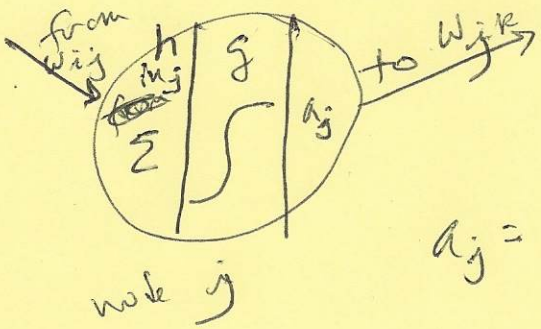
Recurrent NN nodes, W, $\vec{a}$

nodes(:).

- ~~layer~~
- to
- ~~from~~
- h
- g
- W
- a
- in_j
- del



$$a_j = g\left(\sum_{i=1}^{n} w_{ij} a_i\right)$$

$h = W a$

$$\begin{bmatrix} h_1 \\ \vdots \end{bmatrix} = \begin{bmatrix} w_{11} & w_{21} & w_{31} & \cdots & w_{n1} \\ \vdots & & & & \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \end{bmatrix}$$

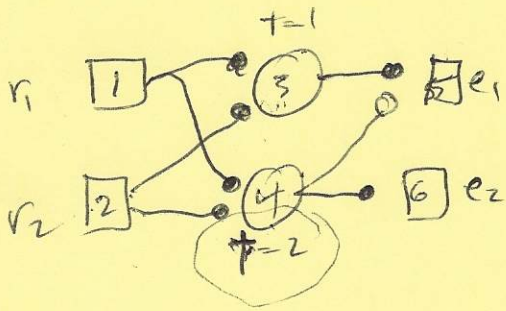
RNN step

input: ~~nodes~~, W, $\bar{a}$, ~~thresh~~

$\bar{inj} = W\bar{a}$;

for n=1:num_nodes

$\bar{a} = g(\bar{inj})$



$$W = \begin{bmatrix} 0 & 0 & p & p & 0 & 0 \\ 0 & 0 & a & b & 0 & 0 \\ d & d & 0 & 0 & 0 & 0 \\ d & d & 0 & 0 & e & a \\ 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & p & 0 & 0 \end{bmatrix}$$

$$\bar{a} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ s \end{bmatrix}$$

$$W * a = \begin{bmatrix} 0 \\ 0 \\ 1 \\ -1 \\ 0 \\ 0 \end{bmatrix}$$

$$W * (W * a) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ p \end{bmatrix}$$

function [a_out = TA_RNN_step (W, a, t)

%

a = (W * a >= t) & (W * a > 0)

---

function a_out = TA_RNN_run(W, a, t)

%

num_nodes = length(W(:,1));

Wn = W;

count = 0;

while (max(Wn(:)) > 0) & (count < num_nodes)

    count = count + 1;

    Wn = Wn * Wn;

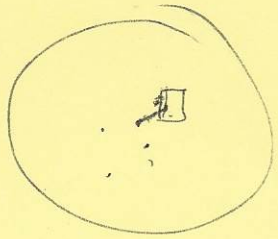
end

a_out = a;

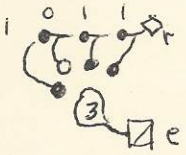
for c = 1: count

    a_out = TA_RNN_step(W, a_out, t);

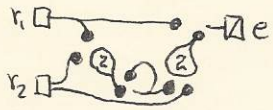
end.



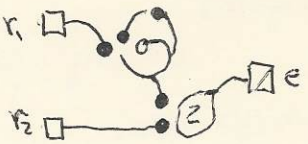
Culbertson, The Minds of Robots



1. Net to recognize a message



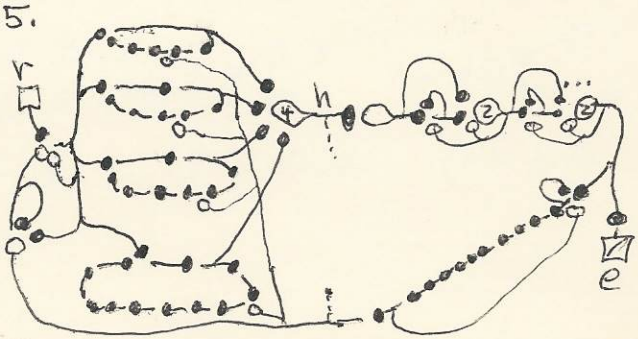
2. Conditioning net



3. Probabilistic response (after r_1 fires, $p(e) = .5$)

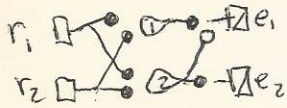
4. If r fires, the state of the cycle changes

5. Time delay: Relatively prime (left), doubling (right) : T.D. : 5 min.

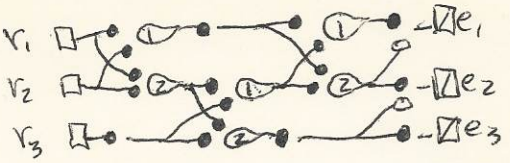


(Culb. cont.'d) 2

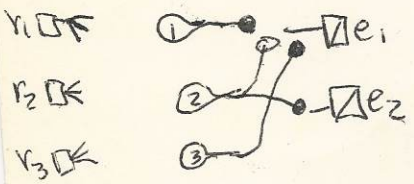
III. Counting



1. e_K fires for K receptors.



2. Same as an r_3 like 1, but simplest elements.



3. Counting in binary