The Thirty-Third International
FLAIRS Conference (FLAIRS-33)

The Objective of Simple Novelty Search

R. Paul Wiegand
School of Modeling, Simulation, & Training
University of Central Florida
Orlando, FL

Abstract

Novelty search is a powerful tool for finding complex objects in complicated, open-ended spaces; however, there is very little foundational analysis of how novelty search works. In this paper, we consider a simplified version of novelty search that focuses entirely on how the sparseness metric and the archive update criterion affect search. We find that, once one sees that search in novelty search happens at the level of the archive space, not the individual point space, it is clear that the sparseness measure and archive update criterion create a process that is driven by a clear pair of objectives: spread out to *cover* the space, while trying to remain as efficiently *packed* as possible. Our Simple Novelty Search Evolutionary Algorithm (SNSEA) is driven to converge to an ϵ-net in the sense defined by k Nearest Neighbor (k-NN) theory. We provide constructive advice for balancing mutation and the sparseness criterion, as well as how to monitor convergence in novelty search.

Introduction

Evolutionary computation can be a powerful problem solver (De Jong 2006; Mitchell 1997; Michalawicz 1996; Goldberg 1989; Holland 1975); however, increasingly researchers are leveraging these biologically-inspired approaches for reasons that transcend straightforward optimization. Evolutionary algorithms (EAs) are now used in a wide array of creative endeavors from creating computer programs (Langdon and Poli 2002; Banzhaf et al. 1998; Koza 1992) to helping find faults (Hanes and Wiegand 2019; Elhadef and Ayeb 2000) to producing art (Dreher 2014; Secretan et al. 2008; Romero and Machado 2007; Dawkins 1996). Indeed, there is an entire branch of evolutionary computation dedicated to its ability to generate, create, and innovate (Goldberg 2002). Unfortunately, there is not a lot of theory or foundational analysis for such applications.

One of the most successful and aggressively promoted concepts springing from innovative power of evolutionary methods is the idea of *Novelty Search* (Stanley and Lehman 2015; Lehman, Stanley, and Miikkulainen 2013; Lehman and Stanley 2011b; 2008). The basic notion behind novelty search is that for certain complex spaces, it might be better to *ignore* the optimization objective and instead *explore* the space by finding objects that are increasingly "different" from those that the search process has encountered before. Though perhaps counter intuitive, novelty search has been surprisingly effective at finding good solutions to a number of challenging problems without even looking for those solutions explicitly, for instance discovering effective grasping behaviors for highly articulated robotic arms (Huang et al. 2014), unsupervised feature learning for deep networks (Szerlip et al. 2015), discovering of sophisticated gaits for quadruped locomotion (Morse et al. 2013), and goal-seeking in complex multiagent simulations (Lehman and Stanley 2011a).

Though there is very little theoretical analysis for novelty search, proponents of these methods suggest at least three ideas for *why* novelty search has been so successful. The first basic premise is that novelty search avoids traps and deceptive local optima by abandoning the objective in favor of a novelty metric (Lehman and Stanley 2011a). Second, novelty search is typically used in conjunction with open-ended or generative representations (Lehman and Stanley 2008) to encourage a process referred to in the literature as "complexification" (Stanley and Miikkulainen 2004) — that is, systematic and steady increase of complex representations via evolutionary search in conjunction with a search geared to find novel items will lead to surprising and nuanced discoveries. Finally, by employing distance metrics within the final solution space (e.g., robot behaviors) rather than the genotype space, novelty search finds what is interesting in the space on which the designers are focused.

This paper focuses on the first proposed idea for how novelty search works. Specifically, this paper is a preliminary examination of the claims that novelty search has no objective, and that it is a "divergent" search process (Lehman, Wilder, and Stanley 2016; Lehman and Miikkulainen 2015; Stanley and Lehman 2015). We find neither claim is true in the general sense: under the most basic components of traditional novelty search (sparseness and an archive) the archive is driven toward increasingly better ϵ-covers of the search space while also trying to optimize the ϵ-packing of the archive. Simple variants of novelty search that employ the traditional mechanisms can and do converge, even when the space is unbounded. Understanding this allows us to give some constructive advice for how to apply novelty

search more effectively: it is essential to balance the mutation rate and the minimum sparseness criterion appropriately, and there may be value in periodic archive resets.

We are not claiming that novelty search is a poor method, nor that novelty search *cannot* diverge in certain situations. Moreover, we recognize that our simplified variant of novelty search captures only a part of what more traditional implementations involve — most notably, we draw parents directly from the archive and do not maintain a population. Finally, we are deliberately isolating the claim as to whether novelty search is objectiveless, and we say nothing at all about complexification or discovery in behavioral spaces. Instead, this paper serves as an indicator that more foundational study of methods like novelty search is needed.

Technical Approach

Sparseness and Archives

The *sparseness* of some candidate object y, ρ_y, is computed as the average distance to the k nearest neighbors in some set A:

$$\rho_y := \frac{1}{k} \sum_{i=1}^{k} \delta\left(x_i^A, y\right), \quad (1)$$

where x_i^A is the i^{th} closest point in set A to y and δ is some kind of distance calculation.

For traditional novelty search, this sparseness score is used in two ways. The first is as a fitness value for evolution: parent and/or survival selection can be based on maximizing sparseness. In such cases, the set under consideration is the population itself. The second is as an update mechanism for an *archive* of points the search is maintaining: Novelty search adds individuals to the archive only if their sparseness over that archive is above a certain threshold, ρ_{min}. In this case, the set under consideration is the archive.

Though there are many choices for search operators and selection, the rest of the method is essentially an evolutionary algorithm (De Jong 2006). The archive continues to expand as the search progresses, and it is this fact combined with the drive toward new, novel search points that gives rise to the notion that novelty search "diverges".

Archive-Based Searches

In a traditional application of an evolutionary algorithm, each individual in the population typically represents a candidate solution to some problem. That is, in a traditional EA application, the space being searched is the space of candidate solutions. From this perspective, it is easy to understand why novelty search *appears* to be objectiveless: though individuals may be encoding candidate solutions, the algorithms pay no attention to the optimization objective associated with that candidate solution. We imagine search diverging in the sense that increasingly different candidate solutions are progressively proposed in an ever-growing archive.

However, there are alternative ways to understand search, and some search methods are inherently focused on searching the space of *archives*, not individual points. The multiobjective optimization problem is (typically) to find an approximation of the Pareto non-dominating *set* for some solution space (Seada and Deb 2018; Zitzler 2012; Zitzler, Deb, and Thiele 2000). Further, *cooptimization* problems (those approached by coevolutionary algorithms, for example) are supersets of multiobjective optimization problems, where the goal is to simultaneously identify the set of objectives and produce a set obeying some solution concept — typically also Pareto-based (Ficici and Pollack 2001; Bucci and Pollack 2002; Popovici et al. 2012; Ficici 2008).

Our insight is the observation that novelty search, at its core, is searching the space of *novel archives*, not the space of candidate solutions. The true goal of this search is to *cover* a solution space as much as possible (develop a set that spreads out in the space) while remaining as efficiently *packed* as possible (keep points in the set far apart). We will discuss concepts from k nearest neighbor (k-NN) theory on page 3 that define these ideas more formally, and we describe heuristic approximations for measuring these in empirical studies.

The Simple Novelty Search EA

In order to more carefully focus our study on the question of whether or not the archive update mechanism within novelty search has an objective, we begin by simplifying the algorithm to only the critical components needed: namely a genetic operator, an archive, and an update rule. Our algorithm does not maintain a population at all. Instead, a parent is selected uniformly at random from within the archive itself, copied then mutated to produce a child. The child's sparseness is measured against the archive using Equation 1. If that value is greater than some selected ρ_{min}, then the child is added to the archive. This is repeated until some termination criterion is met.

Algorithm 1: Simple Novelty Search Evolutionary Algorithm (SNSEA)

input : ρ_{min}, k, termination criterion
output: $archive$

initialize individual x
$archive$ = $\{x\}$
while *not reached termination condition* **do**
  draw $parent$ uniformly at random from $archive$
  $child$ = mutate(copy of $parent$)
  ρ = sparseness($child, archive, k$)
  if $\rho \geq \rho_{min}$ **then**
    $archive = archive \cup \{child\}$
  end
end

For this paper, we use $k = 3$ for the sparseness computation in all cases for simplicity. The termination criterion is simply a maximum number of generations (500 in most cases). We will demonstrate the algorithm in a discrete binary space, a bounded Euclidean space, and an unbounded Euclidean space.

For the discrete experiment, the SNSEA will use a binary representation where individual points exist in the space

$\{0,1\}^n$, where n is the length of the binary string. Mutation is performed by independently flipping each bit with probability $1/n$. Without loss of generality, we initialize the first point at 0^n. Distance is computed in this space using Hamming distance, thus we refer to it as a *Hamming Space*.

For the Euclidean space experiments, the SNSEA will use a real valued representation where the individual points exist in the space $\mathbb{R}^d$, where d is the dimension of the space. Mutation is performed by independently adding an offset to every value in the vector according to $N(0,\sigma)$. Without loss of generality we initialize the first point at 0^d. Distance is computed in this space using Euclidean distance, thus we refer to it as a *Euclidean Space*. In the experiment where the space is bounded, we restrict mutation so that it cannot produce child gene values outside of $[0,1]$ — we redraw until the gene is inside this region.

We'll talk more about the implications of omitting a population in the Discussion section. It suffices to say here that this represents a substantive change to the way novelty search typically works. Another difference is that all our distance calculations are in the genotype space; however, this is not an important departure since we are only evaluating the claim about whether an objective exists, not what it looks like for a given problem instance.

Packing and Covering

In order to develop bounds on how efficiently distance-based, lazy methods like k-nearest neighbor algorithms are able to develop hypotheses for a given space, the theory community for that field have developed several formal definitions (Clarkson 1999).

The first idea is the ϵ-cover of a set: How well does the set cover some space?

Definition 1 *An **ϵ-cover** of some space $Z = \langle U, \delta \rangle$, where $\delta : U \times U \mapsto \mathbb{R}$ is a distance measure over U, is a set $A \subset U$ such that $\forall x \in U, \exists a \in A$ with $\delta(x,a) < \epsilon$.*

The second idea is ϵ-packing of a set: How efficient is the distribution of points in the set?

Definition 2 *Given the space $Z = \langle U, \delta \rangle$, where $\delta : U \times U \mapsto \mathbb{R}$ is a distance measure over U, a set $A \subset U$ is an **ϵ-packing** iff $\delta(a,b) > 2\epsilon \; \forall a, b \in A$.*

Put more simply, a set that *covers* a space well is one that is spread out over that space such that no point in the space is too far from at least one point in the set, while a set that is *packed* well is one in which points inside the set aren't too close together. Note that for cover, smaller is better (points in U are closer to points in A); however, for packing, larger is better (points in A are further apart).

Definition 3 *An **ϵ-net** $A \subset U$ is a set that is an ϵ-cover of U and an $(\epsilon/2)$-packing.*

In k-NN theory, an ϵ-net is optimal: A set that is both efficiently packed and covers the space well.

This preliminary paper begins a careful introduction of methods for how to evaluate the performance and claims of novelty search methods. For example, it provides a few empirical counter-examples to the "objectiveless" claim. For packing this is straightforward: It is simply half the maximum pairwise distance in the archive. Estimating the cover measures is computationally infeasible cover large discrete spaces and impossible over Euclidean spaces. We approximate this as follows. First, we select points uniformly at random from inside the search space. When the space is bounded, these points are selected from inside that bounded region. We'll explain how we address the unbounded approximation in the section with those results. For the binary case, we also explicitly include the 1^n point in that sample. We then find the closest point in the archive to each sample point. The maximum of such distances is reported as our estimate for ϵ-cover.

The SNSEA Optimization Process

Though the SNSEA does not have a population in the sense in which traditional novelty search has, another perspective is that the archive *is* the population. Seen this way, the SNSEA is most comparable to a $(\mu+1)$-EA (De Jong 2006), where parent selection occurs randomly from the population (archive) and truncation survival selection is being used. Of course, it differs from this in that the fitness of an individual depends on the current population (archive), which grows.

So the algorithm *does* have selective pressure and progresses in a particular direction; it is not random search. Generally, the algorithm gradually grows the size of the archive, adding new points as they are discovered. The SNSEA will steadily increase packing and decrease cover.

Mutation & Minimum Sparseness

One important observation to make about the SNSEA is that it is clear that the minimum sparseness criterion and the magnitude of the mutation are deeply related. More formal proof is needed; however, here we appeal to intuition. Consider the first SNSEA step in a Hamming space of dimensionality n and a bit-flip mutation rate of $1/n$. If $\rho_{min} = O(\lg n)$ then we will expect to wait for n steps just to see a point far enough away to add to the archive. As the archive expands, the problem gets worse because we must select a random archive point near the surface of the archive *and* generate a sufficiently far away point. In the Euclidean space, this problem is magnified since the relative ratio of the volume of the interior to the convex hull of the archive increases quickly as the archive grows. The curse of dimensionality leads to further complications with highly dimensional spaces.

On the other hand, if ρ_{min} is *constant* relative to the dimensionality of the space, then the SNSEA will spread out very slowly in the space — indeed, exponentially slowly as n increases for Hamming space.

So it is very important to get the balance between the mutation rate and the sparseness criterion. For Hamming spaces, we suggest setting the mutation rate to $1/n$ and the sparseness criterion to $c \lg n$, where $c \in (0,1)$. We do not have a recommendation yet for Euclidean spaces.

SNSEA Converges in Hamming Space

The SNSEA in Hamming space will add new points to the archive until there are no points left in $\{0,1\}^n$ that can be

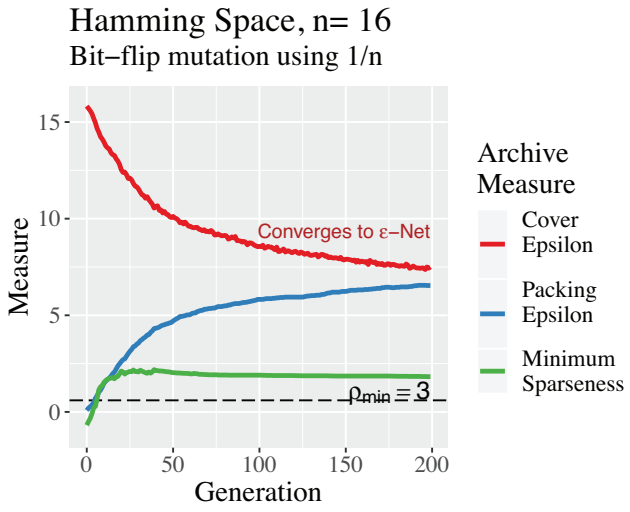

Figure 1: SNSEA applied to a Hamming space with $n = 16$ and $\rho_{min} = 3$ averaged over 50 independent trials.

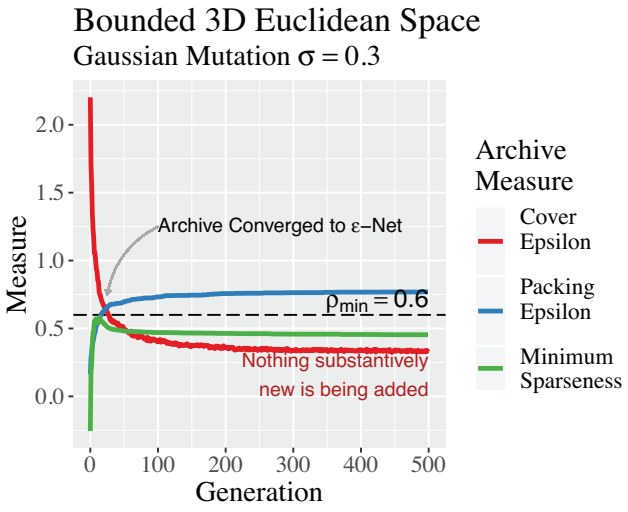

Figure 2: SNSEA applied to a bounded 3-dimensional Euclidean space averaged over 50 independent trials.

added that will meet the sparseness criterion. It's easy to see this for a small example, where $n = 3$ and $\rho_{min} = 2$: start with the 000, then adding the 111 point, now all other points are at most Hamming distance 2 from one of these two points, so no new points can be added. This is true for all Hamming spaces for $k > 1$: eventually there will be an archive that fully covers the space with $\epsilon \sim \rho_{min}$.

Empirically, it is easy to construct an example illustrating this effect. We consider $n = 16$, $\rho_{min} = \frac{3}{4} \lg 16 = 3$, and a max generation of 200. We ran 50 independent trials and reported the mean values for our packing and cover approximations for each generation. For comparison purposes, we also compute the average minimum sparseness for the archive in each generation — that is, we find the minimum sparseness of the archive at each step for each independent trial, and report the average over all trials. This gives us a sense for how the internal SNSEA mechanics will relate to the packing and cover measure. We can see from Figure 1 that, indeed, the cover of the Hamming space decreases steadily and the packing increases steadily. This experiment clearly shows SNSEA converging to an ϵ-net. The internal sparseness measure also levels off with the other two curves.

Though not shown in this paper, this behavior is consistent in any Hamming space. While from the perspective of individual points in the space, one might be led to believe the process is divergent (the archive grows steadily until the space is filled), really we can see that the search is occurring at the level of the *archive* space, not the individual point space. In that sense, the process is clearly convergent.

SNSEA Converges in Bounded Euclidean Space

The property of a Hamming space that leads to the notion that an archive can saturate the space is not inherent to that space. This property is true for *any* bounded space: Eventually the archive will cover the space completely and will be maximally packed. In the case of real-valued spaces, though, the process is slightly more complicated by the fact that the space is no longer discrete (i.e., there are an infinite number of arbitrarily precise points).

Again, it is easy to construct an example illustrating this effect. We consider $d = 3$, $\rho_{min} = 0.6$, $\sigma = 0.3$, and a max generation of 500. We ran 50 independent trials and again reported the mean values for our packing, cover, and min sparseness approximations for each generation. Figure 2 clearly shows an example where the SNSEA on a bounded 3D Euclidean space converges to an ϵ-net. Note that after the net is formed, the algorithm continues to occasionally add new points. These points are not substantively improving any measure, including and importantly the internal sparseness measure. A local optimum has been identified by the algorithm, it has converged into that local optimum, and now it is making small fine-tuning changes to move closer to true locally optimal archive configuration. The archive is not improving the overall cover of the space by very much.

This is useful to know! We can implement approximations of cover and packing in our algorithm, and when we see that an ϵ-net is formed either stop the search or reset the archive to enable the search to explore new archive configurations.

SNSEA Can Converge in Unbounded Spaces

It should be noted that the original authors of novelty search were clear that bounded spaces are not where novelty search is intended to be run (Lehman and Stanley 2008). Perhaps it is obvious that any bounded space will eventually be saturated by the archive? What the above examples do, though, is illustrate that if your understanding is that novelty search is *diverging* until it runs into the "walls" of the bounded space, then you may be looking at the process incorrectly: The processes are *converging* right from the beginning of the optimization, and the resulting archive is implicitly a local optimum (an ϵ-net).

But the SNSEA is driven to optimize the dual objectives

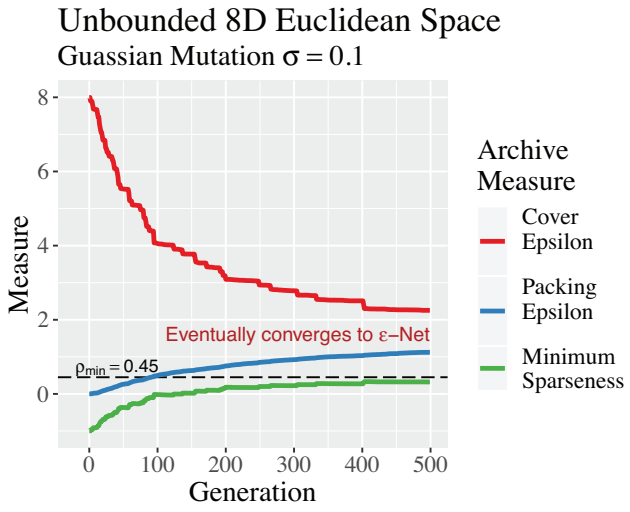

Figure 3: SNSEA applied to an unbounded 8-dimensional Euclidean space averaged over 50 independent trials.

of cover and packing whether spaces are bounded or unbounded. It is true that some parameterizations will lead to an apparent continual increase in the packing; however, this is precisely the same as a traditional real-valued EA maximizing a function like $f(x) = 2x - 3$; the EA will continue to produce new x values that give larger $f(x)$. Whether one calls this "convergence" or "divergence" depends on one's point of view; however, the SNSEA process *is* optimizing against the packing/cover objectives, regardless.

However, we can eschew the semantic debate in specific cases: There are parameterizations where the algorithm will converge in every sense of the word, even when the space is unbounded. When the mutation σ is small relative to the ρ_{min}, we can deliberately exacerbate the problem that occurs where large archive sizes make the algorithm increasingly unlikely to select points close enough to the surface of the archive that mutation has a reasonable probability to generate a point that will meet the sparseness criterion.

Figure 3 illustrates such an example, where $\sigma = 0.1$, $\rho_{min} = 0.45$, $d = 8$, and the algorithm is run to 500 generations. The space is completely unbounded. While packing can still be estimated as it was above, cover cannot be since the space is infinitely large. To address this, we assume that the search will (with high probability) remain within the bound $\pm\sigma \cdot maxGen$ in all dimensions, and we estimate cover as above but inside that region. We also confirmed that though any 8D point was *possible* in principle, in all runs none were generated outside that region. Again, 50 independent trials were performed. This process asymptotes toward an ϵ-net very, very slowly. We keep the max generation count low so that the visualization above conveys useful information. There is no doubt that this process is a convergent one, even though the space is unbounded.

Discussion

Novelty search is a powerful tool for finding complex objects in complicated, open-ended spaces. It relies on at least three key pieces: a sparseness metric and archive, generative representations, and distance measures in the actual solution space (e.g., behaviors) rather than genotype space. Unfortunately, there is very little foundational analysis of how novelty search works. This is important because there are several claims about novelty search that can and should be analyzed, perhaps first and foremost the idea that the search is "divergent" and "objectiveless".

In this paper, we consider only the first piece, using an extremely simplified version of novelty search so that we can focus entirely on how the sparseness metric and the archive update criterion affect search. We find that, once one sees that search in novelty search happens at the level of the archive space, not the individual point space, it is clear that the sparseness measure and archive update criterion create a process that is driven by a pair of objectives: spread out to *cover* the space, while trying to remain as efficiently *packed* as possible. Our SNSEA is driven to converge to an ϵ-net in the sense defined by k-NN theory — we claim that most novelty searches do something like this.

One obvious concern about our work is the lack of a true population for the SNSEA: we rely on only the archive itself. This concern is understandable given that the very issue we describe in the unbounded case (the process must find a point on the surface of the archive, then mutate sufficiently far away) appears related to this choice. In a traditional novelty search with a population that has a fixed size, the population may reflect the leading edge of the search (this has yet to be analyzed). We defend our choice in three ways.

First, this is the first step in beginning to try to understand the mechanics of novelty search in a more careful and formal way. It's best to do this with the simplest variants first. Second, the observation that the sparseness and archive update mechanisms within novelty search are creating a process that is driven to optimize the cover and packing of a space is independent of this choice: novelty search does this, whether or not there is a population. Third, it isn't clear that the parent selection, new point generation problem is entirely fixed by a population. If novelty search is indeed optimizing, it is possible that the population (as a set) may have similar problems. At the very least, the present research suggests that it is worthwhile studying that question.

This article provides three key contributions to novelty search. First, it lays out a vision for novelty search as an optimizer of *archive* space and introduces tools from k-NN theory to help understand this process. Second, it establishes guidance for configuring the mutation and the ρ_{min}, which are intimately tied in novelty search. Finally, it provides constructive advice for monitoring archive performance and dealing with scenarios where novelty search appears to have converged (e.g., reset the archive).

Our next step will be to empirically explore these questions about sparseness and archives with population-based novelty search methods, preliminary results for which also appear to show convergence properties. The overall goal is to develop tools and improved understandings for how nov-

elty search works, and how engineers can make productive design choices when employing this algorithm. Finally, we are also interested in exploring when novelty search can get stuck in local suboptima.

References

Banzhaf, W.; Nordin, P.; Keller, R. E.; and Francone, F. D. 1998. *Genetic Programming – An Introduction: On the Automatic Evolution of Computer Programs and Its Applications*. San Francisco, CA: Morgan Kaufmann.

Bucci, A., and Pollack, J. B. 2002. A mathematical framework for the study of coevolution. In *Foundations of Genetic Algorithms VI*, 221–236.

Clarkson, K. L. 1999. Nearest neighbor queries in metric spaces. *Discrete & Computational Geometry* 22(1):63–93.

Dawkins, R. 1996. *The Blind Watchmaker*. W. W. Norton & Company.

De Jong, K. A. 2006. *Evolutionary Computation: A Unified Approach*. MIT Press.

Dreher, T. 2014. *History of Computer Art*. chapter IV.3.

Elhadef, M., and Ayeb, B. 2000. Evolutionary algorithm for identifying faults in t-diagnosable systems. 74–83.

Ficici, S. G., and Pollack, J. B. 2001. Pareto optimality in coevolutionary learning. In *Advances in Artificial Life, 6th European Conference*, 316–325.

Ficici, S. G. 2008. Multiobjective optimization and coevolution. In *Multiobjective Problem Solving from Nature*. 31–52.

Goldberg, D. E. 1989. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison Wesley.

Goldberg, D. 2002. *The Design of Innovation: Lessons from and for Competent Genetic Algorithms*. Springer.

Hanes, J., and Wiegand, R. P. 2019. Analytical and evolutionary methods for finding cut volumes in fault trees constrained by location. *EEE Transactions on Reliability*.

Holland, J. 1975. *Adaptation in Natural and Artificial Systems*. Cambridge, MA: The MIT Press.

Huang, P.; Lehman, J.; Mok, A. K.; Miikkulainen, R.; and Sentis, L. 2014. Grasping novel objects with a dexterous robotic hand through neuroevolution. In *2014 IEEE Symposium on Computational Intelligence in Control and Automation*, 125–132.

Koza, J. 1992. *Genetic Programming*. MIT Press.

Langdon, W., and Poli, R. 2002. *Foundations of Genetic Programming*. Springer.

Lehman, J., and Miikkulainen, R. 2015. Enhancing divergent search through extinction events. In *Proc. of the Genetic and Evolutionary Computation Conference*, 951–958.

Lehman, J., and Stanley, K. O. 2008. Exploiting open-endedness to solve problems through the search for novelty. In *Proceedings of the Eleventh International Conference on the Synthesis and Simulation of Living Systems*, 329–336.

Lehman, J., and Stanley, K. O. 2011a. Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary Computation* 19(2):189–223.

Lehman, J., and Stanley, K. O. 2011b. Improving evolvability through novelty search and self-adaptation. In *Proceedings of the IEEE Congress on Evolutionary Computation*, 2693–2700.

Lehman, J.; Stanley, K. O.; and Miikkulainen, R. 2013. Effective diversity maintenance in deceptive domains. In *Proc. of the 15th Annual Conference on Genetic and Evolutionary Computation*, 215–222. ACM.

Lehman, J.; Wilder, B.; and Stanley, K. O. 2016. On the critical role of divergent selection in evolvability. *Frontiers in Robotics and AI*.

Michalawicz, Z. 1996. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag, 3rd edition.

Mitchell, T. 1997. *Machine Learning*. McGraw-Hill.

Morse, G.; Risi, S.; Snyder, C. R.; and Stanley, K. O. 2013. Single-unit pattern generators for quadruped locomotion. In *Proceedings of the 15th Genetic and and Evolutionary Computation Conference*, GECCO '13, 719–726.

Popovici, E.; Bucci, A.; Wiegand, R. P.; and de Jong, E. D. 2012. Coevolutionary principles. In *Handbook of Natural Computing*. 987–1033.

Romero, J., and Machado, P., eds. 2007. *The Art of Artificial Evolution: A Handbook on Evolutionary Art and Music*. Springer.

Seada, H., and Deb, K. 2018. Non-dominated sorting based multi/many-objective optimization: Two decades of research and application. In *Multi-Objective Optimization - Evolutionary to Hybrid Framework*. 1–24.

Secretan, J.; Beato, N.; D'Ambrosio, D. B.; Rodriguez, A.; Campbell, A.; and Stanley, K. O. 2008. Picbreeder: Evolving pictures collaboratively online. In *Proceedings of the Computer Human Interaction Conference*.

Stanley, K. O., and Lehman, J. 2015. *Why Greatness Cannot Be Planned - The Myth of the Objective*. Springer.

Stanley, K. O., and Miikkulainen, R. 2004. Competitive coevolution through evolutionary complexification. *Journal of Artificial Intelligence Research* 21:63–100.

Szerlip, P. A.; Morse, G.; Pugh, J. K.; and Stanley, K. O. 2015. Unsupervised feature learning through divergent discriminative feature accumulation. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2979–2985.

Zitzler, E.; Deb, K.; and Thiele, L. 2000. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation* 8(2):173–195.

Zitzler, E. 2012. Evolutionary multiobjective optimization. In *Handbook of Natural Computing*. 871–904.