A Generator for Dynamically Constrained Optimization Problems

Gary Pamparà
Department of Industrial Engineering
Stellenbosh University, South Africa
gpampara@gmail.com

Andries P. Engelbrecht
Department of Industrial Engineering, and Computer Science Division
Stellenbosh University, South Africa
engel@sun.ac.za

ABSTRACT

Dynamic constrained optimization problems (DCOPs) provide larger complexity for an optimization algorithm by changing the problem landscape throughout the optimization process. Introducing constraints to an already changing dynamic environment increases the observed complexity of the problem space. Allowing such constraints to have irregular shapes which change along with the problem space itself provides an even greater level of complexity for an optimization algorithm. This paper proposes a function generator capable of creating dynamically constrained dynamic environments by extending the moving peaks benchmark (MPB) function generator. An analysis of the resulting environments produced by the generator is performed using fitness landscape analysis (FLA). A visual inspection of the resulting generated environments is also included.

CCS CONCEPTS

• **Computing methodologies** → **Heuristic function construction**; **Continuous space search**; **Model verification and validation**;

KEYWORDS

Dynamic optimization problem, constraint optimization, Moving peaks benchmark, generator function

ACM Reference Format:
Gary Pamparà and Andries P. Engelbrecht. 2019. A Generator for Dynamically Constrained Optimization Problems. In *Genetic and Evolutionary Computation Conference Companion (GECCO '19 Companion), July 13–17, 2019, Prague, Czech Republic.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3319619.3326798

1 INTRODUCTION

Dynamic optimization problems (DOPs) are difficult problems to solve due to the search landscape of these problems changing over time. The number of different kinds of dynamic optimization problems is potentially limitless, ranging from slight movement of an optimum to drastic relocation and potential change in the optimum value. Moreover, the rate at which such changes occur creates even more DOP types. DOPs are of particular interest because most real-world problems are not static, unchanging problems but instead may maintain conflicting objectives [26] and/or restrict valid problem solutions based on predefined criteria [20].

Literature provides a number of classification schemes [2, 3, 6, 7, 9, 10, 31, 32] for DOPs, where each scheme considers a limited number of DOP characteristics. Duhain and Engelbrecht [8] proposed a combination of the existing classifications to create 27 unique classes of DOPs. With such an extensive set of different DOP types, researchers are able to extensively evaluate algorithms designed to solve DOPs. The moving peaks benchmark (MPB) generator [4–6] provides researchers with a tool to create instances of DOPs, for each of the 27 different classes of DOP. The MPB generator provides the researcher with precise knowledge of the generated search landscape, and how the landscape changes over time. Such knowledge facilitates the analysis and understanding of algorithm behavior with changes in the search landscape.

Current research in the development of DOP benchmark generators focuses on boundary constrained problems. Little research can be found in the development of DOP benchmark generators for constrained problems, whether that be static or dynamic constraints. While constrained DOP benchmarks do exist in the literature [19, 22], none of these provide a systematic approach to model different changes in dynamic constraints, and to combine the 27 DOP classes with the different ways in which constraints change over time.

This paper proposes a constrained DOP generator, where constraints can be static or changing over time. The constrained DOP generator uses the MPB to generate the functional constraints. These functional constraints are then applied to a DOP instance, which is also generated by the MPB, to produce a constrained search space. Fitness landscape analysis (FLA) for constrained problems is used to show how the generated constrained spaces change over time. A visualization for instances of the generated constrained DOPs is also provided.

The main contribution of this paper is the proposed constrained DOP generator, which provides to the researcher an approach to systematically analyse algorithm performance of a wide range of constrained DOP types.

The remainder of the paper presents the following structure: Section 2 discusses DOPs in general, while Section 3 introduces the MPB generator. Constraints within dynamic environments is discussed in Section 4. The constrained DOP generator is presented in Section 5, with the empirical approach followed, discussed in

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
GECCO '19 Companion, July 13–17, 2019, Prague, Czech Republic
© 2019 Association for Computing Machinery.
ACM ISBN 978-1-4503-6748-6/19/07…$15.00
https://doi.org/10.1145/3319619.3326798

Section 6. An analysis of empirical results is presented in Section 7, with Section 8 presenting the conclusion to the paper.

2 DYNAMIC OPTIMIZATION PROBLEMS

Dynamic optimization problems (DOPs) are search spaces where the search landscape changes over time. An environment change modifies optima within the environment, with the modification varying in degree from slight to aggressive. When compared to the previous search landscape, slight modifications produce a similar resultant search landscape. Aggressive modification, however, generally produces an environment with seemingly random characteristics when compared to the previous environment. Due to the sheer number of possible dynamic environments, a classification scheme for environments allows for a grouping of environments, based on specific environment characteristics. Different DOP classification schemes exist, based on the frequency and severity of the changes observed, the type of movement of the observed change, and the trajectory of the change itself.

Eberhart and Shi [9] and Hu and Eberhart [10] describe the changes of an optimum as either:

- Type I, where the value of optima remain the same, but the location of optima within the environment change;
- Type II, where the value of optima change, but the location of optima within the search space remain the same; or
- Type III, where both the position of optima and the value of optima change.

Angeline [2] categorized the movement of an optimum within an environment to be either linear, circular, or random. Duhain and Engelbrecht [8] proposed a classification scheme that considers spatial and temporal severity of changes, resulting in the following DOP classes:

- **Quasi-static** environments which have both low spatial and temporal severity.
- **Progressive** environments which have a low spatial severity, but frequent changes. The changes result in a search space where optima move gradually over time.
- **Abrupt** environments which have infrequent changes, but with a large spatial severity. The problem space remains constant for a period of time before experiencing a large change.
- **Chaotic** environments experience large spatial adjustments that occur at a frequent interval.

Duhain and Engelbrecht [8] proposed that the three classification schemes above be combined to produce 27 different unique DOP classes. Duhain and Engelbrecht [8] also proposed parameterizations of the moving peaks benchmark (MPB) generator to produce instances of each of these 27 DOP classes.

Note that the classification of the different DOP classes above have been proposed for boundary constrained, single-objective DOPs. Section 5 shows how these DOP classes can be applied to include static and dynamic constraints.

3 MOVING PEAKS BENCHMARK PROBLEM

Branke [4, 5, 6] introduced the MPB generator to create DOPs. The MPB generator produces DOPs which contain a predefined set of independent peaks which move through a multi-dimensional problem space. The generator produces these problem landscapes based on a set of input peak functions. The fitness evaluation of the MPB takes the maximum of all the peak functions for a given search space point. The MPB fitness function is therefor defined as:

$$F(\mathbf{x}, t) = \max\{0, p_0(\mathbf{x}, e_0), p_1(\mathbf{x}, e_1), \dots, p_n(\mathbf{x}, e_n)\} \tag{1}$$

where $p_0, p_1, \dots, p_n$ are individual peak functions which evaluate the provided vector, $\mathbf{x}$, and individual peak configuration parameters, e_i, for the current time-step, t. If the provided search space vector $\mathbf{x}$ does not have a fitness determined by a peak fitness function, the function evaluates to a default value of 0.

Individual peaks within the search space define a cone-like shape which have a width, height and location. The peak parameter initialization samples a probability distribution to generate the dimensions of the peak. The construction of peak p_i requires the following parameters, maintained within a "peak environment" record structure, e_i:

- *minHeight* and *maxHeight* are the lower and upper bounds on the size of the peaks themselves;
- *minWidth* and *maxWidth* are the limits on the width of the peak within the problem space;
- Problem domain to determine the bounds for the peak location, $\mathbf{v}$, within the search space with height, h, and width, w. The domain is a vector of intervals defining lower and upper bounds for each dimension of the problem search space; and
- *shift* vector, $\mathbf{s}_v$, with the same dimension of the problem domain; used to influence the movement of a peak.

Formally, the evaluation of an independent peak function is (expanding the peak environment record as the second parameter):

$$p_i(\mathbf{x}, \{\mathbf{v}, h, w\}) = h - w\sqrt{\sum_{i=0}^{|\mathbf{x}|} (\mathbf{x} \odot \mathbf{v})_i} \tag{2}$$

where $\odot$ is a binary operator which determines the component-wise squared difference between the search space position $\mathbf{x}$ and the position of the peak $\mathbf{v}$ within the problem search space.

The MPB generator describes a recurrence relation, whereby subsequent search landscapes may be constructed from the current peak environment records. Using the current peak environment parameters as input, a modification function produces a new set of adjusted peak environments. These updated peak environments allow for the construction of the changed problem landscape. The peak environment modification process requires severity thresholds which scale the magnitude of the adjustments:

- *hSeverity* and *wSeverity* determine the scaling factors for peak height and width adjustments;
- *changeSeverity* (s) is a constant influencing the amount of change of a peak between two environments;
- $\sigma(t) \sim N(0, 1)$; and
- λ is a scaling coefficient on the amount of random peak movement.

The modification process to the parameters of each peak creates a new parameter set for the peak:

$$\begin{aligned}\{e_{new} \mid &\text{height} = e_{old}\{height\} + hSeverity * \sigma(t),\\ &\text{width} = e_{old}\{width\} + wSeverity * \sigma(t),\\ &\mathbf{s}_v = \frac{s}{\|\mathbf{p}_r + e_{old}\{\mathbf{s}_v\}\|}((1-\lambda)\mathbf{p}_r + \lambda e_{old}\{\mathbf{s}_v\})\\ \}\end{aligned} \tag{3}$$

where $\mathbf{p}_r$ is a random vector normalized to length s. Based on the classification of the environment types in Section 2, Van der Stockt and Engelbrecht [27] described the parameterization to the MPB generator to construct each of the 27 unique DOP instances. Although within this formulation of the MPB, peak functions are cone-like which aid in the smoothing of the resultant landscape. Other peak shapes are also feasible, including shapes which are pyramid-like, spherical, and cubic.

4 CONSTRAINED DYNAMIC OPTIMIZATION PROBLEMS

Constraints applied to an optimization problem render parts of the problem search space to be infeasible. The goal of an optimization algorithm operating on an optimization problem is to find solutions which are feasible. When constraints are present within an optimization problem, the constraints allow for two different perspectives for the optimization problem search space. The *objective landscape* determines the quality of a solution through the use of the problem fitness function. Replacing the fitness function with a *violation function* defines the *violation landscape* [18], which quantifies the level of constraint violation within the problem search space and thus the infeasible parts of the problem search space.

A constraint is simply a function which can be dynamic or static. Constrained optimization problem search spaces are the resultant composition of the problem function and the constraint functions. If a constraint function should change, the composed search space would necessarily also experience change. Constraint changes may include shifting an infeasible area to an alternative location within the problem search space; an increase or a decrease of an infeasible area itself; the shape of an infeasible area by change; or a combination of these effects.

Nguyen and Yao [21, 22] described a collection of environments which contain such constraints, collectively terming these environments dynamic constrained optimization problems (DCOPs). DCOP environments may present the following behavior:

- Both the problem search space and the constraints within the environment are dynamic and change over time.
- The constraints remain static, allowing the problem search space to change over time.
- The problem search space remains unchanging, but the constraints change over time.

Due to the interaction of constraints on the search space over time, feasible regions of the problem search space may become infeasible and vice versa. Nguyen and Yao [21, 22] also discussed the characteristics they deem important to characterize a DCOP problem:

(1) constraints may result in changes to the shape, percentage, or structure of feasible and infeasible regions;
(2) the global optima may switch from a disconnected feasible region to another, in problems with disconnected feasible regions; and
(3) environments with a static objective function and changing constraints may expose new, better optima without changing existing optima.

At the time of publication, [22] stated that no benchmark functions existed which fulfill the defined criteria, even though Liu [12] and Richter [25] both provided proposals for benchmark functions. Nguyen and Yao [22] proposed the G24 benchmark of DCOPs. Though the DCOPs in the G24 benchmark suite address the defined criteria, the constrained benchmarks are limited to two-dimensional search spaces.

A generator may be a valid alternative to a new set of benchmark problems. Basing a DCOP generator on an already established and understood DCOP problem generator is advantageous. Allowing such a generator to additionally define constraints on the produced problem will also allow for a more focused understanding of algorithm behavior, when adding constraints to a DOP instance. The generator is also not limited in problem dimensionality, allowing for more than two dimensions.

5 CONSTRAINED MOVING PEAKS BENCHMARK GENERATOR

The MPB generator presents a popular mechanism to create optimization problems for both static and dynamic optimization. Using the classification scheme by Duhain and Engelbrecht [8], a wide variety of DOPs are possible. It is desirable to generate constraints in the same way in which the problem search space is generated, allowing constraints to also vary over time. This section proposes a benchmark generator which creates a dynamic environment that can change over time, and constraints which may, also, change over time. The approach creates a composed problem space from two generated problem spaces, both created by the MPB generator. Each problem space represents a different aspect to the DCOP problem:

(1) The dynamic objective function
(2) The dynamic constraints

The composed function is the difference between the two problem spaces (the MPB generator produces maximization problems), defined as:

$$h(\mathbf{x}) = f(\mathbf{x}) - g(\mathbf{x}) \tag{4}$$

where f is the generated MPB problem defining the base problem and g is another generated MPB problem which defines the violation space. Each generated problem contains a unique set of peaks, with the difference between the two generated problem spaces defining the constrained DOP problem. The composed, resultant problem space has a range within the interval $[-\text{maxHeight}_g, \text{maxHeight}_f]$, but importantly produces a new maximization problem with infeasible regions specified where $h(\mathbf{x}) < 0$.

The parameters that configure the composed MPB generator for each of the objective and constraint MPB problem spaces can be changed at different change frequencies and with different change severities. It should be noted that the generators use different random sequences. Although different random sequences are strictly not necessary, it does allow for a cleaner interaction with the generator modification process.

The composed generator approach has notable advantages:

- The problem and constraint problem spaces are independent from each other. The problem space may remain constant, whilst varying the constraint space, or any combination thereof.
- The dimensionality of the objective problem space and the constraint problem space need not match. For example, a more complex constraint problem space and a less complex objective problem space.
- Plotting the generated composed problem space is trivial (for 2D and 3D visualizations).
- The composition allows for 27 environment types for both the objective problem space and the constrained problem space, yielding 27^2 total DCOP combinations.
- Inclusion of additional equality and inequality constraints to further constrain the resultant composed problem space is still possible.

This problem formulation addresses the previously mentioned criteria of Nguyen and Yao, in that the structure/percentage/shape of the feasible and infeasible regions within the resulting problem space changes over time, optima may appear in different disconnected regions, and changing constraints may reveal better optima in an unchanging problem space. Figure 2 illustrates a composed problem landscape over 9 environment changes.

6 EXPERIMENTAL APPROACH

The purpose of this paper is to introduce a benchmark function generator for constrained (static or dynamic) DOPs. This section describes the implementation of the DCOP benchmark function generator and discusses fitness landscape analysis (FLA) techniques that will be used to show that the generator produces DCOPs where landscape characteristics change over time.

6.1 Software tools

The computational intelligence library (CIlib) [23] is an open-source computational intelligence (CI) software library that provides a pure (containing no undesirable side-effects), monadic, declarative and fully deterministic execution environment. The determinism of the software structures within CIlib encourages implementation transparency and allows for perfect experiment duplication, which is a primary requirement for reproducible research.

Building upon the core CIlib structures, an open-source sister project (also called FLA) implements FLA algorithms. Benchmark functions are also available within the open-source project "benckmarks". All software used for the experimental work are freely available on the internet [1]. Versions for the individual libraries used in this study are: CIlib (version 2.0.1), FLA (version 0.0.3) and benchmarks (version 0.1.1).

6.2 Fitness Landscape Analysis through landscape walks

Fitness landscape analysis (FLA) [24] through landscape walks [14, 17] defines a process whereby potential solutions to a problem search space are sampled at regular intervals across the problem domain. The sampling process yields a multi-dimensional candidate solution for a point within the problem search space. The sampling process results in a collection of neighboring candidate solutions, with the collection referred to as a "walk" through the problem search space. In order to quantify landscape characteristics, fitness landscape metrics require a sufficient sample of candidate solutions taken from problem search space together with phenotypic information for each sampled point. Ideally, the sampled collection of candidate solutions should cover as much of the problem search space as possible.

For the purposes of evaluating the problem search spaces produced by the constrained MPB generator function, only random walks through the problem search space are considered. Starting from a random point on the edge of the problem search space, n consecutive candidate solutions are sampled, with the sampling determined by the walk algorithm. A number of random walks are executed, and the resulting sequence of sample points are used by fitness landscape measures to quantify different characteristics of the search landscape.

For the purposes of this paper, the following fitness landscape measures are used:

- The **feasibility ratio (FSR)** [16] is an approximation of the feasible portion of the problem search space, compared to the total search space. The metric is simply:
$$\mathrm{FSR} = \frac{n_f}{n} \tag{5}$$
where n_f is the number of feasible solutions within a random walk of the search space.
- The **ratio feasibility boundary crossings (RFBx)** [16] determines how disjoint feasible regions of the problem space are. The metric traverses the solutions within a random walk and counts the proportion of solutions that cross between feasible and infeasible space. Transforming the random walk into a binary string b, the bit value 0 indicates that the solution is in feasible space, whereas a bit value of 1 represents infeasible space. The RFBx is calculated as:
$$\mathrm{RFBx} = \frac{\sum_{i=1}^{n-1} cross(i)}{n-1} \tag{6}$$
where
$$cross(i) = \begin{cases} 0 & \text{if } b_i = b_{i-1} \\ 1 & \text{otherwise} \end{cases} \tag{7}$$
- The **dispersion** [13] metric estimates the global topology of a problem search space from the provided solution information within the random walk. The dispersion for a random walk determines how spread out the points are in relation to each other.
- The **first entropic measure (FEM)** [28–30] determines the classification of the problem search space as either rugged, smooth or neutral, using the search space information between neighbouring solutions. The result of the measure is a value within the range [0.0, 1.0], which indicates a search space from smooth to total ruggedness.
- The **fitness cloud index (FCI)** [15] indicates the evolvability of an evolutionary search. The metric makes use of a

particle swarm optimization (PSO) algorithm to determine the neighbourhoods of a sample of solutions, and normalizes all fitness values of the sample considering only solutions that are within the bounds of the problem search space. The resulting value is within the range [0, 1], where 0 indicates the worst searchability and 1 the best searchability.

- The **fitness distance correlation (FDC)** [15] metric informs whether the information presented by a problem space could guide an optimization algorithm to an optimum. The premise is that a problem is simple to search if the fitness of the solutions increases/decreases as the distance to an optimum increases/decreases. The correlation value may range from -1.0 (totally uncorrelated) to 1.0 (totally correlated).
- The **gradient measures** [14] estimate the steepness of the gradients present within a problem search space. Three different gradient metrics exist. The first is the average gradient, G_{avg}, which provides an indication of the average gradient between neighbouring solutions within a random walk. The second metric determines the standard deviation of the walk gradients, G_{dev}. Larger values for the deviation measure indicate the presence of steep gradient changes, such as cliffs, peaks, or valleys in contrast with the rest of the neighboring solutions. The last gradient measure is the maximum gradient, G_{max}, which provides the largest estimated gradient within a random walk. If G_{max} is larger than G_{avg} for a given random walk through the problem search space, then there are parts of the problem space that stand out from the remainder of the problem space. Larger values for both the average gradient and the maximum gradient indicate a rugged problem space.

A total of 30 random walks were executed for each of the 10 generated constrained search spaces. Each generated search space maintained 10 peaks within the problem space function and in the constrained MPB function. Once the main problem landscape is composed with the constraint problem landscape, the effect of composition may remove or add peaks in the final problem landscape. It is important to stress that the same random walks were used for all of the generated problem spaces. The Manhattan progressive random walk [11] algorithm was used.

7 RESULTS

This section presents and discusses the fitness landscape characteristics of constrained DOPs generated by the proposed benchmark generator. Figure 1 illustrates the fitness landscape measurements over the 10 generated problem landscapes, whilst Figure 2 provides a visualization of each individual problem landscape. The visualization labels the consecutive generator problem landscapes, 0 to 9, with the initial generated problem landscape instance labeled as 0.

With reference to the FSR metric, illustrated in Figure 1, the generated problem landscapes display a ratio of feasibility within the range $(0.38, 0.86)$. The visualized landscapes in Figure 2 confirm the changing feasible and infeasible regions of the generated problem space; the black regions within the figure indicate infeasible regions, which change in size and connectedness as the problem landscape transforms from the initial problem space to the final generated problem space. The random seed of $123456789L$ generated the initial problem space.

The RFBx metric displays minimal variation in the obtained values, which indicates that feasible regions are not disjoint. The visualized problem landscapes of Figure 2 confirm that the metric does not reflect that the generated problem spaces actually do contain disjoint feasible regions. The domain of the generated problems is large and the random walks are simply not effective, by excluding the exploration these areas of disjoint feasibility. Using more random walks with longer sampling lengths will cover more of the problem search space.

The dispersion metric results in a similar value for all of the generated problem landscapes. The dispersion value is indicative of a good spread of solutions within the problem space.

The ruggedness of the generated problem spaces, as provided by the FEM metric, indicates that the generated problem spaces vary from marginally to slightly rugged. These values confirm the expectations for the generation of MPB problem landscapes, because the MPB is a flat plane with super-imposed peaks onto the problem space. Using more peaks as input to the problem generator will result in a more rugged generated landscape.

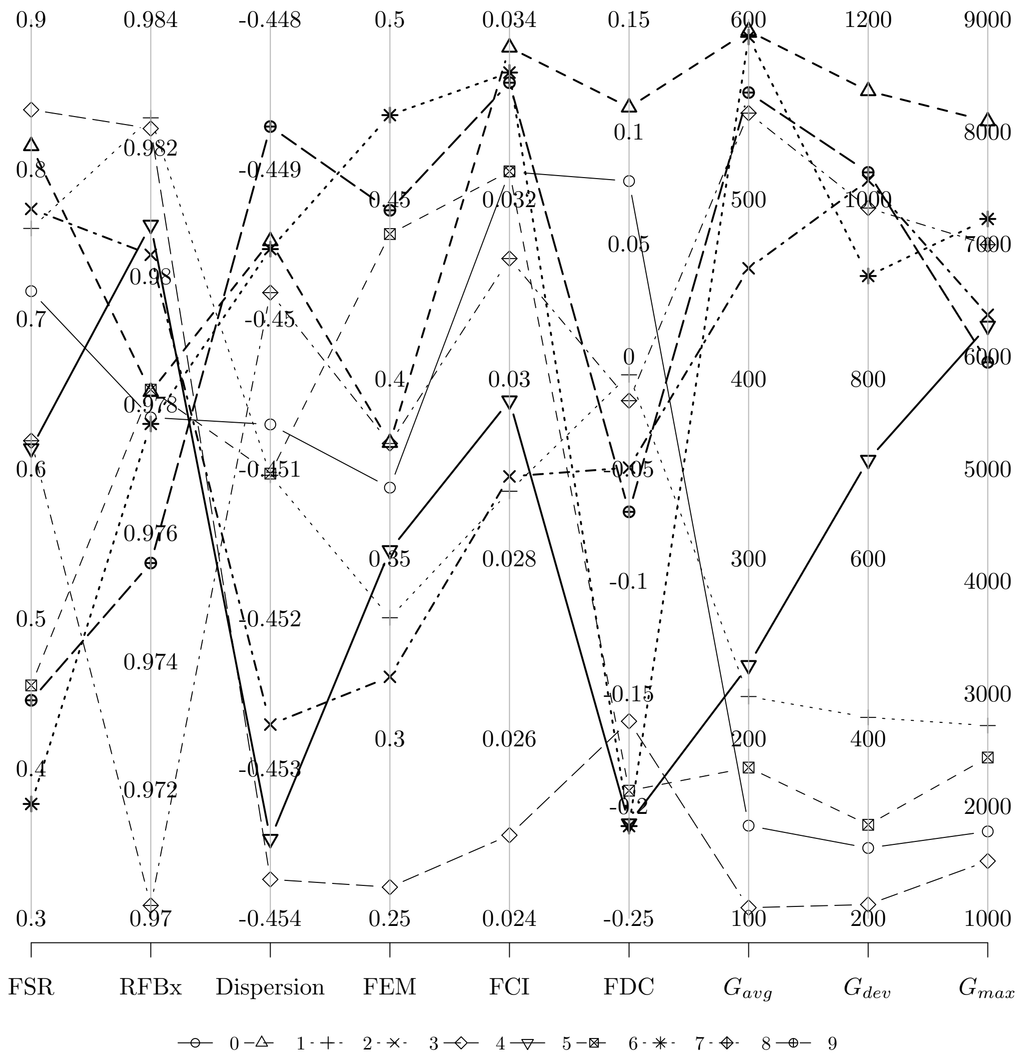
Searchability of the generated problem spaces, provided by the FCI metric, indicates that the generated problem landscapes do not aid in the searchability of the problem itself. This is an expected result, as the disjointedness of the feasible regions, both in and outside of infeasible spaces, do not guide the algorithm to a better solution.

The FDC metric shows that the generated problem landscapes display correlations close to 0, indicating marginal correlations between the observed fitness values and the distance to optima. These values suggest that the information present in the problem search space does not aid an algorithm to a single optimum value, but instead presents unbiased alternate solutions.

The gradient metrics all display a similar trend, that of generally larger gradients. The results are not surprising based on the definition of the MPB and that the function generator has a degree of randomness which may generate large peaks with narrow widths. The gradient metrics all display large values which is indicative of large gradient changes in the problem space.

8 CONCLUSION

This paper proposed a benchmark function generator for dynamically constrained DOPs, based on the MPB function generator. The function generator, which is a composition of two different dynamic changing functions, can create a variety of dynamically constrained optimization problems. The resulting environments address concerns from earlier research with respect to the location of feasible regions, the movement of constraints, and providing for dynamism of both the problem and constrained spaces of the final problem space. A fitness landscape analysis of an instance of the dynamic constrained DOP generator, with 10 environments, showed that the resulting environments were robust and diverse. Visualization of the 10 environments confirmed the results collected from the fitness landscape measures. As a result, the dynamically



RFBx = Ratio feasibility boundary crossings
FEM = First entropic measure
FCI = Fitness cloud index
FDC = Fitness distance correlation
G_{avg} = Average walk gradient
G_{dev} = Walk deviation
G_{max} = Maximum gradient in walk

Figure 1: Fitness landscape characteristics of a dynamically constrained DOP

constrained DOP generator provides a configurable, well understood, dynamic, constrained environment generator which will allow for better understanding of the benchmark problems used to analyze the performance of optimization algorithms. Future work will include a detailed investigation on the use of different peak types for the problem and constraint component functions and the influence thereof. Sets of generator function parameters will be defined which create problem landscapes that address specific aspects of optimization algorithm effectiveness, allowing for fair algorithm performance comparisons on the same problem instance.

REFERENCES

[1] [n. d.]. CIRG GitHub Organization. https://github.com/cirg-up. Accessed: 2018-06-01.
[2] P.J. Angeline. 1997. Tracking Extrema in Dynamic Environments. In *Proceedings of the International Conference on Evolutionary Programming*. Springer, 335–345.

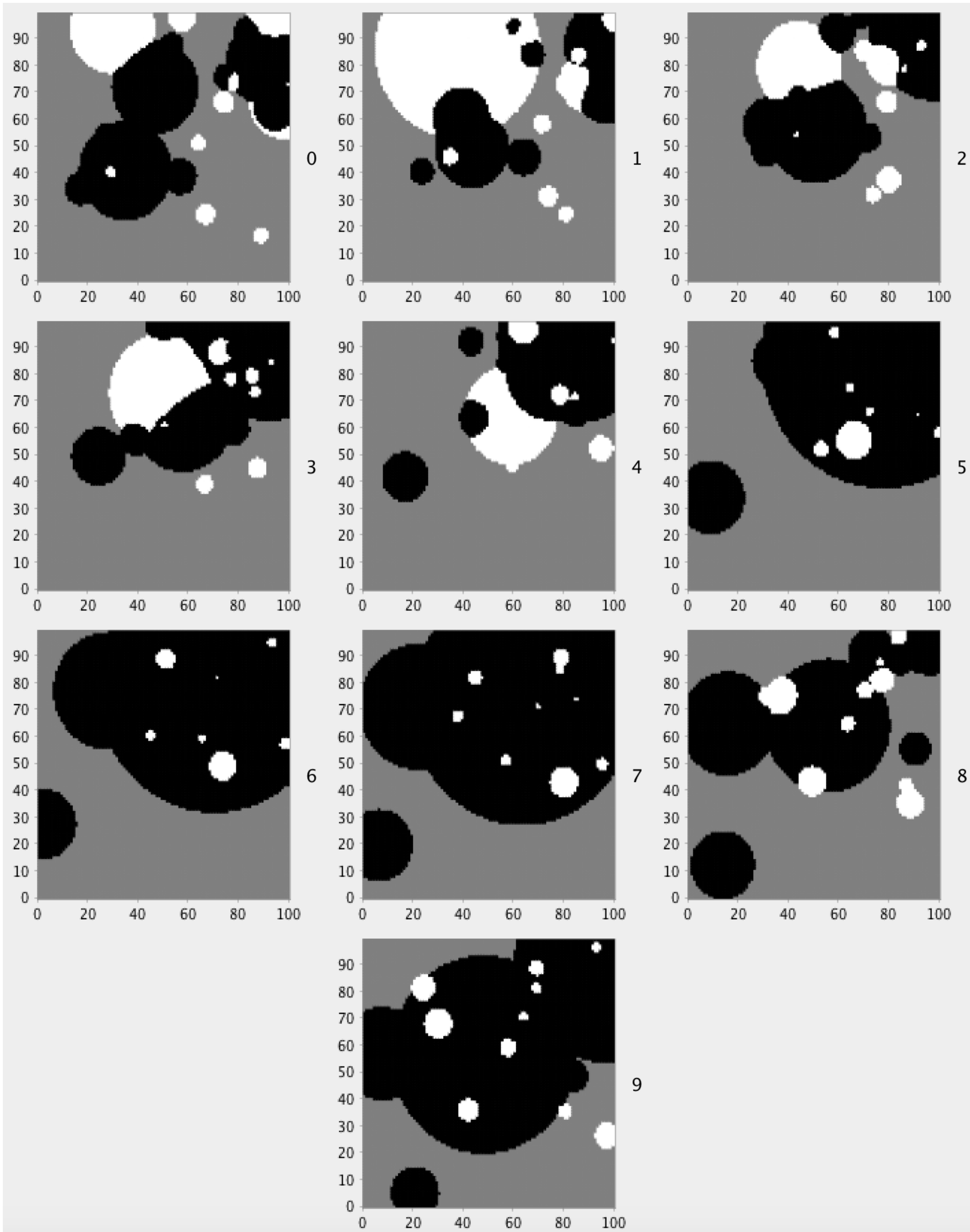


Figure 2: Constrained DOP instance with nine consecutive environment changes. Black colored regions are infeasible ($h(\mathbf{x}) < 0$). Gray regions indicate feasible regions without solutions ($h(\mathbf{x}) = 0$); white regions are feasible with solutions present ($h(\mathbf{x}) > 0$).

[3] T. Back. 1998. On the behavior of Evolutionary Algorithms in Dynamic Environments. In *Proceedings of the IEEE International Conference on Evolutionary Computation*. 446–451. https://doi.org/10.1109/ICEC.1998.699839

[4] J. Branke. 1999. Evolutionary Approaches to Dynamic Optimization Problems - A Survey -. 134–137.

[5] J. Branke. 1999. Memory Enhanced Evolutionary Algorithms for Changing Optimization Problems. In *Proceedings of the Congress on Evolutionary Computation*, Vol. 3. 1875–1882. https://doi.org/10.1109/CEC.1999.785502

[6] J. Branke. 2001. *Evolutionary Optimization in Dynamic Environments*. Kluwer Academic Publishers, Norwell, MA, USA.

[7] K. De Jong. 1999. Evolving in a changing world. In *Proceedings of Foundations of Intelligent Systems*, Z. W. Raś and A. Skowron (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 512–519. https://doi.org/10.1007/BFb0095139

[8] J.G.O.L Duhain and A.P. Engelbrecht. 2012. Towards a more complete classification system for dynamically changing environments. In *Proceedings of the IEEE Congress on Evolutionary Computation*. IEEE, 1–8.

[9] R.C. Eberhart and Y. Shi. 2001. Tracking and Optimizing Dynamic Systems with Particle Swarms. In *Proceedings of the IEEE Congress on Evolutionary Computation*, Vol. 1. 94–100. https://doi.org/doi:10.1109/CEC.2001.934376

[10] X. Hu and R.C. Eberhart. 2001. Tracking Dynamic Systems with PSO: Wheres the Cheese. In *Proceedings of the Workshop on Particle Swarm Optimization.* 80–83.

[11] E. F. Krause. 2019. *Taxicab geometry : an adventure in non-Euclidean geometry.*

[12] C. Liu. 2008. New Dynamic Constrained Optimization PSO Algorithm. In *Proceedings of the Fourth International Conference on Natural Computation*, Vol. 7. 650–653. https://doi.org/10.1109/ICNC.2008.742

[13] M. Lunacek and D. Whitley. 2006. The Dispersion Metric and the CMA Evolution Strategy. In *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation (GECCO).* ACM, New York, NY, USA, 477–484. https://doi.org/10.1145/1143997.1144085

[14] K.M. Malan. 2014. *Characterising Continuous Optimisation Problems for Particle Swarm Optimisation Performance Prediction.* Ph.D. Dissertation. University of Pretoria.

[15] K.M. Malan and A.P. Engelbrecht. 2014. Characterising the Searchability of Continuous Optimisation Problems for PSO. *Swarm Intelligence* 8, 4 (01 Dec 2014), 275–302. https://doi.org/10.1007/s11721-014-0099-x

[16] K.M. Malan and I. Moser. 2018. Constraint Handling Guided by Landscape Analysis in Combinatorial and Continuous Search Spaces. (03 2018), 1–23.

[17] Katherine M. Malan and Andries P. Engelbrecht. 2013. A survey of techniques for characterising fitness landscapes and some possible ways forward. *Information Sciences* 241 (2013), 148 – 163. https://doi.org/10.1016/j.ins.2013.04.015

[18] K. M. Malan, J. F. Oberholzer, and A. P. Engelbrecht. 2015. Characterising constrained continuous optimisation problems. In *Proceedings of the IEEE Congress on Evolutionary Computation.* 1351–1358. https://doi.org/10.1109/CEC.2015.7257045

[19] R. Mallipeddi and P. Suganthan. 2010. Problem definitions and evaluation criteria for the CEC 2010 Competition on Constrained Real-Parameter Optimization.

[20] Zbigniew Michalewicz, Dipankar Dasgupta, Rodolphe G. Le Riche, and Marc Schoenauer. 1996. Evolutionary algorithms for constrained engineering problems. *Computers & Industrial Engineering* 30, 4 (1996), 851 – 870. https://doi.org/10.1016/0360-8352(96)00037-X

[21] T.T. Nguyen and X. Yao. 2009. Benchmarking and Solving Dynamic Constrained Problems. In *Proceedings of the IEEE Congress on Evolutionary Computation.* 690–697. https://doi.org/10.1109/CEC.2009.4983012

[22] T.T. Nguyen and X. Yao. 2012. Continuous Dynamic Constrained Optimization - The Challenges. *IEEE Transactions on Evolutionary Computation* 16, 6 (Dec 2012), 769–786. https://doi.org/10.1109/TEVC.2011.2180533

[23] G. Pamparà, F. Nepomuceno, and B. Leonard. 2014. CIlib. https://doi.org/10.5281/zenodo.12371

[24] Erik Pitzer and Michael Affenzeller. 2011. *A Comprehensive Survey on Fitness Landscape Analysis.* Vol. 378. 161–191. https://doi.org/10.1007/978-3-642-23229-9_8

[25] H. Richter. 2010. Memory Design for Constrained Dynamic Optimization Problems. In *Proceedings of the International Conference on Applications of Evolutionary Computation*, C. Di Chio, S. Cagnoni, C. Cotta, M. Ebner, A.ó Ekárt, A.I. Esparcia-Alcazar, C. Goh, J.J. Merelo, F. Neri, M. Preuß, J. Togelius, and G.N. Yannakakis (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 552–561. https://doi.org/10.1007/978-3-642-12239-2_57

[26] Patrick D. Surry, Nicholas J. Radcliffe, and Ian D. Boyd. 1995. A multi-objective approach to constrained optimisation of gas supply networks: The COMOGA method. In *Evolutionary Computing*, Terence C. Fogarty (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 166–180.

[27] S. Van der Stockt and A. P. Engelbrecht. 2018. Analysis of Selection Hyper-Heuristics for Population-based Meta-Heuristics in Real-valued Dynamic Optimization. *Swarm and Evolutionary Computation* 43 (2018), 127–146. https://doi.org/10.1016/j.swevo.2018.03.012

[28] V.K. Vassilev. 2000. *Fitness Landscapes and Search in the Evolutionary Design of Digital Circuits.* Ph.D. Dissertation. Napier University.

[29] V. K. Vassilev, T. C. Fogarty, and J. F. Miller. 2000. Information Characteristics and the Structure of Landscapes. *Evol. Comput.* 8, 1 (March 2000), 31–60. https://doi.org/10.1162/106365600568095

[30] V. K. Vassilev, T. C. Fogarty, and J. F. Miller. 2003. *Smoothness, Ruggedness and Neutrality of Fitness Landscapes: from Theory to Application.* Springer Berlin Heidelberg, Berlin, Heidelberg, 3–44. https://doi.org/10.1007/978-3-642-18965-4_1

[31] K. Weicker. 2000. An Analysis of Dynamic Severity and Population Size. In *Proceedings of Parallel Problem Solving from Nature PPSN VI*, M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. J. Merelo, and H. Schwefel (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 159–168.

[32] K. Weicker. 2002. Performance Measures for Dynamic Environments. In *Preceedings of the Conference on Parallel Problem Solving from Nature*, J. J. M. Guervós, P. Adamidis, H. Beyer, H. Schwefel, and J. Fernández-Villacañas (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 64–73.