Okay, so let us start. Could you explain how we defined operators and actions when doing classical planning?

An operator is like a template for an action. It has parameters, and preconditions and effects which depend on these parameters. And if you ground the operator, you get a set of actions which have no parameters anymore but only atomic preconditions and effects. Preconditions express in which world states an action is applicable, and effects express what happens to the world if the action is applied.

Yes. Say someone shows you a PDDL planning domain with an operator which has ten parameters of type "car", and in the problem file you find twenty "car" objects. What would you think about that planning model?

It's problematic because the number of produced actions will explode. You need to insert every possible parameter combination into the operator definition, so that's potentially $20^{10}$ actions after grounding, and that's definitely too much regarding runtime and memory constraints.

While we are talking about efficiency: What is the general complexity of planning?

Planning is PSPACE-complete. So it can be solved with a polynomially space-restricted Turing machine, and planning is at least as hard as every other PSPACE-complete problem.

Could you sketch how we can simulate a Turing machine with a planning problem?

You can convert a Turing machine into a planning problem if you interpret the machine's moves as actions for planning. And there was something with conditional effects, I think … Alright. Are there also some classes of planning problems which are easier than PSPACE?

Yes, for example if there are only positive preconditions and effects, you can find a plan easier, I think it was either polynomial or NP-complete.

How could such an "easy" algorithm work?

Well, you can begin with the initial state and then apply some applicable action and get a new state, and you can repeat this until the goal is contained in the state. And then all of the applied actions together form a valid plan.

Yes, that is indeed a polynomial-time algorithm. And what if there is no solution?

Then at some point you would reach a point where the state cannot change any more, but the goal is still not reached. Then you can let the algorithm return unsatisfiability.

Now that you already talked about that planning subclass with positive preconditions and effects only: Does it have some actual use for planning?

I'd say that it's not useful, because only trivial planning problems can be expressed in such a way.

Well, okay, so it's not a realistic planning model in itself, but maybe there is some other use for it? Say we write a general planning problem without its negative preconditions and effects, how could this help?

Ah, yes, it can be used for heuristics.

Exactly. Do you know which heuristic does something like that, and how exactly that works?

It's the Fast Forward heuristic. Basically, if you want to evaluate a state, you find an optimal, or near-optimal plan for the delete-relaxed problem from that state to the goal. And you take the cost of that relaxed plan as the heuristic value for the state.

Does it matter if the found relaxed plan is optimal?

The heuristic is only admissible if you always extract optimal relaxed plans. So if you don't do that, the planner using this heuristic will not do optimal planning any more.

To which kind of general planning technique does this heuristic belong?

* Disclaimer: Not an actual exam protocol. Fake report of an imaginary exam with purely fictional people.
It’s used for Forward state-space search planning.

Yes. And are there other planning strategies which also make use of heuristics in some way?
   Almost all of them do, for example in GraphPlan the backwards search needs some heuristic as well.

Are heuristics important for scheduling problems as well? Why, or why not?
   I’d say they are less important than in planning, because most scheduling problems are simple to solve compared to planning problems, and then you can have a direct algorithm without any heuristic search. But some problems are NP-complete, and then I guess it makes sense to use heuristics in many cases, like if you do a branch-and-bound algorithm for example.

Can you provide an example for such a scheduling problem in Graham notation?
   Uhm ... I don't remember the Graham notation unfortunately.

Okay, then back to planning.

Suppose that you have a number of planning problems, and you have a fixed time frame to solve each of these problems, say ten seconds per problem. Your planner solves each problem extremely fast, but finds extremely long plans. How could you extend your planning approach such that it makes use of the whole timeframe and leads to plans of higher quality?
   A plan optimization approach could be used which works anytime, so we can just cancel it after ten seconds. For example we could employ the sliding window technique with an increasing window size, and remove redundant actions from each of the small sub-problems with MaxSAT or something.

How does such a MaxSAT based sub-procedure work?
   You can basically take the usual SAT encoding which encodes a planning problem for k steps, and k is just the length of the found plan for that window. But you don't have variables for every action in the problem, but only for the original action. And then you can add soft clauses for each step saying that you don't want to execute the action, and a MaxSAT solver can find the assignment where the maximum number of actions is omitted.

Will this technique eventually result in an optimal plan if our window has the size of the entire problem in the end?
   No, it just removes redundant actions, there might still be a much better plan which works with a completely different sequence of actions.

Maybe, at the end, you can elaborate a bit on the determinism assumption that we made when doing classical planning, and how this assumption can be relaxed?
   In classical planning, when we execute an action, we know exactly what will happen. Instead, you can use non-deterministic models like Markov Decision Processes if the state transitions are probabilistic in some way. Then you have a probability distribution $P_a(s'|s)$ for each action $a$, and you can only pick the action which has the expected maximum utility.

What is utility in that case? Maybe you should explain first how the overall objective of the planning problem changes from deterministic to probabilistic planning.
   Yes, so in classical planning you have a goal which you want to reach. But in probabilistic planning you do not want a plan leading to a goal, but instead a policy which maps states to actions. And the policy should lead to the maximum expected utility, which rates states and actions regarding their cost and reward.

Okay, time's up.

Thorsten Rucking received the grade 1.3 for his overall great knowledge on the covered topics, but some insecurities regarding planning complexity and scheduling problems. In addition, as Thorsten eagerly participated in the exercises, he received a grade improvement of 0.3 and thus got the final grade 1.0.

* Disclaimer: Not an actual exam protocol. Fake report of an imaginary exam with purely fictional people.