A Developers Survey of Artificial Intelligence

BARRY S. STAHL - @BSSTAHL

Barry Stahl
@bsstahl

I think it is noteworthy that I am the type of person who has both favorite physicists and favorite mathematicians.

8:35 PM - 16 Apr 2017 from Phoenix, AZ

About Me

Favorite Physicists	Favorite Mathematicians
Harold “Hal” Stahl	Ada Lovelace
Carl Sagan	Alan Turing
Neil Degrasse Tyson	Johannes Kepler
Nikola Tesla	René Descartes
Marie Curie	Isaac Newton
Richard Feynman	Leonardo Fibonacci
Albert Einstein	George Boole

Other notables: Niels Bohr, Galileo Galilei, Michael Faraday, Blaise Pascal, Johann Gauss, Grace Hopper, Stephen Hawking, Marvin Minsky, Daphne Koller, Benoit Mandelbrot

About Me

https://meetup.com/azgivecamp/

Join us!

What do I mean by “Artificial Intelligence”?

AI > ML

Artificial Intelligence — Machine Learning

What do I mean by “Artificial Intelligence”?

A Computational System that behaves rationally

1) Makes decisions
2) Attempts to make the best decision
   a) Best available understanding (model)
   b) Best available information (data)
3) May act on those decisions (automation)

What do I mean by “Artificial Intelligence”?

AI <> DSS

Artificial Intelligence

Decision Support Systems

When do I Need an AI?

When I need to make decisions at scale

- Many Decisions
- Very Quickly
- Large *Datasets*
- Large *Solution Space*

AIs are not (yet) great when the *Problem Space* is large

See http://www.cognitiveinheritance.com/post/Scalable-Decision-Making.aspx

What do I Need for an AI?

Data

- Information about the state of the problem space

Model

- Representation of the possible methods of making the decision

Test Data

- Method of validating the model

Types of AI Models

Logic

- Reducible to conditionals
  - *Object Oriented (everything we've ever done before)*
  - *Rules Engine*

Probabilistic/Learning

- Results in a prediction of best solution often derived from earlier data
  - *Neural/Bayesian Networks*
  - *Genetic Algorithms*

Search/Optimization

- Based on reducing and searching the *Solution Space*
  - *Dynamic Programming*
  - *Linear Programming*

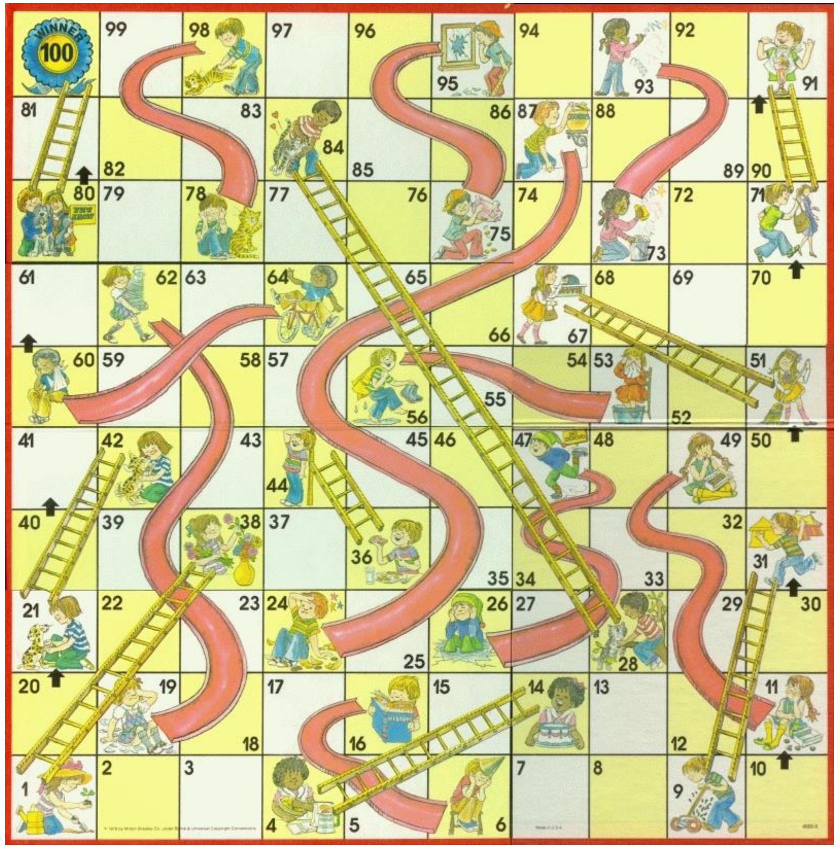
Logical Models

MODELS OF ARTIFICIAL INTELLIGENCE THAT REDUCE TO A SERIES OF CONDITIONALS

Features of Logical Models

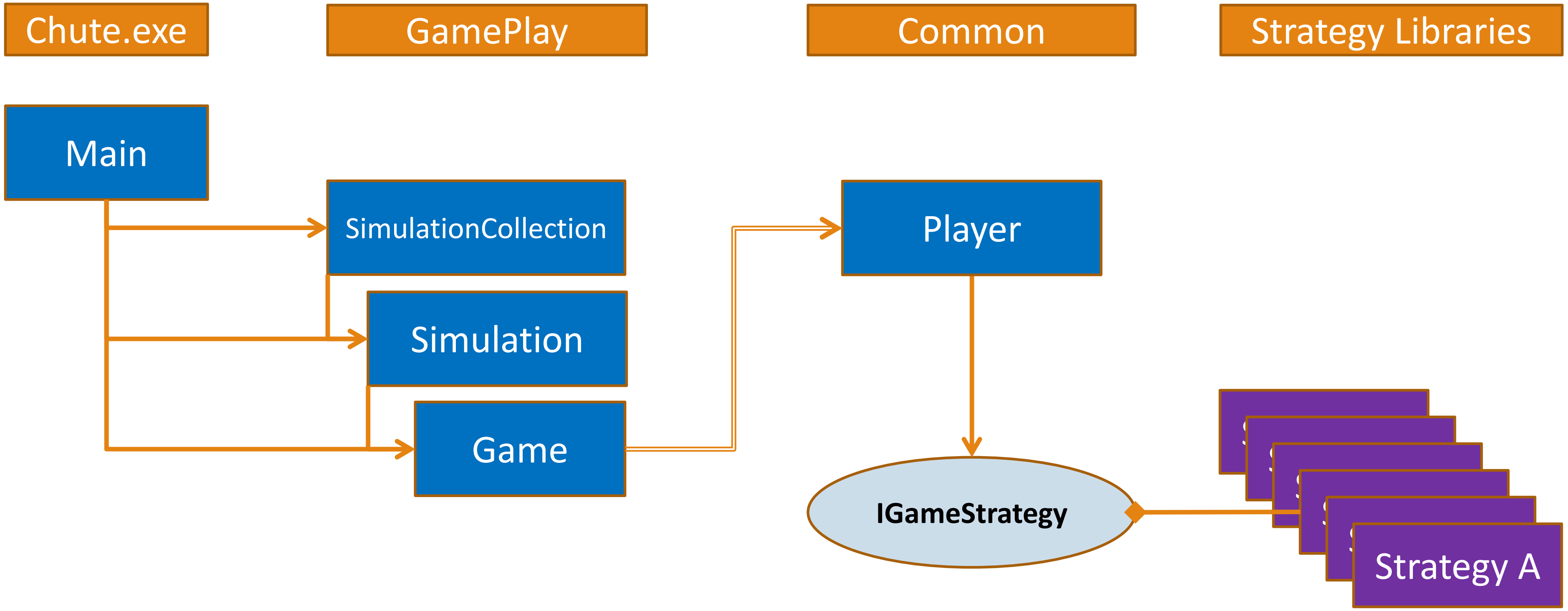
Feature	Expectation	Comments
Results are explainable	Generally	Code is highly imperative
Correctness is understood	Generally	Code is highly imperative
Easy to design	Sometimes	Solutions must be fully understood
Easy to build/maintain	Very	Most devs are comfortable with logic implementation
Solution Discoverability	Low	Solutions will only deliver pre-conceived answers
Works well	Problem & solution understood	Code is highly imperative

Rules of Chutes & Ladders



- For 2-6 Players
- All start at same space
- Start is random (0-25)
- Each player spins to determine how many spaces to move (1-6)
- Player has option to take or skip chute/ladder when leaving origin space
- If you can move the correct spaces, you must - if you can't, your turn is skipped

Project Structure



Logical Models in the Demo Code

Object-Oriented Logic

- Greedy Algorithm
  - Usually the best place to start
  - Selects the highest value space
- Linear Strategy
  - Never take a chute or ladder
  - Obviously not the best strategy
- Aggressively Bad Strategy
  - Selects the lowest value space
  - Can probably never win
  - Used as a hideous warning

Rules Engine

- Greedy
- Linear
- Take all Ladders

Probabilistic/Learning Models

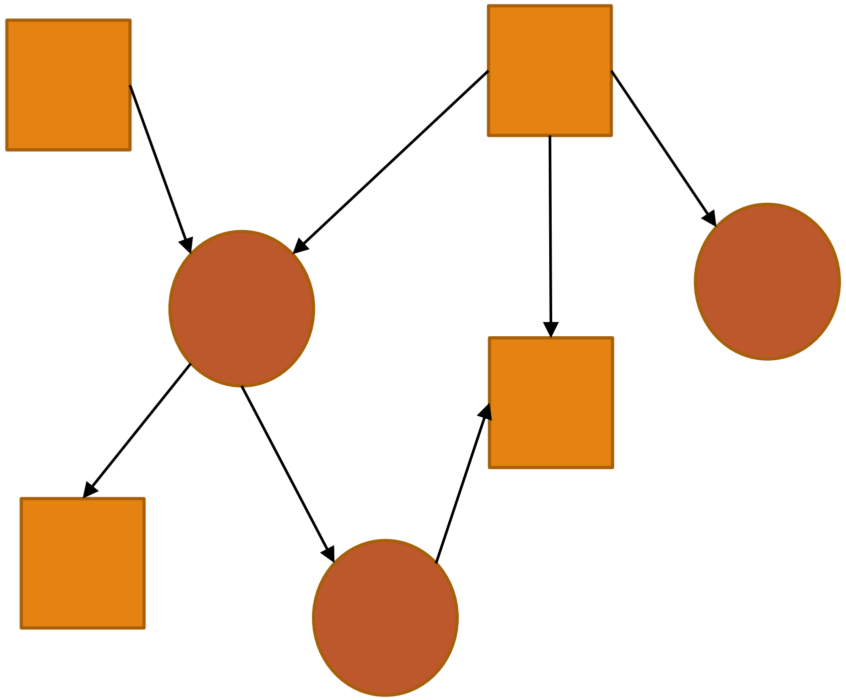
MODELS OF ARTIFICIAL INTELLIGENCE THAT RESULT IN A PREDICTION OFTEN DERIVED FROM EARLIER DATA

Types of Probabilistic/Learning Models

- Probabilistic Graphical Models – Represent conditional dependencies in a graph
  - Bayesian network
  - Markov Model
  - Clique Tree

- Evolutionary Algorithms – Evolve behavior based on natural models
  - Genetic Algorithms
  - Ant-Colony Optimization
  - Bees Algorithm

Probabilistic Graphical Models



- Conditional dependencies are indicated by edges
  - An undirected edge represents correlation
  - A directed edge represents causation
- Used for
  - Text Classification
  - Spam filtering
  - Recommendation systems
  - Causal Inference

Where to Eat Bot

BSStahl: I’d like to try Italian food for lunch

RobRich: Italian sounds good, can we meet in Chandler somewhere

DigitalDrummerJ: I need to be back to the office by 2pm

WhereToEatBot: *Floridino’s Pizza & Pasta* is an Italian restaurant in Chandler rated 4 of 5 stars with a table available at 12:30pm. Should I book it for you?

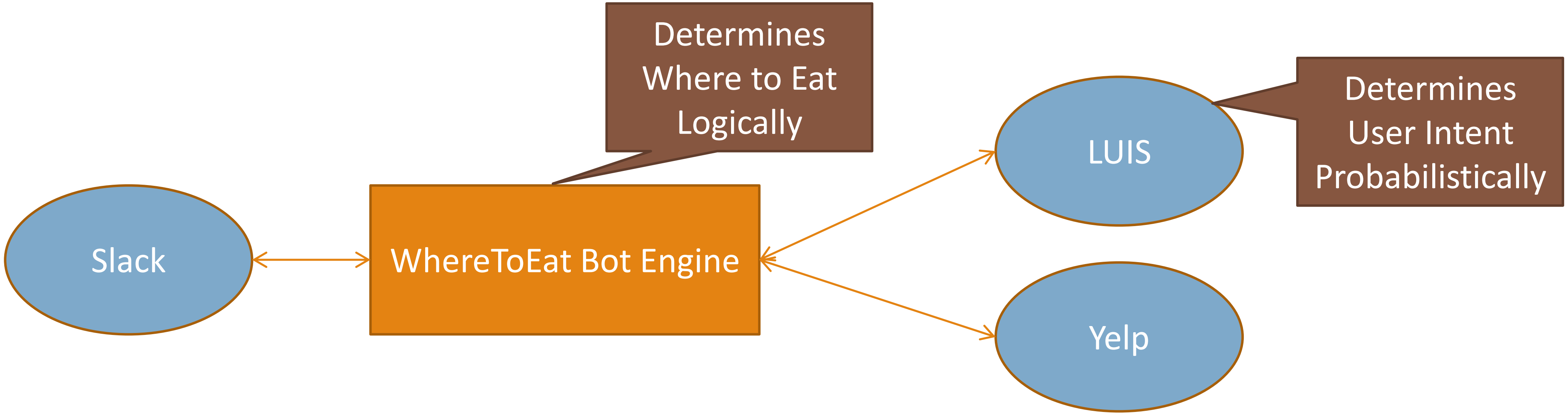
BSStahl: Yes, book it for 3 people

WhereToEatBot: Your reservation for 3 at Floridino’s is confirmed. Reservation code: AT7434

LUIS - Language Understanding Intelligent Service


```
{"query": "I'd like to try italian nearby",
  "topScoringIntent": {  "intent": "AddFocus",  "score": 0.249762028  },
  "intents": [
    { "intent": "AddFocus", "score": 0.249762028 },
    { "intent": "AddFilter",  "score": 0.233643577 },
    { "intent": "RemoveFilter", "score": 0.218146175 },
    { "intent": "RemoveFocus", "score": 0.00575467059 },
    {"intent": "None", "score": 0.004777815 },
    { "intent": "Start Conversation", "score": 0.0009793207 }
  ],
  "entities": [
    { "entity": "italian", "type": "Cuisine", "startIndex": 16, "endIndex": 22, "score": 0.6817578 }]}
```

Where to Eat Bot Process



Let's try Italian food for lunch — AddFocus Italian Cuisine

Let's meet in Chandler — AddFocus Chandler Location

I need to be back by 2pm — AddFocus 2pm Timeframe

Query: Italian in Chandler ending by 2pm

Decision: Floridino's at 12:30 pm

Features of Probabilistic Models

Feature	Expectation	Comments
Results are explainable	Rarely	Many models produce no indication whatsoever about why a solution was chosen
Correctness is understood	Somewhat	Predictable but variable
Easy to design	Somewhat	Some models can produce good results without a full understanding of the problem
Easy to build/maintain	No	Tools are available to help
Solution Discoverability	High	Solutions may surprise the original implementers
Works Well	Understanding is limited	Can help identify solutions even when the mechanism isn't fully understood

Probabilistic Models for Chutes & Ladders

- Features of this problem
  - Huge solution space
  - Non-deterministic system
    - Human opponent
    - Unpredictable start location
    - Unpredictable spins

- Possible implementations
  - PGM: Correlate moves -> wins
  - Genetic Algorithm: Evolve the strategy

Opportunity!

Genetic Algorithms

- Simulate Darwinian Evolution
  - Each candidate solution is defined by its properties (chromosomes)
  - A fitness function is used to determine which solutions “survive”
  - Surviving solutions may mutate and evolve other solutions
  - Optimality is never guaranteed

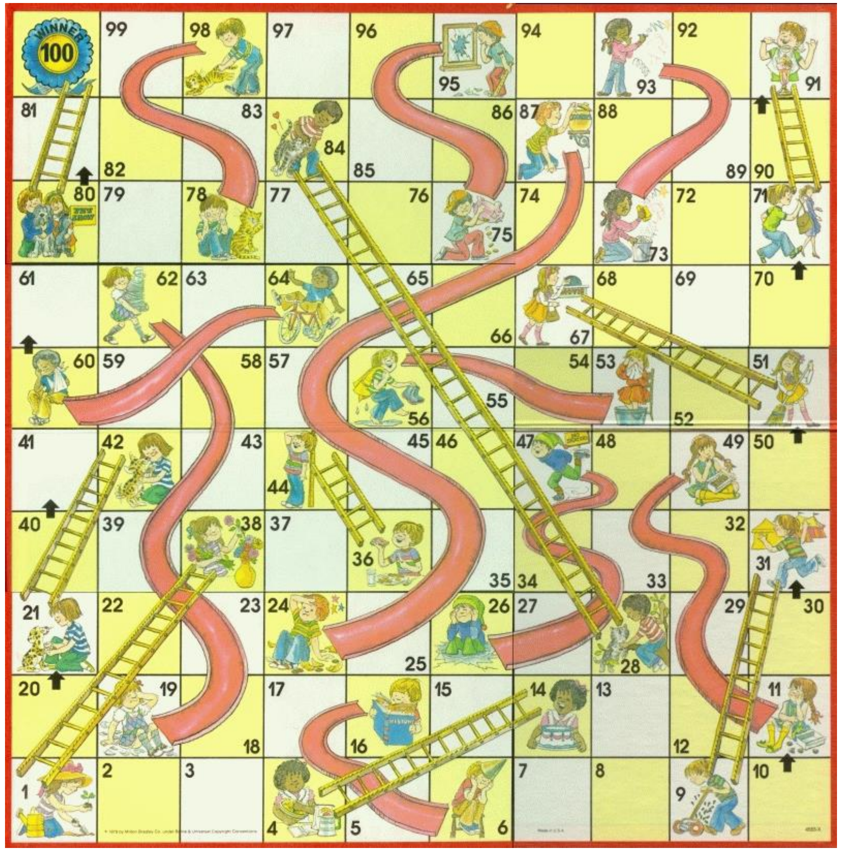
- Define the DNA of a Solution
  - Varies greatly by problem
  - Ideally, each option is a chromosome

- Define a Fitness Function
  - How do we know if the solution is good?
  - If we can’t define a good fitness function we probably can’t use a Genetic Algorithm

- Determine how the Solution Evolves
  - What solutions evolve and under what circumstances
  - How does the DNA change to evolve the solution

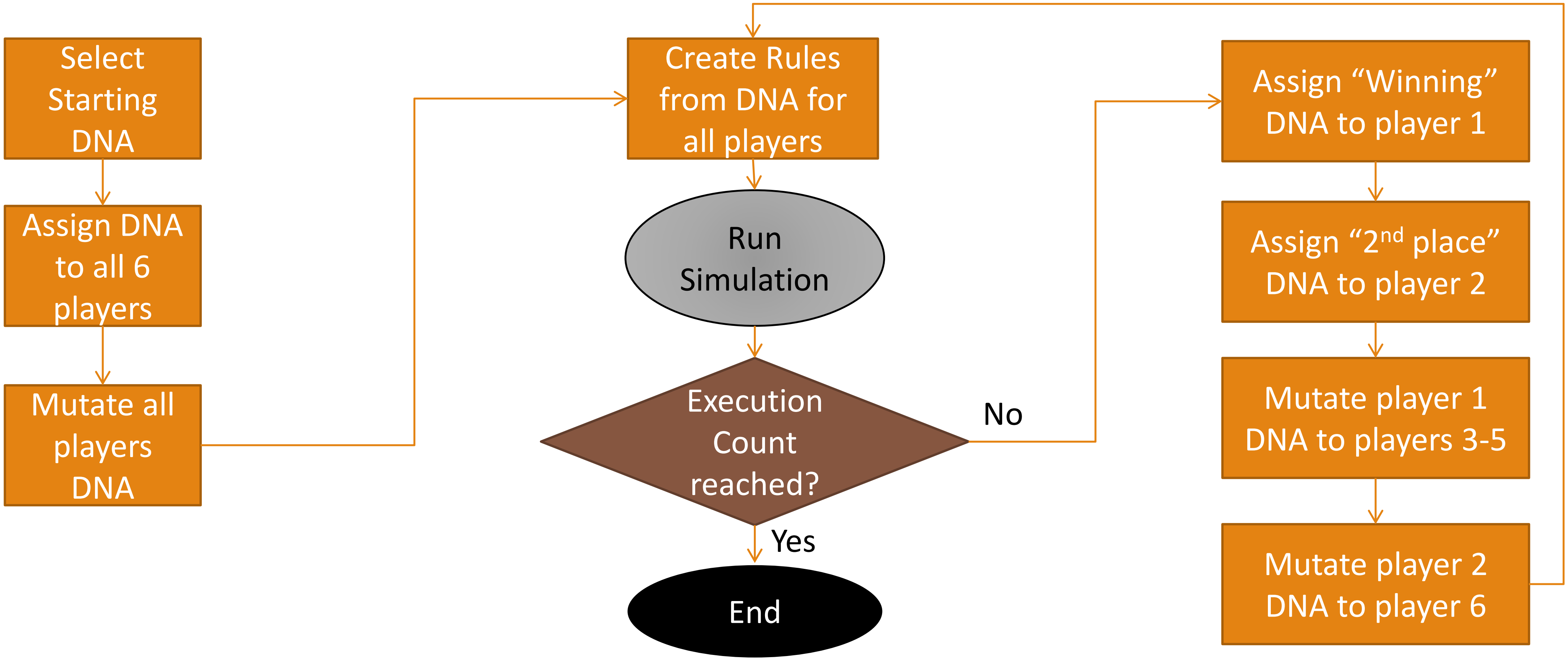
DNA of Chutes & Ladders

Starting Point	Spin	Option 1	Option 2	Option 3	Option 4	Option 5
45	3	48	26			
45	4	49	27			
45	5	50	11	28		
45	6	51	12	29	84	
46	2	48	26			
46	3	49	27			
46	4	50	11	28		
46	5	51	12	29	84	
46	6	52	67	13	30	85



299 Chromosomes
679 Total Selections
1.54×10^{103} Combinations

Our Genetic Algorithm



Search/Optimization Models

MODELS OF ARTIFICIAL INTELLIGENCE BASED ON REDUCING AND SEARCHING THE SOLUTION SPACE

Types of Search/Optimization Models

- Constraint Programming
  - Linear Programming
  - Mixed-Integer Programming
- Other Search Methods
  - Local Search
  - Branch and Bound
  - Dynamic Programming

Building AI Solutions with Google OR-Tools

Features of Search/Optimization Models

Feature	Expectation	Comments
Results are explainable	Sometimes	Highly dependent on implementation
Correctness is understood	Somewhat	Highly dependent on implementation
Easy to design	No	Requires a detailed knowledge of the problem to design the model
Easy to build/maintain	No	Tools are available to help
Solution Discoverability	Limited	Most implementations will powerfully limit the possibilities of the results returned
Works Well	Large solution space	Many optimization techniques function well with inconceivably large solution spaces

Optimization Models for Chutes & Ladders

- Mixed Integer Programming
  - Based on Constraint/Linear Programming
  - Uses boolean variables to indicate if an option is selected
  - Google OR-Tools can help implement

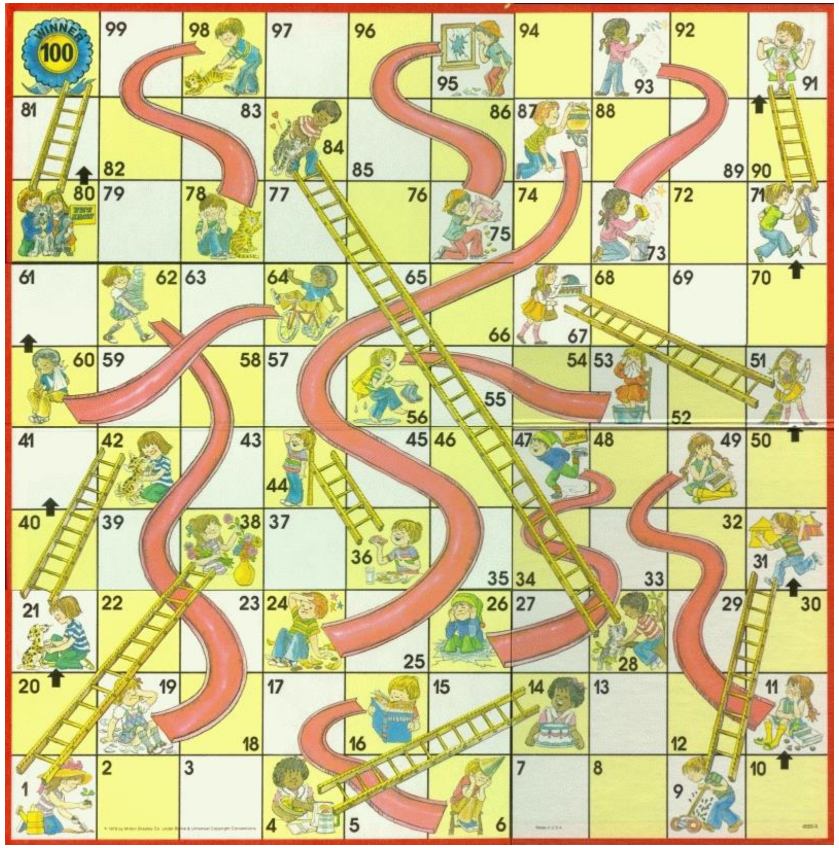
Opportunity!

- Shortest Path Algorithm
  - Dijkstra's Algorithm
    - Based on Dynamic Programming

Dynamic Programming

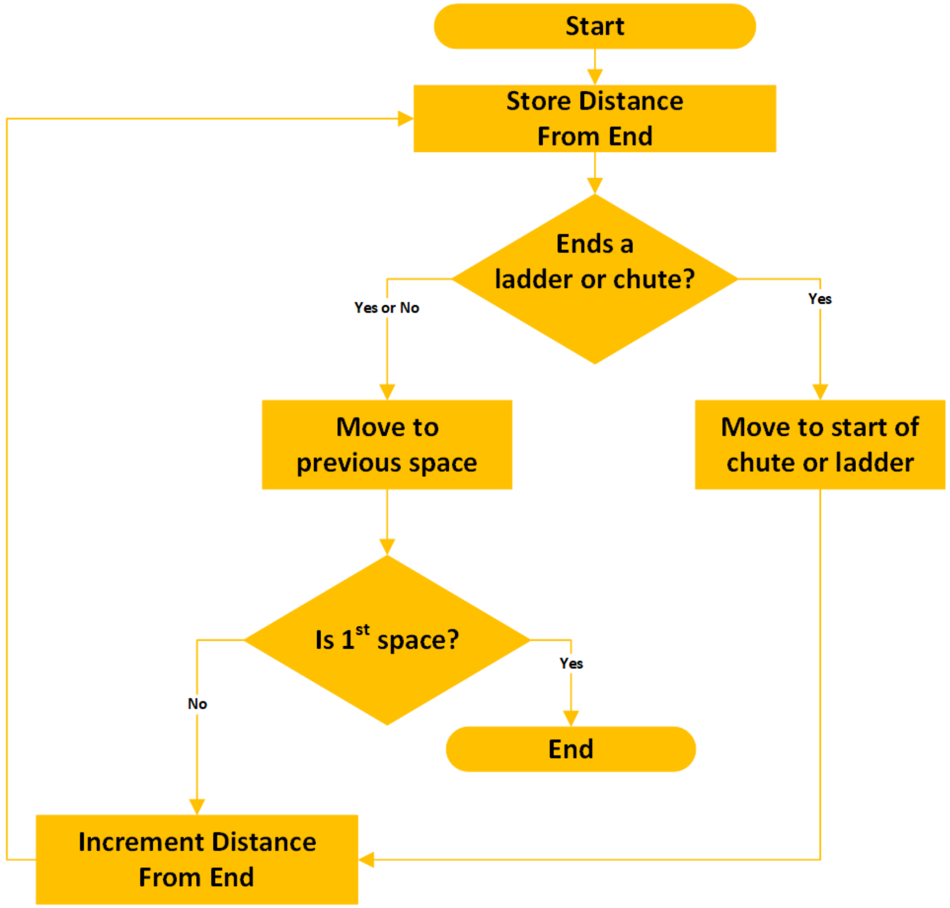
- Break a problem down into smaller, simpler subproblems
- Solve each subproblem only once, building on the previous
- Memoize (cache) the solution to each subproblem
- Guarantees an optimal solution
- Calculating the Fibonacci sequence is a good candidate for this technique

Dynamic Programming of Chutes & Ladders



00	01	02	03	04	05	06	07	08	09
19	18	17	16	15	14	13	12	11	10
01	02	03	04	05	06	07	08	09	10
20	19	18	17	16	15	14	13	12	11
21	22	23	24	25	26	27	28	29	15
25	24	23	22	21	20	19	18	17	16
26	27	28	29	23	24	25	26	27	28
24	23	22	21	20	19	18	17	30	29
25	26	27	28	29	30	31	32	33	34
29	34	33	32	33	32	31	30	29	35

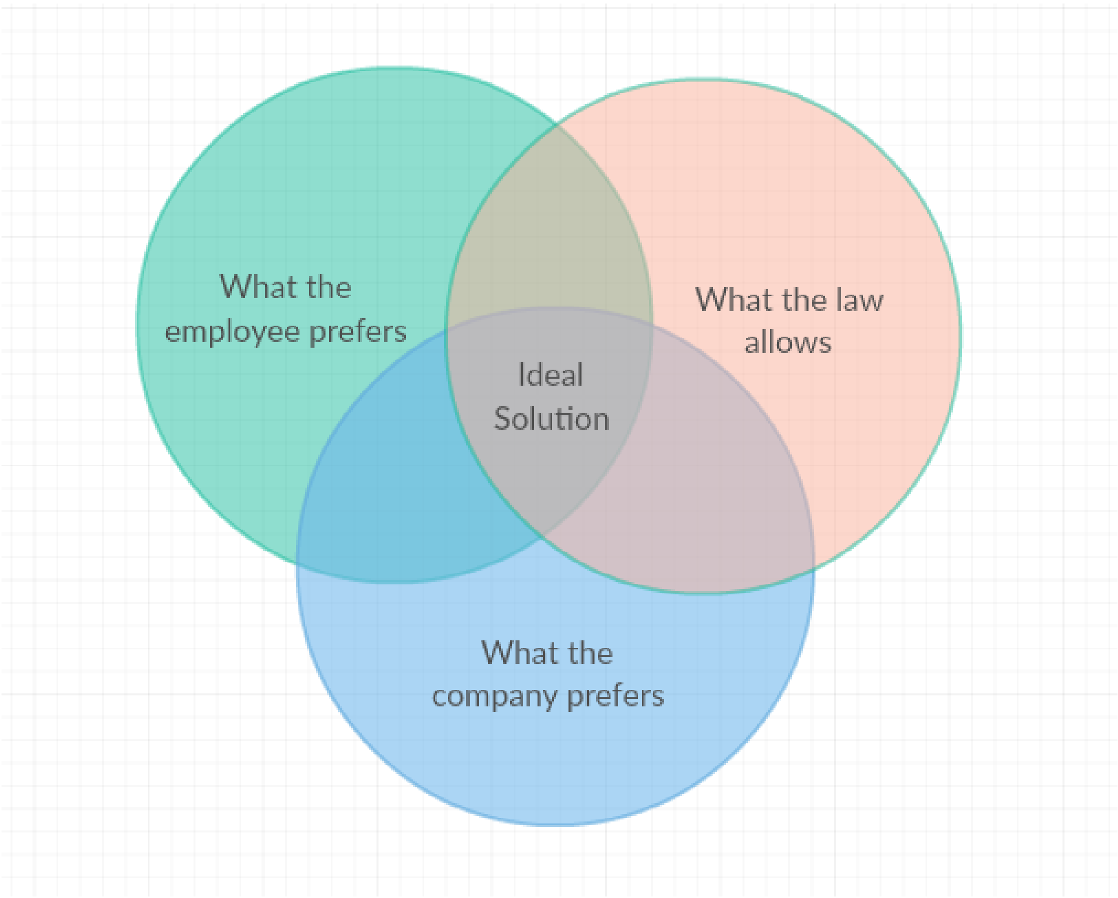
Dynamic Programming of Chutes & Ladders



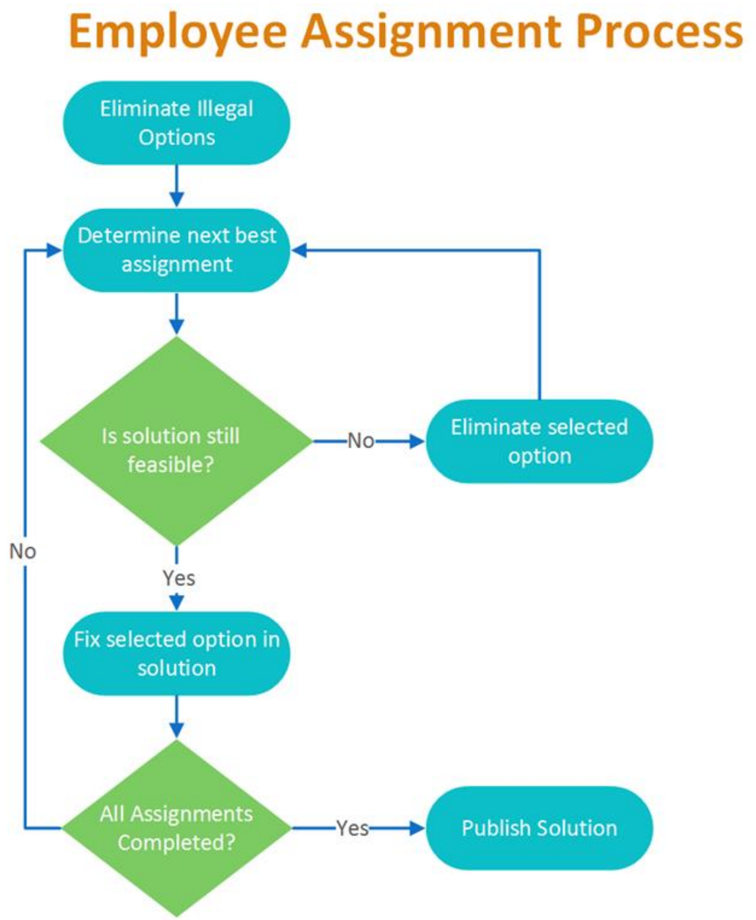
Hybrid Models

MODELS OF ARTIFICIAL INTELLIGENCE THAT COMBINE THE BEST OF MULTIPLE TECHNIQUES

Employee Assignment Problem



Employee Assignment Process



Features of AI Model Types

Feature	Logical	Probabilistic	Search/Optimization
Results are explainable	Generally	Rarely	Sometimes
Correctness is understood	Generally	Somewhat	Somewhat
Easy to design	Sometimes	Somewhat	No
Easy to build/maintain	Very	No	No
Solution Discoverability	Low	High	Limited
Works well	Problem & solution understood	Understanding is limited	Large solution space

Summary

Artificial Intelligence is about making automated decisions

Logical methods reduce the problem to conditionals

- *Object Oriented*
- *Rules Engine*

Probabilistic/Learning methods result in a prediction derived from earlier data

- *Neural/Bayesian Networks*
- *Genetic Algorithms*

Search/Optimization methods are based on reducing and searching the *solution space*

- *Dynamic Programming*
- *Linear Programming*

Hybrid methods allow us to take advantage of the best features of multiple model types

Resources

Code

- https://github.com/bsstahl/AIDemos

Articles

- http://www.cognitiveinheritance.com/post/Scalable-Decision-Making.aspx
- http://www.cognitiveinheritance.com/post/AI-That-Can-Explain-Why.aspx
- http://www.cognitiveinheritance.com/post/An-Example-of-a-Hybrid-AI-Implementation.aspx

Videos

- https://youtu.be/zZAobExOMB0

Courseware

- https://www.coursera.org/specializations/probabilistic-graphical-models
- https://www.coursera.org/learn/discrete-optimization

Tools

- https://azure.microsoft.com/en-us/services/cognitive-services/
- https://www.ibm.com/watson-analytics
- https://developers.google.com/optimization/