WHAT DOES DINNER COST?

David H. Wolpert

NASA Ames Research Center

http://ti.arc.nasa.gov/people/dhw/

NASA-ARC-05-097

THE COST OF MEALS

Early work on the cost of lunch: Hume, Goodman, Wolpert

Later work, not in the GA literature: Koehler, Ho, Dembski, Pepyne, Zhao, Zhu, Rohwer, Schaffer, Spears, Perakh, Forster, Cataltepe, Abu-mostafa, Magdon-ismail

Later work, in the GA literature: Macready, English, Whitley, Schmumacher, Vose, De Jong, Christensen, Oppacher, Corne, Knowles, Culberson, Droste, Jansen, Wegener, Igel, Toussaint, Jansen, Montgomery, Radcliffe, Surry, Shallit, Woodward, Neil

Work on the cost of other meals: Godel, Turing, Landauer, Moore, Wolpert, Lloyd

ROADMAP

1) Personal view on **NFL** *for search*

↓

2) ***Other domains: Bandits, self-play, coevolution***

↓

3) ***Generalized Optimization (GO) framework: Analyze the cost of lunch for all those domains.***

↓

4) NFL ***for supervised learning***

↓

5) ***The price of other meals***

NFL FOR SEARCH - DEFINITIONS

1) **Input space** ***X, and*** **Output space** ***Y.***

2) Objective Function $f : X \rightarrow Y$

3) m (distinct) sampled points of f:

$$d_m = \{d_m(1), d_m(2), \ldots, d_m(m)\}$$

where $\forall\ t$,

$$d_m(t) = \{d_m^X(t), d_m^Y(t)\}$$

4) Search algorithm $a = \{d_t \rightarrow d_m^X(t+1) : t = 0,..,m\}$

(Typically no repeats allowed.)

5) Real-valued **Cost function** $C(d_m)$

Obvious extensions to stochastic f, a.

NFL FOR SEARCH - PRIMARY RESULT

$$\sum_f P(d_m^Y \mid f, m, a) = \sum_f P(d_m^Y \mid f, m, a')$$

$$\forall \; a, a', d_m$$

So for any C(.), and any set of f's, Φ :

$$a \text{ beats } a' \text{ on all } f \in \Phi$$

$$\Rightarrow$$

$$a' \text{ beats } a \text{ on } F - \Phi$$

NFL FOR SEARCH - PRIMARY RESULT

$$\sum_f P(d_m^Y \mid f, m, a) = \sum_f P(d_m^Y \mid f, m, a')$$

$$\forall \; a, a', d_m$$

1) *Same result for many non-uniform averages over f*

2) *Same result if average over P(f)'s*

3) *NFL quantifies luck ("intelligence"):*

$C \leq \varepsilon \;\; \Rightarrow$ our luck in the match of f to a (which we chose **before we saw any data***) is at least $K(\varepsilon)$.*

Must use knowledge about f to choose a. (Saying "real-world P(f) non-uniform" doesn't justify any particular a.)

GEOMETRY OF SEARCH

$$P(d_m^Y \mid m,a) \;=\; a_{d_m^Y,m} \bullet p$$

where

$$p = P(f),\;\; a_{d_m^Y,m} = P(d_m^Y \mid m,a,f)$$

are both vectors indexed by f

1) *Similarly for E(C | m, a), etc.*
2) *Intuition: a must be aligned with P(f) - or else.*
3) *NFL theorem: All* $a_{d_m^Y,m}$ *have same projection on diagonal p*
4) *All deterministic* $a_{d_m^Y,m}$ *have same Euclidean magnitude*

AVERAGES OVER ALGORITHMS

- ***Rather than fix a and average over f, do the opposite:***

1) ***Let G and H be*** **<u>choosing procedure</u>** ***maps:***

   {[d (generated by a); d′ (generated by a′)]} $\rightarrow$ ***{a, a′ }***

2) ***Let*** $c_{>m}$ ***be the costs in a subsequent set of k samples of f.***

$$\sum_{a,a'} P(c_{>m} \mid f, m, k, a, a', G) = \sum_{a,a'} P(c_{>m} \mid f, m, k, a, a', H)$$

$\forall\ m, k, G, H,$ **<u>and any *f*</u>**

3) ***Since the sum is independent of f, all this holds for any P(f).***

AVERAGES OVER ALGORITHMS - 2

- ***Example:***

  Let G be the procedure "always choose a",

  Let H be the procedure "always choose a´ ".

- ***Then the f-independence of the sum implies:***

  Say that for each y, f_1 and f_2 have the same total number of x's such that f(x) = y. However f_1 is "well-behaved" (e.g. smooth) and f_2 is "poorly-behaved" (e.g. jagged).

  Say over a set of algorithms S, f_1 gives better performance than f_2.

  Then the opposite holds for the remaining algorithms, {a} - S

PAIRWISE DISTINCTIONS BETWEEN ALGORITHMS

1) *NFL only says first moments over f are a-independent*
2) *For higher order moments coupling the algorithms, there are* **a priori** *distinctions between algorithms.*
3) *E.g., there exist* a_1, a_2, $d^Y_{m,1}, d^Y_{m,2}$ *such that*

$$\sum_f P(d^Y_{m,1} = z, d^Y_{m,2} = z' \mid f, m, a_1, a_2) \neq \sum_f P(d^Y_{m,1} = z', d^Y_{m,2} = z \mid f, m, a_1, a_2)$$

PAIRWISE DISTINCTIONS - 2

3) However if there is no overlap between $d^X_{m,1}, d^X_{m,2}$ ***, then***

$$\sum_f P(d^Y_{m,1} = z, d^Y_{m,2} = z' \mid f, m, a_1, a_2) = \sum_f P(d^Y_{m,2} = z', d^Y_{m,1} = z \mid f, m, a_1, a_2)$$

4) On the other hand, there are C(.), a_1, a_2, δ ***where***

$$\exists \; f \text{ for which } E(C \mid f, m, a_1) - E(C \mid f, m, a_2) = \delta$$

but

$$\neg\exists \; f \text{ for which } E(C \mid f, m, a_2) - E(C \mid f, m, a_1) = \delta$$

ROADMAP

1) *Personal view on* NFL *for search*

2) *Other domains: Bandits, self-play, coevolution*

3) *Generalized Optimization (GO) framework: Analyze the cost of lunch for all those domains.*

4) NFL *for supervised learning*

5) *The price of other meals*

MULTI-ARMED BANDITS

1) *K "arms", each a real-valued stochastic process.*
2) *You know something about the arms.*

   E.g., each arm is a Gaussian, and all have the same standard deviation.

3) *You sample the arms, one at a time, m times total. You record those sample values as "rewards".*
4) *A* **strategy** *maps*

   (all arm-reward pairs by time t) → (next arm)

   for all t.

5) *What strategy maximizes summed reward at t = m?*

SELF-PLAY

1) *There is an N-player non-cooperative game whose payoff matrix Γ you don't fully know.*

2) *You repeatedly:*

   i) *Choose the moves (strategies) of all N players;*

   ii) *Have them play those moves;*

   iii) *Record the resultant payoffs.*

3) *After this, player 1 (the* <u>champion</u>*) plays a move for a new set of N - 1* <u>antagonists</u> *whom you don't control.*

4) *How best perform (2), and then use its results, to choose champion's move for that subsequent game?*

CO-EVOLUTION

1) *N-player non-cooperative game with payoff matrix Γ.*

2) *In addition to its strategy s_i, each player i is associated with a* **population size** *or* **population frequency***, u_i.*

3) *There is a fixed function T (perhaps partially determined by you), mapping*

$$\Gamma, \quad \{s_i(t), u_i(t), : i = 1, \ldots, N\}$$

$$\rightarrow$$

$$\{s_i(t+1), u_i(t+1), : i = 1, \ldots, N\}.$$

E.g., the replicator dynamics.

5) *Analyze this. E.g., what can T guarantee, for any Γ?*

ROADMAP

1. *Personal view on* **NFL** *for search*
2. *Other domains: Bandits, self-play, coevolution*
3. *Generalized Optimization (GO) framework: Analyze the cost of lunch for all those domains.*
4. **NFL** *for supervised learning*
5. *The price of other meals*

GENERALIZED OPTIMIZATION (GO) FRAMEWORK

1) *Two spaces X and Z.*

   E.g., X is inputs, Z is distributions over outputs.

2) **Fitness Function** $f : X \rightarrow Z$

3) *m (perhaps repeated) sampled points of f:*

$$d_m = \{d_m(1), d_m(2), ..., d_m(m)\}$$

where $\forall\ t$,

$$d_m(t) = \{d_m^X(t), d_m^Z(t)\}$$

each $d_m^Z(t)$ *a (perhaps stochastic) function of* $f[d_m^X(t)]$

E.g., $d_m^Z(t)$ *could be a sample of* $f[d_m^X(t)]$

E.g., $d_m^Z(t)$ *could be mean of* $f[d_m^X(t)]$

E.g., $d_m^Z(t)$ *could be* $f[d_m^X(t)]$

GO FRAMEWORK - 2

4) **Search algorithm** ***a*** = $\{d_t \rightarrow d_m^X(t+1) : t = 0,..,m\}$

5) Euclidean vector-valued **Cost function** ***C(f, d_m)***

6) To capture a particular type of optimization problem, much of the problem structure is expressed in C(., .)

NFL theorems depend crucially on having C be independent of f.

If C depends on f, free lunches may be possible.

E.g., have C independent of (f, d_m), unless $f = f^*$.

MULTI-ARMED BANDITS IN GO FRAMEWORK

1) *X is the set of arms.*
2) *Each z is a Gaussian of known (x-independent) variance, with unknown (x-varying) mean.*
3) *Each* $d_m^Z(t)$ *is a random sample of the distribution* $f[d_m^X(t)]$
4) *C is independent of f:* $C(d_m) = \sum_{t \leq m} d_m^Z(t)$
5) *The search algorithm allows repeats.*
6) *Therefore there are free lunches; even without knowledge about the means of the Gaussian (i.e., about f's), some algorithms are preferred.*

SELF-PLAY IN GO FRAMEWORK

1) *For simplicity, take N = 2.*

2) *X is joint move. For simplicity, deterministic f;*
   Z is (a delta function about the) payoff to player 1.

   (Recall we don't know payoff function, i.e., f.)

3) *We choose the search algorithm a.*

4) *We also choose a function A(.) mapping our data d_m to the champion's move for the subsequent game.*

SELF-PLAY IN GO - 2

5) *More precisely, A's image is*

   A set of all $x \in X$, with some particular value of x_1 (which will be our champion's move).

6) *For simplicity, have $C(d_m, f)$ reflect worst case behavior of the antagonist.*

7) *More precisely,*

$$C(d_m, f) = \min_{x \in A(d_m)} f(x)$$

8) *N.b., A(.) is specified in the "cost function" C.*

SELF-PLAY IN GO - 3

9) Since C depends on f, free lunches may be possible

- in fact, they exist.

10) Example:

i) 2 possible moves for opponent, many for champion.

ii) $m = 4$.

iii) In those 4 games, a selects the 4 moves $\{(1, x_2), (2, x_2)\}$.

iv) A sets x_1 to either 1 or 2, depending on which was maximin superior in the 4 observed game outcomes, d_m.

v) A' sets x_1 to whichever was maximin inferior.

$$E(C \mid f, m, A, a) \geq E(C \mid f, m, A', a) \quad \forall f; \text{ a free lunch.}$$

ROADMAP

1) Personal view on **NFL** *for search*

2) *Other domains: Bandits, self-play, coevolution*

3) *Generalized Optimization (GO) framework: Analyze the cost of lunch for all those domains.*

4) NFL *for supervised learning*

5) *The price of other meals*

NFL FOR SUPERVISED LEARNING - DEFINITIONS

1) **Input space** *X, and* **Output space** *Y.*
2) **Target Function** $f : X \rightarrow Y$
3) **Training set** *of m sampled points of f:*

$$d_m = \{d_m(1), d_m(2), ..., d_m(m)\}$$

where $\forall t$,

$$d_m(t) = \{d_m^X(t), d_m^Y(t)\}$$

4) **Learning algorithm** *for predicting outputs:* $a = (d_m,\ q \in X) \rightarrow Y$
5) *Real-valued* **Cost function** $C[f(.), a(d_m, .)]$. *(Certain formal restrictions, e.g., off-training set q.)*

Obvious extensions to stochastic f, a.

NFL FOR LEARNING - PRIMARY RESULTS

$$\sum_f P(C \mid f, m, a) = \sum_f P(C \mid f, m, a')$$

$$\forall \; a, a', d_m$$

Whether or not you use cross-validation, kernel machines, etc.

There is also an inherent geometry:

$$P(C \mid m, a) \;\; = \;\; a_{C,m} \bullet p$$

where

$$p = P(f), \;\; a_{d_m^Y, m} = P(C \mid m, a, f)$$

are both vectors indexed by f

ROADMAP

1) Personal view on **NFL** *for search*

↓

2) *Other domains: Bandits, self-play, coevolution*

↓

3) *Generalized Optimization (GO) framework: Analyze the cost of lunch for all those domains.*

↓

4) **NFL** *for supervised learning*

↓

5) *The price of other meals*

LIMITS ON MATH, SCIENCE AND BEYOND

1) ***NFL for supervised learning formalizes Hume:***

   Science cannot give guarantees about future experiments based on results of previous experiments.

2) ***Godel's theorems say math cannot give guarantees about its own conclusions.***

3) ***No matter what simulation program it runs, no computer can give guarantees about any future physical experiment.***

More generally, no system - even the universe itself - can give guarantees about prediction, control or observation.

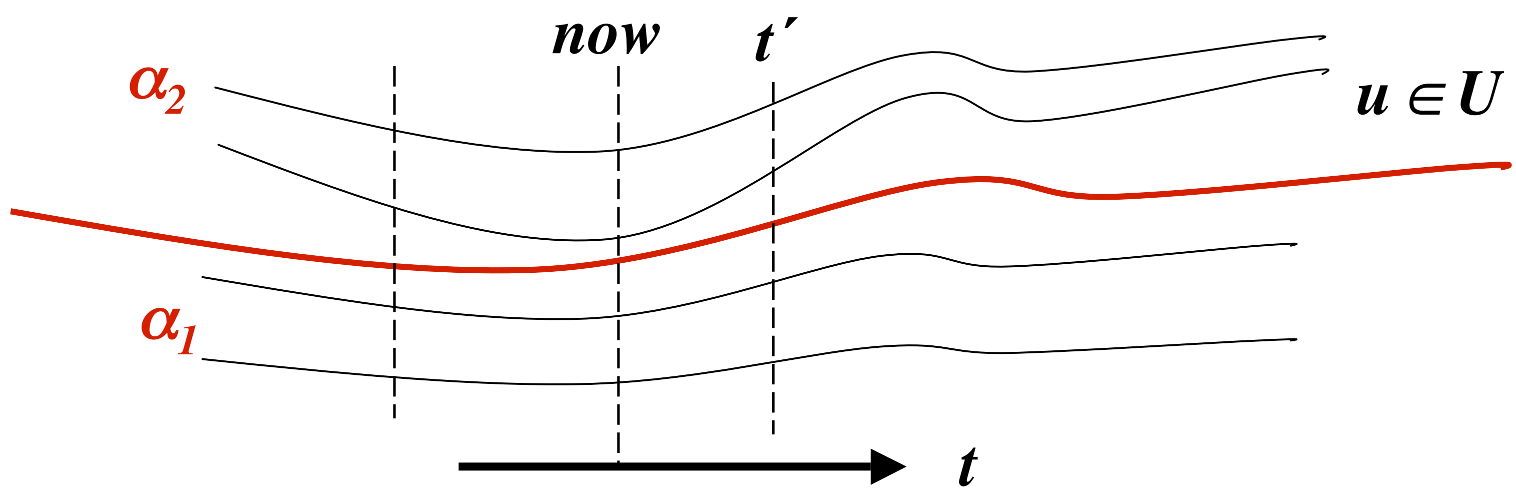
COMPUTATION AND PHYSICS

1) **Physical limitations of computational systems**
   - **Landauer's law, reversible computation, etc.**
2) **Computational limitations of physical systems**
   - **How fast / large can computation be while consistent with the fundamental laws of physics.**
3) **More profoundly, might the universe *be* a computer?**
   - **Wheeler: "It from bit"**

Difficulty: Chomsky hierarchy ill-suited to (3). What would it mean for universe to "be" a tape with a read/write head?

Solution: Formalize computation - more generally inference - as actually done in physical systems.

PREDICTION (REMEMBERING)



1) What α contains/contained the universe's worldline u at t´?

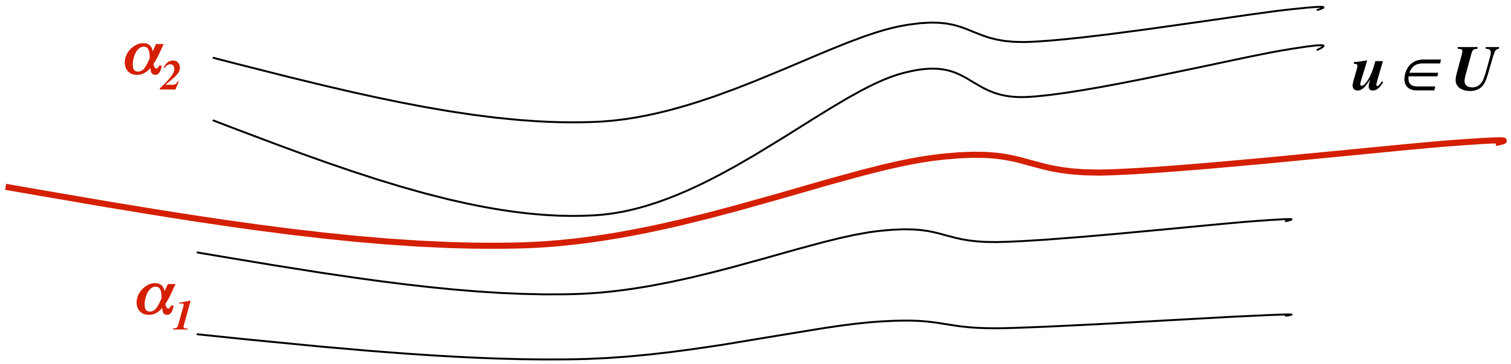
- ***The possible answers (outputs) of my computer themselves ... form a partition of U. (The computer lives in the universe.)***

2) Must tell my computer what program it should run.

- ***Those possible inputs to the computer form a partition of U.***

***Computer* = (input partition, output partition)**

INFERENCE DEVICES



1) An *input partition* X : u → x, the label of the input.

2) An *output partition* Y : u → (A, α ∈ A), the pair of a set of possible *answers,* and an element of that set.

3) An *inference device* C is such a pair (X, Y).

Observation devices, control devices, computers: all are inference devices.

IMPOSSIBILITY OF INFERENCE

- *No device can infer itself.*
- *No two distinguishable devices can infer each other*

1) *The universe may contain one device that can predict the rest of the universe - but no more than one.*

2) *If you have many distinguishable devices, at most one can infer all the others: a* God *device.*

   I.e., at most one device that can (infallibly) observe / predict / control all distinguishable others: "Monotheism".

3) *A time-translated copy of a God device cannot be a God device.*

   I.e., God can only be infallible once: "Intelligent design".

ENGINEERING IMPLICATIONS OF IMPOSSIBILITY RESULT

1) ***For any device simulating physical systems, there is always a prediction by it that cannot be guaranteed correct.***

   (Even if just simulating external universe, if the simulator isn't a God device, always a prediction by it that can't be guaranteed.)

   • Laplace was wrong.

2) ***For any recording apparatus, there is always a past event that cannot be guaranteed to have been correctly recorded.***

3) ***For any observation apparatus, there is always an observation by it that cannot be guaranteed to be correct.***

   • Non-quantum mechanical "uncertainty principle"

CONCLUSIONS

1) ***Much still to be investigated about search:***
   - ***i) P(f)-independent results (e.g., algorithm averages).***
   - ***ii) The geometry of search***
   - ***iii)* A priori *distinctions between search algorithms - higher order correlations.***

2) ***Much still to be investigated about supervised learning:***
   - ***i) Relation between NFL and statistical learning theory***
   - ***ii)* A priori *distinctions between learning algorithms - cross-validation vs. anti-cross-validation?***

3) ***Much still to be investigated about inference devices:***
   - ***i) Analogs of algorithmic information complexity***
   - ***ii) Graphical relations between inference devices.***