45th Southeastern Symposium on System Theory
Baylor University, Waco, TX, USA, March 11, 2013

Conservation of Information in Relative Search Performance

Winston Ewert & Robert J. Marks II
Dept. of Electrical & Computer Engineering
Baylor University
Waco, Texas

William A. Dembski
Discovery Institute
Seattle, WA

***Abstract*—While *conservation of information* popularized by the *No Free Lunch* (NFL) theorem establishes that all search algorithms have the same performance on the average, this appears not to be true when performance is compared in a relative manner. Some algorithms look to perform better than others. However, this advantage is lost when averaging is over a group of related algorithms. Every advantage against one algorithm is balanced by a disadvantage against a related algorithm. From this perspective, conservation of information still applies. As a consequence, comparative transitivity does not hold. If search procedure Z beats Y which, in turn, beats X, we cannot conclude that Z beats X. Indeed, the opposite might be true.**

***Index Terms*—conservation of information, No Free Lunch theorems, active information**

I. Introduction

Conservation of information (COI) [1], [11], [13]–[15] demonstrates the futility of attempting to solve general intermediate or larger sized problems without application of knowledge about the problem being solved. In the case of an arbitrary search problem, the *No Free Lunch* (NFL) theorem [8], [10], [13], [15], [16], for example, dictates any search algorithm can be substituted for any other algorithm without changing the search performance on average. This is true regardless of how the performance is defined as long as it only depends on the values obtained by the search process.

However, comparison of the relative performance of search algorithms do not exhibit this property [2], [9], [15]. Indeed, Wolpert and Macready [15] presented a such an example in their original paper referring to it as a minimax distinction between algorithms.

As a simple example consider a treasure buried in one of three possible locations in an desert island. Two pirates, X and Y, arrive at the island with the intent of digging up the treasure. Each pirate has to pick a strategy consisting of choosing the order of locations to be visited. If one of the pirates searches a location after his rival, he will not find the treasure. We will assume that the treasure is equally likely to be found in any of the locations.

As illustrated in Figure 1, assume Pirate X searches for the location in a specific order such as (1,2,3). Assuming uniformity [3], the probability of the treasure being found in any of those locations is $\frac{1}{3}$. Pirate Y uses a related but different search order: (2,1,3). Again, the probability of finding the treasure at any of the locations is $\frac{1}{3}$. Each strategy alone will thus have the same performance as dictated by the NFL theorem.

However, this changes if both pirates are hunting for treasure at the same time. Pirate Y has chosen locations such that in two of the three cases, he will have searched a location and taken the treasure before Pirate X. If the treasure is at location 1, Pirate X will get the treasure. However, if the treasure is location in locations 2 and 3, Pirate Y will have checked all those locations first, and thus Pirate Y will win. Pirate X will get treasure one in three times, whereas Pirate Y will claim the treasure two out of three times. In this sense, the strategy of Pirate Y is better than the strategy of Pirate X. This is true despite the two strategies performing the same when considered separately.

Due to the generality of the conservation of information results, any exception to the general principle, such as these unbalanced performance results, give pause. Does such a basic and simple variation cause conservation of information to no longer be valid? And if there is a violation of the conservation of information here, should there not be ways to exploit the failure to construct superior search algorithms?

The answer to these questions is no. The apparent "free lunch" of Pirate Y over Pirate X is not a failure of conservation of information. One strategy may beat another head-to-head, but when compared to a group of related strategies, losses and wins will balance out. Victories against one strategy are paid for by losses against another. Consequently, there exist

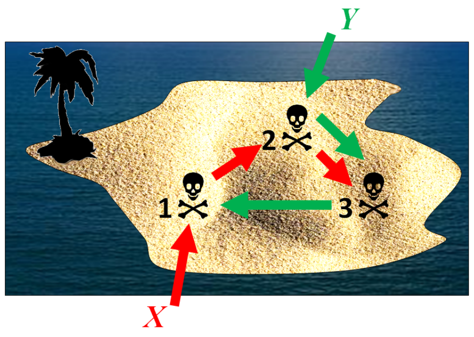

Fig. 1. Pirates X and Y search for buried treasure in parallel. The treasure can be in one of three locations.

TABLE I
INAPPLICABILITY OF TRANSITIVITY EXAMPLE.

Teams → / Treasure Location ↓	ZY	YX	XZ
		↓Winner↓	
1	Z	X	X
2	Y	Y	X
3	Z	Y	Z
Overall Winner →	Z beats Y	Y beats X	X beats Z

As shown in Figure 1, Pirate X's search order is (1,2,3) and Pirate Y's order is (2,3,1). Not shown is Pirate Z with order (3,1,2). This table shows who wins (always with probability $\frac{2}{3}$) when the pirates are paired. Transitivity does not hold.

no generally superior search algorithms.

The transitive property necessary to establish an overall superior search algorithm is inapplicable to search. Pirate Y has the advantage over Pirate X. A third Pirate Z chooses the sequence (3,1,2) which beats Pirate Y. Thus Pirate Z beats Pirate Y who beats Pirate X. If transitivity applies, Pirate Z beats Pirate X. But the opposite is true, Pirate X beats Pirate Z. Details are in Table I. This transitivity inapplicability was previously noticed by Koppen *et al.* [9].

More generally, Table II shows the probability of a strategy for finding the treasure first as compared against all other strategies. Each strategy has the same bag of performances albeit against different strategies. Every strategy has an advantage against some other strategy, but also has another strategy with an advantage over it. Thus, no way exists to gain an absolute advantage over all other strategies.

The lack of an absolute advantage limits the sense in which one strategy can be better then another. One can beat a specific algorithm, but not in a way that actually performs better against all other algorithms. Pirate A is only able to outperform his rival if he somehow knows the strategy his rival will employ. Thus, as would be expected from the idea of the conservation of information, knowledge of the rival's strategy is necessary in order to beat it.

Consider the random algorithm. This is equivalent to randomly choosing which strategy to employ. Playing against this strategy will produce the average performance of all strategies. This average is the same regardless of the opposing strategy, so no way exists to consistently best a random search algorithm.

The application of knowledge of Pirate Y about Pirate X to Pirate Y's strategy can be measured in terms of *active information*. [4]–[7], [12]. Table II shows that, with no knowledge of Pirate X's search strategy, Pirate Y has a probability of $p = \frac{1}{2}$ of finding the treasure first. When Pirate X's schedule is known, Pirate Y's chances increase to $q = \frac{2}{3}$. This knowledge gives rise to

$$I_+ = -\log_2\left(\frac{p}{q}\right) = -\log_2\left(\frac{3}{4}\right) = 0.415 \text{ bits}$$

of active information.

As an extreme example, consider the alternate scenario where Pirate X and Y decide who keeps the treasure by the majority of wins in three quick games of rock-paper-scissors. If Pirate Y knows the strategy of Pirate X, Pirate Y can win all three games. With $p = \frac{1}{2}$ and $q = 1$, Pirate Y's knowledge of Pirate X's strategy can be translated into one bit of active information.

The contest between Pirate X and Y illustrated in Figure 1 can be generalized. For any given search algorithm, we can generate a family of related search algorithms. When averaged over a distribution of functions closed under permutations as well as over the family of search algorithms, no advantage exists and the principle of conservation of information still applies.. Advantages against certain algorithms on certain fitness functions must be paid for by disadvantages against other algorithms on other fitness functions.

II. The General Result

Given a search algorithm, we can define a new search algorithm by applying a permutation to the query space. Let $\alpha(f)$ be the fitness values obtained by the search algorithm, α on the fitness function f. Let α_p, where p is a permutation on the query space, be another algorithm such that $\alpha_p(f) = \alpha(f \circ p)$. Whenever α_p makes a query, it interprets the query through the permutation thus a producing a related by distinct search algorithm. Let P be all possible one to one functions on the query space.

$$T(\alpha) = \{\alpha_p | p \in P\} \tag{1}$$

is the family of search algorithms related to α.

The NFL theorem holds for any distribution over functions which is closed under permutation [8]. That is

$$\sum_{f \in F} \Pr[f]\chi(\alpha(f)) = \sum_{f \in F} \Pr[f]\chi(\delta(f)) \tag{2}$$

where χ is some performance metric, $\alpha(f)$ is the values obtained by the search algorithm α run on the fitness function f. δ is any search algorithm. The search algorithm can be substituted for any other algorithm without changing the averaged performance. The essential reason for this is that the values obtained by a search algorithm are independent of the algorithm, and thus any function of those will be independent as well.

The relative performance of a pair of algorithms, α and β can be defined as

$$\varrho(\alpha, \beta) = \sum_{f \in F} \Pr[f]\chi(\alpha(f), \beta(f))$$

Notably, substituting different algorithms in this expression may produce a different average even when averaged over the uniform distribution.

Theorem 1. *If for any $f \in F$, $\Pr[f] = \Pr[g]$ for any $g \in \Pi(f)$ where $\Pi(f)$ is the set of permutations of f then*

$$\wp := \sum_{\gamma \in T(\beta)} \varrho(\alpha, \beta) = \sum_{\gamma \in T(\beta)} \sum_{f \in F} \Pr[f]\chi(\alpha(f), \gamma(f))$$

is independent of α and β

Proof:

$$\wp = \sum_{\gamma \in T(\beta)} \sum_{f \in F} \Pr[f]\chi(\alpha(f), \gamma(f))$$

TABLE II
PROBABILITY OF FINDING THE TREASURE FIRST FOR ALL STRATEGIES.

	(1, 2, 3)	(1, 3, 2)	(2, 1, 3)	(2, 3, 1)	(3, 1, 2)	(3, 2, 1)
(1, 2, 3)	0.00	0.25	0.25	0.50	0.25	0.25
(1, 3, 2)	0.25	0.00	0.25	0.25	0.25	0.50
(2, 1, 3)	0.25	0.50	0.00	0.25	0.25	0.25
(2, 3, 1)	0.25	0.25	0.25	0.00	0.50	0.25
(3, 1, 2)	0.50	0.25	0.25	0.25	0.00	0.25
(3, 2, 1)	0.25	0.25	0.50	0.25	0.25	0.00

From (1) and the definition of α_p, $\gamma(f) = \beta(f \circ p)$. Thus

$$\wp = \sum_{p \in P} \sum_{f \in F} \Pr[f] \chi(\alpha(f), \beta(f \circ p)).$$

Exchanging the summations give

$$\wp = \sum_{f \in F} \Pr[f] \sum_{p \in P} \chi(\alpha(f), \beta(f \circ p))$$

p is only used in the expression $f \circ p$, so we can rewrite this as

$$\wp = \sum_{f \in F} \Pr[f] \sum_{g \in \Pi(f)} \chi(\alpha(f), \beta(g))$$

The inner summation is a function of the values obtained by a search algorithm averaged over a permutation. This follows from (2). We can therefore substitute the algorithm β with another arbitrary algorithm, δ.

$$\wp = \sum_{f \in F} \Pr[f] \sum_{g \in \Pi(f)} \chi(\alpha(f), \delta(g))$$

Let Y be the set of subsets of F such that each subset contains all functions which are permutations of each other.

$$\wp = \sum_{y \in Y} \sum_{f \in y} \Pr[f] \sum_{g \in \Pi(f)} \chi(\alpha(f), \delta(g)).$$

By the assumption of the theorem, all $\Pr[f] = \Pr[g]$ for any f, g in a specific $y \in Y$. Thus we can move the probability out of the summation.

$$\wp = \sum_{y \in Y} \Pr[f] |y| \sum_{f \in y} \sum_{g \in \Pi(f)} \chi(\alpha(f), \delta(g))$$

We observe that $\Pi(f) = y$ where $f \in y$. Thus

$$\wp = \sum_{y \in Y} \Pr[f] |y| \sum_{f \in y} \sum_{g \in y} \chi(\alpha(f), \delta(g)).$$

The summation over f fits the form of (2) so we can substitute the algorithm α with another arbitrary algorithm, δ'.

$$\wp = \sum_{y \in Y} \Pr[f] |y| \sum_{f \in y} \sum_{g \in y} \chi(\delta'(f), \delta(g))$$

Thus the relative performance averaged over the set of related functions does not depend on the choice of algorithm α and β. ■

The relative performance of an algorithm when averaged over a closed-under-permutation distribution and over the family of related algorithms is independent of the choice of algorithms. Above average performance in one situation must be paired with below average performance in another situation.

III. CONCLUSION

We have shown that no generally superior algorithm can be produced even when using relative performance metrics. Better performance against some algorithms is paid for by worse performance against other algorithms. One can devise an algorithm that is relatively superior to another, but not against all other algorithms. In order to best an algorithm, active information is required from extra knowledge of the problem. This parallels the case of the NFL requiring information about the fitness function in order to improve performance. Finally, random search exhibits the same average performance regardless of which algorithm it faces. Thus no way exists to gain an advantage on average over random search.

The principle of conservation of information still applies in the case of relative performance metrics. The appearance of free lunches in relative performance metrics does not give us any way to exploit them to create generally superior optimizers.

ACKNOWLEDGEMENTS

This work was supported in part by the *Center for Evolutionary Informatics*.

REFERENCES

[1] Steffen Christensen. What can we learn from no free lunch? a first attempt to characterize the concept of a searchable function. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, 2001.

[2] David W. Corne and Joshua D. Knowles. No free lunch and free leftovers theorems for multiobjective optimisation problems. In *Evolutionary Multi-Criterion Optimization (EMO 2003) Second International Conference*, pages 327–341. Springer LNCS, 2003.

[3] W.A. Dembski and R.J. Marks II. Bernoulli's principle of insufficient reason and conservation of information in computer search. In *Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on*, pages 2647 –2652, oct. 2009.

[4] William A. Dembski and Robert J. Marks II. Life's Conservation Law: Why Darwinian Evolution Cannot Create Biologica Information. In Bruce Gordon and William A. Dembski, editors, *The Nature of Nature*, Wilmington, Del., 2009. ISI Books.

[5] William A. Dembski and Robert J. Marks II. The Search for a Search: Measuring the Information Cost of Higher Level Search. *Journal of Advanced Computational Intelligence and Intelligent Informatics*, 14(5):475–486, 2010.

[6] Winston Ewert, William A. Dembski, and Robert J. Marks II. Evolutionary synthesis of nand logic: Dissecting a digital organism. In *2009 IEEE International Conference on Systems, Man and Cybernetics*, number October, pages 3047–3053. IEEE, October 2009.

[7] Winston Ewert, George Montañez, W.A. Dembski, and Robert J. Marks II. Efficient per query information extraction from a hamming oracle. In *System Theory (SSST), 2010 42nd Southeastern Symposium on*, pages 290 –297, march 2010.

[8] Christian Igel and Marc Toussaint. A No-Free-Lunch theorem for non-uniform distributions of target functions. *Journal of Mathematical Modelling and Algorithms*, 3(4):313–322, January 2005.

[9] M. Koppen, D.H. Wolpert, and W.G. Macready. Remarks on a recent paper on the "no free lunch" theorems. *IEEE Transactions on Evolutionary Computation*, 5(3):295–296, June 2001.

[10] Tom Loredo. No Free Lunch : Challenging Problems for Frequentist & Bayesian Approaches No Free Lunch for Optimization. *Outlook*, (July), 2009.

[11] TM Mitchell. The need for biases in learning generalizations. Technical report, Department of Computer Science, Rutgers University, 1980.

[12] George Montañez, Winston Ewert, William A. Dembski, and Robert J. Marks II. A Vivisection of the ev Computer Organism: Identifying Sources of Active Information. *BIO-Complexity*, 2010(3):1–6, April 2010.

[13] D.L. Pepyne and Y.C. Ho. Simple explanation of the no free lunch theorem of optimization. In *Proceedings of the 40th IEEE Conference on Decision and Control (Cat. No.01CH37228)*, volume 5, pages 4409–4414. IEEE, 2001.

[14] Cullen Schaffer. A conservation law for generalization performance. In Cohen WW and Hirsch H, editors, *Proceedings of the Eleventh International Machine Learning Conference*, pages 259 – 265, 1994.

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

[16] John R Woodward and James R Neil. No Free Lunch, Program Induction and Combinatorial Problems. *Genetic Programming Proceedings of EuroGP2003*, 2610:475–484, 2003.