untrustworthy
- Always betray
- Betray until other person stays silent
  - Assume they are untrustworthy until proven otherwise
- Stochastic
  - e.g. flip a coin (that may not be 50/50)
- Stochastic with memory
  - Make a random decision informed by what has happened so far

SUMMER SPRINGBOARD
Look Inward. Go Upward.

There is an objectively correct answer

- ~~Always stay silent~~
- ~~Stay silent until you are betrayed~~
    - Co-operate until they prove themselves untrustworthy
- Always betray
- Betray until other person stays silent
    - Assume they are untrustworthy until proven otherwise
- Stochastic
    - e.g. flip a coin (that may not be 50/50)
- Stochastic with memory
    - Make a random decision informed by what has happened so far

SUMMER SPRINGBOARD
Look Inward. Go Upward.

There is an objectively correct answer

- ~~Always stay silent~~
- ~~Stay silent until you are betrayed~~
    - Co-operate until they prove themselves untrustworthy
- ~~Always betray~~
- Betray until other person stays silent
    - Assume they are untrustworthy until proven otherwise
- Stochastic
    - e.g. flip a coin (that may not be 50/50)
- Stochastic with memory
    - Make a random decision informed by what has happened so far

SUMMER SPRINGBOARD
Look Inward. Go Upward.

There is an objectively correct answer

- ~~Always stay silent~~
- ~~Stay silent until you are betrayed~~
  - Co-operate until they prove themselves untrustworthy
- ~~Always betray~~
- ~~Betray until other person stays silent~~
  - Assume they are untrustworthy until proven otherwise
- Stochastic
  - e.g. flip a coin (that may not be 50/50)
- Stochastic with memory
  - Make a random decision informed by what has happened so far

SUMMER SPRINGBOARD
Look Inward. Go Upward.

There is an objectively correct answer

- ~~Always stay silent~~
- ~~Stay silent until you are betrayed~~
  - Co-operate until they prove themselves untrustworthy
- ~~Always betray~~
- ~~Betray until other person stays silent~~
  - Assume they are untrustworthy until proven otherwise
- ~~Stochastic~~
  - e.g. flip a coin (that may not be 50/50)
- Stochastic with memory
  - Make a random decision informed by what has happened so far

SUMMER SPRINGBOARD
Look Inward. Go Upward.

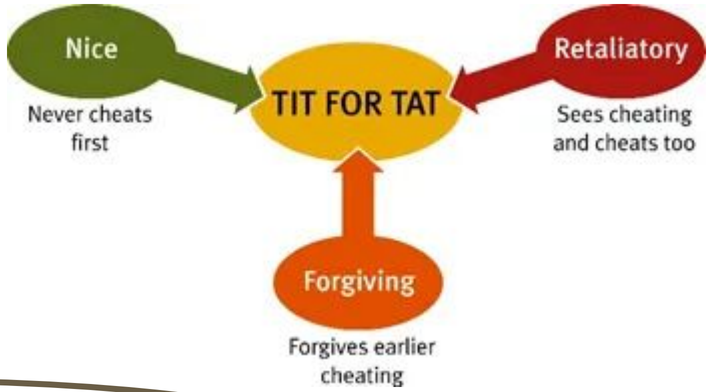
There is an objectively correct answer

- ~~Always stay silent~~
- ~~Stay silent until you are betrayed~~
  - Co-operate until they prove themselves untrustworthy
- ~~Always betray~~
- ~~Betray until other person stays silent~~
  - Assume they are untrustworthy until proven otherwise
- ~~Stochastic~~
  - e.g. flip a coin (that may not be 50/50)
- ~~Stochastic with memory~~
  - Make a random decision informed by what has happened so far

SUMMER SPRINGBOARD
Look Inward. Go Upward.

Tit-for-Tat: the objectively correct answer

- On the first move, stay silent
- In every subsequent move, repeat what your partner did on the last move



[What Can Game Theory Teach Us about Life? (scilogism.com)]

What *else* can game theory teach us about life?

SUMMER SPRINGBOARD
Look Inward. Go Upward.

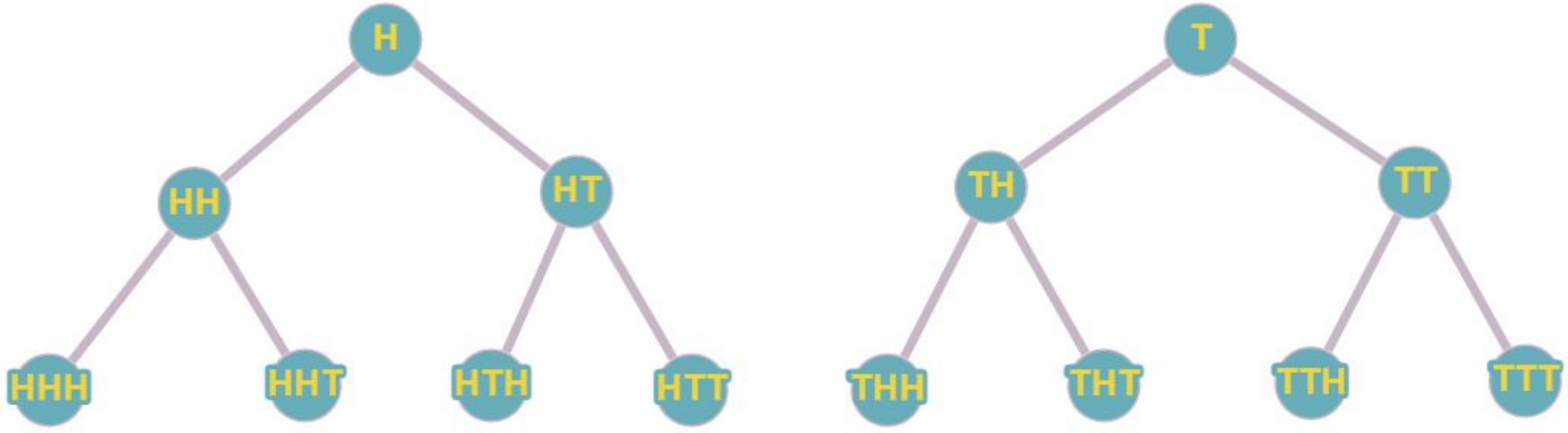
MUTUALLY ASSURED DESTRUCTION

[(1) The Infographics Show - YouTube]

A really dumb “game”

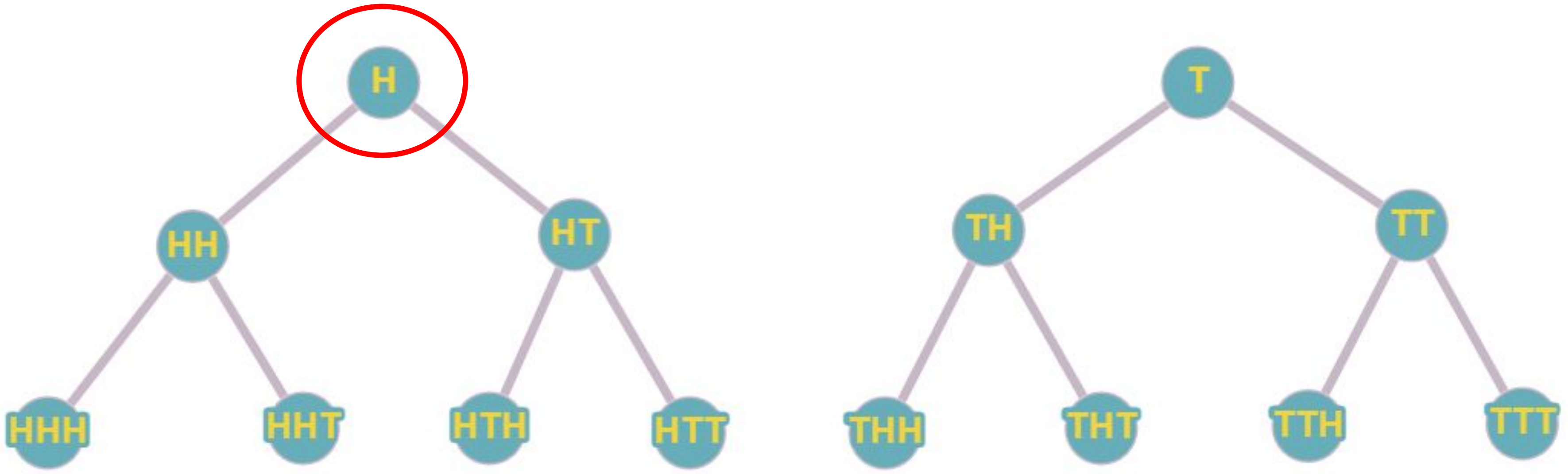
- We will take turns placing a coins heads up or heads down
- Play alternates player 1, player 2, player 1
- If there are two heads, player 1 wins
- Otherwise, player 2 wins
- Boring question: who wins?
- Interesting question: how did you know that?

SUMMER SPRINGBOARD
Look Inward. Go Upward.

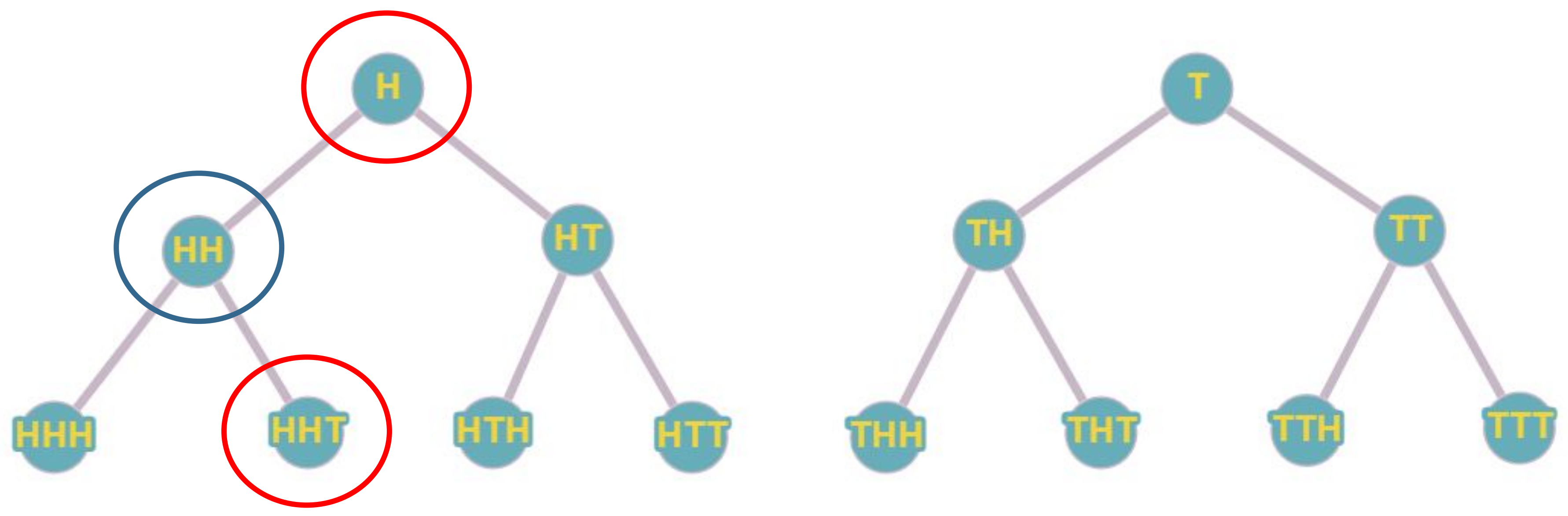


1 2 3 4 5 6 7 8

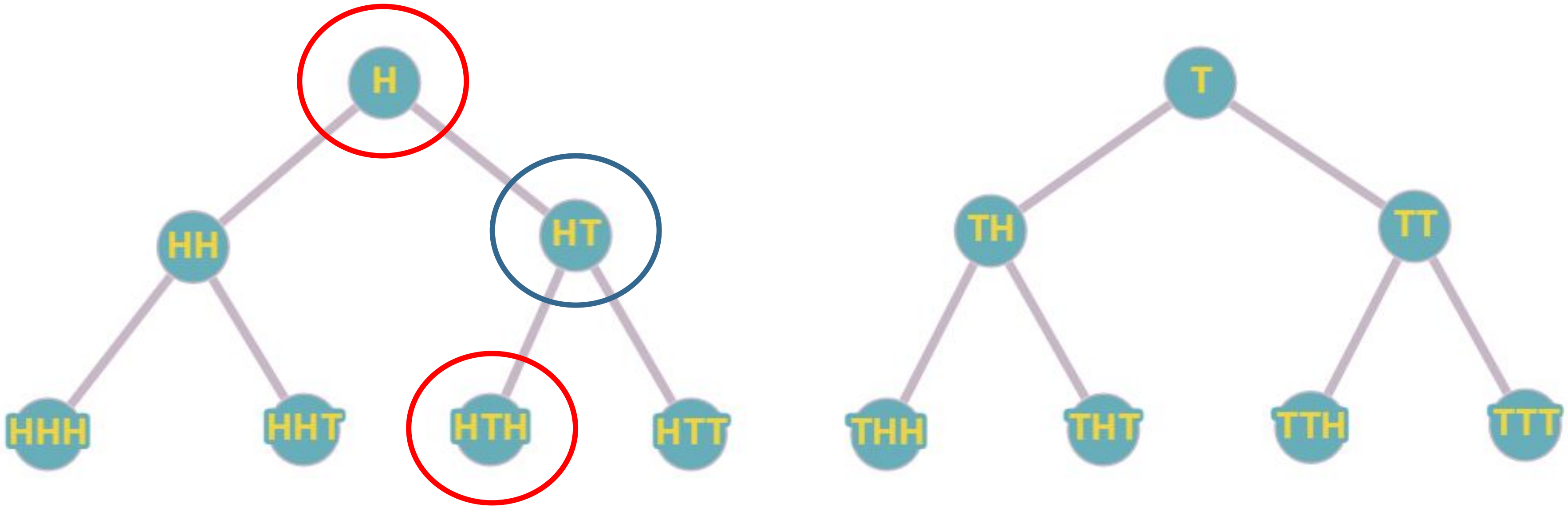
- There are 8 possible games



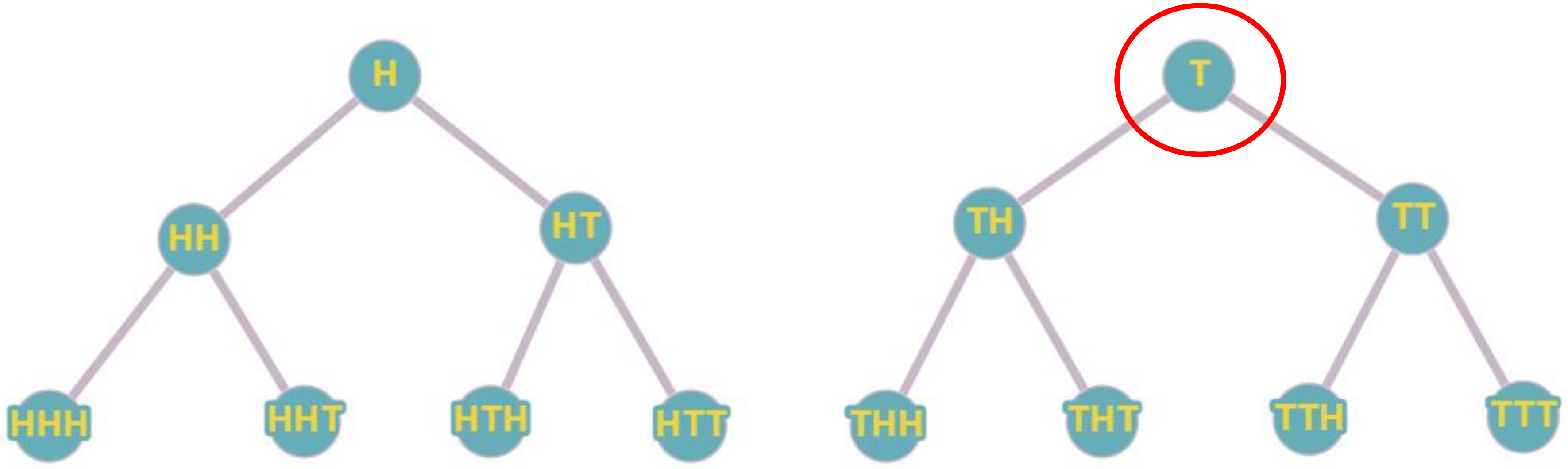
- If player 1 picks heads



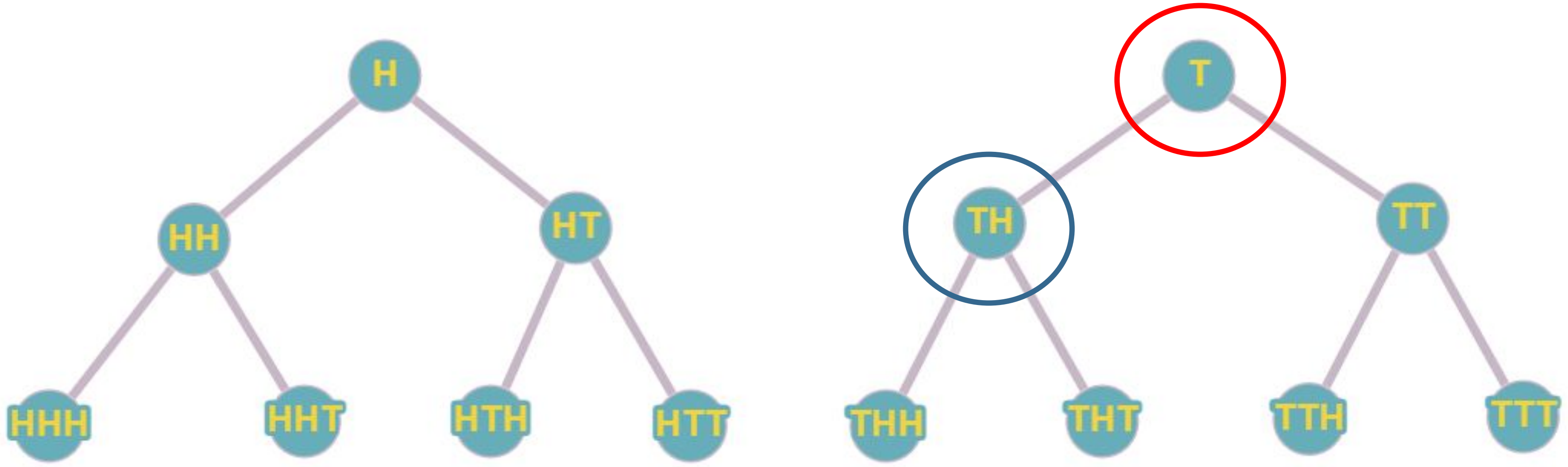
- If player 1 picks heads
    - Player 1 wins no matter what player 2 does



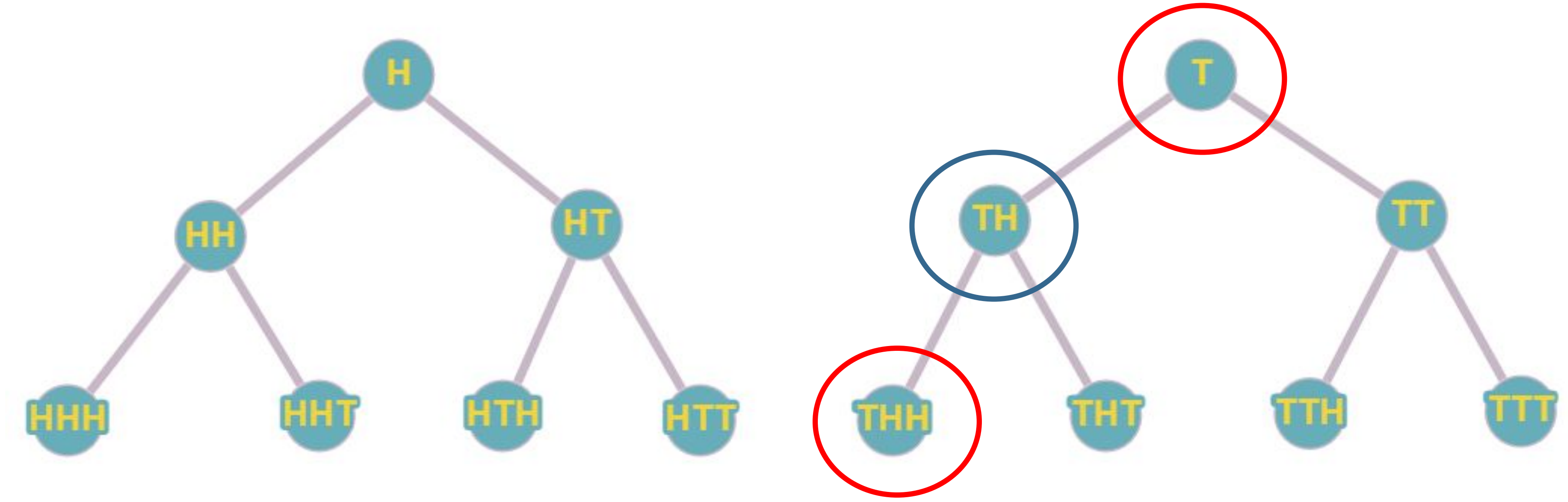
- If player 1 picks heads
  - Player 1 wins no matter what player 2 does



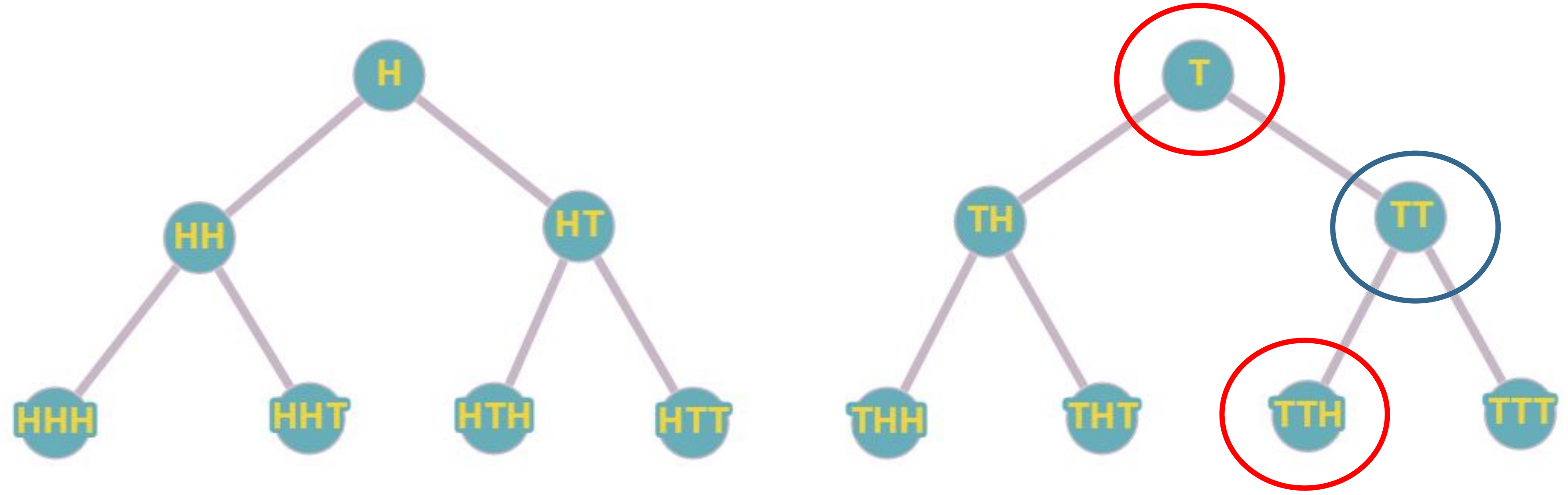
- If player 1 picks tails



- If player 1 picks tails
  - And player 2 picks heads

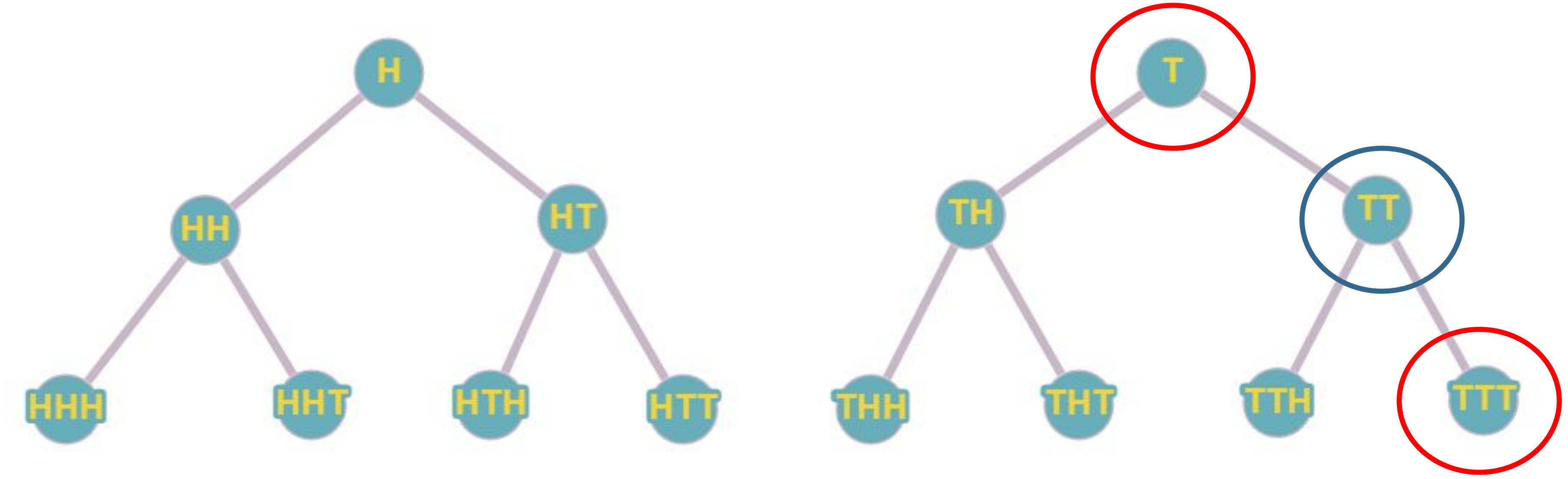


- If player 1 picks tails
    - And player 2 picks heads
        - Then player 1 wins

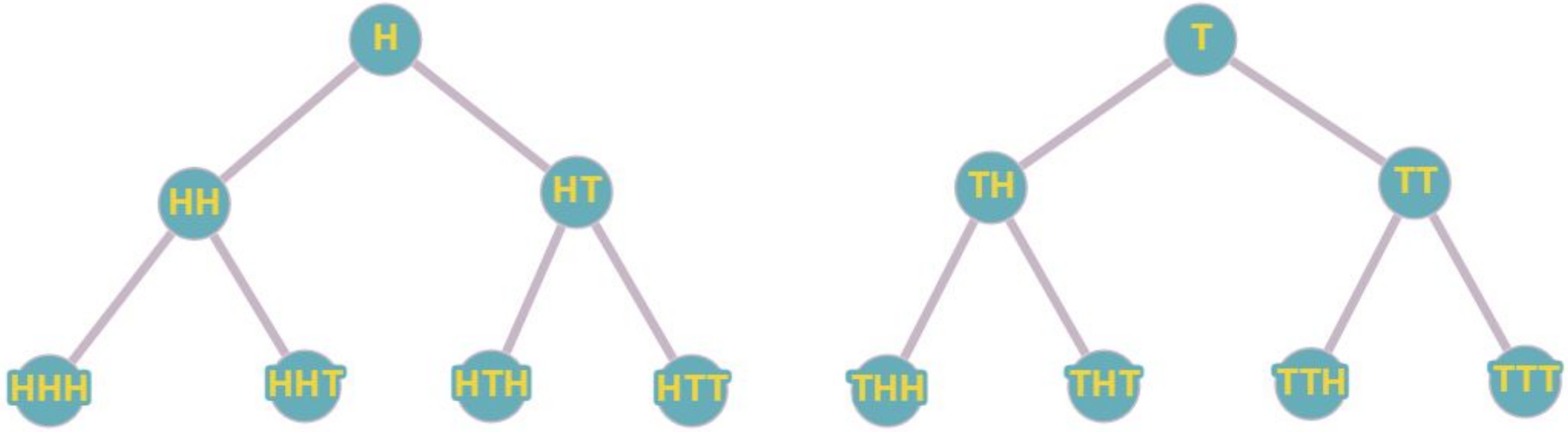


- If player 1 picks tails
  - And player 2 picks heads
    - Then player 1 wins
  - And player 2 picks tails
    - Then player 2 wins no matter what player 1 does

SUMMER SPRINGBOARD
Look Inward. Go Upward.



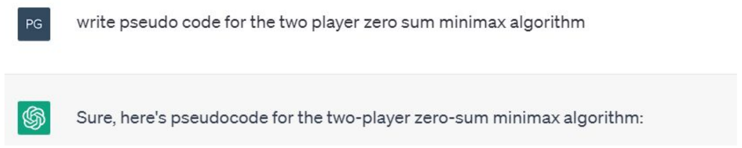
- If player 1 picks tails
    - And player 2 picks heads
        - Then player 1 wins
    - And player 2 picks tails
        - Then player 2 wins no matter what player 1 does



- Who wins?
- Player 1
- They will always pick heads on their first move

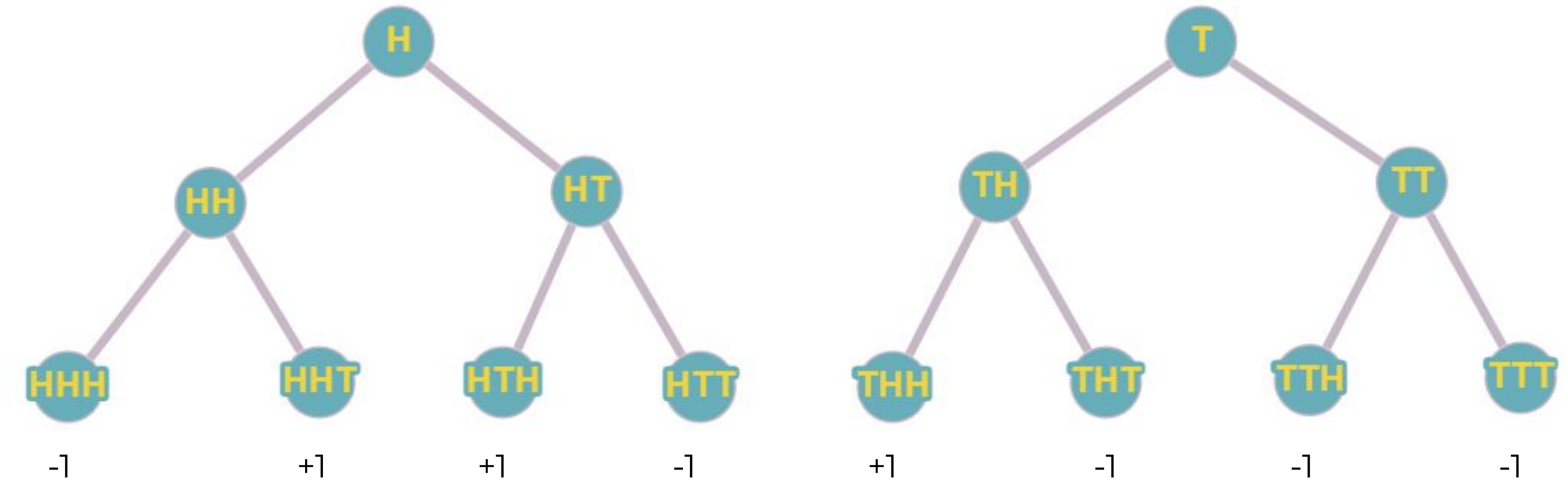
The Minimax Algorithm: How to play Chess (slowly)

```
function minimax(node, depth, maximizingPlayer):
    ~~if depth == 0 or node is a terminal node:~~
    if the game is over or someone won:
        return the score of the node

    if maximizingPlayer:
        bestValue = -infinity
        ~~for each child of node:~~
        for each possible move:
            value = minimax(child, depth - 1, false)
            bestValue = max(bestValue, value)
        return bestValue
    else:  # minimizing player
        bestValue = infinity
        ~~for each child of node:~~
        for each possible move:
            value = minimax(child, depth - 1, true)
            bestValue = min(bestValue, value)
        return bestValue
```


PG write pseudo code for the two player zero sum minimax algorithm

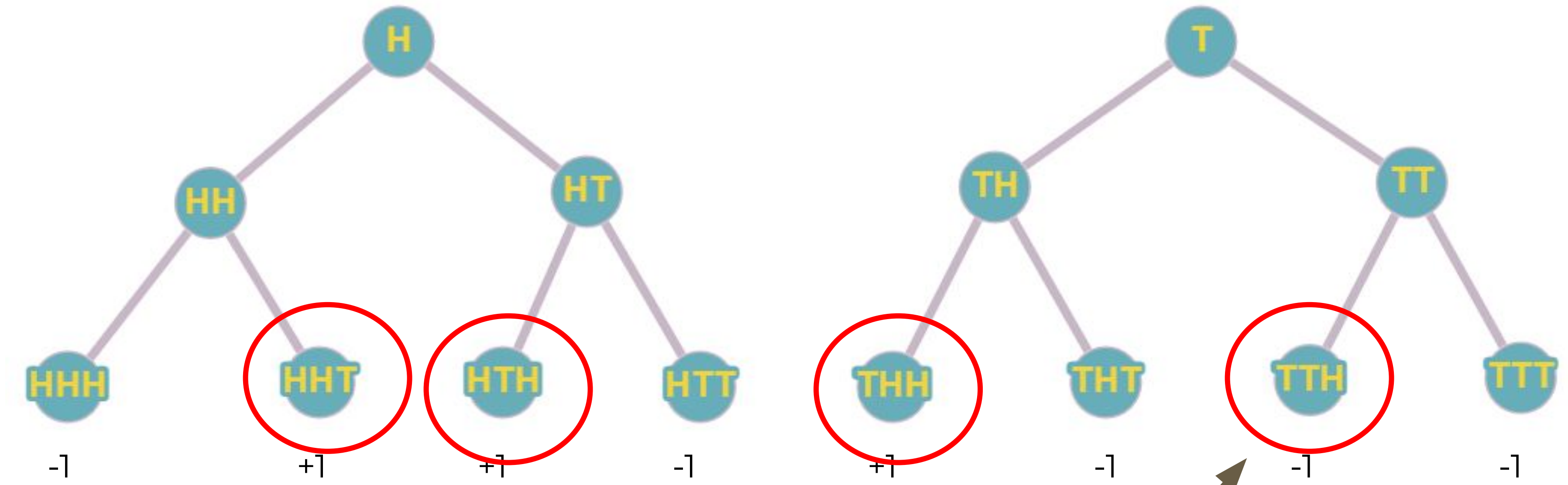
Sure, here's pseudocode for the two-player zero-sum minimax algorithm:

The Minimax Algorithm: How to play Chess (slowly)



SUMMER SPRINGBOARD
Look Inward. Go Upward.

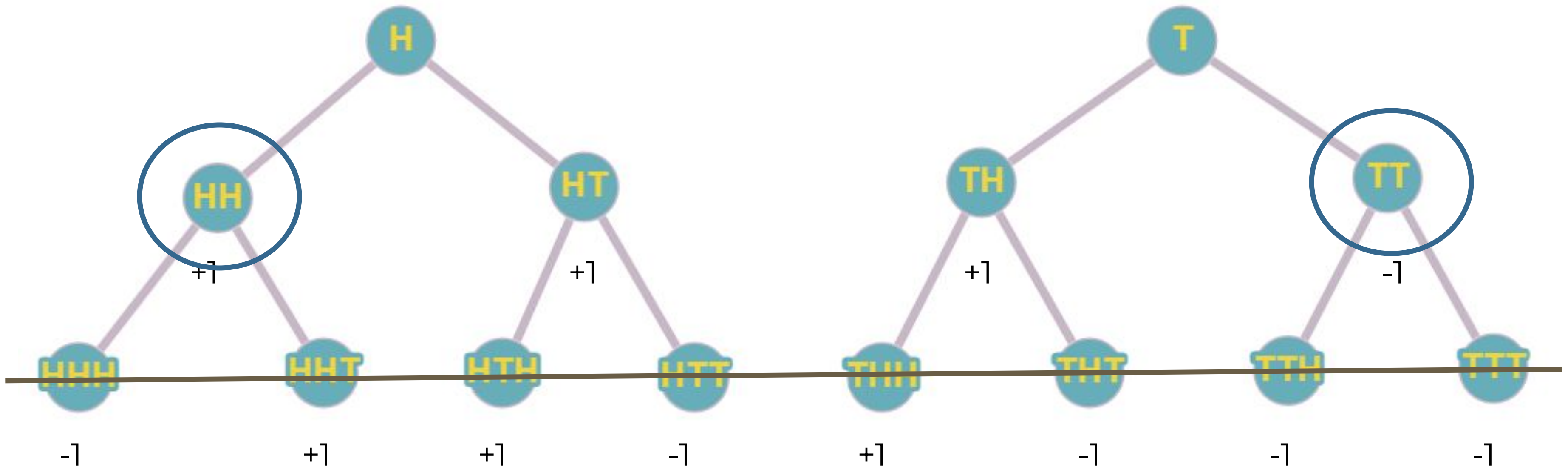
The Minimax Algorithm: How to play Chess (slowly)



My turn: MAX

Chosen arbitrarily

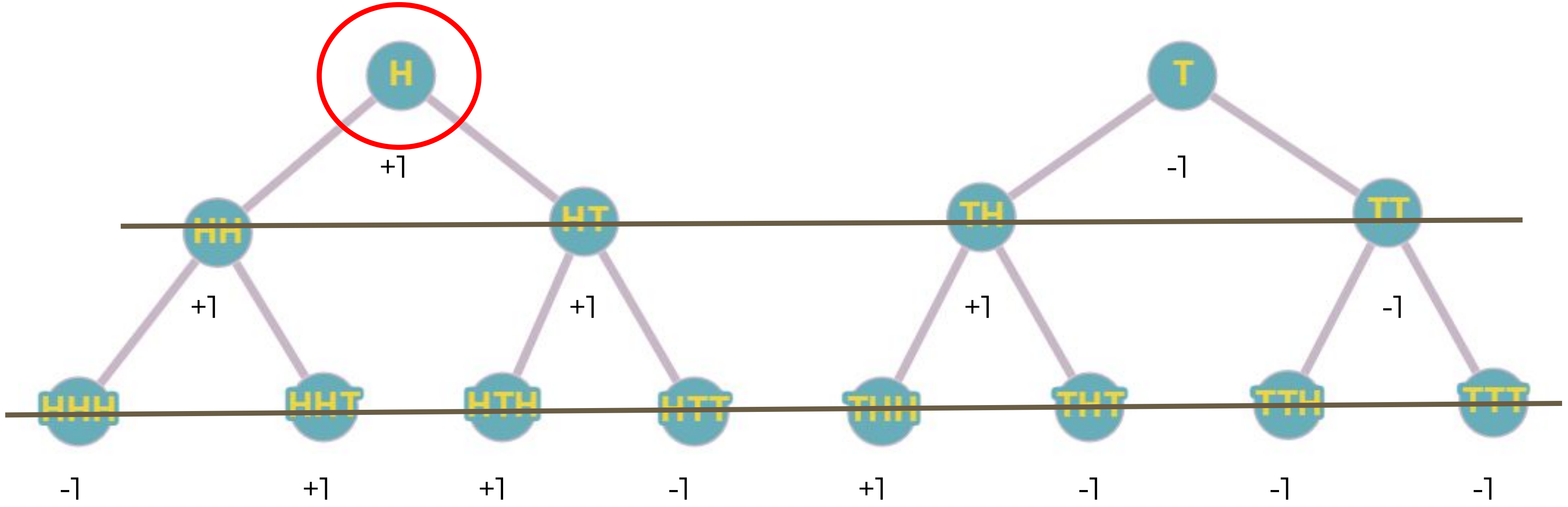
The Minimax Algorithm: How to play Chess (slowly)



Their turn: MIN
My turn: MAX

SUMMER SPRINGBOARD
Look Inward. Go Upward.

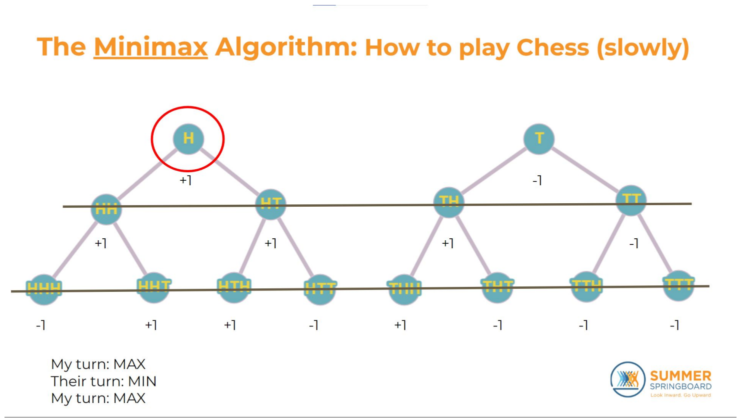
The <u>Minimax</u> Algorithm: How to play Chess (slowly)

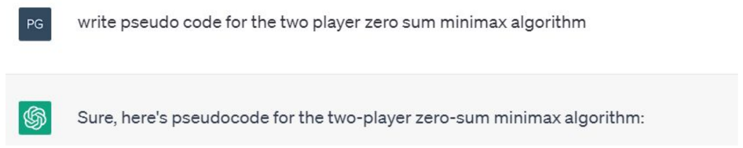


My turn: MAX
Their turn: MIN
My turn: MAX

The Minimax Algorithm: How to play Chess (slowly)

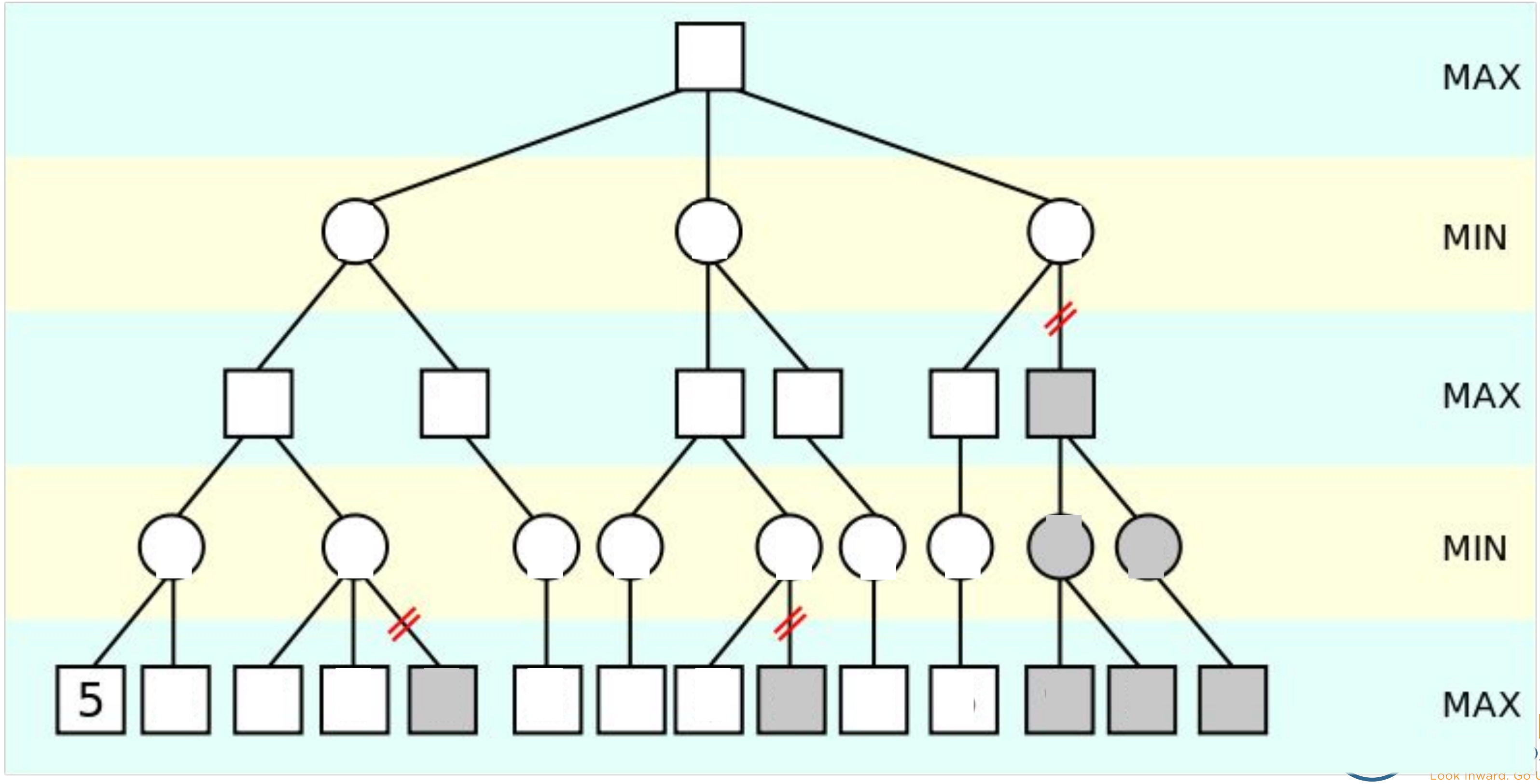
```
function minimax(node, depth, maximizingPlayer):
    ~~if depth == 0 or node is a terminal node:~~
    if the game is over or someone won:
        return the score of the node

    if maximizingPlayer:
        bestValue = -infinity
        ~~for each child of node:~~
        for each possible move:
            value = minimax(child, depth - 1, false)
            bestValue = max(bestValue, value)
        return bestValue
    else:  # minimizing player
        bestValue = infinity
        ~~for each child of node:~~
        for each possible move:
            value = minimax(child, depth - 1, true)
            bestValue = min(bestValue, value)
        return bestValue
```

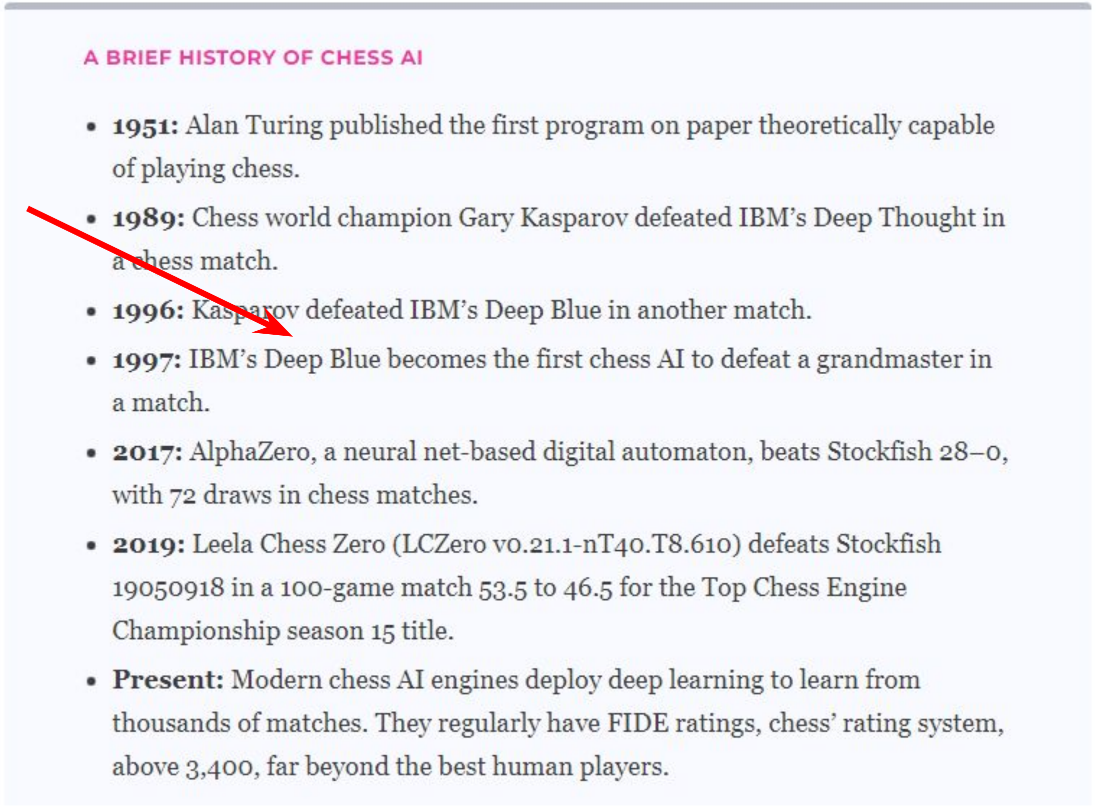



PG write pseudo code for the two player zero sum minimax algorithm

Sure, here's pseudocode for the two-player zero-sum minimax algorithm:

α-β Pruning: How to play Chess (quickly)





A BRIEF HISTORY OF CHESS AI

- **1951:** Alan Turing published the first program on paper theoretically capable of playing chess.
- **1989:** Chess world champion Gary Kasparov defeated IBM's Deep Thought in a chess match.
- **1996:** Kasparov defeated IBM's Deep Blue in another match.
- **1997:** IBM's Deep Blue becomes the first chess AI to defeat a grandmaster in a match.
- **2017:** AlphaZero, a neural net-based digital automaton, beats Stockfish 28–0, with 72 draws in chess matches.
- **2019:** Leela Chess Zero (LCZero v0.21.1-nT40.T8.610) defeats Stockfish 19050918 in a 100-game match 53.5 to 46.5 for the Top Chess Engine Championship season 15 title.
- **Present:** Modern chess AI engines deploy deep learning to learn from thousands of matches. They regularly have FIDE ratings, chess' rating system, above 3,400, far beyond the best human players.

How did they do it?

- You could, in theory, try every possible move from every possible board position each time it's your turn
- In practice, there are probably more moves to check than there have been seconds since the Big Bang
  - Seconds since the Big Bang: 4.36 x ^{17}
  - $2^{59} = 5.76 \times 10^{17}$
- Deep Blue does do α-β pruning, but if it doesn't go all the way to the end of the game, what is the score it's using to prune?

Heuristics

- “A **heuristic function**, also simply called a **heuristic**, is a function that ranks alternatives in search algorithms at each branching step based on available information to decide which branch to follow. For example, it may approximate the exact solution.”--Wikipedia
- Deep Blue didn’t search for a winning move. Deep Blue searched for the board position that was modeled as having the highest winning “score”

The secret sauce

Bottom line: If you can’t solve a problem, try and find a problem you *can* solve that’s close enough

So what?

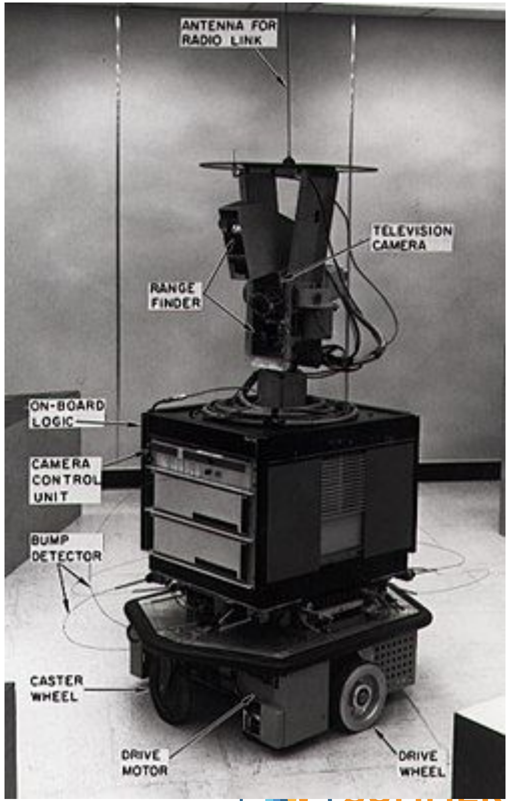
Artificial Intelligence is the field of formalizing real world problems and searching (efficiently) for a good enough solution

- Economics
  - Pricing strategies
  - Impact of regulation
- Auctions
- Resource usage
- Public policy
- Networks
- Social sciences
- Military strategy
- Evolution
- ...

SUMMER SPRINGBOARD
Look Inward. Go Upward.

Searching for things that aren't games

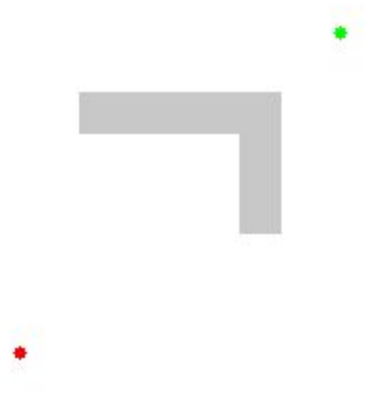
- Shakey the Robot (roughly 1966-1972) "was the first general-purpose mobile robot able to reason about its own actions. While other robots would have to be instructed on each individual step of completing a larger task, Shakey could analyze commands and break them down into basic chunks by itself."
  – Wiki
- Shakey would be put in a room with obstacles and given an objective to move towards. How would it pick it's path?
- Options:
  - Complete traversal of the space
  - Guess and check w/ stochasticity (a la Roombas)
  - A* Search: Move in the directions minimizes
    - The number of steps from the beginning, plus
    - A heuristic (the distance between the next point and the target)



SPRINGBOARD
Look Inward. Go Upward.

upload.wikimedia.org/wikipedia/commons/d/de/A_Star_Algorithm.webm

- Weight given to heuristic function is increased from left to right



“Classical” approaches to AI

- Graph Search
- Knowledge Representation
- Evolutionary Programming

What does “meaning” mean?

SUMMER SPRINGBOARD
Look Inward. Go Upward.

Figuring it out (like a boss)

SUMMER SPRINGBOARD
Look Inward. Go Upward.

Expert Systems

- An expert system is a form of artificial intelligence designed to **emulate** (and enhance) human **reasoning**
- Expert systems were very successful in the 70s and 80s. At one point they were *the future* of artificial intelligence
- Expert systems are are still widely used today

SUMMER SPRINGBOARD
Look Inward. Go Upward.

How they work

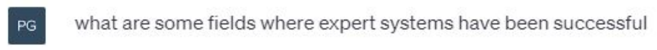
- Expert systems are composed of two parts
  1. Knowledge base
     - Facts
     - Implications
  2. Inference engine
- The user inputs statements that may or may not be true according to the knowledge base
- The inference engine checks
  1. Whether the facts are true, and
  2. What therefore must also be true

Video Game Recommendation Expert System

https://golb.us/summerspringboardai/expert_system/

SUMMER SPRINGBOARD
Look Inward. Go Upward.

Real world applications


PG what are some fields where expert systems have been successful

- **Healthcare**: Expert systems have been widely used for diagnostic purposes. An example is MYCIN, an early expert system developed in the 1970s that helped identify bacteria causing severe infections and suggested appropriate antibiotics. **Drug interaction effects.**
- **Finance**: Expert systems are used in financial institutions for credit scoring, fraud detection, stock trading, and investment advising. For instance, they can analyze financial data to assess the creditworthiness of an applicant or identify unusual transactions that may indicate fraud.
- **Engineering**: Expert systems are used for fault detection in machinery, control of manufacturing processes, and design of complex systems. For example, the CADUCEUS system was used for complex circuit board design.
- **Geology**: Expert systems like PROSPECTOR were used in the 1970s and 1980s to aid mineral exploration by evaluating geological data and suggesting promising sites for further investigation.
- **Information Technology**: Expert systems can help configure complex computer systems and networks. XCON, developed for Digital Equipment Corporation, is a classic example, assisting in the configuration of large computer systems.
- **Customer Service**: Expert systems are used in the form of chatbots to respond to customer queries, guide users in troubleshooting, and provide recommendations based on the customer's needs and preferences.
- **Legal**: Expert systems can assist in interpreting and applying legal rules. For instance, they can help users understand complex regulations, assess eligibility for certain benefits, or evaluate the potential outcomes of different legal strategies.

SUMMER SPRINGBOARD
Look Inward. Go Upward.

What do I know and how do I know it?

SUMMER SPRINGBOARD
Look Inward. Go Upward.

Metaphysics

- Epistemology:
  - What can we know? What does it mean to know something? How do we know things? How do we know that we know?
- Ontology:
  - What exists? What can be said to exist? What are the possible relationships between things that exist?

Ontologies in Computer Science

- How can I accurately record what I know about what exists and the relationships between them?
- This looks *a lot* like object-oriented programming / class-based design

Video Games

- Class(Game)
- Class(Genre)
- HAS-A(Game, Genre)
- IS-A(Street Fighter II, Game)
- Street Fighter II:
    - Genre: Fighter
- IS-A(Peter, Player)
- Here’s where the magic happens!
    - Plays(Player, Game)

Programming is (literally) magic

SUMMER SPRINGBOARD
Look Inward. Go Upward.

Imagine...

- Someone sits down in a special chair at a special table in a special room in a special building
- While wearing a special robe, they
- Speak a formulaic incantation while everyone listens with rapt attention
- Then they pick up a special hammer
- And bang it on a special block on the special table
- And then
- ...
- People appear and drag you away

What would you call that?

SUMMER SPRINGBOARD
Look Inward. Go Upward.

Imagine...

- While sitting in front of a special box
- You press some buttons while moving something around in circles
- And then
- ...
- A few days later...
  - A package arrives at your house
- At the end of the month...
  - A bank tells you that you owe them money

What would you call that?

SUMMER SPRINGBOARD
Look Inward. Go Upward.

The layers of meaning

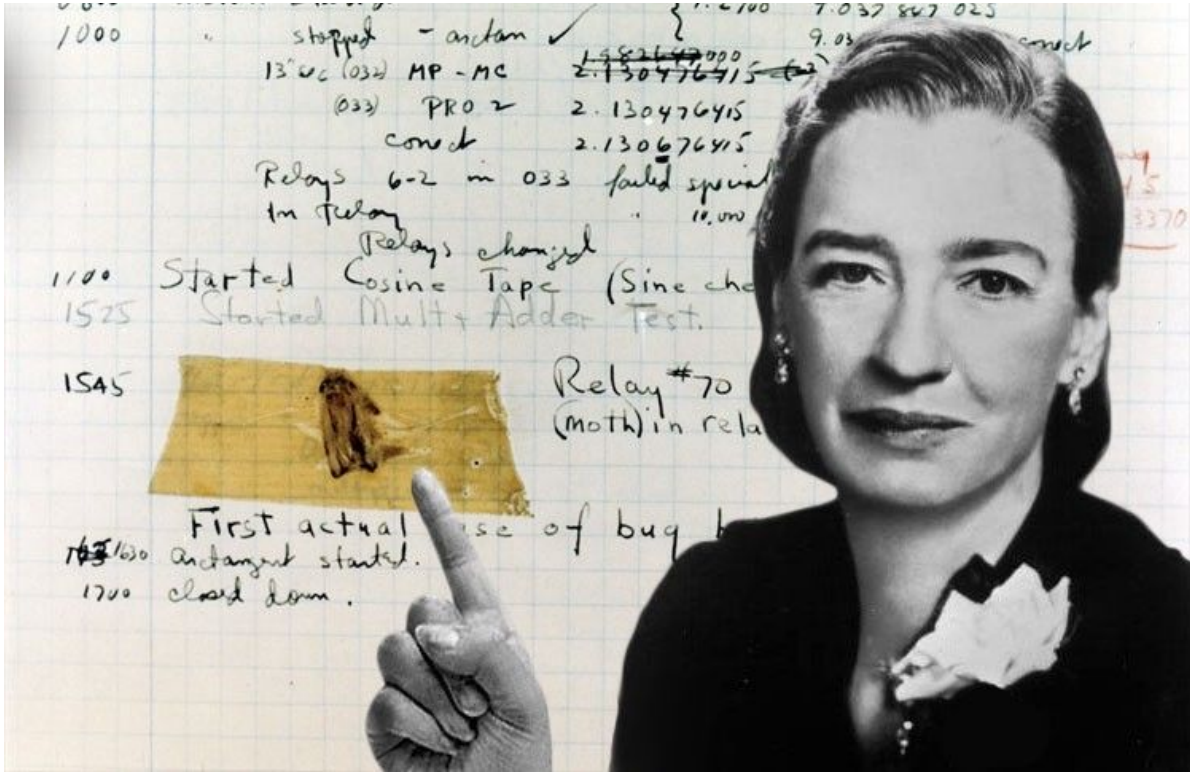
1. Lexicon: What are the words?
2. Syntax: How do I combine the words into sentences?
3. Semantics: What do the sentences mean?
    - For artificial languages, the syntax IS the semantics

Rear Admiral Grace Hopper

- In the early 1950s, Rear Admiral Grace Hopper set out to invent a middle ground between machine code and natural language
- Everyone thought that was ridiculous
- They were wrong

SUMMER
PRINGBOARD
ok Inward. Go Upward.

She was on the team with the first bug



1000 stopped - arctan ✓ 9.03 correct

13"uc (032) MP - MC ~~1.982147000~~ ~~2.130476415~~

(033) PRO 2 2.130476415

const 2.130676415

Relays 6-2 in 033 failed special

in Relay 10,000

Relays changed

1100 Started Cosine Tape (Sine che

1525 Started Mult+ Adder Test.

1545 Relay #70 (moth) in rela

First actual se of bug b

~~143~~ 1630 Arctangent started.

1700 closed down.

Where does (artificial) meaning come from?

- RELATION(Instance, Instance) is just syntax
  - It’s just an instruction that a computer can follow, no more no less
- “Peter plays Steet Fighter 2” is a sentence
  - This has a meaning intelligible to anyone who speaks English
- PLAYS(Peter, Street Fighter 2) is
  - An instruction that a computer can follow
  - That has a meaning intelligible to anyone who speaks English.
- Let me say that again

SUMMER SPRINGBOARD
Look Inward. Go Upward.

Where does (artificial) meaning come from?

- RELATION(Instance, Instance) is just syntax
  - It’s just an instruction that a computer can follow, no more no less
- “Peter plays Steet Fighter 2” is a sentence
  - This has a meaning intelligible to anyone who speaks English
- PLAYS(Peter, Street Fighter 2) is
  - An instruction that a computer can follow
  - That has a meaning intelligible to anyone who speaks English.

SUMMER SPRINGBOARD
Look Inward. Go Upward.

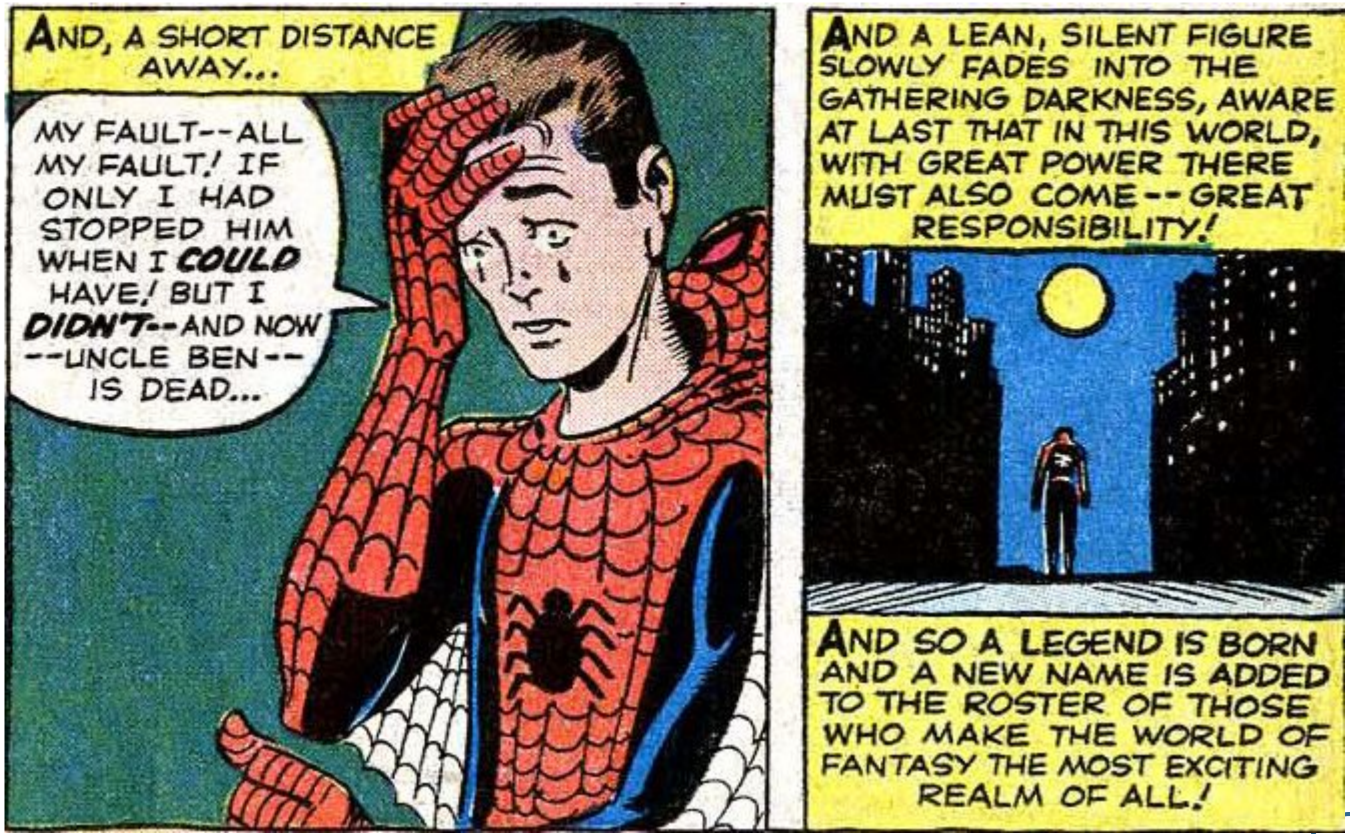
Programmers speak reality into being by naming the relationships between data

No one is asking the right question

- All of a sudden, it feels like anything is possible
- We are asking ourselves: “What can I do?”
- We need to ask ourselves: “What *should* I do?”
- “What **shouldn’t** I do?”

“After the thing went off, after it was a sure thing that America could wipe out a city with just one bomb, a scientist turned to Father and said, 'Science has now known sin.' And do you know what Father said? He said, 'What is sin?”

— Kurt Vonnegut, Cat's Cradle


AND, A SHORT DISTANCE AWAY...
MY FAULT--ALL MY FAULT! IF ONLY I HAD STOPPED HIM WHEN I COULD HAVE! BUT I DIDN'T--AND NOW --UNCLE BEN-- IS DEAD...
AND A LEAN, SILENT FIGURE SLOWLY FADES INTO THE GATHERING DARKNESS, AWARE AT LAST THAT IN THIS WORLD, WITH GREAT POWER THERE MUST ALSO COME--GREAT RESPONSIBILITY!
AND SO A LEGEND IS BORN AND A NEW NAME IS ADDED TO THE ROSTER OF THOSE WHO MAKE THE WORLD OF FANTASY THE MOST EXCITING REALM OF ALL!

SUMMER SPRINGBOARD
Look Inward. Go Upward.

SUMMER SPRINGBOARD
Look Inward. Go Upward.

“Classical” approaches to AI

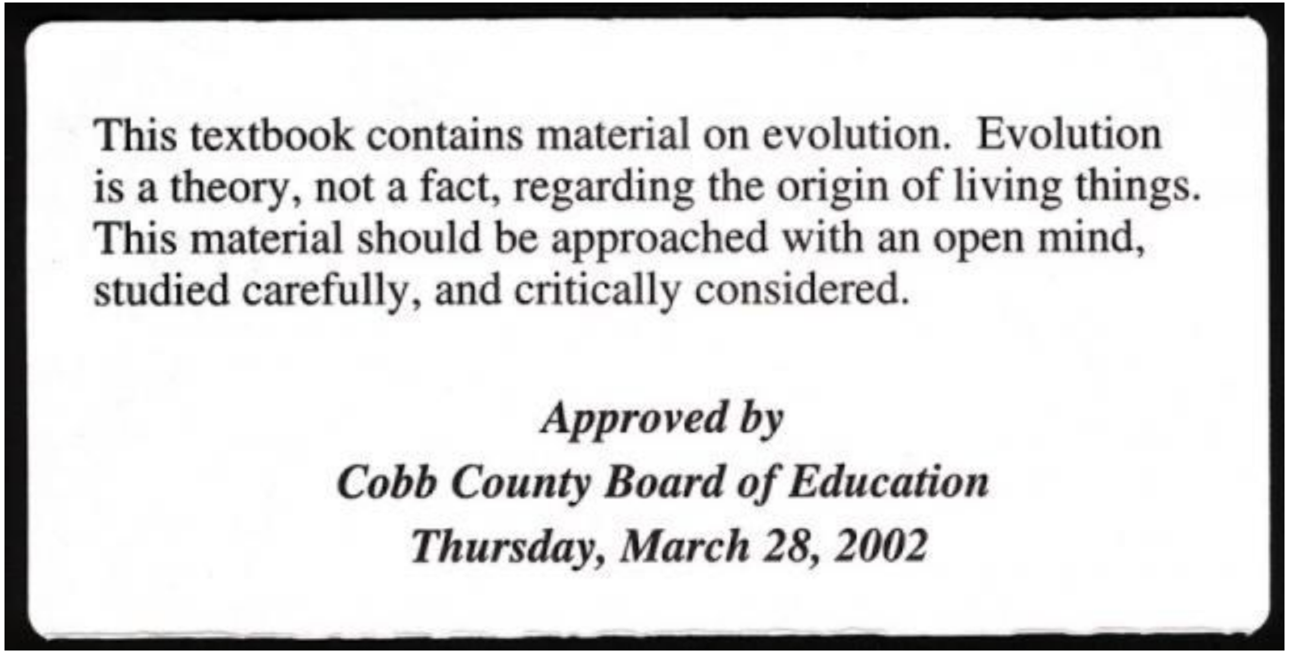
- Graph Search
- Knowledge Representation
- Evolutionary Programming

SUMMER SPRINGBOARD
Look Inward. Go Upward.

Teaching Solutions to

Collaborate

SUMMER SPRINGBOARD
Look Inward. Go Upward.



This textbook contains material on evolution. Evolution is a theory, not a fact, regarding the origin of living things. This material should be approached with an open mind, studied carefully, and critically considered.

Approved by
Cobb County Board of Education
Thursday, March 28, 2002

SUMMER SPRINGBOARD
Look Inward. Go Upward.

This is such an amazing encapsulation of the scientific method!

SUMMER SPRINGBOARD
Look Inward. Go Upward.

Facts vs Theories

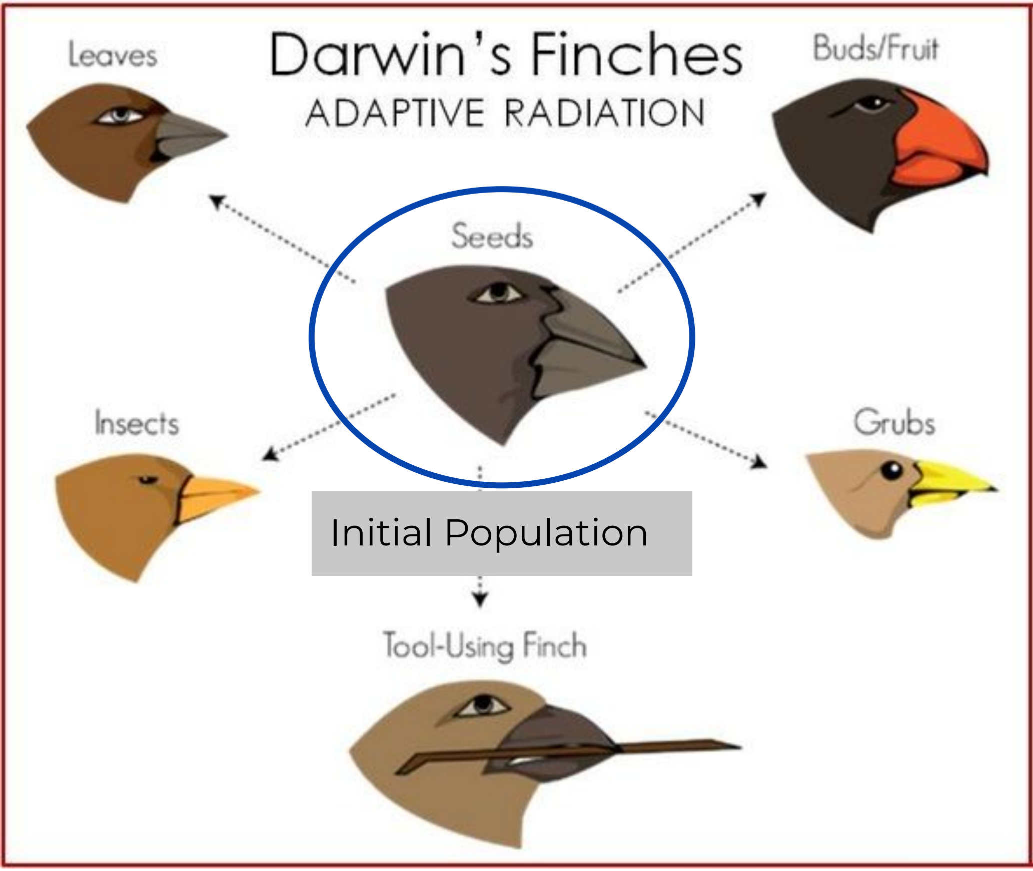
- Facts are true or false
  - If I drop something, it will fall at a rate of 32ft/sec^2
- Facts are not meaningful and / or useful
- Theories are meaningful and / or useful
  - Newtonian gravity
  - General relativity
- Theories are not true or false
  - Those can't *both* be true
- Theories are more or less accurate
- Theories are more or less useful
  - Newtonian gravity helps me use rockets to navigate to the moon
  - General relativity helps me use GPS to navigate around town

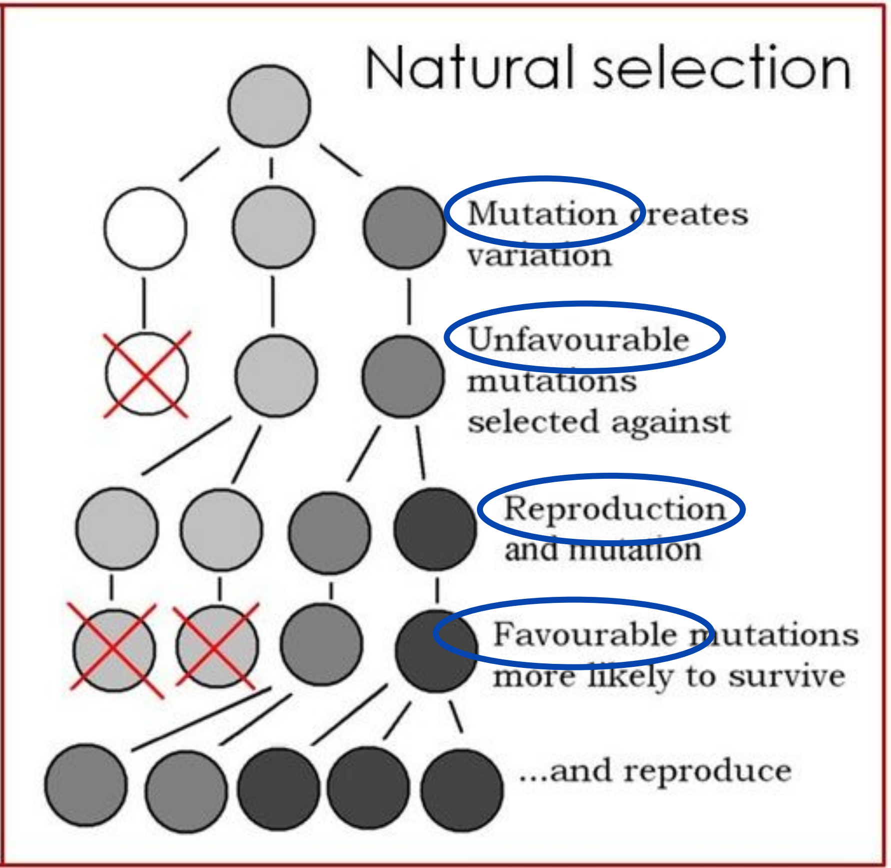
SUMMER SPRINGBOARD
Look Inward. Go Upward.

Evolution is a Theory, not a Fact

SUMMER SPRINGBOARD
Look Inward. Go Upward.

Evolution is a Theory, not a Fact
Natural Selection is a Fact, not a Theory





Natural selection
Mutation creates variation
Unfavourable mutations selected against
Reproduction and mutation
Favourable mutations more likely to survive
...and reproduce

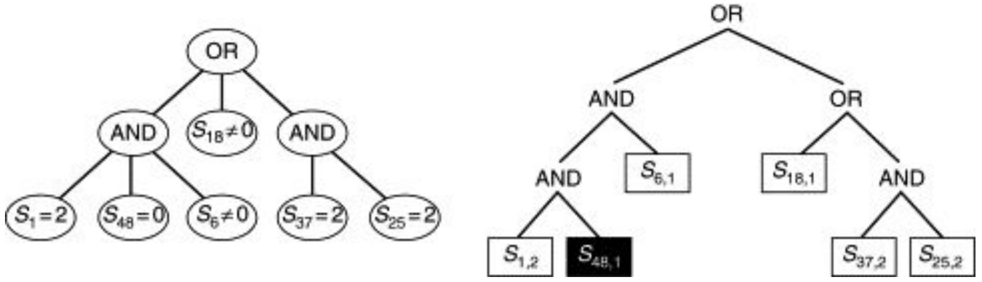
SPRINGBOARD
Look Inward. Go Upward.

Components of an Evolutionary Program

- Our goal, as always, is to search through the space of possible solutions efficiently
- An Evolutionary Program needs to implement the following five components
  1. Generation
     - Generate potential solutions
  2. Fitness function
     - Heuristic that measures “favorability”
  3. Selection
     - Choose which individuals will contribute to the next generation
  4. Mutation
     - Randomly perturb a potential solution
  5. Crossover
     - Merge “parent” solutions into “child” solutions

Two (Main) kinds of Evolutionary Programs

1. Genetic Algorithms: Search for the best solution, usually as a bitstring
2. Genetic Programs: Search for the best implementation
   - This tends to be a good way to design hardware



Genetic Algorithms

- Components
    - Generation: flip a fair coin n times
    - Selection: pick population / 2 pairs, weighted by fitness
    - Crossover: randomly select pairs of individuals
        i. Identify a split point
        ii. Swap the values to the left of the split point

Crossover

Fitness function: count the number of 1s

1. Pick two individuals

```
01001101: 4
10100100: 3
```

2. Identify a split point

```
010 | 01101: 4
101 | 00100: 3
```

3. Swap the values to the left of the split point

```
101 | 01101: 5
010 | 00100: 2
```

Genetic Algorithms

- Components
  - Generation: flip a fair coin n times
  - Selection: pick pairs weighted by fitness population / 2 times
  - Crossover: randomly select pairs of individuals
    1. Identify a split point
    2. Swap the values to the left of the split point
  - Mutation: for each individual
    1. Generate a random number for each bit
    2. Flip the bit if the number is less than some mutation rate

SUMMER SPRINGBOARD
Look Inward. Go Upward.

Genetic Algorithms

- Components
  - Generation: flip a fair coin n times
  - Selection: pick pairs weighted by fitness population / 2 times
  - Crossover: randomly select pairs of individuals
    1. Identify a split point
    2. Swap the values to the left of the split point
  - Mutation: for each individual
    1. Generate a random number for each bit
    2. Flip the bit if the number is less than some mutation rate
  - Fitness function: <u>your problem goes here</u>

SUMMER SPRINGBOARD
Look Inward. Go Upward.

Canonical Problems

- Find a string consisting of all 1s
  - Fitness function: count the number of 1s
- The (0-1) knapsack problem
  - Given a set of items with weights and values, and a backpack with a maximum weight capacity
  - Select the subset of items that maximize the value and fit in the backpack
- Approaches to the knapsack problem
  - Exact: dynamic programming
  - Approximate:
    - Greedy Algorithm
    - Genetic Algorithm

SUMMER SPRINGBOARD
Look Inward. Go Upward.

Which do you think matters more: Mutation or Crossover?

SUMMER
SPRINGBOARD
Look Inward. Go Upward.

Knapsack problem set-up

- `items` is an array of weight / value pairs
    - `[(weight, value)]`
- An individual (a knapsack) is a bitstring where a 1 in position `i` indicates that `item[i]` is in the knapsack
- Your challenge: implement a fitness function that causes a genetic algorithm to approximate the solution to the knapsack problem
- Try different values for mutation rate and crossover rate. Which makes a bigger difference?

In general

- Crossover matters more for Genetic Algorithms
- Mutation matters more for Genetic Programs

SUMMER SPRINGBOARD
Look Inward. Go Upward.

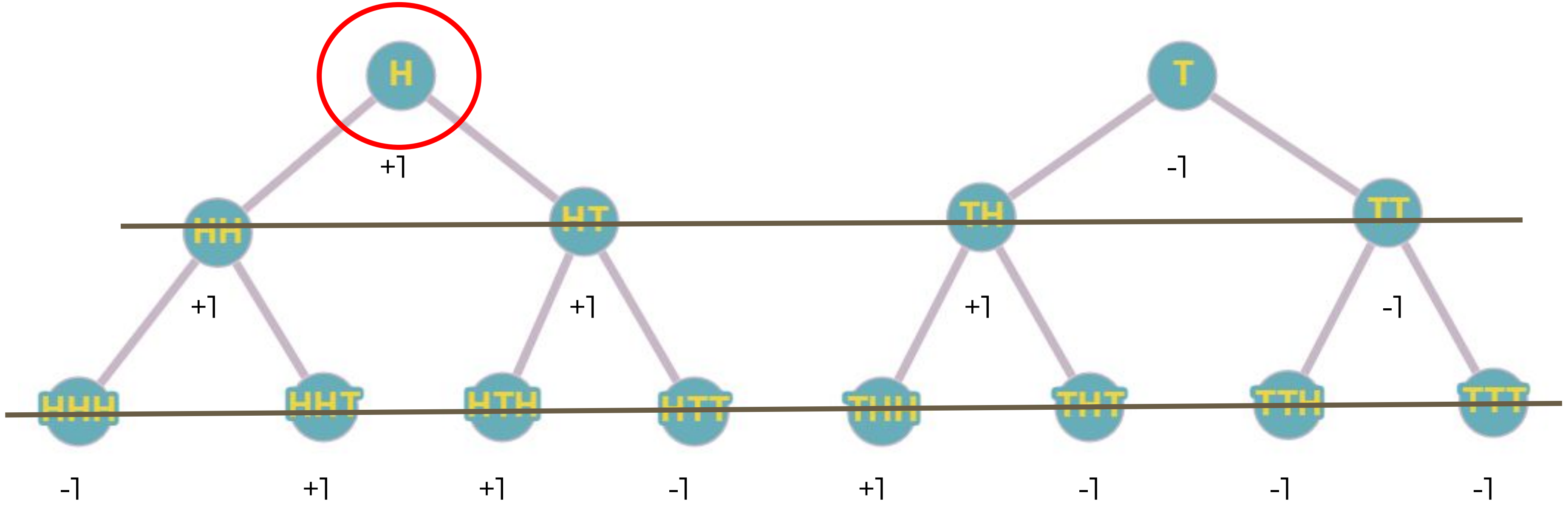
Classical AI

Recap

“Classical” approaches to AI

- Graph Search
- Knowledge Representation
- Evolutionary Programming

The Minimax Algorithm: How to play Chess (slowly)



My turn: MAX
Their turn: MIN
My turn: MAX

Heuristics

- “A **heuristic function**, also simply called a **heuristic**, is a function that ranks alternatives in search algorithms at each branching step based on available information to decide which branch to follow. For example, it may approximate the exact solution.”--Wikipedia
- Deep Blue didn’t search for a winning move. Deep Blue searched for the board position that was modeled as having the highest winning “score”

The secret sauce

Bottom line: If you can’t solve a problem, try and find a problem you *can* solve that’s close enough

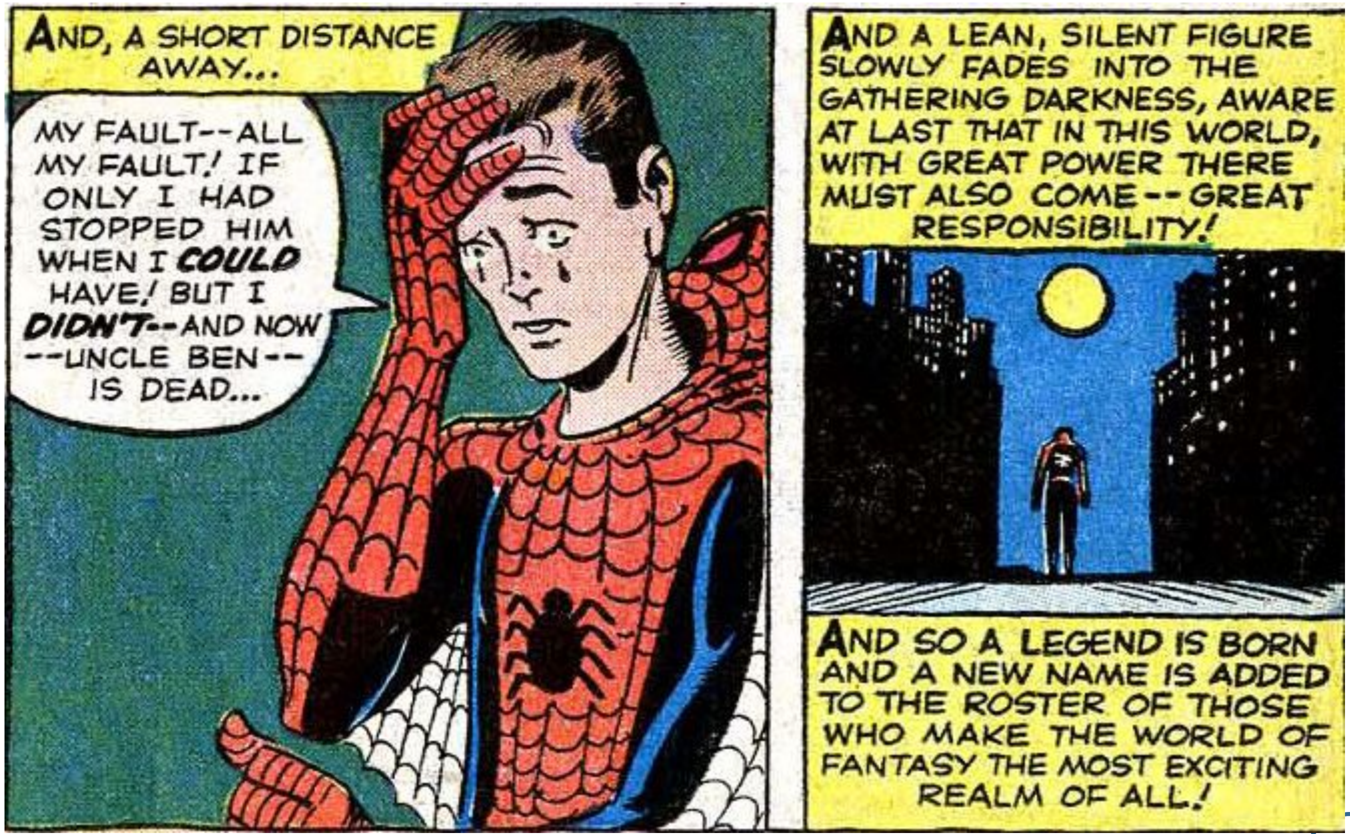
SUMMER SPRINGBOARD
Look Inward. Go Upward.

Video Games

- Class(Game)
- Class(Genre)
- HAS-A(Game, Genre)
- IS-A(Street Fighter II, Game)
- Street Fighter II:
  - Genre: Fighter
- IS-A(Peter, Player)
- Here’s where the magic happens!
  - Plays(Player, Game)

SUMMER SPRINGBOARD
Look Inward. Go Upward.

Programmers speak reality into being by naming the relationships between data

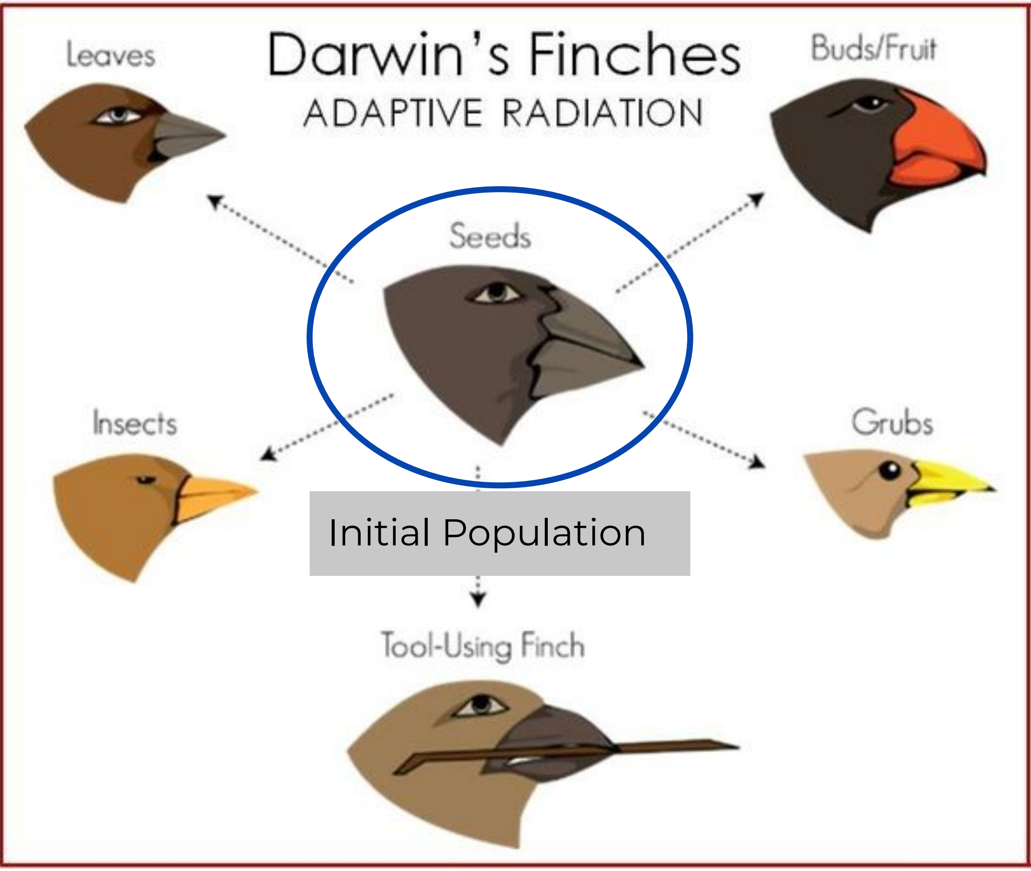

AND, A SHORT DISTANCE AWAY...
MY FAULT--ALL MY FAULT! IF ONLY I HAD STOPPED HIM WHEN I COULD HAVE! BUT I DIDN'T--AND NOW --UNCLE BEN-- IS DEAD...
AND A LEAN, SILENT FIGURE SLOWLY FADES INTO THE GATHERING DARKNESS, AWARE AT LAST THAT IN THIS WORLD, WITH GREAT POWER THERE MUST ALSO COME--GREAT RESPONSIBILITY!
AND SO A LEGEND IS BORN AND A NEW NAME IS ADDED TO THE ROSTER OF THOSE WHO MAKE THE WORLD OF FANTASY THE MOST EXCITING REALM OF ALL!

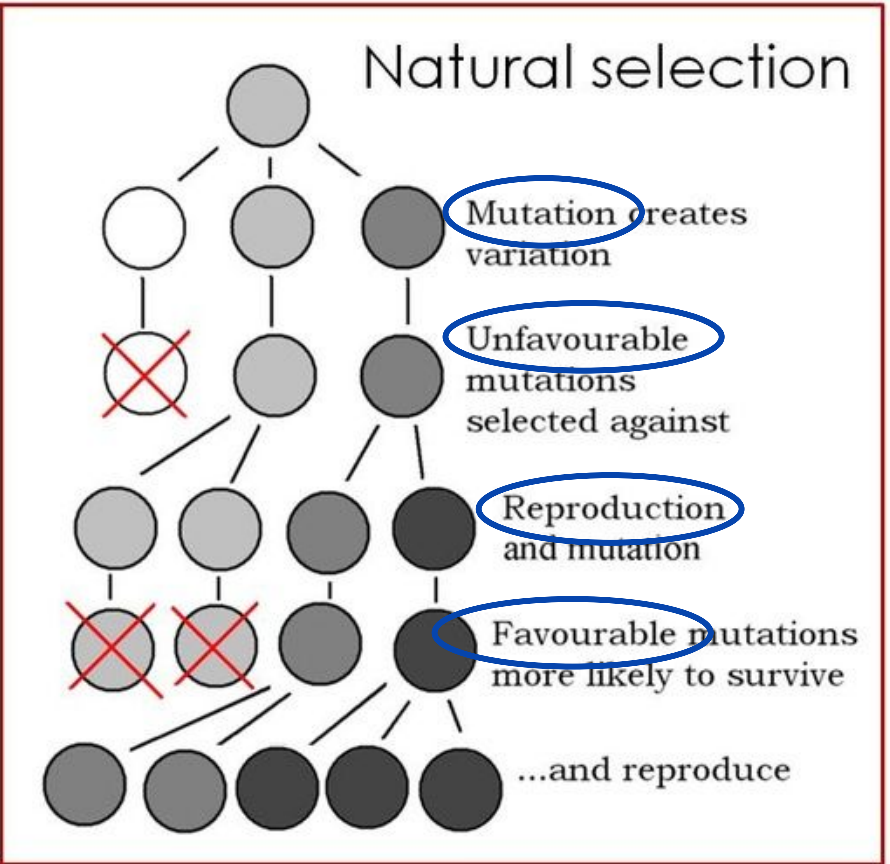
SUMMER SPRINGBOARD
Look Inward. Go Upward.

Facts vs Theories

- Facts are true or false
    - If I drop this pen, it will fall at a rate of 32ft/sec^2
- Facts are not meaningful and / or useful
- Theories are meaningful and / or useful
    - Newtonian gravity
    - General relativity
- Theories are not true or false
    - Those can't *both* be true
- Theories are more or less accurate
- Theories are more or less useful
    - Newtonian gravity helps me use rockets to navigate to the moon
    - General relativity helps me use GPS to navigate around town

SUMMER SPRINGBOARD
Look Inward. Go Upward.




Natural selection
Mutation creates variation
Unfavourable mutations selected against
Reproduction and mutation
Favourable mutations more likely to survive
...and reproduce

SPRINGBOARD
Look Inward. Go Upward.

Genetic Algorithms

- Components
    - Generation: flip a fair coin n times
    - Selection: pick pairs weighted by fitness population / 2 times
    - Crossover: randomly select pairs of individuals
        1. Identify a split point
        2. Swap the values to the left of the split point
    - Mutation: for each individual
        1. Generate a random number for each bit
        2. Flip the bit if the number is less than some mutation rate
    - Fitness function: <u>your problem goes here</u>