

CS 541 Final Exam

1. *The Emile paper by Gratch discusses the difference between intrinsic and extrinsic importance. What is the difference and how do plans facilitate their computation?*

A goal's intrinsic importance is a measure of the pure happiness promised by the achievement of the goal. For humans, there are probably very few goals that actually offer much intrinsic importance: sleeping, being accepted in a social community, and eating a burrito from La Costena in Mountain View spring to mind. Extrinsic importance is much more easily had, since this measures the extent to which a goal helps a person achieve other intrinsic goals. For example, getting an A in this course is an extrinsic goal, because it increases the chance that I may eventually get a high-paying job which can fund burrito purchases well into the future.

Since intrinsic importance must be more or less stipulated—or at best, reconnoitered through some kind of psychological experiment—planning can be of the most help in computing extrinsic importance. Gratch proposes a formula whereby the extrinsic importance of a subgoal is defined by computing the intrinsic importance of each impacted goal, and the extent to which that goal is impacted by the subgoal. This metric also includes a factor accounting for the “difficulty” of achieving the impacted goal, so that more extrinsic importance is assigned to those subgoals that help achieve difficult goals.

2. *Name two planners that use a truth criterion*

TWEAK – Chapman identified the modal truth criterion as a way of identifying when a proposition is necessarily true. A proposition p is necessarily true in a situation s iff two conditions hold: there is a situation t equal or necessarily previous to s in which p is necessarily asserted; and for every step C possibly before s and every proposition q possibly codesignating with p which C denies, there is a step W necessarily between C and s which asserts r , a proposition such that r and p codesignate whenever p and q codesignate. (alternatively substitute possibly for necessary for the possible TC)

SIPE – Uses a truth criterion that does not always enforce the ordering constraints that would ensure soundness (though final plan is guaranteed sound).

3. *What kind of approach would you suggest to an industrial plant manager that needs to control machines in an production plant, taking into account machine breakdown and production delays in real time? What planner would you think of using if she asked you to build a prototype of this system?*

Initially, this looks like the type of problem where some sort of uncertainty planning would be applicable. That way, machine breakdown could be factored into the decision. However, the DCAPS planner described in Rabideau et al, “Using Iterative Repair to

Automate Planning and Scheduling of Shuttle Payload Operations” seems like a near perfect fit for this problem. It is designed for scheduling the use of limited resources on a space vehicle, which looks like a similar problem to the control of machines in a factory. In both cases, the main problem is finding the optimal schedule for the use of a limited number of resources in order to achieve a certain set of goals in the minimum amount of time. Additionally, the authors of the DCAPS paper claim that their system can reschedule when there is an unexpected machine fault or a change in schedule.

4. The Prodigy system was presented as a system that learned search control knowledge in the form of rules that asserted preferences over, or rejected certain planning decisions. The Myers paper discussed how a user could alter the planning process through the use of advice. Discuss one similarity and one difference between search control knowledge and advice.

Search control knowledge and advice both prune nodes away from the planning tree. However, each method prunes different types of nodes. Search control knowledge prunes away nodes that are likely to be irrelevant to planning, in order to improve performance. For example, in the Blocksworld domain, appropriate search control knowledge might be to ignore all operators involving blocks that are not part of the goal state specification. Advice allows the user of the planning system to specify details about the desired plan, causing the system to ignore some nodes. However, in the case of advice, the idea is not to speed up planning, but to yield a plan that accommodates the user's specific desires. For example, in the travel domain, the user might give the system advice that she prefers to travel by hot-air balloon when possible. The planner would then try to return plans that satisfy this advice.

5. Show how the following operators would be encoded in a probabilistic planner such as Buridan:

To open a door the agent must be standing in front of it and must possess the key, which must be unbroken. With probability 0.75 the door is open when the operator is completed. With probability 0.25, the door is still closed and the key is broken.

To pick up a box, the box must be on the floor and the agent must be in the same room as the box. With probability 0.75 the agent is holding the box and it is no longer on the floor when the operator is completed. Independently with probability 0.75 the box is crumpled.

OPEN-DOOR(?d)

Preconditions: InFrontOf(?d) ; agent is in front of door

Possess(?key-d) ; agent possesses the key

~ Broken(?key-d) ; the key is not broken

Effect (two branches):

- 0.75 Open(?d) ; 0.75 probab. door is open
- 0.25 ~Open(?d) ; 0.25 probab. Door is not
Broken(?key-d) ; open and key is broken

(note: in the second branch, it may not be necessary to put ~Open(?d), since for that branch we could assume the door is in the state it is in before the action is done, whatever that state is).

For the PICK-BOX operator: first we have to calculate the probabilities. Since the results "holding box and box not on floor" and "box crumpled" are independent, we could just multiply the probabilities. We end up with 4 possible results.

PICK-BOX (?b)

Preconditions: On(?b, Floor) ; box is on floor
 In(?b, ?r1) ; box is in room 1
 In(Agent, ?r2) ; agent is in room 2
 ?r1 = ?r2 ; and those are the same rooms

Effects (4 branches)

- 0.5625 ; 0.5625 = 0.75 * 0.75
 Holding(?b) ; Box is held, not on floor anymore,
 ~ OnFloor(?b) ; and crumpled.
 Crumpled(?b)
- 0.1875 ; 0.1875 = 0.75 * 0.25
 Holding(?b) ; Got the box, which is not on floor
 ~ OnFloor(?b) ; and not crumpled.
- 0.1875 ~ Holding(?b) ; Not holding the box, but it is
 Crumpled(?b) ; crumpled
- 0.0625 ~ Holding(?b) ; Not holding the box, and it is not
 ; crumpled

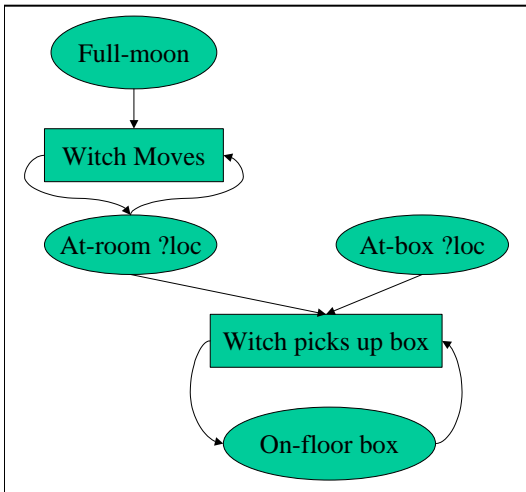
Note: a. We don't put ~Crumpled in branch 2 and 4 since we assume that is the situation the box is already in before the action. We only put literals changed by the operator. (And in case the box is crumpled already, it won't be correct to say ~Crumpled anyway).

b. Actually, it may not be necessary to state ~ Holding in the 3rd and 4th branch either, for the same reason as in note a.

6. Dorothy (D) is picking up and moving boxes using the operators you

described in the last question. Wicked witch (WW) shares the same world as D. Each time D takes an action, WW may move randomly from one room to another with probability 0.25, but only if there is a full moon. WW doesn't seem to need keys or doors. Also, if there is a box in the room with WW, she may pick up the box with probability 0.1 before she moves.

Dorothy has a plan to pick up a box in another room. Create an event graph to help her decide what facts to consider to compute the probability of success for her plan. Does Dorothy need to worry about the full moon?



Yes, Dorothy does need to worry about the full moon. Even if the moon isn't full, the WW might start off being in the same room as the box, in which case she might pick it up. But if the moon is full, then the witch has a greater chance of messing up Dorothy's plan, because as the turns pile up, she'll eventually end up in the room with the box. This constrains Dorothy's plans, because shorter plans are more likely to succeed.

7. If Dorothy switches to a Markov decision problem (MDP) to model her box-picking plan, she will model her potential actions in each world state and solve the MDP to find the best action in each state. But the MDP has no explicit model of the wicked witch's actions or any other external events. How does this information get reflected in the MDP that Dorothy builds?

In this case, the effects of external events will be incorporated by:

1. Adding the state of the witch (which room she is in) explicitly as a literal in each state. This means multiplying the number of states by 2.
2. Statement no. 1 means we are implicitly incorporating the effects

of the witch's events in the effects of Dorothy's actions in the domain. Note that this is possible since the witch only moves when Dorothy makes an action.

For instance, the effects of Dorothy moving to the other room should have a transition to all states that are combinations of Dorothy's, the witch's, and the box's new state.

Note that without an event graph this might be inefficient because the effects of events are duplicated in every operator in the domain, and each operator is implicitly modeling many events.

8. In Distributed Negotiated Planning, agents are typically self-interested. Discuss why 2 cooperative planning agents (that share a common goal) might still need to negotiate.

They must coordinate plans as well as goals. The agents need to make sure that their plans are compatible. Without coordination, agents might solve the goal in completely different ways, rendering their teamwork meaningless as the agent with the quickest plan does everything without help from the other agent. Even worse, there might be some resource conflict that will actually prevent either agent from executing their plan. I.e. Both agents plan to exit a room through the same door at the same time. They may each have planned around this constraint ("I'll go first, then he will") but unless they coordinate, or are lucky (both assumed agent 1 would go first), they will run into problems.

9. Suppose that you are reading a paper that says "Our planner generated plans that were often shorter than other planners used in the test. After doing the test, we found some errors in the operators that this planner used for the test. We believe that fixing these errors will improve the planner's performance further." How would you explain what happened in the test? Do you agree with the authors in their hypothesis?

The author's did not say if their planner's plans were correct or not with regards to the underlying domain theory. My assumption would be that the errors in their operators led to situations where actions that should be necessary were not, and thus not planned for. Suppose a precondition in a transportation domain is that each truck is full of gas before setting out on a run. Omitting this precondition from the "move-truck" operator would enable their planner to generate plans that didn't involve refueling and of course therefore would be shorter than those generated by a system that is worrying about refueling. I therefore do not agree with their hypothesis. I think that if anything, the "performance" of their system will decline with correct operators.

10. One of the papers we read in class described how the threat

detection/resolution mechanisms of POP-style planners could be used to assist multi-agent planning. Which paper, and how did the author(s) suggest this was to be done?

The paper that you are describing is by Gratch, "How to make your planner rude...." The main idea is that planning agents can take different "stances" to other agents, depending on how they handle threats. There are a variety of stances that I, as a planning agent can take. At one extreme, I can be adversarial, deliberately introducing threats into other agents' plans, even if they don't help my agenda. I can be authoritative, demanding that other agents alter their plans in order to remove threats to my own plans. I can be rude, ignoring threats that I introduce into the plans of other agents. I can be meek, never introducing threats into anyone else's plan, and trying to work around threats that are introduced to my plan by others. Or, at the opposite extreme from the adversarial stance, I can be helpful, adding tasks to my own agenda to ensure that other agents reach their goals.