Modern Heuristic Search: Towards a Unifying Framework

Martin Müller, University of Alberta
Outline of the Talk

- What is heuristic search?
- Some textbook examples
- What is different in modern heuristic search?
- Examples of recent work
- Towards a general framework
What is Heuristic Search?
Heuristic Search Example

- Heuristic search is a research area in computing science
- It is considered a part of the field of Artificial Intelligence
- It can be used for sequential decision-making problems
- Many applications: automated planning, optimization problems, pathfinding, games, puzzles,…
Which Kind of Search?

- There are many other kinds of search in computing science
  - Internet search, database search, binary search, …
- In heuristic search, we search ahead into the future
  - Which sequences of actions can happen?
  - What is their effect?
- Goal: make decisions about best actions
What is a Heuristic?

- Heuristic is a rough, inexact rule
- A heuristic can *guide* the search
- Roughly, how good is an action?
- Roughly, how good is a state?
- Main question: How to use them to make good decisions?
Why Use a Heuristic?

- Contrast: heuristic vs exact knowledge
- Why not use exact knowledge instead?
  - Often, it is simply not available
    - Example: how good is this Go position?
  - Sometimes, it is available but too expensive to compute
- Problem: how to build a robust system on inexact heuristics
Making Complex Decisions

- We make decisions every moment of our lives
- What is the process that leads to our decisions?
- How to make good decisions?
- Consider many alternatives
- Consider short-term and long-term consequences
- Evaluate different options and choose the best-looking one
Making Sequential Decisions

- Make decision:
  - Get current state of world
  - Analyze it
  - Select an action
  - Observe the world’s response
- If not done: make another decision

Image Source: http://www.prenhall.com
Some Textbook Examples of Heuristic Search
A* Algorithm, Shortest Path

- State space with start state, end state
- Heuristic \( h(s) \) estimates cost-to-go from \( s \) to goal
- \( g(s) \) is cost-so-far from start to \( s \)
- A* always expands a node of smallest sum \( g(s) + h(s) \)
- Greedy, always follows heuristic, no other steps
Minimax, Alphabeta Algorithm

- Standard algorithm for game tree search
- Very successful for chess, checkers, many other games
- Tree search, then call heuristic evaluation function in leaf node
- Problem: always trusts the evaluation function, not robust against errors
- Mostly useless in Go, evaluation quality too bad
Main Problem of Classical Heuristic Search

- Classical methods have two main ingredients
  - Search algorithm
  - Knowledge expressed as heuristic (evaluation) function

Problem: search is greedy/naive

- Always trusts the heuristic
- Not robust against errors in heuristic
- Search can amplify the errors
What is Different in Modern Heuristic Search?
Exploration and Exploitation

- We often deal with information that is:
  - Heuristic, incomplete, stochastic, sparse,…

- Fundamental trade-off:
  - **Exploitation**: make decision based on the information we have
  - **Exploration**: go find more information
Exploration Algorithm for Bandit Problems

- Different actions, unknown “payoff” value
- Can sample each action, at a cost
- Value of action = expected payoff
- Uncertainty about value from lack of samples
Bandit Problems and UCB

- Explore = get more statistics
- Exploit = play best action
- UCB combines both ideas into one balanced formula
- One fundamental algorithm for solving exploration-exploitation problems
The Many Forms of Exploration

✦ UCB is one of the best known algorithms for exploration
✦ Many others
  ✦ Random walk
  ✦ Random simulation
  ✦ Epsilon - greedy
✦ Many more…
Exploration in Modern Heuristic Search

- Doing exploration is the key difference between classic and modern heuristic search
- Many success stories
- Many different approaches to exploration
- I try to understand the common principles
- At this point, we are just doing many case studies
The Three Plus One Pillars of Modern Heuristic Search

❖ Three main ingredients:
  ❖ Search (old)
  ❖ Knowledge (old) plus machine learning
  ❖ Simulations for exploration (new)

❖ All of these are used in AlphaGo
❖ All of these are used in many modern systems
Examples of Recent Work
Game of Go

- Search = Monte Carlo Tree Search
- Knowledge, machine learning = deep convolutional neural networks
- Simulation = play full games until the end
Game of Amazons

- Modern two player game with aspects of both chess (queens) and Go (make territory)
- Search = Monte Carlo Tree Search
- Knowledge = traditional evaluation function
- Simulation = short random move sequences (about 5 moves deep) followed by evaluation
- Interesting case mixing aspects of old and new methods
Automated Planning

- **Search** = Greedy Best-first Search
- **Knowledge** = automatically constructed heuristic, specific for each problem
- **Simulation** = random walks, random sequences of actions
- (Much work done in my group, e.g. Arvand system)
Motion Planning

- Move robot through terrain
- RRT - rapidly exploring random tree (LaValle 1998)
- RRT* - approach optimal paths (Karaman and Frazzoli 2010)
- Extremely popular in robotics
- Early example of random walks

Image: Sertac Karaman

Yellow: start
Purple: goal
Red: obstacles
Green: RRT* tree
Red line: near-optimal path
Towards a General Framework
Many Results, More Questions

- Modern heuristic search has been extremely successful
- Taking proper account of exploration makes algorithms much more robust, and able to handle harder problems
- Advances in search allow to integrate different exploration techniques (simulations, random walks)
- Machine learning gives much stronger domain knowledge (deep neural nets, AlphaGo)
Many More Questions

✦ Each success story is one data point in a larger space

✦ How and why exactly do these programs work?
  ✦ We don’t know

✦ Much development is by trial and error, not by systematic design

✦ Example in Go: change program, then play thousands of test games to check it
Examples of Open Questions

- Given a new problem to solve:
- What is the right exploration method?
- Which machine learning techniques should we use?
- How do we scale to similar but harder problems?
- How do we transfer results to other problems?
Summary

- Modern heuristic search considers exploration
- Search, simulations, machine-learned knowledge
- Many diverse examples of programs which follow this pattern
- Work in progress: Looking for common ground