September 25th - 28th, 2017
INTERFACES
architecture . engineering . science IASS 2017 hamburg

Proceedings of the IASS Annual Symposium 2017
"Interfaces: architecture . engineering . science"
September 25 - 28th, 2017, Hamburg, Germany
Annette Bögle, Manfred Grohmann (eds.)

Choosing parents to produce better performing children: a comparison of selection methods used for evolutionary search

Peter VON BUELOW*

*University of Michigan
Taubman College of Architecture and Urban Planning, Ann Arbor, MI 48130, USA
pvbuelow@umich.edu

Abstract

Architecture problems are by their very nature multivariable. Evolutionary algorithms have been shown to be an effective way to explore solutions to multivariable problems. But what is the most effective way to search for high performing solutions using evolutionary algorithms? In population based search methods, the selection of parents has a direct effect on the children produced. ParaGen is a design exploration method developed at the University of Michigan which makes use of Evolutionary Multiobjective Optimization (EMO) in selecting parents in a multivariable search space. This paper compares two selection methods used by ParaGen: 1. multiobjective selection using SQL queries and 2. selection through a Pareto set. How each of these methods works as well as a comparison of the resulting children is shown. Coordinate plots are used to depict the relative location of selected individuals in the solution space. Image sets are also displayed, which allow for further qualitative selection of solutions based on appearance. Using ParaGen it is demonstrated how final selection refinement can be made to include both quantitative and qualitative architectural criteria.

Keywords: optimization, conceptual design, generative design, multiobjective optimization, concrete shells, genetic algorithms, performance optimization

1. Introduction

ParaGen, under development at the University of Michigan Hydra Lab, is an evolutionary based exploration method intended for early design phases of architectural problems. The method has been described in more detail in preceding papers [4] [5], and can be used for a wide range of applications. The method is cyclic and is configured with a single server and any number of parallel client machines connected through a standard web browser interface. The server maintains a SQL database of all solutions generated, and delivers new child solutions to the local client computers. Each child solution is processed on a client machine to find the geometry using parametric software (such as Grasshopper for Rhino, Generative Components by Bentley Systems or Dynamo by Autodesk, etc.). With the geometry established, any number of simulations can be run on the child solution to discover desired performance data. ParaGen links together any commercial or public domain simulation software to perform the performance analysis. Both the geometry along with performance data is subsequently uploaded back to the server where everything is stored in the SQL database. Because images and files of all sorts can be attached and uploaded, the server is able to deliver graphic as well as performance data to describe a solution. Figure 1. shows the ParaGen cycle between web server and local clients. Designers are then able to access the stored data in a variety of ways and explore the best solutions to the problem.

ParaGen makes use of a Non-destructive Dynamic Population GA (NDDP-GA) [4] to explore the multivariable design space comprised of the parametrically defined geometries. The geometries are analyzed for a set of multiobjective performance criteria which include in this example structural

performance, cost, weight and daylight criteria. The values for the parametric variables which define the generated geometries define the children which are produced in the evolutionary process. In this study, two different methods for the selection of the parents of these children are presented to show the effects that selection has on the overall search process.

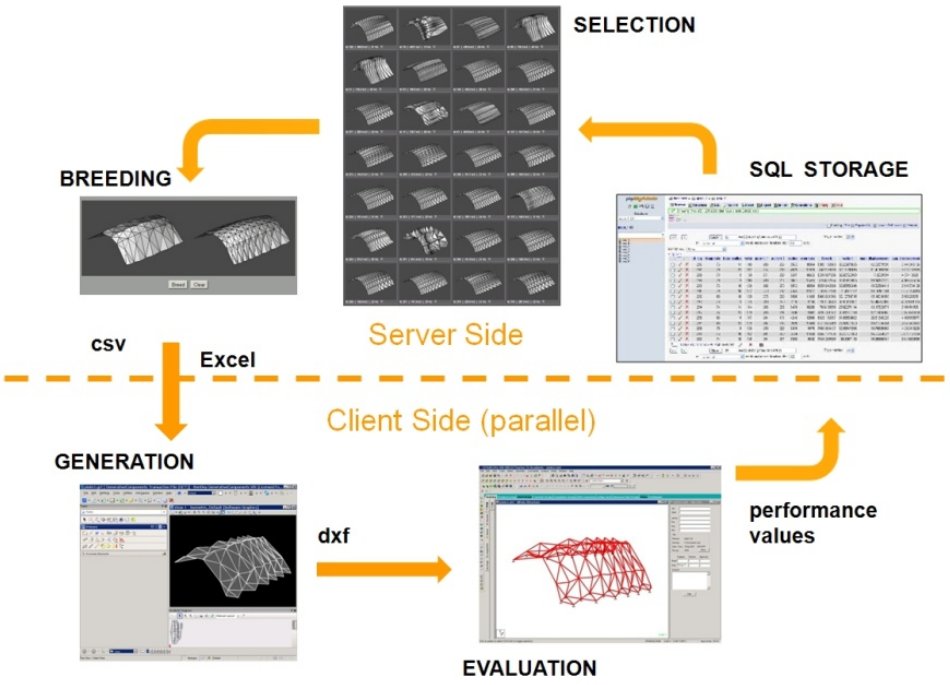

Figure 1: The cycle used in the ParaGen method

ParaGen makes use of a Non-Destructive Dynamic Population Genetic Algorithm (NDDP GA) to build a database of solutions. This approach is functionally a bit different from a traditional GA. Because all solutions are stored in the database, the NDDP GA can pull a population of parents out of that database of all previous solutions *dynamically* as each child is produced. So unlike traditional GAs which select parents from an ever evolving population of a set size, ParaGen actually selects the population first from the database of all previous solutions and then selects the parents from that selected population. The use of the SQL database has several advantages over traditional GAs:

- Prevention of duplicate solutions
- Enables multiobjective search
- Makes possible dynamic populations
- Allows changeable search direction
- Allows interactive search
- Stores Pareto trade-off sets with levels
- Suitable for parallel computation
- Graph data relationships

A description of each of these points is given in another recent paper [5]. This study looks in more detail at aspects related to selection. But of course all of the above aspects are interrelated. Selection for example is what enables “multiobjective search”, and the “dynamic populations” are what get selected.

2. Genetic Selection

How the selection of parents is made is key to how a GA is able to find a particular solution or range of solutions. If selection were merely random, there would be some degree of exploration, but the generation of new solutions would not be directed toward any particular objectives. In ParaGen, since all solutions are saved, given sufficient random solutions, better solutions could of course be searched and selected with the post processing tools (by simply applying SQL queries to the solution database), but to generate solutions which have higher performance, parents need to be strategically selected and bred.

Selection in a traditional GA generally follows one of several methods such as: "roulette wheel", sigma scaling, elitism, Boltzmann selection, rank selection, tournament, steady-state, etc. [3] Each of these methods attempts to varying degrees to converge on better solutions while maintaining a broad sampling of the solution space to prevent getting stuck on local maxima. An NDDP GA could use any of these methods to select parents from the population, although simple random selection of parents generally works fine since the breeding population itself is what has already been selected. Only one set of two parents is selected from a given selected population.

2.1. NDDP GA population selection

More critical for the NDDP GA is the selection of the parent population, rather than the selection of the breeding parents out of that population. The NDDP GA selects the breeding population from the entire database of all solutions found to that point. How this breeding population is selected is the factor that actually guides the search through the solution space. The selection is elitist in that it always chooses the solutions which best match the SQL query. The search maintains diversity by choosing the individual parents randomly, and by increasing the size of the parent population (set with a SQL "limit" parameter).

This study compares two population selection techniques: through SQL queries and through Pareto sets. Actually the two can also be combined which is perhaps more effective. A Pareto set can be constrained or limited through a SQL query. The combination is generally more effective because the extreme ends of the Pareto set are less of a compromise and more of a single objective optimization.

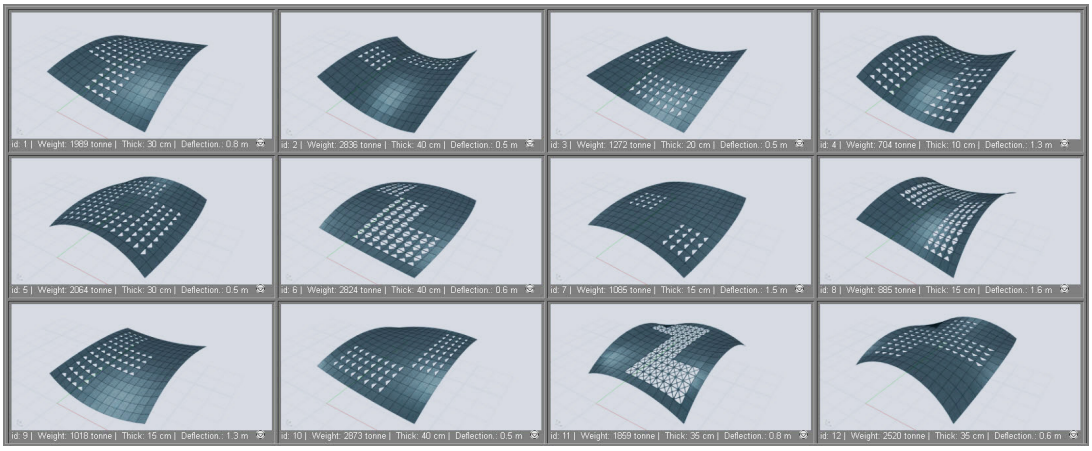

Figure 2: Example of initial random generations – "bird's eye view"

2.2. Initializing the database of solutions

In order to select a population from the database of solutions, the database needs to have some initial solutions added to it. This can be accomplished either through random generation or systematic

selection. ParaGen gives the options either to randomly generate a child solution (within the constraints set for the problem) or mutate a single parent to form a child or breed two parents to form a child. It is also possible for the user to simply supply or modify a child manually. Typically, and most simply, ParaGen is set to generate an initial field of random solutions. Each of these child solutions will be run through the required simulations to calculate the desired performance data. The example used for this study is the exploration of shape and perforation options of a curved concrete shell. A complete and more detailed description of this problem is presented in Emami, et al. [2]. The problem has eight variables which define the geometry. Values for these variables are read into a Rhino/Grasshopper script which both generates the geometry and performance values calculated with Karamba (structural parameters) and DIVA (daylighting) and Archsim (energy parameters). The database table contains a total of 54 rows which include the initial geometry variables, the structural and daylighting performance values, as well as other calculated values relating to cost, construction and design. Figure 2 shows some initial randomly generated solutions.

In this trial, about 400 solutions were randomly generated and about 100 were systematically selected by the user. Figure 3. contains two plots showing a comparison of the random solutions versus the systematically selected solutions.

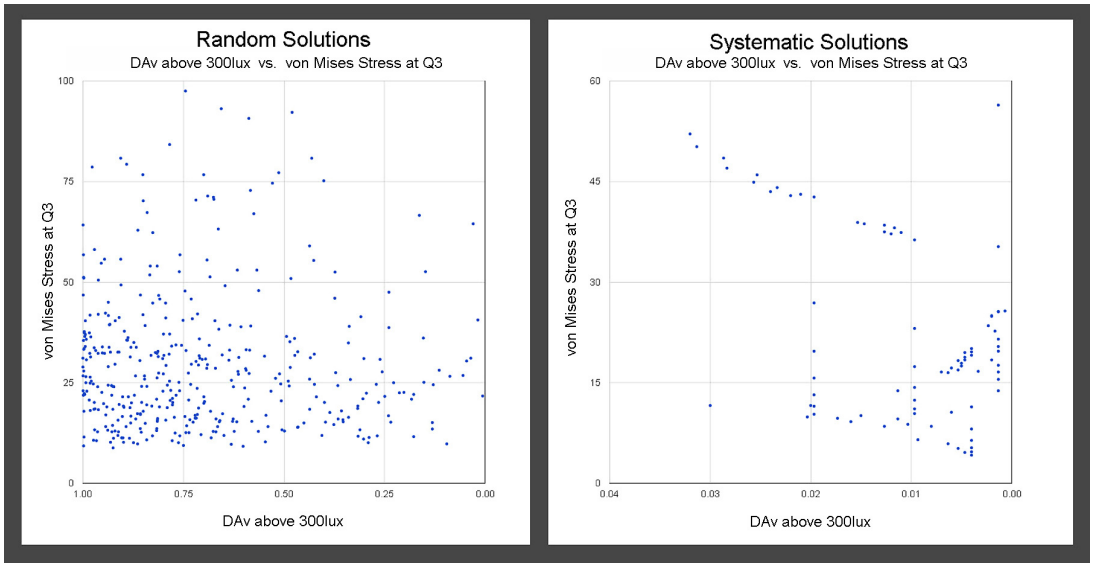

Figure 3: A comparison of initial solutions: randomly generated (left) and systematically created (right)

3. Search by SQL query

Having first generated about 500 solutions both randomly and systematically, the next step is to set the selection criteria to guide further exploration toward more desired solutions. In this case the first selection criteria were chosen through a SQL query. Queries can range from simple single objective searches to more complex, multiobjective exploration with limiting criteria. The query used here combines lighting criteria with some structural parameters. A more detailed explanation can be found in the paper by Emami, et al. [2]. The query found solutions that meet these criteria:

1. Spatial Daylight Autonomy of 300 lux for at least 50% of the year
2. The maximum amount for Daylight Autonomy is taken as 3000 lux (considered oversupply – glare). It is limited to less than 5% of the year

3. Daylight Glare Probability < 4%
4. The third quartile of von Mises stresses across the shell which are less than 30 MP
5. From the set of solutions defined by the first four criteria, the query set is limited to 20 by least weight

This query produced a population of 20 solutions from which two parents were chosen at random to breed one child. Figure 4 shows the solutions bred with this query. As expected the solutions produced a cluster in response to the parameters of the query.

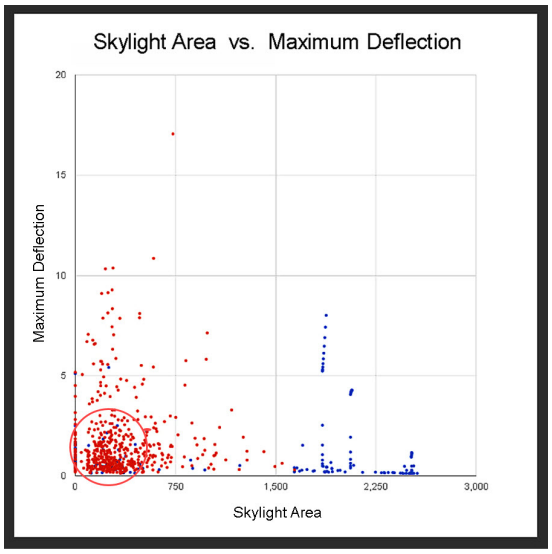

Figure 4: Plot showing solutions found by the SQL query (red dots).

4. Search by Pareto sets

Pareto optimization and the calculation of Pareto sets are a well know approach to multiobjective search especially when dealing with conflicting criteria [1]. A Pareto (level 1) set is composed of solutions which are non-dominated for the given criteria by any other solutions in the population. These are commonly known as best tradeoff solutions. A level 2 Pareto set is found in the same way, but with the level 1 set removed. In this way the width of the solution set can be increased to include more solutions. Figure 5 compares a level 1 solution set with a set which includes levels 1 to 3.

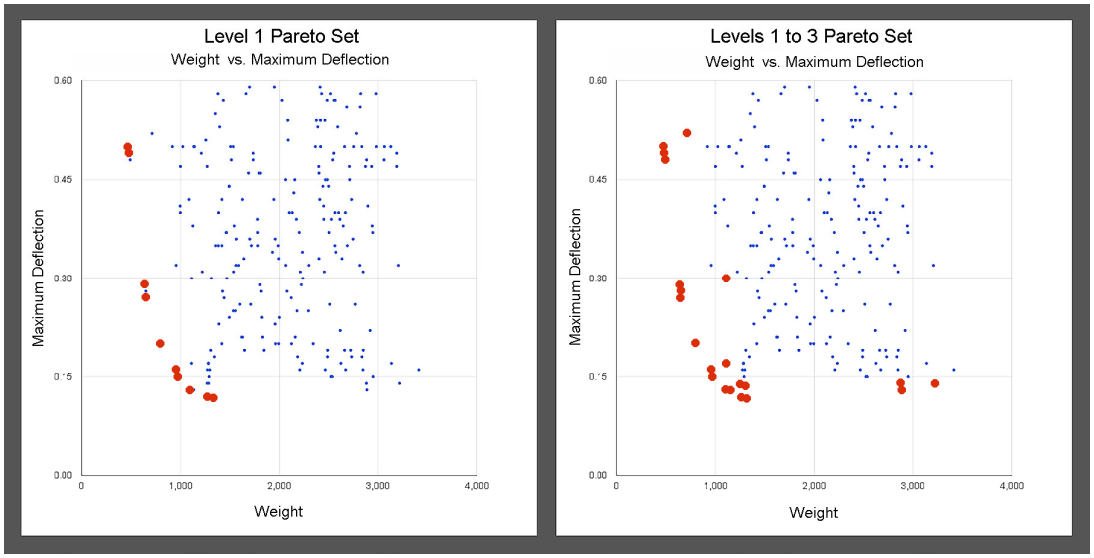

Figure 5: A plot showing in red the Pareto level 1 set (left) and a set including levels 1 to 3 (right)

In order to use the Pareto sets in the breeding populations, the sets need to be first defined and then added to the SQL database. Any number of sets can be defined. A separate program on the server recalculates the Pareto levels for each set each time a new solution is added to the database. The Pareto level of each solution is then recorded along with all the other parameters for that solution in the database. In that way, the Pareto level can be called up and used in the same way any other parameter is used – either in post-processing graphics or in forming a breeding population. This can be useful in that a Pareto set can be plotted in relation to other parameters as well. Figure 6 shows the Pareto set of solutions for weight versus deflection plotted against skylight area of the shell. The red cluster of dots shows the location of the level 1 Pareto solutions in relation to skylight area.

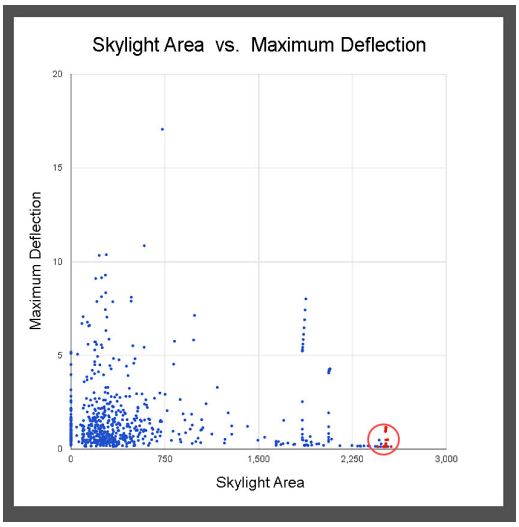

Figure 6: A plot showing the level 1 Pareto set (red dots) for weight vs. deflection in relation to skylight area.

Another advantage of ParaGen is that by clicking on the dots of any graph, the image of the solution is displayed. Figure 7 shows how this can be used to explore the range of a Pareto set.

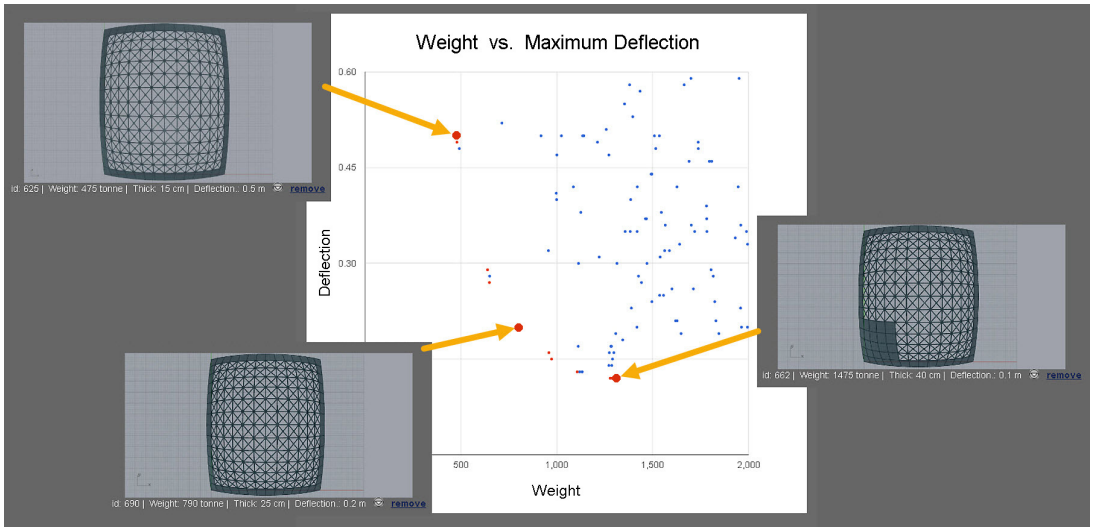

Figure 7: A Pareto plot of minimized weight and deflection, with three solutions selected for viewing

A Pareto set can also be displayed directly in the solution window using any of the image sets collected during the run. Figure 8 shows the same level 1 Pareto set for weight versus deflection. With such image arrays, one can quickly see the relationship between the chosen parameters and the corresponding forms – in this case the open grid shell topology with saddle shape.

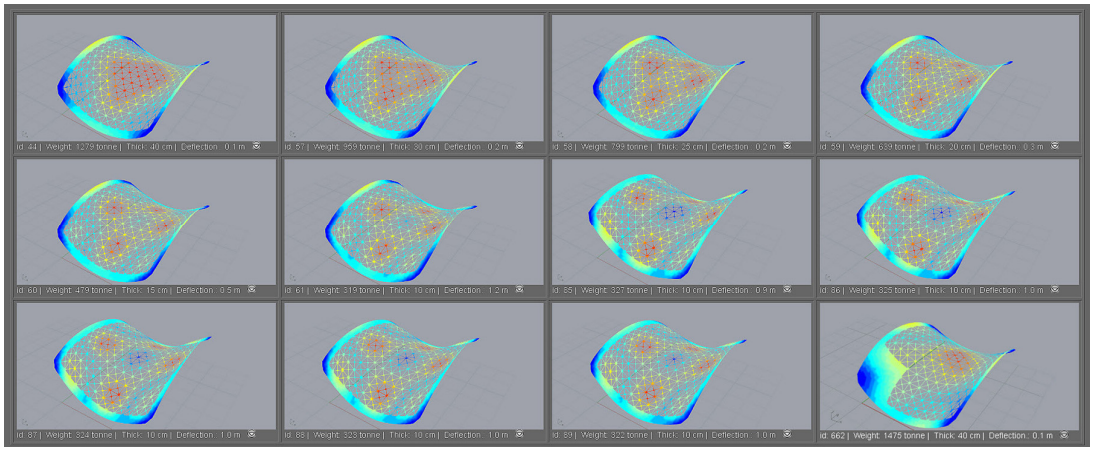

Figure 8: The level 1 Pareto set for least weight versus least deflection. Images show deflection levels.

When used for selecting a population, limiting criteria can be defined for the Pareto set in the same way as they are for other SQL queries. Since the Pareto level number is another column in the database table, it can be combined with any of the other parameters stored in the database. In this way the selection can be limited to the more useful range of the Pareto set which usually means the extreme ends of the set are removed. For example when defining the selection for the above Pareto set of weight versus least deflection, the deflection range was limited to less than 60 cm (about 1/100 of

the 60 m span). In this way, the Pareto set level can be used as any other criteria in forming the selection criteria made by a SQL query.

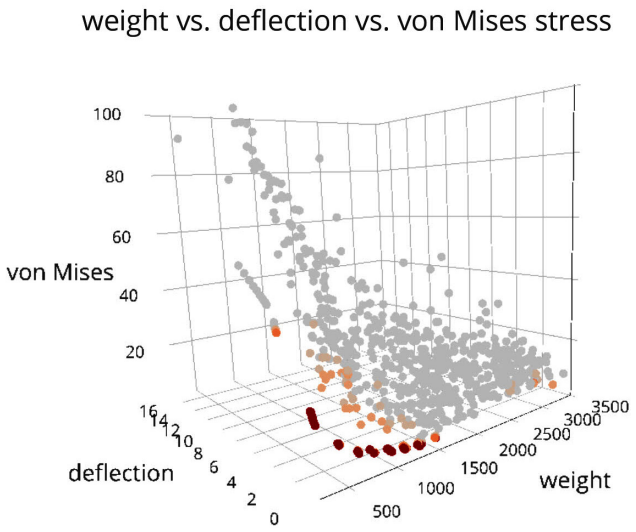

Figure 9: A plot showing a 3-dimensional Pareto set of minimized weight, deflection and von Mises stresses. Level 1 is shown in red with levels 2 and 3 in orange.

It is also possible to define multidimensional Pareto sets. These are less often seen in the literature perhaps because they are harder to illustrate, but also because as more dimensions are compared it becomes harder to decipher the interplay of the individual parameters. As an example, three parameters were chosen: weight vs. deflection vs. von Mises stress. Figure 9 shows a 3D plot of this space with the level 1 Pareto set indicated in red. Higher dimensional sets are also possible.

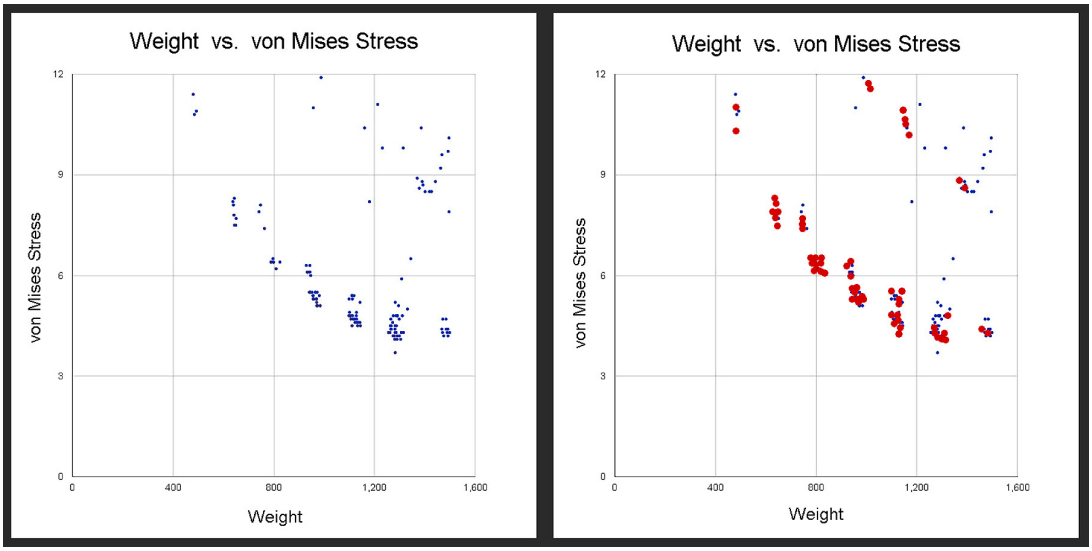

Figure 10. (left) Solutions before Pareto breeding (right) Result of Pareto breeding - red dots show new solutions

Finally the calculated Pareto set itself can be taken as the breeding population. In doing this the number of Pareto levels may also be specified in order to give a broader band of breeding solutions. The result of this breeding is again more solutions similar to the Pareto set. Because the full Pareto set ranges from each extreme of performance tradeoffs, the solutions along the full length of the frontier are not necessarily similar. In NSGA-II the solution is to breed nearest neighbors along the frontier. In the ParaGen implementation individuals along the frontier are chosen and mutated rather than bred. This works well even with a sparsely populated frontier. Figure 10 compares a set of solutions before and after the Pareto optimization. 100 cycles were run using a Pareto level 1 and 2 solutions which minimized both weight and von Mises stresses. The breeding population was further limited to solutions with deflection less than 50 cm. Before the Pareto breeding, there were 38 level 1 Pareto front solutions in this range. Out of the 100 new solutions, 54 ranked as level 1 when compared to the original Pareto set. Of course as new solutions are added they can displace some of the former level 1 solutions, so that in the end the size of the level 1 Pareto frontier increased from 54 to 79. It can be seen in figure 10 that most of the new solutions cluster around the Pareto frontier. Thus the breeding is successful in finding more Pareto optimal solutions.

5. Conclusions

This study has focused on the selection method employed by the NDDP GA used in ParaGen. In an NDDP GA the defining selection is of the breeding population rather than individual parents. The selection is made using SQL queries, but those may include Pareto sets. Pareto sets may be computed at any time during the exploration (from the outset or defined later), but need to be recalculated each time new solutions are added to the database. Once the Pareto levels for a group of parameters are entered in the database, they can be used as any other parameters in the exploration of the solution space. More solutions can be added to the Pareto set simply by using the Pareto set itself as the selected population for breeding. This is show to be a very efficient way to increase the number of Pareto optimal solutions. When building the Pareto set in this way, a focus can be made of certain areas of the set by using additional limiting criteria in the search. The Pareto set definitions combined with other performance parameters give a very effective means of exploring and comparing solutions to complex design problems found in architecture.

References

[1] Deb K, *Multiobjective Optimization using Evolutionary Algorithms*. Wiley, 2001.

[2] Emami N, et al., Continuous to discrete: computational performative design and search of shell structures, in *IASS 2017. Interfaces: architecture. engineering. science*, Bögle A, Grohmann M (eds.), Hamburg, 2017.

[3] Mitchell M., *An Introduction to Genetic Algorithms*. MIT Press, 1998.

[4] von Buelow P., ParaGen: Performative Exploration of Generative Systems. *Journal of the International Association for Shell and Spatial Structures*, 2012; vol. **53**; no. 4; pp. 271-284.

[5] von Buelow P., Genetically Enhanced Parametric Design in the Exploration of Architectural Solutions, in *ICSA 2016. Structures and Architecture - Beyond their Limits*. Cruz P (ed.), CRC Press 2016.