Chapter 17

# PARETO-FRONT EXPLOITATION IN SYMBOLIC REGRESSION

Guido F. Smits[1] and Mark Kotanchek[2]
[1]*Dow Benelux, Terneuzen, NV;* [2]*Dow Chemical, Midland, MI USA*

**Abstract** Symbolic regression via genetic programming (hereafter, referred to simply as symbolic regression) has proven to be a very important tool for industrial empirical modeling (Kotanchek et al., 2003). Two of the primary problems with industrial use of symbolic regression are (1) the relatively large computational demands in comparison with other nonlinear empirical modeling techniques such as neural networks and (2) the difficulty in making the trade-off between expression accuracy and complexity. The latter issue is significant since, in general, we prefer parsimonious (simple) expressions with the expectation that they are more robust with respect to changes over time in the underlying system or extrapolation outside the range of the data used as the reference in evolving the symbolic regression.

In this chapter, we present a genetic programming variant, ParetoGP, which exploits the Pareto front to dramatically speed the symbolic regression solution evolution as well as explicitly exploit the complexity-performance trade-off. In addition to the improvement in evolution efficiency, the Pareto front perspective allows the user to choose appropriate models for further analysis or deployment. The Pareto front avoids the need to *a priori* specify a trade-off between competing objectives (e.g. complexity and performance) by identifying the *curve* (or surface or hyper-surface) which characterizes, for example, the best performance for a given expression complexity.

**Keywords:** genetic programming, Pareto front, multi-objective optimization, symbolic regression, ParetoGP

## 1. Introduction

Unlike normal regression in which a model structure (e.g., second-order polynomials) is hypothesized and fit to available data, symbolic regression involves the discovery of the *structure* as well as the *coefficients* within that structure. One way to accomplish this is to use genetic programming (GP)

techniques to evolve expressions which match the observed system behavior. In this section, we briefly review the practical motivations for symbolic regression as well as the classical problems characteristic to the GP approach. Finally, we outline a variant of GP which addresses some of the classical issues and has resulted in a significant improvement in the speed and robustness of symbolic regression. This variant focuses on the evolutionary effort on improving the Pareto front (which captures the trade-offs between competing objectives) rather than optimizing a single composite criteria. The rest of the paper is devoted to exploring the algorithm, its benefits and its performance.

In this chapter we assume that the reader has a working knowledge of GP concepts (Banzhaf et al., 1998, Jacob, 2001)) as well as its application to symbolic regression (Kotanchek et al., 2003).

## Motivations for Symbolic Regression

In addition to the real-world benefits of empirical modeling for system modeling, emulation, monitoring and control, symbolic regression has several unique contributions. These contributions, which are especially important when faced with multivariate data from a nonlinear but unknown system, include:

- **Human insight** — examination of the evolved expressions can be indications of underlying physical mechanisms as well identification of *metavariables* (combinations or transforms of variables) which can simplify subsequent empirical modeling efforts. Additionally, examining the structure of an evolved model can be comforting in the sense that the model behavior, variables and metavariables agree with human expectation; this explainability helps to instill trust in the model(s).

- **Compact models** — generally solutions can be identified which perform well and are parsimonious with respect to structure complexity and/or number of input variables. Such models are attractive because they can be interpreted more easily and deployed easily in many environments. Such models may also be more robust and capture underlying fundamentals rather than system noise in the data.

- **Limited *a priori* assumptions** — unlike traditional regression which assumes a model structure for the data, symbolic regression allows the data to determine which structures of variables, functions and constants are appropriate to describe the observed behavior. Of course, appropriate functional building blocks as well as the pertinent data variables must be supplied for the evolution.

- **Natural variable selection** — the evolutionary processes of GP have a remarkable ability to focus on the driving variables necessary to capture

the system behavior. Furthermore, post-processing can do sensitivity analysis on the evolved expressions to discard superfluous variables.

- **Diverse models (possibly)** — the evolutionary process will often develop models of similar performance using different variables (symbolic regression does not require that correlated variables be eliminated *a priori*) or different structures. Although it is difficult to characterize nonlinear model diversity, ensembles of diverse models can be useful to: (1) indicate operation outside the domain of the training data (due to divergence of model predictions) as well as (2) assemble robust online predictive models which are resistant to sensor failures and related pathologies.

- **Physical model integration** — integration with fundamental (first–principles) models can control the extrapolation behavior of developed models and, thereby, result in more robust models which are aligned with the underlying physics. Of course, examining the structure of evolved models within the context of theoretical insight can help to identify the nature of the applicable fundamental behavior.

## Problems with Symbolic Regression

Despite the plethora of advantages discussed above, there are at least three fundamental problems with symbolic regression via GP:

- **slow discovery** — classically, symbolic regression is very CPU intensive and slower than other nonlinear modeling techniques such as neural networks. That said, it is impressively efficient when the infinite size of the search space is considered.

- **difficulty in selection of good solutions** — during the course of an evolution, many candidate solutions will be evolved. Selecting the "best" solutions while balancing performance vs. model complexity trade-offs is a difficult exercise, as is the issue of detecting model pathologies in regions in which a model is not constrained by data.

- **good-but-not-great models** — other nonlinear techniques will typically outperform the symbolic regression models on training data. That said, symbolic regression can be a precursor to great models due to its capabilities for variable selection, metavariable identification, secondary rounds of symbolic regression evolutions and iterative prototyping including human input.

As will be demonstrated by the rest of this paper, the proposed symbolic regression variant which exploits the Pareto front can mitigate these problems.

## New Variant: Exploit the Pareto Front

Unconstrained, GP has a terrible problem with *bloat* wherein the size of the evolved expressions grows to massive proportions due to the presence of *introns* (nonfunctional substructures) as well as the pursuit of mimicking each nuance of the data — i.e., modeling noise rather than fundamentals. Typically, the approach adopted to control this tendency has been to apply parsimony pressure so that the fitness metric considers both the performance of the evolved expression in terms of matching the data behavior *and* the complexity of the evolved expression. The problem with this single metric approach is that the complexity-performance trade-off cannot be made prior to model discovery despite the need to do so.

To resolve this quandary, we can leverage the notion of a Pareto front from the multi-objective optimization community. Given a population of solutions, the Pareto front considers all objectives to be equally important and identifies those solutions which are *nondominated*. This is illustrated in Figure 17-1 wherein the performance of a population is shown for two performance criteria with smaller values of each being preferable. The Pareto front consists of those members of the population for which there exists no solution which is better in *both criteria* than the Pareto set member. Thus, we can easily focus on the proper population members to explicitly make trade-offs of model complexity vs. performance.

Given that the Pareto front represents the best individuals in a population, it is reasonable to assume that we want to focus the evolutionary process on individuals on or near the Pareto front with a goal of pushing the Pareto down and to the left (in the case of the example shown in Figure 17-1) At the same time, given the limited size of the Pareto set relative to the overall population, we do not want to lose the diversity of building blocks present in that population nor the ability to discover new solutions and structures.

Once we realize the significance of the Pareto-based criteria, there are many ways to exploit the representation. As will be described below in more detail, an approach which has proven effective is to breed the Pareto set members with the best (in terms of model error) members of the population At the end of each generation, the Pareto set is updated to reflect any improvements in the Pareto set and the process is repeated. The Pareto archive is maintained across multiple GP *cascades* with each independent GP cascade contributing new genetic material in the exploration of the model structure space. Multiple independent *runs* (each containing their own set of cascades) are used to mitigate against evolutionary lock-in on a successful structure.

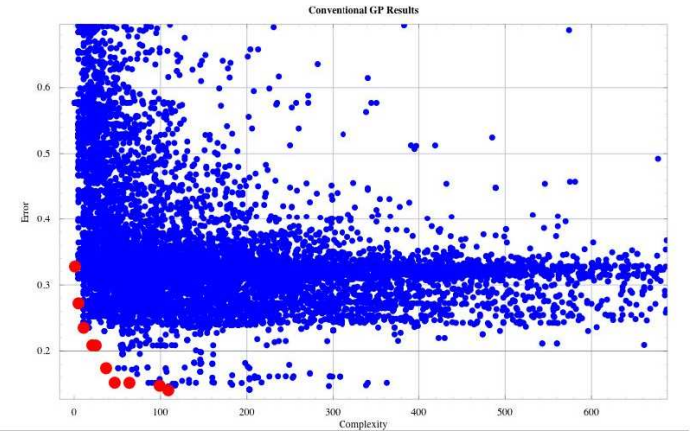


*Figure 17-1.* Here we illustrate the Pareto front (large red dots) in a population where the two axes represent competing objectives (with smaller values being better). This illustration uses the results from a conventional GP symbolic regression run; note that much of the computational effort was spent exploring entities of high complexity despite marginal (if any) incremental improvements over their less complex peers.

## 2. Pareto (Optimization) Axes

In this section, we discuss possible definitions of "good" — i.e., the various criteria which might be used to define the axes of the Pareto front. Symbolic regression will need to include model performance as well as model complexity; however, there are many possible definitions within these characteristics. Since the notion of a Pareto front is not constrained to two dimensions, we can also incorporate other performance metrics into the Pareto front. However, in general, the number of dimensions should be kept relatively low for computational performance reasons.

### Model Performance

Model *performance* defines the absolute fitness (quality or accuracy) of an evolved structure and characterizes how well the model matches the observed data behavior. In our industrial applications, we have used:

- **1-norm** (sum of absolute values of error)
- **2-norm** (i.e., least squares criteria)
- ***n*-norm** (higher values of $n$ weight large errors more; $\infty$-norm returns magnitude of largest error)
- **absolute correlation** (measures response surface matching independent of scaling and translation)

- **products** of the above
- **epsilon-insensitive zones** (only errors greater than some threshold count against performance)

Selection of the model performance criteria is model dependent. Also note the role of data balancing as a data conditioning step since the typical industrial symbolic regression data set is not the result of a designed experiment and, therefore, is likely to have some regions of parameter space over-represented relative to others — which can skew the performance metrics towards those regions at the expense of overall performance.

## Model Complexity Measures

Characterizing the *complexity* of an expression should, in principle, consider two aspects: (1) the complexity of the expression structure and (2) the complexity of the derived response surface. Although the first aspect is difficult, it is much more tractable than the second aspect — especially for targeted systems having many input parameters. Hence, for computational simplicity reasons, we have chosen to define model complexity based upon the structure rather than the response. Even with this reduction in scope, there are many possible metrics for complexity; these include (assuming a tree-based GP implementation):

- **Tree depth** — the number of levels in the structure
- **Tree nodes** — the number of leaves and branch points in the structure
- **Component function nonlinearity** — e.g., "+" is less nonlinear than exponentiation, sines or if-than-else constructs
- **Number of variables** — either number of variables as leaves or the count of unique variables within the expression
- **Combinations of the above**

Although still an open area of research, we are currently using as a complexity metric the sum of the complexities of the tree structure and all subtrees — where the complexity is defined as the number of nodes (branch points plus leaves). This sort of metric has the advantage of favoring fewer layers as well as providing more resolution at the low end of the complexity axis of the Pareto front so that more simple solutions may be included in the Pareto front. Figure 17-2 illustrates the computation of this complexity measure whereas Figure 17-3 illustrates the complexity difference possible when two different genotype representations result in the same phenotype expression.

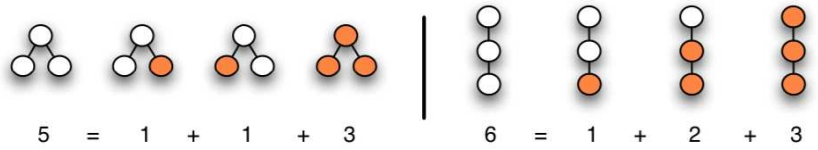


*Figure 17-2.* Here we illustrate the computation of expression complexity with two simple three node structures. This complexity metric of the sum of the number of nodes of all subtrees will favor flatter and balanced structures for equivalent node counts.

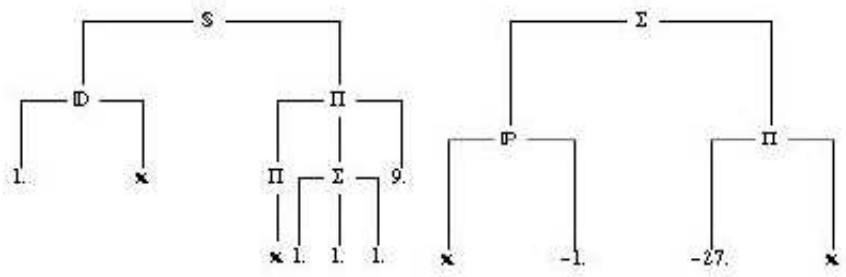


*Figure 17-3.* These two genome tree representations are equivalent to the expression $\frac{1}{x} - 27x$. Despite their equivalent raw performance, the left one has a complexity measure of 36 whereas the right one has a measure of 17; hence, the second structure would *dominate* the first from a Pareto perspective. (Here the genome representation is S $\rightarrow$ subtract, $\Sigma\rightarrow$ sum, D$\rightarrow$divide, P$\rightarrow$power, and $\Pi\rightarrow$product.)

## Other Dimensions

There are other criteria which we may want to include in evaluating symbolic regression expressions. For example,

- performance on different data sets (e.g., training, test, and validation)
- uniqueness of included variables, number of variables, structures etc.

Although we could articulate these as additional dimensions in the multi-objective optimization space represented by the Pareto front, we generally want to keep the dimensionality as low as possible for practical reasons, since additional dimensions bring the "curse of dimensionality" to bear and greatly increases the difficulty of covering the Pareto front (hyper)surface.

## 3. Defining Pareto Optimality

Although defining the Pareto set is relatively straightforward since the definition if unambiguous, a definition of Pareto optimality is required if our interest extends to solutions not on the Pareto front or if we have an objective to fully

populate the front (i.e., fill in gaps). This isn't as critical for the symbolic regression application as it is for more traditional multiobjective optimization where having a detailed understanding of the shape of the Pareto front is critical. For symbolic regression, "good enough" can be more easily achieved.

Although the subtleties of Pareto optimality are not critical for the currently proposed algorithm, an understanding of such issues may be important for successor algorithms. Hence, we include the discussion of this section. This section draws heavily from (Eckart Zitzler and Bleuler, 2004) and (Jensen, 2003).

## Definition of Pareto-Front

As discussed in Section 2 the notion of the Pareto front is founded on a concept of *dominance* with the Pareto front at any instant consisting of *nondominated* solutions — for example, no other solution is better than the solutions of the Pareto front in *both* complexity and performance. The curve (or surface or hyper-surface, depending upon the number of objectives considered) defined by the Pareto points does not need to be convex. Thus, the user can judge whether an incremental gain in performance is worth the associated increase in expression complexity or if there is major improvement associated with an incremental increase in complexity.

## Pareto Performance Metrics

There are a variety of ways to characterize the Pareto performance. We would, in general, prefer to fully explore and populate the Pareto front as well as refine and enhance the demonstrated success of the known members of the Pareto set. Thus, we are faced with the common problem of evolutionary algorithms of balancing exploration and exploitation.

A natural approach to ensuring the exploration of the Pareto front is to reward uniqueness. Unfortunately, explicitly rewarding this is difficult due to difficulty of choosing a proper scaling and distance metric for the diverse axes of the multiobjective optimization and, as a result, should typically be avoided. The alternative is to adopt a dominance-based criterion; there are three basic types of dominance-based building blocks:

- **Domination (or Dominance Rank)** — by how many entities is an entity dominated? A small number is good. This tends to reward exploration at the edges of the known Pareto front or in new regions.

- **Dominance (or Dominance Count)** — how many entities does an entity dominate? A large number is good. This tends to reward exploitation in the middle of the front.

- **Dominance Layer (or Dominance Depth)** — at which depth front is an entity located? This rewards being in layers (where a layer is defined as entities having a common dominance rank) close to the Pareto front.

From these fundamentals, many metrics have been derived, e.g., NGSA, NSGA-II, SPEA, SPEA2, MOGA, NPGA, DMOEA, VLSI-GA, PDE, PAES, PESA, etc. The summary is that it is possible — although not necessarily trivial — to shift the selection process away from the actual Pareto front to include those individuals near the front and reward uniqueness using the dominance based building blocks.

## Computational Issues

As noted by Jensen (Jensen, 2003), brute force computation of the Pareto dominance of a population of $N$ entities for $M$ objectives will have a $O(MN^2)$ computational bound. For large population sizes, this can dominate algorithm performance. More clever algorithms can reduce that demand to $O(N \log^{M-1} N)$ or $O(N \log^{M-2} N)$ depending upon the Pareto performance metric flavor. However, as a general rule, the computational load effect of population size needs to be considered in the algorithm design.

# 4. Pareto Exploitation: User Selection

There are two potential users of the Pareto front: the human and the algorithm. Although the focus of this chapter is the algorithmic exploitation, we should note that the evolutionary process will identify a Pareto front of discovered solutions. The ability to characterize this front has two major benefits to the user:

- allows the user to focus on the top solutions for inspection and trade-offs of complexity vs. performance and
- provides insight into the problem difficulty by examining the shape of the Pareto front.

Of course, the user also benefits from the improved discovery speed and performance (both accuracy and robustness) which result from the algorithmic exploitation of the Pareto front. An example is shown in Figure 17-4 which displays the Pareto front for a biomass inferential sensor (Kordon et al., 2004). Every dot in this graph represents a GP-model with its associated fitness (in this case $1 - R^2$, i.e. lower values are better) and a normalized complexity measure indicated as the ratio of the number of nodes. Of the 88 models displayed, only 18 are lying on the Pareto front. The number of interesting models is actually even lower since it is clear that little can be gained by having models with a normalized complexity measure larger than 0.3.

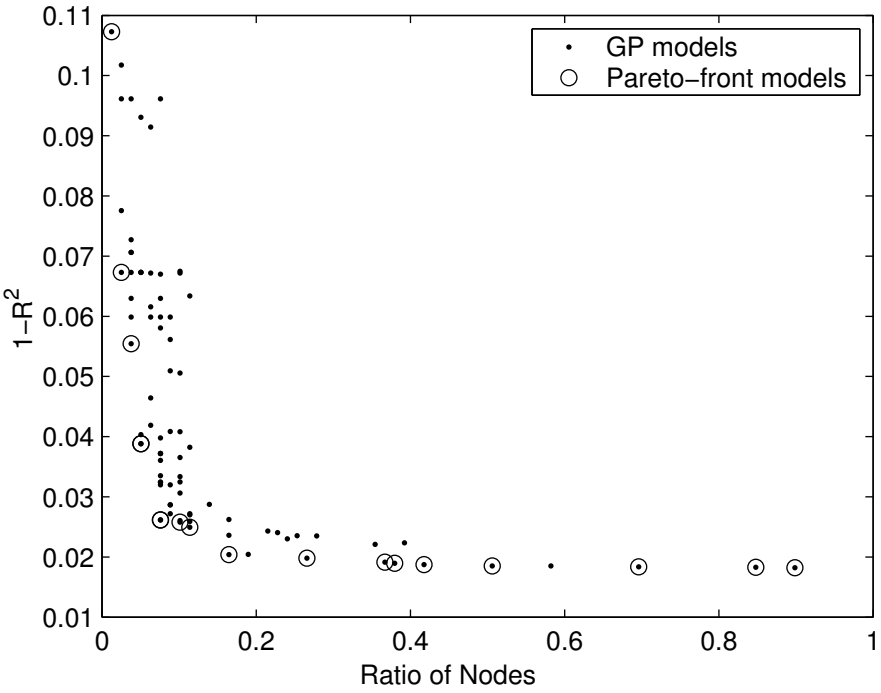


*Figure 17-4.* Here we show the Pareto front of (normalized) complexity vs. performance for a biomass inferential sensor derived using GP. Lower complexity corresponds to lower ratio of the number of constituent nodes; hence, we see the diminishing returns from increases in expression complexity.

## 5. Pareto Exploitation: GP Strategies

The Pareto front representation allows algorithms as well as humans to quickly scan for promising solutions from large populations of expressions and focus subsequent analysis effort on those solutions. In this section, we describe one algorithm to exploit that representation. Although exploiting the multi-objective representation is a relatively unexplored topic, there have been a number of other strategies proposed (e.g., (Bleuler et al., 2001, de Jong and Pollack, 2003, Saetrom and Hetland, 2003)); we will discuss these after presenting the ParetoGP algorithm.

### Algorithm Objectives

The objectives of ParetoGP are threefold:

1 **Multi-Objective Characterization of Evolved Expressions** — Optimizing the Pareto front instead of a single fitness measure (even includ-

ing parsimony pressure) allows the automatic generation of a hierarchy of solutions of increasing complexity and fitness without having to specify a problem-dependent parsimony pressure *a priori*.

2 **Accelerated and Robust Evolution** — Significantly accelerate the search as well as improve the quality of the solutions. ParetoGP significantly alleviates the problem of bloat.

3 **Metavariable Identification** — Use the low-complexity equations as potential variable transforms that can lead to more physical insight.

## The ParetoGP Algorithm

ParetoGP algorithm

In this section we'll discuss a number of aspects in which the proposed algorithm differs from a more conventional GP system. The main features are:

- Pareto front archive definition & persistence across generations and cascaded evolutions
- Generation-based update of the Pareto front archive
- Random crossover between Pareto archive members and generation population ranked by expression accuracy performance
- Random mutation of Pareto archive members

An archive of potential solutions is maintained between generations and between different cascades (except for the persistence of the archive, a cascade is an independent run with a freshly generated starting population). The end result of the computation is the archive which contains all the models on the Pareto front. As discussed before there are several benefits associated with this. One is that the shape of the Pareto front gives the user quite some insight into the intrinsic complexity of the problem. The shape of the Pareto front turns out to be very reproducible across independent runs. Also, the particular choice of the complexity measure (sum of the number of nodes of all subtrees) allows for additional resolution in the description of the Pareto front beyond that offered by a simple node count metric. An additional benefit is that the models at the low-complexity end of the Pareto front very often turn out to be "building blocks" or relatively simple variable transformations that can lead to more physical insight. These building blocks then sometimes can be used to develop linear models with new variables (Castillo et al., 2002). Of course the notion of extracting the Pareto front from a collection of models can also be used to analyze the population resulting from a conventional GP run. Here significant benefits can result from the fast analysis of large collections of models. There

is, however, a significant difference between using a Pareto front as a post-run analysis tool vs. actively optimizing the Pareto front during a GP-run. In the latter case the Pareto front becomes the objective that is being optimized instead of the fitness (accuracy) of the "best" model.

At the end of every generation the archive is updated and contains the Pareto front of the combination of all models and (optionally) all the subtrees of the current population as well as the current archive. By including all the subtrees of all the models in the population (this involves no overhead in our MATLAB™ implementation of the system) we are effectively using a population size equal to the sum of the number of nodes in all equations in the population. Depending on how much complexity we allow, this can lead to effective population sizes which are significantly larger than usual.

In generating the new population, crossover occurs between a random member of the archive and members of the current population. Selection of from the population is based on the conventional GP paradigm of accuracy. Random members of the archive are chosen to maintain diversity in the Pareto front. This is important since we want to develop the entire Pareto front and not bias the search into the low nor the high complexity region. It is important to note that in ParetoGP there is often still a lot of progress even while the model with the highest fitness does not change. This model is just one point at the high-complexity end of the Pareto front.

An archive of potential solutions is maintained not only between generations but also between different cascades. Whereas in a conventional GP system multiple runs are executed starting from scratch i.e. all runs are independent, the archive is maintained between runs in the current system. The result is that while the starting population is regenerated the subsequent generations quickly rediscover the results from the previous runs because of the cross-breeding with the archive. This changes the mode of execution to more runs with less generations compared to a conventional GP-system. The purpose of the independent runs is, therefore, to introduce new genetic material into the Pareto front development.

## Practitioner Comments

Although ParetoGP has already proven itself to be a major addition to the Dow Chemical empirical modeling capability, we are still exploring the features and nuances of its behavior and performance. (Symbolic regression toolboxes have been developed internally in The Dow Chemical company both in Matlab and *Mathematica*.) Maintaining a minimum size archive (nominally 5-10% of the population size) helps the robustness of the symbolic regression in situations where a single variable explains a large portion of the targeted response behavior; if required, the additional models are assembled by adding Pareto layers

until the threshold archive size is met. By construction, each layer has a different accuracy metric which, presumably, avoids inclusion of a more complex version of a model already contained within the archive.

Surprisingly, we often turn off parsimony pressure; however, we do maintain hard limits on expression complexity as a safety. Intuitively, assigning breeding rights to the population based upon accuracy would seem to make the evolution emphasis be model accuracy and, therefore, place the evolutionary effort on the complex side of the Pareto front — which, to some extent, is what we want for the symbolic regression. Using the Pareto archive as half of the breeding pool constrains the complexity of the overall population. Watching the shape of the evolving Pareto front is interesting in the sense that a "tail" of increasing complexity with incremental accuracy gains will be eliminated as a higher performing but much simpler expression is discovered. We also tend to see step changes (major jumps in accuracy) in the Pareto front backfilled in succeeding generations or cascades as the evolutionary process explores this new structure. Evaluating the constituent subtrees may help in this respect by maintaining a bias in the search process for simpler solutions.

The human aspect involved in assessing the evolved models is critical. Hence, we have developed tools for response surface visualization, statistical analysis, error analysis, etc. to facilitate understanding of the developed models.

## Other Pareto Front Exploitation Algorithms

Other researchers have proposed algorithms to exploit the Pareto front. (Bleuler et al., 2001) assign breeding rights based upon the SPEA2 metric of a population with members of the Pareto front persisting across generational boundaries. Their approach is not dependent upon the use of the SPEA2 metric — other than the requirement to have some scalar-valued criteria to award propagation rights. Philosophically, this is a conventional GP approach with the Pareto front used to transfer genetic material across generational boundaries and a selection metric which combines the competing objectives. (Saetrom and Hetland, 2003) have also essentially followed this approach. (de Jong and Pollack, 2003) propose a Pareto front-centric approach wherein they synthesize new entities each generation which are merged with a persistent Pareto front and used to define an updated Pareto front. The Pareto front has three criteria: performance, size and diversity. Genetic propagation is restricted to those entities on the Pareto front.

## 6. ParetoGP Algorithm Performance

In these following figures we compare the results of multiple runs for a polymer reactivity problem. Figure 17-5 shows the results from 300 runs of a conventional GP implementation for 128 generation; notice that conventional

GP was rarely able to generate any solutions with fitness greater than 0.9 for the 300 runs. In contrast, ParetoGP generated superior results with the maximum fitness exceeding 0.9 for almost every case of a similar set of 300 runs as shown in Figure 17-6. Similar results have been confirmed in a number of other projects (Kordon et al., 2004).

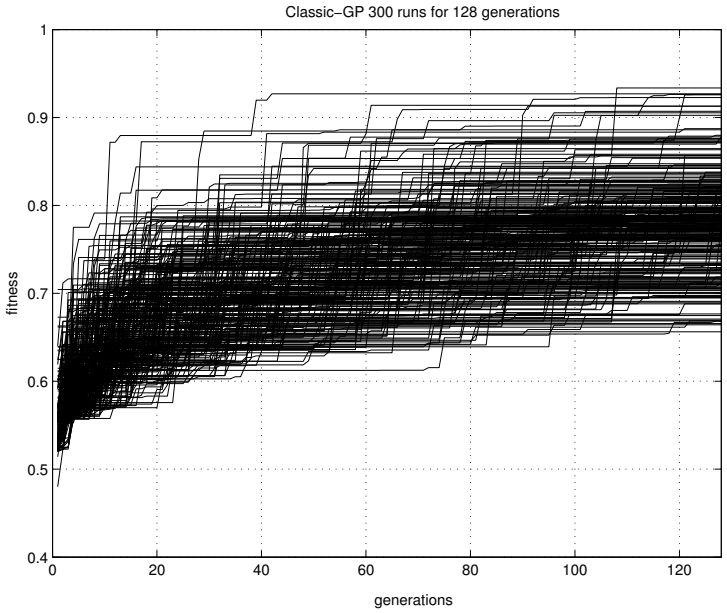


*Figure 17-5.* Example of ten runs of 150 generations with conventional GP for a polymer reactivity problem The highest performance (accuracy) is displayed.

## 7. Conclusions

ParetoGP is a major advancement in Dow Chemical's empirical modeling tool portfolio due to greatly increasing the efficiency and robustness of symbolic regression. In this section we summarize the improvements as well as indicate future research directions.

### Major Improvements Over Classical GP

The advantages of ParetoGP relative to classical GP implementations essentially reduces to (a) symbolic regression speed and robustness and (b) understanding of the complexity-accuracy trade-off of evolved models.

Changing the objective function from a single fitness criterion to the Pareto front of fitness versus a measure of complexity has proven to speed up symbolic regression significantly. Our current estimate is that the entire development

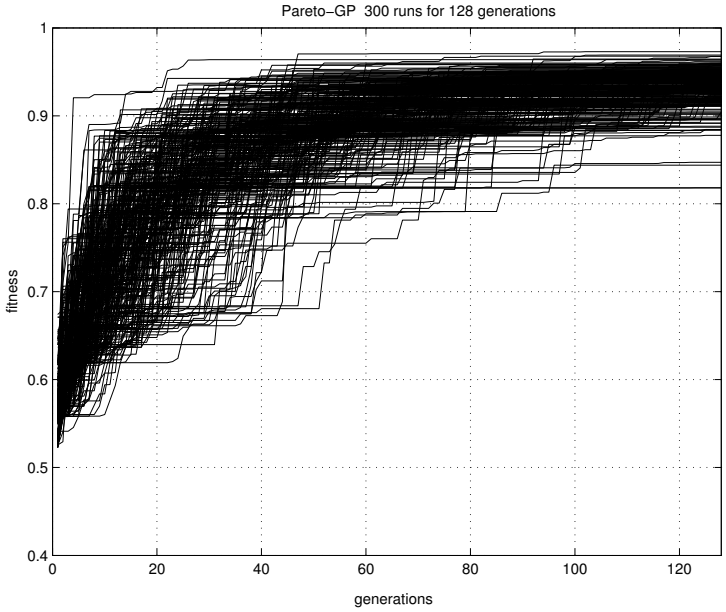


*Figure 17-6.* Example of five runs of thirty generations with Pareto GP for a polymer reactivity problem. The highest performance is displayed.

process speeds up at least tenfold — and even more if one includes the post-analysis phase. Because there are fewer problems with bloat and because of the improved discovery process, bigger problems can be tackled within the same CPU and memory constraints.

The fact that the user can pick one or more functions at the right level of complexity generates more buy-in from the end-user as well as more robust solutions. Also, the natural development of low-complexity transforms between variables at one end of the Pareto front helps to generate more physical insight and end-user buy-in.

## Obvious Extensions

The current implementation of ParetoGP is still basic in many aspects and can be enhanced in various directions.

- **Metavariable identification** — One obvious direction is a more systematic consideration of the discovery of metavariables or building blocks that simplify the evolution process or to identify transforms that linearize the problem. This is currently being investigated.
- **Expression complexity metrics** — Another direction that is being explored is the generation of better complexity measures that not only de-

pend on the tree structure but also take into account the functions that are present at the nodes. Some of these function obviously generate more nonlinearity than others. This is not a simple exercise since this needs to consider the particular scaling of the input variables.

- **Diversity metrics** — More work is probably also needed in developing explicit diversity metrics for the Pareto front. Perhaps this can also be used to direct and make the discovery process even more efficient.
- **Diverse model identification** — As in any empirical modelling effort, there is no such thing as the perfect model so very often the final implementation is built from an aggregate of different models. This model stacking helps us to calculate a "model disagreement" factor that is used to the decide whether the models are interpolating or extrapolating with respect to the original training data set. We have used the Pareto front as a source for these stacked models, but there is also a need for metrics that quantify the diversity in these model sets.
- **Convergence criteria** — Our convergence criteria are currently still based on the fitness of the best individual, but future measures could take the movement or lack of movement of the Pareto front into account to develop better measures.

One last point which is the subject of considerable research is more active control of the parsimony pressure as a function of the particular problem. ParetoGP solves many of the practical problems related to bloat, but there still is a parsimony factor (which is far less critical compared to standard GP) that controls the maximum size of the equations that can be generated. In this sense it is used to control how far the Pareto front extends to the high complexity side.

## References

Banzhaf, Wolfgang, Nordin, Peter, Keller, Robert E., and Francone, Frank D. (1998). *Genetic Programming – An Introduction; On the Automatic Evolution of Computer Programs and its Applications*. Morgan Kaufmann.

Bleuler, Stefan, Brack, Martin, Thiele, Lothar, and Zitzler, Eckart (2001). Multiobjective genetic programming: Reducing bloat using SPEA2. In *Proceedings of the 2001 Congress on Evolutionary Computation CEC2001*, pages 536–543, COEX, World Trade Center, 159 Samseong-dong, Gangnam-gu, Seoul, Korea. IEEE Press.

Castillo, Flor A., Marshall, Ken A., Green, James L., and Kordon, Arthur K. (2002). Symbolic regression in design of experiments: A case study with linearizing transformations. In Langdon, W. B., Cantú-Paz, E., Mathias, K., Roy, R., Davis, D., Poli, R., Balakrishnan, K., Honavar, V., Rudolph, G., Wegener, J., Bull, L., Potter, M. A., Schultz, A. C., Miller, J. F., Burke, E., and

Jonoska, N., editors, *GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1043–1047, New York. Morgan Kaufmann Publishers.

de Jong, Edwin D. and Pollack, Jordan B. (2003). Multi-objective methods for tree size control. *Genetic Programming and Evolvable Machines*, 4(3):211–233.

Eckart Zitzler, Marco Laumanns and Bleuler, Stefan (2004). A tutorial on evolutionary multiobjective optimization. In Xavier Gandibleux, Marc Sevaux, Kenneth Sšrensen and T'kindt, Vincent, editors, *Metaheuristics for Multiobjective Optimisation*, chapter 1, pages 1–32? Springer Verlag.

Jacob, Christian (2001). *Illustrating Evolutionary Computation with Mathematica*. Morgan Kaufmann.

Jensen, Mikkel T. (2003). Reducing the run-time complexity of multiobjective eas: The nsga-ii and other algorithms. *IEEE Transactions on Evolutionary Computation*, 7(5):503–515.

Kordon, Arthur, Jordaan, Elsa, Chew, Lawrence, Smits, Guido, Bruck, Torben, Haney, Keith, and Jenings, Annika (2004). Biomass inferential sensor based on ensemble of models generated by genetic programming. In in process, editor, *GECCO-2004*, page tbd, New York, New York.

Kotanchek, Mark, Smits, Guido, and Kordon, Arthur (2003). Industrial strength genetic programming. In Riolo, Rick L. and Worzel, Bill, editors, *Genetic Programming Theory and Practise*, chapter 15, pages 239–256. Kluwer.

Saetrom, Pal and Hetland, Magnus Lie (2003). Multiobjective evolution of temporal rules. In B. Tessem, P. Ala-Siuru, P. Doherty and Mayoh, B., editors, *Proceedings of the 8th Scandinavian Conference on Artificial Intelligence*. IOS Press.