Homework III Solutions

Question I

A) Optimal parallel plan:

A1
A2

B) Graphplan plan:

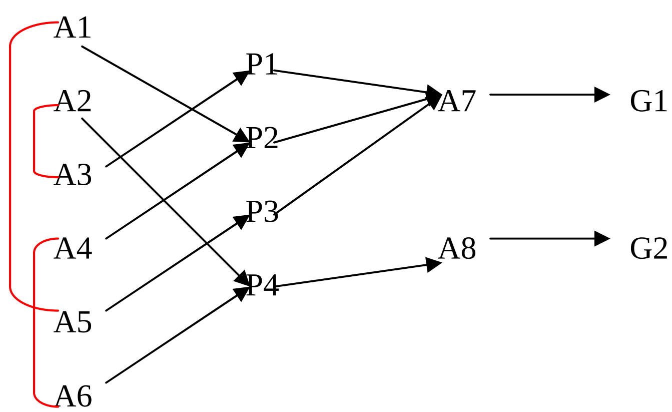
A1 – A2.
No step optimal

C) We can change the action interference definition such that two actions A1 and A2 are interfering if their interacting effects are not relevant, in other words, if they do not belong to the list of propositions that need to be supported. Following our example, A1 and A2 would not be interfering since their interacting effects W and ~W do not need to be supported. We only need to support P and Q. This change would not be easy because we have to check such condition with each action introduced at each level, and more important we also have to identify which subgoals are relevant with respect to the top level goals at each level of the graph.

Question II.

A) Because there will be additional actions supporting such conditions (at least Noops), relaxing the mutex propagation.

B) No. When higher order mutexes are involved we may miss additional pair mutexes. For example:

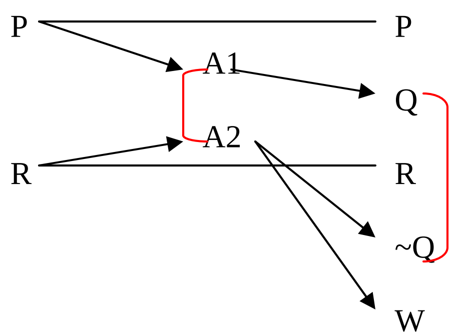


For this example, although we are able to produce P1 to P4 by the individual actions, we can not achieve the whole set. In consequence, GP can not find that A7 and A8 are mutex, and that G1 and G2 are not possible.

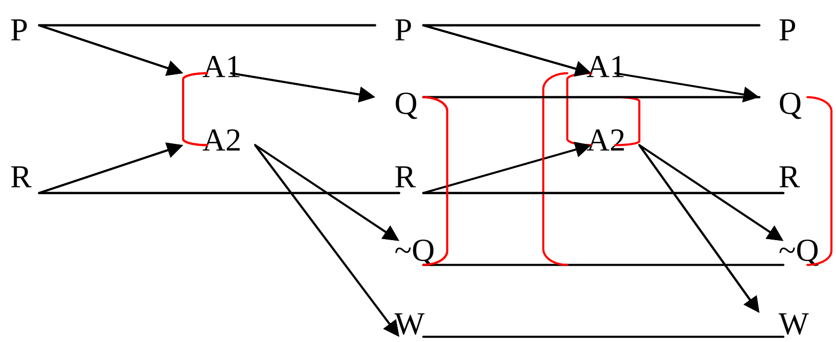
C) Yes. Just do a serial planning graph. A serial planning graph is a graph in which each pair of non noop actions is marked as mutex. Forcing only one action at each step of the solution.

Question III.

A)

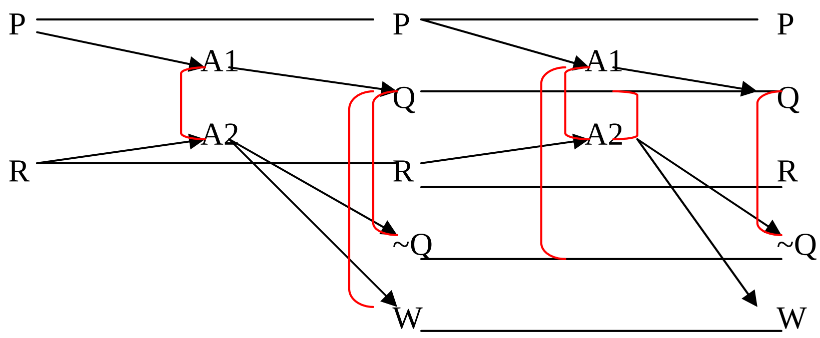


Since only static interference is considered we stop at level 1 of the graph construction phase, and start searching for a solution. Given that Q and W are not mutexes. So, we can support Q with A1, and W with A2. At this time we stop since A1 and A2 are mutex with each other. Given that there are no more choices for the subgoals, we write a memo at level 1 {P,Q: 1} giving the explanation for the failure.



At level 2, we can search again for a solution. This time, we can choose to support Q with A1, and W with its persistent action. We have to satisfy then the preconditions of such actions, subgoals P and W. P can only be supported by its persistent action, and W by A2. This time we have regressed to the initial state, and we have found a solution to our problem: A2-A1.

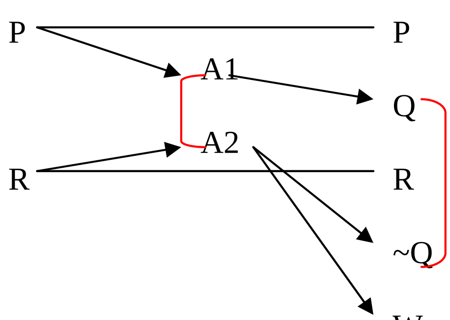
B) With normal mutex propagation, we have to build our planning graph up to level 2, without search because even tough our goals are present at level 1, they are also mutex to each other.



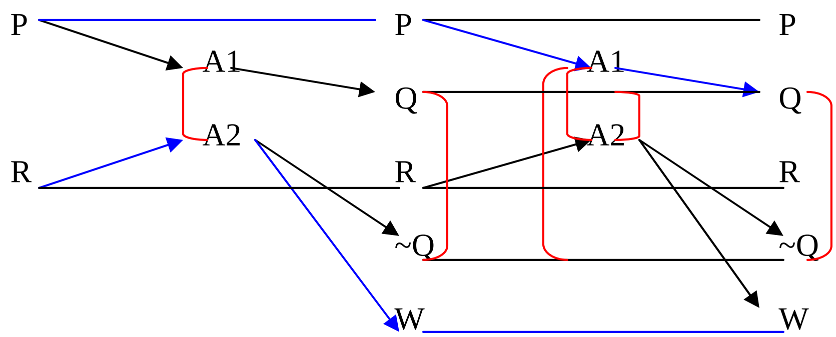
The search is conducted in a similar way to that one of III.a , finding the same solution. A2-A1.

C). Memos can be considered as finding mutexes during search, some of them could be higher order ones (e.g. more than 2 relations). In order to find a Memo, we need to fail first, and find the explanation for such failure. So, memos are built on a demand driven basis while mutexes are computed globally during the propagation of the planning graph. If mutex propagation gets relaxed as in III.a, more memos will be found since we may fail more times (we are not seeing the mutexes in front), such memos will be then used as kind of mutexes to prune inconsistent states later during search.

Question IV.



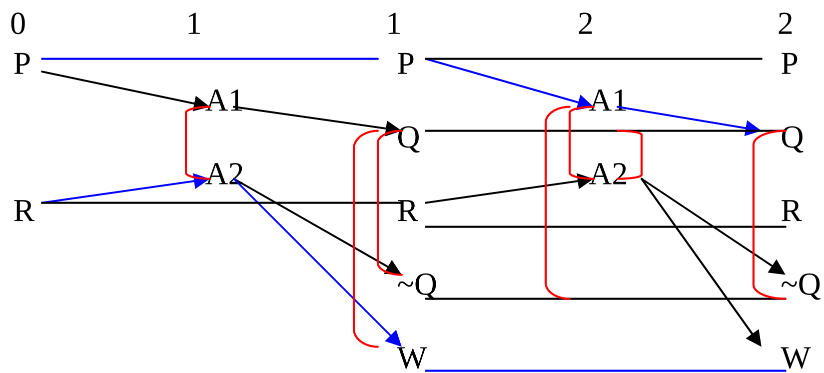
Initial conflict sets are: Q={Q} and W={W}. First, we support Q with A1 and then W with A2, but we find that such assignment is incon sistent because both actions are mutexes. So, the conflict set for W is updated to {Q,W}. We then backtrack to our precious choice, in this case Q, but we find it does not have any more values, so Q absorbs W's conflict set. Given that we can not satisfy the goals Q and W, we build the planning graph one more level and we store {Q,W:1} as a memo at level 1.



At level 2, we can search again for a solution. This time, the conflicts sets are Q={Q} and W=W. We can choose to support Q with A1, and W with its persistent action. We have to satisfy then the preconditions of such actions, subgoals P and W. P can only be supported by its persistent action, and W by A2. This time we have regressed to the initial state, and we have found a solution to our problem: A2-A1.

Question V.

From III.b



A) CSP Encoding
Variables and Domains:

P-2 : {#, NooP-P2}, Q-2:{#,NooP-Q2,A1-2}, R-2:{#,Noop-R2}, ~Q-2:{#,Noop-~Q2,A2-2}, W-2:{#,NooP-W2,A2-2}

P-1: {#, NooP-P1}, Q-1:{#,A1-1}, R-1:{#,Noop-R1}, ~Q-1:{#,A2-1}, W-1:{#,A2-1}

P-0:{#,T}, R-0:{#,T}

Goals: Q-2!=#, W-2!=#

Activation Constraints:

P-2=NooP-P2 => P-1!=#, Q-2=NooP-Q2 => Q-1!=#, Q-2=A1-2 => P-1!=#
R-2=NooP-R2 => R-1!=#, ~Q-2=NooP-~Q2 => ~Q-1 !=#, ~Q-2=A2-2 => R-1!=#
W-2=NooP-W2 => W-1 !=#, W-2=A2-2 => R-1!=#

P-1=NooP-P1 => P-0!=#, Q-1=A1-1 => P-0!=# R-1=NooP-R1 => R-0!=#,
~Q-1=A2-1 => R-0!=#, W-2=A2-1=> R-0!=#

Mutex Constraints:

Q-2=A1-2 => ~Q-2!=A2-2, Q-2=A1-2 => W-2!=A2-2,
Q-2=A1-2 => ~Q-2!=NooP-~Q2, Q-2=Noop-Q2 => ~Q-2!=A2-2,
Q-2=Noop-Q2 => W-2!=A2-2, Q-2=Noop-Q2 => ~Q-2!=NooP-~Q2,
~Q-2=A2-2 => Q-2!=A1-2, ~Q-2=A2-2 => Q-2!=NooP-Q2,
~Q-2=NooP-~Q2 => Q-2!=A1-2, ~Q-2=NooP-~Q2 => Q-2!=NooP-Q2
W-2=A2-2 => Q-2!=A1-2, W-2=A2-2 => Q-2!=NooP-Q2,

Q-1=A1-1 => ~Q-1!=A2-1, Q-1=A1-1 => W-1!=A2-1,
~Q-1=A2-1 => Q-1!=A1-1, W-1=A2-1 => Q-1!=A1-1,

Solution:

Q-2=A1-2,	P-2=NooP-P2,	R-2=NooP-R2,	~Q-2=#,	W-2=#
Q-1=#,	P-1=NooP-P1,	R-1=NooP-R1,	~Q-1=A2-1	W-1=A2-1
P-0=T,	R-O=T			

B) SAT Encoding
Initial State: P-0 & R-0
Goal State: Q-2 & W-2

Graph Propagation: (cond x at k => one of its supporting actions)
P-2 => NooP-P2 Q-2 => NooP-Q2 || A1-2 ~Q-2 => NooP-~Q2 || A2-2
R-2 => NooP-R2 W-2 => NooP-W2 || A2-2
P-1 => NooP-P1 Q-1 => A1-1 ~Q-1 => A2-1
R-1 => NooP-R1 W-1=>A2-1

Actions => Preconditions
A1-2 => P-1, A2-2=> R-1, NooP-P2 => P-1, NooP-Q2=>Q-1,
NooP-~Q2=>~Q-1, NooP-R2=>R-1, NooP-W2=>W-1,
A1-1 => P-0, A2-1=> R-0, NooP-P1 => P-0, NooP-R1=>R-0,

Mutexes:
~Q-2 || ~(~Q-2), ~A1-2 || ~A2-2, ~A1-2 || ~NooP-~Q2
~A2-2 || ~NooP-Q2, ~Q-1 || ~(~Q-1), ~Q-1 || ~W-1, ~A1-1 || ~A2-1

Solution: A2-1 & NooP-P1 & NooP-W2 & A1-2