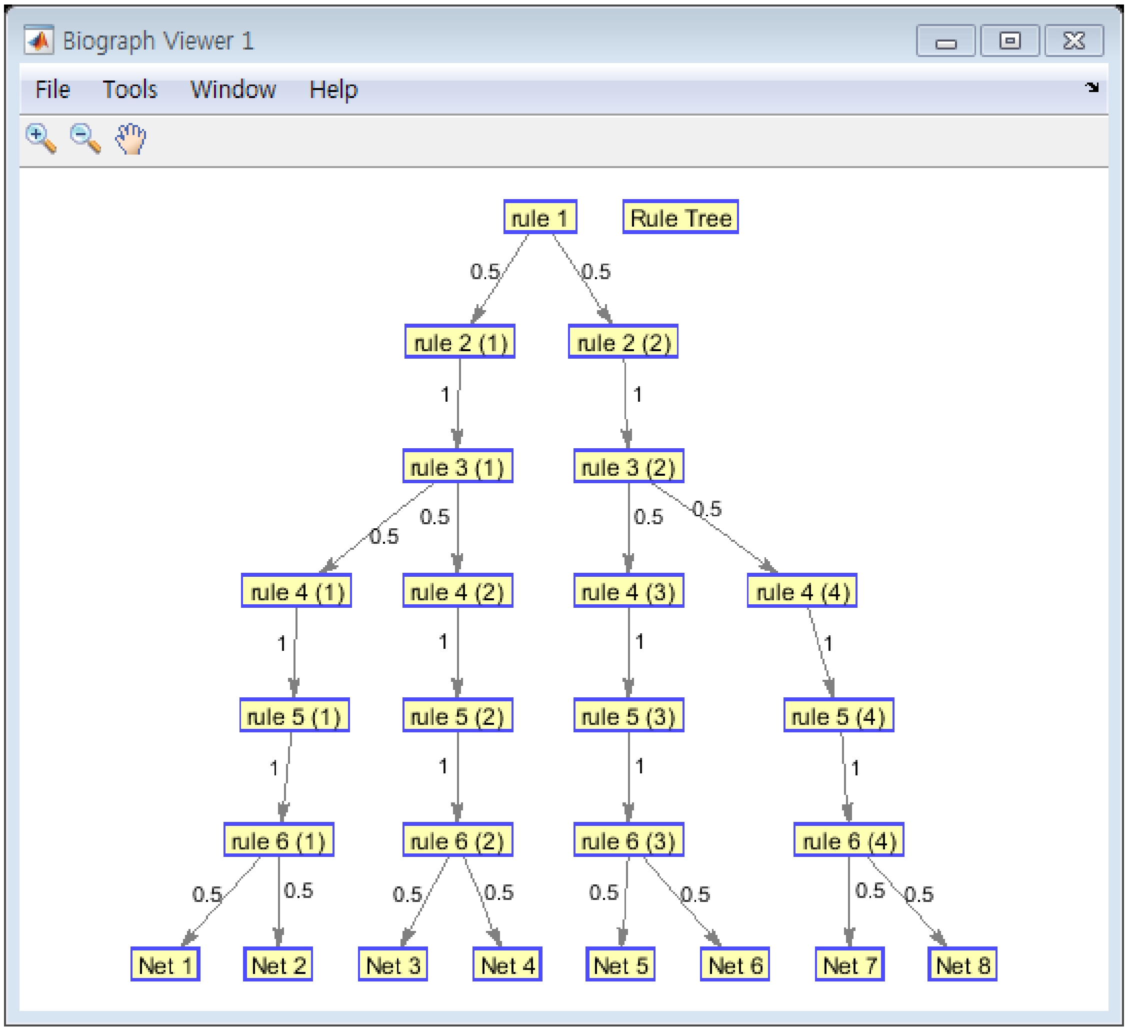
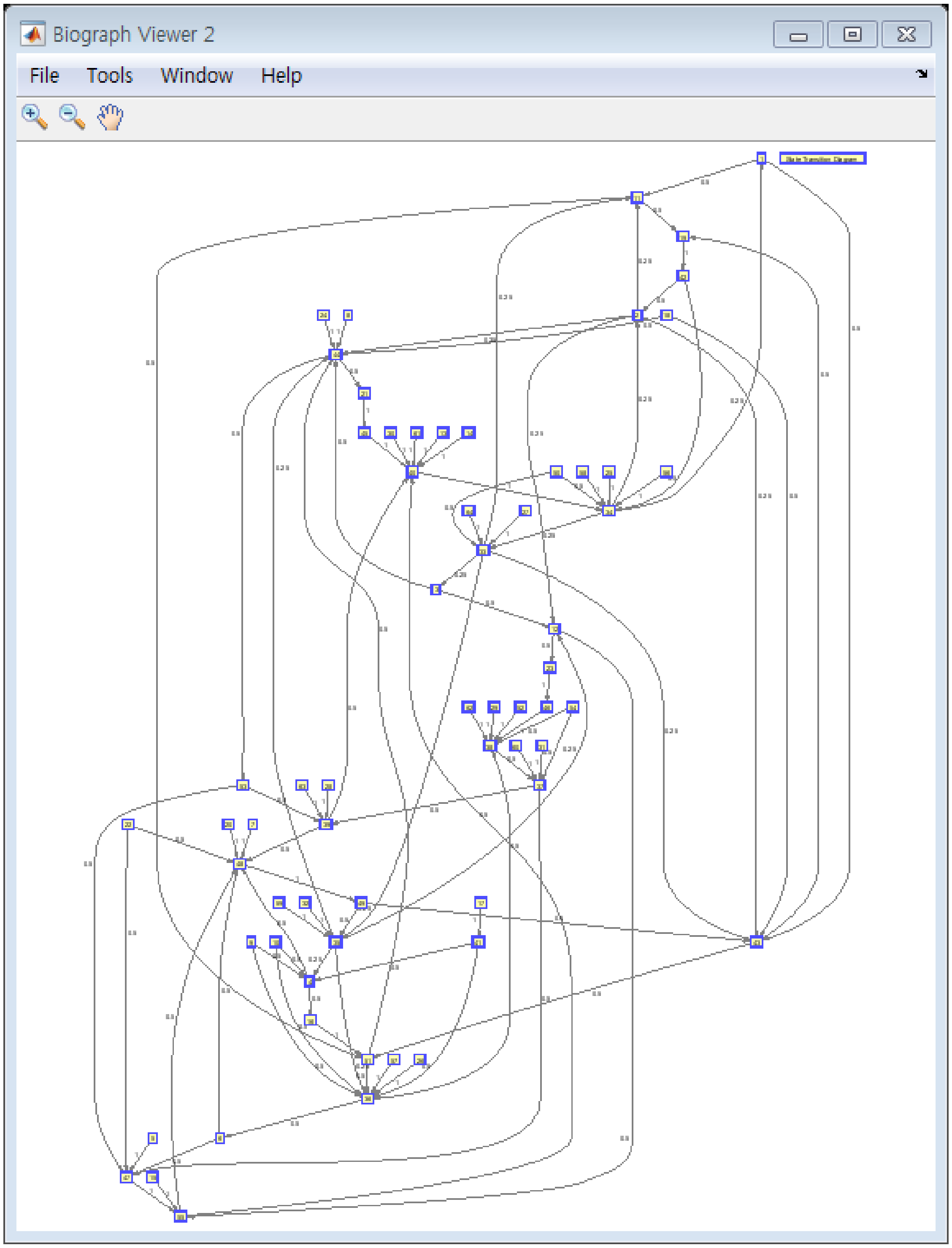
```
%Minjun Park, CAAM 210, 11/23/15
%
%Description: shows how each gene interact with each other with given
%probabilities.
%Structure: pbndriver, ruletree, genestm, d2b, b2d
%
%USAGE: assign wires, rules, probability.
%
function pbndriver
wire = [4 2 4;5 3 2;3 6 1;5 4 6;6 1 2;3 5 6]; %represents genes
rule = {[231 230], 64, [5 7], 108, 61, [62 60]}; %new rules
prob = {[1/2 1/2],1,[1/2 1/2],1,1,[1/2 1/2]}; %given probability

 [pnet,rnet] = ruletree(rule,prob);
 PSTM = zeros(2^size(wire,1)+1,2^size(wire,1)+1);
 newrule = zeros(1,numel(rule)); %preallocation
 for i = 1:numel(rnet) %one to numel(rnet)
     a = num2str(rnet(i)); %defining a
     for j = 1:numel(a)
         b = rule{j};%original rule
         newrule(j) = b(str2num(a(j))); %new rule
     end
     [STM,~] = genestm(wire,newrule); %taking STM from previous
 assignment
     PSTM = PSTM + STM*pnet(i); %forming PSTM
 end
 n = size(PSTM,1); %preallocation
 for i=1:n-1
     ids{i} = num2str(i); %name the genes
 end
 ids{n} = 'State Transition Diagram'; %title
 g = biograph(PSTM,ids,'showweights','on'); %designating each gene
 view(g)
end

function [pnet,rnet] = ruletree(rule,prob)

for i = 1:numel(rule)
    R(i) = numel(rule{i});
end

numnodes = 1;
Ntree = 1; %preallocation
for i = 1:numel(rule)
    numnodes = R(i) * numnodes; %number of nodes
    Ntree = Ntree + numnodes;
end

ln = cumprod(R); %keeping tract of layers
l = numel(R); %number of layers
num = 1; %initialization
for i = 1:l
```

```
    layer{i} = [num+1 : num+ln(i)]; %moving to different layers
    num = num + ln(i); %added num
end

ptree = zeros(Ntree + 1);
ptree(1,2:1+R(1)) = prob{1}; %assigning probability
for i = 1:(l-1)
    t = 0; %preallocation
    for j = 1:ln(i)
        ptree(layer{i}(j),layer{i+1}(1+R(i+1)*t):layer{i+1}(R(i
+1)*(1+t))) = prob{i+1}; % making ptree
        t = t + 1;
    end
end
%{this nested for-loops create ptree}

ids{1} = 'rule 1'; %preallocation
count = 1;
for i = 2:numel(layer)
    count2 = 1;
    for k = 1:ln(i-1)
        ids{count + k} = ['rule ' num2str(i) ' ('
 num2str(count2) ')']; %assigning name for each rule
        count2 = count2 + 1;
    end
    count = count + ln(i-1);
end
count3 = 1;
for i = 0:prod(R)-1 %from 0 to prod(R)-1
    ids{Ntree-prod(R)+1+i} =  ['Net ' num2str(count3)]; %assigning net
 number
    count3 = count3 + 1;
end
ids{size(ptree,2)} = 'Rule Tree'; %title

g = biograph(ptree,ids,'showweights','on');%bigraph method
view(g)

stepprob{1} = prob{1};
for i = 2:numel(layer)
    stepprob{i} = []; %getting step probabilities
    for k = 1 : numel(layer{i-1}) %from 1 to number of layers
        stepprob{i} = [stepprob{i} stepprob{i-1}(k)*prob{i}];
    end %this shows different probabilities of genes
end
pnet = stepprob{end};

catprod = [1 cumprod(R)];
steprnet{1} = 0; %preallocation
for i = 2:numel(layer)+1 %2 to number of layers
    steprnet{i} = []; %getting next rnet
    for k = 1: catprod(i-1)
        for j = 1 : numel(rule{i-1})
            steprnet{i} = [steprnet{i} 10*steprnet{i-1}(k)+j];
        end
    end
end
%{this shows how rnet is formed}
rnet = steprnet{end};

end


%Function name: genestm
%This function takes in wire and rule vector to produce State
 Transition
%Matrix and Rule Matrix.
%Example: genestm([4 2 3; 5 3 2; 3 6 1; 5 4 6; 6 1 2; 3 5 6],[231; 64;
 5;
%108; 61; 62])

function [STM,rulemat] = genestm(wire,rule)

%preallocate
rulemat = zeros(length(rule),2^size(wire,2));
STM = zeros(2^size(wire,1)+1,2^size(wire,1)+1);

for i = 1:length(rule)
    rulemat(i,:) = d2b(rule(i),2^size(wire,2)); %create rulematrix
end

%preallocate
n = size(wire,1);

%create STM
for i=1:2^n
    s = d2b(i-1,n); %decimal to binary
    ns = zeros(length(s),1);
    for k=1:n
        ns(k) = rulemat(k,size(rulemat,2)-b2d(s(wire(k,:))));
        %next state (using s, wire and rulematrix)
    end
    j = b2d(ns); %binary to decimal
    STM(i,j+1) = 1;
end
end

%Function name: d2b
%This function takes in decimal values and number of bits in order to
%change the decimal values to binary values.
%Example; d2b(14,8)

function b = d2b(r,C) %decimal value to binary

%preallocate
b = zeros(1, C);
while r > 0 %condition
    %use log2 to figure out the extent to which we floor
    b(C-floor(log2(r)))=1;
    r = r - 2^(floor(log2(r)));
end
end

%Function name: b2d
%This function takes in binary vector and produces its decimal
 version.
%Example: b2d([1 0 0 0 1 1 0 1])

function d = b2d(bin)

%preallocate
d = 0;
for i = 0:numel(bin)-1
    d = d + bin(numel(bin)-i)*2^i; %multiply 2^i correspondingly
end
end
```

Warning: Self connecting nodes are not allowed, ignoring the diagonal of CM.





Published with MATLAB® R2015b