# Computational Darwinism, or Who Teaches the Teacher?*

Barry McMullin

September 1989

## School of Electronic Engineering

TECHNICAL REPORT: bmcm8901

©1989 Barry McMullin,
School Of Electronic Engineering,
Dublin City University,
Dublin 9,
IRELAND

**Telephone:** +353-1-704 5432
**Fax:** +353-1-704 5508
**E-mail:** McMullin@EENG.DCU.IE

* *This is a re-formatted version of a paper presented at the conference* AI and Cognitive Science '89*, Dublin, 1989.*

## Abstract

The “homunculus” problem is discussed, in the context of machine learning. Arising from this, a class of adaptive computational systems (“D-machines”), based on darwinian principles, is defined. A D-machine consists of a computing substrate (“interpreter”) supporting a dynamic population of diverse computational structures referred to as *demons*. Demons interact with each other and with an “external” environment. Interactions include the possibility of reproduction (genetic or otherwise). Demons must compete for finite resources.

It is conjectured that a D-machine, *as a whole*, may “adapt” to its environment (i.e. “learn”) in novel and interesting ways.

D-machines and related prior systems are compared and contrasted. A research program, involving both theoretical and empirical investigation, is outlined.

## 1 Introduction

AI has had some considerable successes—as an Engineering discipline. That is, there are a number of AI tools and systems which are in widespread use, solving practical problems in the "real" world. However, AI has been much less successful in elucidating the essential nature of intelligence, and there is a significant, and growing, scepticism as to whether, in its conventional form, it ever will (e.g. [Searle, 80], [Hofstadter, 83]). The (re-)birth of "connectionism" can be seen as a manifestation of this ([Rumelhart et al, 86]); however there is considerable debate as to whether, or what, connectionism has to offer which is *genuinely* new or distinctive (e.g. [Fodor & Pylyshyn, 88]).

In this paper I will focus on that philosophical criticism of AI usually known as the "homunculus" problem. The proposed resolution (the D-machine) is neither a conventional AI system, nor a connectionist network; rather, it is a distinctive architecture, deriving elements from both of these paradigms, but also from a number of other, separate, prior developments.

## 2 Who Teaches the Teacher?

The homunculus problem in AI is this: how can symbols in an AI system acquire "genuine semantics", unless we postulate some "inner" entity (the homunculus) who interprets them; but then, of course, it would be the homunculus who is intelligent, not the AI system. The naïve response is to replace the homunculus with another AI (sub)system—but then we have to ask how *its* symbols acquire meaning. The homunculus has been displaced, not removed, and an infinite regress opens up. In practise, it is suggested that AI symbols have meaning *only* in the eye of the (human) beholder (usually the programmer), who is therefore playing the role of the "ultimate" homunculus.

One might seek a resolution to this problem by studying artificial "learning": for surely if a system *constructs* its own symbols then *it* must be giving meaning to them, rather than some original programmer or non-existent homunculus.

Consider then, the conventional, canonical, model of a learning machine, shown in Figure 1. It comprises a "performance" element, and an adaptive or "teaching" element. The performance element is charged with implementing the task or skill which is to be learned; the teaching element is to modify (teach) the performance element to achieve this. The teaching element typically does this by comparing the behaviour of the learning element with some target behaviour, and making modifications which should reduce observed discrepancies (i.e. it is fundamentally a negative feedback controller). The model is conceptually straightforward, although implementing it effectively in real task environments is, of course, far from easy.

Clearly, if the system works, then, at the end, there will exist symbols, in the performance element, which designate objects in the task environment, and which have been spontaneously created by the learning machine—surely these "genuinely" designate, without appeal to any external agency? Well no: their meaning derives from the interpretation placed on them by the entity which created them—the teaching element; but *its* ability to do such interpretation derives from the meaning (or lack of it) of its own symbols, which were not self constructed, but supplied by its programmer(s). We could, of course, introduce another layer of adaptation or teaching, but that would only displace the problem, not solve it. The original homunculus problem has thus been recast as "Who teaches the teacher?".

This problem is not new, and a proposed resolution, in principle, is well known. It is presented, in slightly different forms, in [Selfridge, 59], [Hofstadter, 79], and [Minsky, 87]. It is this: instead of an hierarchical architecture, implement a *tangled* hierarchy. Make the system, directly or indirectly, self-referencing or reflexive. In effect, we require meaningful symbols to *bootstrap* themselves into existence. That is, every element modifies another, giving meaning to the other's symbols, while being also modified by some other, thus acquiring meaning for its own symbols. In this way, The hierarchical learning machine of Figure 1, turns into the reflexive system of Figure 2; but now the distinction between "teaching" and "performance" elements has disappeared, and any (static) decomposition of the system into subsystems is contradictory—we are really left with the unitary, self-modifying, architecture, of Figure 3. I will refer to this model for intelligent systems as the *Reflexive Hypothesis*.

We have now exchanged an abstract philosophical problem for a (mere?) engineering one: how to actually design and build such reflexive systems. More carefully: it is easy to design a system which is reflexive—the problem is that it will tend to immediately self-destruct. This phenomenon is familiar to all who have had programs "accidentally" treat their own instructions as data, and overwrite themselves—a "crash" is the inevitable result. Thus we need to identify what properties or constraints a

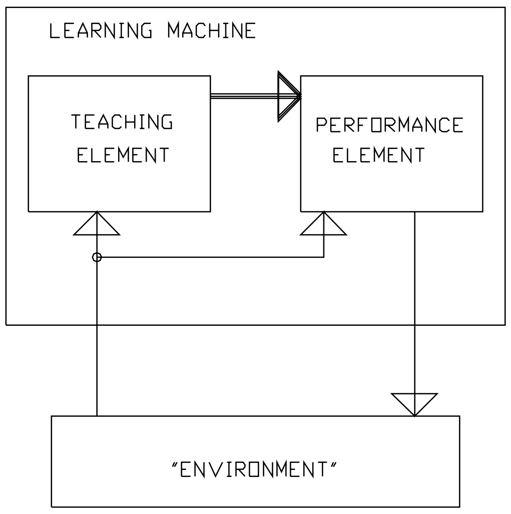

Figure 1: Canonical Learning Machine.

reflexive system should have so that it will spontaneously evolve toward greater internal organisation, and correspondingly sophisticated external behaviour. In short, a system which, even if not initially intelligent, can *become* intelligent.

## 2.1 D-machines

In response to this, I propose a somewhat new (and, as yet, entirely hypothetical) class of adaptive computational systems, referred to as *D-machines* ("D" for "Darwinian"). It should be emphasised that "D-machine" is intended as a collective term—it does not refer to any single specific system.

The proposal is an elaboration of previous work in DCU (then NIHED) into the application of darwinian adaptation to machine learning ([McMullin, 88]).

Briefly, a D-machine is a form of information processing ecology. It consists of an underlying processing substrate, which supports and constrains a large collection of diverse, interacting, computational structures, or processes. These will be referred to as "demons" (following [Selfridge, 59]).

The key notion of the D-machine is to construct a substrate which can support arbitrary (i.e. computationally complete) demons, including (but not limited to) *self reproducing* demons, and to "seed" this substrate with some initial variety of demon "species". The demons are like (benign) "computer viruses", deliberately cultured. Thereafter, in the face of competition (for finite resources), darwinian adaptation *may* produce arbitrarily more elaborate and organised demons, and, indeed, *systems thereof.* Of course, at any time, new "handcrafted" demons may be introduced into the system—they will then survive or not in accordance with their abilities relative to the existing population.

If interface mechanisms are provided in the substrate, such that demons can interact with the external environment, then the darwinian competition between demons can be biased, in some sense, by their contribution to external behaviour. In this way it is anticipated that the adaptation of the internal demon ecology *may* give rise to "external" machine learning.

It should be emphasised that, although the underlying mechanism is described as "darwinian", this does not imply adaptation over evolutionary timescales. A key requirement of the project is to formulate effective darwinian mechanisms which operate on timescales of the order of, say, months or years, rather than millenia (cf. [Holland, 75]).

The conjectural nature of this position should be reiterated: I *hope* that darwinian mechanisms may cause useful adaptation of computational systems,

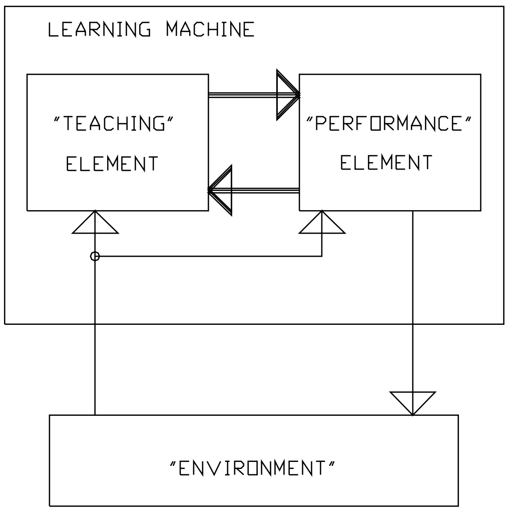

Figure 2: Reflexive Learning Machine.

but I cannot claim they *necessarily* will. The purpose of the research program is to investigate precisely this - by trying to build D-machine(s), experimenting with their behaviour, and trying to formulate effective mathematical models to support this empirical work.

## 3 Prior Work

This section reviews a variety of published research which, in one way or another, has influenced the D-machine concept. This is a concise summary—for details, see [McMullin, 89].

### 3.1 Pandemonium

Pandemonium ([Selfridge, 59]) was a seminal innovation by Oliver Selfridge in the area of adaptive computational systems. The primary contribution of Pandemonium to the present work is its early advocacy of a population of competing, concurrent, processes as a model for an adaptive computational system.

Selfridge proposed a hierarchy of adaptive mechanisms for Pandemonium—variation of "weights" of existing demons, generation of new demons by combinations of "good" precursor demons, and, possibly, some mechanism for modifying the way(s) that new demons are generated. Critically, Selfridge also saw the problem implicit in this kind of hierarchical architecture—that it admits of an infinite regress. He comments as follows:

> "Furthermore ... some of the demons, presumably, will be in a position to change themselves, for otherwise we should need another level of possible change, *and so on*."
>
> (Emphasis added)

This is the *reflexive hypothesis* as already discussed. Paradoxically, however, having made this clear statement of the reflexive hypothesis, Selfridge himself seems to back off from it immediately, by going on to suggest another, distinct, layer of hierarchical adaptation. Specifically, he suggests constructing a "crowd" of Pandemonia (?) which would then be subject to some form of natural selection *en masse*. These issues were, unfortunately, not explored further in this original and thought-provoking paper. In particular, Pandemonium, as actually implemented, was not, in fact, reflexive.

### 3.2 Genetic Algorithms

[Holland, 75] introduces an abstract mathematical framework in which to discuss adaptive systems of

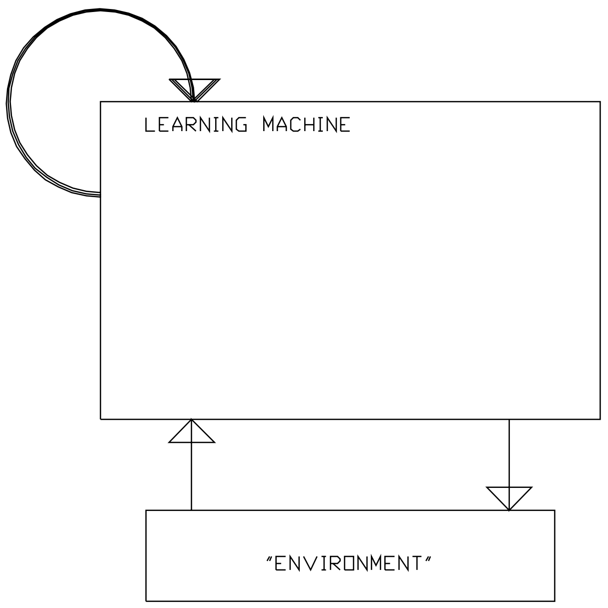

Figure 3: Unitary Reflexive Learning Machine.

all kinds. Within this framework, the notion of a generalised "Genetic Algorithm" (GA) is introduced, and it is demonstrated how, through "implicit parallelism" a GA can produce adaptive improvement in behaviour many orders of magnitude faster than could be expected based on naïve mechanisms such as exhaustive search or random mutation.

GAs provide a very well characterised, simple, and powerful, model for "darwinian" adaptation. This is based on a form of genetic reproduction which is deliberately constrained to be in proportion to relative fitness, coupled with competition for finite "living" resources (essentially, slots in a finite population).

The state of the art in GAs is comprehensively reviewed in [Schaffer & Grefenstette, 88]. The crucial point made explicit there is as follows:

> "... This evolutionary approach to machine adaptation had been tried before Holland ... but with mediocre results. The genetic operator most often used in these earlier experiments was mutation, which later analysis has shown is only capable of providing a random walk through the space of possibilities..."

Thus, Holland rehabilitated the notion of artificial adaptation based on an evolutionary metaphor, after it had previously been discredited, and largely abandoned. The D-machine proposal relies on this body of GA theory to provide a reasoned justification for thinking that, in principle at least, a machine of this sort can exhibit useful adaptation on a realistic time scale.

However, it is *not* proposed to include a GA, in the conventional sense defined by Holland, in a D-machine. GAs rely on the notion of a (programmer-defined) "fitness" function. Such a function is, however, incompatible with the reflexive hypothesis. Instead, while we provide competitive *mechanisms*, we do not constrain the ways they will be used or combined—i.e. the specific competition(s) which will emerge.

## 3.3 The Broadcast Language

In [Holland, 75], the homunculus problem is not ignored, though it emerges in a slightly different guise. Holland identifies the problem as being that of "adapting the representation". That is, he presents GA as an effective method of adaptation within the context of some particular representation of the problem—but which is incapable of compensating for any deficiencies of that representation. A naïve answer would be to use a GA to adapt the representation—but immediately we are led to consider adapting the representation of the representation etc. Again we face the problem of an infinite regress, and again, Holland, as with Selfridge, suggests the reflexive hypothesis. That is, he suggests a mechanism whereby the GA can be implemented by structures that are, themselves, accessible to the GA. He also seems to follow Selfridge in failing to pursue this hypothesis to a satisfactory conclusion.

Holland presents his concept, in fair detail, under the title of the "Broadcast Language". A computationally complete, reflexive, programming formalism is presented and Holland shows how programs (strings) in this formalism can be used to implement the basic functions required of a GA—in particular, reproduction. However, Holland does *not* follow this line through to a full blown, reflexive, GA based, system. Instead the concept was transmuted into what became known as Classifier Systems. Classifier Systems are separately discussed below. For the moment, it is sufficient to note that Classifier Systems do *not* retain the reflexive character of the Broadcast Language model.

In a sense then, the D-machine might be viewed as simply a return to Holland's original Broadcast Language idea. However, it is necessary to go beyond this. Holland implies that reflexivity of the language, alone, permits a reflexive GA to be implemented; whereas I claim that the payoff function requirement (of GAs) makes it impossible to actually implement such a reflexive GA.

It is important to isolate the problem here. It is not that the Broadcast Language is not reflexive; nor is it that a GA cannot be implemented within the Broadcast System; nor even that the GA cannot be so constructed that it will (attempt to) apply to itself. The problem is that for any of this to make any sense (even initially) some all embracing mechanism for determining the fitness of the (parts of the) GA itself must be provided—and this seems to be an intractable problem.

The resolution proposed in the D-machine is that there will be no fitness function. There will be no set of strings "implementing" the Genetic Algorithm—on themselves or anything else. Instead, we propose to introduce strings (demons) which can (self-)reproduce and compete for finite resources. Reproduction rates will be determined by open competition for resources—not by some hypothetical fitness function.

This is perhaps a reasonable point at which to emphasise that the D-machine is not being presented as some form of panacea. It is not suggested that previous approaches are, in some absolute sense, "inferior" to D-machines. Rather, it is the thesis of this paper that the D-machine is a natural progression from certain other systems, but one that has not previously been specifically explored. It may be better, it may be worse, it may be simply irrelevant. For the moment, the only strong claim being made is that it merits investigation.

## 3.4 $\alpha$-Universes

[Holland, 76] attempts to estimate how long it would take for self reproducing systems to emerge in an initially unorganised "chemical soup". Holland's contention is that, even before self-reproducing structures emerge, the evolution of such a system can exhibit implicit parallelism, in much the same sense described in [Holland, 75]. As a result, self-reproducing systems can emerge much more quickly than might otherwise be expected.

Of course, any realistic model of the original primeval soup of planet Earth would be far too complex to admit a closed form analysis. Instead, Holland formulates a family of much simpler model "universes"—which he dubs the "$\alpha$-Universes"—which are used to develop a proof of the *principle* that implicit parallelism can greatly speed the emergence of self reproducing systems.

A detailed description of the $\alpha$-Universes, and the theorems that can be proven for them, will not be presented here. For the present purposes it is sufficient just to note certain of the key characteristics. Primary among these is that there is no Genetic Algorithm being applied, and no fitness function being computed. Rather, there is an unstructured competition for certain finite resources, mediated by low level stochastic "operators" roughly analogous to diffusion and random activation in real, chemical, systems. "Molecules" which are "fit" in the sense of being more stable, or able to persist for longer in the face of these disturbing effects, achieve higher densities and have a greater effect on the subsequent evolution or development of the system—but this *emerges* rather than being explicitly "programmed in". It *is* assumed that certain molecules (should they emerge) have "special" properties that make them roughly analogous to such substances as catalysts, enzymes, and antibodies. The effect of these so-called "emergent operators" is to produce, or encourage the production of, molecular fragments which would be quite rare under the action of the primitive operators alone.

The significance of the $\alpha$-Universes is that they suggest that implicit parallelism is a more general concept than Genetic Algorithms, and that it can exist and be effective even where there is no fitness function, and no extant reproductive process.

While $\alpha$-Universes exemplify certain defining characteristics of the D-machines, they are too idiosyncratic to serve as direct "D-machine prototypes". The $\alpha$-Universes are at once both simpler and more complex than D-machines. They are simpler in that the set of defined operators is restricted to the minimum useful for demonstrating the emergence of self-reproductive systems; in particular, $\alpha$-Universe molecules are *not* computationally complete. This is not a defect of the $\alpha$-Universes—they were not intended as "computational" systems. Similarly, there is no concept of an environment "external" to an $\alpha$-Universe, with which it interacts, as this is irrelevant to its purposes—whereas such an external environment is a key aspect of the D-machine concept. On the other hand, $\alpha$-Universes are more complex than D-machines in that the operators were intended as analogs of practical chemical operators and, as such, may be more sophisticated than are actually required in a D-machine. In particular, all the $\alpha$-Universe operators are defined to be "conservative" in terms of the "elements" of the universe: it is not clear that this is a necessary or even desirable constraint on related computational systems (such as D-machines) in general.

Finally, before leaving [Holland, 76], we should note that Holland abstracts a set of properties which, in his words, "we would expect to find in most interesting models of evolving universes". Holland dubs these the "Omega" properties. The $\alpha$-Universes do, of course, possess these properties, and we would expect that any D-machine will also have to have them. Thus, they are likely to form the foundation for any theoretical treatment of the D-machine.

## 3.5 Classifier Systems

Classifier Systems were originally introduced in [Holland & Reitman, 78], specifically as a model for "cognitive systems". Holland has described Classifier Systems as a successor to the Broadcast Systems ([Holland, 87]). Classifiers (the components of Classifier Systems) are clearly related to the strings of the Broadcast System—the fundamental operation of both is to do a form of pattern matching and recoding, with interactions based on "messages" exchanged through a globally accessible message list or string environment. Classifiers however are a much simplified version of strings in the Broadcast Language—they are fixed length with a more restricted syntax. In contrast to broadcast strings, individual classifiers are *not* computationally complete, though arbitrary sets of them are.

Classifiers can be viewed as very simple condition/action rules, or productions, and the "basic" Classifier Systems can be viewed as a form of production system. This basic system may be thought of as the "performance" element of a learning machine, in the terms previously discussed.

The first level of adaptation comes by way of varying "weights" or "strengths" attached to the classifiers. This affects which classifiers are actually activated when too many are simultaneously matched, and thus affects the overall behaviour of the system. This is very roughly analogous to the basic adaptive mechanism in Pandemonium also. The mechanism used to modify the weights is called the "Bucket Brigade Algorithm" (BBA).

The problem with the BBA method of adaptation is that it is limited to the context of the Classifiers which are already in existence. If appropriate Classifiers are lacking then BBA can never correct for that. Therefore, BBA is normally complemented with a higher level adaptive mechanism which generates "plausible" new Classifiers. In the early implementations of Classifier Systems this was achieved by a GA operating on the population of Classifiers, with fitness equated to the weight accumulated under BBA. Experience with this arrangement has been mixed and, more recently, there has been increased interest in alternative mechanisms for generating new Classifiers, and, in particular, encouraging the development of useful sequences of coupled Classifiers. The difficulty with the usage of GA in this context is that the GA effectively tries to optimise individual Classifiers, implicitly assuming that fitness is well defined and independent of the other Classifiers present. In practise fitness is not well defined, because it *does* critically depend on the other classifiers present, so a GA simply applying to the population of Classifiers has severe limitations.

Holland has identified the question of generating new Classifiers (or rules) as a prime area for further investigation ([Holland, 87]). It is worth quoting his comments:

> "In a precursor of classifier systems, the broadcast language ([Holland, 75]), provision was made for the generation of rules by other rules. With minor changes to the definition of classifier systems, this possibility can be reintroduced ... With this provision the system can invent its own candidate operators and rules of inference. *Survival of these meta- (operator-like) rules should then be made to depend*

*on the net usefulness of the rules they generate . . .*"

(Emphasis added)

Superficially this is again the reflexive hypothesis: to avoid an infinite regress in the hierarchy of adaptive systems, make the adaptive mechanisms reflexive. On closer inspection this proposal falls short of a truly reflexive system in precisely the same way as its named precursor (the Broadcast Language): the requirement for a well defined fitness function. Holland addresses this in the highlighted phrase—but this suggestion plainly only advances a single level up a hierarchy, and implicitly requires a demarcation between "rules" and "meta-rules" which confounds the purpose of making the system reflexive in the first place. Thus it should be clear that Classifier Systems, even in this hypothetical reflexive form, are not at all the same class of objects as D-machines.

### 3.6 Active Symbols

Douglas Hofstadter has written extensively about his view of the nature of intelligence, and, implicitly, the need for a reflexive architecture. Briefly, Hofstadter's view is summarised in this quotation from [Hofstadter, 83]:

> "It is my belief that until AI has been stood on its head and is 100 per cent bottom-up, it won't achieve the same level or type of intelligence as humans have. To be sure, when that type of architecture exists, there will still be high-level, global, cognitive events—but they will be epiphenomenal, like those in a brain. They will not, in themselves, be computational. Rather, they will be constituted out of, and driven by, many many smaller computational events rather than the reverse. In other words, subcognition at the bottom will drive cognition at the top. And, perhaps most importantly, the activities that take place at that cognitive top level will neither have been written nor anticipated by any programmer."

Hofstadter's term for emergent structures with effective representational power is "Active Symbols". He conceives of these Active Symbols as arising from the collective statistical properties of lower level components which, in themselves, have no representational power, and do not therefore constitute symbols. He emphasises the word *Active* in contrast to the conventional notion of a symbol in AI: Active Symbols are not "formal tokens" manipulated by some program—they, themselves, are capable of active interactions with each other and (indirectly) with the external environment.

In this view, the D-machine is simply a particular proposal for a substrate which might support the kinds of emergent phenomena which Hofstadter identifies. It should be stressed that the demons are not identical with Active Symbols on this interpretation. It will only be if, or when, particular demons (or systems of demons) evolve which have representational power, in Hofstadter's terms, that it will be possible to say that Active Symbols have appeared.

## 4 Outline of Proposed Research

The D-machine is not a unitary object; we cannot simply build it and see what happens. Rather, it is a concept or a framework within which any of an infinite variety of examples might be chosen for investigation. Thus we must invest our resources with circumspection, and in the context of some overall plan.

First note that the definition of the D-machine concept has been left distinctly (and deliberately) vague. Through much of the review of section 4, I emphasised what D-machine are *not*—without ever being very explicit about what they actually *are*. The crux of this definitional uncertainty lies in the core aspect of the D-machines—the competition between demons.

The problem here is not the usual problem of a synthetic exercise—that we have too many constraints to satisfy. Quite the opposite: it is not clear what constraints exist at all, and so we hardly know where to start, or how to weigh the possibilities, or even what the space of possibilities is. This is probably the most fundamental issue to be addressed: that is, we must translate the current, informal, notion of D-machine into something much more formal, and circumscribed, and which provides some guidance in the construction of such devices (even if it is only heuristic). Unfortunately, while this is an excellent objective, it is, almost by definition, intractable. It is certainly not amenable to a direct or focused attack at this stage. For the moment then, we simply relegate it to the status of aspiration, and hope that progress will be made as a natural side-effect of the concrete work we *can* suggest.

Given the open nature of the topic the proposal is, in the first instance, to rely as much as possible on previous work which addresses similar issues. It is also proposed to approach the D-machine incrementally, through lesser objects which are plainly *not* D-machines in themselves, rather than in one fell swoop—hopefully this will allow some useful intuitions to emerge along the way.

How can we back off from a full D-machine, but still make progress toward it? I envisage two worthwhile stepping stones. The initial backward step is to eliminate the interaction with an external environment. The organisation of this interaction is clearly a deep problem which will profoundly affect whether useful adaptation to the environment occurs; but unless we *already* have a system which, in the absence of environmental interaction, exhibits spontaneous evolution toward more complex and sophisticated structures, then the addition of such interaction seems unlikely to have a beneficial effect. Thus, it makes sense to examine this "isolated" D-machine first.

The second backward step is to stop short of a computationally complete substrate (in the sense of individual demons and/or demon systems). Computational completeness is defined into D-machines to, in some sense, guarantee their generality as adaptive computational systems. However, we can try to establish darwinian competition, and evolution, as such, on simpler structures first: once this is demonstrated, full computational completeness can be added.

## 5 Conclusion

The ideas presented in this paper are not static: they have evolved even as the paper itself took shape. The result is not entirely a coherent whole, and certain points are perhaps not presented with as much clarity as one would like. Nonetheless, I hope that the central core of the D-machine concept, its genesis, and its proposed development, have been communicated satisfactorily.

In conclusion then, I simply reiterate that this is a tentative proposal, which would surely benefit from criticism, correction, refinement, and further comparison with other contemporary work in AI; I invite, and welcome, all such commentary.

## Acknowledgments

I should like to express my thanks to DCU, particularly in the person of Dr. Charles McCorkell, for generous support of this research; to John Kelly of UCD, and Noel Murphy of DCU, for many stimulating discussions; and to Professor John Holland of the University of Michigan for his helpful correspondence and comments. This does not, of course, imply that any of these people necessarily endorse the ideas presented here.

## References

**[Fodor & Pylyshyn, 88]**
Fodor, J.A., Pylyshyn, Z.W., *Connectionism and Cognitive Architecture: A Critical Analysis,* Cognition, **28** (1988) pp. 3–71.

**[Grefenstette, 87]**
Grefenstette, J.J. (Ed.), *Genetic Algorithms and their Applications: Proceedings of the Second International Conference on GAs*, Hillsdale, NJ: Lawrence Erlbaum Associates, 1987.

**[Hofstadter, 79]**
Hofstadter, D.R., *Gödel, Escher, Bach: An Eternal Golden Braid*, New York: Basic Books, 1979.

**[Hofstadter, 83]**
Hofstadter, D.R., *Artificial Intelligence: Subcognition as Computation.* In: [Machlup & Mansfield, 83], pp. 263–285.

**[Holland & Reitman, 78]**
Holland, J.H., Reitman, J.S., *Cognitive Systems Based on Adaptive Algorithms.* In: [Waterman & Hayes Roth, 78], pp. 313–329.

**[Holland, 75]**
Holland, J.H. *Adaptation in Natural and Artificial Systems*, Ann Arbor: The University of Michigan Press, 1975.

**[Holland, 76]**
Holland, J.H. *Studies of the spontaneous emergence of self-replicating systems using cellular automata and formal grammars.* In: [Lindenmayer & Rozenberg, 76], pp. 385–404.

**[Holland, 87]**
Holland, J.H., *Genetic Algorithms and Classifier Systems: Foundations and Future Directions.* In: [Grefenstette, 87], pp. 82–89.

**[Lindenmayer & Rozenberg, 76]**
Lindenmayer, A., Rozenberg, G. (Eds.), *Automata, Languages, Development*, New York: North-Holland Publishing Company, 1976.

**[Machlup & Mansfield, 83]**
Machlup, F., Mansfield, U. (Eds.), *The Study of*

*Information: Interdisciplinary Messages*, New York: Wiley Interscience, 1983.

**[McMullin, 88]**
McMullin, B., *Darwinism Applied to Machine Learning*, Technical Report Number `NIHED/EE/88-11`, School of Electronic Engineering, NIHED, Dublin. 1988.

**[McMullin, 89]**
McMullin, B. *Computational Darwinism: A Research Proposal*, Technical Report Number `NIHED/EE/89-11`, School of Electronic Engineering, NIHED, Dublin. 1989.

**[Minsky, 87]**
Minsky, M., *The Society of Mind*, London: Heineman, 1987.

**[NPL, 59]**
(National Physical Laboratory), *Mechanisation of Thought Processes*, (Proceedings of a symposium held at the National Physical Laboratory, 24–27th. November, 1958), London: HMSO, 1959.

**[Rumelhart et al, 86]**
Rumelhart, D.E., McClelland, J.L., (and the PDP Research Group), *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, Cambridge, MA: MIT Press, 1986.

**[Schaffer & Grefenstette, 88]**
Schaffer, J.D., Greffenstette, J.J., *A Critical Review of Genetic Algorithms*, (Forthcoming in "Critical Reviews in AI", Cleveland: CRC Press, 1988.)

**[Searle, 80]**
Searle, J., *Minds, Brains, and Programs*, The Behavioural and Brain Sciences, **3** (September, 1980), pp. 417–57.

**[Selfridge, 59]**
Selfridge, O.G., *Pandemonium: A Paradigm for Learning.* In: [NPL, 59], pp. 511–531.

**[Waterman & Hayes Roth, 78]**
Waterman, D.A., Hayes Roth, F. (Eds.), *Pattern Directed Inference Systems*, New York: Academic Press, 1978.