# Anytime AND/OR Best-First Search for Optimization in Graphical Models

**Natalia Flerova** NFLEROVA@UCI.EDU
University of California Irvine

**Radu Marinescu** RADU.MARINESCU@IE.IBM.COM
IBM Research, Dublin, Ireland

**Rina Dechter** DECHTER@UCI.EDU
University of California Irvine

## Abstract

Depth-first search schemes are known to be more cost-effective for solving graphical models tasks than Best-First Search schemes. In this paper we show however that *anytime Best-First* algorithms recently developed for path-finding problems, can fare well when applied to graphical models. Specifically, we augment best-first schemes designed for graphical models with such anytime capabilities and demonstrate their potential when compared against one of the most competitive depth-first branch and bound scheme. Though Best-First search using weighted heuristics is successfully used in many domains, the crucial question of weight parameter choice has not been systematically studied and presents an interesting machine learning problem.

## 1. Introduction

In recent years we have seen a variety extensions of best-first search algorithms into flexible anytime scheme making these algorithms more practical approximate algorithms. The most popular algorithms are based on *Weighted A** (WA*) (Pohl, 1970). The idea is to inflate the heuristic values by a constant factor of $w \geq 1$, making the heuristic inadmissible to a degree controlled by $w$, but still guaranteeing a solution cost within a factor of $w$ from the optimal one while typically yielding faster search. If the approximate solution is found quickly and additional time is available, the search for a better solution with smaller weight resumes. Several anytime weighted best-first search schemes were investigated in the past decade (Hansen and Zhou, 2007; Likhachev, Gordon, and Thrun, 2003; van den Berg et al., 2011; Richter, Thayer, and Ruml, 2010) and shown to be quite effective.

All these developments were conducted primarily in the context of path-finding problems (e.g., planning problems) for which best-first strategies are the method of choice. In contrast, for combinatorial optimization over *graphical models*, such as MAP/MPE or WCSP, depth-first branch and bound are the preferred schemes. These algorithms were extensively studied for finding both exact and approximate solutions (Kask and Dechter, 2001; Marinescu and Dechter, 2009b; Otten and Dechter, 2011; de Givry et al., 2005). Best-first search algorithms, though known to be more time efficient (Dechter and Pearl, 1985), are rarely considered due to their inherently large memory requirements and lack of anytime behavior. Moreover, since in graphical models all solutions have the same length, the main benefit of best-first search of avoiding the exploration of unbounded paths, becomes irrelevant.

Some recent studies (Marinescu and Dechter, 2009b) showed that if given sufficient memory, best-first can be superior to depth-first search, while otherwise, it will often fail. Therefore, the recent extensions of Best-

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

first search into anytime methods raise the hope that the algorithm's qualities can be brought to bear, leading to this investigation in their potential as anytime search schemes for graphical models as well.

**Contributions.** We extended several variants of weighted A* into graphical models by applying the ideas to the AOBF class of algorithms (Marinescu and Dechter, 2009b). The simplest extension we consider, denoted wAOBF, runs AOBF with the weighted heuristic iteratively, decreasing the weight at each iteration, until either $w = 1$ or until a time bound is reached. Additionally we extend Anytime Repairing A* (ARA*) (Likhachev, Gordon, and Thrun, 2003) and Anytime AO* (Bonet and Geffner, 2012), yielding wR-AOBF and p-AOBF respectively. We investigated empirically the effectiveness of each of the schemes and its guiding parameters as an anytime scheme and compared the emerging schemes against Breadth-Rotating AND/OR Branch-and-Bound (BRAOBB) (Otten and Dechter, 2011), one of the best anytime schemes for graphical models.

Our results show that weighted best-first schemes can provide a useful addition to the class of anytime combinatorial optimization scheme for graphical models. The scheme was superior to BRAOBB on various instances, although BRAOBB was more cost-effective on the majority of the instances. In particular, the weighted scheme performed well when the heuristic evaluation function is of weak or medium strength. Our study is only at its initial phase however. To get more conclusive results we plan to conduct further investigation into weight policies and aim to identify problem's features and parameters of the algorithms that favor anytime best-first search over anytime depth-first search for graphical models.

For weighted search algorithms, including wAOBF and wR-AOBF, the weight parameter controls the trade off between the solution accuracy and the time and space requirements. The questions of how to choose the starting weight and how to decrease it over time have yet to receive a thorough study and such choices are usually made manually in ad hoc manner. At this point we experimented with several human-constructed weighting policies. However, it is desirable to learn the starting weight and weight policy that could yield the best anytime behaviour based on the problem parameters. Currently such automatic weight policy choice is a work in progress.

## 2. Anytime Heuristic Search Algorithms

In this section, we describe the anytime algorithms which we explore later in our empirical evaluation section. The main focus of the experiments is on the two anytime Best First schemes: wAOBF and wR-AOBF.

**Iterative Weighted AOBF (wAOBF):** Since the accuracy of a solution found by Weighted AOBF is bounded by $w$, it is possible to formulate a simple anytime scheme, called **wAOBF**, that executes Weighted AOBF iteratively, starting with an initial weight, and decreasing the weight at each iteration according to some predetermined rule. Clearly this approach, similar to the Restarting Weighted A* by Richter at al. (Richter, Thayer, and Ruml, 2010), results in a series of solutions, each with a sub-optimality factor equal to $w$. We experimented with multiple weight decreasing schedules, discussion of which we defer till the next section.

**Anytime Repairing AOBF (wR-AOBF):** Running each search iteration from scratch, as **wAOBF** does, is wasteful, since the same search subspace might be explored multiple times. To remedy this problem, we propose Anytime Repairing AOBF scheme, **wR-AOBF**, which is based on an idea of *Anytime Repairing A* (ARA*)* algorithm (Likhachev, Gordon, and Thrun, 2003). At each iteration, **wR-AOBF** keeps track of the partially explored AND/OR graph and, after decreasing $w$, it performs a bottom-up update of all node values starting from the leaf nodes (whose $h$-values are inflated with the new weight) and continuing upwards towards the root node. Then, the search continues with the newly identified best partial solution tree. Like ARA*, **wR-AOBF** provides the same guarantees with respect to the quality of the suboptimal solutions found.

We compare and contrast the performance of our anytime BF algorithms with the following two schemes:

**Anytime Stochastic AOBF (p-AOBF):** Another anytime scheme for AND/OR spaces is the *Anytime AO** algorithm (Bonet and Geffner, 2012) introduced recently for solving finite-horizon MDPs. The idea is to allow *AO** to also expand nodes that otherwise may not be on any optimal solution. Specifically, at each step the algorithm expands a tip node that does not belong to the current best partial solution graph with a fixed probability $(1 - p)$. The algorithm does not provide any theoretical guarantees on the quality of the intermediate solutions. The extension of the algorithm to the context minimal AND/OR search graph for graphical models is straightforward and is called

**p-AOBF**. As we show in experimental section, for optimization problems over graphical models p-AOBF rarely demostrates truly anytime behavior.

**Anytime AND/OR Branch and Bound (BRAOBB):** Depth first AND/OR Branch-and-Bound (AOBB) (Marinescu and Dechter, 2009a) is a powerful search scheme for graphical models. The algorithm, however, lacks a proper anytime behavior because at each AND node all but one independent child subproblems have to be solved completely, before the last one is even considered. *Breadth-Rotating AND/OR Branch-and-Bound* (BRAOBB) (Otten and Dechter, 2011) remedies this deficiency of AOBB by combining depth-first exploration of the search space with the notion of rotating through different subproblems in a breadth-first manner. Empirically, BRAOBB outputs the first suboptimal solutions significantly faster than plain AOBB (Otten and Dechter, 2011).

## 3. Experiments

We compare the anytime behavior of all the algorithms described above when exploring the same context minimal AND/OR graph for each problem instance. The search space is determined by a common variable ordering. All the algorithms are guided by the mini-bucket heuristics (Dechter and Rish, 2003), whose strength is controlled by a parameter called *i-bound* (higher i-bounds typically yield more accurate heuristics). The algorithms output solutions at different times until the optimal solution is found.

We considered 3 benchmarks: 16 pedigree networks, 17 binary grids, 6 SPOT5 Weighted CSPs and 4 driverlog problems. The task solved was Most Probable Explanation (max-prod), so higher costs are better. Each instance was solved for 11 different values of i-bound, ranging from 2 to 22.

For wAOBF and wR-AOBF we decided on a large initial weight equal to 64 due to our desire to obtain the initial solutions quickly and solve hard instances, many of which are infeasible for the algorithms using non-weighted heuristic.

Our empirical evaluation consists of two sets of experiments, results of which we report separately: 1) the comparison between wAOBF and wR-AOBF with the simplest weighting policies and the competing BRAOBB and p-AOBF algorithms, 2) the exploration of more sophisticated weighting policies.

### 3.1. Anytime weighted BFS vs Depth-First BB

At the first stage of the evaluation the anytime BF schemes used the two most straightforward weighting policies: 1) *substract*(0.1) - at each iteration the weight is decreased by a fixed amount 0.1; 2) *divide*(2) - at each iteration the weight is divided by 2. Preliminary experiments, omitted for lack of space, showed that for wR-AOBF the first policy provides better results, while for wAOBF the second policy was preferable.

We compare these two resulting schemes (denoted wR-AOBF and wAOBF-H respectively) with BRAOBB and p-AOBF (p=0.5), reporting the results in two ways: 1) by showing the performance on a set of representative problems that demonstrate the trends common to the majority of instances; 2) by summarizing statistics over all the instances in each benchmark.

The time limit for this set of experiments was 1 hour, memory limit was 4 Gb.

#### 3.1.1. Results on a selected set of instances

Figures 1 and 2 show the anytime profiles of wAOBF-H, wR-AOBF, BRAOBB and p-AOBF on select instances from all our benchmarks. For each instance the results for two i-bounds are displayed side by side: the left column corresponds to a weaker heuristic and the right one to a stronger one.

Comparing the anytime schemes on these representative problems we immediately see that p-AOBF is not competitive. On the **grid** and **pedigree** instances it either directly outputs the exact solution (e.g. pedigree38, i=12) or a solution close to optimal (e.g. 50-18-5, i=10) after a considerable time, contrary to the desirable anytime behaviour, namely quickly obtaining the initial solution and improving it over time. Moreover, p-AOBF fails to produce any solution within the time limit for instances with weak heuristic (e.g. 50-18-5, i=2) or for any of the **SPOT5** or **driverlog** problems (e.g. 505). As clearly unpromising algorithm p-AOBF is excluded from the subsequent discussion.

The performance of wAOBF, wR-AOBF and BRAOBB greatly depends on the benchmark and the heuristic strength. The following common trends can be distinguished:

**Grids:** wAOBF and wR-AOBF are often superior to BRAOBB when the heuristic is weak, i.e. the i-bound is considerably lower than the problem's induced width (e.g. 50-18-5, i=2). However, when the heuristic is stronger, BRAOBB is the fastest to find the optimal solution (e.g. 50-19-5, i=20). For easy problems solved

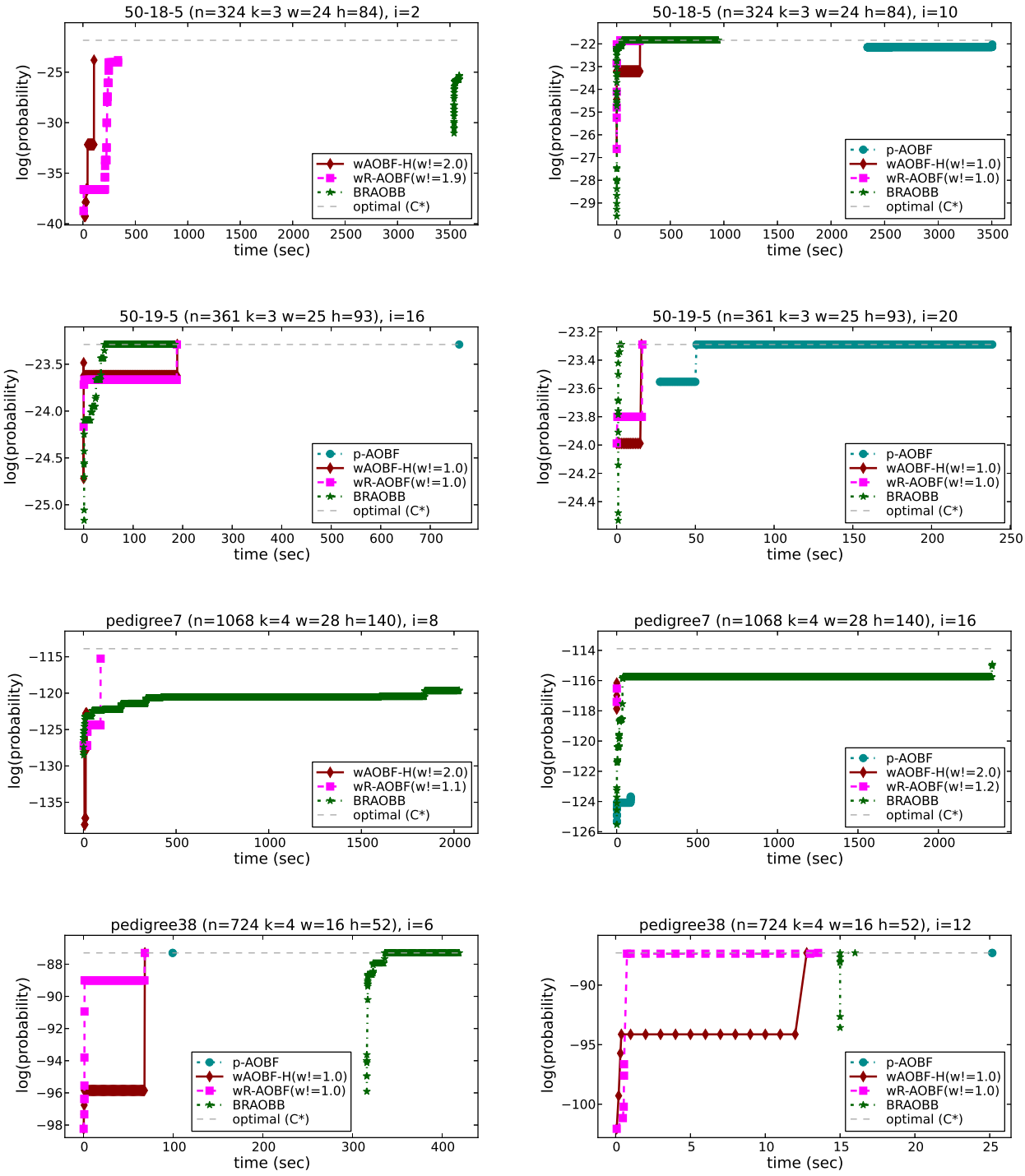


*Figure 1.* Grids and pedigrees, cost as a function of time (wAOBF-H, wR-AOBF, BRAOBB, p-AOBF). Starting weight = 64. $C^*$ - optimal cost, $i$ - i-bound, $n$ - number of variables, $k$ - domain size, $w$ - induced width, $h$ - pseudo-tree height, $w!$ - weight at termination. Time limit - 1 hour, memory limit - 4 Gb.

optimally in under 30 seconds, which are not presented here due to being of little interest, BRAOBB is almost always superior for all i-bounds.

Comparing the two BF schemes we see that wAOBF-H is inferior to wR-AOBF on most grid instances, due to both inefficient re-exploring of a large portion of the

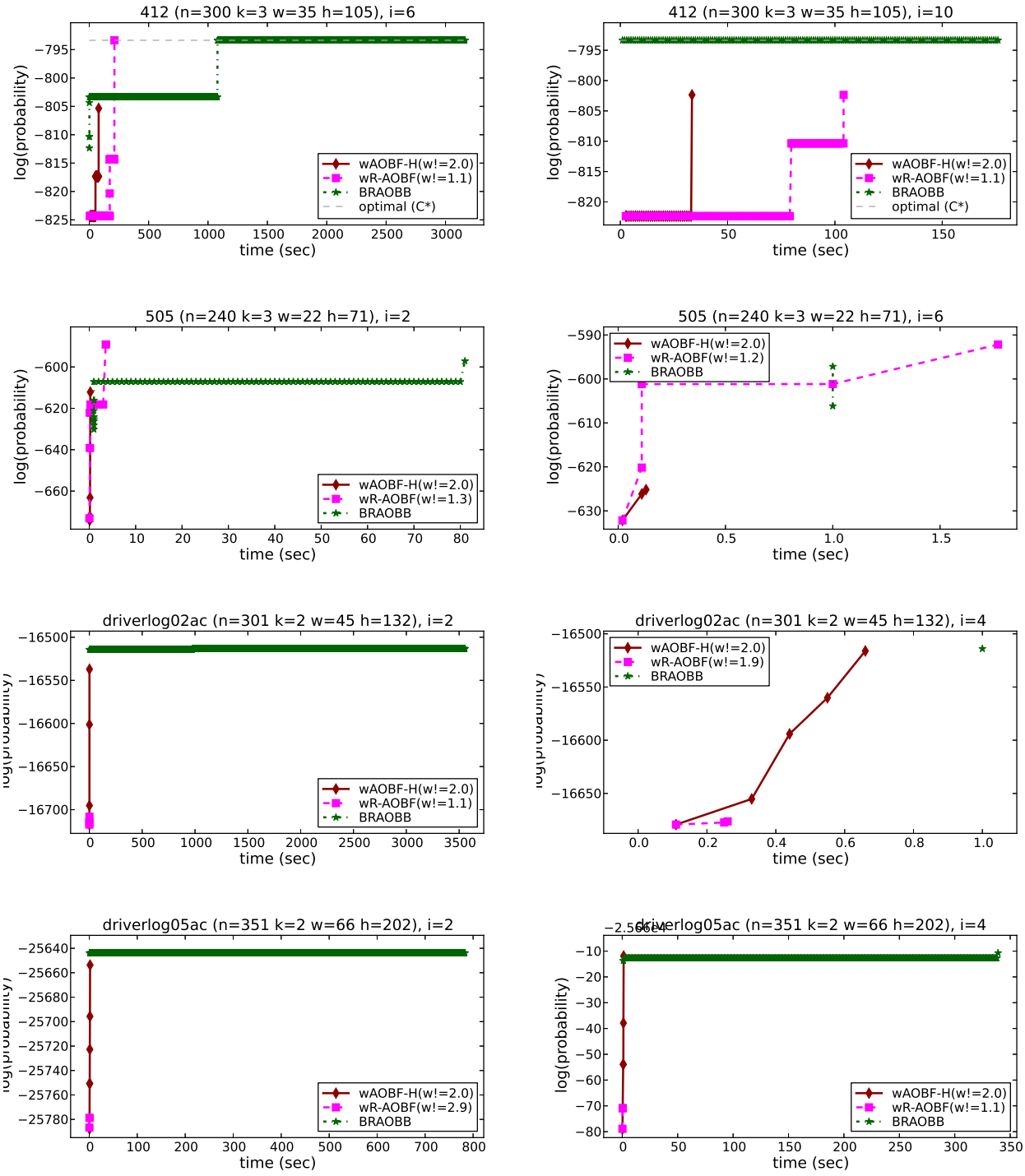


*Figure 2.* SPOT5 and driverlog WCSPs, cost as a function of time (wAOBF-H, wR-AOBF, BRAOBB, p-AOBF). Starting weight = 64, $C^*$ - optimal cost, $i$ - i-bound, $n$ - number of variables, $k$ - domain size, $w$ - induced width, $h$ - pseudo-tree height, $w!$ - weight at termination. Time limit - 1 hour, memory limit - 4 Gb.

search space and fast decreasing of the weight, which leads to fast increase in memory requirements. However, there are some exceptions, such as 50-18-5, i=2, where wAOBF-H achieves better results.

**Pedigrees:** these instances have many similarities to

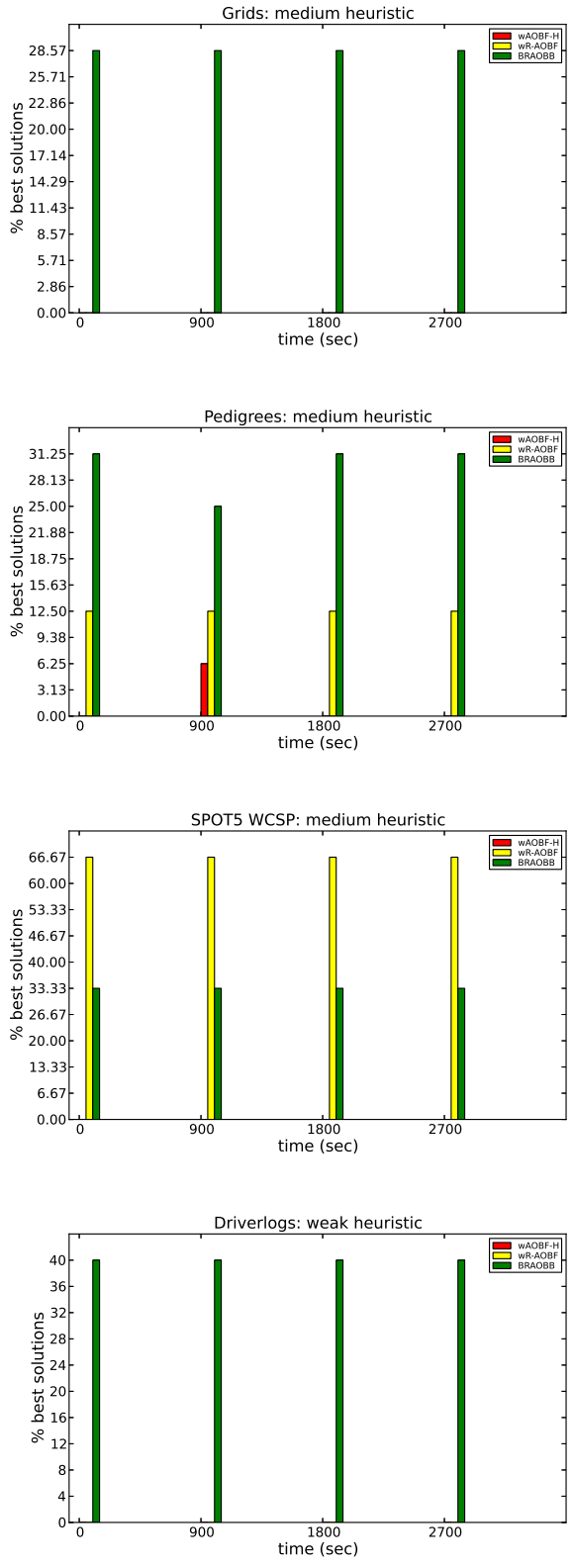


*Figure 3.* % of instances, for which algorithm finds best solution at a given time for a fixed heuristic strength.

the grids in terms of induced width, domain size and amount of determinism, so the behaviour of the algorithms on these two benchmarks are quite similar. Note that on largest problems, such as pedigree7 and pedigree38 shown here, BRAOBB is often inferior to both BF schemes even for heuristics of considerable strength (e.g. pedigree38, i=12), while wAOBF-H almost always performs worse than wR-AOBF.

**SPOT5 WCSPs:** these problems are more challenging for all our schemes. On some of the problems BRAOBB reaches the optimal solution quickly, but fails to prove it's optimality withing the time limit (e.g. 412, i=6), on others it outputs inferior bounds (e.g. 505, i=6). Both BF schemes tend to run out of memory before finding solutions of good accuracy (e.g. 412, i=10). For the hardest instances, such as 505, none of the algorithms managed to produce exact solution.

**Drivelogs:** the hardest of our benchmarks present difficulties for all 3 algorithms. The cost of the optimal solutions for these instances are unknown, though the behaviour of BRAOBB on certain instances (e.g. driverlog05ac, i=2) suggests that it did find the exact solutions without proving their optimality within allocated time. Unlike the previous benchmarks, wAOBF-H is better than wR-AOBF, consistently outputting more accurate solutions. Its success on these memory-intensive instances can be attributed to more modest memory requirements, since wR-AOBF needs to keep track of the partially explored AND/OR graph during each iteration.

### 3.1.2. Aggregated results

Figure 3 displays the fraction of instances, for which a particular algorithm found the best solution at a specific time point. The p-AOBF is discarded from consideration as obviously inferior. For space reasons we only display results for medium heuristic strength, except for driverlogs, for which only weak heuristic is feasible. We do not count the ties between algorithms and thus the results not necessarily sum to 100% at each time point.

For grids and driverlogs BRAOBB is the best scheme in about 30% and 40% of instances respectively for all the time points considered. For pedigrees wR-AOBF finds the most accurate solution for about 12% of instances and even wAOBF-H is superior in around 6% of cases for a particular time point. On SPOT5 WCSPs wR-AOBF demonstrates clear superiority, obtaining the best solutions in up to 2/3 of instances.

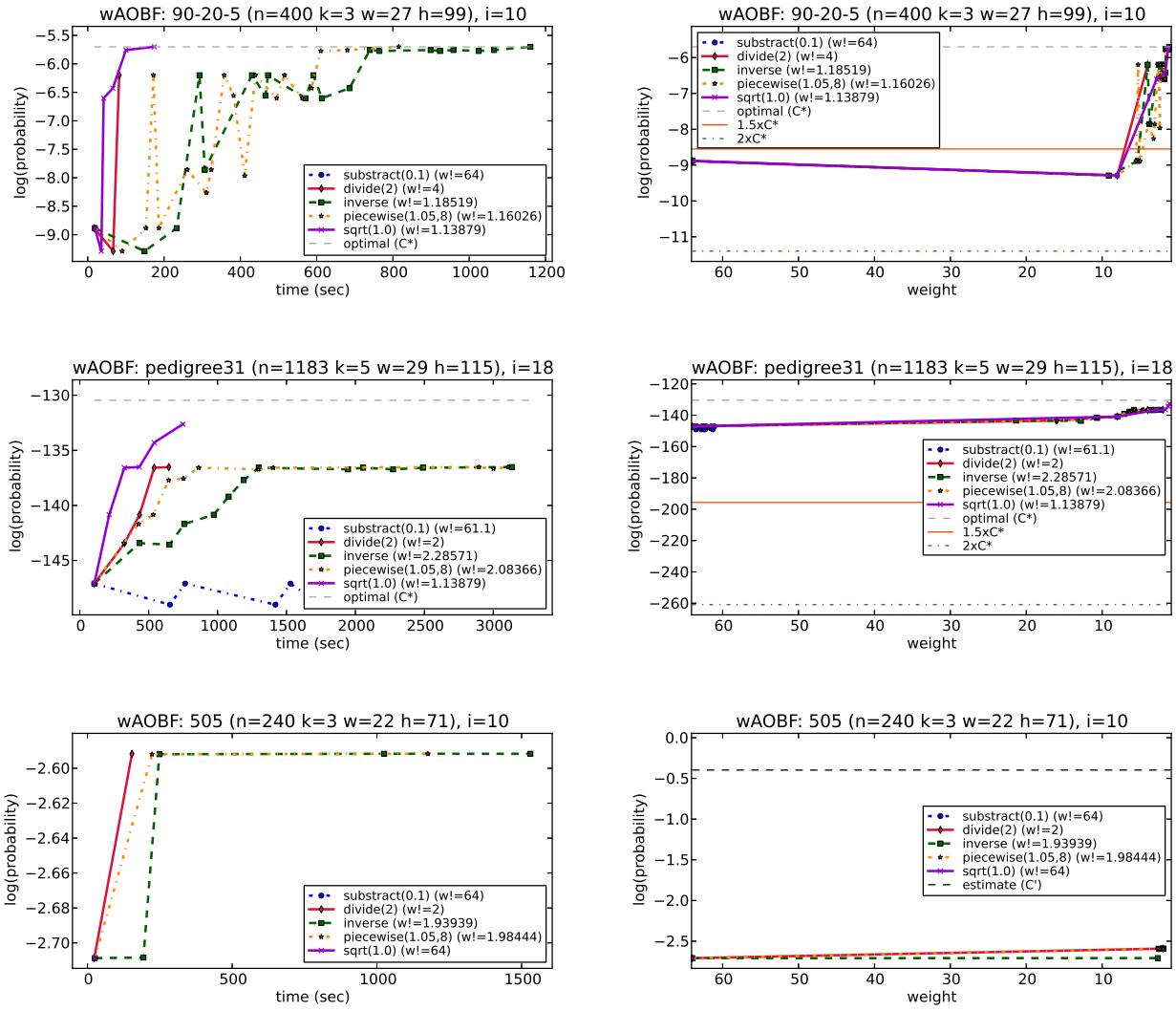


*Figure 4.* wAOBF: solution cost vs time (left) and weight (right), starting weight = 64. $C^*$ - optimal cost, "-i" - i-bound, $n$ - number of variables, $k$ - domain size, $w$ - induced width, $h$ - pseudo-tree height, $w!$ - weight at termination, $C' = w! \cdot C$ - optimal solution estimation.

### 3.2. The influence of the weight on anytime BF schemes.

Aiming to further improve the performance of anytime BF schemes and to gain a better understanding of the impact of weight on their performance we devised more sophisticated weight policies than those discussed in the previous subsection. We experimented with 5 different strategies for decreasing the weight, including the *substract* and *divide* strategies mentioned above.

The notation is as follows: $w_i$ is the weight used at the $i^{th}$ iteration, $w_1 = 64$, $k$ and $d$ are real-valued policy parameters. We tried the following policies, each for several values of parameters:

- $substract(k)$: $w_i = w_{i-1} - k$
- $divide(k)$: $w_i = w_{i-1}/k$
- $inverse$: $w_i = w_1/i$
- $piecewise(k, d)$: if $w_i \geq d$ then $w_i = w_1/i$ else $w_i = w_{i-1}/k$
- $sqrt(k)$: $w_i = \sqrt{w_{i-1}}/k$

The *substract* and *divide* policies lay on the opposite ends of the strategies spectrum. The first method changes the weight very gradually and consistently, leading to a slow improvement of the solution. The second approach yields less smooth anytime behaviour,

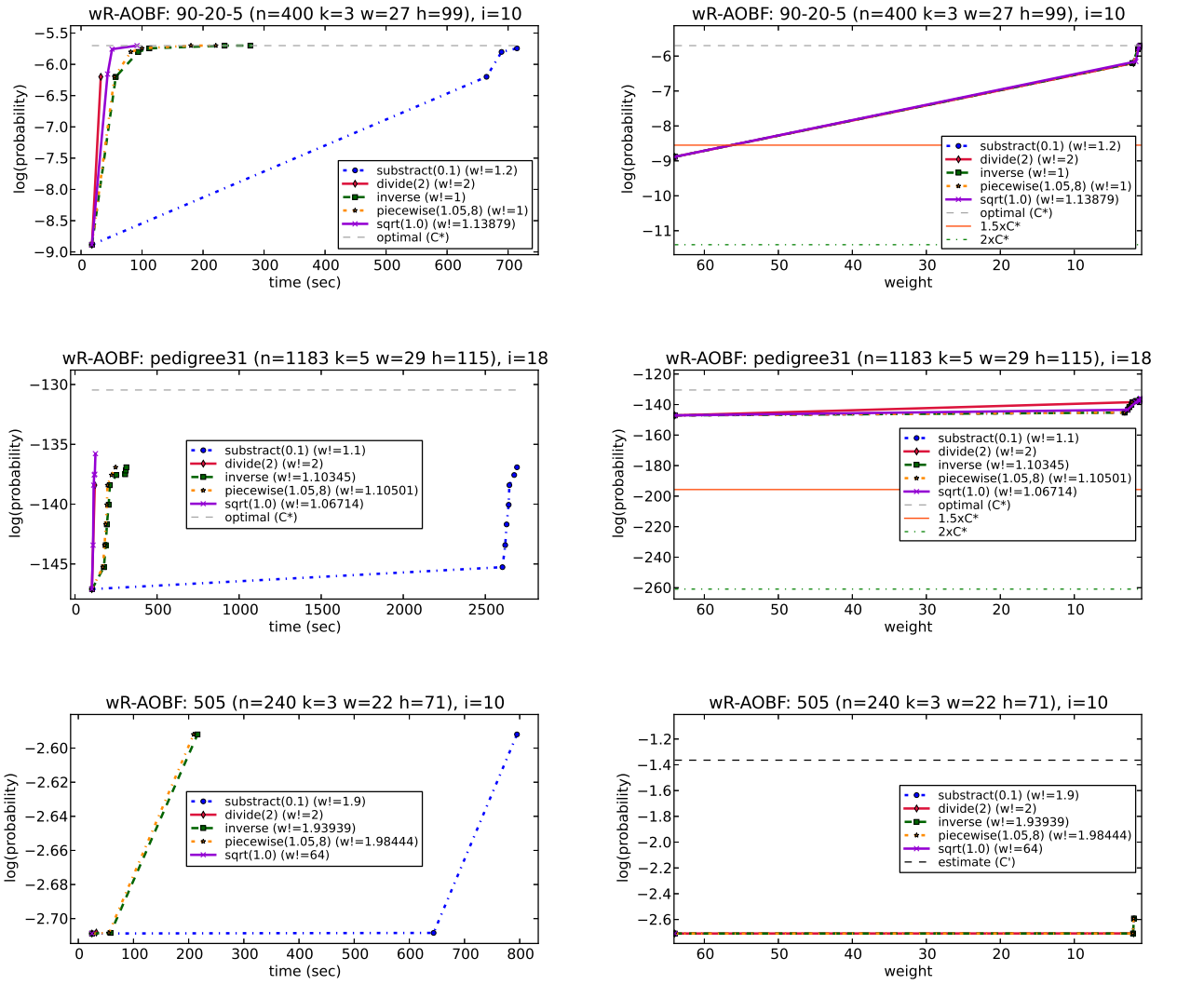


*Figure 5.* wR-AOBF: solution cost vs time (left) and weight (right), starting weight = 64. $C^*$ - optimal cost, "-i" - i-bound, $n$ - number of variables, $k$ - domain size, $w$ - induced width, $h$ - pseudo-tree height, $w!$ - weight at termination, $C' = w! \cdot C$ - optimal solution estimation. Time limit - 1 hour, memory limit - 2 Gb.

since the weight rapidly approaches 1 and much fewer intermediate solutions are found. This could potentially allow the schemes to produce the exact solution fast, but on hard instances presents a danger of leaping directly to a prohibitively small weight and thus failing, without exploring feasible weights that could potentially produce good solutions. The other policies were constructed manually based on a following intuition: we wanted to improve the solution rapidly by decreasing the weight fast initially and then "fine-tune" the solution as much as the memory limit allows, by decreasing the weight slowly as it approaches 1.

The performance of our anytime BF schemes employing these weight policies on the representative instances for each benchmark are shown in Figures 4 (wAOBF) and 5 (wR-AOBF). The left column displays the solution cost as a function of time, the right one - as a function of the weight. This set of experiments was conducted on a weaker machine that the first one, thus the memory limit was 2 Gb, with time limit of 1 hour. Several values of numerical parameters for each policies were tried, the ones that yielded the best performance are included here. The starting weight equals 64, $w!$ denotes the weight at the time of algorithm termination.

Consider the left columns of Figures 4 and 5. The superior policy is benchmark dependent. For most values of i-bound both wR-AOBF and wAOBF algorithms

achieve the best results for hard **grids** and **pedigrees** using the *sqrt* policy, as shown here on the example of instances 90-20-5 and pedigree31 respectively. However, for the **SPOT5** WCSPs (e.g. 505) the results are more varied. For higher values of i-bounds (e.g. i=10) wAOBF achieves the best results using *divide* policy, while for wR-AOBF *piecewise* is superior with *inverse* coming close second. Weaker heuristics yield problems that are too large for either of algorithm to solve for any weight smaller than initial values of 64 and thus are not discussed.

The right columns of Figures 4 and 5 demonstrate that, unsurprisingly, there is little variance between different policies in terms of the dependency of the solution cost on the weight. For the grid and pedigree instances we also show the theoretical bounds, corresponding to weights 1.5 and 2 (i.e. $1.5 \cdot C^*$ and $2 \cdot C^*$). In practice the solutions found by both wAOBF and wR-AOBF are better than suggested by theory. However, when the termination weights $w!$ are close to 1 (e.g. for grid 50-20-5, i=10, $sqrt(1.0)$ $w! = 1.13879$), it allows us to obtain reasonably accurate theoretical bounds, especially useful if the optimal solution is not available, as is the case for problem 505. For this instance we plot an estimation $C'$ equal to the best available bound $w! \cdot C$. The true optimal solution would lay between the lines corresponding to the algorithms' outputs and $C'$.

Overall, it is clear that the new weighting policies, such as *sqrt* allow to achieve better results than the simpler *substract* and *divide* ones that we considered initially. A thorough empirical evaluation of the superior policies using larger memory limit is currently a work in progress.

### 3.3. Summary of the experimental results

The main conclusions that can be drawn from our experiments are:

1. Both wAOBF and wR-AOBF are inferior to BRAOBB in cases of: a) easy instances b) harder instances with strong heuristics c) small memory limit; The anytime BF schemes are competitive or superior large problems and weak to medium heuristics. 2. The performance of wAOBF and wR-AOBF is significantly sensitive to the weight policies.

## 4. Conclusion

In this paper we conducted an extensive empirical evaluation of several anytime best-first schemes for graphical models, and showed that they have potential for providing an alternative to anytime branch-and-bound search for some benchmarks. In particular, we found that even our simplest scheme wAOBF, that iteratively solves the problem from scratch, gradually decreasing the weight, can be reasonably effective for problems with high induced width, such as some SPOT5 WCSPs, while the more sophisticated wR-AOBF proved successful for many problems of moderate difficulty, such as large grids and pedigrees. We predict further improvement of the performance if larger amount of memory is available.

The advantage of the weighted anytime schemes is that they output the weighting parameter which provides an upper-bound guarantee. This bound can be informative when the anytime scheme is employing small weights and optimal solution is unknown.

We plan to further improve the performance of our anytime best-first schemes by using sophisticated weight policies and by investigating the connection between the weight, problem's induced width and the algorithm's performance in order to automatically learn the most effective weighting schedule instead of choosing parameters ad-hoc. We are also exploring possible non-parametric approaches.

## Acknowledgement

This work was supported by NSF grant IIS-1065618.

## References

Bonet, B., and Geffner, H. 2012. Action selection for MDPs: Anytime AO* vs. UCT. In *AAAI*.

de Givry, S.; Heras, F.; Zytnicki, M.; and Larrosa, J. 2005. Existential arc consistency: Getting closer to full arc consistency in weighted csps. In *IJCAI*, 84–89.

Dechter, R., and Pearl, J. 1985. Generalized best-first search strategies and the optimality of A*. *Journal of the ACM* 32:506–536.

Dechter, R., and Rish, I. 2003. Mini-buckets: A general scheme for bounded inference. *Journal of the ACM* 50(2):107–153.

Hansen, E., and Zhou, R. 2007. Anytime heuristic search. *Journal of Artificial Intelligence Research* 28(1):267–297.

Kask, K., and Dechter, R. 2001. A general scheme for automatic search heuristics from specification dependencies. *Artificial Intelligence* 129(1–2):91–131.

Likhachev, M.; Gordon, G.; and Thrun, S. 2003.

ARA*: Anytime A* with provable bounds on sub-optimality. *NIPS* 16.

Marinescu, R., and Dechter, R. 2009a. AND/OR Branch-and-Bound search for combinatorial optimization in graphical models. *Artificial Intelligence* 173(16-17):1457–1491.

Marinescu, R., and Dechter, R. 2009b. Memory intensive AND/OR search for combinatorial optimization in graphical models. *Artificial Intelligence* 173(16-17):1492–1524.

Otten, L., and Dechter, R. 2011. Anytime AND/OR depth first search for combinatorial optimization. In *SOCS*.

Pohl, I. 1970. Heuristic search viewed as path finding in a graph. *Artif. Intell.* 1(3-4):193–204.

Richter, S.; Thayer, J.; and Ruml, W. 2010. The joy of forgetting: Faster anytime search via restarting. In *ICAPS*, 137–144.

van den Berg, J.; Shah, R.; Huang, A.; and Goldberg, K. 2011. Anytime nonparametric A*. In *AAAI*.