Introduction

Planning and games

Adam Horky

Lectures

- Trajectorial planning (today Jan Faigl)
- Carmel Domschlack
  - block of lectures 4.-8.3.2013
- Rest at the end of semester
  - 22.4. Graphplan
  - 29.4. Hierarchical planning
  - 6.5. Game theory introduction
  - 13.5. From one to many

Tutorials

- Before Carmel
  - State Space Search revision
  - Trajectorial planning
- After Carmel
  - Classical planning
  - PDDL

Assignemnts

- Two (2 x 15)
- The first will have deadline before Carmel
  - 3.3.2013
  - Trajectorial planning
- The second around 21.4.2013
  - Classical planning - PDDL

State Space Search

Planning and games

Search Space

- Search Space S is a set of states, where the goal is to find the states that satisfy the condition g using actions (operators).
- Formally the problem is defined as a tuple (s0,g, O), where:
  - s0 is the initial state
  - g is the goal condition
  - O is a set of state – transition operators

Uninformed search

- There are various strategies for “uninformed search” (blind search)
  - breadth-first (BFS)
  - depth-first (DFS)
  - iterative deepening (IDS)
  - bidirectional search (BS)
  - Uniform cost search

Uninformed search complexity

- Breadth-First Search (complete, optimal)
  - Time $O(b^d)$
  - Space $O(b^d)$
- Depth-First Search (not complete, not optimal)
  - Time $O(b^d)$, can be infinite
  - Space $O(bd)$
- Iterative deepening (complete, optimal)
  - Time $O(b^d)$
  - Space $O(bd)$

Imformed search

- Based on heuristic function h(x)
- The most common
  - Hill climbing
  - Best-first search
  - A*

State Space Search Framework

```
begin
    OPEN := [Start], CLOSED := []
    while OPEN != [] do
        remove node x with the best f(x) value from OPEN
        if(x == goal)
            return path from Start to x
        else expand node x and for each child x_i
            if(x_i is not in OPEN and CLOSED)
                compute f(x_i)
                add x_i to OPEN
            if(x_i is in OPEN)
                reset value f(x_i), if the new is better
        add x to CLOSED
return err
```

State Space Search Framework

- General algorithm
- Based on the computation of **f(x)** we get different method
- **f(x) = g(x)**, g(x) the cost from start to node x,
- **f(x) = h(x)**, h(x) estimated cost from x to end,
- **f(x) = g(x) + h(x)**, the overall estimated cost

State Space Search Framework

- **f(x) = g(x)** -> Uniform Cost Search (uninformed),
- **f(x) = h(x)**, Greedy Best First Search,
- **f(x) = g(x) + h(x)**, A*

Example

Homework

- Implement A*
- Try to do it as a general framework
  - use general concepts – state, action, node
  - it can be reused then for any other planning problem