Exploring the Evolution of Complexity in Signaling Networks

John H. Holland

SFI WORKING PAPER: 2001-10-062

SANTA FE INSTITUTE

Exploring the Evolution of Complexity in Signaling Networks

John H. Holland

Abstract

Signaling networks are exemplified by systems as diverse as biological cells, economic markets, and the Web. After a discussion of some general characteristics of signaling networks, this paper explores the adaptive evolution of complexity in a simple model of a signaling network. The paper closes with a discussion of broader questions concerning the evolution signaling networks.

You are immersed in a world that has two resources distributed in patches, call them "shelter" and "food", and your needs are determined by the levels of two internal reservoirs, one for each resource. The reservoirs are depleted at a constant rate to keep your system running, so that a low reservoir level implies high need for that resource. At each instant, you can execute one of six simple actions: rest, approach, flee, turn right 45 degrees, turn left 45 degrees, consume. You can increase the level of a reservoir only by giving an appropriate response when the resource is present: "rest" when "shelter" is present, "consume" when "food" is present. Your information about the world is supplied by a "vision cone" that indicates resource locations relative to a line of sight: resource at present location, or straight ahead, or within an angle of 45 degrees to the left or right. Question: Is there a simple adaptive algorithm that can, on the basis of experience, discover action sequences (e.g. "turn until food is visible", "approach", "consume") that exploit opportunities for filling the reservoirs?

Though the format seems one of animal cognition, the question applies equally to other systems requiring coordinated responses, such as ecosystems, and the Web, and signaling networks in biological cells. In this broader context, we come to more difficult questions: Are there "general purpose" adaptive algorithms that can discover good action sequences over a wide variety of environments, without prior tuning for each environment? If so, what kinds of ontogeny and phylogeny should we expect to see as the adaptive algorithms modify the system? The objective of this paper is to explore these questions, starting from a simple computer-based model of the "cognitive world" just described.

To answer these questions, we must first describe the "cognitive" repertoire of the *adaptive agent* -- the system doing the adapting. Here I'll restrict the repertoire to a familiar set of possibilities: sets of IF(condition satisfied)/THEN(action) rules. Such rules, in the simplest case, implement the stimulus-response repertoires of

classical behavioral psychology. However, we can extend this repertoire considerably by defining the conditions and actions in terms of messages. That is, the agent is given a message-processing repertoire: The detectors (e.g., the vision cone) produce messages, the effectors (e.g., the elementary actions) are activated by messages, and internal processing (e.g., internal feedback and computation) is accomplished by message circulation and transformation. Because there is a considerable literature on such systems, called *classifier systems* [Lanzi et al. 2000], this paper only presents details directly relevant to the questions being asked. Classifier systems are computation-universal, in the sense that any program that can be written for a general-purpose computer can be executed by an apropos collection of these message-processing rules. This means that any signaling network that can be modeled by a computer simulation can also be modeled by a classifier system.

Using a classifier system to define repertoire refines the earlier question to: What kinds of adaptive algorithm can discover useful sequences of message-processing rules? Of course, the agent can accomplish the task by simply trying rules at random, gradually collecting those that "work", but such trail-and-error algorithms are neither interesting nor feasible. Under random trials it takes an unreasonably long time to find even short rule sequences. Is there something better? Can evolutionary processes produce useful sequences in feasible times? Classifier systems are designed for "on-line" modification by genetic algorithms and other adaptive algorithms. This feature opens the possibility of computer-based observation of adaptive changes in simple versions of signaling networks. In particular, simple classifier system models have the possibility of mimicking aspects of the ontogeny and evolution of *bio-circuits* -- the complex signaling networks of molecular biology.

The paper begins (section 1) with a general description of complex signaling networks, then uses this description as a guide to present (section 2) an exploratory computer-based model of an adaptive agent in the two-resource world described in the first paragraph. As already suggested, the agent will be implemented (section 3) as a classifier system that evolves as the agent accrues experience. Execution of this model demonstrates (section 4) the adaptive evolution of a resource-seeking signaling network, going from a single rule that produces a random walk to sequences of rules that provide compatible resource-seeking actions. Though the model is quite simple, the underlying program is written to provide easy extensions to much more complex worlds. In particular, the adaptive mechanisms apply without change to the full panoply of agents that can be modeled by classifier systems. The paper closes (section 5) with a discussion of the implications of the model for broader questions about the evolution of complex signaling networks.

1. Complex signaling networks.

Complex signaling networks abound, integrating systems as widely different as biological cells and the Web. Despite substantial differences in implementation,

complex signaling networks share important characteristics. Chief among these are:

(i) Parallelism and coordination. Complex signaling networks, by definition, consist of large numbers of "transmitter/receiver" nodes that send and receive signals. The networks of interest here involve massive simultaneity: many nodes act at the same time, producing large numbers of simultaneous signals. Bio-circuits, for example, typically use proteins as signals. These proteins operate in reaction cascades and cycles, providing positive and negative feedback to other cascades and cycles. A biological cell has large numbers of active proteins and their interactions must be tightly coordinated if the cell is to continue to function. Indeed, in all the networks of interest, coordination is a major problem. Each signal must go to appropriate destinations, and it must be appropriately interpreted at those destinations.

(ii) Conditional action. In complex signaling networks, nodes only act when they receive an appropriate signal. That is, they have the IF/THEN structure discussed earlier: IF [an apropos signal is present] THEN [act]. The act may itself be a signal, allowing quite complicated feedbacks, or the act may be an overt action such as such as shutting off a mechanism or binding to some site. Interlocking sequences of message-processing rules become programs that are executed in parallel, with all that implies for flexibility and breadth of repertoire.

(iii) Modularity. In a sense, a complex signaling network is automatically modular, the nodes being the modules. But that is not what is meant here. If we look to the rules associated with the nodes, it is unlikely that the system can handle a broad range of situations by having one rule for each distinct situation. A rule for reacting to "a red Saab by the side of the road with a flat tire" has many elements, or building blocks, in common with a rule for "a blue Chevy stalled at the intersection". It is better if the agent can activate a *set* of rules that react to the elements of the situation. The foregoing situations can be handled easily by combining rules dealing with "car", "roadside", "flat tire", "stalled", and the like. Reaction cycles that serve as building blocks are a common feature of bio-circuits. For example, the Krebs cycle of eight proteins is used by almost all aerobic organisms

.

The agent, by simultaneously activating a set of building block rules, can react to a broad range of novel situations, making combinatorics work for the system instead of against it. Also, because appropriate building blocks are used frequently in a wide range of situations, they are tested and confirmed at a high rate. When tested building blocks work in a parallel, coordinated fashion they provide great flexibility, but they force the coordination problem discussed in (i).

(iv) Adaptation and evolution. Complex signaling networks change over time. Many of these changes are more than random variations, they are adaptations that improve performance. The performance itself is the result of an intricate skein of interactions extended over space and time. Most of the interactions are distant,

in both space and time, from the *direct* causes of changes in performance. As a result, there is a considerable problem in determining which interactions were responsible for the changes. This problem is often called the *credit assignment* problem. The play of a game of strategy, such as checkers or chess, provides a useful metaphor for understanding the need for credit assignment: After a long sequence of moves, the player receives notification of a "win" or a "loss" and, perhaps, an indication of the size of the win or loss. There is little information about which moves along the way were critical to that performance. The problem, then, is to determine which moves might be useful in future games. Similarly, in a biological cell, the “reward” of reproduction results from the interactions of hundreds to thousands of signaling proteins over hours or days. The general question is: How does a signaling network allocate credit for desirable outcomes back to the responsible nodes (rules)?

There is a mitigating factor that is helpful in resolving this problem: The environments in which signaling networks operate do exhibit perpetual novelty, as does a complicated game like chess, but there are repeating sub-patterns in those environments. In chess these repeating sub-patterns have names like “fork”, “pin”, “gambit”, and so on. In the environment described at the outset the patches constitute repeating elements that can be exploited. In general, such sub-patterns can be exploited by particular arrangements of the signaling nodes, the modules alluded to in (iii). Close attention to the origins and ontogeny of modules used by a signaling network offers vital clues to its organization and performance in response to different environments.

It is the thesis of this paper that exploratory models built around these shared characteristics can give insights into the operation and evolution of natural and artificial signaling networks.

2. Signaling networks implemented as classifier systems.

This section starts with a definition of classifiers (section 2.1), then goes on to systems of such rules (section 2.2), and concludes with the description of an adaptive agent defined with the help of a classifier system (section 2.3).

2.1 Classifiers.

The components of a classifier system are condition-action rules, called *classifiers*. The condition part of a classifier "looks for" certain kinds of messages; when the rule's conditions are satisfied, the action part specifies a message to be sent. A computer-based definition of these rules requires a proper language for representing classifiers. I’ll first give the symbols used and then explain their use.

In this version, all messages are strings of length k, where each position contains one letter from the three-letter alphabet {1,0,?}. For example, if k=5, a

typical message would be 110?0. The set of all possible messages is the set M = $\{1,0,?\}^k$. In a similar fashion, the set of all possible conditions, C = $\{1,0,\#\}^k$, is the set of all strings of length k over the alphabet {1,0,#}. A classifier rule has either one condition, c drawn from C, or two conditions, c and c' drawn from C . The action part of the classifier is a message m drawn from M. A rule is written as c/m (one-condition rule) or c&c'/m (two-condition rule). In the rest of this section I will present definitions for two-condition rules, assuming the obvious simplifications for one-condition rules.

[It is easy to provide for messages of variable length and rules with multiple conditions, but it does complicate the exposition. Interestingly, the restricted system described here is still computationally-complete.]

A classifier is *satisfied* when its two conditions are satisfied. A condition c is *satisfied* by a message m if

(i) at each position in c that has a 1, the message has either a 1 or ? at that position;

(ii) at each position in c that has a 0, the message has either a 0 or ? at that position.

Accordingly, at positions where c has a #, no requirement is made on the message (that is, the condition "doesn't care" what occurs at positions where it has a #). It is also clear from (i) and (ii) that a ? in a message satisfies any condition requirement at that position (that is, the message "fits all" possible condition requirements, {1, 0, or #}, at all positions where it has a ?). For example, with k=5, the message 100?1 satisfies the conditions 1#### and 10#11, but it does not satisfy either the condition 0#### or the condition 10111.

2.2 Classifier systems.

For simplicity of modeling and exposition I will use a particularly simple version of a classifier system, though there are many varieties [Lanzi et al. 2000]. In this version, each rule has a *strength* that indicates it past usefulness to the system and, when a classifier is satisfied, it enters a strength-based competition to *post* its message. The details of strength assignment and the competition will be discussed next; for now, simply note that many rules can simultaneously win the competition. As a result many messages can be posted simultaneously. From a computational point of view, it is convenient to think of the messages as collected in a list. A posted message stays on the list only one time-step. To keep a message on the list it must be repeatedly posted by a winning classifier; as is the case for a television image, the list must be "refreshed" each time step. Messages provide communication between classifiers.

Strengths in a classifier system are modified in two ways. When an effector causes input to a reservoir, classifiers that have posted messages activating that effector are strengthened (classical conditioning). All other strength changes in a classifier system are the outcome of an ongoing competition that treats the whole

classifier system as a kind of marketplace. Each classifier in the system is treated as a go-between (middleman, broker) in this market. At any given time, the "suppliers" of a classifier are those classifiers that posted messages satisfying its conditions. When classifier wins a competition it is the "consumer" of each classifier that has just posted a message a message satisfying its conditions.

In more detail, a satisfied classifier makes a *bid* based on its strength, treating its strength as "cash-in-hand". The highest bidders win the competition, paying the bid to their suppliers. For this payment, the winners win the right to sell (post) their messages, hoping for consumers that will let them recoup their payments. Classifiers gain in strength if they make a "profit", paying out less than they receive in these transactions. This procedure for adjusting strengths is called the *bucket brigade* algorithm (See [Holland 1995, pp. 53-56] for details).

The basic cycle for this system is:
(1) All messages originating from the system's environment (via detectors) are added to the message list.
(2) All rules check all messages on the message list to determine which rules are satisfied.
(3) All messages are deleted from the message list.
(4) Satisfied rules enter a strength-based competition based; the winning rules post their messages to the message list.
(5) The bucket brigade algorithm adjusts the strengths of the winning rules.
(6) Changes in the system's environment, caused by messages activating effectors, are executed.
(7) Return to step (1).

Classifier systems model each of the characteristics of complex signaling networks listed in section 1. The simultaneous activity of the nodes of a signaling network is directly modeled by the simultaneous activity of the classifiers in a classifier system. The coordination provided by the signals in the networks is modeled by the message-passing of the classifier system, and conditional action is directly built into the IF/THEN format of the rules. As will be shown in section 3, modularity is provided by tagging loci in the messages [Holland 1995, pp. 12-15]. Tags serve much like headers on Web messages; modules result when a subset of rules coordinates its activity by the use of a common tag. Most importantly, classifier systems are designed to be modified by adaptive algorithms (see sections 3 and 4), so that we can examine the ontogeny and evolution of the system under various adaptive regimes.

2.3 Defining an adaptive agent with a classifier system.

This paper centers on agents that use classifier systems to determine their behavior and adaptation. The agents have five principle components:

(1) A list of classifiers.

This list may be modified in various ways as the agent adapts to its

environment.

(2) A list messages.

This list changes each time step, in accord with the output of the classifiers that win the competition.

(3) A set of detectors.

Detectors code information about the environment into messages

(4) A set of effectors.

Effectors have conditions, like those of the classifiers, that are satisfied by messages. The action part of an effector causes some change(s) in the environment.

(5) A set of reservoirs.

Certain effector actions, at appropriate places in the environment, cause the reservoirs to be filled. As in the introductory scenario (section 1), reservoirs are depleted at a constant rate.

In the implementation that follows (section 3), the agent will use a "vision cone" to collect information from a 2-dimensional environment with patches of resources. This detector will produce a single message that encodes that information. In addition, each reservoir will send a message when it is "low". As a result, the message list will contain the messages generated by active rules, the vision cone message, and any messages from low reservoirs. A further agent-environment interaction must be defined to make adaptation meaningful: There must be some actions the agent can take that provide needed resources from the environment. For the agents defined here that means that there must be actions that fill the reservoirs. The efficiency with which the agent manages to fill its reservoirs gives a measure of performance.

3. Implementation.

This section describes a particular implementation (in Mathematica) of the agent described more generally above. This simple version serves as an existence proof , exhibiting an adaptive procedure that takes a signaling system from a simple one-rule "founding" repertoire to a more complex, more effective repertoire. There are several easy generalizations of this implementation -- more resources, more environmental detectors, more effectors, etc. -- most of them attained by simply changing parameters in the program. Moreover, we'll see (section 5) that the mechanisms have counterparts in broader contexts.

The section begins (3.1) by setting some of the parameters for the classifier system, message length and the like. It continues (3.2) by describing the agent's environment, presenting details of the agent's input and output in that environment. Then the section describes (3.3) the details of the agent's classifier system, including the bucket brigade. The section concludes (3.4) with a detailed description of the rule discovery mechanisms used by the agent.

3.1 General parameters.

All messages in this implementation have length k=30. The first five loci in each message are used as a tag that identifies the origin of the message:

Messages originating from the reservoirs have a prefix tag 01010.

Messages originating from the environment have a prefix tag 101??.

Messages originating from the classifiers have tag prefix 10000.

Classifiers have either one or two conditions and one outgoing message. There is no limit on the number of classifiers or the number of messages on the message list.

3.2 Environment, detectors, and effectors.

The environment is a 20x20 grid wrapped around in both dimensions (a torus) so that there are no edges. 4x4 patches of resources of two kinds -- call them "shelter" and "food" -- are distributed irregularly in the grid; all other grid points are treated as empty.

This agent uses a single detecting mechanism to acquire information from the environment, a "vision cone". At any time, the vision cone points in a specific direction, some multiple of 45 degrees, and can be rotated in 45 degree increments by appropriate effector action (see below). It encompasses all grid points to a depth of 4 grid points within 45 degrees to the left and right of the vision direction. The agent is always assumed to be oriented in the same direction as the vision direction.

The vision cone produces the following 30-locus encoded message:

1 0 1 ? ? (v1)(v2)(v3)(v4) ... (v14) ? ? ? ? ? ? ? ? ? ? ?.

The values v1 through v14 at loci 6 through 19 encode information from the environment as follows:

v1 = 1 if and only if resource 1 is abundant at the current location of the agent, otherwise v1 = 0.

[A resource is considered abundant if its level exceeds the threshold set by the parameter *rare*.]

v2 = 1 if and only if resource is available, but not abundant, at the current location of the agent, otherwise v2 = 0.

v3 and v4 are similarly assigned with respect to resource 2.

v5 through v8 are assigned similarly, but with respect to the grid point that is one layer ahead in the current vision direction.

v9 =1 if and only if resource 1 is present in some amount at any grid point that is two to four layers ahead in the current vision direction.

v10 is similarly assigned with respect to resource 1.

v11= 1 if and only if resource 1 is present in any grid point within 45 degrees left of the vision direction, not including the center line, up to four layers away.

v12 is similarly assigned with respect to resource 2.
v13 and vi14 are similarly assigned for resources to the right of the vision direction.

The agent has six effectors, each of which is capable of one specific action. The actions are:
<r>, “rest”, stay in the same location with the same orientation.
<a>, “approach”, move one grid point forward in the vision direction.
<f>, “flee”, move one grid point backward, retaining the same vision direction.
<l>, “turn left”, rotate the vision direction 45 degrees to the left, while staying in the same location.
<r>, “turn right”, rotate the vision direction 45 degrees to the right, while staying in the same location.
<c>, “consume”, deplete resource 2 if it is present at the current site.
Each effector has an associated condition that can be satisfied by an appropriate message:

? 0 0 1 ? 1 1 1 0 0 ? <s>
? 0 0 1 ? 1 1 1 1 1 ? <a>
? 0 0 1 ? 1 0 0 0 0 ? <f>
? 0 0 1 ? 1 0 0 0 1 ? <l>
? 0 0 1 ? 1 1 1 1 0 ? <r>
? 0 0 1 ? 1 1 0 0 0 ? <c>

Note that sequence 001 at loci 21-23 serves as an internal tag that distinguishes messages directed to the effectors.

If the response <c> is given when resource 2 (“food”) is present, the level of that resource at that site is reduced by an amount determined by the parameter *twodec*. The reservoir level for resource 2 is increased by that amount until the reservoir is full. The resource at a depleted site recovers at the rate *inctwo*. If the response <s> is given when resource 1 (“shelter”) is present that resource is not depleted, but the reservoir level is increased by the amount *onedec*.

3.3 Classifier execution and the bucket brigade.

As pointed out earlier, each classifier on each time step checks the message list to see if its condition part is satisfied. If so, the classifier bids to place its outgoing message on the message list for the next time step. Because messages are only interpreted in terms of the classifier or effector conditions they satisfy there is no possibility of conflict at this level: additional active classifiers simply means additional messages on the message list.

Most classifiers in this system belong to one of three broad categories:
(i) A *default* classifier is a classifier with one condition, that is satisfied by one of

the “low reservoir” messages, and an outgoing message that activates one or more effectors.
(ii) A *bridging* classifier is a classifier with a condition, that is satisfied by one of the “low reservoir” messages, and an outgoing message that does *not* activate any effector.
(iii) A *bridge-supported specialist* is a classifier with one condition, that is satisfied by some environment message(s), a second condition, that is satisfied by a message from a bridging classifier, and an outgoing message that causes satisfies some effector condition(s).
These categories will be directly useful when we come to the agent’s discovery mechanisms (3.4). Also, as we’ll see in a moment, the categories simplify the determination of bids.

Just how is the size of a classifier’s bid determined once its conditions are satisfied? A classifier is assigned a *specificity* constant *spc* at the time it is formed, and typically the bid is calculated as a simple product,

$$bid(t) = (spc)*(strength(t)).$$

In many classifier systems [Holland 1995, pp. 57-60], the specificity of a condition is determined by the number of #’s in the condition: the fewer #’s, the greater the specificity. In this model specificity is assigned according to category:

Default rules are assigned *spc* = .02.
Bridging rules are assigned *spc* = .03.
Specialist rules are assigned *spc* = .05.

For technical reasons that I’ll describe later in this section, it is better to make the bid an S-shaped function of the strength rather than using the strength directly.

In this model, the winners of the bidding competition are the satisfied classifiers that make the largest bids. The actual procedure is stochastic (so that lower bidders can sometimes become winners):

Probability (win(j,t)) = probability that classifier j wins at time t

$$= [\text{bid}(j,t)/\text{maxbid}(t)]^{1/2},$$

where bid(j,t) is the bid of classifier j at time t, and maxbid(t) is the maximum of all bids made at time t.

Once we’ve determined the winners, we can execute the bucket-brigade. The exchange is between the current winning classifiers and the winning classifiers of the previous time step. Each current *winning* classifier has its strength *reduced* by the amount of its bid. The bid is distributed to its “suppliers” -- the classifiers that posted messages satisfying the winning classifier’s conditions -- *increasing* their strengths. In the present model, the bid is apportioned among the suppliers according to their strengths. The intuitive idea behind this means of apportioning credit, as outlined earlier, is that classifiers belonging to a supplier-consumer chain leading to a “good product” (reservoir filling) will prosper, as in the economic counterpart.

It is useful to determine the equilibrium (fixed point) strength, str*, of a

classifier under the assumption that it receives a fixed income I each time it is active. At equilibrium, the bid, spc∗strength, must equal the income I, so

$$str^* = I/spc.$$

At equilibrium, then, the bids of all classifiers with the same income will be the same, regardless of specificity. This defeats the intent that more specific classifiers should override (outbid) less specific classifiers in a default hierarchy. This difficulty can be corrected by making a bids an S-shaped function of strength, as suggested earlier in this section. Indeed, a simple function suffices,

$$spc*str*(2\text{-}str/maxstr),$$

where maxstr is the maximum strength that can be assigned.

Classifier strengths also enter into the resolution of conflicts when the conditions for several incompatible effectors are satisfied. The strengths of the suppliers of each effector condition are summed and the two effectors with the highest associated sums, *maxind* and *secind* respectively, enter a stochastic competition. In the present model the competition is resolved by

$$Prob(maxind) = 1-((seceff/maxeff)^2)/2),$$

were *seceff* is the sum associated with the the classifier with the second highest sum, *maxeff* is the highest sum, and secind is executed if maxind does not win.

3.4 Rule discovery.

A genetic algorithm (GA) is usually used to generate new classifiers in a classifier system [Holland 1995, pp. 60-80]. However, this paper centers on the *triggered generation* of classifiers. So, in the interests of getting a clear picture of the possibilities of triggered generation, the GA is not used in this implementation. I emphasize that the GA is compatible with triggered generation and provides a good way of exploiting "building blocks" discovered by triggered generation.

Triggered generation provides an experience-based procedure for bridged classifier sequences, without waiting for the classifiers to appear under mutation and recombination. The basic idea is to produce a lengthening sequence of coupled specialists supported by a bridge that is activated by a low reservoir condition. If this bridge-supported sequence captures causal (or highly correlated) interactions leading to a needed resource in the environment, then the bucket brigade establishes the sequence as an established part of the repertoire.

There are two objectives that serve as a sine qua non in setting up a triggered generation procedure.

(1) The mechanism(s) should be general-purpose in the sense set forth in the introduction; that is, the mechanisms should be effective over a wide range of environments, requiring no "tuning" that depends on prior knowledge of the environment.

(2) The mechanism(s) should not generate large numbers irrelevant classifiers; such "clutter" is almost certain to destroy the effectiveness of other

routines for credit assignment, recombination of building blocks, and the like.

Two triggering mechanisms are used in this model. One provides bridging classifiers when none exist, and the other provides new bridge-supported specialists coupled to already existing bridge-supported specialists. Some additional notation will make it easier to illustrate the description that follows; this notation is also useful in describing other signaling networks.

[m1, m2] indicates a classifier condition that is satisfied by either of messages m1 or m2.

<a> indicates a message that satisfies the condition for effector a.

The mechanism for generating a new bridging classifier is invoked only when the following three criteria are satisfied:

(i) the agent has just received a significant input to a reservoir;

(ii) the corresponding "low reservoir" message is on the message list;

(iii) there is no active bridging classifier.

If these conditions are met, a new classifier is added to the system. It has a single condition that is satisfied only by the particular "low reservoir" message meeting criterion (ii). The outgoing message has a distinctive *bridge*-tag and is designed so that it does not satisfy any extant conditions (typically achieved by hash-coding the non-tag part of the message).

At the same time the bridging classifier is formed, as part of the same triggered operation, a specialist supported by that bridge is also formed. This specialist classifier has two conditions. The first condition is satisfied by the bridge's outgoing message, and the second condition is satisfied by relevant parts of the current message from the environment. The specialist's outgoing message satisfies the condition of the just executed effector. Note that all information to form this pair of classifiers is directly available at the time they're formed.

For example, assume the "low food reservoir" message *hunger-mess* is on the message list, that "food present" is indicated by environmental message *food-pres*, and the response *consume* has just been executed. The following two classifiers would then be added to the system:

[hunger-mess]/m1

[m1]&[food-pres]/m2 <c>,

where m1 is hash-coded and m2 is a message that satisfies the condition for effector *consume*.

The mechanism for generating a bridge-supported specialists is invoked only when the following three criteria are satisfied:

(i) a bridging classifier was active on the previous time-step and remains active;

(ii) a bridge-supported specialist is currently active;

(iii) there was no bridge-supported specialist active on the previous time-step.

When these criteria are met a new two-condition, bridge-supported specialist is generated using the following information:

time	t-1	t
environment	*env(t-1)*	*env(t)*
active classifiers	*bridge*	*bridge*
		spec(t)
effector	*eff(t-1)*	*eff(t)*

The first condition of the new specialist, [*bridge*], is satisfied by the *bridge* message, while the second condition, [*env(t-1)*], is satisfied by the relevant parts of the previous message from the environment, *env(t-1)*. The specialist's outgoing message satisfies the condition of the effector *eff(t-1)* executed on the previous time step. When the specialist is formed the conditions of the active *bridge* and the pre-existing bridge-supported specialist *spec(t)* are generalized so that each is satisfied by new specialist's message. That is, the bridge's revised condition, [*new spec*, *low reservoir*], now accepts the specialist message as well as the "low reservoir" message, and the revised condition of the pre-existing specialist, [*new spec*,*bridge*], now accepts the new specialist's message as well as the bridge classifier's message.

By way of example, assume that

[hunger-mess]/m1

is active at time-step t-1, but no bridge-supported specialist is active at that time. Further, assume that environmental message *f-targ* indicates that food is "in sight", and that the "approach" response is currently being executed. Let

[m1]&[food-pres]/m2 <c>

be activated at time-step t. The three criteria for generating a bridge-supported specialist have been satisifed, so the new bridge-supported specialist

[m1]&[food-targ]/m3 <a>

is formed. The bridge condition and one condition of the pre-existing specialist are simultaneously generalized, so that the bridging classifier now has the form

[hunger-mess,m3]/m1,

and the pre-existing specialist now has the form

[m1,m3]&[food-pres]/m2 <c>.

The bridge-generating mechanism clearly works hand-in-glove with the bridge-supported-specialist mechanism. The first mechanism provides the "founders" upon which increasingly long sequences of specialists are built. The specialists formed by the specialist mechanism are coupled so that the bucket brigade transfers credit back up the line. Moreover, the bridge's strength is immediately increased by the bucket brigade, because the bridge is active at the time of reinforcement. As a result, when the sequence is next activated, there is a quick increase in the strength of the initial specialist classifier in the sequence, because the bridge is a "supplier" to that initial classifier. In other words, because of the bridge, credit need not filter back slowly, trial-by-trial, to the stage-setting initiator.

4. Execution.

While it seems plausible that an agent using these mechanisms will generate effective resource-acquiring behaviors, "the proof is in the pudding". No amount of abstract argumentation will provide a "proof" -- there are just too many ways the process can go wrong. For example, the generating mechanisms may generate an unwieldy clutter of classifiers that interfere with each other, or the bucket brigade may interfere with the long-term stability of the bridge-supported sequences, preventing critical overrides of default rule(s). An executable, proof-of-principle model is necessary (the counterpart, in this more mundane context, of von Neumann's model showing that self-reproducing machines are possible).

To this end, the foregoing model has been implemented in Mathematica. The agent starts with a single classifier

[fatigue-mess, hunger-mess]/m <s,a,f,l,r,c>.

This classifier serves as a broad default, with enough '?' in its output message to assure that the conditions of all 6 effectors are satisfied. To resolve the effector conflict, as detailed above, the effectors go into a competition where each is equally likely to be a winner. The result is a random sequence of effector actions, producing a kind of random walk in the environment.

Under the action of the generating mechanisms, the agent acquired the following repertoire after just 250 time steps in a typical run of the program:

(1) [fatigue-mess, hunger-mess]/m <s,a,f,l,r,c>.

(2) [hunger-mess,m3]/m1
(3) [m1,m3]&[food-pres]/m2 <c>
(4) [m1,m4]&[food-targ]/m3 <a>

(5) [fatigue-mess,m7]/m5
(6) [m5,m7]&[shelter-pres]/m6 <s>
(7) [m5]&[shelter-targ]/m7 <a>

There were no other classifiers; the strengths of these classifiers at time-step 300 were, in the order given,

{1000, 958, 1000, 685, 733, 995, 500},

where 1000 is the maximum strength allowed.

At the end of 500 time-steps the repertoire had expanded to include

(8) [m5]&[no shelter]/m8 <r>

with classifiers (5) and (7) modified to

(5') [fatigue-mess,m7,m8]/m5
(7') [m5,m8]&[shelter-targ]/m7 <a>

There still was no clutter, and the strengths of the classifiers were
{1000, 1000, 1000, 787, 733, 926, 1000, 500}.

Logic alone would suggest that the generated classifier sequences produce an improved search for patches, as compared to the random search generated by the default classifier. Still, a rough measure of the improvement is helpful. One way to measure performance in this context is to keep track of the average level of the reservoirs. As a control experiment, the above run was repeated without triggered generation. For this run, the average reservoir levels between time steps 200 and 500 were 108 and 109, respectively. During the same time interval, in the above triggered generation run, the average reservoir levels were 212 and 182.

Of course, a few runs like these only hint at the long-run structure of the agent, and they only suggest the effects of these generating mechanisms alongside the GA in more complex environments. However, the run described does constitute an existence proof that an agent can use experience to go from a simple default to a more complex goal-directed repertoire, without external supervision.

5. Relevance to the evolution of signaling networks.

Because of the message-passing basis of the demonstration, this existence proof is suggestive for other signaling networks. There are details, of course, that only have clear counterparts in some signaling networks. The bucket-brigade, for instance, has a clear counterpart in economic networks, but it is less clear what the counterpart would be in a bio-circuit. Still, there is a clear path from this model to models that are much closer to real bio-circuits. In this section I'll sketch that path for bio-circuits, noting that the relation extends *mutatis mutandis* to other signaling networks with agents that employ message-passing, such as food webs, neural networks, ecosystems, and markets.

Control, synthesis, and transport in biological cells are all mediated by the complex signaling networks we've been calling bio-circuits. The interactions are typically conditional interactions that use proteins and other bio-molecules as signals. They involve positive and negative feedback, repression and de-repression of genes, and coactivation of multiple genes. The ubiquitous Krebs citric acid cycle and the Lambda Switch in E. coli are familiar examples of highly regulated bio-circuits. Chemotaxis in E. coli and the formation of the fruiting body in slime mold offer more complex examples, while the interaction of induction and competence in a developing metazoan provides a still more complex example. Ultimately, bio-circuits underpin "epistatic" interactions among genes, giving rise to effects ranging from the repression and de-repression of genes to the directed placement of

proteins, such as cell surface receptors.

Even when the same genes are being expressed, bio-circuitry produces dynamic sequences of protein synthesis, modification, and regulation. But the complexity does not end there: Because bio-circuitry can turn genes on and off, there can be complex, time-mediated interactions between genes. For example, in a circuit as simple as the lambda switch in E. Coli, which of two gene-controlled pathways is invoked (lysis or lysogenesis) depends upon a complex bio-circuit with sophisticated interactions triggered by the cell's environment.

There are useful commonalties between bio-circuits and classifier systems. First of all, classifier systems exhibit the characteristics of signaling networks set out in section 1:

- Parallelism and coordination
- Conditional action
- Modularity
- Adaptation and evolution.

In addition to these characteristics, *tagging* is a mechanism that is shared with bio-circuits.. Tags in a bio-circuit are exemplified by the particular amino acid sequences that identify antigens, the sets of amino acids that define actives site in enzymes, and the amino acid sequences that bind to specific loci in DNA. Tags go under a variety of names in molecular biology -- receptors, ligands, motifs, active sites, and so on -- and they are critical in controlling reactions involving transcription factors, enhancers, activators, and co-activators, and the like. In each case, there is a sub-configuration of the carrier (polypeptide, RNA, DNA, ...) that satisfies some binding condition. The counterpart in a classifier system is a rule of the form

IF (message with apropos tag present} THEN (send message with new tag).

Tag-like motifs identify, and coordinate, modules in a bio-circuit, much as a coordinated group of classifiers is activated by messages with similar tags. The condition for activation of a module can be more or less precise, the equivalent of adding or deleting #'s in a classifier condition. For example, one major enhancer for cancerous growth is a reduced selectivity for growth factors in the cascade initiating cell division, allowing the cell to divide in foreign contexts. We will come shortly to the evolutionary "tuning" of tags as a critical feature of cell phylogeny.

There is a straightforward way to go from simple classifier models, of the kind examined in the last section, to more realistic classifier models of bio-circuits. One starts with a coarse-grained, over-general model that describes a few well-established generalizations. Then through a series of iterations, one reaches a classifier model of a bio-circuit, wherein each classifier specifies what happens to some actual biomolecular cell constituent (signaling protein, promoter, repressor, or the like).

For example, starting with a healthy cell, we might adopt as a starting point

four simple rules to model the transformation of a normal cell to a cancer mass:

(1) IF [healthy cell & DNA damage] THEN [apoptosis or immortality].
(2) IF [immortality] THEN [stable existence or genetic instability].
(3) IF [genetic instability] THEN [ephemeral clonal expansion or robust clonal expansion].
(4) IF [robust clonal expansion] THEN [cancer mass].

This initial set of rules serves as a framework for more elaborate models that are more closely connected to actual observations. Rule (1), for instance, might be elaborated to:

(1.1) IF [healthy cell & DNA damage] THEN [apoptosis or mutation for resistance to apoptosis].
(1.2) IF [resistance to apoptosis] THEN [susceptibility to growth inhibitory signals or mutation for loss of susceptibility to growth inhibitory signals].
(1.3) IF [loss of susceptibility to growth inhibitory signals] THEN [selective growth advantage and immortality].

It takes several such iterations to arrive at a model where (i) the conditions (the IF part) are specified in terms of actual cell constituents (signaling proteins, promoters. repressors, and the like), and (ii) the actions (the THEN part), likewise involve the production of cell constituents (through gene expression, for instance). Rules at this level may also represent DNA loci that are repressed or de-repressed by proteins having particular tags (e.g., configurations that provide binding action). Representing DNA loci allows us to examine a critical aspect of cell-development: the effects of repressors, promoters, and the like, on the dynamics of the bio-circuit. A classifier model at this level can be tested against data produced, for example, by microassays.

There's a fragment of a bio-circuit involved in tumor growth that can be used to exemplify an IF/THEN model at the level of bio-molecules.
The rule

IF[ras gene expressed] THEN[tumorigenesis]

can serve as the default starting point. The specialist rules at the bio-molecule level have a *growth factor* (gf) as their central component. Growth factors control the growth and replication of cells; normal cells only replicate when a highly selective, organ-specific *growth factor receptor* is activated by the appropriate growth factor:

(initiation) IF[apropos growth factor] THEN[gf receptor activated].

The cascade of events leading to mitotic transcription (replication) is initiated by the bio-molecule Ras-GTP. Ras-GTP is obtained by adding a phosphorous ion (P)

to Ras-GDP, a reversible process controlled by the bio-molecules GDS and GAP. These relations can be represented by the following simple set of IF/THEN rules:

(regulation) IF[gf receptor activated]&[GDS]&[Ras-GDP] THEN[Ras-GTP].
IF[gf receptor activated]&[GAP]&[Ras-GTP] THEN[Ras-GDP].

(execution) IF[Ras-GTP] THEN[mitotic transcription & stress fiber cascades].

Even this simple set of rules suggests several factors relevant to tumor growth:

(i) genes producing GAP could be tumor suppressors.

(ii) genes producing GDS could be oncogenes.

(iii) a mutation in the ras gene, allowing susceptibility of Ras to inappropriate growth factors, could cause inappropriate activation of Ras.

(iv) stress fiber cascades in inappropriate conditions, e.g. production of membrane ruffles, could be used as indicators of aberration.

Because this bio-circuit fragment is so simple -- a single two-component regulatory process and one expressed gene -- these suggestions are readily apparent to "common sense". However, when we come to more complex bio-circuits with interlocking feedback loops and interacting gene expression, the resulting behavior is far from apparent. The situation then is much like trying to determine the behavior of a lengthy computer program using only the listing of instructions, a notoriously difficult task.

Even when the full description of the bio-circuit is available, it is quite difficult to anticipate the effects of gene mutations, exogenous signals, and the like. Yet, it is just these interlocking causative factors that offer possibilities for targeted intervention. As with a computer program, execution of the bio-circuit "program" under controlled conditions becomes one of the few feasible ways of attaining this understanding. Sometimes in vivo or in vitro experiments can serve this purpose, but control can be extremely difficult, as decades of benchwork in the study of cancer has made clear. Computer-based models of bio-circuits offer a powerful complement to this benchwork, suggesting lines of research that might not be apparent otherwise.

Once the classifier system reaches the bio-molecular level, there are several useful properties of the model that aid in exploring the activities of the corresponding bio-circuit:

(i) There is a clear correspondence between each component rule of the model and each component of the bio-circuit.

(ii) It is easy to develop standard models of important modular bio-circuits such as the Krebs cycle or the lambda switch. By using appropriate tags to mimic receptors and ligands, these modules are easily incorporated in larger bio-circuits, offering a convenience like that of standard sub-routines in a computer program.

(iii) Because the classifier system is computation-universal, the model is

easily modified to account for any shortcomings or errors vis-a-vis the bio-circuit. In contrast, models using simultaneous linear differential equations (typical in physics and chemistry) can handle conditional actions only with great difficulty, limiting such models to the simplest bio-circuits

(iv) Classifier systems are built to be used as grist for a genetic algorithm, so they can be subjected to artificial evolution, making it possible to explore phylogenetic relationships between different bio-circuits. It is easy to trace generalizations or specializations of conditions under an evolutionary regime, say by point mutation, because of the role of # ('don't care") and ? (fits all) in defining conditions and messages. As an example, a good classifier system model for early-stage cancer would enable us to observe the likelihood of mutations that transform the cancer to a more aggressive stage.

When constructing a bio-circuit model, it is important to look for basic building blocks in the system being modeled. Just as enzymes have basic structural components -- alpha helices, beta sheets, and the like -- constructed from a 20-amino-acid alphabet, so there are standard "signaling" proteins for turning genes "on" and "off". There are also standard " autocatalytic bio-circuits", such as the citric acid cycle, that perform similar functions over extraordinarily wide ranges of species. The goal is to come up with building blocks that can be fitted together to make larger building blocks.

Extracting building blocks is accomplished by examining the quantities measured to keep track of the system's behavior. The construction of a flight simulator provides an easy example: You list the modules that generate the instrument readings in the plane's cockpit; you then determine the rules that describe how the modules interact. The process has much in common with defining a new board game by listing the pieces and rules. This description in terms of modules or building blocks is not just a matter of convenience; when we look to the evolution of bio-circuits we see the same basic building blocks occurring over and over again in different combinations.

Models built around classifiers complement, but contrast strongly with, the statistical models produced by bioinformatics. Genome sequencing has made it possible for us to identify the building blocks of important signaling networks, while new tools such as automated gene sequencing, cDNA micro-assays, and tissue arrays produce torrents of data about these components. It is difficult to organize this torrent in ways that tell us more about the signaling networks. As with building blocks sitting in a box, many known components await assembly into coherent structures. In attempting to assemble these building blocks, it is helpful to note again that modules in bio-circuits, as chains of conditional interactions with feedback loops, are quite like sub-routines for a computer.

It is impossible to reconstruct a computer program from the statistics of its output. For similar reasons, the interactions in a bio-circuit (e.g., cooperative repression) are usually hidden from bio-informatic data searches. This limitation on

statistical techniques is familiar in other areas: No one would expect to use the statistics of a chess game (number of times each piece was moved, for example) to recover the conditional strategic maneuvers of the game. On the other hand, we can execute a classifier system model, much like playing a game of chess, to see if it simulates known data (cascades, feedbacks, and the like). In particular, we can examine the dynamic time-sequence of gene expression, so we can learn how the action of the bio-circuit is altered by transformations in the cell DNA. Correlations, regressions, and similar statistical techniques can increase confidence that a proposed bio-circuit reconciles the data, but statistical techniques alone cannot reveal the circuit.

There is a particular role for bio-circuit models as provisional hypotheses for guiding and refining experiments: They tell us "where to look". To see just what this means, it is helpful to turn to physics. In physics, the theory of relativity suggested that the path of light passing near a substantial mass would be bent. In particular, the image of a distant star, observed from earth as it orbited the sun, would be displaced if the sun came close to the line of sight to the star. Theory, then , suggested observations of relative star positions during an eclipse (which would allow star images near the sun to be seen). These new and unusual observations were one of the first verifications of the theory of relativity. In a similar fashion, bio-circuits models should suggest new experiments for filling in missing signal pathways.

There is a question about the bio-circuits of different species that, if answered, will make a strong contribution to our understanding of biological signaling networks: Why and how do certain motifs become common in bio-circuits, acting as building blocks for a wide range of functionally similar proteins, while other configurations remain particular and local? In short: How are building blocks selected and how do they spread across species? Darwin gained insights into the origin of species by tracing the origin and variations of the beaks of Galapagos finches. Similarly, a phylogeny of the origin and variations of common bio-circuit modules should tell us much about the origins and organization of bio-circuits. And, if some of these modules are implicated in diseases or cancers, we gain targets for targeted intervention.

Acknowledgments.

The urge to undertake this line of investigation was much amplified by three SFI meetings on resilience, robustness, and the evolution of language, respectively. I particularly thank Buzz Holling, Jim Brown, Erica Jen, and Bill Wang for gathering these working groups and for discussing with me what I might contribute to the effort.

References.

Holland, J.H. 1995. *Hidden Order*. Reading MA: Addison Wesley. pp. 53-56.

Lanzi, P.L., W. Stolzmann, S.W. Wilson (eds.). 2000. *Learning Classifier Systems*. Berlin: Springer.