CS325 Artificial Intelligence
Ch. 11, Advanced Planning

Cengiz Günay, Emory Univ.

Spring 2013

Advanced Planning Concepts

This lecture:

Time: Scheduling

This lecture:

Time: Scheduling

Resources: Consumables

Advanced Planning Concepts

This lecture:

Time:	Scheduling
Resources:	Consumables
Active perception:	Look and feel?

Advanced Planning Concepts

This lecture:

Time:	Scheduling
Resources:	Consumables
Active perception:	Look and feel?
Hierarchical plans:	Abstracting

Entry/Exit Surveys

Exit survey: Game Theory

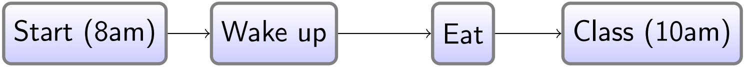
- Why don't we take the mixed strategy if there is a dominant strategy?
- What advantage is gained by a player by *looking* irrational?

Entry survey: Advanced Planning (0.25 points of final grade)

- Do you think a classical planning planning approach can be used for solving scheduling problems?
- What would be the advantage of making hierarchical plans?

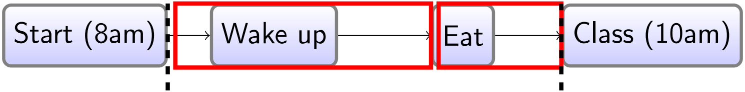
I'm Late Again!

- Fixed times for day start and class
- Durations
  - Wake up:10 minutes
  - Eat: 30 minutes



I'm Late Again!

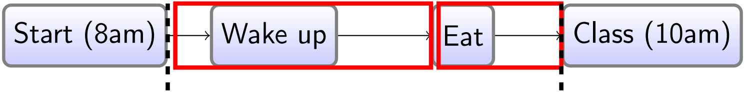
- Fixed times for day start and class
- Durations
  - Wake up:10 minutes
  - Eat: 30 minutes



- Earliest and latest start times of each event?

I'm Late Again!

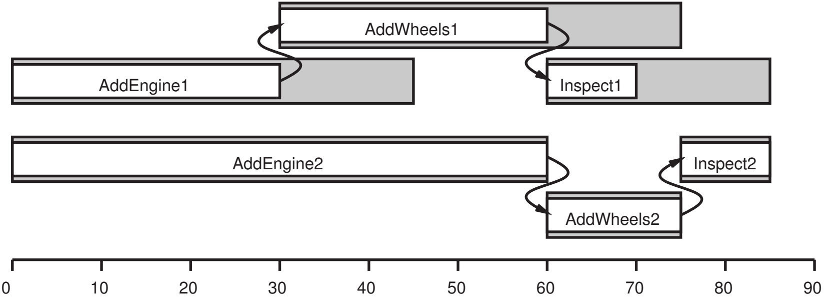
- Fixed times for day start and class
- Durations
  - Wake up:10 minutes
  - Eat: 30 minutes



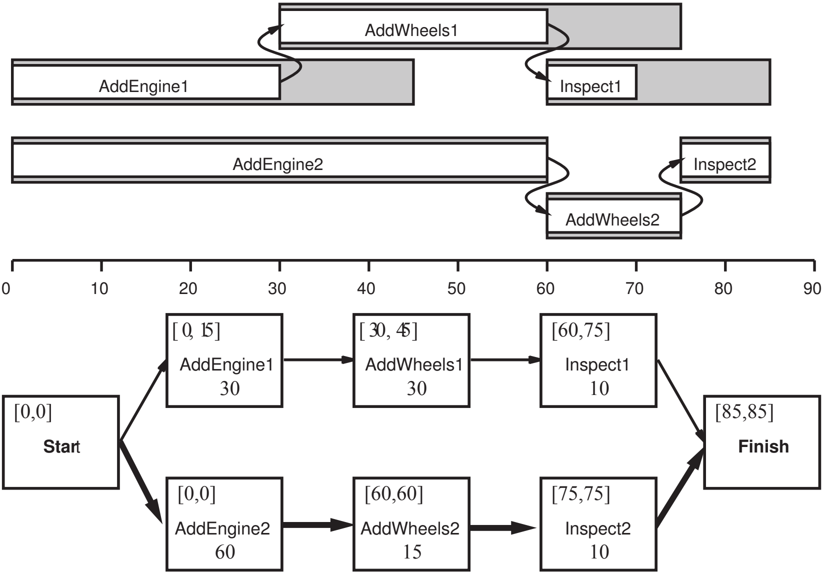
- Earliest and latest start times of each event?
  - Earliest(Wake up)=8am
  - Latest(Wake up)=?
  - Earliest(Eat)=?
  - Latest(Eat)=$10{:}00 - 00{:}30 = 9{:}30$am

Multiple Paths to Finish: Car with 2 Engines

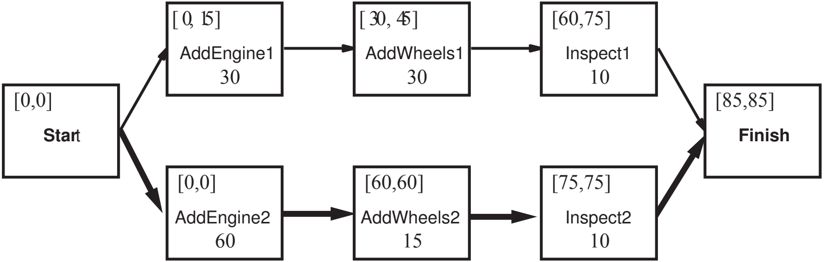
- Critical path?



Multiple Paths to Finish: Car with 2 Engines



Multiple Paths to Finish: Car with 2 Engines



Earliest(Start)=0
Earliest(B)=$\max_{A \to B}$ Earliest(A) + Duration(A)
Latest(A)=$\max_{A \leftarrow B}$ Latest(B) − Duration(A)
Latest(Finish)=Earliest(Finish)

Resources: Can We Use Action Schemas?

$Action(Inspect(n_1, n_2, n_3, n_4, n_5, b_1, b_2, b_3, b_4, b_5,),$

$\quad$ PRE: $Fastened(n_1,\ b_1),\ \ldots,\ Fastened(n_5,\ b_5)),$

$\quad$ EFF: $Inspected$)

$Action(Fasten(n,\ b),$

$\quad$ PRE: $Nut(n) \land Bolt(b)$

$\quad$ EFF: $Fastened(n,\ b) \land \neg Nut(n) \land \neg Bolt(b))$

$Init(Nut(N1),\ \ldots,\ Nut(N4),$

$\quad\quad Bolt(B1),\ \ldots Bolt(B5))$

$Goal(Inspected)$

Resources: Can We Use Action Schemas?

$Action(Inspect(n_1, n_2, n_3, n_4, n_5, b_1, b_2, b_3, b_4, b_5,),$

$\quad$ PRE: $Fastened(n_1,\ b_1), \ldots, Fastened(n_5,\ b_5)),$

$\quad$ EFF: $Inspected$)

$Action(Fasten(n,\ b),$

$\quad$ PRE: $Nut(n) \land Bolt(b)$

$\quad$ EFF: $Fastened(n,\ b) \land \neg Nut(n) \land \neg Bolt(b))$

$Init(Nut(N1), \ldots, Nut(N4),$

$\qquad Bolt(B1), \ldots Bolt(B5))$

$Goal(Inspected)$

- Will it reach the goal?

Resources: Can We Use Action Schemas?

$Action(Inspect(n_1, n_2, n_3, n_4, n_5, b_1, b_2, b_3, b_4, b_5,),$

$\quad \text{PRE: } Fastened(n_1,\ b_1), \ldots, Fastened(n_5,\ b_5)),$

$\quad \text{EFF: } Inspected\ )$

$Action(Fasten(n,\ b),$

$\quad \text{PRE: } Nut(n) \wedge Bolt(b)$

$\quad \text{EFF: } Fastened(n,\ b) \wedge \neg Nut(n) \wedge \neg Bolt(b))$

$Init(Nut(N1), \ldots, Nut(N4),$

$\qquad Bolt(B1), \ldots Bolt(B5))$

$Goal(Inspected)$

- Will it reach the goal? **No**, one nut is missing.

$Action(Inspect(n_1, n_2, n_3, n_4, n_5, b_1, b_2, b_3, b_4, b_5,),$
PRE: $Fastened(n_1,\ b_1),\ \ldots,\ Fastened(n_5,\ b_5)),$
EFF: $Inspected\ )$

$Action(Fasten(n,\ b),$
PRE: $Nut(n) \wedge Bolt(b)$
EFF: $Fastened(n,\ b) \wedge \neg Nut(n) \wedge \neg Bolt(b))$

$Init(Nut(N1),\ \ldots,\ Nut(N4),$
$Bolt(B1),\ \ldots Bolt(B5))$

$Goal(Inspected)$

- Will it reach the goal? **No**, one nut is missing.
- Depth first tree search: how many paths to eval?

Small: 1, 4, 5 ?
Medium: $4 + 5$ or 4×5 ?
Large: $4!$, $5!$, or $4! \times 5!$?

Resources: Can We Use Action Schemas?

$Action(Inspect(n_1, n_2, n_3, n_4, n_5, b_1, b_2, b_3, b_4, b_5,),$
PRE: $Fastened(n_1,\ b_1), \ldots, Fastened(n_5,\ b_5)),$
EFF: $Inspected\ )$
$Action(Fasten(n,\ b),$
PRE: $Nut(n) \wedge Bolt(b)$
EFF: $Fastened(n,\ b) \wedge \neg Nut(n) \wedge \neg Bolt(b))$
$Init(Nut(N1), \ldots, Nut(N4),$
$Bolt(B1), \ldots Bolt(B5))$
$Goal(Inspected)$

- Will it reach the goal? **No**, one nut is missing.
- Depth first tree search: how many paths to eval?

Small: 1, 4, 5 ?
Medium: $4 + 5$ or 4×5 ?
Large: 4!, 5!, or $4! \times 5!$. **Really inefficient!**

Modify Action Schemas to Optimize Resource Problems

No need to try combinations of same resources.

Define:

Resources: Specify quantity.

Use: Specify requirement.

Consume: Removes resource.

Modify Action Schemas to Optimize Resource Problems

No need to try combinations of same resources.

Define:

- Resources: Specify quantity.
- Use: Specify requirement.
- Consume: Removes resource.

$Action(Inspect(n_1, n_2, n_3, n_4, n_5, b_1, b_2, b_3, b_4, b_5,),$
PRE: $Fastened(n_1,\ b_1), \ldots, Fastened(n_5,\ b_5)),$
USE: $Inspector(1)$
EFF: $Inspected\ )$
$Action(Fasten(n,\ b),$
CONSUME: $Nuts(1), Bolts(1)$
EFF: $Fastened(n,\ b))$
$Resources(Nuts(5), Bolts(4), Inspectors(1))$

Modify Action Schemas to Optimize Resource Problems

No need to try combinations of same resources.

Define:

- Resources: Specify quantity.
- Use: Specify requirement.
- Consume: Removes resource.

Action(*Inspect*($n_1, n_2, n_3, n_4, n_5, b_1, b_2, b_3, b_4, b_5$),
  PRE: *Fastened*(n_1, b_1), ..., *Fastened*(n_5, b_5)),
  USE: *Inspector*(1)
  EFF: *Inspected*)
Action(*Fasten*(n, b),
  CONSUME: *Nuts*(1), *Bolts*(1)
  EFF: *Fastened*(n, b))
Resources(*Nuts*(5), *Bolts*(4), *Inspectors*(1))

No exponential explotion anymore!

Hierarchical Planning: Abstraction

Remember Stanley:

- High-level goal:
  - Reach target at GPS coordinates
  - Drive on road
- Low-level actions:
  - Adjust steering wheel
  - Press/release gas/break pedals

Hierarchical Planning: Abstraction

Remember Stanley:

- High-level goal:
  - Reach target at GPS coordinates
  - Drive on road
- Low-level actions:
  - Adjust steering wheel
  - Press/release gas/break pedals

How to connect

1. high-level (abstract) planning with
2. low-level planning?

Hierarchical Planning: Abstraction

Remember Stanley:

- High-level goal:
  - Reach target at GPS coordinates
  - Drive on road
- Low-level actions:
  - Adjust steering wheel
  - Press/release gas/break pedals

How to connect

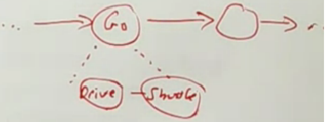
1. high-level (abstract) planning with
2. low-level planning?

Solution: **Refinement**

Refining Abstractions

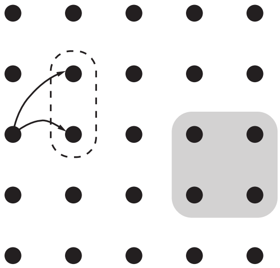
Multiple ways to refine abstractions:

$Refinement(Go(Home, SFO),$
  STEPS: $[Drive(Home, SFOLongTermParking),$
  $Shuttle(SFOLongTermParking, SFO)])$
$Refinement(Go(Home, SFO),$
  STEPS: $[Taxi(Home, SFO)])$

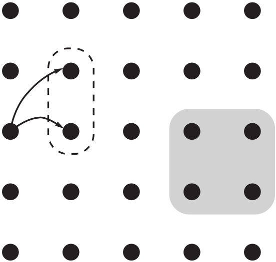


$Refinement(Navigate([a, b], [x, y])$
  PRECOND: $a = x \land b = y$
  STEPS: $[\,])$
$Refinement(Navigate([a, b], [x, y])$
  PRECOND: $Connected([a, b], [a - 1, b])$
  STEPS: $[Left, Navigate([a - 1, b], [x, y])])$
$Refinement(Navigate([a, b], [x, y]),$
  PRECOND: $Connected([a, b], [a + 1, b])$
  STEPS: $[Right, Navigate([a + 1, b], [x, y])])$

Reachable?

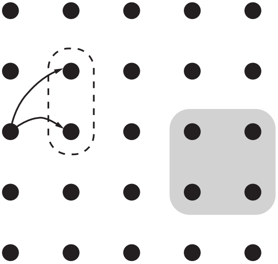


Reachable?



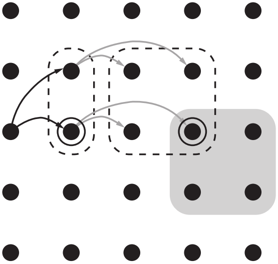
No.

Reachable?



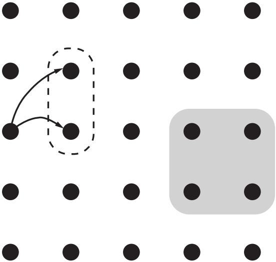
No.

Found solution:



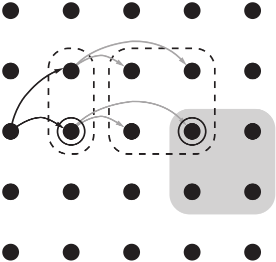
Hierarchical Planning: Reachable States

Reachable?



No.

Found solution:



Now backtrack from solution.

Reachable States Question

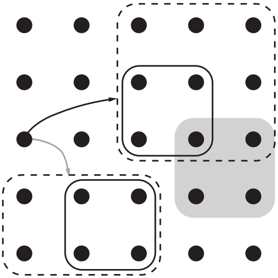
Estimates of refinement:

Underestimate: We reach it for sure.

Overestimate: Possibly reachable.

Below examples:

- Reachable? Yes, No, Maybe?



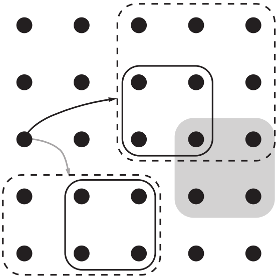
Estimates of refinement:

Underestimate: We reach it for sure.

Overestimate: Possibly reachable.

Below examples:

- Reachable? Yes, No, Maybe?



No

Yes.

Reachable States Question

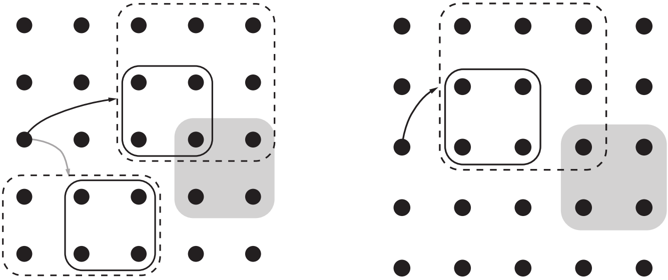
Estimates of refinement:

Underestimate: We reach it for sure.

Overestimate: Possibly reachable.

Below examples:

- Reachable? Yes, No, Maybe?



No Yes.

Reachable States Question

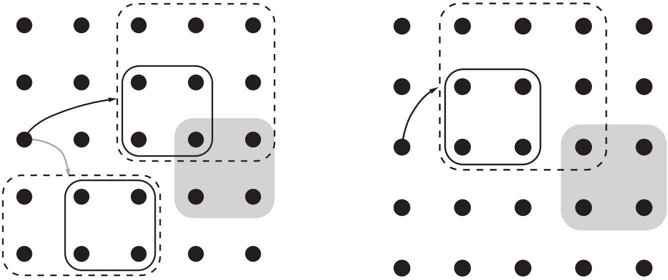
Estimates of refinement:

Underestimate: We reach it for sure.

Overestimate: Possibly reachable.

Below examples:

- Reachable? Yes, No, Maybe?



No **Yes.** **Maybe.**

Extending Planning with Observations

Sometimes the agent needs to look first.

Extending Planning with Observations

Sometimes the agent needs to look first.

$Init(Object(Table) \land Object(Chair) \land Can(C_1) \land Can(C_2) \land InView(Table))$

$Goal(Color(Chair, c) \land Color(Table, c))$

$Action(RemoveLid(can),$
  PRECOND: $Can(can)$
  EFFECT: $Open(can))$

$Action(Paint(x, can),$
  PRECOND: $Object(x) \land Can(can) \land Color(can, c) \land Open(can)$
  EFFECT: $Color(x, c))$

$Percept(Color(x, c),$
  PRECOND: $Object(x) \land InView(x)$

$Percept(Color(can, c),$
  PRECOND: $Can(can) \land InView(can) \land Open(can)$

$Action(LookAt(x),$
  PRECOND: $InView(y) \land (x \neq y)$
  EFFECT: $InView(x) \land \neg InView(y))$