Genetic Programming and Evolvable Machines manuscript No.
(will be inserted by the editor)

Better GP Benchmarks: Community Survey Results and Proposals

David R. White, James McDermott, Mauro Castelli, Luca Manzoni, Brian W. Goldman, Gabriel Kronberger, Wojciech Jaśkowski, Una-May O'Reilly, and Sean Luke

Received: date / Accepted: date

Abstract We present the results of a community survey regarding genetic programming (GP) benchmark practices. Analysis shows broad consensus that improvement is needed in problem selection and experimental rigor. While views expressed in the survey dissuade us from proposing a large-scale benchmark suite, we find community support for creating a "blacklist" of problems which are in common use but have important flaws, and whose use should therefore be discouraged. We propose a set of possible replacement problems.

Keywords Genetic programming · benchmarks · community survey

David R. White, david.r.white@glasgow.ac.uk
School of Computing Science, University of Glasgow, UK.

James McDermott, jmmcd@jmmcd.net
School of Business and Complex and Adaptive Systems Laboratory, University College Dublin, Ireland.

Mauro Castelli, mcastelli@isegi.unl.pt
Instituto Superior de Estatística e Gestão de Informação (ISEGI), Universidade Nova de Lisboa, Portugal.

Luca Manzoni, luca.manzoni@disco.unimib.it
Dipartimento di Informatica, Sistemistica e Comunicazione, University of Milano-Bicocca, Italy.

Brian W. Goldman, brianwgoldman@acm.org
BEACON Center for the Study of Evolution in Action, Michigan State University, USA.

Gabriel Kronberger, gabriel.kronberger@heuristiclab.com
University of Applied Sciences Upper Austria, Austria.

Wojciech Jaśkowski, wjaskowski@cs.put.poznan.pl
Institute of Computing Science, Poznan University of Technology, Poland.

Una-May O'Reilly, unamay@csail.mit.edu
CSAIL, Massachusetts Institute of Technology, USA.

Sean Luke, sean@cs.gmu.edu
Department of Computer Science, George Mason University, USA.

1 Introduction

This paper discusses the results of a survey of the Genetic Programming community concerning the current state of benchmarks as used in the field, suggests that several problems in common use should be blacklisted, i.e. discouraged from further use, and proposes some possible replacements. The survey arose after a spirited discussion at GECCO 2012 following presentation of the position paper *Genetic Programming Needs Better Benchmarks* [32]. That paper listed the many problems that have been used as benchmarks, summarized recent benchmark practice, argued that improvements were needed, and described the potential advantages of a standardized benchmark suite. It also identified the importance of community involvement in any debate or standardization effort. Audience discussion during the paper's presentation broadly supported the project, with some reservations.

The central goal of the GP Benchmarks Project[1] is to identify and help to implement benchmarking improvements needed in GP. It aims to be community-driven. The long and diverse author lists of the GECCO paper and the current paper reflect the goal of being led by the opinions of the whole community rather than those of a small in-group. The first step is to identify *whether* the community feels improvements are needed, and if so *what* they are. This was the purpose of the survey whose results are detailed here.

Although the creation of a standardized benchmark suite is one possible outcome of this process, that only becomes a goal insofar as it reflects a consensus among the community. The possible roles of the project include: focusing discussion, and providing a forum for it; curating documents, code, and data; raising awareness of benchmark issues; and gathering and publishing data on community attitudes. It is not the role of the project to carry out original research such as developing new benchmarks. We believe that individual researchers with expertise in particular areas of GP are best-qualified to carry out that task.

The creation of a standard suite has been recognized as an important issue for the future of GP [35]. Of course, common test problems have always existed in GP. Some of the problems introduced by Koza [28, 29] have become very widely used. Two notable efforts at creating updated benchmark suites were *GP-Beagle* [7] and the *Evolutionary Computation Benchmarking Repository* (EvoCoBM) [43], but neither is now active.

In Section 2, next, we present the methodology behind the survey and its results. In Section 3 we include an extended review of recent benchmark usage in the GP literature. In Section 4 we summarize the issues and present our rationale for the next steps: in Section 5 we propose a "blacklist" of benchmarks that should not be used, and in Section 6 we provide a list of possible replacements.

[1] `http://gpbenchmarks.org/`

2 Community Survey

The goal of the survey was to gather the opinions of the community on benchmarking practice and related matters. We developed the key areas of interest in the survey through discussion on the GP mailing list, and within the GP benchmarks special interest group. The areas of opinion solicited are given in Table 1.

	Category	Survey Questions
1.	Demographics	24, 25, 26, 27, 28
2.	Current empirical practice in GP	1, 2, 3, 4, 7, 8
3.	Problems with current practice	5, 6
4.	Other benchmarking efforts	9, 10, 11
5.	The potential creation of a new benchmark suite	12, 13, 14, 15, 16
6.	Composing a new benchmark suite	17, 18, 19, 20, 21, 22, 23
7.	Other comments and email address (optional)	29, 30

Table 1 Areas of interest for our survey and their corresponding questions.

There were 30 questions in total. The majority were multiple-choice or box-checking questions, some with free text options. Five questions required only free text responses. No questions were marked as mandatory. The survey questions, raw quantitative data, and some summary data are available for download from `http://gpbenchmarks.org`.

2.1 Notes on Survey Methodology

The survey was designed by two GP practitioners, with the assistance of experienced survey designers from other Computer Science disciplines. Several rounds of testing and review were completed with colleagues of the authors to pilot the survey. The final survey was opened on July 9th 2012 and closed on July 31st 2012. The survey was implemented online, via the SurveyMonkey questionnaire web service. It was advertised via the GECCO presentation, the GP mailing list, Twitter, and a handful of personal emails to practitioners within the field.

There were a total of 15 opportunities for free text answers, including the length of experience the respondent had and their email address. To facilitate analysis and anonymization, these answers were subsequently coded by the survey designers. *Coding* [41] is the standard practice of categorizing fragments of qualitative text into abstract identifiable groups of response types. A *grounded theory* approach was used [45], meaning that the response types were extracted from the text, rather than beginning with an *a priori* list. Response types were organized into tree-structures, one per question. The coding was carried out independently by two researchers and then differences were resolved through a process of discussion and refinement. These methods are standard practice [41]. The resulting codings are included online with the rest of the survey results.

Fig. 1 Question 26: "For how many years have you worked in or studied GP?" 5 respondents skipped this question.

2.2 Results

In total 79 responses were received. The following subsections describe the results from each of the categories set out in Table 1. Responses to Question 29 ("any other comments") are placed according to the topic they refer to, and responses to Question 30 (email addresses) are excluded.

2.2.1 Demographics

Figure 1 shows the length of time respondents had been working in the GP field. The two main channels through which respondents were recruited to the survey were the GECCO presentation itself (29%) and GP mailing list (37%). Of all the respondents, 37% had attended the GECCO presentation, and 57% had read the GECCO paper. Almost all (89%) claimed to have medium to high expertise in GP benchmark issues.

2.2.2 Current Empirical Practice

Question 1 asked "In your work on Genetic Programming, do you run experiments to measure algorithm performance?" Over 80% of practitioners responded that they performed such measurements in every or almost every paper, which underscores the highly empirical nature of the field. Only a few respondents never produce algorithm performance measurements.

The types of problems used by respondents (Question 2) are detailed in Figure 2. Around 70% used well-known GP benchmarks, thus demonstrating that *de facto* benchmarks have emerged in the absence of any formally established suite. Two thirds of respondents used real-world problems, while large numbers also used self-developed problems, problems from other fields, or more atypical GP problems. A few respondents mentioned competitions, UCI datasets, randomly-generated benchmarks, and other sources.

Question 3 asked "When running experiments to measure performance of a new technique, do you use existing techniques as controls?" Most respondents

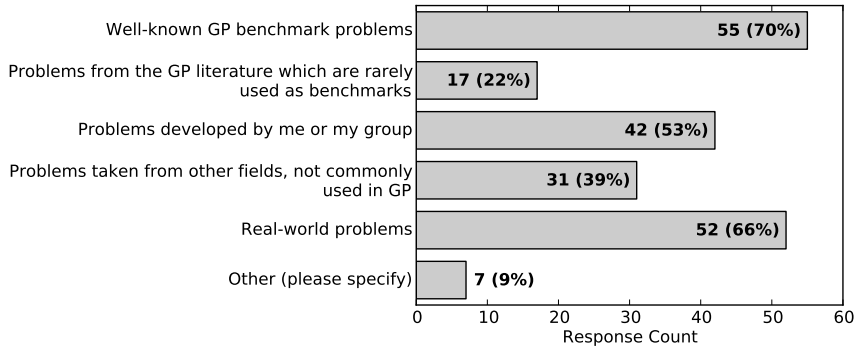


Fig. 2 Question 2: "What types of problems do you use? Please check all that apply."

(63%) said they run new experiments with existing techniques, but some (19%) rely on previously published results for comparison. Around 8% of respondents do not use controls. The remainder failed to answer or reported that the question was not applicable to their work.

When running control experiments using existing techniques (Question 4), 33% of respondents reported that they re-implement those techniques, 44% use third-party code, and the remainder failed to answer or found the question not applicable.

Figure 3 gives the responses for questions 7 and 8, which asked about the different types of GP that practitioners were using. Standard tree-based GP was the clear winner, with grammatical GP and standard GP with strong typing or other modifications the two runners-up.

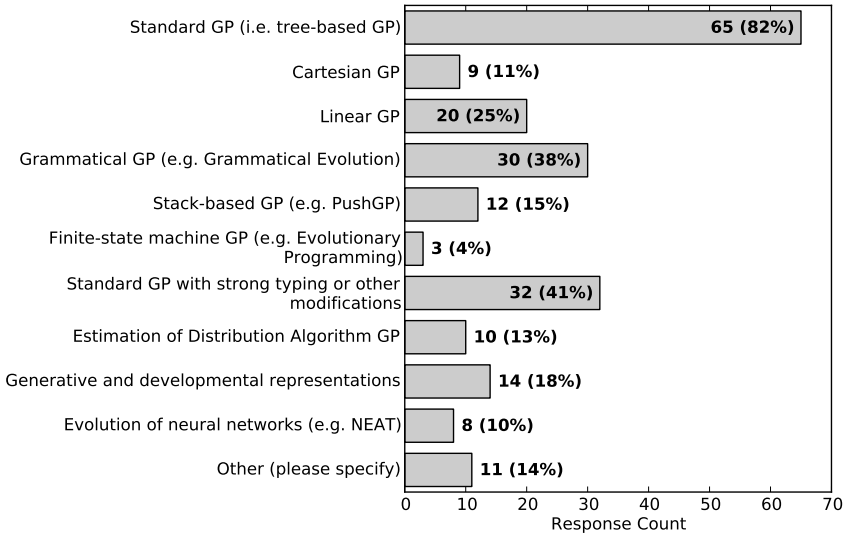
2.2.3 Problems with the Current Experimental Regime

In answer to Question 5, the vast majority—94% of respondents—thought that the "current experimental regime" has some disadvantages. They were asked to specify in detail what those disadvantages were, in Question 6.

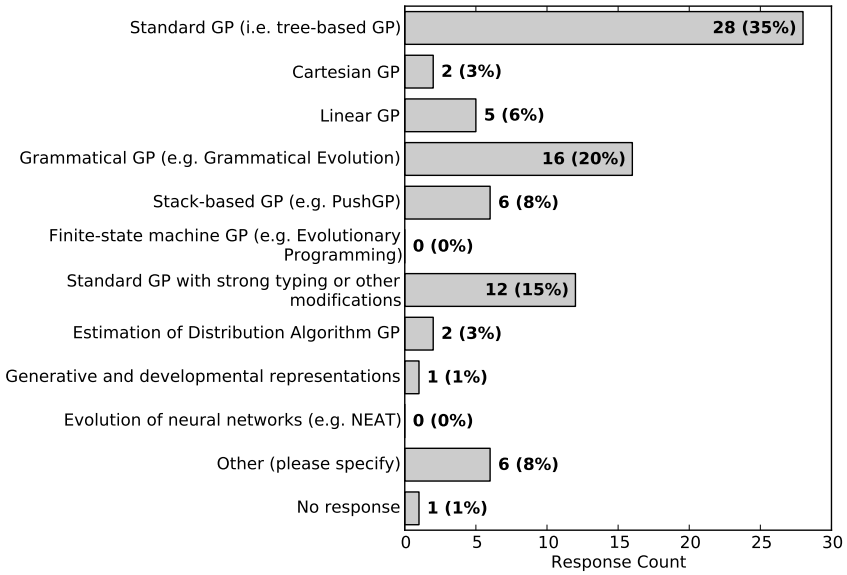
Figure 4 gives the results for pre-specified answers. A total of 76% mentioned the lack of standardization and difficulty of comparing results across papers, 73% mentioned the issue of "toy problems," and 28% mentioned wasted effort when running control experiments.

The optional free-text "other" response also yielded interesting results, with 22 respondents mentioning: poor choices of problems (often "toy problems," or out-of-date or non-standardized ones) and poor quality experimentation (non-standard methods, cherry-picking of results), statistics (non-standard and unrigorous), and reporting (omitted details prevent replicability). Many of these issues were again mentioned in free-text responses to Question 29.

In summary, respondents had two prime concerns with current empirical practice in the field. Firstly there was a strong consensus against the use of



(a) Question 7: Which types of GP do you use (tick all that apply)?



(b) Question 8: Which type of GP do you use the most?

Fig. 3 Questions 7 and 8 concerned the types of GP used by respondents.

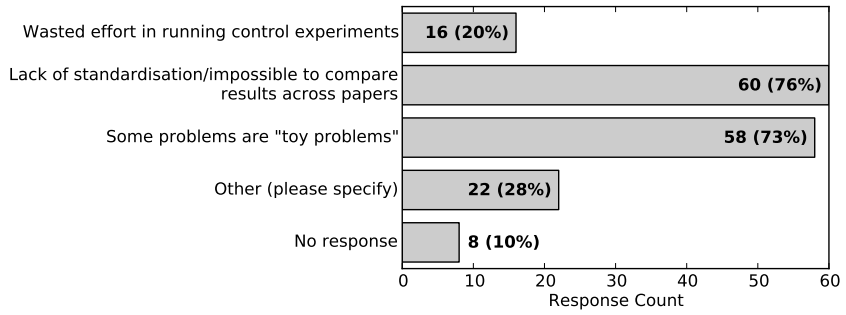


Fig. 4 Question 6: "If yes [current experimental regime has disadvantages], what are those disadvantages? Please check all that apply."

particular benchmarks, to be further discussed in Section 5. The second concern was that a lack of strong rigor and methodical reporting of results and methods is hampering replicability and progress in the field. These two issues appeared to be of approximately equal importance to respondents.

2.2.4 Other Benchmarking Efforts

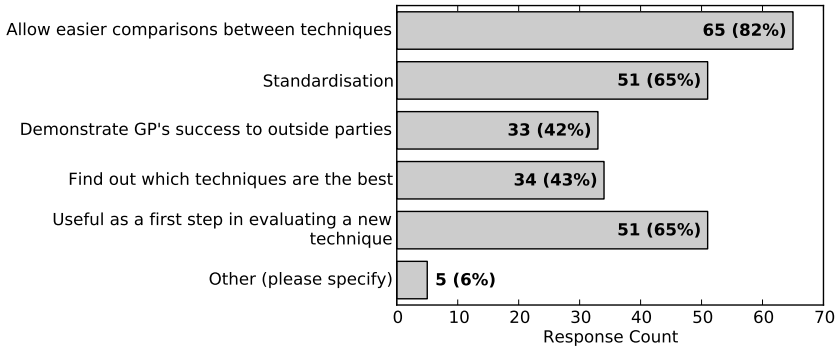
The next set of questions concerned other benchmarking efforts. Questions 9 and 10 asked about previous efforts at creating standardized benchmark suites for GP. A slight majority had heard of such efforts, but only 20% had used them. Question 11 asked about benchmark suites in areas other than GP: 47% said they had used such suites. That is, past attempts to create benchmark suites have failed to gain significant adoption by the GP community, in contrast to other fields.

2.2.5 Attitudes Towards a New Benchmark Suite

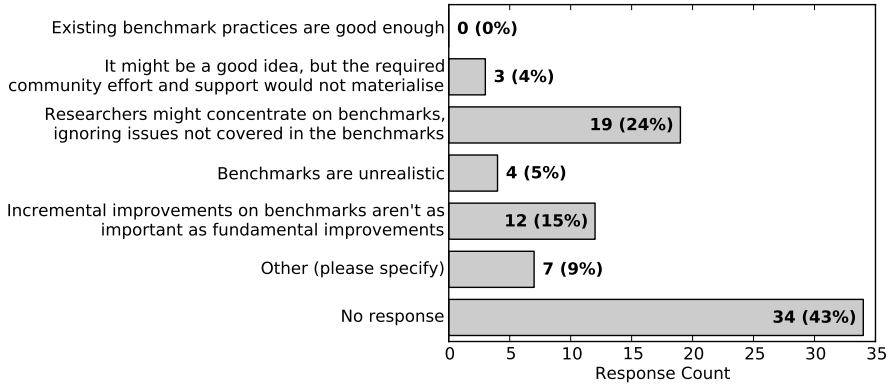
Questions 12–16 gauged respondents' attitudes towards the creation of a new benchmark suite. An overwhelming majority of responses were positive: 83% were in favor of the creation of a suite (Question 12), while 84% said they would use such a suite if it was created (Question 15). 91% said they would use such a suite in addition to their own preferred problems, rather than using the suite exclusively (Question 16). However, this was not the end of the argument; there was a small but strongly vocal minority against creating a standardized benchmark suite, and also a diversity of opinions about the why, the what, and the how of any potential suite.

The reasons given for and against the creation of a benchmark suite illustrate the diversity of views that exist in the community. Figure 5 shows responses covered by the pre-specified answer options.

Note that some respondents who were in favor of the creation of a benchmark suite nevertheless gave some reasons against it. The conditional wording of



(a) Question 13: "If yes, why? Please check all that apply."



(b) Question 14: "If not, why not? Please check all that apply."

Fig. 5 Questions 13 and 14 asked why respondents were or were not in favor of the creation of a benchmark suite.

Question 14 ("if not, why not?") was not enforced by the survey software. Therefore, it is best to read Question 14 as giving reasons against the creation of a suite, even if the respondent was on balance in favor (and vice versa for Question 13).

Arguments in favor of a benchmark suite were evenly divided among the several responses shown in Figure 5(a). Free-text responses in favor were for the most part simply elaborations on the pre-specified responses. A few extra motivations included improving the ability to track state-of-the-art results, to provide a taxonomy of GP problems, and to identify in such a taxonomy the types of problems for which GP is suitable.

Arguments against the creation of a benchmark suite were again divided among several responses, but the danger of "research distortion", i.e. concentration on benchmark performance to the exclusion of other issues, dominated.

The free-text responses here were very strongly argued. They might be grouped into two main categories:

1. Some said that a standardized suite could be actively harmful:
   - benchmarks are not realistic, and real-world applications are not always amenable to benchmarking; it is difficult to choose a benchmark that predicts performance on a specific real-world problem;
   - the existence of benchmarks can distort a field, encouraging "teaching to the test" and marginal improvements; one respondent said the problem of research distortion was "obvious in other fields";
   - a small group should not create a centralized benchmark in an attempt to dictate to the wider field;
   - a static benchmark suite could limit innovation; and
   - creating a benchmark suite that truly allows fair comparison of results across papers, without replication, is very difficult.
2. Several respondents argued that, although a standardized suite might not be harmful, GP experimental practice has other, more pressing problems, in particular its *rigor*. Within this category, four main issues were raised:
   - a lack of open source code and open data;
   - weak or non-standard statistical analysis;
   - unprincipled comparisons to results obtained with weak, unoptimized versions of other methods; and
   - insufficient reported detail for replication of experiments.

The free-text Question 29 was used by several respondents to expand on this topic, and responses were mixed: 30% of those who responded to Question 29 made generally positive comments, e.g. wishing the project good luck, while 15% made generally negative comments, including warnings that the creation of a benchmark suite could be harmful.

2.2.6 Composition of a New Benchmark Suite

Finally, opinions were solicited on the composition of any benchmark suite.

Question 17 asked whether real-world or synthetic problems should be used, and 93% were in favor of using *both*.

Question 19 asked about the *types* of problems that a benchmark suite should include. Six types were pre-specified: symbolic regression, Boolean functions, route-finding/planning, constructed problems, algorithmic problems (called "True programming" in the survey), and classification. All attracted

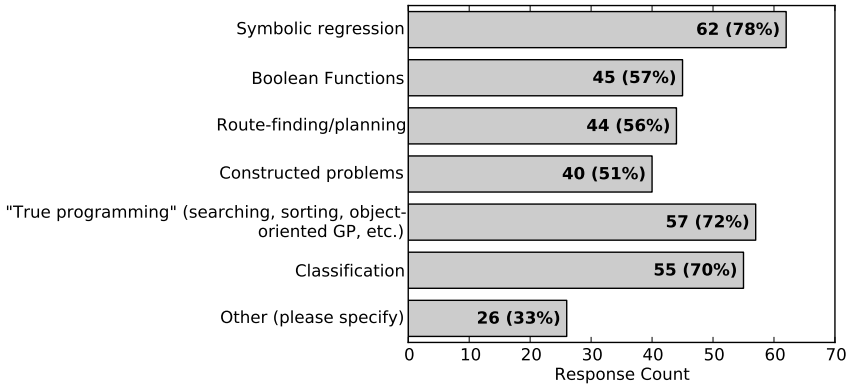


Fig. 6 Question 19: "What application domains and problem types should the benchmark suite contain? Please check all that apply."

majority support (see Figure 6). This is in accord with several of the free-text responses, which emphasized diversity of problems. However, two respondents pointed out that too many problems or too diverse a suite would also present difficulties to experimenters.

Other problem types mentioned by respondents were: games, pattern identification, agent control, signal processing, design, video compression, bioinformatics, data mining, text processing, computer vision, stock forecasting, and numerical problems such as finite element methods, partial differential equations, and time series prediction. Six respondents referred to "real-world" problems in general. Responses to the later free-response Question 29 mentioned pole-balancing and (again) algorithmic problems. Two respondents mentioned that theoretical or synthetic problems can have value as benchmarks.

Other responses to Question 29 argued that GP has become inward-looking, solving problems that would not be seen as interesting outside of the field. Four responses called for representation-independent problems that allow comparisons with non-GP techniques.

Questions 22 and 23 were for free-text responses only. They asked respondents to suggest *specific* problems that should and should not form part of a benchmark suite, respectively.

For *problems to include*, several respondents suggested new variants of symbolic regression, including dynamic symbolic regression, the Q-function used by Phong et al. [39], the chaotic flow function described by Sprott [44], symbolic regression for protein folding predictions, the Dow Chemical dataset used in previous symbolic regression EvoCompetitions (see Section 6.3), and the spline regression problems defined by Friedman [11].

Others proposed classification datasets available from UCI and KDD, and classification for bioinformatics applications. The Physical TSP problem and several games were mentioned. There was again support for signal processing,

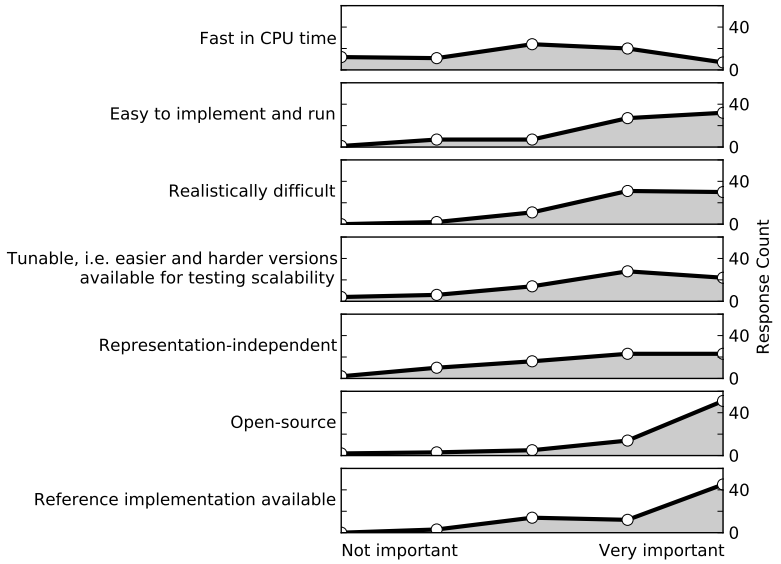


Fig. 7 Question 18. "How important are these properties of benchmarks?"

algorithmic and synthetic problems. Other suggested problems rarely mentioned elsewhere included prime number prediction [52] and real-valued optimization. One respondent recommended Boolean problems, but suggested that better ones than the typical parity and multiplexer problems were required, such as the multiple output parallel multiplier [53].

Concerning *problems to exclude*, the responses to Question 23 were dominated by arguments against the *de facto* benchmark suite of the quartic polynomial, lawnmower, artificial ant, multiplexer, and parity. In total 24 respondents argued against continued use of Koza's original demonstration problems in general, or one of the aforementioned problems directly. Four respondents argued against "toy problems". There was some support for moving away from symbolic regression, Boolean, and control problems entirely. Other specific problems mentioned were cart centering, two-box, and the royal tree. However, several respondents argued that all problems should be retained, subject to appropriate use.

Question 18 asked respondents to weigh the importance of various benchmark properties. Figure 7 shows the results. There is a skew towards regarding all properties as "important", but this does not prevent comparison of their relative importance. The property judged least important was being fast in CPU time, while being open-source and having a reference implementation available were regarded as the most important. Responses to the later free-text Question 29 also commented on this topic, mentioning that benchmarks should

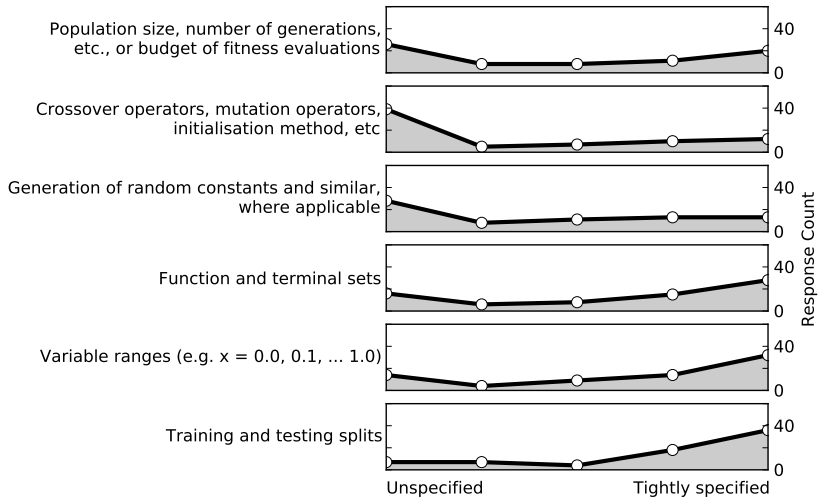


Fig. 8 Question 20: "If using a standardized benchmark suite, how tightly should details be specified? Please bear in mind that sometimes specifying details prevents comparison of non-standard or new techniques."

be real-world, representation-independent, fast and simple to execute and understand, realistically difficult, or tunably difficult.

Question 20 asked how much detail should be specified in a benchmark suite. Results (see Figure 8) were generally quite extreme: most respondents said that details should be either entirely unspecified or tightly specified. The strongest support for tight specification came for the function and terminal sets, variable ranges, and training and testing splits. The weakest support for tight specification came for properties such as population size, number of generations, or fitness evaluation budget; operators and initialization method; and generation of random constants "and similar, where applicable."

Question 20 did not allow free-text responses, but the following Question 21 asked respondents to give any other details they thought should be specified. 33 respondents gave a very great variety of responses. Several respondents proposed to use something other than fitness evaluations as the unit of budgeting: the number of node evaluations, the number of *fitness case* evaluations, or the number of runs. A few respondents wanted more details of the experimental method, analysis and reporting to be specified as part of every benchmark. Some also called for a standard reporting methodology, while others asked for solution complexity to be reported as standard. There was no clear consensus on what should be specified outside of the problem definition.

Other respondents emphasized that the programming language, the function set and terminal set, the methodology and range for constants, and the cross-validation strategy must be specified. Others said that the fitness function and the data generation method must be precisely specified, but one suggested that fitness should be black-box to prevent inadvertent bias. Special emphasis was

placed on the nitty-gritty details of interaction with a dynamic environment and exceptional cases such as overflows and range errors.

Some respondents used this question as an opportunity to emphasize that papers should include sufficient detail for replicability, whether a benchmark suite exists or not.

Other respondents cautioned against over-specification. One respondent stated that the crossover, mutation, and initialization operators should not be specified, while several more decided that details should be flexible where needed, and that a benchmark suite should be open to new ideas. One cautioned again against the creation of a "nanny state."

2.3 Threats to Validity

2.3.1 Internal Validity

The design and interpretation of a survey can affect its results. To avoid weaknesses in our survey design, we enlisted the help of several experienced survey designers to moderate our designs. Furthermore, we piloted the survey with individuals who were excluded from the final survey, and produced several revisions of the design. The full survey text, raw data for numerical responses and coded data for free-text responses is available online, so that third parties can form their own conclusions. Interpretation (coding) of the free text answers was performed independently by two researchers and subsequently discussed and resolved, in order to reduce the subjective element of the process.

2.3.2 External Validity

External validity is often threatened by low response rates. In order to gather as many responses as possible, we actively promoted the presentation of our previous GECCO paper, the discussion, and the survey itself via social media, mailing lists, and direct emails. Whilst this addressed the risk of low response rates, it also increased the risk of sample bias, i.e. an unrepresentative sample of views from the community. The good response rate partly mitigates this risk, but we have also been careful not to generalise from minority opinions that may be considered outliers. To further our policy of transparent reporting, we asked respondents where they heard of the survey, so that if any bias is introduced, it is at least reported.

3 Other Sources of Data and Opinion

Next, we summarize data and opinion from two other sources: the discussion following the presentation of our previous paper at GECCO 2012, and a review of current practice, in terms of the problems used in recent GP papers.

Category	Number of Papers	Percentage
Symbolic Regression	77	36.2
Quartic Polynomial	15	7.0
Classification	45	21.1
UCI database examples	23	10.8
Predictive Modeling	30	14.1
Boolean	37	17.4
Parity	31	14.6
Multiplexer	21	9.9
Path finding and planning	44	20.7
Artificial Ant	24	11.3
Control Systems	5	2.4
Game Playing	5	2.4
Dynamic Optimisation	7	3.3
Traditional Programming	8	3.8
Constructed Benchmarks	12	5.6
Other	10	4.7
Max problem	5	2.4

Table 2 Problem domains used in EuroGP and GECCO GP track papers 2009–2012.

3.1 GECCO Discussion

The informal audience discussion that followed the GECCO 2012 paper's presentation was a useful step in gathering community opinion. The discussion was attended by approximately 60 people, but less than 20 contributed to the conversation and thus this cannot be regarded as a representative sample or a community consensus. More comments and detail are available online.[2]

The feedback from this discussion is similar to that found in the survey responses mentioned above. There was a general sentiment that things must be improved, and that new benchmarks and better empirical practice were the two most promising ways of making this improvement.

As in the survey, there was a backlash against the "toy problems" that have become *de facto* benchmarks, and a few respondents suggested that in particular we should begin to move away from symbolic regression and classification, and towards planning and algorithmic problems (see Section 6.6). It was in this discussion that the idea of a "benchmark blacklist" was first proposed to discourage further usage of toy problems. New problems, it was argued by one attendee, should be much more difficult. Others asserted that benchmarks should be chosen in order to provide explicit direction to the field.

It was also debated as to whether multi-objective problems should be included in the suite; we feel that this is outside of the scope of this paper, and leave it the rest of the community to improve benchmarking in that area.

[2] `http://gpbenchmarks.org/gecco-2012-discussion/`

3.2 Review of Recent GP Practice

To assess current benchmarking practice, we surveyed the 229 articles presented in EuroGP and the GP track of GECCO from 2009 to 2012. Of these, 213 (93%) used benchmarking experiments. The results are shown in Table 2. Raw data from this complete review is available for download.[3]

4 Implications

4.1 Would a Benchmark Suite be Harmful?

The survey has revealed broad support from the community for the creation of a good, standardized benchmark suite. The potential advantages of such a suite are clear: it would allow direct comparisons of best results among GP methods and between GP and non-GP techniques; it would help to identify the state of the art, and the strengths and weaknesses of different approaches; and it would ensure that reported results represented real progress.

However, the community also expressed reservations. Over-tight specification of benchmarks might limit innovation. To give an example not taken from the survey, the results of the novel linear-scaling approach to symbolic regression fitness advocated by Keijzer [25] were not directly comparable to previously-published results with the same functions. A novel technique to improve performance in the context of GP numerical constants might be "outlawed" by a benchmark that tightly specified their mechanism.

The community is also aware of the danger that a benchmark suite could result in research distortion, i.e. researchers focusing only on benchmarks, "teaching to the test," and neglecting problems with large, real-world impact. The same argument has been made recently in the broader field of machine learning [5,51].

4.2 Rigor, Openness, and Replicability are more Important than Benchmarks

It is the strong consensus of the GP community that the current experimental regime is inadequate: see the responses to Questions 5 and 14. In many cases, respondents argued that it would be better to focus on serious problems concerning experimental and statistical rigor, openness and replicability, and reporting methodologies, rather than purely on the choice of benchmark problems. Responses to the demographics and other questions confirm that the vast majority of survey respondents came from within the field of GP. These criticisms of the field should be understood in this context: it is a case of GP practitioners wanting to improve their field, rather than the field being criticized from outside.

[3] `http://gpbenchmarks.org/publications/`

We broadly agree with these criticisms. In response, we argue that addressing the problem of badly-chosen benchmarks will not prevent progress on these other issues. It also seems unlikely that any "big bang" solution to these problems exists. Instead, improvements are likely to come about through a gradual process of education of new entrants to the field, raising awareness, and gradually raising the bar during the review process. We hope that this paper will contribute to raising awareness of the broader issue of experimental rigor in addition to that of benchmarks.

The issues raised in the survey overlap with points made by Johnson in his classic paper on the experimental analysis of algorithms [24]. A major problem with the current experimental regime is the difficulty of comparing results across papers (Question 6), which corresponds to Johnson's "ensure reproducibility" and "ensure comparability" principles. More generally, Johnson's paper contains a wealth of hard-won knowledge of what to do and what not to do when reporting experimental results.

Many GP results are measured using *ideal solution counts*: whether or how often the optimum, or some threshold near the optimum, is reached. The most common approach [28] defines the *computational effort* measure as an estimate of the minimum number of individuals to be processed in a generational algorithm in order to achieve a high probability of discovering a solution. This measure has received significant criticism [37,3,31,34,2,24]. Critics have argued that ideal solution counts are really a measure of how well a method solves trivial problems, and that other measures would be better, such as *best fitness of run* (appropriate for problems where the goal is optimization) and *generalization measures* such as final testing against a withheld generalization set, or K-fold validation (appropriate for problems where the goal is to perform model-fitting). Thus, reporting the distribution of fitness quality across independent runs is a better choice.

However, the expected fitness reflects real-world usage only when one anticipates that the user will be able to perform only a single run of the algorithm on a problem. Consider three systems for solving a hypothetical problem: system A achieves 0.4 error in all runs, system B achieves 0.1 error half the time and 0.9 error the other half, and system C achieves 0.01 error once every ten runs, but 0.99 error on all others. If the real-world usage scenario for this problem allows just one run, then examining just the mean error is appropriate, and favors system A. If the usage scenario allows for several runs, rather than one, system B is favored. If the usage scenario allows for a larger number of runs, the preferred system changes again to C. Johnson [24] discusses a computationally efficient bootstrapping method for estimating the "best of k" statistic that we suggest as a measure to correctly compare these three systems given the number of available runs k.

The issues of open source and open data, as raised in responses to Question 18 and others, are being discussed throughout the sciences.[4] We have made one contribution in this area, adding 57 symbolic regression benchmarks (those

[4] `http://sciencecodemanifesto.org`

listed in [32]) and several constructed problems (K-Landscapes, Royal Trees, Lid, Order and Majority, and Order Tree) to the open-source ECJ toolkit.[5]

4.3 No Consensus on Benchmark Suite

There is some community support for the creation of a new benchmark suite. However, responses concerning hypothetical future use of such a suite must be balanced against the reality that most respondents had not used previous benchmark suites (Question 10). There is also no prospect any new suite will take over and displace individual researchers' preferred problems (Question 16). This may help to reduce the danger of research distortion from a suite. There is also *no* consensus as to which problems should form part of a suite. A huge variety of problems and problem domains were suggested, with no clear winners. A variety of sometimes contradictory responses were also received on the features desirable in benchmarks and in a suite.

However, there was a strong consensus that certain problems should *no longer* be used for benchmarking purposes. In the next sections, we therefore propose a benchmark "blacklist" and for each blacklisted problem, we propose one or more appropriate replacements.

5 A Benchmark Blacklist

We claim broad support from the community for the proposition that a benchmarking experiment using *only* the problems given in Table 3 is inadequate.

Domain	Problem
Symbolic Regression	Quartic polynomial Low-order polynomials
Boolean	Parity (fixed input size) Multiplexer (fixed input size) Other simple one-output Boolean functions
Classification	Pima, Iris, Wine datasets
Path-finding and Planning	Artificial Ant Lawnmower

Table 3 A proposed blacklist of benchmark problems.

The review of current practice (see Section 3.2) suggests that these problems are still in use. The survey results in Section 2, especially the answers to Questions 6 and 23, demonstrates support for the blacklist. To leave the reader in no doubt as to the strength of feeling against these benchmarks, we provide a few quotes from the survey's free text responses:

[5] `http://cs.gmu.edu/~eclab/projects/ecj/`

Domain	Problem	Replaces
Symbolic Regression	Keijzer-6 Korns-12 Vladislavleva-4 *Original paper incorrectly stated 14.* Nguyen-7 Pagie-1 Dow Chemical dataset GP Challenge dataset	Quartic polynomial
Boolean functions	Multiple output multiplier	Multiplexer, Parity
Classification	PSP (protein structure) CHIRP suite	Iris, Pima, Wine
Planning and Control	Mario gameplay Physical TSP	Artificial ant Lawnmower
Synthetic Problems	K Landscapes Order Tree	(none)

Table 4 Alternatives to the blacklisted problems. References are provided below.

- "Far too many papers include results only on simple toy problems which are often worse than meaningless: they can be misleading";
- "[we should exclude] irrelevant problems that are at least 20 years old";
- "get rid of some outdated, too easy benchmarks";
- "the standard 'easy' Koza set should not be included";
- "[it is] time to move on."

6 Recommended Alternative Benchmarks

If some of the field's most common benchmark problems are to be blacklisted, we should offer alternatives. Since the community survey has revealed a broad consensus that the problem domains used in existing benchmarks should continue to be used (Question 19), it makes sense to attempt like for like replacements. Thankfully, good alternatives already exist in almost all cases. Our proposals are listed in Table 4.

The algorithmic and synthetic problem domains are not represented in the blacklist, but did receive majority approval from survey respondents for inclusion in future benchmark suites. We therefore wish to propose good benchmarks in these domains. However, we do not perceive any clear candidates for the algorithmic domain, as discussed in Section 6.6, and so it is omitted from the proposed benchmarks.

For each blacklisted problem domain, Table 4 lists one or more alternatives in the same domain. Our goals in choosing these problems were: to achieve a diverse set of modern, interesting, and difficult problems; to allow some comparison with non-GP techniques; to strike a balance between problems

Name	Variables	Equation	Training Set Testing Set
Keijzer-6 [25] [46]	1	$\sum_{i}^{x} \frac{1}{i}$	E[1, 50, 1] E[1, 120, 1]
Korns-12 [27]	5	$2 - 2.1\cos(9.8\,x)\,\sin(1.3\,w)$	U[-50, 50, 10000] U[-50, 50, 10000]
Vladislavleva-4 [50]	5	$\frac{10}{5+\sum_{i=1}^{5}(x_i-3)^2}$	U[0.05, 6.05, 1024] U[-0.25, 6.35, 5000]
Nguyen-7 [33]	1	$\ln(x+1)+\ln(x^2+1)$	U[0, 2, 20] None
Pagie-1 [36]	2	$\frac{1}{1+x^{-4}}+\frac{1}{1+y^{-4}}$	E[-5, 5, 0.4] None
Dow Chemical *(see Section 6.1)*	57	chemical process data[6]	747 points 319 points
GP Challenge [56] *(see Section 6.1)*	8	protein energy data	1250–2000 per protein None

Table 5 Proposed symbolic regression benchmarks. In the training and testing sets, U[*a*,*b*,*c*] is *c* uniform random samples drawn from *a* to *b*, inclusive. E[*a*,*b*,*c*] is a grid of points evenly spaced with an interval of *c*, from *a* to *b* inclusive.

which are fast to evaluate and problems which require more complex processing; and to allow some tunability of problems.

We do not argue that papers using the replacement problems must necessarily use all of them. However, we caution against cherry-picking, a problem identified in the community survey: if one runs experiments on multiple problems, clearly one must not choose to report only the best results.

The following sub-sections briefly discuss the problem domains, explaining our choice of benchmarks. We emphasize that we welcome feedback from the community on both the blacklist and the proposed replacements.

6.1 Symbolic Regression

As Symbolic Regression was the most requested problem domain from the survey (Question 19) as well as the most commonly used benchmark in the literature survey, we have suggested more benchmarks for this domain than any other. The seven proposed problems are listed in Table 5.

In order to ensure variety, we have selected one problem from each of several well-known sources in the specialist literature. We have chosen the more difficult problems available, both real-world and synthetic. The selection aims for *robust* difficulty: changes in alphabet, GP technique, numerical constant creation, and so on should not render the problems trivial. In fact, we have chosen not to specify such details here (they are specified in the original papers and collected in our previous work [32]), in order to make the problems representation-independent. These were issues mentioned by several respondents

[6] `http://casnew.iti.upv.es/index.php/evocompetitions/105-symregcompetition`

to the community survey. However, this means that in-paper controls will continue to be needed in most cases.

Several of these problems first appeared in papers with large numbers of citations, suggesting that they are well-known. The Keijzer-6 and Vladislavleva-4 problems require extrapolation, not just interpolation. The Korns-12 problem is the only problem unsolved in [27] despite the application of several specialized techniques. Note that the dataset is specified to have 5 variables, even though only 2 affect the output: the aim is to test the ability of the system to discard unimportant variables and avoid using them to over-fit. The authors of [50] say that Vladislavleva-4 is "our favourite problem", and that their system "appears to have most difficulties in discovering the simple and harmonious input–output relationship". The Pagie-1 problem has a reputation for difficulty [36,19] despite being smooth, and is scalable, in that extra dimensions could be added if or when the original version is seen as solved (the same is true of Vladislavleva-4). The Nguyen-7 problem features different basic functions from the others. Two of these problems provide no separate testing sets. In these cases a testing protocol can still be implemented using a hold-out set or cross-validation.

The Dow Chemical dataset was the subject of the Symbolic Regression EvoCompetitions event of the 2010 EvoStar conference. The dataset is available online, along with a description of the competition.[7] The goal was "to generate a robust symbolic regression model", leaving unspecified the objective function, the set of allowed functions, constants, and other details. This real-world example further motivates our choice not to specify the alphabet for the synthetic problems. Another dataset of the same origin was used as the "Tower" problem [50]. It is available online.[8]

The GP Challenge dataset[9] was proposed as a GP benchmark by Widera et al. [56]. The problem is to model the energy function of various proteins in terms of numerical input features.

6.2 Boolean Problems

Boolean problems have always been a mainstay of GP benchmarking. They abstract away difficult real-world issues and there are few if any edge-cases that require special attention (no division by zero, overflow, negative roots, infinities, or similar, which are problematic in symbolic regression). Similarly, issues such as numerical constants, sampling methodology, or approximate success criteria are avoided. Boolean functions often require reuse of input values and intermediate values and functionality, making them good tests for techniques designed to exploit functional redundancy in a problem.

However, the community has clearly shown that many are against the over-use of typical Boolean problems such as the parity and multiplexer problems for low, fixed arity. Some respondents argued that they are "toy" problems,

[7] http://casnew.iti.upv.es/index.php/evocompetitions/105-symregcompetition

[8] http://www.symbolicregression.com/

[9] http://www.infobiotics.org/gpchallenge

are uninteresting because they have been solved many times, or are trivial to solve using non-GP techniques. Previous work has also argued that, since the multiplexer's fitness values tend to chunk in powers of 2, statistical treatment of mean best fitness is problematic [32]. Individuals can also become expensive to evaluate for large input sizes, since it is usually necessary to test all 2^n fitness cases [23] (this argument does not necessarily apply to Boolean problems of *variable* input sizes [26]).

Our proposed replacement Boolean problem solves some of these issues, but not all. According to Walker and Miller [53], the multiple output parallel multiplier is by far the most difficult problem they attempted, and is now being seen as a benchmark. They state that the 5 bit multiplier is currently the largest successfully evolved. The problem has not been over-used in GP. It is interesting because multiple outputs are not native to standard GP. We also suggest that using multiple difficult Boolean problems of differing input sizes can help to avoid some of the drawbacks mentioned in the survey responses.

6.3 Classification

Classification is one of the major applications of machine learning. The best-known collection of datasets is the University of California, Irvine (UCI) machine learning repository[10] [10]. At the time of writing UCI had 228 datasets, with the majority being classification problems, as well as a small collection of regression and clustering datasets across a wide variety of domains. The datasets are popular in the machine learning community and frequently used for benchmarking. As described in Section 3.2, several of these problems are also popular in GP experimentation. However, some are either very easy (e.g. Iris or Wine), or very noisy (e.g. Pima). Several survey respondents argued against using them as benchmarks for these reasons. It has also been pointed out that very simple classifiers perform well on many frequently used UCI datasets [22], and the unreflective usage of UCI problems for benchmarking purposes has been strongly criticized [42,17].

Nevertheless, the state of the art in machine learning classification is often demonstrated in terms of performance on a selection of UCI and similar datasets. An example is the collection of 20 diverse classification problems used in [58], which we will refer to as the "CHIRP suite". Although this suite was not mentioned in survey responses, neither was there strong support for any other classification problem. Therefore we propose this suite for use in testing GP classification. The Weka machine learning toolkit[11] is a very useful resource for classification benchmarking: it provides many datasets, including the 20 CHIRP suite problems, and implementations of many common machine learning classification algorithms.

[10] `http://archive.ics.uci.edu/ml/`

[11] `http://www.cs.waikato.ac.nz/~ml/weka/`

The protein structure prediction (PSP) dataset[12] provides real-world data on protein folding in both regression and classification formulations [1]. We have proposed the datasets for classification because protein structure is an interesting and important problem, and the dataset is curated by GP practitioners with the aim of testing robustness and scalability in a realistic setting. We have omitted the regression datasets only because several good problems have already been identified for regression. The PSP datasets are very large, so training on an entire dataset may not be feasible.

Competitions are another possible source of datasets. SIGKDD hosts an annual data mining and knowledge discovery competition, the KDD Cup.[13] The tasks usually include building regression or classification models on large real-world datasets. The results of previous competitions are well documented and can be used as reference values. Kaggle[14] is a crowd-sourcing platform for data prediction competitions. Sponsored competitions on Kaggle feature time limits and provide data under a license that forbids use for other purposes. However, Kaggle also frequently hosts research-oriented or educational competitions. The NIPS feature selection challenge[15] includes five datasets, with a large number of features and few training examples, which could be suitable for benchmarking GP algorithms especially designed for such problems. However, there are also disadvantages to such competitions: algorithm comparisons can be difficult as the impact of various data preprocessing and model selection steps must be considered. Some well-known competitions such as the Netflix prize[16] require a great deal of domain knowledge, and so seem unsuitable as GP benchmarks.

6.4 Planning and Control

The real-world domain of planning and control can involve applications in vehicle piloting, robotics, and scheduling. Computer games and agent controllers offer cut-down, abstracted versions of these tasks, and can be entertaining and challenging problems. Several representation-independent competitions already exist for computer game controllers. GP could use these competitions to test new ideas against AI techniques developed in other fields, as well as against human players and their hand crafted controllers.

Some competitions (TORCS[17] [6,30], Mario Gameplay[18] [47], and Physical TSP[19] [38]) lend themselves to offline comparison between GP techniques, as controllers can be tested against static environments. Others (Ms. Pacman[20]

[12] http://icos.cs.nott.ac.uk/datasets/psp_benchmark.html
[13] http://www.sigkdd.org/kddcup/
[14] http://www.kaggle.com/
[15] http://www.nipsfsc.ecs.soton.ac.uk/datasets/
[16] http://www.netflixprize.com/
[17] http://torcs.sourceforge.net/
[18] http://www.marioai.org/
[19] http://www.ptsp-game.net/
[20] http://www.pacman-vs-ghosts.net/

[12], AI Challenge[21] [8], Robocode[22] [20], and Battle Code[23]) require head to head comparison of controllers. The advantage is obvious in a competition scenario: the difficulty of the problem increases as the quality of solutions improve. However, comparisons to previous techniques can be difficult, since fitness is relative. It might be possible to allow benchmarking with these problems by publishing a set of periodically updated reference opponents.

Games can also have significant limitations as benchmarks. The need to simulate a game to evaluate each individual makes these time intensive benchmarks. The problems being solved may not translate into real world applications. Also, the best competitors may need to utilize domain specific information, detracting from the generality of results.

We propose two interesting games as possible benchmarks: Mario gameplay, and the Physical TSP. Both are suited to offline comparisons, and relative to some other games, are relatively amenable to re-implementation. They do not require significant domain knowledge to get started. The Physical TSP competition includes a real-time aspect which prevents reproducibility, but this can be turned off for benchmarking purposes.[24] Evaluation of a game controller is very time-consuming for these games, relative to many of the other problems we have recommended. However this seems to be unavoidable in this domain. Feedback is solicited on better alternatives.

Many game maps suitable for testing path-finding agents are available online.[25]

6.5 Synthetic Problems

Several purely synthetic problems have been defined in the GP literature. They have several advantages: they are usually self-contained, in that they avoid difficulties in definition or implementation that are common in real-world problems; and they are often tunably difficult. In some cases the aim is to demonstrate and study specific issues such as bloat or crossover behavior. Examples include the Lid problem [4], the Max problem [13], Order and Majority [14], the Royal Trees [40], and Trap functions [48]. In other cases, the problems are intended as genuine benchmarks, including the K Landscapes [49], Order Tree [21], and Tree String [16]. All of the latter three have similar advantages and disadvantages: they have been deliberately designed as benchmarks, are modern, and are tunably difficult; on the other hand they are quite tied to the tree representation and are not obviously related to real-world problems. The Tree String problem is multi-objective, an advantage in terms of realism, but

[21] http://aichallenge.org/

[22] http://robocode.sourceforge.net/

[23] http://www.battlecode.org/

[24] https://groups.google.com/d/msg/the-ptsp-competition/TvtWhELr-z4/ACYCrNMVxToJ

[25] http://www.aigameresearch.org/

an obstacle to some experimenters. For this reason, we have not included it among our proposed replacement problems.

6.6 Algorithmic Programming Problems

To emulate the type of programming done by human programmers has always been one of the goals of the GP field. Work in this domain is sometimes referred to as "real programming", "object-oriented GP", "traditional programming", or "Turing-complete GP". We will refer to it as the "algorithmic" domain. It is a very different task from, for example, evolving numerical or Boolean formulae. Key features include stateful code, looping or recursion, conditional execution, multiple data types and compound data structures, and (compared to many GP applications) a very high degree of difficulty. This domain did not feature in the proposed blacklist of Section 5.

Survey results suggested that many respondents were in favor of including algorithmic GP benchmarks in any benchmark suite: in Question 19 (see Figure 6), it was the second most popular category after symbolic regression. However, it seems that there are few algorithmic GP benchmarks that have gained wide acceptance. This was one of the disadvantages of the current GP experimental regime highlighted in free-text responses to Question 6. Two respondents mentioned "software engineering" problems, and problems such as those typically found on undergraduate programming courses, in response to Question 19. However, no specific benchmarks were proposed.

One possibility is to look outside the field of GP for benchmark problems. Other fields often aim to solve problems similar to those tackled using GP. In the field of inductive programming [9], Gulwani [15] tackles several programming problems rarely seen in the field of GP, such as the creation of sorting algorithms, inverse functions like deserialization (given a serializer), and text-editing macros. Many bitwise algorithms and similar problems are proposed and solved (often through brute force) in "Hacker's Delight" [54].

Search-based software engineering [18] is another field that could serve as a source for algorithmic GP benchmark problems. This field has already seen some GP applications as a tool for repairing bugs found in human created software [55,57].

Competitions are again a possible source of benchmarks in this domain. For example, Google Code Jam is an ongoing competition for human coders.[26] Problems range from the very easy, such as reversing a list of words, to very challenging algorithm design. They are specified in textual form, usually with example input-output pairs. Since it is a competition for human coders, good GP efforts might qualify as human-competitive. "Online judges", essentially automated graders for programming tasks, may be a useful resource. For example, the Sphere online judge[27] provides automated grading of 3086 problems.

[26] `http://code.google.com/codejam/contests.html`

[27] `http://www.spoj.pl/`

7 Conclusions

In this paper, we have contributed the following:

1. A community survey capturing current attitudes to empirical practice within Genetic Programming.
2. The results of a discussion held at the GECCO 2012 conference.
3. An extended review of problems used in empirical work published in recent years at major conferences.
4. A proposed blacklist of GP problems that should no longer be used.
5. Proposed replacement problems for those excluded through the blacklist.
6. Further resources for those seeking new problems and domains for their research.

7.1 What Next?

For the moment, we do not plan to make any further proposals. Instead, we wish to see what impact, if any, the proposed blacklist and replacements have on the field. We intend to revisit the survey of papers over time in order to assess this impact. The suggested replacements may act as a vehicle for assessing the palatability of any extensive, newly-proposed benchmark suite.

In a few areas there is a clear need for the development of *new* benchmarks. As stated, we think that this task should be left to individual researchers with specific experience. For algorithmic GP, a good problem or set of benchmark problems in this area would be a significant contribution. In the games domain, the required task is not to invent new problems, but to make existing games more usable as GP benchmarks.

Finally, we wish to appeal to the GP community to make full use of the information presented here. There are many interesting issues raised, and by making the data freely available we hope to encourage practitioners to draw their own conclusions. There is an opportunity to drive the field forward by developing new methodologies, improving statistical approaches, ensuring replicability of results and pushing the boundaries of GP towards more diverse and challenging problems.

Acknowledgements Thanks to Ricardo Segurado and the University College Dublin CSTAR statistics consultancy; thanks to Marilyn McGee-Lennon in the School of Computing Science at the University of Glasgow for her advice on survey design, and to the School itself for providing the supporting web service. Thanks to all those who participated in the GP survey and have engaged in discussion through the GP mailing list, the benchmark mailing list, and the GECCO 2012 debate. Thanks to the anonymous reviewers of this paper.

David R White is funded by the Scottish Informatics and Computer Science Alliance. James McDermott is funded by the Irish Research Council. Gabriel Kronberger is supported by the Austrian Research Promotion Agency, Josef Ressel-centre "Heureka!" Wojciech Jaśkowski is supported by Polish Ministry of Science and Education, grant no. 91-531/DS.

References

1. Bacardit, J., Stout, M., Krasnogor, N., Hirst, J.D., Blazewicz, J.: Coordination number prediction using learning classifier systems. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO), p. 247. Seattle, Washington, USA (2006). DOI 10.1145/1143997.1144041
2. Barrero, D.F., R-Moreno, M., Castano, B., Camacho, D.: An Empirical Study on the Accuracy of Computational Effort in Genetic Programming. In: Proceedings of the Congress on Evolutionary Computation (2011)
3. Christensen, S., Oppacher, F.: An Analysis of Koza's Computational Effort Statistic for Genetic Programming. In: Proceedings of EuroGP. Springer-Verlag (2002)
4. Daida, J.M., Bertram, R., Stanhope, S., Khoo, J., Chaudhary, S., Chaudhary, O.: What Makes a Problem GP-Hard? Analysis of a Tunably Difficult Problem in Genetic Programming. Genetic Programming and Evolvable Machines **2**, 165–191 (2001)
5. Drummond, C., Japkowicz, N.: Warning: Statistical benchmarking is addictive. Kicking the habit in machine learning. Journal of Experimental & Theoretical Artificial Intelligence **22**(1), 67–80 (2010)
6. Espié, E., Guionneau, C., Wymann, B., Dimitrakakis, C., Coulom, R., Sumner, A.: TORCS—the open racing car simulator (2005)
7. Feldt, R., O'Neill, M., Ryan, C., Nordin, P., Langdon, W.B.: GP-Beagle: A benchmarking problem repository for the genetic programming community. In: Late Breaking Papers at GECCO (2000)
8. Fernández-Ares, A., Mora, A., Merelo, J., García-Sánchez, P., Fernandes, C.: Optimizing player behavior in a real-time strategy game using evolutionary algorithms. In: Proceedings of the Congress on Evolutionary Computation, pp. 2017–2024. IEEE (2011)
9. Flener, P., Schmid, U.: An introduction to inductive programming. Artificial Intelligence Review **29**(1), 45–62 (2008)
10. Frank, A., Asuncion, A.: UCI machine learning repository (2010). URL `http://archive.ics.uci.edu/ml`
11. Friedman, J.: Multivariate adaptive regression splines. The Annals of Statistics pp. 1–67 (1991)
12. Gallagher, M., Ryan, A.: Learning to play Pac-Man: An evolutionary, rule-based approach. In: Proceedings of the Congress on Evolutionary Computation, vol. 4, pp. 2462–2469. IEEE (2003)
13. Gathercole, C., Ross, P.: An Adverse Interaction between Crossover and Restricted Tree Depth in Genetic Programming. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO) (1996)
14. Goldberg, D.E., O'Reilly, U.M.: Where does the Good Stuff Go, and Why? How Contextual Semantics Influence Program Structure in Simple Genetic Programming. In: Proceedings of EuroGP (1998)
15. Gulwani, S.: Dimensions in program synthesis. In: Proceedings of the 12th international ACM SIGPLAN symposium on Principles and practice of declarative programming, pp. 13–24. ACM (2010)
16. Gustafson, S., Burke, E.K., Krasnogor, N.: The Tree-String Problem: An Artificial Domain for Structure and Content Search. In: Proceedings of EuroGP (2005)
17. Hand, D.J.: Classifier technology and the illusion of progress. Statistical Science **21**(1), 1–14 (2006)
18. Harman, M., Jones, B.: Search-based software engineering. Information and Software Technology **43**(14), 833–839 (2001)
19. Harper, R.: Spatial co-evolution: quicker, fitter and less bloated. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO), pp. 759–766. ACM (2012)
20. Hartness, K.: Robocode: using games to teach artificial intelligence. J. Comput. Sci. Coll. **19**(4), 287–291 (2004)
21. Hoang, T.H., Hoai, N.X., Hien, N.T., McKay, R.I., Essam, D.: ORDERTREE: a New Test Problem for Genetic Programming. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO) (2006)

22. Holte, R.C.: Very simple classification rules perform well on most commonly used datasets. Machine Learning **11**, 63–90 (1993)
23. Imamura, K., Foster, J., Krings, A.: The test vector problem and limitations to evolving digital circuits. In: Proceedings of the Second NASA/DoD Workshop on Evolvable Hardware, pp. 75–79. IEEE (2000)
24. Johnson, D.: A theoretician's guide to the experimental analysis of algorithms. Data structures, near neighbor searches, and methodology: fifth and sixth DIMACS implementation challenges **59**, 215–250 (2002)
25. Keijzer, M.: Improving Symbolic Regression with Interval Arithmetic and Linear Scaling. In: Proceedings of EuroGP (2003)
26. Kirshenbaum, E.: Iteration over vectors in genetic programming. HP Laboratories Technical Report HPL-2001-327 (2001)
27. Korns, M.F.: Accuracy in Symbolic Regression. In: Proceedings of Genetic Programming Theory and Practice (2011)
28. Koza, J.R.: Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press (1992)
29. Koza, J.R.: Genetic Programming II: Automatic Discovery of Reusable Programs. MIT Press (1994)
30. Loiacono, D., Togelius, J.: Competitions@WCCI-2008: simulated car racing competition. ACM SIGEVOlution **2**(4), 35–36 (2007)
31. Luke, S., Panait, L.: Is The Perfect The Enemy Of The Good? In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO) (2002)
32. McDermott, J., White, D.R., Luke, S., Manzoni, L., Castelli, M., Vanneschi, L., Jaśkowski, W., Krawiec, K., Harper, R., Jong, K.D., O'Reilly, U.M.: Genetic programming needs better benchmarks. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO). ACM, Philadelphia (2012)
33. Nguyen, Q.U., Nguyen, X.H., O'Neill, M., Mckay, R.I., Galván-López, E.: Semantically-Based Crossover in Genetic Programming: Application to Real-valued Symbolic Regression. Genetic Programming and Evolvable Machines **12**, 91–119 (2011)
34. Niehaus, J., Banzhaf, W.: More on Computational Effort Statistics for Genetic Programming. In: Proceedings of EuroGP (2003)
35. O'Neill, M., Vanneschi, L., Gustafson, S., Banzhaf, W.: Open Issues in Genetic Programming. Genetic Programming and Evolvable Machines **11**(3/4), 339–363 (2010)
36. Pagie, L., Hogeweg, P.: Evolutionary Consequences of Coevolving Targets. Evolutionary Computation **5**, 401–418 (1997)
37. Paterson, N., Livesey, M.: Performance Comparison in Genetic Programming. In: Late Breaking Papers at GECCO (2000)
38. Perez, D., Rohlfshagen, P., Lucas, S.M.: Monte-Carlo tree search for the physical travelling salesman problem. In: C. Di Chio, A. Agapitos, S. Cagnoni, C. Cotta, F. Fernández, G.A.D. Caro, R. Drechsler, A. Ekárt, A.I. Esparcia-Alcázar, M. Farooq, W.B. Langdon, J.J. Merelo-Guervós, M. Preuss, H. Richter, S. Silva, A.S. oes, G. Squillero, E. Tarantino, A.G.B. Tettamanzi, J. Togelius, N. Urquhart, A. Şima Uyar, G.N. Yannakakis (eds.) Proceedings of EvoApplications, *Lecture Notes in Computer Science*, vol. 7248, pp. 255–264. Springer Berlin Heidelberg (2012)
39. Phong, D., Hoai, N., McKay, R., Siriteanu, C., Uy, N., Park, N.: Evolving the best known approximation to the q function. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO), pp. 807–814. ACM (2012)
40. Punch, B., Zongker, D., Goodman, E.: The Royal Tree Problem, a Benchmark for Single and Multiple Population Genetic Programming. In: Advances in Genetic Programming 2, pp. 299–316. MIT Press (1996)
41. Ritchie, J., Lewis, J. (eds.): Qualitative research practice: A guide for social science students and researchers. Sage (2003)
42. Salzberg, S.: On comparing classifiers: Pitfalls to avoid and a recommended approach. Data Mining and Knowledge Discovery **1**, 317–328 (1997)
43. Sendhoff, B., Roberts, M., Yao, X.: Evolutionary Computation Benchmarking Repository. IEEE Computational Intelligence Magazine **1**(4), 50–60 (2006)
44. Sprott, J.C.: Simplest dissipative chaotic flow. Physics letters A **228**(4), 271–274 (1997)
45. Strauss, A., Corbin, J.: Grounded Theory in Practice. Sage (1997)

46. Streeter, M., Becker, L.A.: Automated discovery of numerical approximation formulae via genetic programming. Genetic Programming and Evolvable Machines **4**, 255–286 (2003). URL http://dx.doi.org/10.1023/A:1025176407779
47. Togelius, J., Karakovskiy, S., Baumgarten, R.: The 2009 mario ai competition. In: Proceedings of the Congress on Evolutionary Computation (2010)
48. Tomassini, M., Vanneschi, L., Collard, P., Clergue, M.: A Study of Fitness Distance Correlation as a Difficulty Measure in Genetic Programming. Evol. Comput. **13**, 213–239 (2005). DOI http://dx.doi.org/10.1162/1063656054088549
49. Vanneschi, L., Castelli, M., Manzoni, L.: The K Landscapes: a Tunably Difficult Benchmark for Genetic Programming. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO) (2011)
50. Vladislavleva, E., Smits, G., Den Hertog, D.: Order of Nonlinearity as a Complexity Measure for Models Generated by Symbolic Regression via Pareto Genetic Programming. IEEE Transactions on Evolutionary Computation **13**(2), 333–349 (2009)
51. Wagstaff, K.L.: Machine learning that matters. In: J. Langford, J. Pineau (eds.) Proceedings of the 29th International Conference on Machine Learning (ICML-12) (2012)
52. Walker, J., Miller, J.: Predicting prime numbers using Cartesian genetic programming. Proceedings of EuroGP pp. 205–216 (2007)
53. Walker, J., Miller, J.: The automatic acquisition, evolution and reuse of modules in Cartesian genetic programming. IEEE Transactions on Evolutionary Computation **12**(4), 397–417 (2008)
54. Warren, H.: Hacker's delight. Addison-Wesley Professional (2003). URL http://hackersdelight.org/
55. Weimer, W., Nguyen, T., Le Goues, C., Forrest, S.: Automatically Finding Patches using Genetic Programming. In: Proceedings of the 31st International Conference on Software Engineering (2009)
56. Widera, P., Garibaldi, J., Krasnogor, N.: GP challenge: Evolving energy function for protein structure prediction. Genetic Programming and Evolvable Machines **11**, 61–88 (2010)
57. Wilkerson, J.L., Tauritz, D.R., Bridges, J.: Multi-objective coevolutionary automated software correction system. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO). ACM, Philadelphia (2012)
58. Wilkinson, L., Anand, A., Tuan, D.: CHIRP: a new classifier based on composite hypercubes on iterated random projections. In: Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD), vol. 11, pp. 6–14 (2011)