Autopoietic Networks

Autopoietic means self producing and self organizing. An autopoietic network is a network, which through autonomous elements called agents, with internal rules of what is 'good behaviour', learns how to solve some information processing task that an external supervisor presents to it.

After the following describtion of the agents follows a describtion of each of three types of agents. The document concludes with a describtion of how the network deals with memory. Since the document is still on a development stage, some terms are introduced 'without warning' or sufficient describtion, sorry for that...

Agents

Autopoietic networks are composed of three types of agents: Providers, inpoders and effectors. In common for all three is that

- they have a specific *appearance*
- they have a certain *fitness*
- they have a certain *processing state*
- they have a certain *mating state*
- they have a maximum *lifetime*

when referring to the state of an agent, the processing state is meant. When referring to the mating state (ready to mate or not) the word mating will be included.

Each agent has a limited lifetime, ranging from a few seconds and up in realtime. The lifetime is counted in number of cycles. An agent can die before its time, if its fitness goes negative. This will e.g. happen to agents who bid very agressively for messages, and therefor wins the auctions, but have no buyers when it have processed the message (maybe because the customers have died before their time). These agents will pass away without the possibility to mate. Healthy agents, i.e. agents that through successfull buying and selling have increased their fitness, will mate with other healthy agents.

<u>Definition of neighbourhood</u>

The neighbourhood of an agent is minimum the 26 positions ($3^3 – 1$) enclosing the position held by the agent in the three dimensional network. An agent with this neighbourhood is said to have a vicinity of 1. An agent with vicinity 2 has a neighbourhood consisting of 124 positions ($5^3 – 1$), vicinity 3 corresponds to 342 positions ($7^3 – 1$) and so on. The maximum neighbourhood for an agent is set to vicinity 4 for computational reasons.

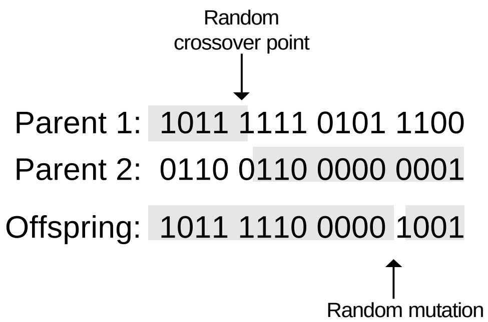
The four behaviour states

Time in the autopoietic network progresses in what is called cycles. A cycle is much like a turn in games like e.g. monopoly. In each cycle each agent will have one of the following four behaviour states.

The observer is an agent who observes a number of other agents in its neighbourhood, waiting for any one of them to announce an auction. If that happens it determines if it wants to make a bid for a message, and if so how large the bid should be. In the end of the present cycle it will be informed if it made the highest bid, and if so it pays the fitness points, receives the message and changes its status to *processor* for the next cycle. If it didn’t make any bid or didn’t make the highest bid it keeps the observer status.

The processor is an agent who won the auction, because of a bid it made in the previous cycle, and therefor has a message to process in this cycle (the information processing is distinct for each of the three kinds of agents and is therefor described for each of them below). It process the message and announce an auction to any observers there might be. In the end of the present cycle it will receive fitness points from the highest bidder and then change the status to either pilgrim, observer or breeder for the next cycle depending on the neighbourhood and its fitness.

The breeder is an agent who is ready to mate (an agent is ready to mate whenever its fitness is above some internal threshold value) and has another agent in the neighbourhood, who is also ready to mate and is within its accepted mating-appearances. This spotted agent also has the status breeder, but is referred to as *mating subject* to distinguish the two. If the appearance of the breeder is within the accepted mating-appearances of the mating subject everything is jolly and they mate. If however the appearance of the breeder lies outside the accepted mating-appearances of the mating subject, reproduction only occurs if the fitness of the breeder is higher than the fitness of the mating subject. If all the mating subjects in the neighbourhood are unwilling and too strong, the agent changes it status to pilgrim for the next cycle (note that if there were no mating subjects in the neighbourhood the status of the agent would already be pilgrim). If on the other hand reproduction occurs, an offspring is produced using the crossover and mutation principle, as illustrated in the following figure.

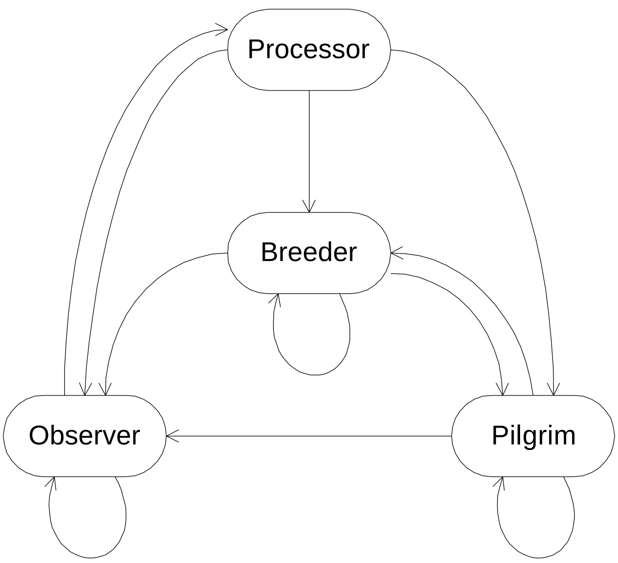


The bit sequences shown on the figure is referred to as the DNA string or genetic string. The DNA is the blueprint used to construct the agent.
Each of the parents will transfer a certain percentage of their fitness to the offspring. If this causes the fitness of the breeder to go below the mating threshold, it will change its status to observer or pilgrim (depending on the presence of supliers in the neighbourhood) for the next cycle. If the fitness keeps above the mating threshold and there is still mating subjects in the neighbourhood (maybe the same one), it keeps the status breeder for the next cycle.

The pilgrim is an agent who search for a better spot in the network. The motivation to move is always because agents in the neighbourhood doesn't look right to the agent, according to an internal idea of accepted appearances. If the agent is ready for mating (fitness above a certain internal threshold) it moves because there is no agent in the neighbourhood with an accepted appearance, who is also ready to mate. In the other case, where the agent is not ready to mate, it moves because there is no agents matching the supplier image it has. A pilgrim with vicinity 1 takes one step in a random direction. Pilgrims with vicinity 2 takes two steps and so on. When the pilgrim have moved to the new position it looks around and if it finds agents with the proper appearances it changes its status to either observer or breeder and then waits for the next cycle. If it doesn't find the agents it is looking for it keeps the pilgrim status and hopes that the next cycle will bring more luck.

The four agent behaviour states and their transitions is shown on the following figure:

State diagram for agent behaviour



Two examples of how the life for two agents, with a maximum lifetime of 10 cycles could occur:

Cycle	1	2	3	4	5	6	7	8	9	10	11
Agent1	Obs	Obs	Proc	Obs	Proc	Pil	Pil	Pil	Pil	Breed	(Dead)
Agent2	Obs	Proc	Obs	Proc	Obs	Obs	Obs	Obs	Obs	Obs	(Dead)

In the example agent 2 has lived in vain, in the sense that it is forgotten after cycle 10. Agent 1 on the other hand managed to reproduce, thereby leaving its mark on the network through its offspring.

Agent DNA

The part of the DNA string that is in common to all three types of agents is

Agent DNA

Agent feature	**# of bits**
Appearance	10
Vicinity	2
Ideal appearance (mating)	10
Pickiness_appear_mating	3
Percentages of fitness transferred to offspring	6
Promilles of fitness used for bid	8
Minimum bid	6
Minimum lifetime (value x 10 cycles)	5
Lifetime (value x 100 cycles)	5
Total	56

Comments to the chosen values:

The appearance is simply a string of 10 bits. The pickiness for appearances is the number of differing bits, in this case between 0 and 7. A pickiness of 7 differing bits out of 10 in the appearance, has in effect virtually no constraint on the accepted appearances (it is very unlikely that only 2 or less bits are the same out of 10). Agents with a pickiness of 7 will not spend as much time being pilgrims as an agent with a pickiness of 0 will.

The lifetime is maximum 3100 cycles. If the lifetime is below the minimum lifetime, the minimum lifetime is used. The minimum lifetime is with the chosen values between 10 and 320 cycles. Assuming that one cycle takes 20ms this means that a provider will exist between 0.2 and 62 seconds. Really freaky agents may exist for a shorter time than 0.2 sec, since an agent dies if the fitness goes negative. Its hard to say whether these lifetime intervals are apropriate for all three types of agents (will also depend on the task for the network), but the philosophy is that if a longer lifetime is needed for any agent type it will be evident from the genepool after a certain training period. Intuitively one might argue that a long lifetime is always beneficial, since this gives more time for reproduction finding the proper spot in the network etc., but this is not necessarily the case since long lifetime also slows down evolution; e.g. parents blocking the potential success of their offspring. But an implementation will answer this for the specific problem at hand.

The percentage of fitness transferred to offspring is between 1% and 64%. Parents that transfer a small percentage to their offspring will produce more offspring, but their offspring will have a smaller chance of survival, and vice versa.

With the choice of 8 bits determining the promille of the fitness used in bidding, this gives possible values from 0% to 25,5% of the fitness used in the bid. The minimum bid is between 1 and 64 fitness points. High bids increases the chance of winning auctions, but also requires many and/or strong buyers in order for the information processing to be profitable, and vice versa. Small bids can be profitable, as high bidding agents who usually ‘takes it all’ in the neighbourhood, sometimes changes their status to breeder or they die. In these cases agents who were bidding for the same type of messages will now win the auction, because the ‘rich dictator’ is now gone either permanent or temporarily. The ones that step in, could e.g. be the offspring or pilgrims that happen to have usefull decision rules.

Agent type: Provider

The provider provides messages or information to the autopoietic network. The providers buys messages (information strings) from either the environment or from effectors within the network. The messages from effectors are described below. Messages from the environment are in the case of a network used for speech recognition, acoustical observations, e.g. as a minimum a number of vectors with information about energy in specific frequency bands.

$$\text{Message: } \{ V_1, V_2, ..., V_N \}$$

When the message is bought from the environment, the fitness paid for it is stored in the money bag belonging to the environment. (It is from this money bag that effectors are rewarded.)

Because all the vectors in the message have the same dimension and content, each vector V_i is said to belong to a certain resource category. Hence a resource category is just a convenient way to refer to a vector with a certain dimension and contents (the contents is important since a vector with cepstrum coefficients is different than e.g. a vector with FFT coefficients). A usefull metaphor is to think of the categories as e.g. fruit, insects and metal. A specific value of a vector can then be interpreted as e.g. able, ant and kobber, respectively (just a metaphor, take it or leave it ;-)).

The DNA structure for providers is

Provider DNA

Provider feature	**# of bits**
(Agent features)	56
Ideal appearance (observing)	10
Pickiness_appear_observe	3
Total	69

(Exactly how providers buys messages from the environment is more an implementation question than a design question; a possibility is to add “fountains” which pour in messages from the environment)

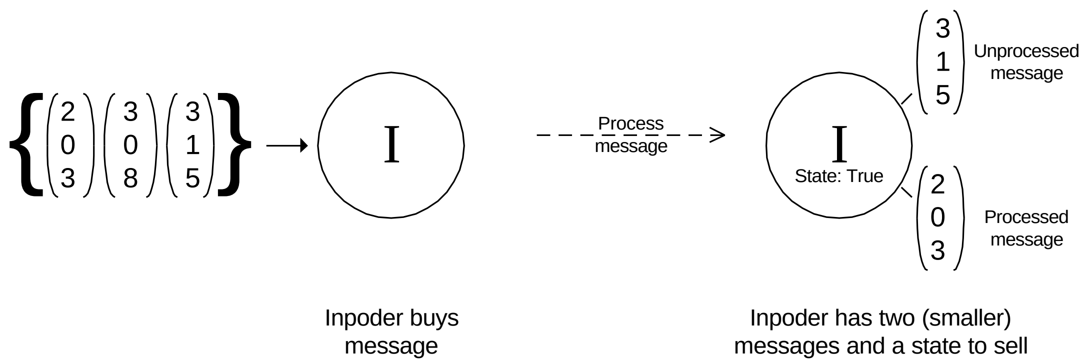
By definition the state of a provider that has a message to sell is *false*, and the message is labelled *processed*. This is something inpoders use to determine whether they want to bid or not.

Agent type: Inpoder

An inpoder is able to proces a message in the following way. When it buys a message it starts looking for its 'favorite food', i.e. a vector with a specific value. It looks at each resource in the message one at a time from left to right, comparing them to its own internal ideal value. In addition to the ideal value the inpoder has an internal pickiness value, which specifies how picky the inpoder is.

As an example consider an inpoder with a favorite food, $[3\ 0\ 2]^T$ and a pickiness value $[0\ 1\ 99]^T$. So the food that this inpoder eats has values in the range: $[3 ; (-1:1); (-97:101)]^T$.

Assume that the inpoder win an auction and buy the message $\{[2\ 0\ 3]^T,[3\ 0\ 8]^T,[3\ 1\ 5]^T\}$. It looks at the first resource and discards it, since it is not within the pickiness range of its ideal food. The second resource, however, is within the pickiness range. So it eats the resource and is now happy, i.e. its state is set to true. Since the inpoder is now happy it stops further processing and therefor has two messages that it can sell to other inpoders, a *processed message* $[2\ 0\ 3]^T$ and an *unprocessed message* $[3\ 1\ 5]^T$. If the inpoder has added any information value to the message, it will be able to sell its messages for a higher price (total) than it paid and thus increase its fitness. The process is shown on the following figure:



In order for it to be able to sell its messages, at least one inpoder will have to be observing it. The state information, in this case true, can be sold to effectors.

Lets assume that three inpoders are observing and thus are notified now that the inpoder announce an auction.

The observing inpoders then apply some internal decision rule to decide whether they want to bid for any of the messages or not. Each inpoder can only bid for one message in a cycle. The decision rules that the observing inpoders use, depends on the state of the seller, the type of the message (processed or unprocessed) and the size of the message[1].

[1] In addition to these three conditions, the decision could depend on the age of the message, i.e. how many times it has been bought and sold. But this information is more or less present already, because of the position of the inpoder in the autopoietic network. Therefor this condition is omitted.

Indirectly the decision rule also depends on the appearance of the inpoder (or provider) since an inpoder will not even observe inpoders that doesn't look 'right' to it.

The following could be the "reasoning", that lies behind the decision rule applied for each of the three observing inpoders:

Inpoder 1:
If the sellers state is true then my experience tells me that I can not sell my processed or unprocessed messages to anyone, and noone is ever interested in buying my state information. Therefor I only buy from inpoders whose state is false.
Decision = don't bid (seller state not correct)

Inpoder 2:
If his state is true and the offered processed message is less than 3, I'll bid for his message, since many effectors are interested in my state after I have processed the message, but only when his state is true and the processed message size is small, my experience tells me. That is I add a lot of information, by determining whether my ideal food is present (within pickiness range) or not in the small processed message when his state is true, otherwise there would not be so many buyers.
Decision = bid (seller state ok and message size is 1 (small))

Inpoder 3:
I'm only interested in his unprocessed messages, since my experience tells me that there might still be resources left that I like. And if there are, and my state therefor becomes true, I have many buyers for the messages I want to sell. If my food is not there, bad luck, but if the unprocessed message is larger than 9 resources, I'll give it a shot.
Decision = don't bid (seller state ok, but unprocessed message too small).

The word 'experience' is used because an inpoder on a specific place in the three dimensional network, is at that spot with the internal decision rule that it has and observe the agents that it does, because it's ancestors are those with the highest fitness in that area, i.e. those who got close to the ideal decision rules in terms of rewards from the environment. Many other types of inpoders have existed in the area (remember each agent only lives a few seconds), with different internal decision rules, e.g. some which only bought messages larger than 999, to take an extreme, but they would have faded away quickly without passing on their genes.

If there were no other inpoders observing, inpoder number 2 would buy the processed message $[2\ 0\ 3]^T$ and the unprocessed message $[3\ 1\ 5]^T$ would be wasted.

The bid is as mentioned calculated as a certain percentage of the fitnes of the inpoder. Some inpoders have a 'live strong, die young' mentality and thus bid a high percentage of their fitness. Others are more conservative and only bid a small percentage of their fitness. In any case the highest bidder always take it all, including the risk that there is no buyers or only low bidding buyers.

The DNA structure for an inpoder is:

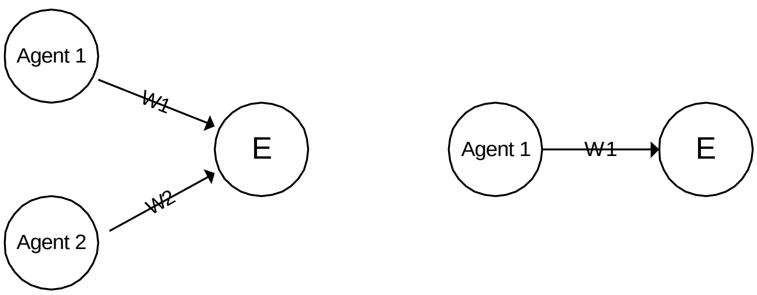
Inpoder DNA	
Inpoder feature	**# of bits**
(Agent features)	56
Ideal appearance (observing)	10
Pickiness_appear_obs	3
Care about seller state	1
Accepted seller state	1
Likes small messages	1
Message size threshold	4
Likes processed messages	1
Bids for second best message	1
Ideal resource	8
Pickiness_resource	7
Total	93

The inpoder is the most complex of the three agent types. It takes 93 bits to uniquely specify an inpoder (spatial and temporal information excluded), giving the possibility for 2^{93} different inpoders.

Note that the resource can be specified with 8 bits, which means that each energy 'vector' is one dimensional with 256 discrete energy levels. This is of course unrealistic in a real implementation, but is chosen for an initial test. Also it allows for each vector to be represented by an ASCII character, which is convenient in illustrations. The pickiness for resources are (energy) values between 0 and 127, as opposed to appearances where the number of differing bits determines pickiness.

Agent type: Effectors

The effector uses the state from one or two other agents, to determine its own state. It interprets the state false as –1 and the state true as +1, to which it applies a weight.



Two inputs One input

The effectors who use the state from two other agents applies the weight and then adds the two values, to determine its own state (true if sum >= 0, false otherwise). The other kind of effector simply copies or negates the state of the observed agent.

An effector doesn't bid for state information, it simply buys it when it is present. Effectors who takes two inputs, waits until both inputs are present before they buy the state information. The calculated bid (i.e. price in this case since they always buy it) is divided among the two agents bought from. If input 1 is present, several cycles may occur before input 2 is present, in which case the effector simply waits. If input 1 changes in the same cycle as input 2 is ready, the new input 1 is used. When all input is available it pays the fitness points, determines its own state and announce an auction. Note that the state of an effector is *inherently different* from the state of inpoders since the state of inpoders can be sold many times, whereas the state of an effector can only be sold once (to the highest bidder). The state of an effector is treated as a 1 bit message.

The DNA structure for an effector is:

Effector DNA

Effector feature	**# of bits**
(Agent features)	56
Has two inputs	1
Ideal appearance (input1)	10
Ideal appearance (input2)	10
Pickiness_appear_input1	4
Pickiness_appear_input2	4
Total	85

The state of an effector can be treated as an action towards the environment, e.g. a certain phoneme is present or not, or it can be interpreted as a message for further processing in the autopoietic network. The case where the state of the effector is treated as a message for further processing in the network, corresponds to information processing or cognition on a higher abstraction level than when the messages are coming directly from the environment. Note that the functionality on the low abstraction level is the same as the funcionality on higher levels, as illustrated on figure 1, the difference is the type of resources that is processed.

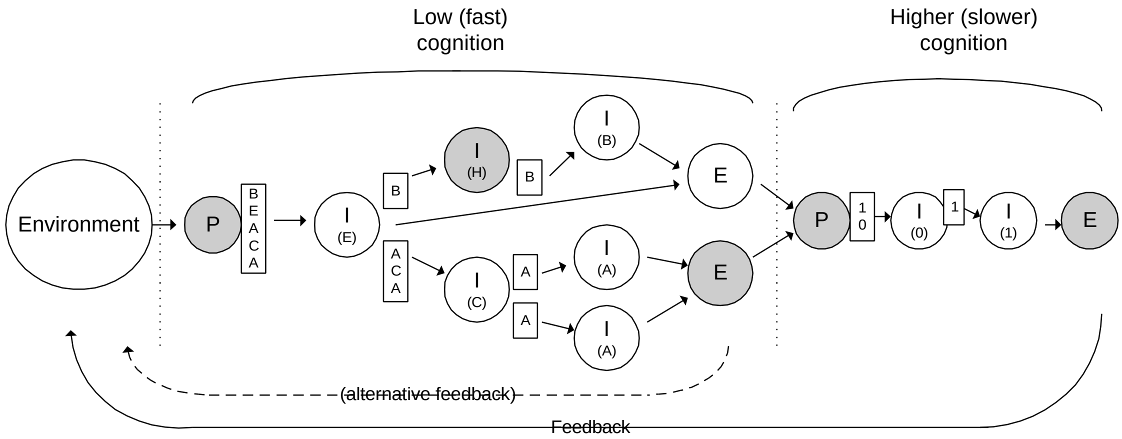


Figure 1 Simplified illustration of an autopoietic network. Grey agents has the state false, and white agents has the state true. Processed messages are positioned lower than unprocessed.

Messages or state information in the figure flows in the direction of the arrows and payment (fitness) flows in the opposite direction. Assuming that the purpose of the illustrated network is to decide whether 'BA' is present (in that order) in the incoming message, the answer 'false' from the last effector would be correct and it would thus be rewarded with what ever the initial provider paid for the message[2].

Although the environment only reward/pay the effectors, the reward propagate down in the network to those who deserve it, i.e. to the suppliers who did the necessary information processing. They are rewarded because their 'customers' now have a higher strength and therefor will pay more fitness points for future messages. In this way structures that give the correct answer will be rewarded more than structures that give incorrect answers, which means that the efficiency/ intelligence of the network increases by training it.

Memory in autopoietic networks

A small part of the network can be treated as an autopoietic network itself. If the sub network is trained the gene pool will be dominated by certain gene strings; mating these genestrings will not cause any significant change for the behaviour of the sub network, since identical parents with identical genestrings produce offspring that looks and behaves exactly like the parents (except for possible mutations), therefor the effector response of that sub network to messages from the environment (including the surrounding network) will not change over time, i.e. the sub network corresponds to memory.

(more)

---

[2] The reward should be adjusted to compensate for a growing or shrinking network. If not adjusted correctly it will cause inflation or deflation (the mechanism corresponds to buying bonds/ printing money or selling bonds/ discarding money in a growing or shrinking economy).

Next step

An implementation of an autopoietic network would be very usefull: To observe how the network reacts to certain changes in input and DNA parameters.

Statistical information about the current gene mass in various parts of the network, complemented by a 3d grid on the screen, with each kind of agent having their own color, maybe shaded according to their behavioral state (pilgrim, breeder, processor or observer), would give an excellent view of how and why the network works or don't work. Realtime representation would of course be ideal, but without explicitly exploiting the possibilities for parallel execution, only very small networks with simple input would be feasible to run realtime on a standard PC.