# EECS 496: Sequential Decision Making

Soumya Ray

sray@case.edu

Office: Olin 516

Office hours : T 4-5:30 or by appointment

# Recap

- What is the complete exclusion axiom in SATplan?
- How do we represent frame axioms?
- Why do we do it this way?
- How do we extract a plan?
- How do we support layered plans?
- How can we integrate Graphplan and SATplan?
- What does SATplan show?

# Today

- Nonclassical Planning: Actions with Durations and Resources

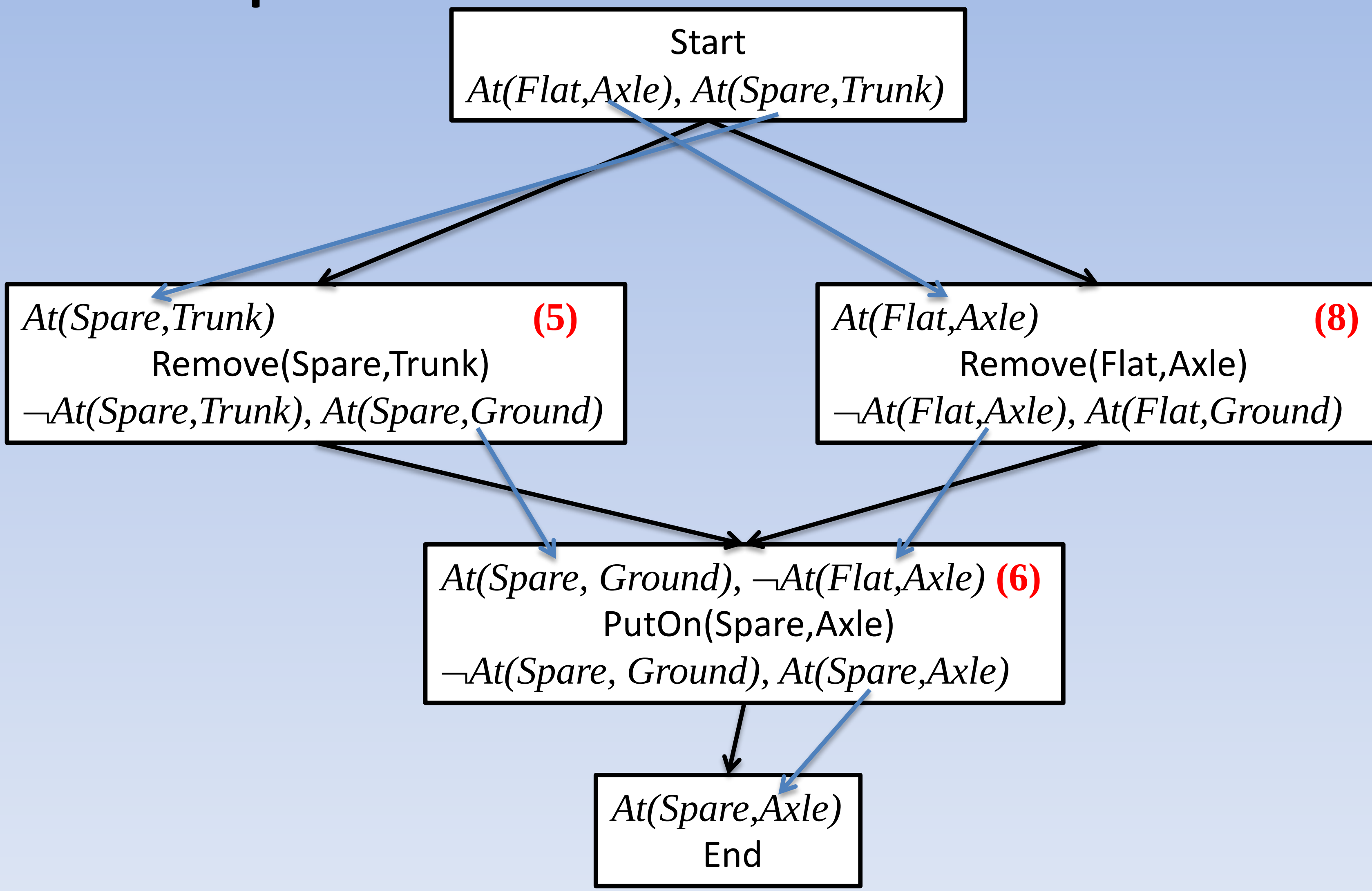
# Nonclassical Planning

- Classical Planning
  - Deterministic
  - Static
  - Fully observable
  - Propositional
  - Actions are instantaneous

# Durative Actions

- Suppose each action has some duration associated with it, and actions can overlap (in time)
  - Represented in Planning Domain Definition Language
- This brings up the issue of *scheduling* actions, in a partial order plan
- Scheduling actions means assigning to each action a time at which the action should start

# Example


Start
At(Flat,Axle), At(Spare,Trunk)
At(Spare,Trunk)
(5)
Remove(Spare,Trunk)
¬At(Spare,Trunk), At(Spare,Ground)
At(Flat,Axle)
(8)
Remove(Flat,Axle)
¬At(Flat,Axle), At(Flat,Ground)
At(Spare, Ground), ¬At(Flat,Axle)
(6)
PutOn(Spare,Axle)
¬At(Spare, Ground), At(Spare,Axle)
At(Spare,Axle)
End

# Scheduling after Planning

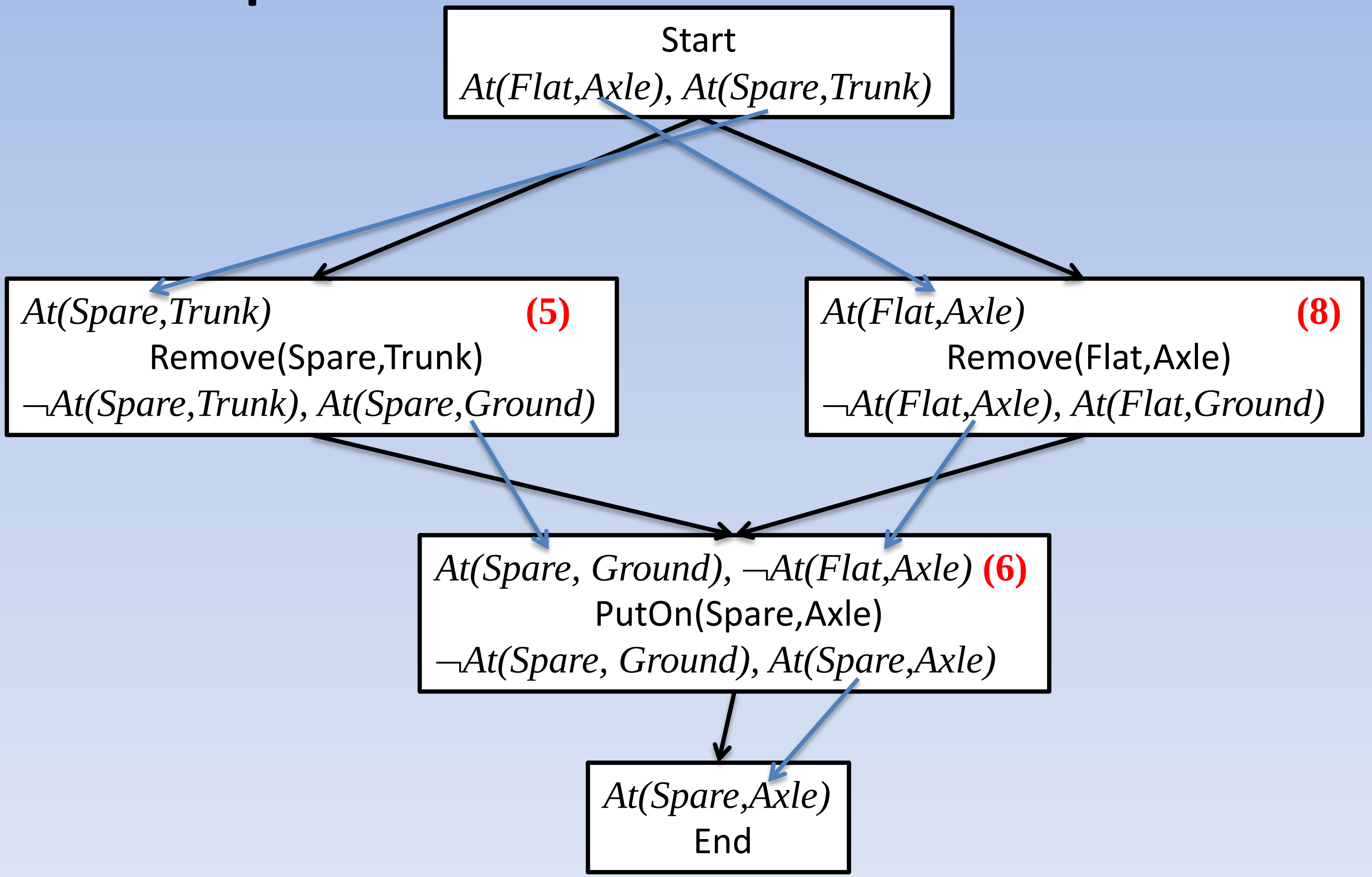
- Suppose we are given a fixed partial order plan such that it meets the goals

- We want to schedule actions to obtain the plan with the lowest total duration
  - Called the *makespan*

# Critical Path

- Consider every path through the POP from Start to End
  - Each path has a duration
- The path with the longest duration is called the Critical Path
  - Shortening other paths does not affect the plan duration
  - Delaying any action on the CP delays the whole plan

# Example



Start
At(Flat,Axle), At(Spare,Trunk)
At(Spare,Trunk)
(5)
Remove(Spare,Trunk)
¬At(Spare,Trunk), At(Spare,Ground)
At(Flat,Axle)
(8)
Remove(Flat,Axle)
¬At(Flat,Axle), At(Flat,Ground)
At(Spare, Ground), ¬At(Flat,Axle)
(6)
PutOn(Spare,Axle)
¬At(Spare, Ground), At(Spare,Axle)
At(Spare,Axle)
End

# Critical Path Method

- Key idea: what is the range of times within which we can start an action in a plan so it can successfully complete?

- For each action, we compute an "earliest start" and a "latest start" time using two dynamic programs

# Forward Pass

- Incrementally compute the earliest time an action can start, from the START dummy action
- Traverse the POP in topological sort order

$$ES(Start) = 0$$

$$ES(B) = \max_{\substack{A \text{ is a predecessor} \\ \text{of } B \text{ in the POP}}} ES(A) + Duration(A)$$

# Backward Pass

- Incrementally compute the latest time an action can start, from the END dummy action

- Traverse the POP in reverse topological sort order

$$LS(End) = ES(End)$$

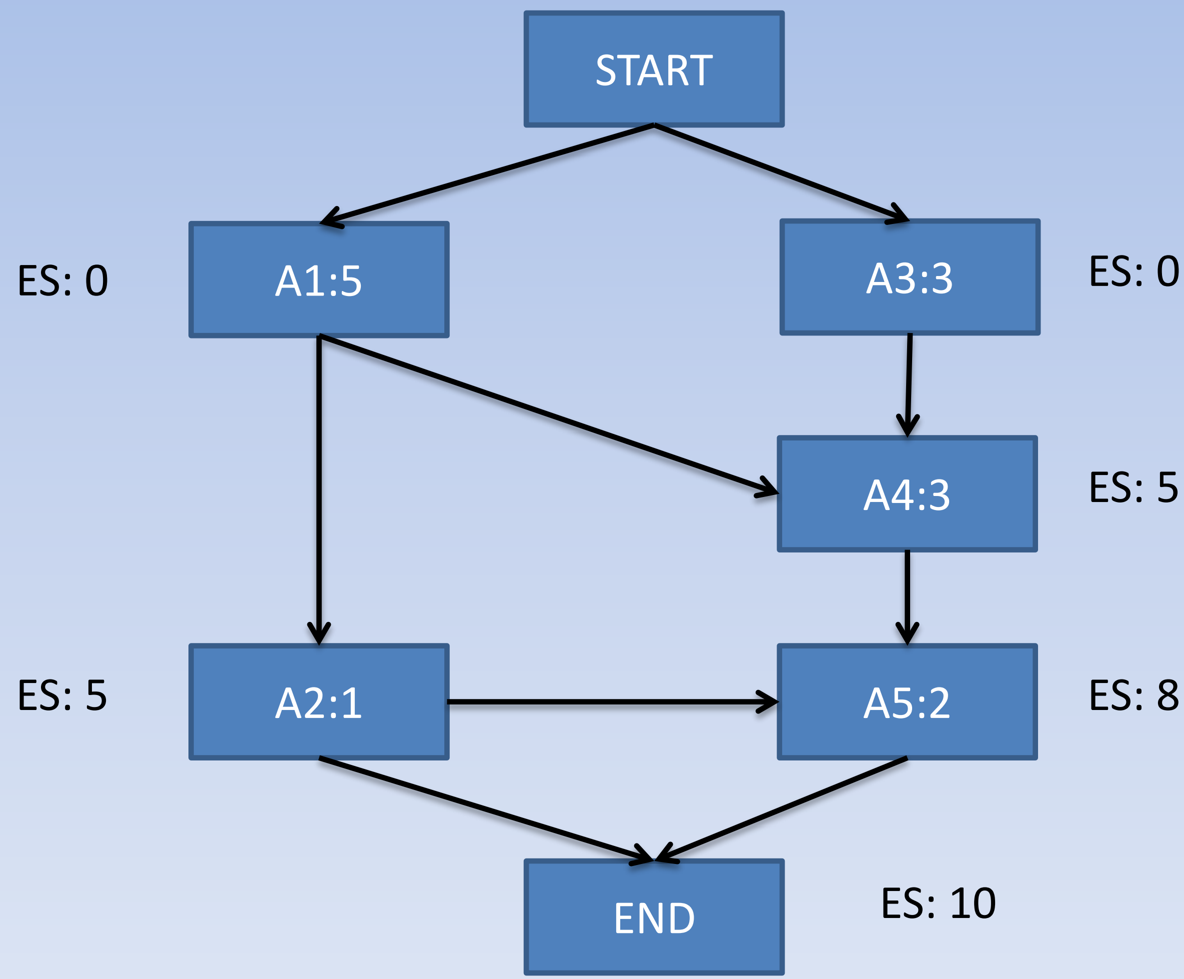
$$LS(B) = \min_{\substack{A \text{ is a successor} \\ \text{of } B \text{ in the POP}}} LS(A) - Duration(B)$$

# Critical Path Method

- For actions on the critical path, ES=LS
- For other actions, each can be started in the interval $(\mathrm{ES}, \mathrm{LS})$ in order to get the plan with the minimal makespan (the length of the critical path)

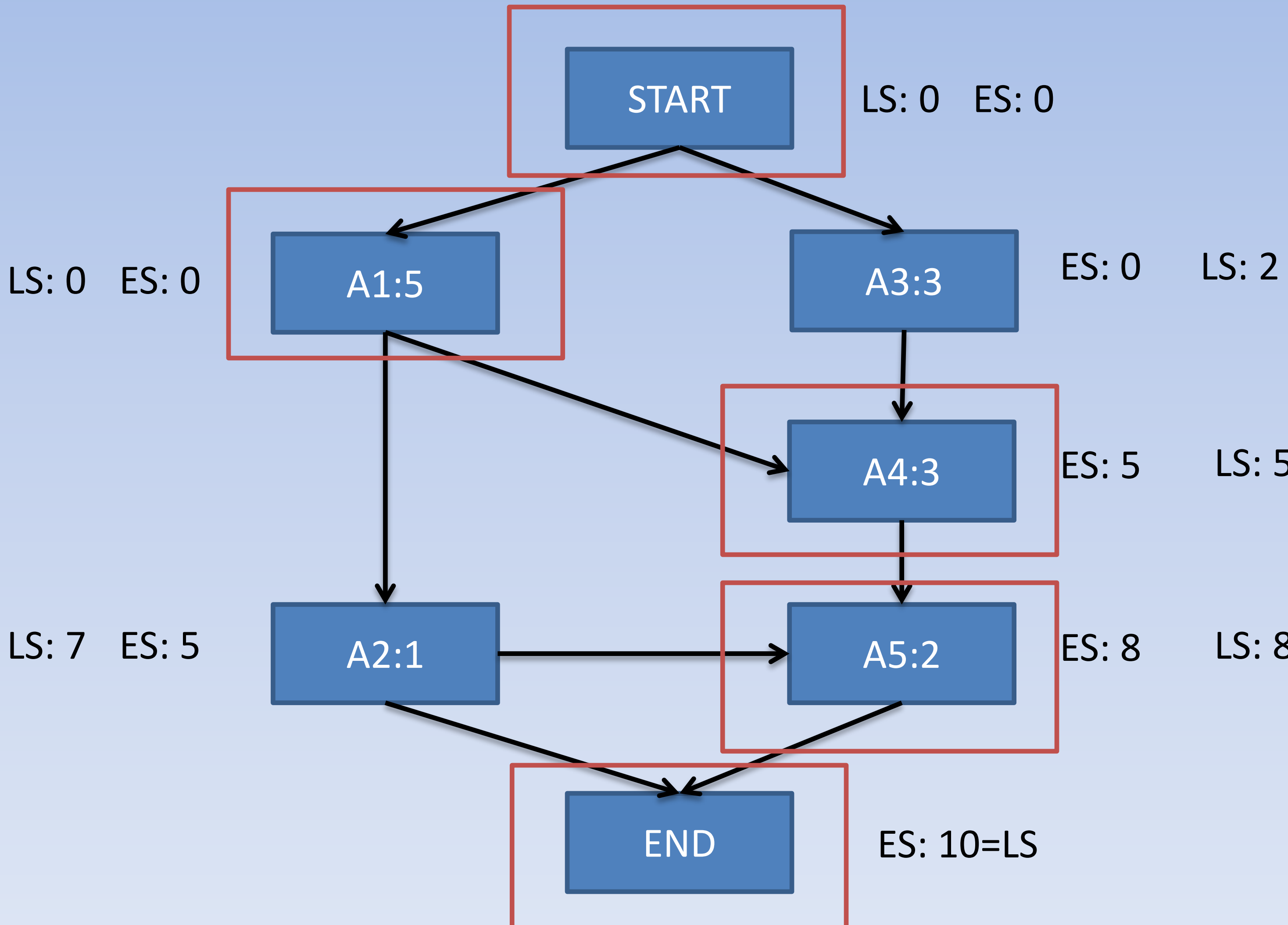
# Example

TS: Start, A3, A1, A4, A2, A5, End



# Example

Reverse TS: End, A5, A2, A4, A1, A3, Start



# Integrating Planning and Scheduling

- In general, scheduling after planning does not make sense since the choice of actions affects the schedule
- Generally, the planner must incorporate a scheduling step into the planning process
- For example, TGP (Temporal Graphplan, see website) is an extension of Graphplan that maintains *temporal mutexes* as well as “normal” mutexes
- However, in general, explicitly minimizing the makespan is a very hard problem; so in most cases, settle for “satisficing” solutions

# Resources

- Resources are artifacts in planning that generally *enable actions*
- Examples are fuel/energy/tools etc.
- Resources are of two kinds:
  - Consumable (Fuel)
    - Once an action uses it, the resource is gone
  - Renewable (Tools)
    - When an action needs it, it is “locked up”; once the action ends, it is available again

# Planning with Resources: Challenges

- Resources are numeric (not propositional)
- Resources  may be *created* during planning (not propositional)

- Resources introduce additional constraints between actions

- They make scheduling harder for durative actions

# Planning with Resources: Challenges

- Practical Issues:
  - Concurrency is necessary for short makespan
  - Actions must be selected in real-time

# Online Planning

- In the real world, we need to deal with noise and failures, action durations changing over time etc
- A good way to do this is to plan *online*
  - The planner prepares an initial plan
  - It starts executing
  - Something goes wrong; or just in general, the results start deviating from the expected values
  - The planner *replans* from the new state
  - Note that this requires computational efficiency; online planners typically drop optimality in favor of satisficing

# RTS Game Example: Wargus

# RTS Game Example: Wargus

# Online Planning for RTS resource collection