Planning, Execution & Learning: Introduction

Reid Simmons, reids@cs.cmu.edu

Manuela Veloso, mmv@cs.cmu.edu

September 8, 2010

Logistics

- Lecture-based
  - No textbook, assigned readings
- Question-driven, summary points on website
- Four homeworks (three plus one project)
  - Programming, questions, readings
  - hmw1 – 15%, hmw2 – 15%, hmw3 – 15%, hmw4 – 25%
  - Due in class
- One 24hour take-home final exam (30%)
  - Available Dec 4 – Dec 11
- http://www.cs.cmu.edu/~mmv/planning

Planning, Execution, and Learning?

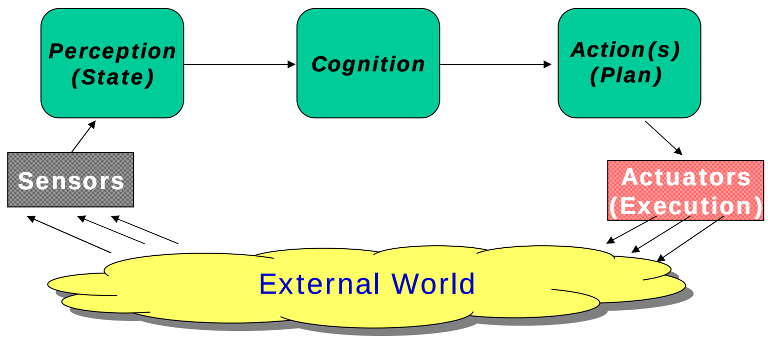
- What is “planning?”
- What is “execution?”
- What is “learning?”

Planning, Execution, and Learning

- Planning
  - ***Generate*** a set of ***actions*** **– a plan –** that can transform an ***initial state*** of the world to a ***goal state***

    [Newell and Simon, 1950s]
- Execution
  - Start at the initial state, and ***perform*** each action of a generated plan
- Learning
  - ***Improve*** plan generation and plan execution from ***experience***

Autonomous Robot: Example of Planning, Execution, and Learning



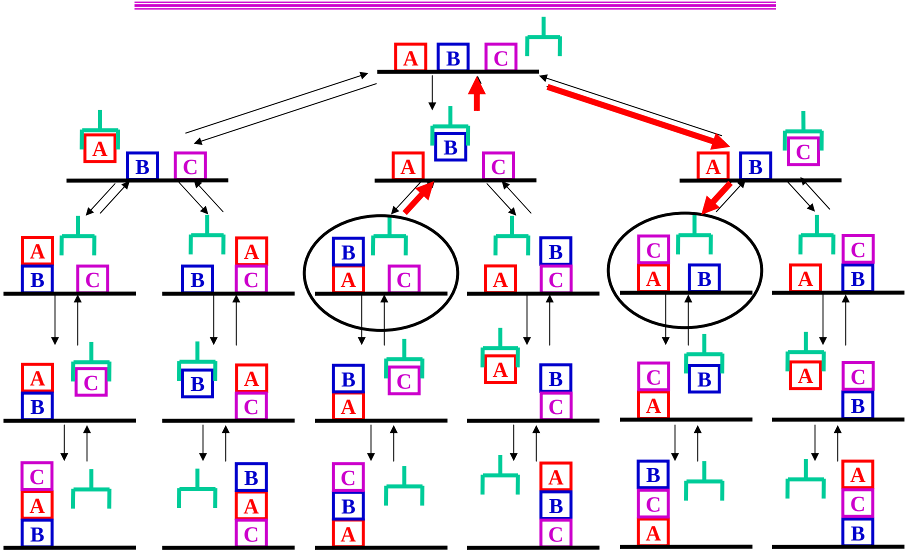
Planning - Generation

- **Generate** a set of ***actions*** – a plan – that can transform an ***initial state*** of the world to a ***goal state***
- Plan generation:
  – Given the possible actions, the states of the world, search for paths between states

Planning = "Thinking"

- Get the state of the world
- Know the goal state
- THINK (not execute) about what will happen if… an action is chosen… and another action is chosen… and another is chosen….
- THINK about STATES, about ACTIONS and their effects.

3-BlocksWorld State Transitions



Planning - States

- **"Information-less" numerical identification (s1, s2,...)**
- **"Symbolic description"**
  - **Features**
  - **Predicates**
- **Conjunctive, enumerative, observable**
- **Complete, correct, deterministic**
- **Probabilistic, approximate, incremental, on-demand**

The Blocks World - States

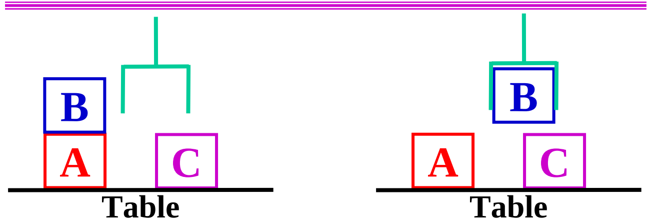
- Objects
  - Blocks: *A, B, C*
  - Table: *Table*
- Predicates
  - *On(A, B), On(C, Table), Clear(B), Handempty, Holding(C)*
  - *On-table(A), On(A,B), Top(B),…*
  - *Tower(A,B,C,…)*
- States – Conjunctive
  - On(A,B) and On(B,C) and Clear(A) and Handempty

Planning - Actions

– A description of ***legal*** actions in the domain
  - Blocks world: “unstack if top is clear”….
  - Chess, “open door if unlocked”,

– ***Model*** of an action

The Blocks World Definition - Actions



- Blocks are picked up and put down by the arm
- Blocks can be picked up only if they are clear, i.e., without any block on top
- The arm can pick up a block only if the arm is empty, i.e., if it is not holding another block, i.e., the arm can be pick up only one block at a time
- The arm can put down blocks on blocks or on the table

Planning - Actions

- STRIPS model of an action:
  - **Preconditions: applicability conditions**
  - **Effects – changes in state**
    - **Adds and deletes**

STRIPS Representation - BlocksWorld

Pickup_from_table(b)
Pre: Block(b), Handempty
Clear(b), On(b, Table)
Add: Holding(b)
Delete: Handempty,
On(b, Table)

Pickup_from_block(b, c)
Pre: Block(b), Handempty
Clear(b), On(b, c), Block(c)
Add: Holding(b), Clear(c)
Delete: Handempty,
On(b, c)

Putdown_on_table(b)
Pre: Block(b), Holding(b)
Add: Handempty,
On(b, Table)
Delete: Holding(b)

Putdown_on_block(b, c)
Pre: Block(b), Holding(b)
Block(c), Clear(c), $b \neq c$
Add: Handempty, On(b, c)
Delete: Holding(b), Clear(c)

Planning - Generation

- Given all the states of the world, ***search*** for paths (shortest?) between states
- **Classical planning algorithms**
  - **Forward from state, backward from goals**
  - **Plan-space search**
  - **State-space search**
  - **Serial, parallel search**
  - **Logical satisfiability**
  - **Heuristic search**
  - **………..**

Planning AND Execution

- Planning – in principle *to execute* the plan……
  - While planning, no execution
  - Planning as *thinking* (stopped)
    - How long/efficiently?
    - How often?
    - How precise?
  - Planning needs world *modeling*
    - How accurately?
    - How detailed?
    - How precise?

Planning AND Execution

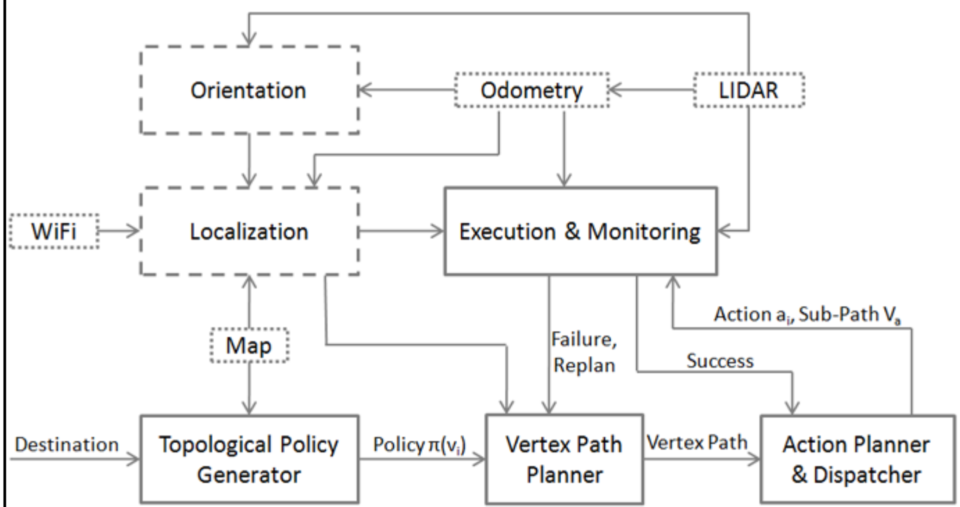
- Execution of a plan
  - Achieve goals in real world
  - Assess the state of the world
  - Check applicability conditions
  - Detect “end” of action from effects
  - Detect effects at end of action

Layered Architecture

Xavier Architecture (1995)

Task Planning (Prodigy)
Path Planning (Decision-Theoretic)
Map-Based Navigation (POMDPs)
Local Obstacle Avoidance (Curvature Velocity Method)
Servo-Control (Commercial)

CoBot Execution Architecture (2009)



Simple Execution Algorithm

- For all steps in the plan
  - Assess state
  - Check if preconds of step are true in state
  - If yes, execute step
  - If no,?………

  (when to execute next step,?....)

If No…..

- Applicability conditions of step are not true in current state
- Maybe this was predicted….
  - Conditional effects
  - Conditional plans
  - Probabilistic models
  - Universal planning
- Maybe it was not predicted….
  - Replan
    - Complicated issues – discuss
- Maybe we don't *plan* – reactive planning
  - one action at a time…
  - State, goals

When Next Step?....

- Plan execution monitoring
  - Is the robot *in front of the door?*
  - Is the robot *making progress towards the desired effects?*
  - *Etc…*

Conclusion: Planning, Execution, and Learning

- Planning
- Execution
- Learning (everywhere)
  - Models, efficiency, quality
  - Action representation
    - (not granularity, not alternative representations,…)
  - Policy/plan
    - Control function from all states
  - Plan quality
  - Multi-robot
    - Joint state beliefs, joint policies
    - what, when, how to communicate