# Remarks on a recent paper on the "No Free Lunch" Theorems

Mario Köppen, David H. Wolpert, William G. Macready*

**Abstract**

This letter discusses the recent paper "Some technical remarks on the proof of the 'No Free Lunch' theorem" [KÖ0]. In that paper, some technical issues related to the formal proof of the "No Free Lunch" (NFL) theorem for search [WM95, WM97] were given. As a result of a discussion among the authors, this letter explores the issues raised in that paper more thoroughly. This includes the presentation of a simpler version of the NFL proof, in accord with a suggestion made explicitly in [KÖ0] and implicitly in [WM97]. It also includes the correction of an incorrect claim made in [KÖ0] of a limitation of the NFL theorem. Finally, some thoughts on future research directions for research into algorithm performance are given.

*M. Köppen is with the Fraunhofer IPK Berlin, David H. Wolpert is with the NASA Ames Research Center, Moffett Field, CA, and William G. Macready is with Bios Group Inc., Santa Fe, NM.

## 1 Introduction

This letter discusses a recent paper [KÖ0] concerning technical issues involved in the proof of the "No Free Lunch" (NFL) theorems [WM95, WM97]. While following the suggestion made in [KÖ0] and [WM97] for a shorter proof of a result leading to the theorem, a claim made in [KÖ0] about circular reasoning within the subsequent proof of the full theorem has to be corrected. In the next section, the shorter proof is presented, and in the following section the incorrect claim of circular reasoning is corrected.

## 2 A simpler proof

In [KÖ0] attention was drawn to the fact that [WM95] contained the outline of a proof of a result needed to establish the NFL theorem, a proof that is simpler than the one elaborated in detail in [WM95]. To flesh out this simpler proof, consider two finite sets $X$ and $Y$ and the set of all cost functions $f : X \rightarrow Y$. Let $m$ be a non-negative integer $< |X|$. Define $d_m$ as a set $\{(d^x_m(i), d^y_m(i) = f(d^x_m(i)))\}, \quad i = 1, \ldots, m$ where $d^x_m(i) \in X \; \forall \; i$ and $\forall \; i, j, \; d^x_m(i) \neq d^x_m(j)$. We are interested in ("data-driven", or "blind") deterministic[1] search algorithms $a$ which assign to every possible $d_m$ an element of $X \setminus d^x_m$:

$$d^x_{m+1}(m+1) = a[d_m] \notin \{d^x_m\}. \tag{1}$$

Define $Y(f, m, a)$ to be the sequence of $m$ $Y$ values produced by $m$ successive applications of the algorithm $a$ to $f$. Let $\delta(.,.)$ be the Kronecker delta function

[1] The analysis can be extended to also include stochastic algorithms. See [WM95, WM97]

that equals 1 if its arguments are identical, 0 otherwise. Then the following holds:

**Lemma 1.** *For any algorithm $a$ and any $d^y_m$,*

$$\sum_f \delta(d^y_m, Y(f, m, a)) = |Y|^{|X|-m}.$$

*Proof.* Consider all cost functions $f_+$ for which $\delta(d^y_m, Y(f_+, m, a))$ takes the value 1:

i) $f_+(a(\emptyset)) = d^y_m(1)$

ii) $f_+(a[d_m(1)]) = d^y_m(2)$

iii) $f_+(a[d_m(1), d_m(2)]) = d^y_m(3)$

$\ldots$

where $d_m(j) \equiv (d^x_m(j), d^y_m(j))$. So the value of $f_+$ is fixed for exactly $m$ distinct elements from $X$. For the remaining $|X|-m$ elements from $X$, the corresponding value of $f_+$ can be assigned freely. Hence, out of the $|Y|^{|X|}$ separate $f$, exactly $|Y|^{|X|-m}$ will result in a summand of 1 and all others will be 0. □

Note that from

$$\sum_f \delta(d^y_m, Y(f, m, a)) = |Y|^{|X|-m} \geq 1$$

immediately follows the universal possibility of algorithm deception: for any algorithm $a$ there is a cost function $f$ such that in its first $m$ steps $a$ produces the worst values from $Y$ in a sequence.

# 3 Correction of a wrong claim

In [KÖ0] it was claimed that the proof of the NFL theorem presented in [WM95, WM97] fails due to circular reasoning. That claim is incorrect. Here we present a proof of the NFL theorem for deterministic algorithms based on Lemma 1 that hopefully clarifies the derivation presented in [WM95, WM97].

Consider any performance measure $c(.)$, mapping sets $d^y_m$ to real numbers.

**Theorem 1.** *For any two deterministic algorithms $a$ and $b$, any value $k \in \mathcal{R}$, and any $c(.)$,*

$$\sum_f \delta(k, c(Y(f,m,a))) = \sum_f \delta(k, c(Y(f,m,b))).$$

*Proof.* Since more than one $d^y_m$ may give the same value of the performance measure $k$, for each $k$ the l.h.s. is expanded over all those possibilities:

$$\begin{aligned}
\sum_f \delta(k, c(Y(f,m,a))) &= \\
&= \sum_{f, d^y_m \in Y^m} \delta(k, c(d^y_m)) \delta(d^y_m, Y(f,m,a)) \qquad (2)\\
&= \sum_{d^y_m \in Y^m} \delta(k, c(d^y_m)) \sum_f \delta(d^y_m, Y(f,m,a)) \\
&= \sum_{d^y_m \in Y^m} \delta(k, c(d^y_m)) |Y|^{|X|-m} \text{ (by Lemma 1)} \\
&= |Y|^{|X|-m} \sum_{d^y_m \in Y^m} \delta(k, c(d^y_m)) \qquad (3)
\end{aligned}$$

The last expression does not depend on $a$. □

The claim made in [KÖ0] refers to the first line of the proof, equation (2). It was thought that the expansion was restricted to those values of $d^y_m$ that can be obtained from $f$ by applying $a$, and only those. If, for whatever reason, that set of $d_m^y$ did depend on $a$, then the expression in equation (2) would depend on $a$ as well. The concern was that this constituted circular reasoning. For proving that a sum does not depend on a variable $z$, it is generally not sufficient to show that the summand does not depend on $z$: it is also necessary to show that the set of terms making up the sum does not depend on $z$ as well. However, this argument explicitly does not apply to the proof given here, since each of the terms corresponding to an impossible $d_m^y$ simply provides an additional value of 0 to the sum. Therefore, it makes no difference whether the expansion includes all possible $d_m^y$.

## 4 On algorithm performance

To date many researchers have considered the NFL theorems from an almost nihilisitc point of view, as establishing that all attempts to claim universal applicability of any algorithm must be fruitless. Many references to this theorem were put into the introductory parts of conference papers, instead of choosing it as a research topic by itself. The discussion in [KÖ0] was an attempt to overcome this situation and to re-consider some basic issues again.

There are still many questions open, and most of them are strongly related to the way we understand computing itself. For example, despite the NFL theorems, there are *a priori* distinctions in absolute measures for algorithm performance, distinctions that are far from being completely understood. One example was given in [WM97], where a so-called "head-to-head minimax" distinction holds. Another simple case where such a head-to-head distinction holds is when the game-theoretic technique of "strategy stealing" is employed by one of a pair of search algorithms. In this scenario, for all timesteps *after the first*, algorithm $b$ samples whichever point algorithm $a$ did one step before (unless $b$'s already sampled that previous point, in which case it chooses an alternative point according to some unimportant pre-fixed process). So say we judge each of the two algorithms' performance by how many steps it takes to find a point whose value exceeds some pre-set threshold value $k$. (Formally, $m$ must equal $|X|$ for this to be a performance measure, i.e. for it to be a function of $Y(f, m, a)$.) Then for no $f$ will $b$'s performance be more than 1 worse than $a$'s. On the other hand, for some $f$'s, due to its different first $Y$ value, $b$ will far outperform algorithm $a$. This asymmetry between the algorithms holds despite the NFL theorems.

Note that this kind of superiority of $b$ over $a$ is non-transitive. (This follows from defining an infinite sequence of algorithms where each algorithm in the sequence steals the strategies of its predecessor (and therefore outperforms that predecessor), and then using the fact that the number of algorithms is finite, so that the sequence must repeat.) Little more is known even about these kinds of head-to-head minimax distinctions, never mind such distinctions in general. In addition, in the context of supervised learning, there are many other ways of contrasting algorithms without invoking *a priori* assumptions about $f$ [Wol96b] [Wol96a]. Many of them presumably have analogues in the search domain considered here.

A second class of interesting and open questions concerns the use of NFL-like frameworks to elucidate the connection between well-tuned algorithms (or poorly-tuned for that matter!) and the problems they are designed to solve. Most attempts to date have analyzed the performance of specific algorithms on a small set of problems. We believe it would be fruitful to explore the implications of NFL in detail on small problems where the space of all possible functions can easily be enumerated (see [Whi99] for a related approach).

We hope to see more research results in such directions in the near future.

## References

[KÖ0] Mario Köppen. Some technical remarks on the proof of the no free lunch theorem. In *Proceedings of the Joint Conference on Information Sciences (JCIS 2000), Atlantic City, NJ*, pages 1020–1024, 2000.

[Whi99] L.D. Whitley. A free lunch proof for gray versus binary encodings. In Wolfgang Banzhaf, Jason Daida, Agoston E. Eiben, Max H. Garzon, Vasant Hanavar, Mark Jakiela, and Robert E. Smith, editors, *GECCO-99 Proceedings of the Genetic and Evolutionary Computation Conference*, volume 1, Orlando, Florida, 1999.

[WM95] David H. Wolpert and William G. Macready. No free lunch theorems for search. Technical Report SFI-TR-95-02-010, Santa Fe Institute, February 6, 1995.

[WM97] David H. Wolpert and William G. Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82, 1997.

[Wol96a] D.H. Wolpert. The existence of a priori distinctions between learning algorithms. *Neural Computation*, 7(8):1391–1420, 1996.

[Wol96b] D.H. Wolpert. The lack of a priori distinctions between learning algorithms. *Neural Computation*, 7(8):1341–1390, 1996.

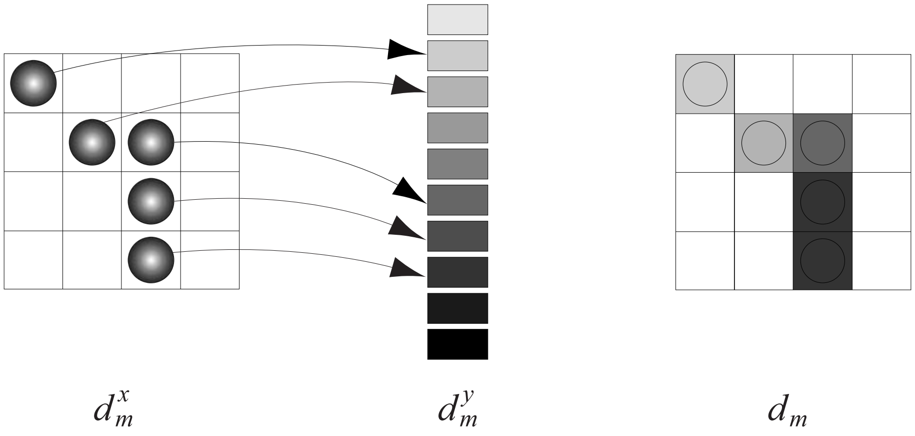

Figure 1: Definition of $d_m, d_m^x$ and $d_m^y$.

# List of Figures

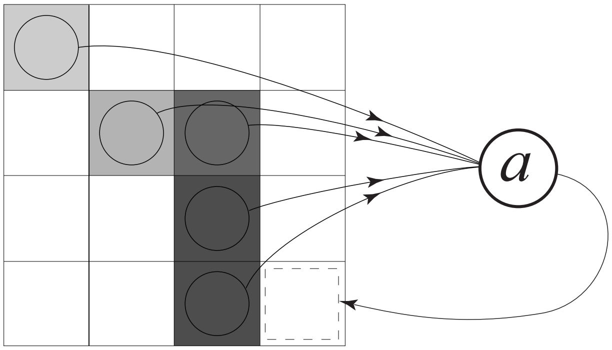

Figure 2: Definition of a data-driven algorithm $a$.