CMU 15-381
Lecture 5: Planning I

Teachers:
Emma Brunskill
Ariel Procaccia (this time)
Motion Planning

• Navigating between two points while avoiding obstacles
• A first approach: define a discrete grid
• Mark cells that intersect obstacles as blocked
• Find path through centers of remaining cells
Is this approach optimal? Complete?
Cell decomposition

• Distinguish between
  • Cells that are contained in obstacles
  • Cells that intersect obstacles

• If no path found, subdivide the mixed cells
Is it complete now?

• An algorithm is resolution complete when:
  a. If a path exists, it finds it in finite time
  b. If a path does not exist, it returns in finite time

• Poll 1: Cell decomposition satisfies:
  1. a but not b
  2. b but not a
  3. Both a and b
  4. Neither a nor b
**Cell Decomposition**

Shortest paths through cell centers

Shortest path
**Solution 1: A* Smoothing**

- Allows connection to further states than neighbors on the grid
- Key observation:
  - If \( x_1, \ldots, x_n \) is valid path
  - And \( x_k \) is visible from \( x_j \)
  - Then \( x_1, \ldots, x_j, x_k, \ldots, x_n \) is a valid path
SMOOTHING WORKS!

--- A shortest path through cell centers

----- Shortest path
SMOOTHING DOESN’T WORK!

--- A shortest path through cell centers
-----. Shortest path
 SOLUTION 2: THETA*

• Allow parents that are non-neighbors in the grid to be used during search

• Standard A*
  ○ $g(y) = g(x) + c(x, y)$
  ○ Insert $y$ with estimate
    $$f(y) = g(x) + c(x, y) + h(y)$$

• Theta*
  ○ If parent$(x)$ is visible from $y$, insert $y$ with estimate
    $$f(y) = g($parent$(x)) + c($parent$(x), y) + h(y)$$
**Theta** ∗ **works!**

---

Theta* path, I think 😊

-----

Shortest path
THETA* WORKS!

[Nash, AIGameDev 2010]
THE OPTIMAL PATH

• **Polygonal path**: sequence of connected straight lines
• **Inner vertex of polygonal path**: vertex that is not beginning or end
• **Theorem**: assuming polygonal obstacles, shortest path is a polygonal path whose inner vertices are vertices of obstacles
Proof of Theorem

• Suppose for contradiction that shortest path is not polygonal

• Obstacles are polygonal \( \Rightarrow \)
  \( \exists \) point \( p \) in interior of free space such that “path through \( p \) is curved”

• \( \exists \) disc of free space around \( p \)

• Path through disc can be shortened by connecting points of entry and exit
Proof of Theorem

• Path is polygonal!
• Vertex cannot lie in interior of free space, otherwise we can do the same trick
• Vertex cannot lie on a the interior of an edge, otherwise we can do the same trick ■
How would we define a graph on which A* would be optimal?
Planning, More Generally

- AI (also) studies rational action
- Devising a plan of action to achieve one’s goal is a critical part of AI
- In fact, planning is glorified search
- We will consider a structured representation of states
Propositional STRIPS planning

- **STRIPS** = Stanford Research Institute Problem Solver (1971)
- State is a conjunction of **conditions**, e.g., \(\text{at}(\text{Truck}_1, \text{Shadyside}) \land \text{at}(\text{Truck}_2, \text{Oakland})\)
- States are transformed via **operators** that have the form
  Preconditions \(\Rightarrow\) Postconditions
PROPOSITIONAL STRIPS PLANNING

- **Pre** is a conjunction of positive and negative conditions that must be satisfied to apply the operation
- **Post** is a conjunction of positive and negative conditions that become true when the operation is applied
- We are given the initial state
- We are also given the **goals**, a conjunction of positive and negative conditions
- We think of a state as a set of positive conditions, hence an operation has an “add list” and a “delete list”
Blocks world

Start

Goal
Blocks world

- Conditions: on(A,B), on(A,C), on(B,A), on(B,C), on(C,A), on(C,B), clear(A), clear(B), clear(C), on(A,Table), on(B,Table), on(C,Table)

- Operators for moving blocks
  - Move C from A to the table:
    clear(C) \land on(C,A)
    \Rightarrow on(C,Table) \land clear(A) \land \neg on(C,A)
  - Move A from the table to B
    clear(A) \land on(A,Table) \land clear(B)
    \Rightarrow on(A,B) \land \neg clear(B) \text{ and } \neg on(A,Table)
THE PLAN

• State: on(C,A), on(A,Table), on(B,Table), clear(B), clear(C)

• Action:
  clear(C) \land on(C,A)
  \implies on(C,Table) \land clear(A) \land \neg on(C,A)
THE PLAN

- **State:** on(A,Table), on(B,Table), clear(B), clear(C), on(C,Table), clear(A)

- **Action:**
  \[ \text{clear(C) } \land \text{ on(B,Table) } \land \text{ clear(B)} \]
  \[ \Rightarrow \text{ on(B,C) } \land \neg \text{clear(C)} \text{ and } \neg \text{on(B,Table)} \]
THE PLAN

• State: on(A,Table), clear(B), on(C,Table), clear(A), on(B,C)

• Action:
  clear(B) \land on(A,Table) \land clear(A)
  \Rightarrow on(A,B) \land \neg clear(B) \text{ and } \neg on(A,Table)
THE PLAN

- State: on(C, Table), clear(A), on(B, C), on(A, B)
- Goals: on(A, B), on(B, C)
COMPLEXITY OF PLANNING

• **PLANSAT** is the problem of determining whether a given planning problem is satisfiable
• In general **PLANSAT** is **PSPACE**-complete
• We will look at some special cases
COMPLEXITY OF PLANNING

• **Theorem 1:** Assume that actions have only positive preconditions and a single postcondition. Then PLANSAT is in \( \mathbf{P} \)

• **Theorem 2:** Blocks world problems can be encoded as above

• **Silly corollary:** Blocks world problems can be solved in polynomial time (Duh)
**Proof of Theorem 2**

- We will convert blocks world operators to operators that have only positive preconditions and a single postcondition.
- Let the blocks be $B_1, \ldots, B_n$.
- Conditions: $\text{off}(i, j)$ means $B_i$ is not on top of $B_j$.

\[ \bigwedge_{k} \text{off}(k, i) \land \bigwedge_{k \neq i} \text{off}(k, j) \Rightarrow \text{off}(i, j) \]
\[ \bigwedge_{k} \text{off}(k, i) \land \bigwedge_{k} \text{off}(i, k) \land \bigwedge_{k} \text{off}(k, j) \Rightarrow \neg \text{off}(i, j) \]
**Proof of Theorem 1**

- **Lemma:** It is sufficient to consider plans that first make conditions true, then make conditions false

- **Proof:**
  - Suppose that $o_i$ and $o_{i+1}$ are adjacent operators s.t. the postcondition $p$ of $o_i$ is negative and the postcondition $q$ of $o_{i+1}$ is positive
  - If $p = q$ then we can delete $o_i$ because its effect is reversed
  - Otherwise, we can switch $o_i$ and $o_{i+1}$ ■

* Just for fun
Proof of Theorem 1*

• By the lemma, if there is a solution, there is an intermediate state $S$ such that
  o $S$ can be reached from the initial state using operations with positive postconditions
  o The positive goals are a subset of $S$
  o Negative goals can be achieved via operations with negative postconditions

• Search for an intermediate state $S$ with these properties

* Just for fun
Proof of Theorem 1*

• Implement procedure TurnOn(X): given set of conditions X, find maximal state S such that $S \cap X = \emptyset$ that can be reached from initial state using operators with positive postconditions
  
  o Preconditions are positive, so:
  
  o Simply apply all such operators until it makes no difference

* Just for fun
Proof of Theorem 1

- Denote $S''$ the state resulting from removing negative goals from $S$
- Implement procedure $\text{TurnOff}(S)$: find the maximal $S'$ such that $S''$ is reachable from $S'$ using operators with negative postconditions in $S$
  - Simply search backwards from $S''$ and reverse operators with
    - (i) negative postconditions in $S$
    - (ii) preconditions satisfied

* Just for fun
Proof of Theorem 1*

• In the first iteration, if positive goals are not satisfied by $S$, there is no way to achieve them.

• If $S \setminus S' \neq \emptyset$, it is impossible to remove these conditions; must be added to $X$.

• $X$ grows monotonically $\Rightarrow$ polynomial time $\blacksquare$

$X = \emptyset$

loop

$S = \text{TurnOn}(X)$

if $S$ does not contain positive goals then return reject

$S' = \text{TurnOff}(S)$

if $S = S'$ then return accept

$X = X \cup (S \setminus S')$

if $X$ intersects with initial state then return reject

* Just for fun
SUMMARY

• Terminology:
  o Cell decomposition
  o Resolution completeness
  o Theta*
  o STRIPS

• Big ideas:
  o A* can be modified to work well in continuous spaces