CS188 Midterm — Shawn Chang, Bob Zasio

Solutions and Grading Scale

March 15, 1999

Problem 1

Part A

False. Suppose we have the two statements:

1. $A \rightarrow B$
2. $\neg A \rightarrow B$

Forward chaining cannot figure out that B is true, even though it obviously is.

Part B

False. Although partial-order planning is sound and complete, we still might have chosen a bad operator to achieve some effect or a bad strategy for resolving a threat (promotion or demotion). In that case, we have to back up to some previous choice point and make another try. Notice that selection of which precondition of the final goal is to be achieved first is not a choice point. For more detail, refer to page 355-356 of your textbook.

Part C

True. This is the basic property of A* search.

Part D

False. Mother, Female, and Parent are just symbols until we have given them meaning. The sentence on its own doesn't do this, and thus has no logical meaning.

Part E

False. Iterative deepening is slower, but not asymptotically slower, which means slower by more than a constant multiple. Although iterative deepening searches some nodes many times over, the total number of nodes looked at (including repetition) is proportional to the number of unique nodes viewed.

Problem 2

Part A[123]

1. A B C D E F G H I J K
2. A B C G H D E F I J L K
3. A B C D E F G H J K M L

The key idea for A* is that even when we've found a solution, we can't stop until we've expanded every node with a sum cost less than our solution, to prove we've found the shortest.

Part A4

Constraint satisfaction would be best because we don't care about the order at which the solution is achieved, and path-finding wastes time searching multiple approaches that reach the same result, as well as adding difficulty in keeping ourselves from searching loops. Many people wrote constraint satisfaction, but in their explanation, were confused as to what exactly the difference between that and path-finding is.

Part A5

A* would still be guaranteed. An admissable heuristic is one that always gives a value less than or equal to the actual distance we are from a solution. Since truncation can only decrease the result, the result is still admissable.

Problem 3

Part A

1. $\forall x Ind(x, Mercedes) \rightarrow Ind(x, Car)$
2. $\forall x Ind(x, Ferrari) \rightarrow Ind(x, Car)$
3. $\forall x Ind(x, Car) \rightarrow \exists y Ind(y, Engine) \wedge PartOf(y, x)$

3 points for each sentence. Some students had different translations for the sentence 3, and full credits were given as long as they were sytactically and logically correct. 1 point partial credit was given for sentence 3 for those who misused universal quantifier for the existential quantifier.

Part B

BillsCar is *either* a Mercedes or a Ferrari:[1]

1 point partial credit was given if the translation to the additional fact was correct.

[1]The "either...or" here can be interpreted as normal disjunction or exclusive or. We accept either translation since the proof can be carried out with either one. When interpreted as exclusive or, the sentence becomes: $(Ind(BillsCar, Mercedes) \wedge \neg Ind(BillsCar, Ferrari)) \vee (\neg Ind(BillsCar, Mercedes) \wedge Ind(BillsCar, Ferrari))$.

$$Ind(BillsCar, Mercedes) \vee Ind(BillsCar, Ferrari) \tag{1}$$

What you are trying to prove is that:

$$\exists x Ind(x, Engine) \wedge PartOf(x, BillsCar). \tag{2}$$

1 point partial credit was given if the translation for the goal is correct.

To prove the goal, we need to apply the third sentence in part A, plus the fact that $Ind(BillsCar, Car)$. Recall that backward chaining uses *Generalized Modus Ponens*, which can only be applied to *Horn* sentences. Hence, at least one difficulty arises while trying to prove the fact $Ind(BillsCar, Car)$: Sentence 1 cannot be written as a Horn sentence. So the generalized Modus Ponens cannot be applied. Refer to section 9.5 of the textbook for a similar example, and page 174 of textbook for the defination of Horn sentences.

Full credits were given if some variant of the above statements were written to point out the Horn sentence and Modus Ponens problem. Partial credits were given if students could describe "which sentence could not be resolved", "we need resolution", or other valid points. The amount of partial credit depends on how close to the main point.

Part C

Translation into Clausal forms (Steps are omitted):

- (1) $\{\neg Ind(x1, Mercedes), Ind(x1, Car)\}$ (from sentence 1)
- (2) $\{\neg Ind(x2, Ferrari), Ind(x2, Car)\}$ (from sentence 2)
- (3) $\{\neg Ind(x3, Car), Ind(EngOf(x3), Engine)\}$ (from sentence 3)
- (4) $\{\neg Ind(x4, Car), PartOf(EngOf(x4), x4)\}$ (from sentence 3)
- (5) $\{Ind(BillsCar, Mercedes), Ind(BillsCar, Ferrari)\}$ (from additional fact)

Note that sentence 3 becomes two sentences and a skolem function EngOf() has to be used.

Negated goal becomes:

- (6) $\{\neg Ind(x6, Engine), \neg PartOf(x6, BillsCar)\}$

Note we don't need skolem function for the negated goal. The reason is that the existential quantifier in the original goal becomes universal quantifier at negation.

Inference steps:

- (7) $\{Ind(BillsCar, Fer), Ind(BC, Car)\}$ $(1+5)$ $\{x1/BC\}$
- (8) $\{Ind(BillsCar, Car)\}$ $(2+7)$ $\{x2/BC\}$
- (9) $\{Ind(EngOf(BC), Engine)\}$ $(3+8)$ $\{x3/BC\}$
- (10) $\{PartOf(EngOf(BC), BC)\}$ $(4+8)$ $\{x4/BC\}$
- (11) $\{\neg PartOf(EngOf(BC), BC)\}$ $(9+6)$ $\{x6/EngOf(BC)\}$
- (12) $\{\}$ $(10+11)$ $\{\}$

Hence, the original goal must be true.

QED.

There are other possiblly correct proofs (or with different ordering of steps). Basically, we gave 1 point for each correctly written clausal form sentence or resolution step, up to 8 points. The rest 3 points were only given (or partially) if your proofs indeed lead to the correct answer. Points were deducted for incorrect clausal form sentences, not using skolem function, or not clearly stating the facts you were resolving.

Problem 4

Part A

The problem is refered to as the *Frame Problem*. Refers to page 207 of textbook for a detailed explaination.

The representational aspect: A large number of *frame axioms* are needed to describe how the world stays the same when an action is applied.

The representational aspect of the frame problem is addressed by using the *successor-state axioms*, while remaining with a purely deductive framework.

The computational aspect: When reasoning about the result of a long sequence of actions in situation calculus, one has to carry each property through all intervening situations one step at a time, *even if the property remains unchanged throughout*. This is true whether one uses frame axioms or successor-state axioms.

2 points for the defining representational aspect of the problem, 3 points for stating how the problem can be addressed, and 2 points for the computational aspect.

Part B

The important point is that a planner with a STRIPS-style of operator representation assumes the things that were initially true remain true unless explicitly changed by an operator. The precondition, add, and delete lists of the STRIPS operator enable one to make only changes required by the action description without recompute the rest of the world. This is certainly a big gain in efficiency since we only expect a small fraction of the world to change by an action.

Full credits were given if the "important point" decribed above was stated clearly and correctly. Partial credits (2 to 5 points) were given if there were some mentioning of precondition, add, delete lists, or how a planner retains a state of world in the knowledge base and modifies it. Small partial credits were also given to other valid points that did not quite answer the question directly.

Problem 5

Part A

```
Marry-Daughter(Usurper, Richard)
pre: Single(Usurper)
add: SonInLaw(Usurper, Richard), Married(Usurper)
del: Single(Usurper)
```

```
Divorce-Daughter(Usurper, Richard)
pre: Married(Usurper)
add: Single(Usurper)
del: Married(Usurper), SonInLaw(Usurper, Richard)

Assasinate(Usurper, Richard)
pre: King(Richard), SonInLaw(Usurper, Richard)
add: King(Usurper)
del: King(Richard)
```

Note that according to the diagram, SonInLaw(Usurper, Richard) is not a prerequisite for Divorce-Daughter(Usurper, Richard), but if it is true, the operator removes it.

Part B

Yes. In our current plan, it's possible to divorce the daughter before killing the king, removing the precondition of SonInLaw for Assassinate. We resolve this by promoting Divorce-Daughter to happen after Assassinate. The planner is not allowed to modify the operators, so any suggestion along the lines of changing the preconditions of Divorce-Daughter is wrong.

Part C

Just saying "It makes things more complicated" really doesn't tell us much. The key thing to notice is that now when the planner sees that Assasinate requires SonInLaw, it can't find an operator that makes this directly true. It now has to infer SonInLaw from the implication, and decide whom to marry.