Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI-16)

# Blind Search for Atari-Like Online Planning Revisited

**Alexander Shleyfman** and **Alexander Tuisov** and **Carmel Domshlak**
Faculty of Industrial Engineering and Management
Technion, Israel

## Abstract

Similarly to the classical AI planning, the Atari 2600 games supported in the Arcade Learning Environment all feature a fully observable (RAM) state and actions that have deterministic effect. At the same time, the problems in ALE are given only implicitly, via a simulator, *a priori* precluding exploiting most of the modern classical planning techniques. Despite that, Lipovetzky et al. [2015] recently showed how online planning for Atari-like problems can be effectively addressed using IW$(i)$, a blind state-space search algorithm that employs a certain form of structural similarity-based pruning. We show that the effectiveness of the blind state-space search for Atari-like online planning can be pushed even further by focusing the search using both structural state similarity and the relative myopic value of the states. We also show that the planning effectiveness can be further improved by considering online planning for the Atari games as a multiarmed bandit style competition between the various actions available at the state planned for, and not purely as a classical planning style action sequence optimization problem.

## 1 Introduction

Since its introduction in 2013, the Arcade Learning Environment (ALE) draws a growing interest as a testbed for general, domain-independent planners and learners through a convenient interface to numerous Atari 2600 games [Bellemare *et al.*, 2013]. These games all feature a fully observable state and actions that have deterministic effect. At the same time, both the action dynamics and the reward structure of the games are given only implicitly, via a simulator, bringing the ALE setup closer to the challenges of real-world applications.

ALE supports two settings of action selection problem: an *online planning* setting where each action selection is based on a relatively short, simulated lookahead, and a *learning* setting that must produce reactive controllers for mapping states into actions after a single long session of interactions with the simulator. In this work we consider the online planning setting.

The implicit, simulator-based problem representation in ALE precludes automatic derivation of heuristic functions and other inferences developed in the scope of classical planning [Ghallab *et al.*, 2004; Geffner and Bonet, 2013]. This state of affairs restricts the algorithmic choices to blind (aka not future estimating) search algorithms, such as blind best-first search and Monte-Carlo tree search algorithms. The first results on domain-independent online planning in ALE have been reported by Bellemare et al. [2013], where the search algorithm of choice was the popular Monte-Carlo tree search algorithm UCT [Kocsis and Szepesvári, 2006]. In particular, UCT was shown there to substantially outperform breadth-first search (BrFS). The latter result probably came at no surprise since blind best-first search methods such as BrFS are inherently ineffective over large state spaces. Recently, however, Lipovetzky et al. [2015] showed that this is not the end of the story for breadth-first search. In particular, they showed that IW$(i)$, a pruning-enhanced successor of BrFS originated in work in classical planning [Lipovetzky and Geffner, 2012], favorably competes with UCT on the Atari games.

In this work we show that the effectiveness of blind state-space search for deterministic online planning in Atari-like problems can be pushed even further by focusing the search using both structural state similarity and the relative myopic value of the states. We introduce and evaluate *prioritized IW*$(i)$, a simple extension of IW$(i)$ that approximates breadth-first search with duplicate detection and state reopening, and show that it very favorably competes with IW$(i)$ on the Atari games. We then revisit the basic objective underlying deterministic online planning. We argue that the effectiveness of online planning for the Atari games and related problems can be further improved by considering this problem as a multiarmed bandit style competition between the actions available at the state planned for (and not purely as a classical planning style action sequence optimization problem). Following this lead, we introduce a simple modification of prioritized IW$(i)$ that fits the modified objective, and empirically demonstrate the prospects of this direction.

## 2 Background

The Atari 2600 games exposed by ALE represent a broad range of problems that are characterized by means of a finite set of states, with each state being represented by a complete assignment to some $n$ finite domain variables $X_1, \ldots, X_n$,

an initial state $s_0$, a finite set of actions, a transition function $s' = f(a, s)$ where $s'$ is the state resulting from applying action $a$ in state $s$, and real-valued rewards $r(a, s)$ that result from applying $a$ in $s$. The transition function and rewards in ALE are implemented by a game simulator and thus are not known to the planner *a priori*. At the same time, the environment is fully observable: when applying action $a$ is simulated in state $s$, the resulting new state $f(a, s)$ and the collected reward $r(a, s)$ are revealed to the planner. The state of the game is simply captured by the content of Atari's RAM of 128 bytes. Different choices of factoring this RAM into a set of state variables are possible, and, as it is typically the case with feature generation in learning, this choice of factoring may have a substantial impact on the planner's performance. This aspect of the problem is tangential to our focus here; for ease of comparability, we adopt the previous work's factoring along 128 variables, each representing the value of the respective memory byte, and thus having the domain of 256 values [Bellemare *et al.*, 2013; Lipovetzky *et al.*, 2015].

IW$(i)$, an algorithm that has recently been shown by Lipovetzky et al. [2015] to exhibit state-of-the-art performance on the Atari games, is a regular breadth-first search with the following modification: When a state $s$ is generated, it is assigned a *novelty* penalty, and is pruned if the penalty is higher than $i$. The novelty penalty of a newly generated state $s$ is 1 if $s$ is the first state generated in the search that makes true some atom $X = x$, else it is 2 if $s$ is the first state that makes a pair of atoms $X = x \wedge Y = y$ true, and so on. If the problem state is represented by $n$ atoms, then IW$(n)$ simply corresponds to breadth-first search with duplicate detection, while values of $i$ lower than $n$ induce breadth-first searches with respectively more aggressive state pruning.

In classical planning, the primary termination condition for the search process is the achievement of the goal. In problems with a more general and/or unknown reward structure, such as the Atari games, the termination is determined by a search resource budget, such as a time window or a limit on the number of generated nodes. The accumulated reward $R(s)$ of a generated state $s$ is $R(s) = R(s') + r(a, s)$ where $s'$ is the unique parent state of $s$. The best path is then a state-space path from the initial state to a state that maximizes the accumulated reward. In online planning, the first action along this path is then executed, the system transitions to a new state, and the search process starts over from that new initial state. In what follows, the state that is planned for at the current iteration is denoted by $s_0$.

## 3 Prioritized Pruning

While the experiments of Lipovetzky et al. [2015] showed that IW$(1)$ performs on the Atari games at the level of UCT, a closer look at the results suggests that the strength of these two algorithms is somewhat complementary: Out of the 54 games used in the experiments, IW$(1)$ scored more than 150% of the UCT's score in 17 games, while UCT scored more than 150% of the IW$(1)$'s score in 13 games. The two algorithms are too different to easily characterize the problems on which each has an *a priori* advantage, and yet one key difference between them immediately suggests itself as a natural explanation of the complementarity of the two: While the exploration of UCT is biased towards the regions of the state-space that already appear rewarding, the exploration of IW$(1)$ has no such a bias whatsoever.

As a variant of blind search that aims at combining both strengths, Lipovetzky et al. [2015] evaluated 2BFS, a best-first search algorithm with two queues: one queue ordered in increasing order of the novelty penalty, and a second queue ordered in a decreasing order of the accumulated reward. In one iteration, the best first search picks up the best node from one queue, and in the second iteration it picks up the best node from the other queue, with all the generated nodes being added to both queues. This way, while IW$(i)$ progresses in a breadth-first manner while pruning states based on their novelty, 2BFS progresses (in its first queue) in the best-first manner while only prioritizing the states based on their novelty.

In that respect, an important property of most of the Atari games is that the state of the game changes not only due to the actions of the player but also due to the change of the environment—the cars keep moving, the rocks keep falling, etc.—while the changes that the player can make to the state are rather limited.[1] As a result, a state reachable in $k$ steps from the initial state is likely to be novel with respect to the states reachable in less than $k$ states. Thus, if the best-first search selects the state to expand based on its relative novelty, then at least some of its children are also likely to exhibit relatively high novelty, ending up high in the queue. Such a chain effect makes the search much more *depth-first*, and, since the state-space is typically much larger than what the search can explore within a reasonable search budget, the explored region of the state-space is likely to have a narrow focus. In turn, this biases action selection at $s_0$ towards action sequences that collect rewards far from $s_0$ while possibly missing alternatives that bring the rewards earlier as well. Interestingly, the second queue of 2BFS, prioritized by the states' reward-so-far, does not necessarily balance this phenomenon: Since the first rewarding state is more likely to be found within the tunnel created by the depth-first-like progress of the first queue, the second queue is likely to join expanding that tunnel, possibly making it wider, but still, abandoning the exploration of the state-space under the *myopically* less appealing alternatives.

At first view, the breadth-first searching IW$(1)$ is expected to behave differently. However, in our experiments we consistently observed IW$(1)$ exhibiting a very similar "single tunnel" phenomenon. This seems to happen precisely because of the aforementioned structure of the Atari games due to which *the likelihood of the states of the same shallowness to survive the novelty pruning decays very rapidly with the position of the states in the queue.* Indeed, in the experiments of Lipovet-

[1] This is very different from the typical structure of the benchmarks used in the classical planning research where the set of actions controlled by the planner is rather rich, but at the same time, these actions are responsible for most, if not all, the changes made to the state. At least in part, the latter can be attributed to the fact that encoding environment changes in the PDDL language is not an easy task.

zky et al. [2015], neither 2BFS exhibited a substantial advantage over IW$(1)$ nor the other way around, leaving open the quest for an effective interplay between the novelty and the reward-so-far promise.

We now show that plugging a bias towards the reward-so-far *into* the actual state pruning mechanism offers a promising direction for addressing this quest: Even if done in a rather simple manner as in the *prioritized IW*$(i)$ procedure described below, this approach results in an algorithm that strongly outperforms IW$(1)$ and UCT. Prioritized IW$(i)$, or p-IW$(i)$, for short, deviates from IW$(i)$ twofold:

1. While preserving the breadth-first dynamics of IW$(i)$, the ties in the queue are broken to favor states with higher accumulated reward.
2. Every $i$-set of atoms $\mathbf{x}$ is schematically pre-assigned a "reward" of $\widehat{r}(\mathbf{x}) = -\infty$. Given that, a generated state $s$ is considered novel if, for some $i$-set of atoms $\mathbf{x}$ in $s$, we have $R(s) > \widehat{r}(\mathbf{x})$. If that holds, then (and only then) $s$ is not pruned, and, for each $i$-set of atoms $\mathbf{x}$ in $s$, we update its reward to $\widehat{r}(\mathbf{x}) := \max\{R(s), \widehat{r}(\mathbf{x})\}$.

The two modifications of p-IW$(i)$ with respect to IW$(i)$ bring the reasoning about the reward accumulated by the states directly into the mechanism of state pruning, addressing two complementary questions—what states should lead the pruning, and what states should be pruned—as follows.

First, the regular breadth-first search is driven by two principles: always expand one of the shallowest states in the queue and never put a state into the queue twice. The later *duplicate pruning* makes BrFS a graph search rather than a tree search, leading to up to exponential savings in the search effort while preserving completeness. In that respect, IW$(i)$ is BrFS in which state duplication is *over-approximated* by state (non-)novelty: If a search node generated by BrFS is pruned due to its duplication, then, for any $i$, that search node would be pruned by IW$(i)$, but not necessarily vice versa (unless $i = n$). In classical planning, this "non-novel as duplicate" over-approximation has a strong semantics via a notion of "problem width" [Lipovetzky and Geffner, 2012]. In the settings of ALE, however, this over-approximation is motivated only informally, by a similarity to the novelty-based search methods developed independently in the context of genetic algorithms [Lehman and Stanley, 2011]. In the latter methods, individuals in the population are not ranked according to the optimization function but in terms of how much they are different from the rest of the population, thus encouraging global exploration rather than (greedy) search for local improvements.

Though better exploration is indeed what online planning effectiveness boils down to [Bubeck *et al.*, 2011], the direct linkage to the diversity-driven genetic algorithms has an important weakness. Suppose that the two shallowest states in the search queue of IW$(i)$ are $s_1$ and $s_2$, and suppose further that the children of $s_1$ make the children of $s_2$ non-novel and vice versa. In other words, expanding any of these two states blocks the search under the other state. In IW$(i)$, the choice between $s_1$ and $s_2$ remains arbitrary. However, if the accumulated reward of $s_1$ is higher than $s_2$, then, *ceteris paribus*, it is only reasonable to assume that the best extension of $s_1$ is more rewarding than the best extension of $s_2$, and thus $s_1$ should better be expanded before $s_2$. This example emphasizes the difference between the evolutionary search in genetic algorithms and state-space forward search: While the former typically examines fully specified candidates to the problem solution, the latter gradually expands *partial solutions* in the form of path prefixes. Under the additive structure of the accumulated reward, the quality of partial solutions lower bounds the quality of their extensions, making total ignorance of the accumulated reward of the states in the queue rather questionable. The first modification of p-IW$(i)$ with respect to IW$(i)$ approaches precisely this issue under the conservative, *ceteris paribus* semantics, preserving the breadth-first search dynamics of the search.

Suppose now that IW$(i)$ generates a state $s$ such that $R(s) > R(s')$ for all the previously generated states $s'$. Despite the fact that the extensions of the respective path to $s$ are now the most promising candidates for the best solution that IW$(i)$ can possibly compute from now on, if $s$ is evaluated as non-novel, then it is pruned, independently of its accumulated reward. The second deviation of prioritized IW$(i)$ from IW$(i)$ takes the accumulated reward of the generated state $s$ into account in the actual decision whether $s$ should be pruned or not. Specifically, a newly generated state $s$ in prioritized IW$(i)$ is pruned if, for every $i$-set of atoms $\mathbf{x}$ in $s$, there was a previously generated state $s'$ that contains $\mathbf{x}$ and has $R(s') \geq R(s)$.

Similarly to the way the state pruning in IW$(i)$ can be understood as an over-approximation of the standard duplicate pruning, the state pruning in p-IW$(i)$ can be understood as an *over-approximation of duplicate pruning with state reopening*. In BrFS, if the solution optimality is of interest, then, if a previously generated state $s$ is rediscovered through a different path with a higher accumulated reward, then $s$ is "reopened", either by re-starting the search from $s$ onwards, or by propagating the new accumulated reward of $s$ to its descendants in the queue. In that respect, if a state generated by BrFS with node reopening is pruned, then, for any $i$, that search node would be pruned by p-IW$(i)$, but not necessarily vice versa. In particular, this modification allows for a substantial alleviation of the "single tunnel" phenomenon exhibited by IW$(i)$, keeping the search wider but only when the extended search breadth is justified by the accumulated reward of the respective states.

We tested p-IW$(1)$ and IW$(1)$ on 53 of the 55 different games considered by Bellemare et al. [2013]: The SKIING game was already left out in the experiments of Lipovetzky et al. [2015] due to certain issues with the reward structure of this game. We decided to also leave out BOXING because, in the single player setting of ALE, scoring in this game boils down to striking in arbitrary directions since the second player is doing nothing, and therefore every algorithm will trivially score the possible maximum.

We used the implementation of IW$(1)$ by Lipovetzky et al. [2015], and have implemented p-IW$(1)$ on top of it. To focus the comparison on the effectiveness of individual online decisions, both algorithms have been evaluated in a memoryless setting. This is in contrast to the experiments of Lipovetzky et al. [2015] in which IW$(1)$ reused the frames in the

sub-tree of the previous lookahead that is rooted in the selected child, to allow for a direct comparison with the results reported for UCT by Bellemare et al. [2013] under a similar setting. Following Lipovetzky et al. [2015], each action selection decision was given a lookahead budget of 150000 simulated frames (or, equivalently, 30000 search nodes), the lookahead depth was limited to 1500 frames, and the accumulated rewards were discounted as $R(s') = R(s) + \gamma^{d(s)+1} r(s,a)$ where $s$ is the unique parent of $s'$, $a$ is the respective action, and the discount factor[2] was set to $\gamma = 0.995$. To reduce the variance, each game was played 30 times, with the reported results being averaged across these runs.

Columns 2-3 in Table 1 shows that p-IW$(1)$ rather consistently outperforms IW$(1)$. Out of the 53 games, p-IW$(1)$ achieved higher average scores in 42 games, 5 games ended up with a draw, and IW$(1)$ achieved higher average scores in 6 games. Of the latter, the highest achievement of IW$(1)$ was the 22% score difference in GOPHER, while p-IW$(1)$ outscored IW$(1)$ by more than 50% on 25 games. In fact, comparing our results with the results reported by Lipovetzky et al. [2015] for IW$(1)$ *with* memory, p-IW$(1)$ without memory scored higher than IW$(1)$ with memory on 14 games.

In general, we have noticed that both p-IW$(1)$ and IW$(1)$ typically did not use the entire budget of 150000 simulated frames per decision. To examine the score improvement as a function of budget, we have also tested them under a budget of only 10000 simulated frames per decision, all else being equal. The results are shown in columns 4-5 of Table 1. As one would expect, the scores here are typically lower than these achieved under the 150000 frames budget, and, since p-IW$(1)$ brings an approximation of state reopening, typically it benefits of the budget increase much more than IW$(1)$. More interestingly, while the higher budget "misled" p-IW$(1)$ on 6 games, in none of these cases the score loss was substantial. In contrast, IW$(1)$ did worse with 150000 frames budget than with 10000 frames budget on 13 games, with the loss being substantial on 6 games, namely ATLANTIS, CENTIPEDE, GOPHER, NAME THIS GAME, POOYAN, and TIME PILOT.

To allow a direct comparison with the results reported by Lipovetzky et al. [2015] and Bellemare et al. [2013] we have also implemented a "memorizing" version of p-IW$(1)$ in which the frames in the lookahead sub-tree rooted in the selected child are reused and no calls to the emulator are made for the transitions that are cached in that sub-tree. In addition, in IW$(1)$ and p-IW$(1)$, the reused states have been ignored in the computation of novelty of the new states so that more states could escape the pruning. The results of the comparison of p-IW$(1)$ with IW$(1)$, 2BFS, UCT, and BrFS are depicted in Table 4. Similarly to the setup behind Table 1, the maximum episode duration was set to 18000 frames and every algorithm was limited to a lookahead budget of 150,000 simulated frames. The results for UCT and BrFS in Table 4 are from Bellemare et al. [2013] and the results for 2BFS are from Lipovetzky *et al.* [2015].

[2]The discount factor results in a preference for rewards that can be reached earlier, which is a reasonable heuristic given the budget limits of the lookahead search. At the same time, choosing the right discount factor is a matter of tuning.

| | 150K | | | 10K | |
|---|---|---|---|---|---|
| Game | p-IW(1) | IW(1) | R. p-IW(1) | p-IW(1) | IW(1) |
| ALIEN | 4939 | 4705 | **4995** | **1638** | 1473 |
| AMIDAR | **1186** | 938 | 911 | 67 | **78** |
| ASSAULT | **1700** | 591 | 572 | **423** | 373 |
| ASTERIX | **172413** | 30780 | 83983 | 5985 | **6683** |
| ASTEROIDS | **63520** | 29884 | 7780 | 2192 | **2224** |
| ATLANTIS | **151720** | 52453 | 133943 | **144850** | 126703 |
| BANK HEIST | 296 | 296 | **303** | **67** | 63 |
| BATTLE ZONE | 7767 | 5000 | **11600** | **2900** | 2133 |
| BEAM RIDER | **4487** | 3398 | 4127 | 2445 | **2730** |
| BERZERK | **854** | 639 | 496 | **208** | 200 |
| BOWLING | 27 | 27 | **36** | **28** | 27 |
| BREAKOUT | 291 | 224 | **455** | **400** | 344 |
| CARNIVAL | 2773 | 2509 | **4270** | **2141** | 1832 |
| CENTIPEDE | **163917** | 59913 | 72441 | **140171** | 134542 |
| CHOPPER COMMAND | **5653** | 2040 | 3433 | **2230** | 2157 |
| CRAZY CLIMBER | **107673** | 37350 | 57693 | **114157** | 37013 |
| DEMON ATTACK | **24153** | 12448 | 13927 | 4845 | **6098** |
| DOUBLE DUNK | -6 | -6 | **-5** | **-14** | -18 |
| ELEVATOR ACTION | **8910** | 4217 | 7010 | **2597** | 2057 |
| ENDURO | 420 | **432** | 218 | **0** | **0** |
| FISHING DERBY | -8 | -9 | **0** | **-82** | -83 |
| FREEWAY | 30 | 30 | **32** | **23** | **23** |
| FROSTBITE | **353** | 199 | 238 | 257 | **259** |
| GOPHER | 9756 | 11852 | **16707** | 9546 | **15019** |
| GRAVITAR | **2943** | 2270 | 1423 | **343** | 315 |
| HERO | 4969 | **5483** | 2922 | 2159 | **2170** |
| ICE HOCKEY | **43** | 41 | 17 | **-7** | **-7** |
| JAMESBOND | 173 | 152 | **183** | **32** | **32** |
| JOURNEY ESCAPE | 7973 | **8560** | 3303 | 440 | **1293** |
| KANGAROO | 1057 | 1130 | **3323** | 587 | **753** |
| KRULL | **10293** | 4332 | 5692 | **4464** | 3475 |
| KUNG FU MASTER | **67163** | 33903 | 31050 | **26610** | 26250 |
| MONTEZUMA REVENGE | 0 | 0 | **17** | **0** | **0** |
| MS PACMAN | **11451** | 8219 | 8861 | 3511 | **3835** |
| NAME THIS GAME | **11302** | 6087 | 7957 | **12445** | 11004 |
| PONG | **14** | 13 | 12 | **-20** | **-20** |
| POOYAN | 2252 | 1312 | **11116** | 1945 | **2271** |
| PRIVATE EYE | **72** | 0 | -1 | **93** | 20 |
| QBERT | 1640 | 1249 | **8838** | 1441 | **1527** |
| RIVERRAID | **8707** | 4055 | 3880 | **3303** | 3095 |
| ROAD RUNNER | **80900** | 39133 | 40547 | **0** | **0** |
| ROBOTANK | **59** | 57 | 26 | **3** | 2 |
| SEAQUEST | **19007** | 2747 | 1112 | 245 | **260** |
| SPACE INVADERS | **2037** | 1151 | 1365 | **227** | 211 |
| STAR GUNNER | **14193** | 2783 | 1293 | 1097 | **1190** |
| TENNIS | **10** | 9 | **10** | **-24** | **-24** |
| TIME PILOT | **31767** | 5903 | 6643 | **18797** | 15140 |
| TUTANKHAM | 136 | 140 | **144** | **182** | 147 |
| UP N DOWN | **93305** | 75088 | 34605 | **2717** | 2576 |
| VENTURE | **240** | 150 | 53 | **0** | **0** |
| VIDEO PINBALL | **413976** | 223772 | 202279 | **286921** | 237078 |
| WIZARD OF WOR | **111487** | 88953 | 70257 | **6373** | 4270 |
| ZAXXON | **15247** | 9200 | 2607 | **0** | **0** |
| # times best (53 games) | **34** | 3 | 16 | **38** | 24 |

Table 1: Performance of different algorithms in 53 Atari 2600 games. Columns 2-3 and 5-6 compare between the performance of p-IW$(1)$ and IW$(1)$ with the lookahead per decision being limited to 150000 simulated frames in columns 2-4 and to 10000 simulated frames in columns 5-6. The algorithms are evaluated over 30 runs for each game. The maximum episode duration is 18000 frames. Column 4 adds the *racing* p-IW$(1)$ described in Section 4. Per lookahead budget, the average scores in bold show best performer and the summary of the performance is given at the bottom of the table.

Table 2 shows that p-IW$(1)$ rather consistently outperforms all the other evaluated algorithms. Out of the 53 games, p-IW$(1)$ achieved higher average scores in 34 games, tying up on BOWLING with IW$(1)$, on TENNIS with IW$(1)$ and 2BFS, and on PONG with all the algorithms except for BrFS. On the other hand, IW$(1)$ was best in only 4 games, and 2BFS and UCT in 6. In general, IW$(1)$ and p-IW$(1)$ with memory mostly outperform IW$(1)$ and p-IW$(1)$ without memorization, with the exceptions being the PRIVATE EYE and STAR GUNNER games. This may happen due to the inheri-

| Game | IW(1) | p-IW(1) | 2BFS | UCT | BrFS |
|---|---|---|---|---|---|
| ALIEN | 28238 | **38951** | 12252 | 7785 | 784 |
| AMIDAR | 1775 | **3122** | 1090 | 180 | 5 |
| ASSAULT | 896 | **1970** | 827 | 1512 | 414 |
| ASTERIX | 145067 | **319667** | 77200 | 290700 | 2136 |
| ASTEROIDS | 52170 | **68345** | 22168 | 4661 | 3127 |
| ATLANTIS | 150327 | **198510** | 154180 | 193858 | 30460 |
| BANK HEIST | 601 | **1171** | 362 | 498 | 22 |
| BATTLE ZONE | 7667 | 9433 | **330880** | 70333 | 6313 |
| BEAM RIDER | 9851 | **12243** | 9298 | 6625 | 694 |
| BERZERK | **1915** | 1212 | 802 | 554 | 195 |
| BOWLING | **69** | **69** | 50 | 25 | 26 |
| BREAKOUT | 401 | 477 | **772** | 364 | 1 |
| CARNIVAL | 5898 | **6251** | 5516 | 5132 | 950 |
| CENTIPEDE | 98922 | **193799** | 94236 | 110422 | 125123 |
| CHOPPER COMMAND | 12310 | **34097** | 27220 | 34019 | 1827 |
| CRAZY CLIMBER | 36010 | **141840** | 36940 | 98172 | 37110 |
| DEMON ATTACK | 20177 | **34405** | 16025 | 28159 | 443 |
| DOUBLE DUNK | 0 | 8 | 21 | **24** | -19 |
| ELEVATOR ACTION | 13097 | 16687 | 10820 | **18100** | 730 |
| ENDURO | **499** | 497 | 359 | 286 | 1 |
| FISHING DERBY | 22 | **42** | 6 | 38 | -92 |
| FREEWAY | 31 | **32** | 23 | 0 | 0 |
| FROSTBITE | 2040 | **6427** | 2672 | 271 | 137 |
| GOPHER | 18175 | **26297** | 15808 | 20560 | 1019 |
| GRAVITAR | 4517 | **6520** | 5980 | 2850 | 395 |
| HERO | 12769 | **15280** | 11524 | 1860 | 1324 |
| ICE HOCKEY | 55 | **62** | 49 | 39 | -9 |
| JAMESBOND | **20668** | 15822 | 10080 | 330 | 25 |
| JOURNEY ESCAPE | 42263 | **65100** | 40600 | 12860 | 1327 |
| KANGAROO | **8243** | 5507 | 5320 | 1990 | 90 |
| KRULL | 6357 | **15788** | 4884 | 5037 | 3089 |
| KUNG FU MASTER | 63570 | **86290** | 42180 | 48855 | 12127 |
| MONTEZUMA REVENGE | 13 | 27 | **500** | 0 | 0 |
| MS PACMAN | 22869 | **30785** | 18927 | 22336 | 1709 |
| NAME THIS GAME | 9244 | 14118 | 8304 | **15410** | 5699 |
| PONG | **21** | **21** | **21** | **21** | -21 |
| POOYAN | 10460 | 15832 | 10760 | **17763** | 910 |
| PRIVATE EYE | -60 | 21 | **2544** | 100 | 58 |
| QBERT | 5139 | **44876** | 11680 | 17343 | 133 |
| RIVERRAID | 6865 | 14437 | 8304 | **15410** | 2179 |
| ROAD RUNNER | 85677 | **120923** | 68500 | 3875 | 245 |
| ROBOTANK | 67 | **75** | 52 | 50 | 2 |
| SEAQUEST | 13972 | **35009** | 6138 | 5132 | 288 |
| SPACE INVADERS | 2812 | 3076 | **3974** | 2718 | 112 |
| STAR GUNNER | 1603 | 1753 | **4660** | 1207 | 1345 |
| TENNIS | **24** | **24** | **24** | 3 | -24 |
| TIME PILOT | 35810 | **65213** | 36180 | 63855 | 4064 |
| TUTANKHAM | 167 | 158 | 204 | **226** | 64 |
| UP N DOWN | 104847 | **120200** | 54820 | 74474 | 746 |
| VENTURE | 1107 | **1167** | 980 | 0 | 0 |
| VIDEO PINBALL | 288394 | **471859** | 62075 | 254748 | 55567 |
| WIZARD OF WOR | 122020 | **161640** | 81500 | 105500 | 3309 |
| ZAXXON | 33840 | **39687** | 15680 | 22610 | 0 |
| # times best (53 games) | 7 | **37** | 8 | 7 | 0 |

Table 2: Performance of p-IW(1) vs. IW(1), 2BFS, UCT, and BrFS. The experimental setup is similar to this of Lipovetzky et al., with the lookahead budget of 150000 frames.

tance of the search tree without updating the novelty table: Both IW(1) and p-IW(1) with memory start with a "broad" tree from the previous step, and the expansion of all leaves in these trees limits the in-depth exploration of the search. Another exception is the CRAZY CLIMBER game, where IW(1) with memory mostly outperform IW(1) without it. Here the case is slightly different because IW(1) without memory and a budget of 10000 frames outperforms the same algorithm with a budget of 150000 frames. We believe that this phenomenon in CRAZY CLIMBER is due to the density of the rewards in this game. Since IW(1) is guided only by the novelty measure, and not by the reward collected so far, its action recommendation often results in a choice not of a "better" action but of the action behind the longest rollout.

## 4 Racing Blind Search

Approximating state duplication by state non-novelty allows IW($i$) to search deeper in the state-space, possibly reaching rewarding states that lie far from the initial state. At the same time, this specific approximation often results in a highly unbalanced exploration of the state space. p-IW($i$) partly alleviates the latter phenomenon, but the extent to which this is achieved depends on the reward structure of the specific game. Recall that the original objective pursued by the, both heuristically guided and blind, best-first forward search procedures is to compute a sequence of actions from the initial state to a state that maximizes the accumulated reward, even if not in absolute terms but only best effort. In the context of this objective, it is actually hard to argue whether a more balanced exploration of the state space is more rational than a less balanced exploration, and if so, what kind of balance we should strive for here. In fact, in the absence of any extra knowledge about the problem, expanding an already rewarding sequence of actions is arguably more rational than searching elsewhere.

In the context of online planning, however, computing an as rewarding as possible sequence of actions is *not* the actual objective of the planner. Let $A(s_0) = \{a_1, \ldots, a_k\}$ be the actions applicable at the current state $s_0$ and, for $1 \leq l \leq k$, let $\pi_l$ be the most rewarding action sequence applicable in $s_0$ that starts with $a_l$. The actual objective in online planning is not to find the most rewarding action sequence $\pi_{l^*}$ among $\pi_1, \ldots, \pi_k$ but only to find the index $l^*$ of that sequence, that is, to find the identity of the first action along $\pi_{l^*}$. At least in theory, the latter objective is less ambitious than the former since computing $\pi_{l^*}$ implies finding $l^*$ but obviously not the other way around. In practice, this difference in objectives suggests that various adaptations can be found beneficial when transferring the techniques from the classical, open-loop AI planning to the closed-loop online planning.

To exemplify the prospects of such adaptation, consider the following example. Whether we apply prioritized or regular IW($i$) (or, for that matter, any other blind forward search procedure), suppose that, at a certain stage of the search process, the states in the queue all happen to be descendants of the same action $a_l$ applicable at the planned state $s_0$. If one of these states has the maximum accumulated reward among all the states generated so far, then the search can be terminated right away: No matter how much further we will continue searching, $a_l$ will remain the action of our choice at $s_0$, that is, $l$ will remain our estimate for the desired action index $l^*$. Furthermore, let $Q_1, \ldots, Q_k$ be a cover of the search queue $Q$ of either prioritized or regular IW($i$), such that, for $1 \leq l \leq k$, $s \in Q_l$ if one of the most rewarding action sequences generated so far from $s_0$ to $s$ starts with $a_l$. Given that, the candidates for $a_{l^*}$ can be restricted to a subset $A$ of $A(s_0)$ if $\{Q_l \mid a_l \in A\}$ induces a set cover of $Q$.

In sum, the search procedures in the context of online planning should aim at the *competition* between the actions in $A(s_0)$. UCT and BrFS actually appear to be more faithful with this objective than both IW($i$) and p-IW($i$) yet not without caveats. The UCT algorithm is grounded in the UCB (upper-confidence bound) Monte-Carlo procedure for optimizing online action selection in multiarmed bandits (MABs) [Auer *et al.*, 2002]. However, UCT has at least two substantial shortcomings in the settings of online planning for the Atari games: First, while the UCB procedure is opti-

mized for the learning-while-acting settings of MAB, the simple uniform and round-robin sampling of the actions provide much better formal and empirical guarantees when it comes to online action selection with MAB simulators [Bubeck *et al.*, 2011]. Thus, within the scope of the Monte-Carlo tree search algorithms, the more balanced sampling algorithms such as BRUE [Feldman and Domshlak, 2014] are *a priori* more appropriate. Second, the usage of the upper-confidence bounds and of the very averaging Monte-Carlo updates in UCT aims at estimating the mean value of the policies under stochasticity of the action outcomes. Since all the actions in the Atari games are deterministic, neither of these tools is semantically meaningful here and actually harm the convergence of the decision process. In contrast, BrFS is built for deterministic actions and by definition runs a fair competition between the actions in $A(s_0)$ in terms of the search horizon. The absence of selectiveness, however, makes BrFS uncompetitive in problems with large search width such as the Atari games where all the actions are applicable in every state.

Combining the selectiveness of p-IW$(i)$ with a uniformly balanced exploration of the actions, we have evaluated the following simple modification of p-IW$(1)$, referred to in what follows as *racing p-IW*$(1)$:

1. Expand the initial state $s_0$. For every subset of actions $A \subseteq A(s_0)$ that result in the same successor of $s_0$, eliminate from $A(s_0)$ all but one action $a \in A$ that maximizes $r(s_0, a)$.
2. Let the (pre-pruned as above) action set $A(s_0)$ be $\{a_1, \ldots, a_k\}$. At iteration $m$, restrict the selection from the search queue only to successors of $f(s_0, a_{m\%k})$.
3. At every stage of the search, if the search queue contains only successors of $f(s_0, a)$ for some action $a \in A(s_0)$ and a state in the queue corresponds to the most rewarding path generated so far, terminate the search and recommend (aka execute) $a$.

Note that nothing in the above modification is specific to p-IW$(1)$, and thus one can adapt any forward state-space search algorithm to the action selection objective of online planning in exactly the same manner.

Column 4 in Table 1 compares the performance of racing p-IW$(1)$ with that of p-IW$(1)$ and IW$(1)$. The experimental setup remains as before, with the lookahead budget of 150000 frames. While p-IW$(1)$ was still the leader with best performance in 34 games, racing p-IW$(1)$ achieved the best average scores in 16 games, including some very substantial leads such as in BATTLE ZONE, GOPHER, POOYAN, and QBERT. A closer look suggests a relative advantage of racing p-IW$(1)$ in games like POOYAN, QBERT, KANGAROO, CARNIVAL and BREAKOUT. In these games, the prospective rewards for many states are quite distant, and thus the dynamics of p-IW$(1)$ on them is no different from this of IW$(1)$, with too many states being pruned due to the absence of novelty. In contrast, the more balanced exploration of racing p-IW$(1)$ alleviates such an over-pruning in these games, keeping more leads open until the distant rewards are being revealed. On the other hand, this balanced exploration leads to a "waste" of too many simulations in games with more densely distributed rewards. Hence, if one can estimate well the density of the rewards earlier in the planning, then it will be possible to select the right degree of exploration balance online, leading to the best of both worlds.

We also note that while racing p-IW$(1)$ was the only algorithm to score in the very challenging game MONTEZUMA REVENGE, at the moment we have no evidence that this should be attributed to anything but pure chance. At the same time, the seemingly small advantage of racing p-IW$(1)$ over p-IW$(1)$ and IW$(1)$ in FREEWAY is actually substantial since the maximum score in this game is 38, and this canonical Atari game posed a challenge to both BrFS and UCT, with IW$(1)$ being the first algorithm to score in this game at all [Lipovetzky *et al.*, 2015].

## 5 Summary and Future Work

Online planning with simulators provide a challenging testbed for action planning since most of the sophisticated techniques for scaling up planning systems rely upon inference over propositional encodings of actions and goals that are "hidden" by the simulator. Previous work showed that a blind forward search algorithm IW$(1)$ achieves state-of-the-art performance in such planning problems around the Atari video games, with the key to success being structural, similarity-based approximation of duplicate pruning [Lipovetzky *et al.*, 2015].

We have shown that the effectiveness of blind state-space search on deterministic online planning like in the Atari games can be further improved by (a) combining approximated duplicate pruning with an approximate state reopening, and (b) reshaping the dynamics of the forward search algorithms to better fit the objective of selecting and executing only a single action at a time. Our experiments show that modifying IW$(i)$ along these two lines results in algorithms, p-IW$(1)$ and racing p-IW$(1)$, that both substantially outperform IW$(1)$ on the Atari games.

The simple concept of the racing search algorithms for deterministic online planning suggests numerous directions for future investigation. First, while the state pruning in our racing p-IW$(i)$ was done based on a global view on state novelty, whether/when this globality is friend or foe is yet to be investigated: On the one hand, it is not hard to verify that the globally reasoned duplicate pruning will always improve the efficiency of the racing search at least as much as any locally reasoned duplicate pruning. On the other hand, pruning a state in one branch based on its similarity (but not equivalence!) to a state in another branch is not necessarily the best thing to do. As another issue, if the number of actions examined for the current state does not decrease for a substantial chunk of the lookahead budget, then it seems natural to consider a mechanism for gradual "candidate rejection", possibly in the spirit of algorithms for budgeted pure exploration in stochastic multiarmed bandit problems like Sequential Halving [Karnin *et al.*, 2013].


### Acknowledgments

This work was supported by the Israel Science Foundation (ISF) grant 1045/12 and by the Technion-Microsoft Electronic-Commerce Research Center.

## References


[Auer *et al.*, 2002] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2-3):235–256, 2002.

[Bellemare *et al.*, 2013] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. The Arcade Learning Environment: An evaluation platform for general agents. *JAIR*, 47:253–279, 2013.

[Bubeck *et al.*, 2011] S. Bubeck, R. Munos, and G. Stoltz. Pure exploration in finitely-armed and continuous-armed bandits. *Theor. Comp. Sci.*, 412(19):1832–1852, 2011.

[Feldman and Domshlak, 2014] Z. Feldman and C. Domshlak. Simple regret optimization in online planning for Markov decision processes. *JAIR*, 51:165–205, 2014.

[Geffner and Bonet, 2013] H. Geffner and B. Bonet. *A Concise Introduction to Models and Methods for Automated Planning*. Morgan & Claypool, 2013.

[Ghallab *et al.*, 2004] M. Ghallab, D. Nau, and P. Traverso. *Automated Planning*. Morgan Kaufmann, 2004.

[Karnin *et al.*, 2013] Z. S. Karnin, T. Koren, and O. Somekh. Almost optimal exploration in multi-armed bandits. In *ICML*, pages 1238–1246, 2013.

[Kocsis and Szepesvári, 2006] L. Kocsis and C. Szepesvári. Bandit based Monte-Carlo planning. In *ECML*, pages 282–293, 2006.

[Lehman and Stanley, 2011] J. Lehman and K. O. Stanley. Abandoning objectives: Evolution through the search for novelty alone. *Evol. Comp.*, 19(2):189–223, 2011.

[Lipovetzky and Geffner, 2012] N. Lipovetzky and H. Geffner. Width and serialization of classical planning problems. pages 540–545, 2012.

[Lipovetzky *et al.*, 2015] N. Lipovetzky, M. Ramírez, and H. Geffner. Classical planning with simulators: Results on the Atari video games. In *IJCAI*, pages 1610–1616, 2015.