

Failure Analysis in Backtrack Search for Constraint Satisfaction

Tudor Hulubei and Barry O’Sullivan

Cork Constraint Computation Centre
Department of Computer Science, University College Cork, Ireland
tudor@hulubei.net, b.osullivan@cs.ucc.ie

Abstract. Researchers have focused on assessing the quality of search algorithms by measuring effort, number of mistakes, runtime distributions and other characteristics. In this paper we attempt to enhance our understanding of these algorithms by employing heatmap-based visualisations. We perform a large empirical study of search, for a variety of problem classes, and we report several interesting observations. Our results show that, contrary to conventional wisdom, it is not always the case that mistakes made at the top of the search tree are exponentially more expensive to refute than those made deeper in the tree, or that over a population of instances the largest mistakes dominate search effort. Furthermore, we show empirically that heavy-tailed runtime distributions can occur even if the number of mistakes is small, and that instance and mistake-based runtime distributions can be very similar. Finally, we study the complex relationship between the promise and fail-firstness of variable ordering heuristics.

1 Introduction

Over the past 30 years many papers have been published reporting empirical studies of systematic search in constraint satisfaction. Typically, the empirical comparison is made on the basis of a measure of search effort in order to assess the quality of a given algorithm. Search effort is typically measured in terms of the number of backtracks, constraint checks, or nodes in the search tree, but measures such as the number of incorrect decisions have also been proposed [3]. Comparisons based on average and median effort are common. However, other researchers focus on studying runtime distributions, where one can observe a (non-)heavy-tailed distribution under certain conditions [9, 10].

There are many questions one may wish to answer in general through empirical studies of search. For example, what makes a runtime distribution (non-)heavy-tailed? Why is one combination of search heuristics better than another? What should heuristics try to achieve in order to give improved performance on a specific class of problems? In this paper we augment our traditional statistics-based approach to studying systematic search with a visualisation method that uses heatmaps. Heatmaps are essentially two-dimensional views of three-dimensional spaces in which colour is used to represent the magnitude in the third dimension. They are easier to interpret than traditional 3D plots, which are often difficult to visualise. We show that heatmaps can be more informative and present different views of the relationship between the depth at which a mistake occurred and the size of the refutation needed to recover from it.

We observe that for the problems and algorithms we considered, the number of mistakes (decisions that take search off the path to a solution) increases only very slowly with the effort, measured in number of nodes, required to solve an instance. Moreover, the runtime distribution of the effort required to refute each individual mistake and that of the overall effort required to solve each instance are very similar. Therefore, we compare search algorithms on the basis of the number of, and effort required to recover from, *individual mistakes* made during search.

We highlight interesting differences between random and real-world problems. In particular, the conventional wisdom is that mistakes made at the top of the search tree are exponentially more expensive to refute than those deeper in the tree. This is exactly the type of behaviour one observes in uniform random binary problems, but it is not the case for problems such as quasigroups of order 10 with 90% random balanced holes.

We also observe some interesting patterns in terms of where most of the search effort is consumed *over a large population of problem instances*. Specifically, it is not always the case that extremely large mistakes account for most of the effort. Indeed, we will show that for easier problems the large number of smaller mistakes made deeper in the search tree consume significantly more effort than the very large, but very much rarer, mistakes. However, as problems become harder the effort required to recover from mistakes made near the top of the tree starts dominating the effort required to recover from all the others.

Finally, our work fits into a broad literature on the study of search heuristics and search effort. Haralick and Elliott [12] have argued that an important property of a variable ordering is its ability to recover from failure. However, Smith and Grant [17] have shown that trying harder to fail earlier is not enough to reduce overall search effort. In this paper, we consider the interrelationship between the promise [1] and fail-firstness exhibited by variable ordering heuristics. We show that variable ordering heuristics alone can avoid making mistakes but that their performance cannot be attributed exclusively to either fail-firstness or promise. Indeed, quite a complex relationship exists between these two properties.

The remainder of this paper is organised as follows. In Section 2 we summarise the basic measurements and tools we used in our analysis. Section 3 presents the details of the experimental setup, as well as descriptions of the problem classes we studied and the size of our data-set in each case. We consider the distribution of search efforts across various depths in the search tree in Section 4, and reveal some interesting phenomena, as referred to above. In Section 5 we show that our approach to visualising search gives some interesting insights into the relationship between search efficiency, fail-firstness of ordering heuristics and their promise. A number of concluding remarks are made in Section 6.

2 Preliminaries

We study the failure characteristics of backtrack search methods in constraint satisfaction problems [6]. Our analysis is based on counting the number of times an assignment was made during search that took us off the path to a solution. We refer to such decisions as *mistakes* [14]. The set of nodes visited by the algorithm in order to recover

We generated the uniform random binary problems with a Model B generator [8], and with the exception of the random 17×8 problems, where we used backtracking, all other experiments used MAC. Our data sets of random problems contain approximately 10,000 instances for each algorithm used. The QWH-10 data set includes a total of over 1,000,000 instances. For both types of problems, we analysed a significant number of distinct instances rather than generating a single instance and relying on random tie-breaking to obtain a distribution of runs. Specifically:

1. QWH-10 with 90% random balanced holes¹. We solved the instances in the data set with various algorithms obtained by combining MAC with several variable ordering heuristics, such as min-domain [12], min-dom/ddeg² [2], brelaz [5] and min-dom/wdeg [4, 15], and value ordering heuristics, such as random and min-conflicts [7].
2. Random binary problems with 17 variables and uniform domain size 8, configured at density 0.84 with two tightness settings: 0.09375, in the easy region, heavy-tailed when combining backtracking with random variable and value orderings; and 0.25, near the phase transition [13], non-heavy-tailed due to uniform hardness. These are the configurations used in [9] and [14].
3. Dense random binary problems with 30 variables and uniform domain size 10, configured at density 0.86 and tightness 0.15, near the phase transition.
4. Sparse random binary problems with 30 variables and uniform domain size 10, configured at density 0.3 and tightness 0.35, in the easy region.
5. Sparse random binary problems with 150 variables and uniform domain size 10, configured at density 0.03356 and tightness 0.52, in the easy region.

Rather than using the overall effort required to solve each instance, which we refer to as *instance-based effort*, as the basis of our analysis, we also considered the effort required to refute each mistake separately, which we refer to as *mistake-based effort*. The distinction between these approaches is that either each instance gives us a single data-point for the total effort required to refute all the mistakes in that instance, or many data-points for refuting each individual mistake, respectively. We verified that such a mistake-based analysis was an accurate characterisation of the overall instance-based effort required to solve the instances in our data-sets by making two key observations.

Firstly, we established that, for the problem classes we have studied, the average number of mistakes per instance grows very slowly with instance-based effort. This is an interesting and somewhat surprising result in its own right, even though the number of possible mistakes in an instance is at most $n \times (d - 1)$, where n is the number of variables and d is the uniform domain size. Figure 2 presents a representative sample of these results (all other problem classes exhibited similar behaviour). Irrespective of the heuristics used, we only observe a slight increase in the average number of mistakes per instance as the effort required to solve such instances (measured in number of nodes in the search tree) increases by several orders of magnitude. Furthermore, it is interesting to note that the extremely difficult instances that we often encounter, for example in heavy-tailed data sets, are not necessarily made up of a significant number of mistakes

¹ Generated using code based on Carla Gomes' *lsencode* quasigroup generator.

² In this paper we abbreviate dynamic-degree as 'ddeg' and weighted-degree as 'wdeg'.

that are moderately easy to refute, but of a small number of mistakes (oftentimes just one) whose corresponding refutations are exponentially large.

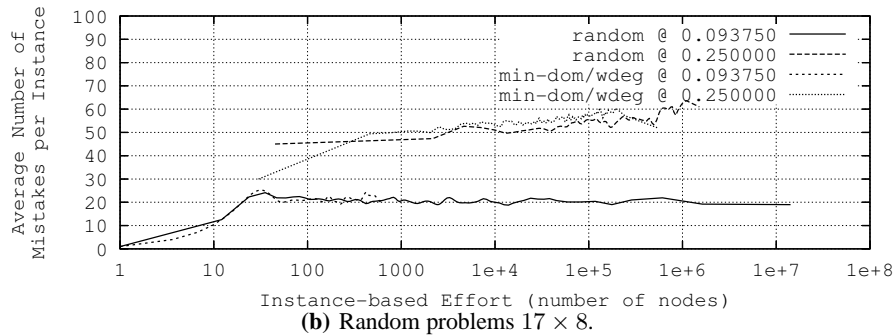
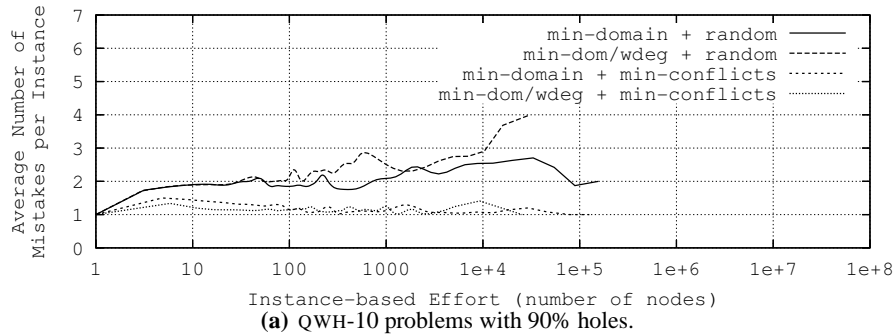


Fig. 2: Bezier approximations of the average number of mistakes per instance for various combinations of heuristics. Similar results are observed for the random binary problems 30×10 .

Secondly, considering the runtime distributions of both the mistake-based and instance-based efforts, Figure 3 shows that these distributions are very similar, in some cases almost identical, with the number of very small mistakes being the factor determining the proximity of the two runtime distributions. Backtrack-free instances do not affect the shape of these plots and have not been included.

The similarity of the shapes proves that for the problems we studied, the runtime distribution of the mistake-based effort is highly correlated with that of the instance-based effort, and so observations made on the former can be used to draw conclusions about the latter. Studying search algorithms at the mistake-level allows us to perform a more detailed analysis of the interactions between variable and value ordering heuristics over a large population of instances. For the remainder of the paper we will base our

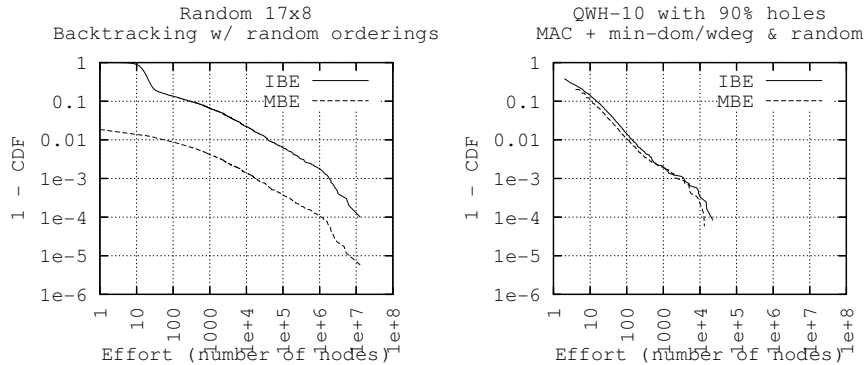


Fig. 3: Comparison of survival functions for the heavy-tailed random problems 17×8 data set (left) and one of our heavy-tailed QWH-10 data set (right). Other plots, heavy-tailed or not, look similar. IBE is *instance-based effort*, MBE is *mistake-based effort*.

comparison of search heuristics on an analysis of mistake-level effort through the use of heatmaps (**best viewed in colour**)³.

4 Distribution of Search Effort

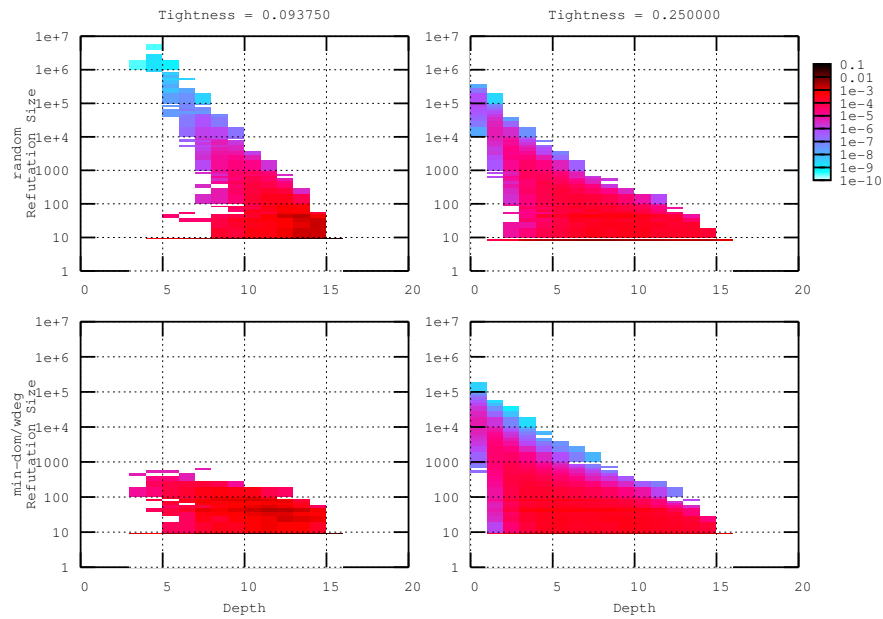
For random binary problems with 17 variables and a uniform domain size of 8, as well as for QWH-10 with 90% holes, Figures 4 and 5 show heatmaps of the probability of encountering mistakes of a certain size at a certain depth (Figures 4(a) and 5(a)), as well as the proportion of effort spent at a certain depth in refutations of a certain size (Figures 4(b) and 5(b)). Note that the colours represent a log-scale.

We begin by observing that in the easy region, for random 17×8 instances, using a random variable ordering with backtrack search gives a heavy-tailed distribution, while in the hard region this behaviour is not present (Figure 6(a)). While the typical survival function-based analysis [9] will demonstrate the presence or absence of such large mistakes, our heatmap visualisations also show us precisely the depth where these mistakes are being encountered.

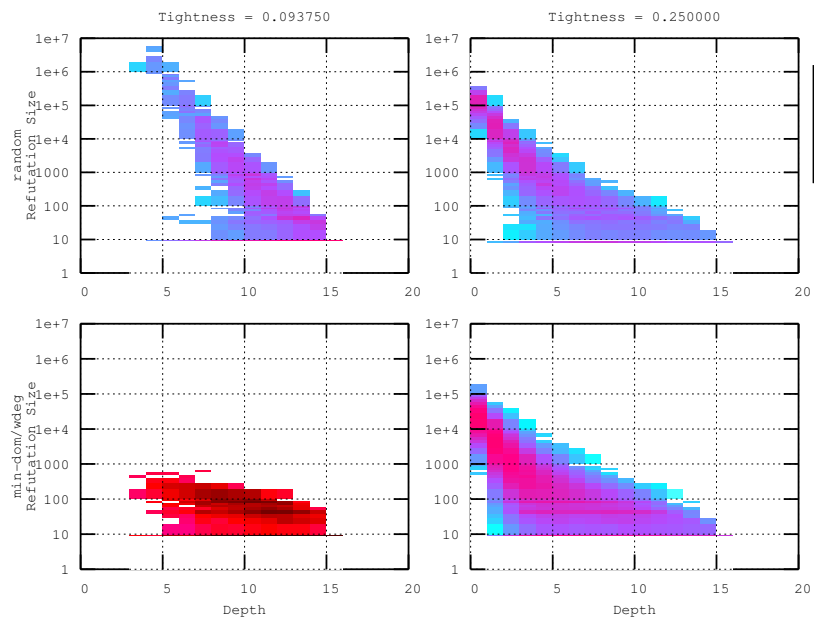
It is also interesting to consider how the size of mistakes varies with depth. The conventional wisdom is that mistakes made at, or near, the top of the tree are exponentially larger than those made deeper in the tree. However, the plots in Figure 5 contradict that assumption – for QWH-10 with 90% holes the largest mistakes occur at *intermediate* depths.

The heatmap in Figure 5(b) depicts, *for a population of instances*, the proportion of effort required to refute, for QWH-10 with 90% holes, mistakes of various sizes at each depth. The darkest spots in the heatmap represent those mistake sizes for which the cumulative effort over all the mistakes in our data set was proportionally the largest at

³ An electronic version of these plots can be seen in full colour online at: <http://hulubei.net/tudor/papers/fabscs>.

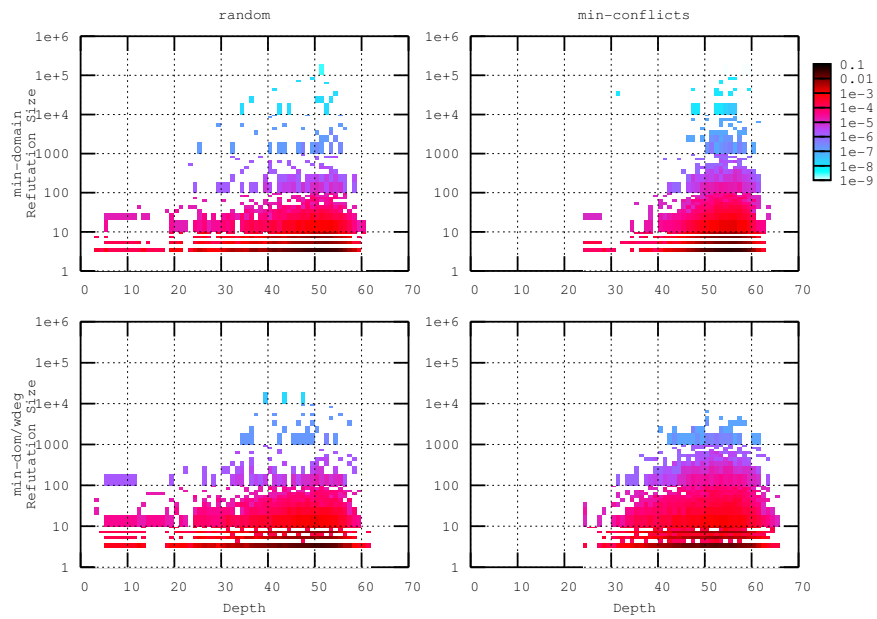


(a) Random problems 17×8 (left: easy, right: hard): probability of a refutation of a certain size (y-axis) occurring at a certain depth (x-axis).

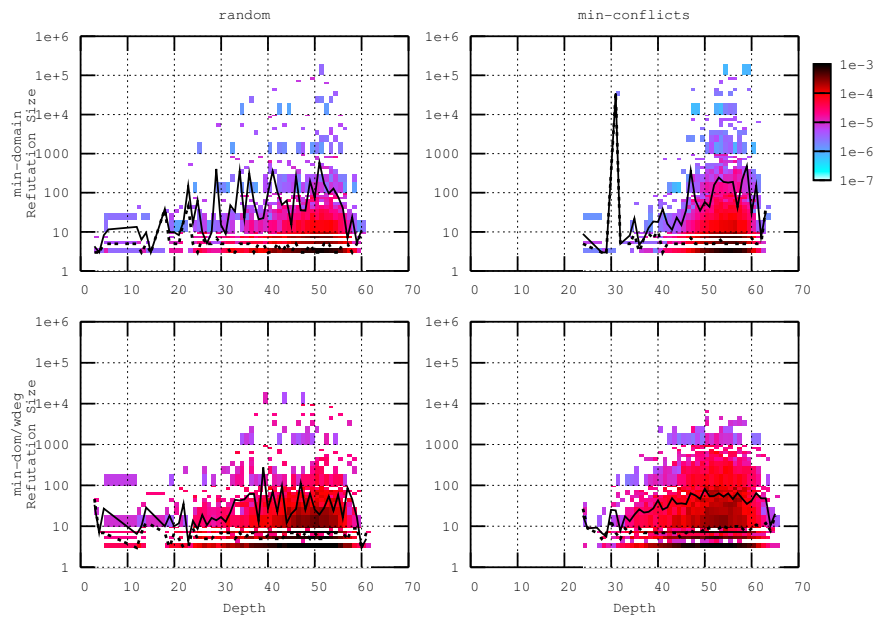


(b) Random problems 17×8 (left: easy, right: hard): proportion of effort spent in refutations of a certain size (y-axis) at a certain depth (x-axis).

Fig. 4: Heatmaps for random problems 17×8 .



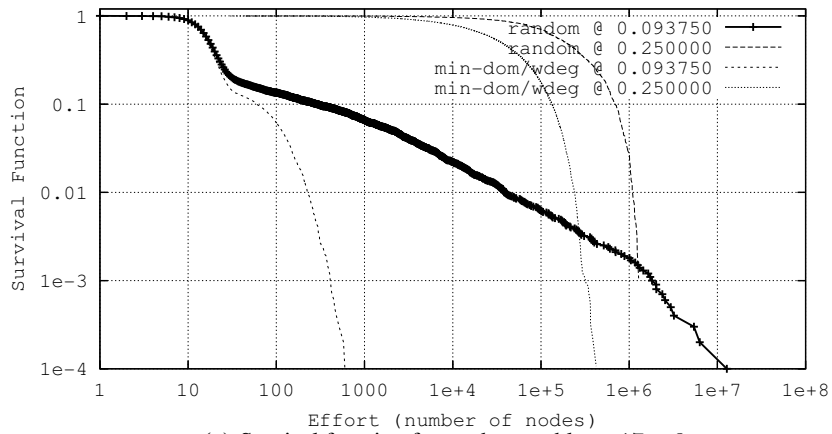
(a) QWH-10 problems with 90% holes: probability of a refutation of a certain size (y-axis) occurring at a certain depth (x-axis).



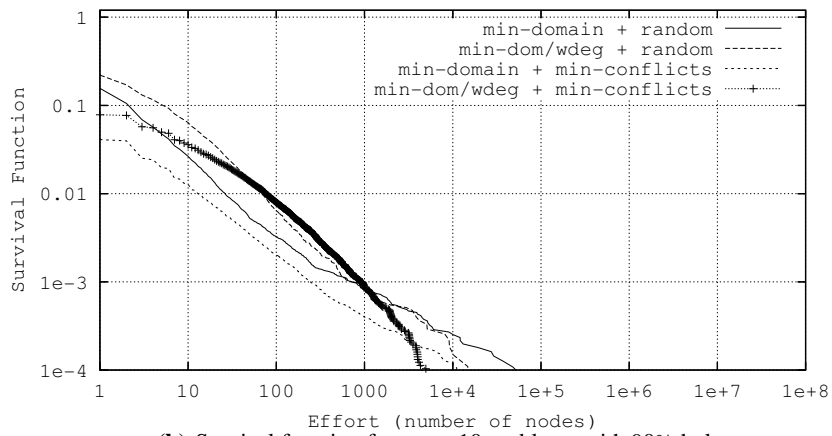
(b) QWH-10 problems with 90% holes: proportion of effort spent in refutations of a certain size (y-axis) at a certain depth (x-axis). Medians as dotted line, averages as continuous line.

Fig. 5: Heatmaps for QWH-10 with 90% holes.

that depth. It is clear that over a population of instances, for all the algorithms we used, the bulk of the effort is spent in refuting the extremely large number of small mistakes (4 to 100 nodes) that occur deep down in the search tree (Figure 5(a)). In fact, when the min-conflicts value ordering heuristic is used, no mistakes occur higher than depth 23. This is not the case when values are selected randomly (or using max-conflicts - not shown for lack of space), and suggests that for this particular class of problems better algorithms tend to avoid making mistakes at, or near, the top of the search tree [14].



(a) Survival function for random problems 17×8 .



(b) Survival function for QWH-10 problems with 90% holes.

Fig. 6: Survival function for runtime distribution.

Random binary problems do exhibit exponential decay of the refutation size with depth (Figures 4(a), 4(b) and 9(b)); similar results have been observed for random problems 150×10 and 30×10). Moreover, as random problems approach the phase

transition, mistakes start occurring close to the root of the tree, and over a population of instances, the bulk of the effort, which corresponds to the darker colouring in the heatmaps, shifts towards the top of the tree. Note that past the phase transition, problems become insoluble and the root of the search tree becomes the only mistake point accounting for the entire search effort.

In terms of methodology, the usefulness of heatmaps becomes even more apparent when looking at the average and median refutation sizes in Figure 5(b), which are plotted as lines on the heatmaps to aid comparison. The plots clearly illustrate how averages and medians would fail to provide any information with respect to the wide range and distribution of refutation sizes encountered for these algorithms. The number of small-to-medium size refutations is so significant that they not only cause the median to remain below 10, but they also prevent the extremely large mistakes from visibly contributing to the average⁴ (log-scale on the y-axis). Furthermore, the medians and averages in Figure 5(b) do not really indicate where the effort is in terms of depth. For instance, the average and medians around depth 40 are similar to those at depth 50, but the intensity of the colour in the heatmap is very different, showing that most of the effort is spent at depth 50. This is only visible with the heatmap.

While *for a population of instances* of QWH-10 the effort seems dominated by the disproportionately large number of small mistakes, this cannot be the case for any particular difficult instance. Figure 2 shows that the average number of mistakes per instance increases very slowly with effort. It follows that, typically, *for any particular difficult instance*, a small number of large refutations dominate the search effort.

MAC combined with min-conflicts and min-dom/wdeg is the only algorithm in our arsenal that can eliminate heavy tails for QWH-10 (Figure 6(b)) [14]. The bottom-right plot in Figure 5(b) shows a far smaller *variation* in the size of the refutations associated with mistake points than any of the other 3 plots. This reduced variation translates directly into a significant reduction in the variation in instance-based effort and is consistent with the non-heavy-tailed nature of that data set. Similar, but more complex behaviour can be observed in Figure 4(b), which depicts the data set whose runtime distribution can be seen in Figure 6(a). The upper-left heatmap is the only one corresponding to a heavy-tailed distribution. There are two characteristics of the other three heatmaps that help in visually determining the absence of heavy-tails. Firstly, in the lower-left plot there is insufficient variation in the refutation sizes encountered for heavy-tails to occur. Secondly, in the plots on the right, although there is significant variation in refutation size, the greatest proportion of our effort is associated with the large refutations. These dominate the search effort to the extent that refutations appear uniformly hard on the average, ensuring that the runtime distribution is not heavy-tailed, a known behaviour near the phase-transition.

Finally, it is worth noting that the heatmap visualisation also gives a nice intuition behind why rapid random restarts [11] is such a useful technique when solving problems with underlying heavy-tailed runtime distributions. For example, consider Figure 5 and the upper left plots in Figures 4(a) and 4(b), all corresponding to heavy-tailed runtime distributions. It is easy to see that while there is a very large variance in refutation sizes

⁴ The spike in top-right plot is caused by the fact that our data set contains only one mistake at depth 31 for that algorithm.

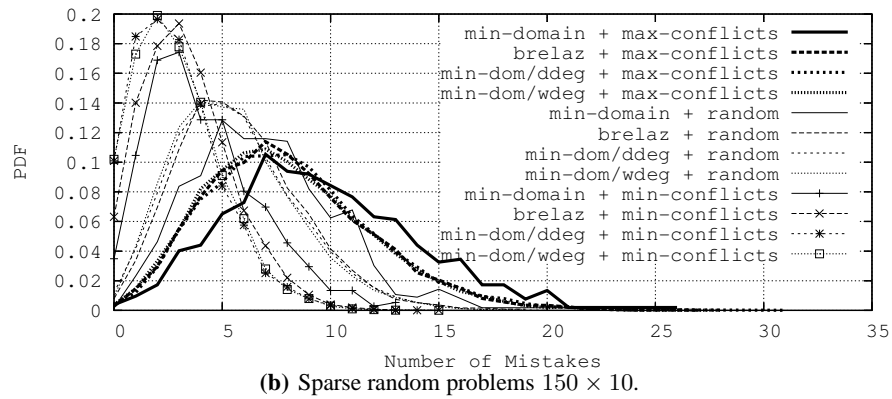
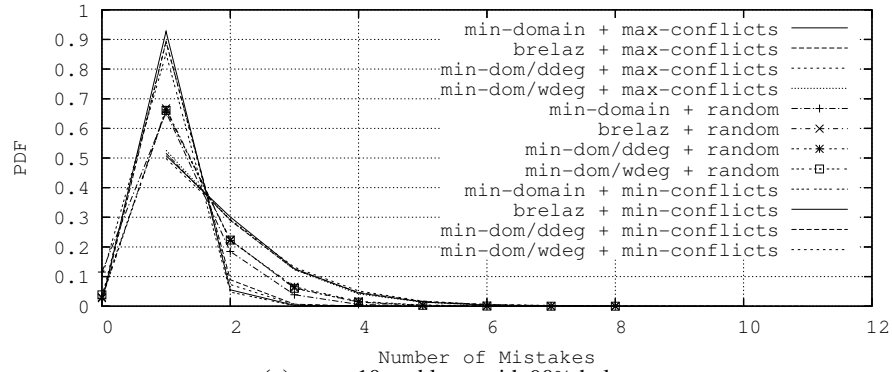


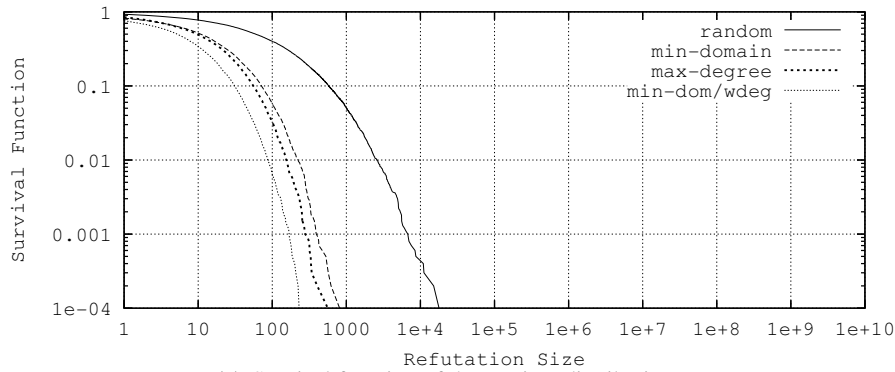
Fig. 7: Probability distribution functions (PDF) of the number of mistakes for different heuristics and problem classes.

at some depths, the mass of the effort is spent in smaller refutations. This is very noticeable in the upper left of Figure 4(a), where at depth 5 we have some refutations that involve 10 nodes, but a tiny proportion involve over 10^6 nodes. By randomly restarting search we are more likely to avoid the very large refutations. This is the standard intuition behind restarts, but the visual support for the intuition is very clear here.

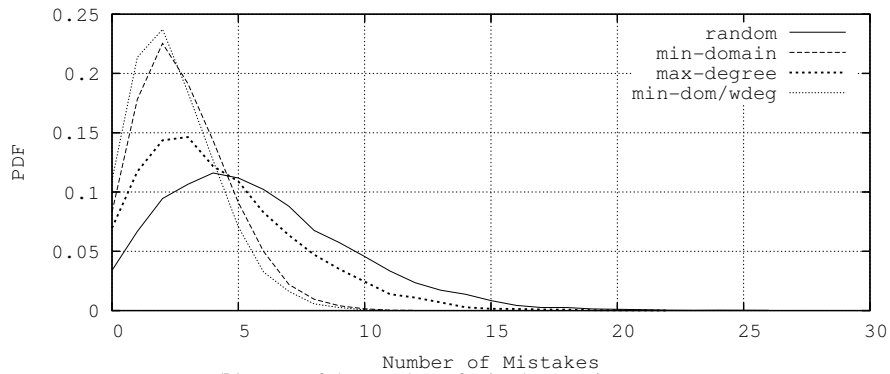
It is clear from these examples that heatmaps nicely complement the use of survival function plots and allow a more granular view of the failure characteristics of search.

5 Promise versus Fail-Firstness

The ability of variable ordering heuristics to reduce the effort required to refute insoluble subtrees, or fail-firstness [12], has been studied extensively. It has also been recently demonstrated that variable ordering heuristics also exhibit promise, independent of the value ordering used, i.e. they can contribute to a search algorithm's ability to avoid mak-



(a) Survival function of the runtime distribution.



(b) PDF of the number of mistakes per instance.

Fig. 8: Sparse random problems 30×10 : Promise versus fail-firstness; four variable ordering heuristics, randomly selected values.

ing mistakes [1, 18]. In that work, promise was measured based on the probability that search remains on the path to a solution. The promise of a variable ordering heuristic was measured using a statistical probing technique that tries to estimate how likely it is that a variable ordering will keep search on the path to a solution independently of the value ordering. Here we measure promise very differently and present an alternative analysis of the complex interaction between fail-firstness and promise.

While Figure 7(a) may suggest that variable orderings do not play a significant role in the number of mistakes an algorithm makes (the probabilities of a certain number of mistakes occurring cluster tightly around value orderings), this is less obvious in sparse random binary problems with 150 variables and uniform domain size 10 (Figure 7(b)).

The remainder of our experiments are based on sparse random problems with 30 variables and uniform domain size 10. Figure 8(a) clearly shows that min-dom/wdeg and random variable orderings are the best and worst heuristics, respectively, and that max-degree performs better than min-domain. Further supporting the usefulness of

heatmaps, we can see how Figure 8(b), as well as the averages and medians in Figure 9(a), portray min-domain and min-dom/wdeg as being very similar. The heatmaps in Figure 9(a), however, clearly show the mistakes made by the two heuristics are distributed differently across depths, with min-dom/wdeg having a higher probability of making more mistakes per instance over almost the entire range of depths it covers.

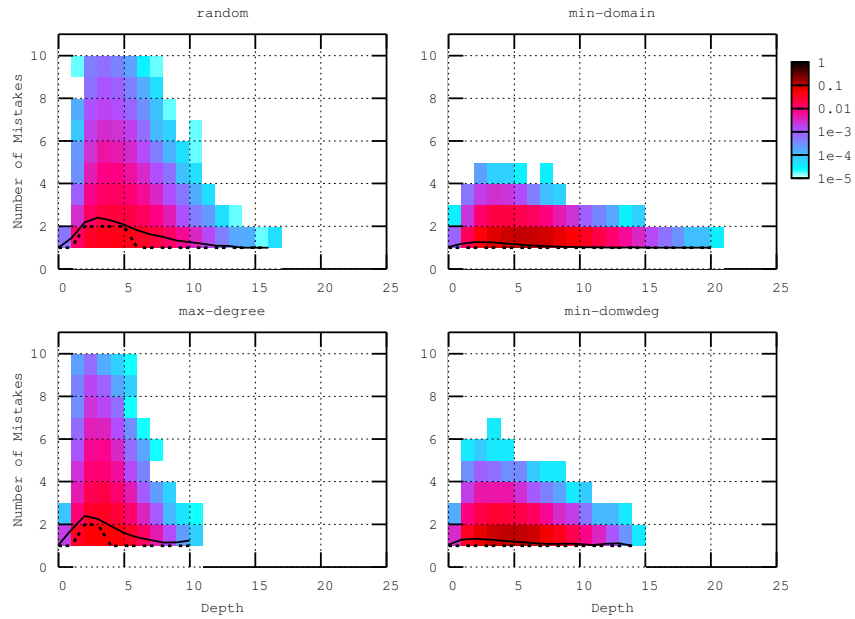
Figures 9(a) and 9(b) present a measure of promise and fail-firstness, respectively, for each variable ordering heuristic, arranged from left to right and top to bottom in increasing order of their performance, as per Figure 8(a). A comparison of the various heuristics depicted in these figures may seem contradictory at first: max-degree seemingly outperforms min-domain due to its better fail-firstness and despite its worse promise (more mistakes), while min-dom/wdeg performs better than max-degree due to its better promise, and despite its slightly worse fail-firstness. Smith and Grant [17] showed that trying very hard to fail first does not always lead to better performance, and our experiments suggest a similar conclusion when it comes to promise. The fact that promise and fail-firstness cannot be independently controlled may cause improvements in one to have a negative impact on the other. It would be interesting to verify whether or not there is any correlation between promise and the change in the branching factor observed by Smith and Grant when trying heuristics with varying degrees of fail-firstness. Out of the group of heuristics studied here, min-dom/wdeg is the one that performs best not because it makes the smallest number of mistakes, or because it refutes them with less effort, but because it strikes a good balance between these two properties.

6 Conclusions

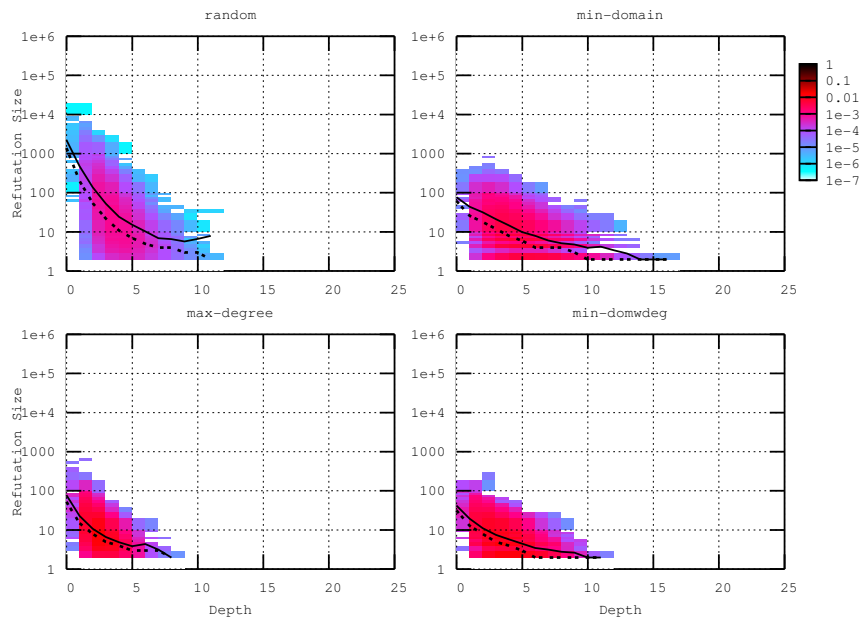
We have performed a detailed analysis of backtrack search for solving constraint satisfaction problems and, in particular, an in-depth study of the differences between variable and value ordering heuristics on a variety of problem classes. From a large population of instances of problems of different sizes, classes and levels of difficulty, we accumulated the effort required to refute each mistake encountered in each instance. Through this mistake-based analysis using heatmaps, in conjunction with more standard views of the search process, we have identified a number of interesting characteristics of search. For example, the effort required to recover from mistakes is not always correlated with the depth where they occur, and better search heuristics do not necessarily make fewer mistakes, or have the ability to recover quickly from them. There is a complex interaction between a heuristic's ability to avoid mistakes and its ability to recover from them. Our use of heatmaps is novel and of interest to researchers wishing to gain a deeper understanding of the relationship between search effort and heuristics.

Acknowledgements

This material is based on work supported by Science Foundation Ireland under Grant 00/PI.1/C075. We would like to thank Rick Wallace, Nic Wilson, Diarmuid Grimes, Barbara Smith, and Radoslaw Szymanek for their comments and suggestions, and to John Morrison and the Boole Centre For Research in Informatics for providing access to their Beowulf cluster.



(a) Probability of making a certain number of mistakes (y-axis) at a given depth (x-axis) in an instance. Medians as dotted line, averages as continuous line.



(b) Proportion of effort spent in refutations of a certain size (y-axis) at a certain depth (x-axis). Medians as dotted line, averages as continuous line.

Fig. 9: Sparse random problems 30×10 : Promise versus fail-firstness; four variable ordering heuristics, randomly selected values.

References

1. J.C. Beck, P. Prosser, and R.J. Wallace. Variable ordering heuristics show promise. In *Proceedings of CP-2004*, LNCS 3258, pages 711–715, 2004.
2. C. Bessière and J-C. Regin. MAC and combined heuristics: two reasons to forsake FC (and CBJ?) on hard problems. In *Proceedings of CP-1996*, LNCS 1118, pages 61–75, 1996.
3. C. Bessière, B. Zanuttini, and C. Fernández. Measuring search trees. In *ECAI-2004 Workshop on Modelling and Solving Problems with Constraints*, pages 31–40, 2004.
4. F. Boussemart, F. Hemery, C. Lecoutre, and L. Sais. Boosting systematic search by weighting constraints. In *Proceedings of ECAI-2004*, pages 146–150, 2004.
5. D. Brélaz. New methods to color the vertices of a graph. *Communications of the ACM*, 22(4):251–256, 1979.
6. R. Dechter. *Constraint Processing*. Morgan Kaufmann, 2003.
7. D. Frost and R. Dechter. Look-ahead value ordering for constraint satisfaction problems. In *Proceedings of IJCAI-1995*, pages 572–578, 1995.
8. I.P. Gent, E. MacIntyre, P. Prosser, B.M. Smith, and T. Walsh. Random constraint satisfaction: Flaws and structure. *Constraints*, 6(4):345–372, 2001.
9. C.P. Gomes, C. Fernández, B. Selman, and C. Bessière. Statistical regimes across constrainedness regions. *Constraints*, 10(4):317–337, 2005.
10. C.P. Gomes, B. Selman, N. Crato, and H. Kautz. Heavy-tailed phenomena in satisfiability and constraint satisfaction problems. *Automated Reasoning*, 24(1/2):67–100, 2000.
11. C.P. Gomes, B. Selman, and H.A. Kautz. Boosting combinatorial search through randomization. In *AAAI/IAAI*, pages 431–437, 1998.
12. R.M. Haralick and G.L. Elliott. Increasing tree search efficiency for constraint satisfaction problems. *Artificial Intelligence*, 14(3):263–313, 1980.
13. T. Hogg, B.A. Huberman, and C.P. Williams. Phase transitions and the search problem. *Artificial Intelligence*, 81(1-2):1–15, 1996.
14. T. Hulubei and B. O’Sullivan. Search heuristics and heavy-tailed behaviour. In *Proceedings of CP-2005*, pages 328–342, 2005.
15. C. Lecoutre, F. Boussemart, and F. Hemery. Backjump-based techniques versus conflict-directed heuristics. In *Proceedings of ICTAI-2004*, pages 549–557, 2004.
16. D. Sabin and E.C. Freuder. Contradicting conventional wisdom in constraint satisfaction. In *Proceedings of ECAI-1994*, pages 125–129, 1994.
17. B.M. Smith and S.A. Grant. Trying harder to fail first. In *Proceedings of ECAI-1998*, volume 14, pages 249–253, 1998.
18. R.J. Wallace. Heuristic policy analysis and efficiency assessment in constraint satisfaction search. In *CP-2005 Workshop on Constraint Propagation and Implementation*, 2005.