# Introduction

## Clue is a board game, instance of treasure hunt problem

## Designing an agent requires solving and integrating these solutions

### Tradeoffs

## My Contribution

### Utility Function: Entropy

### Belief: Particle Filtering

### Path Finding: Value Iteration

# Motivation

## Clue

### Rules

### Formal Description

#### Sensor model

#### Action model

#### States

## Treasure Hunt Problem

### Sensors on mobile platforms

### Path planning

#### inference

## Real World Examples, examples of:

### Logic

### Utility

### Path planning

### Belief State Representation

## By nature there is no solution to these problems

## Tradeoffs, performance

# Idea

## Desired Characteristics

### Guaranteed best policy as much as possible

### Probabilistically approaching best policy other wise

### Low memory footprint AND fast computation

#### Hopefully provably so

## Components

### UI/Environment

### Logic

#### Formal Specification

##### Boolean Terms

###### 21 cards

###### numplayers + 1 locations

#### Inherent Knowledge

##### All cards are somewhere

##### If a card is somewhere it isn’t somewhere else

##### At least one card of each type is in the case file

##### If on card of a type is in the case file then no other card of that type is also in the case file

#### Description of how actions update the KB

##### Suggestions

###### Passed people

They don’t have any of the cards

###### Made by you

You learn the location of the revealed card

###### Made by other

You learn the revealing player has at least one of the cards

### Utility Function

#### What it is, why we need it

#### Derived from mathematical principles

##### Why we do this (reference Chenghui Cai)

#### Entropy

##### Description

##### Entropy of What?

###### Mutual Information

##### Requires probabilistic reasoning

###### Calculable after a few turns, otherwise incalculable

###### This system will work in both situations

#### Other possibility: Machine learning

##### Time vs Space

##### Exact vs Estimate

### Particle filtering? (Heart of my contribution)

#### All possible worlds approach

##### Advantages

###### No memory

##### Disadvantages

###### Incomputable

#### Chenghui’s Q-Learning approach

##### Advantages

###### Don’t need to do any tough calculations

###### Faster

##### Disadvantages

###### Needs a long “growing up” time

###### More memory

#### Filtering

##### Advantages

###### Less memory Q-Learning

###### Less time Calculation

##### Disadvantages

###### Updates

###### Generating sample in highly constrained universes!!

#### Mutation (probabilistic)

##### Solves problem of generating

##### Ways to mutate

##### k-beam + genetic

#### Varying # of particles

#### tradeoffs

### Path Finding

#### Value Iteration

#### Value map drawings explaining various states

#### Justification

# Results

## Results

### Time/Space Guarantees

## Tests

### Friends’ impressions of the AI

## Thoughts on difficulty settings

### “Nintendo Hard”

### How to ease up

# Other Work

## Chris Nash

### Interview

## Chenghui Cai & Silvia Ferrari

### Description

# Further Work

## Keeping track of what the other players know about me

## Learning the discount factor for value iteration?

### Why is this a better candidate for machine learning than utility function?

# Appendix

## Code