No Free Lunch and Benchmarks

Edgar A. Duéñez-Guzmán duenez@oeb.harvard.edu
Department of Organismic and Evolutionary Biology, Harvard University, Cambridge, Massachusetts 02138, USA

Michael D. Vose vose@eecs.utk.edu
Department of Electrical Engineering and Computer Science, University of Tennessee, Knoxville, Tennessee 37969, USA

Abstract

We extend previous results concerning black box search algorithms, presenting new theoretical tools related to no free lunch (NFL) where functions are restricted to some benchmark (that need not be permutation closed), algorithms are restricted to some collection (that need not be permutation closed) or limited to some number of steps, or the performance measure is given. Minimax distinctions are considered from a geometric perspective, and basic results on performance matching are also presented.

Keywords

Benchmarks, black box search, minimax, no free lunch, optimization.

1 Introduction

Black box search algorithms are applied to search and optimization problems without exploiting a priori information about the objective function (such algorithms acquire information about the objective function by evaluating it). Evolutionary algorithms, and other nature-inspired search algorithms, are frequently implemented as black box algorithms, and have been regarded as robust optimization techniques (Holland, 1975; Goldberg, 1989; Mitchell, 1998; R. Haupt and S. Haupt, 2004). In this paper, we refer to nonrepeating/nonrevisiting black box search algorithms simply as algorithms.

The original no free lunch (NFL) theorems were expressed within a probabilistic framework and were used to investigate whether black box algorithms could differ in robustness (Wolpert and Macready, 1995). Sir Francis Bacon observed: "Men suppose that their reason has command over their words; still it happens that words in return exercise authority on reason" (Allibone, 1879). That opinion is apropos to NFL; it has been demonstrated that probability is inadequate to confirm unconstrained NFL results in the general case (Auger and Teytaud, 2007, 2010). In that sense, probability is an unfortunate historical artifact. With one exception, this paper abandons probability, preferring a set-theoretic framework which obviates measure-theoretic limitations by dispensing with probability altogether (Schumacher, 2000; Rowe et al., 2009).[1] That exception is the analysis of randomized algorithms.

[1] In the original context where probabilistic language is adequate (finite domains and codomains), it adds no essential value; probabilistic jargon may be expanded into set-theoretic definitions and simplified away.

The original NFL theorems imply that, if an algorithm's performance were plotted against all objective functions (over a fixed but arbitrary finite domain and codomain), the resulting performance graph would simply be a permutation of the performance graph of any other algorithm. In that sense, no algorithm—including classical genetic algorithms—can be more robust than any other. Results for infinite domains or codomains have a similar nature but are of a more technical character, see Rowe et al. (2009). NFL-like results can be found in diverse areas, including machine learning (Domingos, 1998; Wolpert, 2001), induction and combinatorial problems (Woodward and Neil, 2003), multi-objective optimization (Corne and Knowles, 2003), Boolean functions (Griffiths and Orponen, 2005), discrete Laplace operators (Wardetzky et al., 2007), and others.

Strong preconditions in the original NFL theorems constrained their application. Droste et al. (1999) observed that the likelihood of encountering optimization problems is nonuniform; if average performance is to be a meaningful statistic to summarize aggregate optimization behavior, then it should be a weighted average. Therefore, NFL—which assumed to be a uniform distribution[2]—was inappropriate for that purpose.

Later NFL variants were established which assumed that the set of functions, the distribution of functions, or the relationship between those functions and algorithms considered had special properties: permutation closure in the case of the Sharpened NFL (Schumacher, 2000), specialized distributions in the case of the nonuniform NFL[3] (Streeter, 2003; Igel and Toussaint, 2004), and focused sets in the case of focused NFL (Whitley and Rowe, 2008).

It has been observed that permutation closure is an assumption rarely satisfied (Igel and Toussaint, 2001; Streeter, 2003), which constrains application of sharpened NFL and nonuniform NFL. On the one hand, such observations do not invalidate NFL results, but caution against their misinterpretation or misapplication. On the other hand, Droste et al. (2002) provide counterpoint to the idea that robust search flourishes where NFL does not by establishing an almost NFL theorem (the success of search strategies on functions is paid for with bad behavior on many other functions of similar complexity). Whatever implications that NFL-like theorems have for search, the mathematical properties concerning algorithms which such theorems reveal need not be conflated with applications. The position that a mathematical formalization and investigation of NFL-like properties may be conducted unencumbered by applications has precedent (Rowe et al., 2009), and that is the outlook taken by this paper.

Previous introductory remarks roughly sketch the context for the contributions made by this paper. Given widespread use of benchmarks to assess performance, it is natural to consider NFL from a benchmark perspective. Nevertheless, little as of yet has been said concerning focused sets. Whitley and Rowe (2008) study focused sets—benchmarks compatible with NFL results for given algorithms limited to γ steps—but do not completely characterize them. In this paper, results concerning focused sets are simplified and extended, and the sharpened NFL emerges as a special case. Rowe and Vose (2011) obtain results concerning the use of arbitrary benchmarks to evaluate

[2]Wolpert and Macready (1995) initially assumed a uniform distribution for objective functions. Whereas they later discuss nonuniform distributions (Wolpert and Macready, 1997), the nonuniformities they consider cannot ameliorate the limitations pointed out by Droste et al. (1999).

[3]An interesting—though widely unrecognized—fact is that nonuniform NFL can be obtained from sharpened NFL by a suitable choice of performance measure (Rowe et al., 2009).

randomized algorithms. The underpinnings of their analysis—which their paper used but did not establish—are presented in this paper. We show a sharpened NFL is a few steps from their results, and then generalize to NFL results specific to particular performance measures. Performance matching and minimax distinctions (Wolpert and Macready, 1997) are also briefly considered.

2 Theoretical Background

Following Schumacher (2000), let $f : \mathcal{X} \to \mathcal{Y}$ be a function between finite sets (the set of all such functions is denoted by $\mathcal{Y}^{\mathcal{X}}$) and let y_i denote $f(x_i)$. The sets $\mathcal{X}$ and $\mathcal{Y}$, are arbitrary but fixed, while the function f may vary. A trace T corresponding to f is a sequence of elements from f (a sequence $S = \langle s_0 \ldots s_i \ldots \rangle$ is a function mapping i to s_i),

$$T = \langle (x_0, y_0), \ldots \rangle$$

where the x components are unique. To subscript (the name of) a trace T with an integer γ designates it to have γ elements,

$$T_\gamma = \langle (x_0, y_0), \ldots (x_{\gamma\text{-}1}, y_{\gamma\text{-}1}) \rangle$$

A performance vector is a sequence of values from $\mathcal{Y}$. We use the following notation:

$$\begin{aligned} T_x &= \langle x_0, \ldots \rangle \text{ sequence of } x \text{ components} \\ T_y &= \langle y_0, \ldots \rangle \text{ sequence of } y \text{ components} \end{aligned}$$

The performance vector associated with T is T_y. The range of any sequence S is denoted by S^*. In particular, $T^* \subset f$. Trace T is total if $T_x^* = \mathcal{X}$ (equivalently, $T^* = f$). A trace that is not total is said to be partial. Let $\mathcal{T}(f)$ be the set of all partial traces corresponding to f, and let the set of all partial traces be

$$\mathcal{T} = \bigcup_{f \in \mathcal{Y}^{\mathcal{X}}} \mathcal{T}(f)$$

A search operator is a function $g : \mathcal{T} \to \mathcal{X}$ such that $g(T) \notin T_x^*$. A nonrepeating/nonrevisiting deterministic black box search algorithm $\mathcal{A}$ corresponds to a search operator g, and will be referred to simply as an algorithm. Algorithm $\mathcal{A}$ applied to function f is denoted by $\mathcal{A}_f$, and maps traces to traces

$$\mathcal{A}_f(T) = \begin{cases} T \mid \langle (g(T), f \circ g(T)) \rangle & \text{if } T \in \mathcal{T}(f) \\ T & \text{otherwise} \end{cases}$$

where $\mid$ denotes concatenation. In procedural terms, algorithm $\mathcal{A}$ runs on function f by beginning with the empty trace $\emptyset$, and repeatedly applying $\mathcal{A}_f$; we denote by $\mathcal{A}(f)$ the total trace produced (by running $\mathcal{A}$ on f to convergence). Note that $\mathcal{A}(f)^* = f$. Algorithms $\mathcal{A}$ and $\mathcal{A}'$ are regarded as equal if and only if $\mathcal{A}(f) = \mathcal{A}'(f)$ for all f.

A trend in NFL theory is the progression from permutations of the search space (Radcliffe and Surry, 1995) to group actions on functions and algorithms. This can be seen at least as far back as Schumacher (2000). We will use the notation $\mathcal{X}!$ to refer to the set of permutations (bijections) on $\mathcal{X}$. Given a permutation $\sigma : \mathcal{X} \to \mathcal{X}$, the permutation of f by σ is the function σf defined by $(\sigma f)(x) = f(\sigma^{-1}(x))$. Thus permutations of $\mathcal{X}$ may also be considered as permutations $\sigma : \mathcal{Y}^{\mathcal{X}} \to \mathcal{Y}^{\mathcal{X}}$ via $f \mapsto \sigma f$. Such σ are therefore regarded as elements of $\mathcal{Y}^{\mathcal{X}}!$.

A key observation dating back at least as far as Radcliffe and Surry (1995) is the Uniqueness Theorem.

THEOREM 1 (UNIQUENESS): *If $f \neq f'$, then $\mathcal{A}(f)_y \neq \mathcal{A}(f')_y$, i.e., $\mathcal{A}(\cdot)_y$ is injective. Hence, the map $\mathcal{A}(\cdot)$ is invertible for every algorithm $\mathcal{A}$.*

Although the function $\mathcal{A}(\cdot)$ does not necessarily map onto the set of all traces, the Uniqueness Theorem together with the Completeness Theorem (below) imply $\mathcal{A}(\cdot)_y$ is a bijection from $\mathcal{Y}^{\mathcal{X}}$ to $\mathcal{Y}^{|\mathcal{X}|}$ (the Completeness Theorem is actually a corollary of Theorem 1, because the finite domain and codomain of $\mathcal{A}(\cdot)_y$ have the same size).

THEOREM 2 (COMPLETENESS): *The map $\mathcal{A}(\cdot)_y : \mathcal{Y}^{\mathcal{X}} \to \mathcal{Y}^{|\mathcal{X}|}$ is surjective.*

The map $\mathcal{A}(\cdot)_y$ is abbreviated by $\tilde{\mathcal{A}}$, its inverse is denoted by $\tilde{\mathcal{A}}^{-1}$, and juxtaposition denotes composition of such maps. Thus $\tilde{\mathcal{A}}^{-1}\tilde{\mathcal{B}} \in \mathcal{Y}^{\mathcal{X}}!$ (composing $\tilde{\mathcal{A}}^{-1}$ with $\tilde{\mathcal{B}}$ yields a bijection on $\mathcal{Y}^{\mathcal{X}}$) and in particular $\tilde{\mathcal{A}}^{-1}\tilde{\mathcal{B}}$ is a member of the group $(\mathcal{G}, \circ)$ of permutations of $\mathcal{Y}^{\mathcal{X}}$ under composition.

Given $\sigma \in \mathcal{X}!$, define the corresponding function σ_x which maps traces to traces by $\sigma_x(\langle (x_0, y_0), \ldots \rangle) = \langle (\sigma(x_0), y_0), \ldots \rangle$; hence σ_x operates on the x components of T by applying σ to each of them while leaving the y components unchanged. The permutation $\sigma\mathcal{A}$ of $\mathcal{A}$ by σ is the algorithm corresponding to search operator σg defined by $(\sigma g)(T) = \sigma^{-1}(g(\sigma_x(T)))$, where g is the search operator of $\mathcal{A}$. It follows that

$$\tau(\sigma\mathcal{A}) \;=\; (\sigma\tau)\mathcal{A}$$

for all $\sigma, \tau \in \mathcal{X}!$. The relationship between the permutation of an algorithm and the permutation of a function was given by Schumacher (2000) as the Duality Theorem.

THEOREM 3 (DUALITY): *For any algorithm $\mathcal{A}$, permutation $\sigma \in \mathcal{X}!$, and function $f \in \mathcal{Y}^{\mathcal{X}}$,*

$$\sigma_x(\mathcal{A}(\sigma f)) = \sigma\mathcal{A}(f)$$

Projecting traces in the displayed equality above to their y components yields the following Corollary:

COROLLARY 1: *For any algorithm $\mathcal{A}$ and permutation $\sigma \in \mathcal{X}!$,*

$$\tilde{\mathcal{A}} \circ \sigma \;=\; \widetilde{\sigma\mathcal{A}}$$

where on the left-hand side above, σ is regarded as an element of $\mathcal{Y}^{\mathcal{X}}!$.

One contribution made by sharpened NFL (Schumacher, 2000) was to establish the "only if" of the implication in the extension of NFL to permutation closed sets (Radcliffe and Surry, 1995). A set of functions $\mathcal{F} \subset \mathcal{Y}^{\mathcal{X}}$ is closed with respect to a set $S \subset \mathcal{X}!$ of permutations iff

$$\mathcal{F} \;=\; \{\sigma f \;:\; f \in \mathcal{F},\, \sigma \in S\}$$

The set $\mathcal{F}$ is permutation closed iff it is closed with respect to $\mathcal{X}!$.

THEOREM 4 (SHARPENED NFL): *Let $\mathcal{F} \subset \mathcal{Y}^{\mathcal{X}}$.*

$$\forall \mathcal{A}, \mathcal{B}\,.\,\{\tilde{\mathcal{A}}(f) : f \in \mathcal{F}\} = \{\tilde{\mathcal{B}}(f) : f \in \mathcal{F}\}$$

iff $\mathcal{F}$ is permutation closed.

Thus, the set $\{\tilde{\mathcal{A}}(f) : f \in \mathcal{F}\}$ of performance vectors is independent of the algorithm $\mathcal{A}$ which generated it (iff $\mathcal{F}$ is permutation closed) and any performance measure of that set of performance vectors is likewise independent of the algorithm.

3 Focused No Free Lunch

Focused NFL (Whitley and Rowe, 2008) concerns benchmarks. The basic question is: if attention is restricted to a subset of algorithms, then for what benchmarks (sets of functions) must the algorithms have equal performance? Benchmark $\mathcal{F}$ is focused with respect to a set $\mathfrak{A}$ of algorithms iff $\{\tilde{\mathcal{A}}(f) : f \in \mathcal{F}\}$ is independent of $\mathcal{A} \in \mathfrak{A}$. In other words, algorithms in $\mathfrak{A}$ have the same set of performance vectors over the benchmark. In particular, every performance measure—which therefore is a function of the same set of performance vectors—must necessarily yield the same result over the benchmark, for each algorithm in $\mathfrak{A}$. It is possible, however, that algorithms in $\mathfrak{A}$ may exhibit the same performance on different functions of the benchmark. Note that sharpened NFL can be rephrased as follows:

THEOREM 5: *Benchmark $\mathcal{F}$ is focused wrt the set of all algorithms iff it is permutation closed.*

DEFINITION 1: *For any set $\mathfrak{A}$ of algorithms, let $\mathcal{G}_{\mathfrak{A}}$ be the subgroup of $\mathcal{G}$ generated by*

$$\{\tilde{\mathcal{A}}^{-1}\tilde{\mathcal{B}} : \mathcal{A}, \mathcal{B} \in \mathfrak{A}\}$$

The main result in Whitley and Rowe (2008) is Lemma 1, which asserts:

THEOREM 6: *Let $\mathfrak{A}$ be a set of algorithms and let $f \in \mathcal{Y}^{\mathcal{X}}$. $\mathcal{G}_{\mathfrak{A}}\{f\}$ is the smallest benchmark containing f which is focused wrt $\mathfrak{A}$.*

Theorem 6 connects with the concept of closure under permutation; $\mathcal{G}_{\mathfrak{A}}$ is a group of permutations with respect to which $\mathcal{F}$ is closed. The implication in Whitley and Rowe's result (Theorem 6 above) can be strengthened to an iff as follows.

THEOREM 7 (FOCUSED NFL): *Benchmark $\mathcal{F}$ is focused wrt a set $\mathfrak{A}$ of algorithms iff it is closed wrt $\mathcal{G}_{\mathfrak{A}}$.*

PROOF: If $\mathcal{F}$ is focused wrt $\mathfrak{A}$, then for $\mathcal{A}, \mathcal{B} \in \mathfrak{A}$,

$$\begin{aligned}\{\tilde{\mathcal{A}}(f) : f \in \mathcal{F}\} &= \{\tilde{\mathcal{B}}(f) : f \in \mathcal{F}\} \\ &= \{\tilde{\mathcal{B}}(\tilde{\mathcal{B}}^{-1}\tilde{\mathcal{A}}(f)) : \tilde{\mathcal{B}}^{-1}\tilde{\mathcal{A}}(f) \in \mathcal{F}\} \\ &= \{\tilde{\mathcal{A}}(f) : f \in \tilde{\mathcal{A}}^{-1}\tilde{\mathcal{B}}\mathcal{F}\}\end{aligned}$$

By the Uniqueness Theorem, the membership predicates of the first and last displayed sets above must be identical. In particular, the generators of $\mathcal{G}_{\mathfrak{A}}$ permute $\mathcal{F}$; hence, it is closed wrt $\mathcal{G}_{\mathfrak{A}}$. Conversely, if $\mathcal{F}$ is closed wrt $\mathcal{G}_{\mathfrak{A}}$, then the first and last sets displayed above are identical, whereas the other equalities are trivial. □

It should be noted that the focused NFL demonstrated above (FNFL) can be obtained as a corollary of Whitley and Rowe's Lemma 1 (remarks following their Lemma 1 intimate how). Their result is strengthened in the sense that whereas they presented an implication, FNFL is stated as an iff. Moreover, a significant contribution of FNFL to NFL theory is clarity and simplicity in both statement and proof.

3.1 Algorithms Limited to γ Steps

To further generalize NFL, Whitley and Rowe (2008) consider algorithms restricted to γ steps. Their results are expressed in terms of a pseudocoded computer algorithm. We simplify, extend, and express those results from a set-theoretic point of view.

Let π_γ project sequences $x = \langle x_0, x_1, \ldots \rangle$ to their first γ elements,

$$\pi_\gamma(x) = \langle x_0, \ldots, x_{\gamma\text{-}1} \rangle$$

and to streamline notation, abbreviate $\pi_\gamma \circ \tilde{\mathcal{A}}$ by $\tilde{\mathcal{A}}_\gamma$. Thus, the performance vector $\mathcal{A}_f^\gamma(\emptyset)_y$ associated with the trace $\mathcal{A}_f^\gamma(\emptyset)$ generated by applying algorithm $\mathcal{A}$ to function f for γ steps is $\tilde{\mathcal{A}}_\gamma(f)$. Extend π_γ to multisets[4] by

$$\pi_\gamma S = \{\!\{\pi_\gamma(s) : s \in S\}\!\}$$

A benchmark $\mathcal{F}$ is focused wrt a set $\mathfrak{A}$ of algorithms and integer γ iff the multiset

$$\{\!\{\tilde{\mathcal{A}}_\gamma(f) : f \in \mathcal{F}\}\!\}$$

is independent of $\mathcal{A} \in \mathfrak{A}$. Of particular relevance is the set of permutations $G_\mathcal{A} \subset \mathcal{G}_\mathfrak{A}$ defined for $\mathcal{A} \in \mathfrak{A}$ by

$$G_\mathcal{A} = \{\tilde{\mathcal{A}}^{-1}\tilde{\mathcal{B}} : \mathcal{B} \in \mathfrak{A}\}$$

THEOREM 8 (γ-STEP FOCUSED NFL): *The benchmark $\mathcal{F}$ is focused wrt a collection $\mathfrak{A}$ of algorithms and integer γ iff for some $\mathcal{A} \in \mathfrak{A}$ and all $\alpha \in G_\mathcal{A}$,*

$$\{\!\{\tilde{\mathcal{A}}_\gamma(f) : f \in \mathcal{F}\}\!\} = \{\!\{\tilde{\mathcal{A}}_\gamma(f) : f \in \alpha\mathcal{F}\}\!\}$$

Moreover, if the above holds for some $\mathcal{A} \in \mathfrak{A}$, then it holds for all $\mathcal{A} \in \mathfrak{A}$.

PROOF: Suppose $\mathcal{F}$ is focused wrt $\mathfrak{A}$ and γ. Then for all $\mathcal{A}, \mathcal{B} \in \mathfrak{A}$, and $\alpha = \tilde{\mathcal{A}}^{-1}\tilde{\mathcal{B}} \in G_\mathcal{A}$,

$$\begin{aligned}
\{\!\{\tilde{\mathcal{A}}_\gamma(f) : f \in \mathcal{F}\}\!\} &= \pi_\gamma\{\!\{\tilde{\mathcal{A}}(f) : f \in \mathcal{F}\}\!\} \\
&= \pi_\gamma\{\!\{\tilde{\mathcal{B}}(f) : f \in \mathcal{F}\}\!\} \\
&= \pi_\gamma\{\!\{\tilde{\mathcal{B}}(\tilde{\mathcal{B}}^{-1}\tilde{\mathcal{A}}(f)) : \tilde{\mathcal{B}}^{-1}\tilde{\mathcal{A}}(f) \in \mathcal{F}\}\!\} \\
&= \{\!\{\tilde{\mathcal{A}}_\gamma(f) : f \in \tilde{\mathcal{A}}^{-1}\tilde{\mathcal{B}}\mathcal{F}\}\!\} \\
&= \{\!\{\tilde{\mathcal{A}}_\gamma(f) : f \in \alpha\mathcal{F}\}\!\}
\end{aligned}$$

Conversely, if $\{\!\{\tilde{\mathcal{A}}_\gamma(f) : f \in \mathcal{F}\}\!\} = \{\!\{\tilde{\mathcal{A}}_\gamma(f) : f \in \alpha\mathcal{F}\}\!\}$ for some $\mathcal{A} \in \mathfrak{A}$ and all $\alpha \in G_\mathcal{A}$, then the first and last sets displayed above are identical (for $\alpha = \tilde{\mathcal{A}}^{-1}\tilde{\mathcal{B}}$), whereas the other equalities are trivial. Hence, $\mathcal{F}$ is focused wrt $\mathfrak{A}$ and γ, and the first part of the proof (above) implies that $\{\!\{\tilde{\mathcal{A}}_\gamma(f) : f \in \mathcal{F}\}\!\} = \{\!\{\tilde{\mathcal{A}}_\gamma(f) : f \in \alpha\mathcal{F}\}\!\}$ for all $\mathcal{A} \in \mathfrak{A}$ and $\alpha \in G_\mathcal{A}$. □

Suppose one were interested in constructing a benchmark containing f which is focused wrt $\{\mathcal{A}, \mathcal{B}\}$ and γ. The following corollary provides one answer.

COROLLARY 2: *Let $f \in \mathcal{Y}^\mathcal{X}$ and $0 \leq \gamma \leq |\mathcal{X}|$. Given algorithms $\mathcal{A}, \mathcal{B}$, let $\alpha = \tilde{\mathcal{A}}^{-1}\tilde{\mathcal{B}}$. The benchmark*

$$\mathcal{F}_\gamma = \{\alpha^k f : k \geq 0 \wedge \tilde{\mathcal{A}}_\gamma(f) \notin \{\tilde{\mathcal{A}}_\gamma(\alpha^i f) : 0 < i \leq k\}\}$$

is focused wrt $\{\mathcal{A}, \mathcal{B}\}$ and γ.

[4] A multiset—delimited with doubled brackets {{ }}—extends the set concept by allowing elements to appear more than once.

PROOF: Note that $\mathcal{F}_\gamma = \{\alpha^0 f, ..., \alpha^{\ell\text{-}1} f\}$ where ℓ is the smallest positive integer for which $\tilde{\mathcal{A}}_\gamma(\alpha^\ell f) = \tilde{\mathcal{A}}_\gamma(f)$. In particular,

$$\begin{aligned}\{\{\tilde{\mathcal{A}}_\gamma(h) : h \in \alpha\mathcal{F}_\gamma\}\} &= \{\{\tilde{\mathcal{A}}_\gamma(\alpha^i f) : 0 < i \leq \ell\}\} \\ &= \{\{\tilde{\mathcal{A}}_\gamma(\alpha^i f) : 0 \leq i < \ell\}\} \\ &= \{\{\tilde{\mathcal{A}}_\gamma(h) : h \in \mathcal{F}_\gamma\}\}\end{aligned}$$

Therefore, the condition of γ-step focused NFL is satisfied. □

Corollary 2 generalizes the corresponding algorithmic construction presented in Whitley and Rowe (2008); whereas they restrict attention to path-search algorithms—those for which $\mathcal{A}(f)_x$ is independent of f—there is no such restriction in Corollary 2. Theorem γ-step focused NFL (γFNFL) has no counterpart in Whitley and Rowe (2008), but it reflects (in a sense clarified by the following Corollary and Theorem) a cyclic aspect of focused benchmarks which, if not formalized, was certainly alluded to in their paper.

Given algorithm $\mathcal{A}$ and integer $0 \leq \gamma \leq |\mathcal{X}|$, define the equivalence relation $\equiv$ on $\mathcal{Y}^{\mathcal{X}}$ by

$$f \equiv f' \iff \tilde{\mathcal{A}}_\gamma(f) = \tilde{\mathcal{A}}_\gamma(f')$$

DEFINITION 2: *A benchmark $\mathcal{F}$ is* cyclic *wrt algorithm $\mathcal{A}$, integer $0 \leq \gamma \leq |\mathcal{X}|$, and $\alpha \in \mathcal{Y}^{\mathcal{X}}$! iff for some positive integer ℓ,*

$$\mathcal{F} = \{f_0, ..., f_{\ell\text{-}1}\} \quad \text{where } \alpha f_{i \bmod \ell} \equiv f_{i+1 \bmod \ell}$$

Observe that the benchmarks $\mathcal{F}_\gamma$ of Corollary 2 are cyclic. In fact, every focused benchmark can be decomposed into a disjoint union of cyclic benchmarks.

LEMMA 1: *Let $\mathcal{F}$ be focused wrt $\mathfrak{A}$ and γ. For every $\mathcal{A} \in \mathfrak{A}$ and every $\alpha \in G_{\mathcal{A}}$,*

$$\mathcal{F} = \bigcup_{j>0} \mathcal{F}_j$$

where the union above is disjoint, and each $\mathcal{F}_j$ (which may depend on $\mathfrak{A}$, $\mathcal{A}$, γ, and α) is cyclic wrt $\mathcal{A}$, γ, and α.

PROOF: Let $U_0 = \emptyset$, and inductively construct

$$\begin{aligned}\mathcal{F}_i &\subset \mathcal{F} \setminus U_{i\text{-}1} \\ U_i &= \bigcup_{j \leq i} \mathcal{F}_j\end{aligned}$$

such that

$$\{\{\tilde{\mathcal{A}}_\gamma(f) : f \in \mathcal{F} \setminus U_i\}\} = \{\{\tilde{\mathcal{A}}_\gamma(f) : f \in \alpha(\mathcal{F} \setminus U_i)\}\}$$

as follows. Note that the equality above—the invariant—holds when $i = 0$ (by γFNFL), and the construction process must terminate provided the $\mathcal{F}_j$ are nonempty (since $\mathcal{F}$ is finite). Choose $f_0 \in \mathcal{F} \setminus U_{i\text{-}1}$. The invariant guarantees the existence of $f_1 \in \mathcal{F} \setminus U_{i\text{-}1}$ such that

$$f_0 \equiv \alpha f_1$$

if $f_1 = f_0$, then $\mathcal{F}_i = \{f_0\}$. Otherwise, continue to choose $f_k \in \mathcal{F} \setminus U_{i\text{-}1}$ (by appealing to the invariant) such that

$$f_{k\text{-}1} \equiv \alpha f_k$$

until $f_k \in \{f_0, \ldots, f_{k\text{-}1}\}$. At that point, there exists h such that

$$f_h \equiv \alpha f_{h+1}$$

$$f_{h+1} \equiv \alpha f_{h+2}$$

$$\vdots$$

$$f_k \equiv \alpha f_h$$

Let $\mathcal{F}_i$ be the collection of functions on the left-hand sides displayed above. Reindexing according to

$$f'_j = f_{k\text{-}j} \quad \text{for } 0 < j < \ell = k + 1 - h$$

shows $\mathcal{F}_i$ to be cyclic. Moreover, the invariant is preserved because removing $\mathcal{F}_i$ from the (previous) index set ($\mathcal{F} \setminus U_{i\text{-}1}$) has the effect of removing identical objects (the $\tilde{\mathcal{A}}_\gamma(f'_j)$) from both sides of the invariant. □

Combining previous results yields the following decomposition theorem.

THEOREM 9: *Benchmark $\mathcal{F}$ is focused wrt a set $\mathfrak{A}$ of algorithms and $0 \leq \gamma \leq |\mathcal{X}|$ iff for some $\mathcal{A} \in \mathfrak{A}$ and all $\alpha \in G_{\mathcal{A}}$,*

$$\mathcal{F} = \bigcup_{j>0} \mathcal{F}_j$$

where the union above is disjoint, and each $\mathcal{F}_j$ (which may depend on $\mathfrak{A}$, $\mathcal{A}$, γ, and α) is cyclic wrt $\mathcal{A}$, γ, and α.

PROOF: Lemma 1 establishes one direction (of the iff). The converse follows by appealing to γFNFL; since the $\mathcal{F}_j$ are cyclic, it follows that

$$\{\!\{\tilde{\mathcal{A}}_\gamma(f) : f \in \mathcal{F}_j\}\!\} = \{\!\{\tilde{\mathcal{A}}_\gamma(f) : f \in \alpha\mathcal{F}_j\}\!\}$$

and hence the equality above holds after dropping the subscript on $\mathcal{F}_j$ (because $\mathcal{F}$ is a disjoint union of the $\mathcal{F}_j$). □

This section concludes with a formal demonstration that sharpened NFL can be obtained as a special case of γFNFL. Observe that when $\gamma = |\mathcal{X}|$, the equality condition in γFNFL reduces to

$$\{\!\{\tilde{\mathcal{A}}(f) : f \in \mathcal{F}\}\!\} = \{\!\{\tilde{\mathcal{A}}(f) : f \in \alpha\mathcal{F}\}\!\}$$

which is equivalent—via the Uniqueness Theorem—to the invariance of $\mathcal{F}$ under the action of $G_{\mathcal{A}}$ (because the equality condition is quantified over all $\alpha \in G_{\mathcal{A}}$). It follows that γFNFL reduces to focused NFL when $\gamma = |\mathcal{X}|$, provided invariance under $G_{\mathcal{A}}$ for all $\mathcal{A} \in \mathfrak{A}$ is equivalent to invariance under $\mathcal{G}_{\mathfrak{A}}$. That is indeed the case, since the set of generators of $\mathcal{G}_{\mathfrak{A}}$ is the union over all $\mathcal{A} \in \mathfrak{A}$ of $G_{\mathcal{A}}$. Finally, when $\gamma = |\mathcal{X}|$ and $\mathfrak{A}$ is the set of all algorithms, focused NFL reduces to sharpened NFL—via Corollary 1—since invariance under $\mathfrak{A}$ implies invariance under the generator

$$\tilde{\mathcal{A}}^{-1}\widetilde{\sigma\mathcal{A}} = \tilde{\mathcal{A}}^{-1}\tilde{\mathcal{A}} \circ \sigma = \sigma$$

for all $\sigma \in \mathcal{X}$ (i.e., $\mathcal{F}$ is permutation closed). A technicality remains: is it possible that $\mathcal{F}$ is permutation closed, yet not closed under $\mathcal{G}_{\mathfrak{A}}$? Evidently not, since

$$f' = \tilde{\mathcal{A}}^{-1}\tilde{\mathcal{B}}(f) \implies \tilde{\mathcal{A}}(f') = \tilde{\mathcal{B}}(f) \implies f' = \sigma f$$

for some $\sigma \in \mathcal{X}$ (the last implication is Lemma 2 in Rowe et al., 2009).

4 Randomized Algorithms and Benchmarks

A randomized algorithm is identified with a probability vector μ indexed over the set $\mathfrak{D}$ of algorithms; component $\mu_{\mathcal{A}}$ is the probability the randomized algorithm (described by) μ behaves like $\mathcal{A}$; the total trace $\mu(f)$ resulting from applying μ to f is $\mathcal{A}(f)$ with probability $\mu_{\mathcal{A}}$. In procedural terms, randomized algorithm μ runs on f by choosing $\mathcal{A}$ with probability $\mu_{\mathcal{A}}$ and then applying algorithm $\mathcal{A}$ to f. Note that the collection of randomized algorithms contains the set of (deterministic) algorithms. Randomized algorithms are identified with elements of the simplex $\Lambda_{\mathfrak{D}}$ defined by

$$\Lambda_{\mathfrak{D}} \;=\; \{x \in \mathbb{R}^{|\mathfrak{D}|} \;:\; i \in \mathfrak{D},\; x_i \geq 0,\; \sum_i x_i = 1\}$$

Given $\sigma \in \mathcal{X}!$ and $\mu \in \Lambda_{|\mathfrak{D}|}$, the randomized algorithm $\sigma\mu$ is defined by

$$(\sigma\mu)_{\mathcal{A}} \;=\; \mu_{\sigma^{-1}\mathcal{A}}$$

Procedurally, running $\sigma\mu$ on f amounts to choosing $\mathcal{A}$ with probability $\mu_{\mathcal{A}}$ and then running $\sigma\mathcal{A}$ on f. Random algorithms μ and μ' are equivalent, denoted by $\mu \equiv \mu'$, iff for all $f \in \mathcal{Y}^{|\mathcal{X}|}$ and every trace T,

$$\text{Prob}\{\mu(f) = T\} = \text{Prob}\{\mu'(f) = T\}$$

PROPOSITION 1: *For all permutations τ, σ, and randomized algorithms μ,*

$$\tau(\sigma\mu) = (\sigma\tau)\mu$$

Moreover, $\iota\,\mu = \mu$ where ι is the identity permutation.

PROOF:

$$(\tau(\sigma\mu))_{\mathcal{A}} \;=\; (\sigma\mu)_{\tau^{-1}\mathcal{A}} \;=\; \mu_{\sigma^{-1}(\tau^{-1}\mathcal{A})} \;=\; \mu_{(\sigma\tau)^{-1}\mathcal{A}} \;=\; ((\sigma\tau)\mu)_{\mathcal{A}}$$

Note that $\iota\, g = g$ for every search operator g. Hence $\iota\,\mathcal{A} = \mathcal{A}$ and therefore $\iota\,\mu = \mu$. □

A performance measure is a function $m : \mathcal{Y}^{|\mathcal{X}|} \to \mathbb{R}$. So as to streamline exposition, a performance measure will simply be called a measure. Measure m is extended to randomized algorithms in the natural way; the performance of μ on f as measured by m is

$$m(\mu, f) \;=\; \sum_{\mathcal{A}} m(\tilde{\mathcal{A}}(f))\,\mu_{\mathcal{A}}$$

Note that m is polymorphic: measure m maps performance vectors to values, whereas the performance of μ on f as measured by m is the corresponding expected value.

PROPOSITION 2: *Given randomized algorithm μ, function f, measure m, and permutation σ,*

$$m(\sigma\mu, f) = m(\mu, \sigma f)$$

PROOF: Expanding definitions, appealing to Proposition 1 to justify reindexing below (the special case $\mu_{\mathcal{A}} = 1$ implies $\mathcal{A} \mapsto \sigma\mathcal{A}$ is invertible), and using Corollary 1,

$$\begin{aligned} m(\sigma\mu, f) &= \sum_{\mathcal{A}} m(\tilde{\mathcal{A}}(f))\,\mu_{\sigma^{-1}\mathcal{A}} \\ &= \sum_{\sigma\mathcal{A}} m(\widetilde{\sigma\mathcal{A}}(f))\,\mu_{\mathcal{A}} \end{aligned}$$

$$= \sum_{\mathcal{A}} m(\widetilde{\mathcal{A}} \circ \sigma(f))\, \mu_{\mathcal{A}}$$
$$= m(\mu, \sigma f)$$

□

The expected performance of randomized algorithm μ over benchmark $\mathcal{F}$ is

$$\mathcal{E}(\mu, \mathcal{F}) = \frac{1}{|\mathcal{F}|} \sum_{f \in \mathcal{F}} m(\mu, f)$$

and is referred to as expected average performance (one might argue it is more natural to call it average expected performance, but the average and expectation commute). Explicit dependence on the measure m is indicated by subscript,

$$\mathcal{E}_m(\mu, \mathcal{F})$$

PROPOSITION 3 (LINEARITY): *$\mathcal{E}(\cdot, \cdot)$ is linear in its first argument.*

PROOF:

$$\mathcal{E}\left(\sum_i \lambda_i \mu_i, \mathcal{F}\right) = \frac{1}{|\mathcal{F}|} \sum_{f \in \mathcal{F}} \sum_{\mathcal{A}} m(\tilde{\mathcal{A}}(f)) \sum_i \lambda_i (\mu_i)_{\mathcal{A}}$$
$$= \sum_i \lambda_i \frac{1}{|\mathcal{F}|} \sum_{f \in \mathcal{F}} \sum_{\mathcal{A}} m(\tilde{\mathcal{A}}(f))(\mu_i)_{\mathcal{A}}$$
$$= \sum_i \lambda_i \mathcal{E}(\mu_i, \mathcal{F})$$

□

THEOREM 10 (EXPECTED DUALITY): *For every measure m, randomized algorithm μ, benchmark $\mathcal{F}$, and permutation $\sigma \in \mathcal{X}!$,*

$$\mathcal{E}_m(\sigma\mu, \mathcal{F}) = \mathcal{E}_m(\mu, \sigma\mathcal{F})$$

PROOF: Expanding definitions and appealing to Proposition 2,

$$\mathcal{E}(\sigma\mu, \mathcal{F}) = \frac{1}{|\mathcal{F}|} \sum_{f \in \mathcal{F}} m(\sigma\mu, f) = \frac{1}{|\mathcal{F}|} \sum_{f \in \mathcal{F}} m(\mu, \sigma f) = \mathcal{E}(\mu, \sigma\mathcal{F})$$

□

The results in Section 4 up to this point are theoretical foundations of the analysis presented in Rowe and Vose (2011) which their paper used but did not establish.

4.1 Benchmark Symmetries

Given benchmark $\mathcal{F}$, the group $G(\mathcal{F})$ of benchmark symmetries[5] is

$$G(\mathcal{F}) = \{\sigma \in \mathcal{X}! : \sigma\mathcal{F} = \mathcal{F}\}$$

[5]Previous use of G is superseded by the notation defined here.

(the usual conventions are employed: $tS = \{\{ts : s \in S\}\}$ and $St = \{\{st : s \in S\}\}$). An immediate consequence of the Expected Duality Theorem (above) is Theorem 3 of Rowe and Vose (2011):

For every measure m, randomized algorithm μ, and benchmark $\mathcal{F}$,

$$\mathcal{E}_m(\sigma\mu, \mathcal{F}) \;=\; \mathcal{E}_m(\mu, \mathcal{F})$$

for all $\sigma \in G(\mathcal{F})$.

The above is a FNFL-type result: *Every randomized algorithm in $G(\mathcal{F})\mu$ has the same expected average performance over benchmark $\mathcal{F}$.*

Let $\mathcal{F}(\mu)$ be the randomized algorithm

$$\mathcal{F}(\mu) \;=\; |G(\mathcal{F})|^{-1} \sum_{\sigma \in G(\mathcal{F})} \sigma\mu$$

It follows via Linearity (Proposition 3) that

$$\mathcal{E}(\mathcal{F}(\mu), \mathcal{F}) = \mathcal{E}(\mu, \mathcal{F})$$

Moreover if $\mathcal{F}$ is permutation closed, then

$$\mathcal{F}(\mu) \equiv \mathcal{F}(\mu')$$

for all randomized algorithms μ, μ' (Rowe and Vose, 2011), in which case

$$\mathcal{E}(\mu, \mathcal{F}) \;=\; \mathcal{E}(\mathcal{F}(\mu), \mathcal{F}) \;=\; \mathcal{E}(\mathcal{F}(\mu'), \mathcal{F}) \;=\; \mathcal{E}(\mu', \mathcal{F})$$

that is, every randomized algorithm has the same expected average performance. While not the focus of Rowe and Vose (2011), they present the demonstration above. Before generalizing to NFL-type results specific to particular measures (in the next section), we make observations which sharpen this NFL result.

The support $\mathcal{S}_\mu$ of random algorithm μ is the set of algorithms $\mathcal{A}$ for which $\mu_\mathcal{A} > 0$. A deterministic algorithm is a randomized algorithm whose support is a singleton set; such support is called trivial. A nontrivial randomized algorithm has nontrivial support. A deterministic algorithm μ has a unique nonzero component $\mu_\mathcal{A} = 1$; to streamline notation, $\mathcal{A}$ may be used to denote such μ. Measure m is constant wrt benchmark $\mathcal{F}$ and support set $\mathcal{S}$ iff $\mathcal{E}_m(\mathcal{A}, \mathcal{F}) = \mathcal{E}_m(\mathcal{A}', \mathcal{F})$ for all $\mathcal{A},\ \mathcal{A}' \in \mathcal{S}$.

PROPOSITION 4: *For every measure m, randomized algorithm μ, and benchmark $\mathcal{F}$, if m is nonconstant wrt $\mathcal{F}$ and $\mathcal{S}_\mu$, then there exist infinitely many μ' and μ'' with support $\mathcal{S}_\mu$ such that*

$$\mathcal{E}_m(\mu', \mathcal{F}) < \mathcal{E}_m(\mu, \mathcal{F}) < \mathcal{E}_m(\mu'', \mathcal{F})$$

If m is constant wrt $\mathcal{F}$ and $\mathcal{S}_\mu$, then $\mathcal{S}_\eta \subset \mathcal{S}_\mu \Longrightarrow \mathcal{E}_m(\eta, \mathcal{F}) = \mathcal{E}_m(\mu, \mathcal{F})$.

PROOF: Consider the first case, where $k = \mathcal{E}_m(\mathcal{A}, \mathcal{F})$ for all $\mathcal{A} \in \mathcal{S}_\mu$. If $\mathcal{S}_\eta \subset \mathcal{S}_\mu$, then

$$\mathcal{E}_m(\eta, \mathcal{F}) \;=\; \sum_{\mathcal{A} \in \mathcal{S}_\eta} \eta_\mathcal{A}\, \mathcal{E}_m(\mathcal{A}, \mathcal{F}) \;=\; k \sum_{\mathcal{A} \in \mathcal{S}_\eta} \eta_\mathcal{A} \;=\; k$$

In the second case, $\rho(\mu) = \mathcal{E}_m(\mu, \mathcal{F})$ is linear (by Proposition 3) and nonconstant on $\Lambda_{\mathcal{S}_\mu}$ (because m is not constant wrt $\mathcal{F}$ and $\mathcal{S}_\mu$). Since μ is an interior point of $\Lambda_{\mathcal{S}_\mu}$ with respect to the subspace topology induced from $\mathbb{R}^{|\mathcal{S}_\mu|}$ (because $\mathcal{S}_\mu$ is the support of μ),

there exists some open set B such that $\mu \in B \subset \Lambda_{\mathcal{S}_\mu}$. Since linear ρ is not constant on B (it otherwise would be constant on $\Lambda_{\mathcal{S}_\mu}$), it attains values larger and smaller than $\rho(\mu)$ on B (Roberts and Varberg, 1973). □

If benchmark $\mathcal{F}$ is not permutation closed, then Proposition 4 applies as follows. For some $f \in \mathcal{F}$, there exists $\sigma \in \mathcal{X}!$ such that $\sigma^{-1} f \notin \mathcal{F}$. Let μ have support containing $\{\iota\mathcal{A}, \sigma\mathcal{A}\}$ for some algorithm $\mathcal{A}$. Define the measure $m(T) = [\tilde{\mathcal{A}}^{-1}(T) = f]\,|\mathcal{F}|$ where [*expression*] is 1 if *expression* is true, and 0 otherwise.[6] Note that m is nonconstant wrt $\mathcal{F}$ and $\mathcal{S}_\mu$, since the following depends on choice of $\tau \in \{\iota, \sigma\}$,

$$\mathcal{E}_m(\tau\mathcal{A}, \mathcal{F}) = \sum_{h \in \mathcal{F}} [\tilde{\mathcal{A}}^{-1} \widetilde{\tau\mathcal{A}}(h) = f] = \sum_{h \in \mathcal{F}} [\tau h = f] = [\tau^{-1} f \in \mathcal{F}]$$

4.2 Performance Measures

Conventional use of the term benchmark includes the means by which performance is measured; a performance measure is itself a benchmark. Benchmark symmetries are generalized to depend upon the measure m as well as the set of test functions $\mathcal{F}$, and an FNLF-type result emerges which likewise depends upon m. Given benchmark $\mathcal{F}$, the collection $G_m(\mathcal{F})$ of benchmark invariants[7] is

$$G_m(\mathcal{F}) = \left\{\sigma \in \mathcal{X}! : \forall \mathcal{A}\,.\, \mathcal{E}_m(\sigma\mathcal{A}, \mathcal{F}) = \mathcal{E}_m(\mathcal{A}, \mathcal{F})\right\}$$

THEOREM 11: *$G_m(\mathcal{F})$ is a group (under composition) for every measure m and benchmark $\mathcal{F}$. For every randomized algorithm μ, the randomized algorithms in $G_m(\mathcal{F})\mu$ all have the same expected average performance over $\mathcal{F}$.*

PROOF: Since $\mathcal{X}!$ is finite, $G_m(\mathcal{F})$ is a group if it is closed under composition. Suppose $\sigma,\ \tau \in G_m(\mathcal{F})$. Appealing to Proposition 1,

$$\mathcal{E}_m((\tau\sigma)\mathcal{A}, \mathcal{F}) = \mathcal{E}_m(\sigma(\tau\mathcal{A}), \mathcal{F}) = \mathcal{E}_m(\tau\mathcal{A}, \mathcal{F}) = \mathcal{E}_m(\mathcal{A}, \mathcal{F})$$

Appealing to Expected Duality, Linearity, and the fact that $\mathcal{E}_m(\sigma\mathcal{A}, \mathcal{F}) = \mathcal{E}_m(\mathcal{A}, \mathcal{F})$,

$$\mathcal{E}_m(\sigma\mu, \mathcal{F}) = \sum_{\mathcal{A}} \mathcal{E}_m(\mathcal{A}, \sigma\mathcal{F})\,\mu_{\mathcal{A}} = \sum_{\mathcal{A}} \mathcal{E}_m(\sigma\mathcal{A}, \mathcal{F})\,\mu_{\mathcal{A}} = \mathcal{E}_m(\mu, \mathcal{F})$$

□

Theorem 11 is a γFNLF-type result as follows. Let m_γ be $m \circ \pi_\gamma$ for some $0 \leq \gamma \leq |\mathcal{X}|$ and some measure m. For every randomized algorithm μ, the γ-step expected average performance—as measured by m—of the collection of algorithms $G_{m_\gamma}(\mathcal{F})\mu$ is identical over benchmark $\mathcal{F}$. It should be appreciated that the benchmark invariants are completely independent of μ, no assumptions whatsoever have been made concerning the support $\mathcal{S}_\mu$ of μ, and the choice of μ is arbitrary.

THEOREM 12: *For every measure m and benchmark $\mathcal{F}$,*

$$\sigma \in G_m(\mathcal{F}) \implies G_m(\sigma\mathcal{F}) = G_m(\mathcal{F})$$

[6] Schumacher (2000) introduced this measure, but with different notation.

[7] Previous use of G is superseded by the notation defined here.

PROOF: Let $\sigma \in G_m(\mathcal{F})$, and $\tau \in G_m(\sigma\mathcal{F})$. Appealing to Expected Duality,

$$\mathcal{E}_m(\sigma(\tau\mathcal{A}), \mathcal{F}) \;=\; \mathcal{E}_m(\tau\mathcal{A}, \sigma\mathcal{F}) \;=\; \mathcal{E}_m(\mathcal{A}, \sigma\mathcal{F}) \;=\; \mathcal{E}_m(\sigma\mathcal{A}, \mathcal{F}) \;=\; \mathcal{E}_m(\mathcal{A}, \mathcal{F})$$

It follows (via Proposition 1) that $\tau\sigma \in G_m(\mathcal{F})$. Hence, $\tau \in G_m(\mathcal{F})\sigma^{-1} = G_m(\mathcal{F})$, since benchmark invariants form a group (Theorem 11). Thus, $G_m(\sigma\mathcal{F}) \subset G_m(\mathcal{F})$. Moreover,

$$\mathcal{E}_m(\sigma\mathcal{A}, \sigma\mathcal{F}) \;=\; \mathcal{E}_m(\sigma(\sigma\mathcal{A}), \mathcal{F}) \;=\; \mathcal{E}_m(\sigma\mathcal{A}, \mathcal{F}) \;=\; \mathcal{E}_m(\mathcal{A}, \sigma\mathcal{F})$$

(by Expected Duality), and therefore $\sigma \in G_m(\sigma\mathcal{F})$. Thus, $G_m(\mathcal{F}) \subset G_m(\sigma\mathcal{F})$. □

It is interesting to contrast Theorem 12 with the closing remarks of Section 3.1. Even with $\gamma = |\mathcal{X}|$, it does not follow that $\mathcal{F}$ is invariant under the action of $G_{m_\gamma}(\mathcal{F})$; it is not generally true that $\sigma\mathcal{F} = \mathcal{F}$ when $\sigma \in G_{m_\gamma}(\mathcal{F})$. But the message of Theorem 12 is precisely that invariance is regained, at the price of trading equality, $=$, for similarity, $\sim$, defined by having identical invariants; $\sigma\mathcal{F} \sim \mathcal{F}$.

We conclude with a generalization of the theoretical machinery used by Rowe and Vose (2011). Define $\mathcal{F}_m(\mu)$ to be[8]

$$\mathcal{F}_m(\mu) \;=\; |G_m(\mathcal{F})|^{-1} \sum_{\sigma \in G_m(\mathcal{F})} \sigma\mu$$

The following is a direct consequence of Theorem 11 and Linearity.

COROLLARY 3: *For every measure m, randomized algorithm μ, and benchmark $\mathcal{F}$,*

$$\mathcal{E}_m(\mathcal{F}_m(\mu), \mathcal{F}) \;=\; \mathcal{E}_m(\mu, \mathcal{F})$$

PROPOSITION 5: *For every measure m, randomized algorithm μ, and benchmark $\mathcal{F}$,*

$$\sigma \in G_m(\mathcal{F}) \;\Longrightarrow\; \sigma(\mathcal{F}_m(\mu)) \;=\; (\sigma\mathcal{F})_m(\mu) \;=\; \mathcal{F}_m(\sigma\mu) \;=\; \mathcal{F}_m(\mu)$$

PROOF: To streamline notation, abbreviate $G_m(\mathcal{F})$ by G. Appealing to Proposition 1, and using the fact that G is a group (Theorem 11),

$$\begin{aligned}
(\sigma(\mathcal{F}_m(\tau\mu')))_{\mathcal{A}} &= \mathcal{F}_m(\tau\mu')_{\sigma^{-1}\mathcal{A}} \\
&= |G|^{-1} \sum_{\sigma' \in G} (\sigma'(\tau\mu'))_{\sigma^{-1}\mathcal{A}} \\
&= |G|^{-1} \sum_{\sigma' \in G} ((\tau\sigma'\sigma)\mu')_{\mathcal{A}} \\
&= |G|^{-1} \sum_{\lambda \in G} (\lambda\mu')_{\mathcal{A}} \\
&= \mathcal{F}_m(\mu')_{\mathcal{A}}
\end{aligned}$$

Thus, $\mathcal{F}_m(\mu) = \sigma(\mathcal{F}_m(\mu)) = \mathcal{F}_m(\sigma\mu)$; choosing $\tau = \iota, \mu' = \mu$ yields the first equality and choosing $\tau = \sigma^{-1}, \mu' = \sigma\mu$ yields the second. Finally (appealing to Theorem 12),

$$(\sigma\mathcal{F})_m(\mu) \;=\; |G_m(\sigma\mathcal{F})|^{-1} \sum_{\lambda \in G_m(\sigma\mathcal{F})} \lambda\mu \;=\; |G|^{-1} \sum_{\lambda \in G} \lambda\mu \;=\; \mathcal{F}_m(\mu)$$

□

[8]Previous use of $\mathcal{F}(\mu)$ is superseded by the notation defined here.

PROPOSITION 6: *For every measure m, randomized algorithm μ, and benchmark $\mathcal{F}$,*

$$\mathcal{F}_m(\mathcal{F}_m(\mu)) \;=\; \mathcal{F}_m(\mu)$$

PROOF: Appealing to Proposition 5,

$$\begin{aligned}\mathcal{F}_m(\mathcal{F}_m(\mu)) &= |G_m(\mathcal{F})|^{-1} \sum_{\sigma \in G_m(\mathcal{F})} \sigma \mathcal{F}_m(\mu) \\ &= |G_m(\mathcal{F})|^{-1} \sum_{\sigma \in G_m(\mathcal{F})} \mathcal{F}_m(\mu) \\ &= \mathcal{F}_m(\mu)\end{aligned}$$

□

4.3 Matching Performance

Consider the scenario where randomized algorithm μ outperforms μ' on benchmark $\mathcal{F}$. Is there another benchmark $\mathcal{F}'$ for which μ' outperforms μ by the same amount? The answer depends on the measure m; Wolpert and Macready (1997) point out negative examples which they term minimax distinctions. This section analyzes the following predicate, which asserts that the particular measure m admits no minimax distinctions.

$$\forall \mu, \mu' \,.\, \forall \mathcal{F} \,.\, \exists \mathcal{F}' \,.\, \delta(m, \mu, \mu', \mathcal{F}) = \delta(m, \mu', \mu, \mathcal{F}') \tag{1}$$

where $\delta(m, \mu, \mu', \mathcal{F}) = \mathcal{E}_m(\mu, \mathcal{F}) - \mathcal{E}_m(\mu', \mathcal{F})$. It turns out that this predicate makes an interesting geometric claim. Let $\mathbf{1}$ denote the vector $\langle 1, \ldots, 1 \rangle$ of ones, and let $\parallel$ be the binary infix predicate asserting that its vector arguments are parallel.

THEOREM 13: *Let m be a measure, and for $f \in \mathcal{Y}^{\mathcal{X}}$, let $\vec{f}$ be the $|\mathfrak{D}|$-dimensional vector having components $\vec{f}_{\mathcal{A}} = m(\tilde{\mathcal{A}}(f))$. Measure m admits no minimax distinctions iff*

$$\forall \mathcal{F} \,.\, \exists \mathcal{F}' \,.\, \mathbf{1} \parallel \sum_{f \in \mathcal{F}} \frac{\vec{f}}{|\mathcal{F}|} + \sum_{f' \in \mathcal{F}'} \frac{\vec{f}'}{|\mathcal{F}'|} \tag{2}$$

PROOF: Note that $\delta(m, \mu, \mu', \mathcal{F}) = \delta(m, \mu', \mu, \mathcal{F}')$ is equivalent to

$$\mathcal{E}(\mu, \mathcal{F}) + \mathcal{E}(\mu, \mathcal{F}') = \mathcal{E}(\mu', \mathcal{F}) + \mathcal{E}(\mu', \mathcal{F}') \tag{3}$$

whose left-hand side is

$$\begin{aligned}&\frac{1}{|\mathcal{F}|} \sum_{f \in \mathcal{F}} \sum_{\mathcal{A}} \mu_{\mathcal{A}}\, m(\tilde{\mathcal{A}}(f)) + \frac{1}{|\mathcal{F}'|} \sum_{f \in \mathcal{F}'} \sum_{\mathcal{A}} \mu_{\mathcal{A}}\, m(\tilde{\mathcal{A}}(f)) \\ &= \sum_{\mathcal{A}} \mu_{\mathcal{A}} \left(\sum_{f \in \mathcal{F}} \frac{\vec{f}_{\mathcal{A}}}{|\mathcal{F}|} + \sum_{f \in \mathcal{F}'} \frac{\vec{f}'_{\mathcal{A}}}{|\mathcal{F}'|} \right)\end{aligned}$$

Replacing μ with μ' yields an expression for the right-hand side of Equation (3), which when combined with the above shows $\delta(m, \mu, \mu', \mathcal{F}) = \delta(m, \mu', \mu, \mathcal{F}')$ is equivalent to

$$\sum_{\mathcal{A}} (\mu_{\mathcal{A}} - \mu'_{\mathcal{A}}) \left(\sum_{f \in \mathcal{F}} \frac{\vec{f}_{\mathcal{A}}}{|\mathcal{F}|} + \sum_{f' \in \mathcal{F}'} \frac{\vec{f}'_{\mathcal{A}}}{|\mathcal{F}'|} \right) = 0$$

Stated more succinctly in vector notation, the above is

$$(\mu - \mu')^T \left(\sum_{f \in \mathcal{F}} \frac{\vec{f}}{|\mathcal{F}|} + \sum_{f' \in \mathcal{F}'} \frac{\vec{f}'}{|\mathcal{F}'|} \right) = 0$$

where superscript T denotes transpose. Let $v = \lambda(\mu - \mu')$ where $\lambda \in \mathbb{R}$. Observe that

$$\mathbf{1}^T v \;=\; \lambda \left(\sum_{\mathcal{A}} \mu_{\mathcal{A}} - \sum_{\mathcal{A}} \mu'_{\mathcal{A}} \right) \;=\; \lambda(1-1) \;=\; 0$$

hence $v \in \mathbf{1}^\perp$. Conversely, given $w = \langle w_1, \ldots, w_{|\mathfrak{D}|} \rangle \in \mathbf{1}^\perp$, define λ, μ, μ' by

$$\begin{aligned} \mu &= \mathbf{1}/|\mathfrak{D}| \\ \lambda &= |\mathfrak{D}| \max_i |w_i| \\ \mu' &= w/\lambda + \mu \end{aligned}$$

and note that

$$\mu'_i = \frac{1}{|\mathfrak{D}|} \left(1 + \frac{w_i}{\max_i |w_i|} \right)$$

which implies $0 \leq \mu'_i$. Since $\mathbf{1}^T w = 0$, it follows that $\mathbf{1}^T \mu'$ is

$$\sum_i \mu'_i = \frac{1}{|\mathfrak{D}|} \left(|\mathfrak{D}| + \frac{1}{\max_i |w_i|} \sum_i w_i \right) = 1$$

Thus, both μ and μ' belong to the simplex $\Lambda_{\mathfrak{D}}$, and therefore represent randomized algorithms. Furthermore, $w = \lambda(\mu' - \mu)$. It follows that quantifying over μ and μ' can be replaced with quantifying over $v \in \mathbf{1}^\perp$ in the sense that predicate Equation (1)—which according to the above is

$$\forall \mu, \mu' \,.\, \forall \mathcal{F} \,.\, \exists \mathcal{F}' \,.\, (\mu - \mu')^T \left(\sum_{f \in \mathcal{F}} \frac{\vec{f}}{|\mathcal{F}|} + \sum_{f' \in \mathcal{F}'} \frac{\vec{f}'}{|\mathcal{F}'|} \right) = 0$$

—is equivalent to

$$\forall v \in \mathbf{1}^\perp \,.\, \forall \mathcal{F} \,.\, \exists \mathcal{F}' \,.\, \sum_{f \in \mathcal{F}} \frac{\vec{f}}{|\mathcal{F}|} + \sum_{f' \in \mathcal{F}'} \frac{\vec{f}'}{|\mathcal{F}'|} \in v^\perp \tag{4}$$

Define the vector $t(\mathcal{F})$ and the set of vectors $S(w)$ by

$$\begin{aligned} t(\mathcal{F}) &= \sum_{f \in \mathcal{F}} \frac{\vec{f}}{|\mathcal{F}|} \\ S(w) &= \left\{ w + t(\mathcal{F}') \,:\, \mathcal{F}' \subset \mathcal{Y}^{\mathcal{X}} \right\} \end{aligned}$$

Using the notation above—and trading quantification over $\mathcal{F}'$ in Equation (4) for quantification over u below—transforms Equation (4) into

$$\forall \mathcal{F} \,.\, \forall v \in \mathbf{1}^\perp \,.\, \exists u \in S(t(\mathcal{F})) \,.\, v^T u = 0 \tag{5}$$

which asserts that for every $\mathcal{F}$, any $v \in \mathbf{1}^\perp$ is orthogonal to some $u \in S(t(\mathcal{F}))$. It follows that Equation (5) is equivalent to the assertion that for all $\mathcal{F}$

$$\mathbf{1}^\perp = \bigcup_{u \in S(t(\mathcal{F}))} u^\perp \cap \mathbf{1}^\perp \tag{6}$$

Observe that $co\text{-dim}(\mathbf{1}^\perp) = 1$ whereas $co\text{-dim}(u^\perp \cap \mathbf{1}^\perp)$ is either 1 or 2. Hence, the only way Equation (6) can be true—since the union is finite—is that $co\text{-dim}(u^\perp \cap \mathbf{1}^\perp) = 1$ for some $u \in S(t(\mathcal{F}))$; thus u is parallel to $\mathbf{1}$. In other words,

$$\mathbf{1} \parallel \sum_{f \in \mathcal{F}} \frac{\vec{f}}{|\mathcal{F}|} + \sum_{f' \in \mathcal{F}'} \frac{\vec{f}'}{|\mathcal{F}'|}$$

for some $\mathcal{F}'$. □

Consider in the proof above that were $co\text{-dim}(u^\perp \cap \mathbf{1}^\perp)$ always 2, the complement of the union on the right-hand side of Equation (6) is dense and open in $\mathbf{1}^\perp$ (subspaces of $\mathbf{1}^\perp$ are closed). Since $\mathbb{Q}$ is a dense subfield of $\mathbb{R}$, that complement contains rational points. Therefore, the components of randomized algorithms can be restricted to $\mathbb{Q}$ without altering Theorem 13 or its proof (except to say $\lambda \in \mathbb{Q}$). If m is likewise required to take values in $\mathbb{Q}$, the $\vec{f}$ will also be rational (i.e., everything is computable).

Moreover, the domain over which benchmarks are quantified—for the definition of minmax distinctions in Equation (1) and the condition in Equation (2)—is intentionally unspecified. That domain may be chosen arbitrarily without altering Theorem 13 or its proof. In particular, the benchmarks $\mathcal{F}$ and $\mathcal{F}'$ may be restricted to satisfy an arbitrarily chosen predicate (for instance, they may be restricted to singleton sets).

The next result is that, for every measure, the performance of any randomized algorithm on any benchmark can be matched in a nontrivial way.

PROPOSITION 7: *For every measure m, randomized algorithm μ, and benchmark $\mathcal{F}$, there exist $\mathcal{F}'$ and μ' such that $\mathcal{E}_m(\mu, \mathcal{F}) = \mathcal{E}_m(\mu', \mathcal{F}')$ where $\mathcal{F}' \neq \mathcal{F}$ or μ' may be chosen arbitrarily.*

PROOF: If $\mathcal{F}$ is permutation closed, then any μ' works with $\mathcal{F}' = \mathcal{F}$ (sharpened NFL). Otherwise, $\mathcal{F}' = \sigma\mathcal{F} \neq \mathcal{F}$ for some σ; let $\mu' = \sigma^{-1}\mu$. Appealing to Proposition 1 and Expected Duality,

$$\mathcal{E}_m(\mu, \mathcal{F}) \;=\; \mathcal{E}_m((\sigma^{-1}\sigma)\mu, \mathcal{F}) \;=\; \mathcal{E}_m(\sigma^{-1}\mu, \sigma\mathcal{F}) \;=\; \mathcal{E}_m(\mu', \mathcal{F}')$$

□

One might hope for a variant of Proposition 7 asserting that for every m, μ, and $\mathcal{F}$, there exists $\mu' \neq \mu$ for which $\mathcal{E}_m(\mu, \mathcal{F}) = \mathcal{E}_m(\mu', \mathcal{F})$ when μ is nontrivial (nontriviality is essential; when m is injective and $m(\tilde{A}(f))$ is minimal, the Uniqueness Theorem implies that $\mathcal{E}_m(\mathcal{A}, \{f\}) = \mathcal{E}_m(\mu', \{f\})$ is equivalent to $\mathcal{A} = \mu'$). Let

$$\mu' = \lambda\mu + (1 - \lambda)\mathcal{F}_m(\mu)$$

Appealing to Linearity and Corollary 3,

$$\mathcal{E}_m(\mu', \mathcal{F}) \;=\; \lambda\mathcal{E}_m(\mu, \mathcal{F}) + (1 - \lambda)\mathcal{E}_m(\mathcal{F}_m(\mu), \mathcal{F}) \;=\; \mathcal{E}_m(\mu, \mathcal{F})$$

However, if $\mu = \mathcal{F}(\eta)$ for some randomized algorithm η, then Proposition 6 implies $\mu' = \mu$. Alternatively, one could appeal to Theorem 11 with

$$\mu' = \lambda\mu + (1 - \lambda)\sigma\mu$$

and $\sigma \in G_m(\mathcal{F})$. However, that may also fail. If $\mathcal{X} = \mathcal{Y} = \{0, 1\}$, then $\mathfrak{D} = \{\mathcal{A}, \mathcal{B}\}$ where

$$\mathcal{A}(f) = \langle (0, f(0)), (1, f(1)) \rangle$$
$$\mathcal{B}(f) = \langle (1, f(1)), (0, f(0)) \rangle$$

If benchmark $\mathcal{F}$ contains only the identity function, and the performance measure is

$$m(\langle y_0, y_1 \rangle) \;=\; 2y_0 + y_1$$

it then follows that $G_m(\mathcal{F}) = \{\iota\}$, because

$$\begin{aligned}
\mathcal{E}_m(\mu, \mathcal{F}) = \mathcal{E}_m(\mu', \mathcal{F}) &\iff \mu_{\mathcal{A}} + 2\mu_{\mathcal{B}} = \mu'_{\mathcal{A}} + 2\mu'_{\mathcal{B}} \\
&\iff \mu_{\mathcal{A}} + 2(1 - \mu_{\mathcal{A}}) = \mu'_{\mathcal{A}} + 2(1 - \mu'_{\mathcal{A}}) \\
&\iff \mu_{\mathcal{A}} = \mu'_{\mathcal{A}} \\
&\iff \mu = \mu'
\end{aligned}$$

The example above demonstrates that the following cannot be improved.

PROPOSITION 8: *Let $|\mathfrak{D}| > 2$. For every measure m, benchmark $\mathcal{F}$, and nontrivial randomized algorithm μ, there exist infinitely many $\mu' \neq \mu$ such that $\mathcal{E}_m(\mu, \mathcal{F}) = \mathcal{E}_m(\mu', \mathcal{F})$.*

PROOF: To simplify notation, let $\mathcal{E}$ abbreviate $\mathcal{E}_m$. Appealing to the finiteness of $\mathfrak{D}$, define

$$\mathcal{A}_+ = \arg\max_{\mathcal{A}} \left\{ \sum_{f \in \mathcal{F}} m(\tilde{\mathcal{A}}(f)) \right\}$$

$$\mathcal{A}_- = \arg\min_{\mathcal{A}} \left\{ \sum_{f \in \mathcal{F}} m(\tilde{\mathcal{A}}(f)) \right\}$$

$$\eta = \lambda \mathcal{A}_+ + (1 - \lambda)\mathcal{A}_-$$

It follows from Linearity that $\mathcal{E}(\eta, \mathcal{F})$ is a nondecreasing function of λ which attains the value $\mathcal{E}(\mu, \mathcal{F})$ for some $0 \leq \lambda' \leq 1$; let η be determined by λ'.

Case 1. $\lambda' \in \{0, 1\}$.

Then $\mu' = \eta \neq \mu$, because η is trivial, whereas μ is not. Moreover, infinitely many alternatives to μ' are provided by $\xi\mu + (1 - \xi)\mu'$ as ξ varies.

That leaves the case $0 < \lambda' < 1$. Since $|\mathfrak{D}| > 2$, there exists an algorithm $\mathcal{B}$ such that $\mathcal{A}_- \neq \mathcal{B} \neq \mathcal{A}_+$.

Case 2. $\mathcal{E}(\mathcal{B}, \mathcal{F}) = \mathcal{E}(\mathcal{A}_+, \mathcal{F})$

Let $\eta' = \lambda'\mathcal{B} + (1 - \lambda')\mathcal{A}_-$. Since $\mathcal{B}$ is a surrogate for $\mathcal{A}_+$—as far as performance is concerned—$\mathcal{E}(\eta', \mathcal{F}) = \mathcal{E}(\eta, \mathcal{F}) = \mathcal{E}(\mu, \mathcal{F})$. Moreover, $\eta \neq \eta'$ since they have different support. Infinitely many μ' are provided by $\mu' = \xi\eta' + (1 - \xi)\eta$ as ξ varies.

Case 3. $\mathcal{E}(\mathcal{B}, \mathcal{F}) \neq \mathcal{E}(\mathcal{A}_+, \mathcal{F})$

Let $\eta' = (\lambda' - \epsilon\lambda')\mathcal{A}_- + (1 - \lambda' - \alpha)\mathcal{A}_+ + (\epsilon\lambda' + \alpha)\mathcal{B}$ where

$$\alpha = \epsilon\lambda' \frac{\mathcal{E}(\mathcal{B}, \mathcal{F}) - \mathcal{E}(\mathcal{A}_-, \mathcal{F})}{\mathcal{E}(\mathcal{A}_+, \mathcal{F}) - \mathcal{E}(\mathcal{B}, \mathcal{F})}$$

and $\epsilon > 0$ is chosen sufficiently small to put the coefficients in the linear combination of $\mathcal{A}_-$, $\mathcal{A}_+$, $\mathcal{B}$ defining η' above in the open interval $(0, 1)$. Observe that

$$\epsilon\lambda'(\mathcal{E}(\mathcal{B}, \mathcal{F}) - \mathcal{E}(\mathcal{A}_-, \mathcal{F})) \;=\; -\alpha(\mathcal{E}(\mathcal{B}, \mathcal{F}) - \mathcal{E}(\mathcal{A}_+, \mathcal{F}))$$

It follows that

$$\begin{aligned}\mathcal{E}(\eta', \mathcal{F}) &= (\lambda' - \epsilon\lambda')\mathcal{E}(\mathcal{A}_-, \mathcal{F}) + (1 - \lambda' - \alpha)\mathcal{E}(\mathcal{A}_+, \mathcal{F}) + (\epsilon\lambda' + \alpha)\mathcal{E}(\mathcal{B}, \mathcal{F}) \\ &= \mathcal{E}(\eta, \mathcal{F}) + \epsilon\lambda'(\mathcal{E}(\mathcal{B}, \mathcal{F}) - \mathcal{E}(\mathcal{A}_-, \mathcal{F})) + \alpha(\mathcal{E}(\mathcal{B}, \mathcal{F}) - \mathcal{E}(\mathcal{A}_+, \mathcal{F})) \\ &= \mathcal{E}(\eta, \mathcal{F})\end{aligned}$$

Moreover, $\eta \neq \eta'$ since they have different support. Infinitely many μ' are provided by $\mu' = \xi\eta' + (1 - \xi)\eta$ as ξ varies. □

Note that the components of randomized algorithms can be restricted to $\mathbb{Q}$ without altering Proposition 8 or its proof (λ' is necessarily rational if performance measures are required to take values in $\mathbb{Q}$).

5 Conclusion

Definitions permit and theorems address algorithms which are not Turing computable. It should be appreciated that general results which hold even for algorithms that need not be Turing computable necessarily specialize to algorithms that are. Readers who embrace transfinite induction will no doubt appreciate the probability-free treatment in Section 3 which is conducted within a framework amenable to generalization (Rowe et al., 2009).

As a concession to readers who desire computability, $\mathcal{X}$ and $\mathcal{Y}$ were kept finite, and conclusions were stated as if randomized algorithms had components in $\mathbb{Q}$ (for instance, Propositions 4 and 8 could have claimed uncountable many μ' if components in $\mathbb{R}$ were permitted). Moreover, care was taken to point out the fact that Theorem 13 and Proposition 8 in no way require nonrational components.

In keeping with the outlook that the investigation of NFL-like properties need not be conflated with or encumbered by applications, this paper has focused on simplifying and extending theoretical results, and presenting new mathematical tools (Theorems, Corollaries, Lemmas, Propositions). Focused NFL (Theorem 7) simplifies and extends the previous account by Whitley and Rowe (2008), and γFNFL (Theorem 8) establishes the analogue for algorithms restricted to γ steps. Corollary 2 and Theorem 9 formalize cyclic aspects of focused benchmarks noted in Whitley and Rowe (2008).

A trend in NFL theory is the progression from permutations of the search space to group actions on functions and algorithms; our results and methods have for the most part followed that path. A series of propositions sort out how permutations act on randomized algorithms, establish linearity of expected average performance, and lead up to Expected Duality (Theorem 10). The demonstration of NFL for randomized algorithms by Rowe and Vose (2011) is first reviewed, and then sharpened (Proposition 4). By placing attention on the particular measure used, benchmark symmetries are extended to benchmark invariants whose properties admit a γFNFL-type interpretation (Theorem 11). The theoretical machinery of Rowe and Vose (2011) is also generalized.

The paper concludes with a geometric characterization for minimax distinctions (Wolpert and Macready, 1997) and basic results on performance matching.

Acknowledgments

The authors would like to thank Suzanne Sadedin, Marte Ramírez, and anonymous referees for comments on this manuscript. Ideas leading to Theorem 13 were initially formulated while M. D. Vose was visiting The University of Birmingham; he is grateful

for the gracious support provided by the School of Computer Science, and for valuable discussions with Jon E. Rowe and Alden H. Wright during that visit. This work was supported in part by NIH grant R01GM056693, and by a Howard Hughes Medical Institute Collaborative Innovation Award.

References

Allibone, S. A. (1879). *Prose quotations from Socrates to Macaulay*. Reprint Services Corp.

Auger, A., and Teytaud, O. (2007). Continuous lunches are free! *Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation, GECCO-2007*, pp. 916–922.

Auger, A., and Teytaud, O. (2010). Continuous lunches are free plus the design of optimal optimization algorithms. *Algorithmica*, 57:121–146.

Corne, D., and Knowles, J. (2003). No free lunch and free leftovers theorems for multiobjective optimisation problems. In *Evolutionary multi-criterion optimization*. *Lecture notes in computer science*, Vol. 2632 (pp. 327–341). Berlin: Springer.

Domingos, P. (1998). How to get a free lunch: A simple cost model for machine learning applications. *Proceedings of the AAAI98/ICML98, Workshop on the Methodology of Applying Machine Learning*, pp. 1–7.

Droste, S., Jansen, T., and Wegener, I. (1999). Perhaps not a free lunch but at least a free appetizer. *Proceedings of the First Genetic and Evolutionary Computation Conference, GECCO-1999*, pp. 833–839.

Droste, S., Jansen, T., and Wegener, I. (2002). Optimization with randomized search heuristics—The (a)nfl theorem, realistic scenarios, and difficult functions. *Theoretical Computer Science*, 287:131–144.

Goldberg, D. (1989). *Genetic algorithm in search, optimization and machine learning*. Reading, MA: Addison-Wesley.

Griffiths, E., and Orponen, P. (2005). Optimization, block designs and no free lunch theorems. *Information Processing Letters*, 94:55–61.

Haupt, R., and Haupt, S. (2004). *Practical genetic algorithms*, 2nd ed. New York: Wiley.

Holland, J. (1975). *Adaptation in natural and artificial systems*. The University of Michigan Press, Ann Arbor.

Igel, C., and Toussaint, M. (2001). On classes of functions for which no free lunch results hold. Retrieved from CoRR cs. NE-0108011

Igel, C., and Toussaint, M. (2004). A no-free-lunch theorem for non-uniform distributions of target functions. *Journal of Mathematical Modeling and Algorithms*, 3(4):313–322.

Mitchell, M. (1998). *An introduction to genetic algorithms*, 3rd ed. Cambridge, MA: MIT Press.

Radcliffe, N., and Surry, P. (1995). Fundamental limitations on search algorithms: Evolutionary computing in perspective. In van Leeuwen, J. (Ed.), *Computer science today*. *Lecture notes in computer science*, Vol. 1000 (pp. 275–291). Berlin: Springer.

Roberts, A., and Varberg, D. (1973). *Convex functions*. New York: Academic Press.

Rowe, J., and Vose, M. (2011). Unbiased black box search algorithms. In N. Krasnogor (Ed.), *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation*, pp. 2035–2042.

Rowe, J., Vose, M., and Wright, A. (2009). Reinterpreting no free lunch. *Evolutionary Computation*, 17:117–129.

Schumacher, C. (2000). Black box search—Framework and methods. PhD thesis, The University of Tennesee, Knoxville.

Streeter, M. (2003). Two broad classes of functions for which a no free lunch result does not hold. In *Genetic and evolutionary computation, GECCO 2003. Lecture notes in computer science*, Vol. 2724 (pp. 1418–1430). Berlin: Springer.

Wardetzky, M., Mathur, S., Kalberer, F., and Grinspun, E. (2007). Discrete Laplace operators: No free lunch. In A. Belyaev and M. Garland (Eds.), *Eurographics Symposium on Geometry Processing*, pp. 33–37.

Whitley, D., and Rowe, J. (2008). Focused no free lunch theorems. *Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation, GECCO-2008*, pp. 811–818.

Wolpert, D. (2001). The supervised learning no-free-lunch theorems. In *Proceedings of the 6th Online World Conference on Soft Computing in Industrial Applications*, pp. 25–42.

Wolpert, D., and Macready, M. (1995). No free lunch theorems for search. (SFI-TR-95-02-010)

Wolpert, D., and Macready, M. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82.

Woodward, J., and Neil, J. (2003). No free lunch, program induction and combinatorial problems. In *Genetic programming. Lecture notes in computer science*, Vol. 2610 (pp. 287–313). Berlin: Springer.