Trying Again to Fail-First

J. Christopher Beck[1], Patrick Prosser[2] and Richard J. Wallace[3]

[1] Department of Mechanical & Industrial Engineering, University of Toronto, Canada
[2]Department of Computer Science, University of Glasgow, Scotland
[3]Cork Constraint Computation Center and Department of Computer Science
University College Cork, Ireland
jcb@mie.utoronto.ca, pat@dcs.gla.ac.uk, r.wallace@4c.ucc.ie

Abstract. For constraint satisfaction problems (CSPs), Haralick & Elliott [1] introduced the Fail-First Principle and defined in it terms of minimizing branch depth. By devising a range of variable ordering heuristics, each in turn trying harder to fail first, Smith & Grant [2] showed that adherence to this strategy does not guarantee reduction in search effort. The present work builds on Smith & Grant. It benefits from the development of a new framework for characterizing heuristic performance that defines two *policies*, one concerned with enhancing the likelihood of correctly extending a partial solution, the other with minimizing the effort to prove insolubility. The Fail-First Principle can be restated as calling for adherence to the second, *fail-first* policy, while discounting the other, *promise* policy. Our work corrects some deficiencies in the work of Smith & Grant, and goes on to confirm their finding that the Fail-First Principle, as originally defined, is insufficient. We then show that adherence to the fail-first policy must be measured in terms of size of insoluble subtrees, not branch depth. We also show that for soluble problems, both policies must be considered in evaluating heuristic performance. Hence, even in its proper form the Fail-First Principle is insufficient. We also show that the "FF" series of heuristics devised by Smith & Grant is a powerful tool for evaluating heuristic performance, including the subtle relations between heuristic features and adherence to a policy.

1 Introduction

Search is at the heart of many AI approaches to problem solving. Despite this importance, there is no understanding at a foundational level of the behavior of heuristic decision making. The answer to the basic question "Why do some heuristics perform better than others?" remains elusive. One long-standing intuition for heuristic performance in Constraint Programming is the Fail-First Principle due to Haralick & Elliott [1] which states: "To succeed, try first where you are most likely to fail." Though initially counter-intuitive, the Fail-First Principle is widely seen as a useful insight into heuristic decision making. Given a set of inter-related decisions, the Fail-First Principle suggests that the one that is most difficult should be made first. This is akin to cautious intelligent behavior, focusing effort on critical choices before allowing ourselves the luxury of solving the easy parts of the problem.

In applying this principle to constraint satisfaction search, Haralick & Elliott made a further critical inference: that minimizing average branch length during search should

also minimize search effort. Because of this, their subsequent formal analysis of 'fail-firstness' did not pertain directly to discovering and solving difficult subproblems, but simply to finding the quickest way to fail during search. We will refer to this as the "radical Fail-First Principle" to distinguish it from the original idea. In this analysis, they showed that the "smallest domain first" (SDF) variable ordering heuristic (choose next to assign a value to the variable with the smallest number of possible values) is a level-one estimate of minimizing branch length.

To test this principle, Smith & Grant [2] created a set of new heuristics designed to aggressively fail early in the search. The hypothesis was that if failing quickly was the explanation for search efficiency, then heuristics that failed earlier should demonstrate lower search effort. Smith & Grant found that, contrary to expectations, increasing the ability to fail early in the search did not always lead to increased search efficiency. They concluded that the (radical) Fail-First Principle cannot be the only thing that explains differences in search cost among heuristics.

Our interest in this problem was sparked by the recent development of a new framework for analyzing heuristic performance [3] [4]. This framework incorporates *fail-firstness* as one of its performance principles. Another, called *promise*, concerns the selection of alternatives most likely to succeed. (This principle should be distinguished from the heuristics of the same name [5]; obviously, however, promise *heuristics* are designed to conform to the promise policy if and when the latter applies.) In this framework, if search deviates from a correct path, then and only then, does fail-firstness come into play.

In this paper, we review the work of Smith & Grant and then provide an overview of our policy-based framework for understanding search heuristics. We discuss work which corrects the experiments performed by Smith & Grant; although our results differ in some respects, like the earlier authors we find discrepancies between expectations based on the radical Fail-First Principle and relative performance of their heuristics. We also find that when forward checking is replaced with maintaining arc-consistency (MAC), the radical Fail-First Principle is *supported*. We then show that if fail-firstness is measured using the mean size of insoluble subtrees, adherence to promise and fail-firstness together can account for the behavior of heuristics when either forward checking or MAC is used.

2 Trying Harder to Fail First (1998)

The basic hypothesis of Smith & Grant was that if failing first is such a good thing then more of it must be better. Therefore, creating heuristics with a stronger ability to fail early should increase search efficiency. Efficiency was defined as the number of constraint checks required to find a solution to a problem or to prove that no solution exists. Because the focus of the study was on the relation between fail-firstness and search efficiency, the computational effort to make those heuristic decisions was (correctly) factored out of the experiments. The goal was to understand the relationship between the radical Fail-First Principle and search efficiency so the computational effort to increase the fail-firstness of a heuristic was irrelevant.

Experiments were performed over randomly generated binary constraint satisfaction problems. Each set of problems was defined by a 4-tuple $\langle n, m, p_1, p_2 \rangle$, where n is the number of variables, m is the uniform domain size, p_1 is the proportion of edges in the constraint graph, and p_2 is the uniform constraint tightness. All experiments were over problems with $n = 20$ and $m = 10$.

Using the forward checking algorithm [1] and standard chronological backtracking Smith & Grant tested four heuristics engineered for increasing levels of fail-firstness.

- **FF**: FF is the same as SDF (Smallest Domain First): choose the variable with the smallest remaining domain.
- **FF2**: The variable, v_i, chosen is the one that maximizes $(1 - (1 - p_2^m)^{d_i})^{m_i}$, where m_i is the current domain size of v_i, and d_i is the future degree of v_i. The FF2 heuristic takes into account an estimate (based on the initial parameters of problem generation) of the extent to which each value of v_i is likely to be consistent with the future variables of v_i. The FF2 heuristic has the flavor of the Brelaz heuristic [6], in that it trades off domain size and forward degree and favors differences in the former over the latter.
- **FF3**: FF3 builds on FF2 by using the current domain size of future variables rather than m. The variable, v_i, chosen is the one that maximizes the expression (1) below, where C is the set of all constraints in the problem, F is the set of unassigned variables, and $P = p_2$.
- **FF4**: Finally, FF4 modifies FF3 by using the current tightness, $P = p_{ij}$, of the future constraints (the fraction of tuples from the cross-product of the current domains that fail to satisfy the constraint) instead of p_2.

$$(1 - \prod_{(v_i, v_j) \in C, v_j \in F} (1 - P^{m_j}))^{m_i} \tag{1}$$

The progression from FF through to FF4 uses more and more information in determining which variable is most likely to fail at any given stage of search. Smith & Grant tested this by measuring the distribution of the depth of backtracks for problem instances at $\langle 20, 10, 0.5, 0.37 \rangle$. Their measurements showed that heuristics performed as expected: FF4 has a distribution skewed to backtracks at a shallower depth in search while the distribution gradually moved to deeper backtracks for FF3, FF2, and FF.

With respect to total search effort, Smith & Grant expected that the heuristics would be ranked as follows: FF $>$ FF2 $>$ FF3 $>$ FF4, where $>$ means "results in greater search effort than". Their results were not as expected. Through a number of experiments Smith & Grant showed that except on easy problems, FF2 incurred the least search effort, followed by FF3, FF and finally FF4. That is, they observed the following order: FF4 $>$ FF $>$ FF3 $>$ FF2. This ordering is clearly at odds with the hypothesis of a simple mapping between the ability to fail-first and search effort. They concluded that there must be some other factor at work, perhaps in concert with the Fail-First Principle, in determining search efficiency for variable ordering heuristics.

3 A Framework for Understanding Search

3.1 Policies and Heuristics

The present work is informed by a recently developed framework for characterizing the performance of search heuristics. This framework has two primary elements. A *policy* identifies goals or end-results that are desirable. A *heuristic* is a rule that is followed to make a decision.

For search problems, there is an overall policy of minimizing search effort. This is normally measured by counting nodes or constraint checks in the search tree. Of greater interest is that two subordinate policies can be distinguished in the search domain. When search is in a state that has solutions in its subtree, search effort will be minimized by making decisions to remain on a path to a solution. As this suggests making decisions to move to the most promising subtree, we call this the *promise policy*. However, for hard problems the best choice will not be made in all cases and search may enter a state where the subtree below it does not contain any solutions. In this case, to minimize effort search should fail as quickly as possible so it can return to a path that leads to a solution. We call this the *fail-first policy*.

Heuristics are based on features of the situation that serve to distinguish choices, so that a selection in these terms increases the likelihood of achieving a goal. In CSP search, these are the variable and value ordering "rules" that exist in the constraint literature (e.g.smallest domain first, brelaz [6], domdeg [7]). The intuition behind variable ordering heuristics is the Fail-First Principle mentioned above, while that for value ordering is related to promise. However, recent work, which has shown how to evaluate variable ordering heuristics in terms of promise, indicates that this policy must also be taken into account in any full evaluation of these heuristics [3] [4].

The contribution of heuristic decisions to performance should depend on how well the heuristic conforms to either subordinate policy. We would expect that adherence to the promise policy will make a difference to search for problems with many solutions. As problems become more difficult, the proportion of time exploring bad subtrees becomes greater, so that the fail-first policy is more often in force and fidelity to that policy should be more important. If problems have no solutions, then the only policy relevant to search effort is fail-first.

3.2 The Fail-First Principle

Within the policy framework the Fail-First Principle states that when we are uncertain as to which policy to adhere to (i.e. we do not know if the current search node is good or bad), we should try to adhere to the fail-first policy. Stated in these terms, the Fail-First Principle is a kind of high-level heuristic for selecting a policy to adhere to under conditions of ignorance, where one does not know what the appropriate policy actually is. As such, it is a conservative mini-max principle that tries to minimize worst case effort by aggressively seeking to fail. Adherence to this principle also implies that heuristics should be evaluated in terms of how well they conform to the fail-first policy alone.

With this restatement of the principle, we can see more clearly that there may be important limits to its range of application, and that for certain problems or conditions

of search it may lead us badly astray. Moreover, there is the assumption in all of this that the promise policy is irrelevant to variable selection. As we have shown, this is not so: an adequate evaluation of ordering heuristics must give some consideration to both policies rather than the fail-first policy alone. We have also shown, perhaps surprisingly, that heuristics such as SDF which were designed to have a high degree of fail-firstness [1] also show a high degree of promise. This means that for problems with solutions, a correlation between fail-firstness and decreased search effort is not sufficient evidence that this factor is critical for differences in performance, since the impact of this factor has not been disentangled from that of promise.

3.3 Measuring the Ability to Fail-First as Branch Depth Minimization

Within this framework we can also restate the Smith & Grant strategy: by devising heuristics to conform in increasing degree to the fail-first policy, we can evaluate the sufficiency of the Fail-First Principle. However, to evaluate sufficiency we must be able to measure fail-firstness, i.e. the degree of adherence to the policy. For now we will continue with the assumption made by Haralick & Elliott and by Smith & Grant that fail-firstness is adequately characterized by branch depth.

For their evaluation, Smith & Grant measured the mean *backtrack depth* to find a solution to a problem or to prove that no solution exists. We believe there are two weaknesses in this methodology. First, a backtrack is counted whenever a domain is emptied and search returns to the previous variable. If that variable has no more values to try, its domain has also been emptied and another backtrack is counted in moving back once again. That is, Smith & Grant measured the average depth of failed leaf nodes *and* failed interior nodes of the search tree explored. The original formulation of the radical Fail-First Principle assumed that minimizing mean *branch depth* would minimize search effort. Therefore, it should be the mean branch depth that is used as a measure of the ability of a heuristic to fail-first. The backtrack depth measure does not do this. A more appropriate measure of the mean branch depth is as follows: whenever a variable is assigned a value and that assignment immediately leads to a domain wipe-out, we count a failure. That is, we measure the average depth of failed leaf nodes in the search tree.

There is an alternative way of measuring branch depth: calculate the difference between the depth of a failed leaf and the depth of the initial mistake that led to the failure. Although this seems like an even more precise measure, it suffers from the effect of a varying ceiling: the largest possible difference is greater when mistakes occur higher in the search tree. So this measure was not used for the initial experiments, although data will be presented in Section 5.

The second weakness with the earlier measurement of a heuristic's ability to fail-first is that by searching only for the first solution to soluble problems the measure of the ability to escape a mistake is contaminated by the heuristic's ability to find a solution (its promise, in the newer formulation). It is at least theoretically possible that a heuristic that is very poor at failing first could be very good at finding a solution when one exists. Therefore, searching for the first solution combines the abilities of a heuristic to escape dead-ends and its ability to find solutions. We are interested in isolating the former ability. For our experiments, we assess the ability of a heuristic to fail-first by

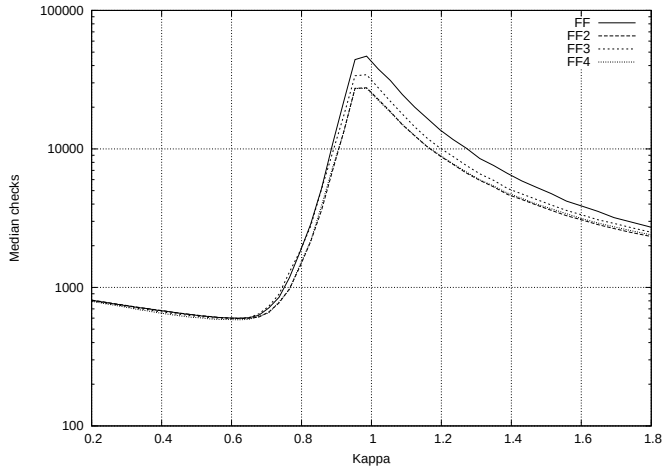

Fig. 1. The median number of checks for the $\langle 20, 10, 0.5 \rangle$ problem set. The heuristics are ranked as follows: FF $>$ FF3 $>$ FF4 $\approx$ FF2.

measuring the branch depth in searching for *all* solutions or showing that no solution exists. (Of course, performance assessment is still based on the work required to find the first solution.)

4 Empirical Investigations

In this section, we report the results of three experiments. The first two explore the relationship between fail-firstness and heuristic quality. They were reported in part by Beck et al. [8], and are included here (with further data) for completeness. The conclusion of the first experiment is that, as Smith & Grant originally found, fail-firstness as measured by branch depth is not equivalent to heuristic quality. The third experiment, therefore, investigates whether promise is the missing factor.

4.1 Fail-first with forward checking

As reported earlier [8], we repeated the experiments of Smith & Grant. Our results were produced by using two solvers coded independently, and we also confirmed these results using the C++ solver of Smith & Grant with an error in FF4 corrected[1]. In our implementation of the heuristics, ties (when more than one variable is judged heuristically best) are broken lexicographically. The problems we investigated were generated at the beginning of our study and stored. They were then used by all our solvers, allowing us to reproduce results across two sites.

Problems were generated using a "probability-of-inclusion" model, in which each possible constraint element (domain value, constraint or constraint tuple) is included

[1] The value of p_{ij} was computed incorrectly for the FF4 heuristic in [2]

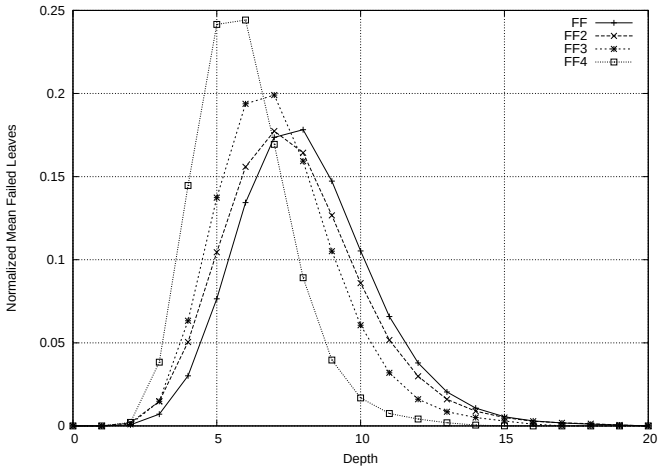

Fig. 2. The fraction of failed leaves at each depth in the tree when searching for all solutions for 1000 problems in the $\langle 20, 10, 0.5, 0.37 \rangle$ set, i.e. the problems at the $\langle 20, 10, 0.5 \rangle$ phase transition.

with a specified probability. The generator allows parameters to be fixed at the expected values given these probability values; in this case a set of elements is generated repeatedly until the cardinality matches the expected value. This allows it to generate problems in accordance with model B [9]. By specifying a probability of 1 for domain element inclusion, all domains have a size specified by a maximum domain-size parameter. Sets of soluble or insoluble problems were sometimes used; these problems were generated in an identical fashion and filtered on the basis of solution testing.

Figure 1 presents the median number of constraint checks to find a solution or to prove that no solution exists for problems from the set $\langle 20, 10, 0.5 \rangle$. The constraint tightness was varied from 0.01 to 0.99 in steps of 0.01. For each combination of parameters 1000 problem instances were generated. Median checks (the same measure used by Smith & Grant) are plotted against κ (kappa), the measure of constrainedness proposed by [10]. Similar results were found for other problem series [8].

In these experiments, FF clearly incurs the highest number of constraint checks, followed by FF3. In Figure 1, there is no discernible difference between FF2 and FF4. Aside from the FF4 results, these graphs agree with the original experiments of Smith & Grant.

In Figure 2 we plot the fraction of failed leaves at each depth when searching for all solutions. Qualitatively, these results match what Smith & Grant found with their measure of the ability to fail-first: FF4 does indeed fail higher in the tree (as judged by mean branch depth), followed by FF3, FF2 and FF. Therefore, we can confirm that Smith & Grant did indeed propose new heuristics that progressively increase in their ability to minimize branch depth.

The crux of these experiments is the fact that the ability to fail earlier in the tree does not necessarily translate into better search effort; FF3 incurs a higher search cost than both FF2 and FF4, yet Figure 2 shows that FF3 is between FF2 and FF4 in its ability

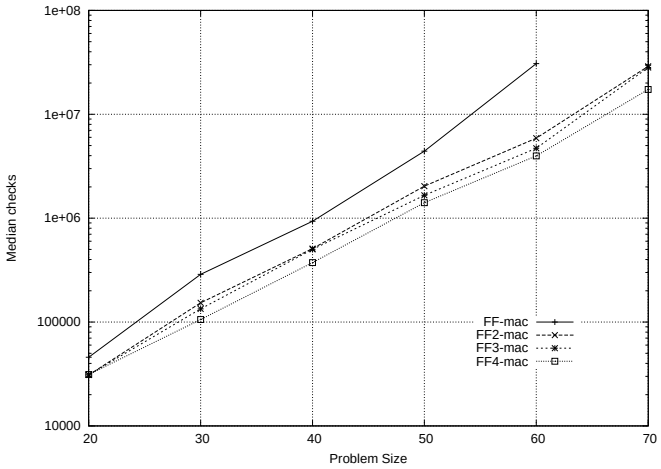

Fig. 3. Median consistency checks for problems with 20 to 70 variables using MAC. All problems have 10 values per variable, a density that results in 10 constraints on each variable, and a tightness set to give $\kappa \approx 0.9$. All problems are soluble and each problem size has 50 instances.

to fail early in the tree. These results have been extended to larger problems, where the ordering with respect to search effort is clearly FF $>$ FF3 $>$ FF2 $>$ FF4 [8].

Thus, even after fixing the errors and improving the measurement of fail-firstness in terms of branch depth, the results do not entirely conform to expectations based on the Fail-First Principle as it was originally formulated. This confirms the main conclusion of Smith & Grant: the radical Fail-First Principle (i.e., branch depth minimization) is not sufficient to account for all variations in search efficiency among variable ordering heuristics.

4.2 Tests with MAC

In the course of this work we also wanted to determine whether the consistency enforcement algorithm might have an effect on the heuristics' adherence to the fail-first policy, and hence the viability of the Fail-First Principle under different degrees of consistency maintenance. To test this we repeated the above experiments, varying problem size, but this time using the maintaining-arc consistency algorithm (MAC) [11]. As the name implies, whenever a variable is instantiated the future sub-problem is made arc-consistent. If this results in a domain wipe-out, a new value is tried, and failing that, backtracking takes place.

For 20-variable problems, there is little variation in fail-depth among heuristics, although the differences that appear at different depths are consistent with the pattern found for forward checking. For this reason we present data on larger problems where a clear pattern of differences emerges.

The results for search efficiency are presented in Figure 3. Interestingly, the ranking of heuristics is different than for FC. We now have the order FF $>$ FF2 $>$ FF3 $>$

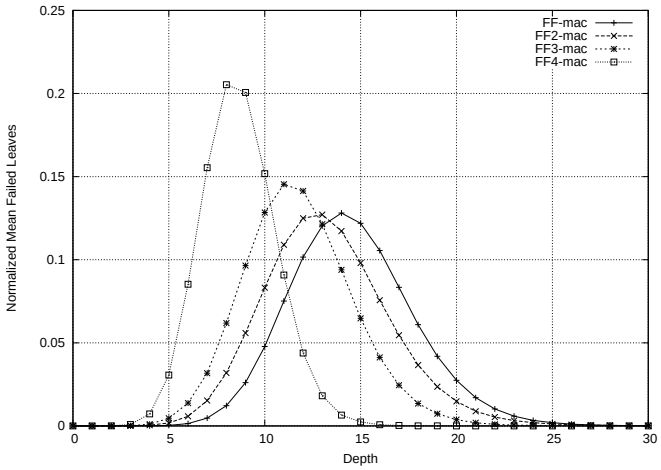

Fig. 4. Distribution of branch depth for soluble problems with 60 variables (parameters $\langle 60, 10, 0.153, 0.369 \rangle$). Qualitatively similar results are seen for insoluble problems.

FF4. The results in Figure 3 are for soluble problems, and though not shown, the same ranking was found for insoluble problems. Figure 4 shows the failure depths of the heuristics when allied with the MAC algorithm, for one set of large problems ($n = 60$). The results are in agreement with those in Figure 2 with FF4 failing earliest, then FF3, FF2, and FF.

Thus, we find that for MAC, differences among heuristics are entirely in line with expectations based on the radical Fail-First Principle. Adherence to the principle appears to be algorithm dependent.

4.3 The Impact of Promise

We have already noted that the efficiency of variable ordering heuristics can be related to promise as well as fail-firstness. It was therefore of interest to see whether FF heuristics could be distinguished on this basis. Figure 5 shows the results of "probe" tests, which can be used to provide unbiased estimates of promise [3]. We find that with forward checking FF3 and FF4 are distinctly inferior to FF and FF2 on this basis.

However, before we conclude that differences in promise are the whole explanation for the order of search effort, we must consider insoluble problems. As noted earlier, promise is only well-defined on problem instances with solutions. If the results on search effort are due to a combination of promise *and* fail-firstness, then the relative ranking of heuristics should differ for soluble and insoluble problems. We therefore refine Smith & Grant's hypothesis and instead hypothesize that on *insoluble* problems as we try harder to minimize branch depth we will reduce search effort.

Figure 6 demonstrates that this hypothesis is false. The relative ordering of the heuristics on insoluble problems for the $\langle 20, 10.0.5 \rangle$ problem set is identical to the ordering on mixed problems. Though not shown, this ordering is the same for the sol-

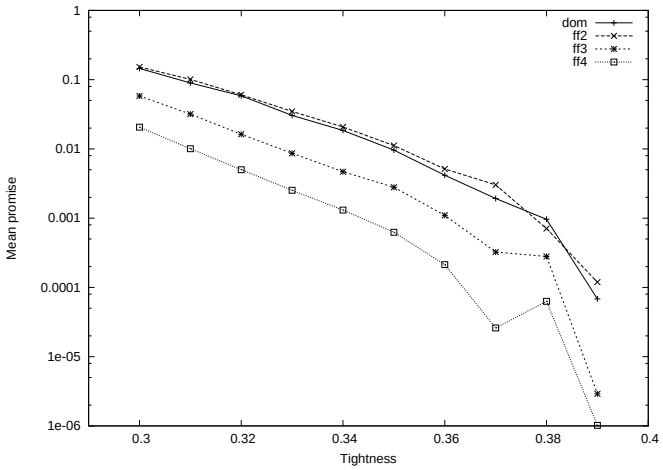

Fig. 5. Promise measurements for the FF heuristics with forward checking, obtained for soluble <20,10,0.5> problems.

uble problems as well. Furthermore, the branch depth on insoluble (as well as soluble) problems follows the pattern of Figure 2: FF4 fails highest followed by FF3, FF2, and FF. Therefore promise cannot be the entire explanation for the discrepancies in search effort. (As we will see shortly, it would be premature to conclude that it does not play a role in the case of soluble problems.)

4.4 Conclusions

Three conclusions are drawn from the experiments above:

1. For forward checking, minimizing mean branch depth does not wholly account for heuristic efficiency.
2. For forward checking, minimizing mean branch depth combined with adherence to the promise policy does not wholly account for heuristic efficiency.
3. For MAC, minimizing mean branch depth does appear to account for heuristic efficiency.

A model of heuristic performance needs to explain each of these results. From the perspective of our policy framework, there appear to be two logical options. Either there is an additional policy that accounts for the discrepancies or the measurements that we have proposed for promise and fail-firstness are deficient. For the balance of this paper, we examine the second option, concentrating on the fail-firstness measurement.

5 The Proper Measure of Fail-Firstness

The fail-first policy is to minimize the effort required to determine that an insoluble subtree is indeed insoluble. This suggests that the average size of insoluble trees would

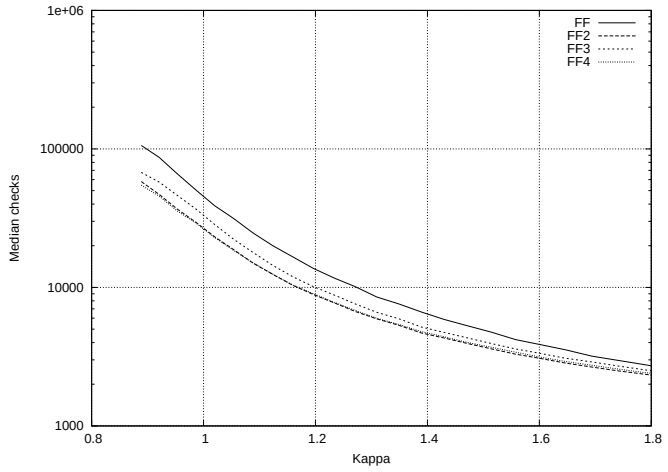

Fig. 6. The median number of constraint checks for the insoluble problem instances in the $\langle 20, 10, 0.5 \rangle$ problem set. The problem sets plotted are those for which at least one instance is insoluble, therefore each κ value may have a different number of instances up to 1000. In particular, the low values of κ are based on few instances.

give a more accurate measure of fail-firstness than measures of branch depth. Something like this was suggested by Nudel [12], who argued that the minimum domain size (FF) heuristic worked by minimizing the size of failed subtrees. Failed subtree size takes into account the branching factor in the search tree. As Smith & Grant point out, Russell & Norvig [13] suggested that successful heuristics such as Brelaz and FF can be justified by the minimization of the branching factor. However, the fail-first policy suggests that a combination of this and branch depth may be critical, and this is more likely if there is a tradeoff between the two.

To examine failed subtree size, a more detailed assessment of search characteristics was performed using the 50-variable problems from the series tested earlier ([8] and Section 4.2; their parameters were $\langle 50, 10, 0.184, 0.369 \rangle$). (Similar results were also obtained in tests on 20-variable problems of varying tightness.) In addition to mean fail-depth, we calculated the following measures:

1. Mean fail-length: the difference in search depth between the initial invalid assignment, or "mistake" and the detection of failure
2. Mean "mistake-tree size" (mistksz): the number of nodes in an insoluble subtree, rooted at the initial invalid assignment
3. Number of failures
4. Number of mistake-trees
5. Mean selected domain size ($|\ d\ |$): the domain size of the variable which is selected for assignment. This is a direct measure of the branching factor.

The basic summary results are shown in Table 1 for each heuristic for forward checking and MAC. These results must be considered in tandem with similar measures

that take the depth of the initial mistake into account. This is because of the problem mentioned earlier, that intensity measures like fail-length and mistake-tree size involve ceiling effects related to the location of a mistake in the search tree. For example, if a mistake is made at level n-1, then the maximum size of the mistake-tree is 1; this means that if, for a given heuristic a large proportion of the mistakes occur at this level the average mistake-tree size will be small because of the low ceiling in these cases. Table 2 displays the fail-length and mistake-tree size at different mistake depths.

In both tables and for each subcolumn the statistical significance of differences between entries was assessed by ANOVAs and post-hoc comparisons between pairs of means that were closest in value [14]. Analyses for Table 2 were based on problems for which there were no zero frequencies for any heuristic. (Because of the number of zero frequencies, analyses were not carried out beyond level 20 for FC and level 10 for MAC.) For all ANOVAs, the F statistic was statistically significant at $p < .05$ or better (typically $\leq 10^{-8}$). The few comparisons between adjacent means that were not statistically significant are indicated in the tables.

There are a number of important results in these tables.

- The ordering of fail-lengths matches that of fail-depths (Table 1), and the same ordering of fail-lengths is found at every mistake depth (Table 2). Since neither measure of branch depth corresponds to the order of search effort, this supports the conclusion that measures of branch depth are inadequate measures of fail-firstness.
- Further evidence on this point comes from a comparison of fail-length and mistake-tree size at mistake depth 1. These results are directly comparable to those with insoluble problems, since in the latter case all mistakes occur at this level. (This has been verified directly on insoluble problems.) In these cases fail-first is the only relevant policy. Here we see that the order of mistake-tree size matches the order of search effort (cf. Figure 6), while the fail-length order does not. This implies that any measure of fail-firstness must be based on the size of insoluble subtrees.
- For these soluble problems, FF3 with forward checking produces more mistake-trees (and consequent failures) than any other heuristic (Table 1). This reflects a relative failure to adhere to the promise policy, so poor performance of FF3 in this case must be partly due to this factor.
- With forward checking, differences in mistake-tree size vary as a function of mistake depth (Table 2); as depth increases, FF3 becomes better than FF2. This shows that relative adherence to a policy by different heuristics can vary at different levels of search.

We conclude that the difference from expectation found for FF3 in the first three experiments is related to deficiencies with respect to both policies. For soluble problems, the fail-first deficiency occurs at the top of the search tree, but there is also a deficiency in promise, reflected in the number of mistakes made by this heuristic. For insoluble problems, since fail-first is the only relevant policy, the deficiency is restricted to this factor. A more general conclusion is that when fail-firstness is properly measured, the order of search effort obtained by these heuristics can be characterized by degree of adherence to one or both policies. Finally, we have shown that valid comparisons of fail-firstness must be made at the same mistake depth. In fact, there are important dif-

ferences among heuristics with respect to fail-firstness that are a function of mistake depth (as well as algorithm type).

Table 1. Failure measures and branching factor for fail-first heuristics

	faildepth	fail-length	#fails	mistksz	#mstktrs	$\lvert d \rvert$
			forward checking			
FF	19.9	17.5	600,303	6679	1662	1.35
FF2	18.3	15.9	103,365	1799	539	1.36
FF3	16.6	13.8	1,030,002	644	10,914	2.25
FF4	13.7	10.9	289,466	339	4885	2.18
			MAC			
FF	12.3	10.0	59,410	786	324	2.09
FF2	11.3	8.9	13,662	247	175	1.64
FF3	10.1	7.8	14,056	203	235	1.37
FF4	7.6	5.4	9,401	119	193	1.25

Note. Values are means per problem. 50-variable problems with solutions. "#fails" is number of times an assignment led to a domain wipeout. "mistksz" is mistake-tree size; this and other measures are defined in text."$\lvert d \rvert$" is mean domain size of variables selected for instantiation. In this and Table 2 nonsignificant comparisons are flagged by underlining the smaller value.

Table 1 suggests that the branching factor plays a significant role in the relative efficiency of these heuristics. With forward checking, FF3 and FF4 each had a higher branching factor than the other heuristics, and it was here that inversions in the expected ordering occurred. (In addition to the means shown in the table, the *range* of domain sizes of the variables selected was much greater with FF3 and FF4, so that sizes up to 10 were chosen even at moderate depths of the search tree.) With MAC, in contrast, the branching factor diminished from FF through FF4, and in this case search effort followed the original expectations. We hypothesize that, with forward checking, the greater branching of FF3 overwhelms the modest improvement in branch depth, which leads to inferior search performance in comparison with FF2.

That it is a tradeoff between branch depth and the branching factor can be seen by comparing the FF and FF4 results for forward checking. If degree of branching were the only factor, FF should incur less search effort than FF4. As shown by all our experiments, this is not the case.

6 Conclusions

This work serves both to clarify the Fail-First Principle and to indicate its limitations. This has been done by using a new framework in which heuristics are evaluated in terms of how well they adhere to either of two basic policies, called promise and fail-first, one of which is in force at any point in search.

In order to evaluate variable ordering heuristics in these terms we have developed measures of adherence to a given policy. For promise, this was done in earlier work [3]

Table 2. Failure measures for fail-first heuristics at different depths of search tree

	mistake depth (nodes)					
	1	5	10	15	20	30
			FC			
			fail-length			
FF	18.2	15.5	11.8	9.6	7.7	4.0
FF2	16.4	14.2	11.2	8.3	5.9	2.1
FF3	14.4	11.8	9.3	6.2	4.5	0.8
FF4	11.6	8.8	5.8	2.7	1.2	0.0
			mistake-tree size			
FF	145,015	4666	219	52	21	10
FF2	24,808	1549	150	27	12	3
FF3	80,484	3154	188	22	8	2
FF4	21,813	678	35	5	2	1
			MAC			
			fail-length			
FF	10.6	7.9	3.9	1.7	0.5	0.0
FF2	9.6	6.6	3.3	1.7	0.9	0.0
FF3	8.3	5.5	2.2	0.7	0.2	0.0
FF4	6.0	3.2	0.4	0.1	0.0	–
			mistake-tree size			
FF	7080	219	12	3	1	1
FF2	1903	94	10	3	2	1
FF3	1489	63	6	2	1	1
FF4	865	25	2	1	1	–

Note. Values are means per problem for each mistake depth. “Mistake depth” is depth at which an initial assignment was made to produce an insoluble subtree.

[4]. In the present paper, we have done this for fail-firstness. An important contribution of this work is to demonstrate that fail-firstness cannot be adequately assessed by measures of branch depth; instead the size of the insoluble subtree rooted at the initial mistake must be calculated.

Within this framework, we can restate the Fail-First Principle as a metaheuristic proposal to act as if adherence to the fail-first policy is the only important consideration. Having cleared up the question of how to measure such adherence, the original conclusion of Smith & Grant, that the radical Fail-First Principle that was proposed by Haralick & Elliott is inadequate, is of course confirmed, since this form of the principle is based on an incorrect measure of fail-firstness. At the same time, we have shown that Smith & Grant’s assumption that their fail-first series of heuristics shows increasing adherence to this policy is not entirely correct, since with forward checking average mistake-tree size is sometimes greater for FF3 than for FF2, and is always greater for insoluble problems. Finally, we have shown that even when put in its proper form, the Fail-First Principle is not an entirely reliable guide (cf. results in Table 1, showing that average mistake-tree size for FF3 is less than FF2); this is because variation in adher-

ence to the promise policy can be more important than variation in adherence to the fail-first policy.

Clearly, features such as the branching factor determine whether a heuristic adheres to either policy or to both. What does the policy framework give us in addition? We think the answer is that it gives us a way of characterizing the effects of heuristic features on performance in terms of basic features of search itself. In addition, the present work shows that we can evaluate performance in these terms without knowing the features of the heuristic that affect its adherence to a policy. Finally, we know from the present work as well as the original study of Smith & Grant that there is no simple relation between heuristic features and adherence to a policy, and there is no guarantee that improved adherence to one policy will result in improved adherence to the other. These seem to be sufficient reasons for making the distinctions that the new framework requires.

Acknowledgment. This work was supported by Science Foundation Ireland under Grant 00/PI.1/C075 and ILOG S.A. We thank Diego Moya for some of the coding and especially Barbara Smith for giving us access to source code and supporting our study.

References

1. Haralick, R.M., Elliott, G.L.: Increasing tree search efficiency for constraint satisfaction problems. Artificial Intelligence **14** (1980) 263–314
2. Smith, B.M., Grant, S.A.: Trying harder to fail first. In: Proc. Thirteenth European Conference on Artificial Intelligence-ECAI'98, John Wiley & Sons (1998) 249–253
3. Beck, J.C., Prosser, P., Wallace, R.J.: Toward understanding variable ordering heuristics for constraint satisfaction problems. In: Proc. Fourteenth Irish Artificial Intelligence and Cognitive Science Conference-AICS'03. (2003) 11–16
4. Beck, J.C., Prosser, P., Wallace, R.J.: Variable ordering heuristics show promise. In: Principles and Practice of Constraint Programming-CP'04. LNCS No. 3258. (2004) 711–715
5. Geelen, P.A.: Dual viewpoint heuristics for binary constraint satisfaction problems. In: Proc. Tenth European Conference on Artificial Intelligence-ECAI'92. (1992) 31–35
6. Brelaz, D.: New Methods to Color the Vertices of a Graph. Communications of the ACM **22** (1979) 251–256
7. Gent, I., MacIntyre, E., Prosser, P., Smith, B., Walsh, T.: An empirical study of dynamic variable ordering heuristics for the constraint satisfaction problem. In: Principles and Practice of Constraint Programming-CP'96. LNCS No. 1118. (1996) 179–193
8. Beck, J.C., Prosser, P., Wallace, R.J.: Failing first: An update. In: Proc. Sixteenth European Conference on Artificial Intelligence-ECAI'04. (2004) 959–960
9. Gent, I.P., MacIntyre, E., Prosser, P., Smith, B.M., Walsh, T.: Random constraint satisfaction: Flaws and structure. Constraints **6** (2001) 345–372
10. Gent, I.P., MacIntyre, E., Prosser, P., Walsh, T.: The constrainedness of search. In: Proc. Thirteenth National Conference on Artificial Intelligence-AAAI'96. (1996) 246–252
11. Sabin, D., Freuder, E.: Contradicting Conventional Wisdom in Constraint Satisfaction. In: Proc. Eleventh European Conference on Artificial Intelligence-ECAI'94, John Wiley & Sons (1994) 125–129
12. Nudel, B.: Consistent-labeling problems and their algorithms: Expected-complexities and theory-based heuristics. Artificial Intelligence **21** (1983) 263–313
13. Russell, S.J., Norvig, P.: Artificial Intelligence: A Modern Approach. Prentice-Hall (1995)
14. Hays, W.L.: Statistics for the Social Sciences. 2 edn. Holt, Rinehart, Winston (1973)