Table 1: New version of Table 3 in our submitted paper. Running time of FF (general AI planner), Fast Downward Stone Soup (winner of Satisfying track 2018 International Planning Competition; a top performing general AI planner) and Sokolution (top Sokoban specialized solver) on the instance Sasquatch7_40 with a 210-step solution. Instances are built pulling backward from the goal and show increasing difficulty.

FF (AI planner)						
steps	<40	50	60	70	80	
time	<10s	3min	21min	2h	>12h	
Fast Downward 2018 (AI Planner)						
steps	<60	70	80	90	100	110
time	<21s	5min	17min	58min	3h	>12h
Sokolution (specialized Sokoban solver)						
steps	<110	120	130	140	150	160
time	<20s	52s	3min	22min	4h	>12h

We thank the reviewers for their careful and detailed reviews. **I Scope of contribution** We agree with several of the reviewers that we stated the title and introduction too broadly about AI planning, while we focus on the Sokoban domain. Following the suggestion by reviewer # 3, we will change the title to "A Novel Automated Curriculum Strategy to Solve Hard Sokoban Instances." We did select this domain because we know this problem to be an extremely hard combinatorial AI planning task, with many open unsolved instances that beyond the reach of all other approaches (both specialized and general solvers). Our approach solves those instances. We will also narrow the scope of the introduction. We should note though that we believe the ideas we used are sufficiently general to extent to other planning domains. E.g., to solve a very hard unsolved planning instance, we can create a series of easier sub-instances by removing grounded predicates from the goal state. However, this is for future work.

II Comparison to State-of-the-Art Solvers and Baseline The reviewers are totally correct that we should have used a more recent AI planner. We ran experiments with the 2018 winner of the planning competition, Fast Downward Stone Soup. See results in the new table 3 above. We see indeed a significant improvement of about 30 more steps over 20 years of AI planning technology. Within 12 hrs compute time, FF cannot find plans further than 80 steps from the goal; Fast Downward cannot go further than 110 steps away; the specialized Sokolution solver cannot go further than 160 steps away. Our approach finds a solution from the original start state at 210 steps away. We again stress that we are solving instances that are not solved by any other method.

Reviewers # 1 and # 2 suggest we only compare to FF and ask about a baseline and other solvers. First we note that we can only compare to the "weakened"instances, with initial states placed closer to the goal, because our real contribution is in solving the full original instances that are not solved by any previous method, including the specialized solver Sokolution (which itself already greatly outperforms general AI planners or any other known RL results on Sokoban). Earlier work on RL for Sokoban, eg by the DeepMind group, could only solve some of the known instances that are trivial for eg Sokolution (solved in seconds). So, we do believe our approach is an advance even for RL.

III Curriculum Strategy and contrast with paper [11]. GET NUMBERS!! As various reviewers noted, a key novelty is the new curriculum strategy combined with sub-instance approach but also several other innovations as highlighted in the ablation studies. Overall, we solve 179 of the total of 225 known open problem instances. The approach presented in [11] only solves dozens of the open problems (in under 12 hrs each). So, we do believe our framework significantly extents that of [11]. More work can be done on studying the bandit instance selection but one core finding is that we do not have the "forgetting" problem as observed in [11]. Our RL policy continues to improve without "forgetting" how to solve earlier instances. This effect is due to our bandit strategy that keeps some easy instances around to retain the basic strategies.

As pointed out by reviewer # 1, we will state more clearly that we learn from unsolved (unlabeled) sub-instances. This is a core feature of the approach.

The work on automated goal generation in robotics (Florensa et al. 2018) is related (reviewer # 2). However, there the emphasis is on learning to operate in more diverse settings. Our approach with the curriculum training and pool of sub-instances is needed to build towards solving a particularly hard unsolved instance. The bandit strategy carefully pushes the training pool to increasingly challenging problems, to finally solve the original hard instance.

We thank Reviewer 3's four detailed and constructive questions. For the clarity, we have stated hyperparameters in the main paper in different places and we will put them together for more clarity. Though our method used more GPUs, the main motivation of Table 3 is to show the exponential scaling of these solvers as the size of sub-instance increases so that there is no hope for these solvers to solve the original instance due to exponential explosion.