Artificial Intelligence 225 (2015) 1–23

Contents lists available at ScienceDirect

Artificial Intelligence

www.elsevier.com/locate/artint

Incorporating weights into real-time heuristic search

Nicolás Rivera [a,*], Jorge A. Baier [b], Carlos Hernández [c]

[a] Department of Informatics, King's College London, London WC2R 2LS, UK
[b] Departamento de Ciencia de la Computación, Pontificia Universidad Católica de Chile, Vicuña Mackenna 4860, Santiago, Chile
[c] Departamento de Ingeniería Informática, Universidad Católica de la Santísima Concepción, Caupolicán 491, Concepción, Chile

ARTICLE INFO

Article history:
Received 14 August 2013
Received in revised form 5 March 2015
Accepted 25 March 2015
Available online 30 March 2015

Keywords:
Real-time heuristic search
A*
LRTA*
Weighted A*
Search in games

ABSTRACT

Multiplying the heuristic function by a weight greater than one is a well-known technique in heuristic search. When this technique is applied to A* with an admissible heuristic it yields substantial runtime savings, at the expense of sacrificing solution optimality. Its applicability to real-time heuristic search, a search approach that builds upon heuristic search, however, has only been explored by a few studies. In this article we present two new approaches to using weights in real-time heuristic search, applicable to a wide range of algorithms. The first one, *weighted lookahead*, is a variant of an existing approach by Shimbo and Ishida, and utilizes the weight while the algorithm performs lookahead search. The second one, *weighted update*, incorporates the weight to the edges of the search graph during the *learning* phase. We implemented both techniques within LSS-LRTA* and evaluated them in path-planning benchmarks. We show that weighted lookahead outperforms an existing approach by Shimbo and Ishida but that it does not improve over existing approaches that do not use weights. Weighted update, on the other hand, yields performance improvements of up to one order of magnitude both in solution cost and total search time. To illustrate further the generality of weighted update, we incorporate the technique in two other well-known real-time heuristic search algorithms: LRTA*-LS and daLSS-LRTA*, and we empirically show significant improvements for LRTA*-LS and modest but still important improvements for daLSS-LRTA*. We analyze the properties of weighted update in depth, showing, among other things, that it guarantees termination. Convergence behavior of LSS-LRTA*, modified to use weighted update, is also analyzed. In such a setting, we prove solutions are w-optimal, and provide additional bounds on solution quality that in practice are tighter than w-optimality.

1. Introduction

Weighted A* [1] is a well-known search algorithm for solving single-agent, deterministic search problems. It is based on A* [2] and uses an evaluation function $f(s) = g(s) + wh(s)$ to rank a state s in the search frontier, where $g(s)$ represents the cost incurred to reach s, $h(s)$ is a (heuristic) estimate of the true cost to reach a solution from s, and the *weight* w is a real value greater than or equal to one. It can find a solution substantially faster than A* as the weight is increased over one. However, the cost of returned solutions may increase as w is increased. If the heuristic h is admissible then the cost of found solutions can be at most a factor w away from optimal.

* Corresponding author.
E-mail addresses: nicolas.rivera@kcl.ac.uk (N. Rivera), jabaier@ing.puc.cl (J.A. Baier), chernan@ucsc.cl (C. Hernández).

http://dx.doi.org/10.1016/j.artint.2015.03.008
0004-3702/© 2015 Elsevier B.V. All rights reserved.

Weighting the heuristic is a simple but powerful technique that is widely used in state-of-the-art heuristic search algorithms. For example, it is key to the performance of both ARA* [3], an algorithm used in outdoor rover applications, and RWA* [4], the search engine underlying LAMA 2011 [5]—among the best-performing satisficing automated planners.

Real-time heuristic search [6] is an approach to solving search problems under tight computational time constraints, with applications ranging from video games to highly dynamic robotics. It builds on heuristic search; in fact, a heuristic function h is used to guide search. In brief, a real-time heuristic search algorithm alternates a *lookahead search* guided by h, that allows deciding which move to perform next, followed by an *update phase*, which makes the h-values of a bounded number of states more informed. The update phase—also called *learning phase*—allows the algorithm to terminate and is key to the algorithm's performance [7,8].

Existing approaches to using weights within real-time heuristic search have shown improved convergence performance[1] [9,10] but have not shown significant improvements at finding one (first) solution. For example, Shimbo and Ishida [9] propose a two-step approach which first multiplies the user-given heuristic h by w at the outset of search, and then runs a standard real-time search algorithm with the resulting heuristic. Their approach can be viewed as a straightforward adaptation of Weighted A*'s technique. Their empirical results on grid problems—which we confirm in this paper—show that by using a weight greater than one, the algorithm slows down and returns worse first solutions. Bulitko's γ-Trap algorithm [11]—later generalized within the LRTS framework in [10]—uses lower-than-one weights applied to the *costs* of the graph. In this way, γ-Trap indirectly gives more importance to the heuristic h in its evaluation function. The only evaluation of γ-Trap for the first solution that we are aware of (Figure 5 in [11]) on Korf's 15-puzzels shows mixed results. While modest improvements are indeed observed for some weight configurations, in the remaining configurations, weights seem to be detrimental.

The main motivation underlying the work presented in this paper was to develop a technique inspired upon Weighted A*'s technique to obtain improved performance at finding one solution given a search problem. Our first attempt was yet another straightforward adaptation of the Weighted A* approach. This approach, which we call *weighted lookahead*, is directly applicable to any real-time search algorithm that uses A* as a lookahead search module, and consists of replacing the A* search by a Weighted A* search. As we explain in detail later, weighted lookahead is closely related to Shimbo and Ishida's approach but is not equivalent and, in fact, outperforms Shimbo and Ishida's. Unfortunately, we show that it does not yield improved performance over real-time heuristic search algorithms that do not use weights.

We explain the failure of straightforward approaches to incorporating weights successfully, like weighted lookahead, by the following fact: total runtime of a real-time search algorithm is generally proportional to solution quality. This is due to the fact that real-time search algorithms, when spending a comparable amount of time in each lookahead search, will run faster if and only if they perform fewer searches, and this will usually happen if and only if the solution returned is shorter. Incorporating weights in real-time search algorithms *à la* Weighted A* leads to increased solution cost and consequently to increased runtime.

This phenomenon leads us to the development of a technique that uses weights but in a very different way. The technique, which we call *weighted update*, modifies the learning phase of the algorithm rather than the lookahead phase. In a nutshell, weighted update is like a regular update but the costs of the arcs in the search graph are multiplied by a weight w greater than or equal to one.

A consequence of our approach used with $w > 1$ is that the algorithm will not be inclined to revisit states because each time a state is visited its h-value is increased by a larger amount than it would were w equal to one. By avoiding revisiting states the algorithm indeed is not bound to a performance glitch observed a number of years ago by Ishida [12], and recently analyzed formally by Sturtevant and Bulitko [13]: standard real-time search algorithms tend to become "trapped" in areas in which *heuristic depressions* exist, leading to poor performance. "Trapped" here simply means that the algorithm will revisit a number of states, several times. While recent research (e.g., [14]) has shown that explicitly avoiding depressions actually leads to improved performance, the approach we present here is different since it does not provide or necessitate any mechanism for identifying or avoiding such depressions. Rather, avoidance of depressions is a natural consequence of the mechanism used to update the heuristic.

As mentioned above, the idea of multiplying arc costs by a weight has been described before by [11] and Bulitko and Lee [10]. Our approach, like theirs, changes the way h is learned by multiplying the arc costs rather than the heuristic by the weight, but, unlike theirs, our weight is greater than or equal to one and it is used only while learning, *not* for lookahead search. Furthermore, unlike them, we show below that our approach yields significant and consistent improved performance for the first solution.

Both weighted update and weighted lookahead are techniques that are extremely simple to implement in standard real-time heuristic search algorithms. In this paper we implement them on top of the state-of-the-art algorithm LSS-LRTA* [7], producing two new algorithms. By applying weighted update to LSS-LRTA* we obtain wLSS-LRTA*, whereas by applying weighted lookahead we obtain LSS-LRTwA*. We evaluate the algorithms over standard videogame path-finding tasks. As mentioned above, we show that the weighted lookahead yields both worse solutions and worse running times as w increases. On the other hand, we show that weighted update does lead to improved performance, both in solution quality

[1] Convergence performance is evaluated by running multiple search trials, each of which solves the same given search problem, without resetting h between trials, and until h has converged.