No Free Lunch and Free Leftovers Theorems for Multiobjective Optimisation Problems

David W. Corne[1], Joshua D. Knowles[2]

[1]Department of Computer Science, University of Reading, UK
d.w.corne@reading.ac.uk
[2]IRIDIA, ULB, Belgium
jknowles@ulb.ac.be

Abstract. The classic NFL theorems are invariably cast in terms of single objective optimization problems. We confirm that the classic NFL theorem holds for general multiobjective fitness spaces, and show how this follows from a 'single-objective' NFL theorem. We also show that, given any particular Pareto Front, an NFL theorem holds for the set of all multiobjective problems which have that Pareto Front. It follows that, given any 'shape' or class of Pareto fronts, an NFL theorem holds for the set of all multiobjective problems in that class. These findings have salience in test function design. Such NFL results are cast in the typical context of *absolute* performance, assuming a performance metric which returns a value based on the result produced by a single algorithm. But, in multiobjective search we commonly use *comparative* metrics, which return performance measures based non-trivially on the results from two (or more) algorithms. Closely related to but extending the observations in the seminal NFL work concerning *minimax distinctions* between algorithms, we provide a 'Free Leftovers' theorem for comparative performance of algorithms over permutation functions; in words: over the space of permutation problems, *every* algorithm has some companion algorithm(s) which it outperforms, according to a certain well-behaved metric, when comparative performance is summed over all problems in the space.

1 Introduction

The so-called 'No Free Lunch' theorem (NFL) is a class of theorems concerning the average behaviour of 'black-box' optimization algorithms over given spaces of optimisation problems. The primary such theorems were proved in [15], and disseminated to the evolutionary computation community in [16], showing that, when averaged over all possible optimization problems defined over some search space X, no algorithm has a performance advantage over any other. A considerable debate has grown over the importance and applicability of these theorems, and the consequences for the field of optimization in general are much discussed in [16], and other work reviewed briefly below. Perhaps the main consequence for the majority of researchers in optimization, particularly those investigating generic *algorithms* rather than those focused on particular *problems*, is that the NFL result denies any claim of the form 'algorithm A is

better than algorithm *B*', for *any* pair of algorithms *A* and *B*. For example, the statement 'Genetic Programming outperforms random search' is false. NFL points out that any such statement must be carefully qualified in terms of the set of problems under consideration. As we see later in a brief review, it has been shown that NFL does not hold over certain subsets of problems, and proofs in this vein are a growing area. In fact, of course, this is easy to show (at least empirically), when the set of problems under consideration is small, such as, perhaps, three particular test problems on which well-designed comparative experiments are performed. However, generalization of any such '*A* is better than *B*' statement to all problems in a large *class* of problems is always unwise without convincing argumentation.

In this paper we are concerned with NFL on spaces of *multiobjective problems*. We will confirm that NFL results hold over such spaces, and in particular we will show that NFL holds over spaces of problems which share a particular (or particular class of) Pareto front(s). We also prove a 'Free Leftovers' theorem for *comparative* metrics; these are technically distinct from the straightforward 'absolute' metrics typically assumed in NFL work, and are the most commonly applied metrics when comparing the performance of multiobjective algorithms..

The remainder is set out as follows. In section 2 we briefly review NFL-related research in (largely) the evolutionary computation community to date, and then start to discuss the question of NFL in the context of multiobjective search. Section 3 then provides some standard preliminaries and notation, and in section 4 we confirm the standard NFL result in multiobjective problem spaces, via showing how it would follow from a 'single-objective' NFL theorem. Section 5 then proves NFL results for interesting subsets of multiobjective spaces: those with particular (or particular classes of) Pareto fronts; this section also notes that a more general (than shown in section 4) NFL result holds, where we allow the range of fitnesses in each objective to be different. Section 6 brings up the issue of *comparative metrics*, which are of particular interest in multiobjective optimization. We show, with our 'Free Leftovers' theorem, that the spirit of NFL does *not* hold in general when we use comparative metrics; this is consistent and closely linked to the observations concerning 'minimax distinctions' in the seminal NFL work. We briefly discuss and conclude in section 7.

2 About NFL and the Multiobjective Case

Work on NFL since the seminal papers has broadly concerned two themes; one theme is new proofs of NFL/and or interesting subclasses of problems for which NFL holds; the other theme concerns classes of problems for which NFL doesn't hold (i.e. proofs of 'Free Lunch' theorems). We first briefly review some work on NFL since [15], covering first theme one and then theme two.

Regarding theme one, quick on the heels of [15], [10] provided another proof of NFL, and reviewed the implications of NFL with a focus on representation issues. In particular, suggestions for representation/operator performance measures were proposed, given any well-defined set over which NFL does not hold. More recently, [14] showed that a general NFL result holds over the set of permutation functions, that is,

problems in which every point in the search space has a unique objective function value. In [14] it is shown that these functions are, on average, 'unusually' complex (uncompressible) – more so than the NP-complete class – implying that this set may not be of practical interest. Furthermore, it is shown that all functions in this set have identical summary statistics, implying that the latter cannot be used to guide search.

A 'sharpened' NFL result is proved in [11], which shows that the NFL holds over a set of functions *F*, if and only if *F* is closed under permutation (c.u.p.). They also show that the compressibility of a set of functions *cannot* be appealed to, as a means of 'escaping' NFL: thus, a highly compressible function such as a needle-in-a-haystack function, when closed under permutation, forms a set over which NFL still holds. Expoiting the result in [11], Igel and Toussaint [7] consider its implications. They prove that nearly all subsets of a typical space of problems are not c.u.p, and, in particular, some practically interesting problem classes are not c.u.p. Hence, certain practically interesting classes of problems are not subject to NFL.

Theme two concerns identifying and focusing on particular, usually 'small', subsets of problems for which NFL does not hold. For example, in [4] it was shown that for a certain very simple set of objective functions, and a particular restriction on this set, NFL does not hold, and adaptive algorithms have provably better performance than non-adaptive ones on this set. In [13] it is proven that Gray encoding is worse than binary encoding over a set of functions with $2^L - 1$ optima, from which it follows that Gray is better than binary over all *other* functions. In [1], a free lunch is proved for a simple search algorithm, applied over a set of multivariate polynomial functions of bounded degree *d*. This is of general interest because it is shown that performance is better than random search up to a very large value of *d*, such that many functions of practical interest could be approximated by the polynomial. Finally, in [5] it is proved that the time to optimize a specific, realistic, and simply defined instance of MAXSAT is exponential for a heuristic designed specifically for the MAX-SAT class, and it is further hypothesised that all 'reasonable' heuristic algorithms will exhibit this performance. This shows that even in complexity-restricted problem classes, in which NFL does not hold, it still 'almost' holds since expected worst-case performance of an optimizer may be very poor.

It so happens that, in the prominent published NFL work so far, the *single-objective* case has been assumed, at least implicitly. For example, [16] talks of the space of fitnesses *Y* as a space of 'values' and indeed cast the set of problems as 'cost functions', and give examples of performance measures which relate to finding minima in that space; when specific examples of functions are given in [11], they are numerically valued; similar can be said of [1], [5] and so forth. Meanwhile, universally, NFL theorems are notated in terms of a space of problems (functions) $f : X \to Y$, clearly implying single-objective optimization. One exception we have found is the early work of [10], which once notes that the domain of the problems concerned could be multiobjective.

That NFL holds over multiobjective optimization is hence not clear from an abrupt reading of the literature in the literature. Since *Y* only needs to be a finite set, we could glibly and simply say 'it does', however this ignores one key difference between multiobjective and single objective optimization: in the multiobjective case, algorithm

performance is commonly done via a *comparative metric*, which is non-trivially distinct from 'absolute' performance metrics. As it turns out, NFL does *not* in hold in general in such a scenario. As we'll see, the fact that fitnesses are vectors rather than values is immaterial with regard to NFL in the context of absolute performance metrics (where performance of an algorithm is a measure of the result produced by that algorithm alone), however as we just indicated, the greater emphasis in multiobjective search on comparative metrics renders this less applicable, and in general there *is* a free lunch (or, rather, 'freee leftovers') when certain well-behaved comparative metrics are involved.

Regarding the embracing of multiobjective optimization by the 'absolute performance' NFL theorem, one may still feel uneasy about this, since various aspects and elements of the various 'single-objective' proofs, which at least implicitly assume single-objective optimization, need to be revisited. So, rather than simply say 'since *Y* only needs to be a finite space, NFL holds for multiobjective optimization', we also provide a proof which shows how this follows from the 'single-objective' case; i.e. if we assume only that NFL for absolute performance holds on single objective problems, we can derive its truth for multiobjective problems in a more interesting way. This is done in section 4, following preliminaries and notation in section 3. As indicated earlier, in section 5 we then look at certain salient subsets of multiobjective problems, in particular those which share a certain Pareto front, and in section 6 we look at the issue of comparative metrics.

2 Preliminaries and Notation

Using similar notation to that in [16], we have a search space *X* and a set of 'fitnesses' *Y*, and we will generally assume maximization. A single-objective optimisation problem *f* is identified with a a mapping $f : X \to Y$ and $F = Y^X$ is the space of all problems. Its size is clearly $|Y|^{|X|}$. A black-box optimisation algorithm *a* generates a time-ordered sequence of points in the search space, associated with their fitnesses, called a *sample*. A sample of size *m* is denoted $d_m \equiv \{(d_m^X(1), d_m^Y(1)),...,(d_m^X(m), d_m^Y(m))\}$, where, in general, $d_m^X(i)$ is the *i*th distinct point in the search space *X* visited by the algorithm, and $d_m^Y(i)$ is the fitness of this point. We will also use the term d_m^X to denote the vector containing only the $d_m^X(i)$, and similarly d_m^Y will indicate the time-ordered set of fitness values in the sample. The space of all samples of size *m* is denoted $D_m = (X \times Y)^m$, and the space of all samples is denoted *D*, and is simply the union of D_m for all positive integers *m*.

An optimisation algorithm *a* is characterised as a mapping: $a : D \to X$ with the restriction that $a(d_m) \notin d_m^X$. That is, it does not revisit previously visited points. This restriction, plus that of the algorithm being *deterministic*, is imposed for simplicity but ultimately does not affect the generality of the resulting theorems.

The (absolute) performance *a* after *m* iterations on problem *f* is measured using $P(d_m^y \mid f, m, a)$; this is the probability of a particular sample being obtained after iterating *a* for *m* iterations on problem *f*. The NFL theorem (for static cost functions) is this [15, 16]: for any pair of algorithms a_1 and a_2,

$$\sum_f P(d_m^Y \mid f, m, a_1) = \sum_f P(d_m^Y \mid f, m, a_2) . \quad (1)$$

Hence, when summed over all cost functions, the probabilities of obtaining a particular sample using a_1 amount to the same as that of using a_2. The key message is that, since *performance* of an algorithm is a function only of the sample it generates, when performance is summed over the whole space of problems, the result is the same for any pair of algorithms. We can define a performance measure as a function $\Phi : d_m^Y \to \Re$ (with the choice of $\Re$ being natural, and relatively arbitrary), and a corollary of equation (1) is:

$$\sum_f P(\Phi(d_m^Y) \mid f, m, a_1) = \sum_f P(\Phi(d_m^Y) \mid f, m, a_2) . \quad (2)$$

Now, we need to establish notation that will be helpful in the multiobjective case. A multiobjective problem with *k* objectives can be cast as a function $g : X \to Y^k$, in which, for simplicity at this stage, we assume that the space of fitness values in each objective is the same (*Y*). We now need new notation for a multiobjective sample, which we will denote as follows

$$v_m = \{(d_1^X(1), v_1^Y(1)), \ldots (d_m^X(m), v_m^Y(m))\} ,$$

where $v_j^Y(j) \in Y^k$; i.e. $v_j^Y(j)$ is the *k*-objective vector of fitnesses resulting from evaluation of $d_j^X(j)$. Finally, using *V* to denote the set of all multiobjective samples, we define a multiobjective algorithm q as a function $q : V \to X$ with the restriction that $q(v_m) \notin v_m^X$, with notation assuming the obvious meaning.

3 No Free Lunch on Multiobjective Problem Spaces

What we wish to confirm is that, for any two multiobjective algorithms q_1 and q_2,

$$\sum_g P(v_m^Y \mid g, m, q_1) = \sum_g P(v_m^Y \mid g, m, q_2) . \quad (3)$$

As indicated, we choose to do so by showing how it follows from a 'single-objective only' NFL theorem. The sketch of the proof is as follows – the idea is straightforward, but the book-keeping issues will take up a little space. We will first assume that equation (3) is false, and hence there is some pair of multiobjective algorithms q_1 and q_2 whose overall performance differs over the space of all *k*-objective problems. Via showing that a $1 \leftrightarrow 1$ mapping exists between the space of *k*-objective problems and a certain space of single objective problems, we can then show that this implies that there are two single-objective algorithms who differ in performance over this single objective space. This must be false (by equation (1)), and hence equation (3) follows by *reductio ad absurdum*. So, we now formally state and prove an NFL theorem for static multiobjective problems.

Theorem 1 Equation (3) holds for any two multiobjective algorithms q_1 and q_2.

Proof of Theorem 1:

Assume that equation (3) does not hold, thus we can state that, for a particular pair of multiobjective algorithms q_1 and q_2,

$$\sum_g P(v_m^Y \mid g, m, q_1) > \sum_g P(v_m^Y \mid g, m, q_2), \tag{4}$$

in which, without loss of generality, we cast q_1 as the algorithm which performs best overall (when the performance measure correlates with achieving the sample v_m^Y).

Now consider the mapping $s: Y^k \to W$, where $W = \{1,2...,|Y|^k\}$, defined as follows, where $z \in Y^k = \{z_1, z_2, ..., z_k\}$:

$$s(z) = \sum_{i=1}^{k} z_i \, |Y|^{i-1}.$$

This is clearly a one-to-one mapping; we omit a proof, but simply note that this map is between a number $s(z)$, and a unique representation of that number in base $|Y|$. The set W simply enumerates, or uniquely labels, all possible multiobjective vectors in Y^k.

Now consider the mapping $s \circ g : X \to W$. This is a single objective function on X. Further, from any multiobjective algorithm $q: V \to X$ we can construct a corresponding single objective algorithm $q': D \to W$ as follows:

$$q'(v'_m) \equiv s(q(v_m)),$$

where $v'_m = \{(d_1^X(1), s(v_1^Y(1))), ..., (d_m^X(m), s(v_m^Y(m)))\}$. Hence, every k-objective algorithm q defined on the search space X and involving the space of fitnesses Y^k corresponds to a unique single objective algorithm defined on the search space X and involving the single-objective space of fitnesses W, and *vice versa*.

Recall our assumption (equation (4)); now, we can say

$$P(v_m^Y \mid g, m, q) = P(v'_m \mid s \circ g, m, q'),$$

since there are one-to-one correspondences between v_m^y and v'_m, g and $s \circ g$, and q and q'. It trivially follows (from this and equation (4)) that:

$$\sum_{s \circ g} P(v'_m \mid s \circ g, m, q'_1) > \sum_{s \circ g} P(v'_m \mid s \circ g, m, q'_2).$$

However, we notice that the space of all functions $s \circ g$ corresponds precisely to the space of all functions $f: X \to W$, that both q'_1 and q'_2 are single objective algorithms from D to X (where fitnesses in d_m^Y are from W), and that any sample v'_m corresponds uniquely to a sample d_m^Y from D. Hence, we can write

$$\sum_f P(d_m^W \mid f, m, q'_1) > \sum_f P(d_m^W \mid f, m, q'_2).$$

But this essentially expresses a free lunch for single-objective algorithm q'_1 at the expense of single-objective algorithm q'_2, over the space of all single-objective prob-

lems $f : X \to W$. By equation (3), this is false, and hence we conclude our original assumption is wrong, and so the following holds:

$$\sum_g P(v_m^Y \mid g, m, q_1) = \sum_g P(v_m^Y \mid g, m, q_2); \quad (5)$$

and hence

$$\sum_g P(\Psi(v_m^Y) \mid g, m, q_1) = \sum_g P(\Psi(v_m^Y) \mid g, m, q_2),$$

in which $\Psi : V \to \Re$ is an arbitrary absolute performance metric for multiobjective samples. So, for any *k*, any search space *X*, and any space of fitnesses *Y*, an NFL result holds for the complete space of *k*-objective problems $g : X \to Y^k$.

4 NFL on Certain Subsets of Multiobjective Problem Spaces

A further (and perhaps more interesting) issue concerns the shape of the Pareto Front; this is a discussion which we will now motivate. The fact that there is no free lunch over the set of all *k*-objective problems means that any multiobjective algorithm (such as SPEA [17]) has no absolute performance advantage over any other algorithm (e.g. NSGA-II [3]), over the (understood) space of all multiobjective problems. This result, however, concerns averaging performance over all possible problems, and hence including all possible shapes of Pareto front. It could be argued, however, that particular algorithms have particular advantages correlated with the shape of the Pareto Front. E.g. we may imagine that SPEA with a large internal population size will do better on problems with a highly discontinuous Pareto front than would SPEA with a small internal population size. We can show, however, that for a particular shape of Pareto front, an NFL result holds over the space of all multiobjective problems which share that shape of Pareto front. We proceed towards proving this, and then note the case of 'general' multiobjective problems.

Consider again the space of all *k*-objective problems $g : X \to Y^k$, and the set $W = \{1,2....,|Y|^k\}$ which labels the space of *k*-objective fitness vectors. It will help to illustrate a small example of the correspondence between *W* and Y^k as follows. Let $Y = \{1,2,3\}$ and $k = 2$, and therefore $W = \{1,...,9\}$. The correspondence between Y^k and *W* is given by Table 1. We will refer to this Table in illustrations next.

Table 1: A simple correspondence between a space of multiobjective fitness vectors Y^k (in this case 2-objective, where $k = 2$ and each objective has 3 possible values) and a single-objective space *W* of the appropriate size (in this case 9). Four points are bold, representing a particular subset of problems, with the underlined representing a particular Pareto front

Vector from Y^k	**(1,1)**	**(1,2)**	(1,3)	**(2,1)**	(2.2)	(2,3)	**(3,1)**	(3,2)	(3,3)
'Label' from *W*	**1**	**2**	3	**4**	5	6	**7**	8	9

Consider a subset $B \subset Y^k$, and the corresponding labels $L(B) \subset W$. E.g., in table 1, the bold entries identify $B = \{(1,1),(1,2),(2,1),(3,1)\}$ and $L(B) = \{1,2,4,7\}$. Any such subset has its own non-dominated 'front' $N(B)$, in this case $N(B) = \{(1,2),(3,1)\}$ and $L(N(B)) = \{2,7\}$.

If we are interested in a *particular* Pareto front *C* which exists for at least some multiobjective problems in $g : X \to Y^k$, we can pick out the points from Y^k which constitute this front, and then easily construct the maximal subset $Q \subseteq Y^k$ such that $N(Q) = C$ and the remainder of *Q* is made up of all points in Y^k which are *dominated* by at least one point in *C*. This corresponds to, in table 1, choosing the front represented by $C = \{(1,2),(3,1)\}$, in which case $Q = \{(1,1),(1,2),(2,1),(3,1)\}$.

Clearly we can construct such a *Q* for any given Pareto front *C* in the space of problems $g : X \to Y^k$, for any *X*, *Y*, and *k*. Given *C* and *Q*, we can ask: does NFL hold over the space *H* of all multiobjective problems $h : X \to Q$, where a restriction on $h \in H$ is that the domain of *h* contains *C*? The restriction ensures that every function $h \in H$ has the same Pareto front *C*, but in general *h* may only contain additionally in its domain a subset of the remainder of *Q*. Hence *H* is the space of all possible multiobjective problems $g : X \to Y^k$ whose Pareto front is *C*. The anwer is affirmative, and we prove it simply, by using this result from [11]:

Theorem 2:
For any two algorithms *a* and *b*, equation (1) holds iff the set of functions *F* concerned is closed under permutation (c.u.p.).

This is (effectively) stated and proved in [11]. Recall that a function $f : X \to Y$ is an assignment of a $y \in Y$ to each $x \in X$, and can hence be viewed as a vector of |*X*| elements from *Y*, where element $f_i = y$ iff $f(x_i) = y$. If a set of functions *F* is 'closed under permutation', this simply means that we can arbitrarily permute the elements of *f*, i.e. we apply a bijective mapping $\pi : X \leftrightarrow X$ and the resulting new function f', with $f'(\pi(x)) = f(x)$, remains a member of *F*. We can now state and prove the following:

Theorem 3:
Given the set of multiobjective functions $g : X \to Y^k$, and a subset *H*, where $h \in H \Leftrightarrow h : X \to Q$ and $\forall c \in C, \exists x \in X, h(x) = c$, where *C* is the Pareto subset of $Q \subseteq Y^k$, equation (5) holds over the *H*. That is, given a particular Pareto front, NFL holds over the space of multiobjective problems which share that Pareto front.

Proof of Theorem 3:
Given *H*, *C* and *Q* as defined in the theorem, we can identify a corresponding subset *J* of single-objective functions, and ask if that set is closed under permutation. This set is $j \in J \Leftrightarrow j : X \to L(Q)$ and $\forall c \in C, \exists x \in X, j(x) = L(c)$. Consider any $j \in J$, and any point *x* such that $j(x) = L(c)$ for some $c \in C$. A permutation $\pi : X \leftrightarrow X$ leads to a function j' such that $j'(\pi(x)) = j(x)$, so clearly there is a point in the new function ($\pi(x) \in X$) whose fitness value is $L(c)$. Thus, every point in $L(c)$ is in the domain of j'. Next, suppose the domain of j' contains some point $u \notin L(Q)$; this means there is some point *u* such that $j'(x) = u$. But, $j'(x) = j'(\pi(z)) = j(z)$ for some $z \in X$,

and the domain of j is $L(Q)$; this is a contradiction, so j' cannot contain a point outside $L(Q)$. This proves that $j' \in J$, and hence the set J is closed under permutation.

We know that NFL holds for the set of single objective problems J. By appeal to the one-to-one correspondences between H and J, Q and $L(Q)$, C and $L(C)$, and by using machinery developed earlier to produce one-to-one correspondences between multiobjective samples and single objective samples, and multiobjective and single objective algorithms, we can conclude that

$$\sum_{h \in H} P(v_m^Y \mid h, m, q_1) = \sum_{h \in H} P(v_m^Y \mid h, m, q_2),$$

where q_1 and q_2 are multiobjective algorithms $q_1, q_2 : V \to X$ and H is as described.

This has shown that, over all k-objective problems with a particular Pareto front C, there is no free lunch in terms of absolute metrics. We can still ask about the set of k-objective problems with a particular *shape* of Pareto front C. We do not define how to characterize a 'shape' here, but would expect that any such definition would amount to enumerating a particular finite set of k-objective Pareto fronts $S = \{C_1, C_2, ..., C_n\}$ where $C_i \subset Y^k$ for each i. Given such a set S, we can easily prove:

Theorem 4:
Given any particular *shape* (or other characterisation) of Pareto front, corresponding to a set of specific fronts $S = \{C_1, C_2, ..., C_n\}$ where $C_i \subseteq Y^k$ for each i, there is no free lunch over the set of all k-objective problems whose Pareto front is in S.

Proof of Theorem 4:
We know that, for each individual C_i, the corresponding space of multiobjective problems H is closed under permutation. Since a union of such sets is also closed under permutation, it follows that the set S is closed under permutation.

Finally, we have confirmed that the absolute performance of black-box multiobjective algorithms is subject to NFL over the space of all problems $g : X \to Y^k$; however, in general, a multiobjective problem will typically have a different range of fitnesses for each objective. That is, a given k-objective problem will be of the form $g : X \to Y_1 \times Y_2 \times ... \times Y_k$, where $|Y_i| > 1$ for each i, but with no other constraint (such as $|Y_i| = |Y_j|$ for different i and j). One way to confirm that NFL holds over this more general multiobjective problem space is by appeal to the results about Pareto fronts. Any Pareto front defined in the space $Y_1 \times Y_2 \times ... \times Y_k$ corresponds to a Pareto front for some set of problems in ${Y_M}^k$, where Y_M has the largest cardinality of the $Y_i, i \in \{1, ..., k\}$. So, that NFL holds over this general space can be shown as a special case of Theorem 4, where the set S includes all Pareto fronts of ${Y_M}^k$ which are also Pareto fronts of $Y_1 \times Y_2 \times ... \times Y_k$.

5 Comparative Metrics

The basis of NFL theorems published to date concerns there being ultimately no performance advantage for one black box algorithm over another, when averaged over a suitable collection of problems. In NFL work published thus far, it tends to be assumed that the measure of performance of algorithm *a* is purely a function of the sample yielded by algorithm *a* when run on a problem. So far, we have confirmed that NFL holds over certain spaces of multiobjective problems, and what follows from this, in terms of statements about algorithm *performance*, can only be stated with respect to 'absolute' performance metrics – i.e. performance measures which are based only on the sample produced by a single algorithm. Several such metrics are occasionally used in multiobjective optimisation studies (such as hypervolume [17], or others which can be found in [12]). However, a more common and favoured way to measure the performance of a multiobjective algorithm is via a metric: $\Psi : V^d \to \Re$, where $d > 1$ typically $d = 2$), where the metric takes in the results of two or more algorithms and produces a measure of relative performance; Zitzler's 'Cover' metric is a commonly employed example [17], while more sophisticated but currently less used examples can be found in [6].

Before we consider NFL in relation to such metrics, it is worth mentioning that they are far from trivial counterparts to 'absolute' performance metrics. E.g., some such metrics yield intransitive performance relationships between algorithms. See [8] for a full treatment of multiobjective comparison metrics and their properties in this sense, as well as [9] and [18].

It turns out that there *is* a potential 'free lunch' when comparative performance is concerned, at least for certain comparative metrics $\Psi : V \times V \to \Re$. This arises owing to an asymmetry which occurs when we look at the performance of a pair of algorithms on the same problem. Every algorithm will generate a given sample on some problem in the set, but the *pair* of samples generated by two algorithms on some problem may not be generated by the same pair of algorithms on any other problem.

Preliminaries to what we call a 'Free Leftovers' theorem are as follows, with the earlier preliminaries (section 2) already in place. Let the set of algorithms be *A*, let the set of all comparative metrics be *C*, and let the set of problems be *F* (we will use the notation of [16] for simplicity, hence often oriented towards the single objective case but not ruling out the multiobjective case). A comparative 'experiment', *E*, is a function $\mathrm{E} : A \times A \times F \times \mathrm{N} \times C \to \Re$ which separately runs two algorithms, $a, b \in A$ on a function $f \in F$ for $m \in N$ distinct function evaluations each, and then compares the resulting two samples using a comparative metric $\Psi \in C$, where $\Psi : D \times D \to \Re$. Hence, given two algorithms *a* and *b*, and a problem *f*, running experiment $\mathrm{E}(a, b, f, m, \Psi)$ produces the result $\Psi(v_m^Y, w_m^Y)$, where v_m^Y is the sample produced by *a* on *f*, and w_m^Y is the result produced by *b* on *f*.

Importantly, we will be concerned here only with comparative metrics which are well-behaved in the following reasonable sense. Where *v* and *w* are samples, $v = w \Rightarrow \Psi(v, w) = 0$, and $\Psi(v, w) + \Psi(v, w) = 0$. Hence, when samples are the same, the 'performance difference' is 0, and in particular the metric is symmetric. That is, assuming without loss of generality that $\Psi(v, w) = r > 0$ indicates that (the algorithm

which produced) sample *v* is better than (the algorithm which produced) sample *w*), the same metric will indicate that $\Psi(w,v) = -r$; hence, that *w* is worse than *v*, to the same extent that *v* is better than *w*. Now, familiarity with the NFL theorems and intuition may suggest that, when summed over all problems *f*, $\mathrm{E}(a,b,f,m,\Psi)$ will amount to zero, since the performance advantage for *a* on some problem will presumably be countered by advantages for *b* on others. However, this is not so in general. In fact, [16] proves by example that

$$\exists a,b \in A, \sum_f P(d_m^Y \mid f,m,a) \cdot P(e_m^Y \mid f,m,b) \neq \sum_f P(d_m^Y \mid f,m,b) \cdot P(e_m^Y \mid f,m,a) . \quad (6)$$

In [16] it is called a *minimax distinction* when this type of asymmetry exists between algorithms, and [16] was concerned with the consequences as regards whether a large performance advantage for *a* over *b* on one function is matched by a similar advantage for *b* on another function, or instead by several small advantages for *b* on other functions. Here, however, we see how this type of result relates directly to the issue of comparative performance, which is especially salient in multiobjective optimisation. In the following, we build on this result and express it in our 'Experiment' notation, and derive a result which guarantees that, at least on a certain large space of problems, every algorithm has a 'straw man' – some other algorithm *b* which it outperforms on the whole over all functions in a space according to some metric. However, we only show here that this holds over spaces of permutation functions. Such a space can be defined as: $\Pi = \{ f : X \leftrightarrow Y \mid \forall x, y \in X, f(x) = f(y) \Rightarrow x = y \}$, i.e. every point in the search space has a unique fitness. Our 'Free Leftovers in Permutation Spaces' theorem shows that *every* algorithm has a comparative performance advantage in a permutation space, (over some other algorithm, and given a particular metric).

Theorem 5 (Free Leftovers in Permutation Spaces)
Given any algorithm *a*, there are other algorithms *b* and well-behaved comparative metrics Ψ such that, when summed over all problems in a permutation space, *a* outperforms *b*. Formally:

$$\forall a \in A, \exists b \in A, \Psi \in C : \sum_{f \in \Pi} E(a,b,f,m,\Psi) > 0 . \quad (7)$$

Proof of Theorem 5
We start by showing a proof of the following statement:

$$\exists a,b \in A, \Psi \in C : \sum_f E(a,b,f,m,\Psi) \neq 0 , \quad (8)$$

which proposes the existence of at least one pair of algorithms whose summed comparative performance, for some well behaved metric, is nonzero. This is not confined to permutation spaces, and a theorem from which this follows is already proved by example in [15], however we also give a proof by example, since in so doing we rehearse and clarify some concepts in the proof of Theorem 6.

Towards proving equation (8), let $X = \{1,2,...,p\}$ and $Y = X$, with $p \in N$, and let problem $f_1 : X \to Y$ simply be defined by: $f_1(x) = x$. Further, let algorithm *a* be the

algorithm which simply enumerates X from point 1 to point p. So, when we apply a to f_1 for m $(\leq p)$ iterations, the sample v_m^Y obtained is simply $\{1,2,...,p\}$. Now consider algorithm b, which enumerates X in a different order as follows: the first point b visits is point 2, and thereon the nth point visited is point n+1, for $n < p$, and the final point visited is point 1. So, on problem f_1, algorithm b generates the sample $w_m^Y = \{2,3,...,p,1\}$. Now, we will construct a problem f_2 on which algorithm a produces sample v_m^Y. This is easily done – a visits points in a particular order, and so we simply arrange f_2 such that the appropriate fitnesses are at the points visited so as to generate w_m^Y. That is, we define f_2 as follows: $f_2(x) = x+1$ for $1 \leq x < p$, and $f_2(p) = 1$. Algorithm a will now generate w_m^Y on this function (and note that this is the *only* function on which algorithm a will generate this sample), but algorithm b does *not* generate v_m^Y; in fact algorithm b generates: $\{3,4,...,p,1,2\}$.

So far, to couch it in the more familiar notation of the NFL theorems, this has shown, by example, that equation (6) is true, which was proved in [15, 16]; we next derive equation (8) from this, which indicates what this means explicitly in terms of well-behaved comparative performance metrics, of the type commonly used in multiobjective optimisation.

Let algorithm a generate sample $v*_m^Y$ on problem f, and let algorithm b generate $w*_m^Y$ on the same problem. Further, let a and b be a pair, whose existence is guaranteed above, such that for any function g on which a generates sample $w*_m^Y$, b does *not* generate $v*_m^Y$. Now, define a well-behaved comparative performance metric Ψ by setting $\Psi(v*_m^Y, w*_m^Y) = 1$ (entailing $\Psi(w*_m^Y, v*_m^Y) = -1$), and $\Psi(v, w) = 0$ for all other distinct pairs of samples. When we run experiment $\mathrm{E}(a, b, f, m, \Psi)$, we get the result $\mathrm{E}(a, b, f, m, \Psi) = 1$, since this experiment results in a pair of samples for which the metric will provide this result. However, given the preceding, there no function g for which $\mathrm{E}(a, b, g, m, \Psi) = -1$, which leads to and proves equation (8).

Now, on to Theorem 6. We will illustrate with a simple example as we go along. Let $a \in A$ and $f \in \Pi$; a generates a sample vaf_m on f. For example, f may be the function $f(x) = x, x \in \{1,...5\}$, with m = 3, and so $vaf_m = \{(1,1),(2,2),(3,3)\}$, and therefore $vaf_m^Y = (1,2,3)$. We will show that another algorithm b can always be constructed with the following properties: b produces a sample $vbf_m \neq vaf_m$ on f, (which, given $f \in \Pi$, guarantees that $vbf_m^Y \neq vaf_m^Y$), however there is no function $g \in \Pi$ such that both a produces sample vbf_m^Y on g and b produces sample vaf_m^Y on g. To show this, we start with the samples vaf_m and vbf_m. We first construct b so as to ensure that these samples are different in terms of performance (i.e. $vbf_m^Y \neq vaf_m^Y$), but that they 'overlap' (by which we mean $\exists x \in X, x \in vaf_m^X \wedge x \in vbf_m^X$ – i.e. the same point in the search space occurs at some position (not necessarily the same) in each sample). We can handle these conditions at one stroke by just arranging that $b(\phi) = vaf_2^X(2)$. I.e., the first point visited by b is the second point visited by a when a runs on f. Continuing our simple example, we have arranged b such that $vbf_m = \{(2,2),...\}$ – i.e. the first point visited by b is the second point visited by a, which ensures that, on problem f, different samples are generated by a and b, however some point in the search space (2) is visited by both. Now, assume that we have a function g on which b generates a sample vbg_m, where $vbg_m^Y = vaf_m^Y$, and on which a generates a sample vag_m where $vag_m^Y = vbf_m^Y$. So, in our example, we are assuming that we have some mapping

$g := \{1,...,5\} \leftrightarrow \{1,...,5\}$ on which algorithm *a* generates vbf_m^Y (which so far we know starts with 2), and *b* generates (1,2,3) First, since *a* is constrained to first visit $a(\phi) = vaf_1^X(1)$, we need $g(vaf_1^X(1)) = vbf_1^Y(1)$. In our example, since *a* visits point 1 first, we must have $g(1)=2$. Similarly, since $b(\phi) = vbf_1^X(1)$, we need $g(vbf_1^X(1)) = vaf_1^Y(1)$. In our example, we must have $g(2)=1$, However, now recall that $b(\phi) = vaf_2^X(2)$, which is the second point visited by *a* on *f*. The second point visited by *b* on *g* must have the same fitness as this, in order to ensure $vbg_m^Y = vaf_m^Y$. In our example, this means that we have to arrange g such that $g(z)=2$, where *z* is the second point visited by *b* on *g*. But, we are free to construct *b* so that this is violated. So far our construction of *b* does not preclude any particular assignment to $b(\{(vaf_1^X(1), vbf_1^Y(1))\})$ (in our example, $b(\{(2,1)\})$ must in fact be 1, if we are to ensure $vbg_m^Y = vaf_m^Y$, since $g(1)=2$. However, we are free to construct *b* such that $b(\{(2,1)\})$ maps to any point we wish other than 2, which has already been visited by *b*. We will impose that $b(\{(vaf_1^X(1), vbf_1^Y(1))\}) = vaf_1^X(3)$, ensuring that no function *g* can exist with the assumed properties. In the example, we have made *b* visit point 3, which cannot have fitness 2 (since we already needed to ensure that point 1 had fitness 2, and each point must have a unique fitness), and so our attempt to construct a function *g* which allowed *a* and *b* to 'swap' their performances on *f* was doomed to failure.

5. Concluding Summary

We have confirmed that the NFL holds for absolute performance metrics in the realm of multiobjective optimization. Interestingly, we have shown that NFL holds over certain salient subsets of multiobjective problems, particularly: over the space of all problems which share the same type of Pareto front (subject to a reasonable way to define 'type' of front as an enumeration of examples). This is of interest since it speaks to the notion of test-function design. For example, Deb [2] has given the community an excellent framework for test function design, which explicitly differentiates between problems on the basis of the shape and nature of their Pareto fronts. The NFL result concerning Pareto fronts essentially means that we ultimately learn nothing from the finding 'algorithm *A* outperforms algorithm *B* on a problem with Pareto front *P*'. This is because the NFL result guarantees that this will be counterbalanced by *B* outperforming *A* on some other problem or set of problems which share the same Pareto front *P*. However, inasmuch as the shape of the Pareto front (for problems in this or similar test suites) correlates with or corresponds to certain regularities of the *landscapes* in these test suites, there is certainly value in such test suites; the point is that any statements about comparative multiobjective algorithm performance should be carefully qualified. We have also shown that, in general, there are 'free leftovers' available for every algorithm, when compared with at least some other algorithm using a particular well-behaved performance metric. Much needs to be done to indicate how applicable this overall result is. For example, it is currently unknown which families of comparison metrics will yield such asymmetry and which won't, or what proportion of algorithms are thus summarily outperformed by a given algorithm. This and related work are the topic of further research.

References

1. Christensen, S. and Oppacher, F. (2001) What can we learn from No Free Lunch? A First Attempt to Characterize the Concept of a Searchable Function, in L. Spector et al (eds), *Proc. of GECCO 2001*, Morgan Kaufmann, pp. 1219–1226.
2. Deb, K. (1999) Multi-objective Genetic Algorithms: Problem Difficculties and Construction of Test Problems, *Evolutionary Computation* **7**(3): 205–230.
3. Deb, K, Agrawal, S., Pratab, A. and Meyarivan, T. (2000) *A fast elitist non-dominated sorting genetic algorithm for multiobjectie optimization: NSGA-II*, KanGAL Technical Report 200001, Indian Institute of Technology, Kanpur, India.
4. Droste, S., Jansen, T. and Wegener, I. (1999), Perhaps Not a Free Lunch But At Least a Free Appetizer, in W. Banzhaf, J. Daida, A. E. Eiben, M. H. Garzon, V. Honavar, M. Jakiela and R. E. Smith (eds.) *Proc. of GECCO 9)*, Morgan Kaufmann Publishers, Inc., pp. 833–839.
5. 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*, to appear, and citeseer.nj.nec.com/droste97optimization.html .
6. Hansen, M.P. and Jaszkiewicz, A. (1998) *Evaluating the quality of approximations to the non-dominated set*, Tech. Report IMM-REP-1998-7, Technical University of Denmark.
7. Igel, C. and Toussaint, M. (2001) On Classes of Functions for which No Free Lunch Results Hold, see http://citeseer.nj.nec.com/528857.html.
8. Knowles, J.D. (2002) *Local search and hybrid evolutionary algorithms for Pareto optimization*, PhD Thesis, Department of Computer Science, University of Reading, UK.
9. Knowles, J.D. and Corne, D.W. (2002) On metrics for comparing non-dominated sets, in *Proc. 2002 Congress on Evolutionary Comp.*, IEEE Service Center, Piscataway, NJ.
10. Radcliffe, N.J. and Surry, P.D. (1995) Fundamental Limitations on Search Algorithms: Evolutionary Computing in Perspective, in *Computer Science Today*, pp. 275–291
11. Schumacher, C., Vose, M.D. and Whitley, L.D. (2001) The No Free Lunch Theorem and Problem Description Length, in L. Spector et al (eds), Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2001), Morgan Kaufmann, pp. 565–570.
12. Van Veldhuizen (1999) *Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations*, PhD Thesis, Dept. of Electrical and Computer Eng., Graduate School of Engineering, Air Force Institute of Technology, Wright-Paterson AFB, Ohio.
13. Whitley, L.D. (1999) A Free Lunch Proof for Gray versus Binary Encodings, in Wolfgang Banzhaf and Jason Daida and Agoston E. Eiben and Max H. Garzon and Vasant Honavar and Mark Jakiela and Robert E. Smith (eds.) *Proceedings of the Genetic and Evolutionary Computation Conference, volume 1*, Morgan Kaufmann, pp. 726–733.
14. Whitley, L.D. (2000) Functions as Permutations: Implications for No Free Lunch, Walsh Analysis and Statistics, in Schoenauer, M., Deb, K., Rudolph, G., Yao, X., Lutton, E., Merelo, J.J. and Schwefel, H.-P. (eds.) *Proc. of the Sixth International Conference on Parallel Problem Solving from Nature (PPSN VI)*, Springer Verlag, Berlin, pp. 169–178.
15. Wolpert, D.H. and Macready, W.G. (1995) *No Free Lunch Theorems for Search*, Santa Fe Institute Technical Report SFI-TR-05-010, Santa Fe Institute, Santa Fe, NM.
16. Wolpert, D.H. and Macready, W.G. (1997) No Free Lunch Theorems for Optimization, *IEEE Transactions on Evolutionary Computation* **1**(1): 67–82.
17. Zitzler, E. (1999) *Evolutionary Algoirthms for Multiobjective Optimization: Methods and Applications*, PhD Thesis, Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, November.
18. Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C.M., Fonseca, V.G. (2002) *Performance assessment of multiobjective optimizers: an analysis and review*, available from the url: http://citeseer.nj.nec.com/zitzler02performance.html.