Announcements

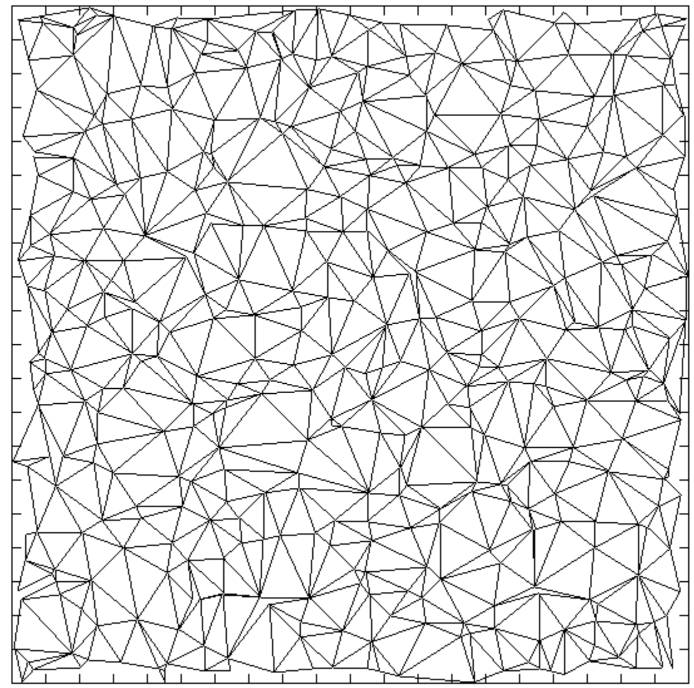
- P6 due today at 11pm
- Final exam: 12/10 (Fri) 9am at Barton West (indoor field)
- Please fill out course evaluation on-line (hosted by College of Engineering, see "Exercise15")
- Please fill out evaluation on iRobot Create Simulator on CMS. It is worth 1 project point (to make up for any lost project point)!
- Regular office/consulting hours end today. "Study Break" hours start next week.
- Review Session: 12/8 Wednesday 1-2:30pm, UP B17
- Pick up any paper from consultants (prelim, regrade results) during consulting hours. Everything will be shredded afterwards.
- Read announcements on course website!

- Previous Lecture:
  - Efficiency
  - Recursion

- Today's Lecture:
  - Recursion review
  - A model to quantify importance: Google "Page Rank"

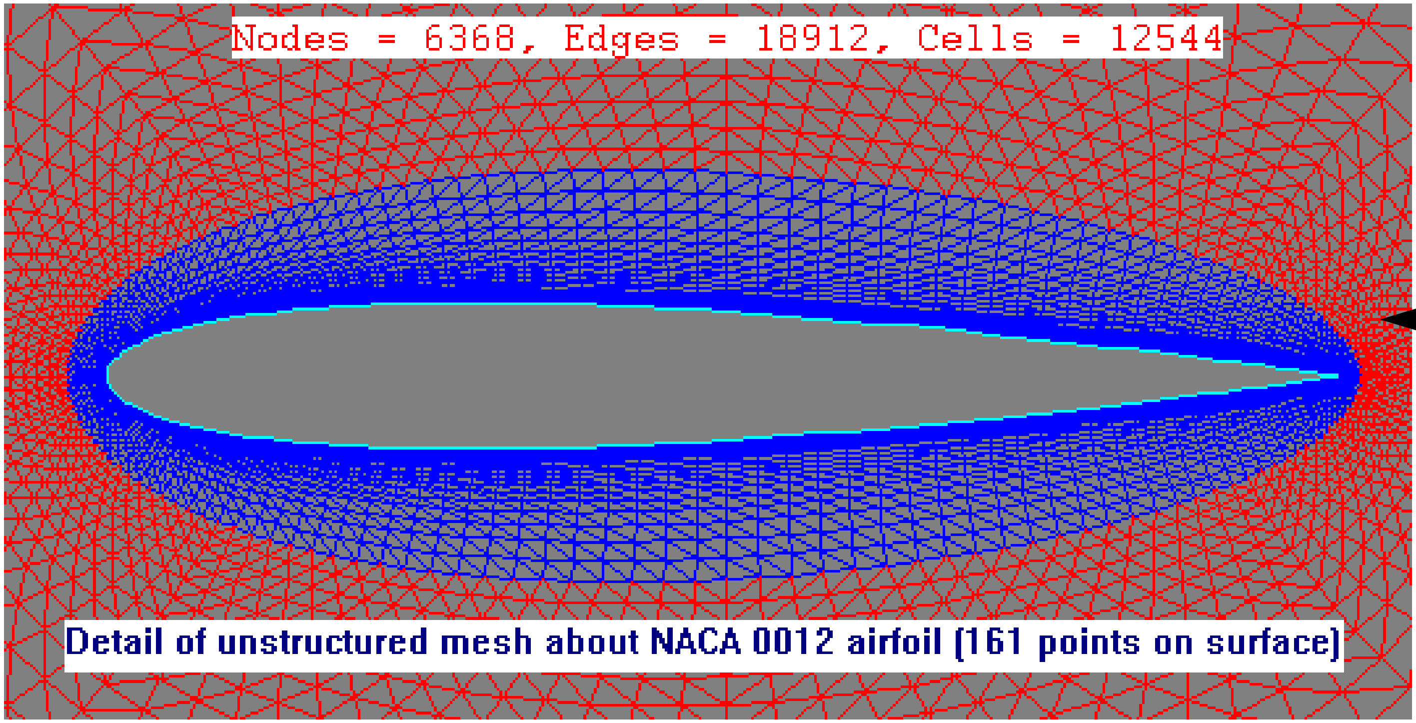
Divide-and-conquer methods also show up in geometric situations

Chop a region up into triangles with smaller triangles in "areas of interest"



Recursive mesh generation

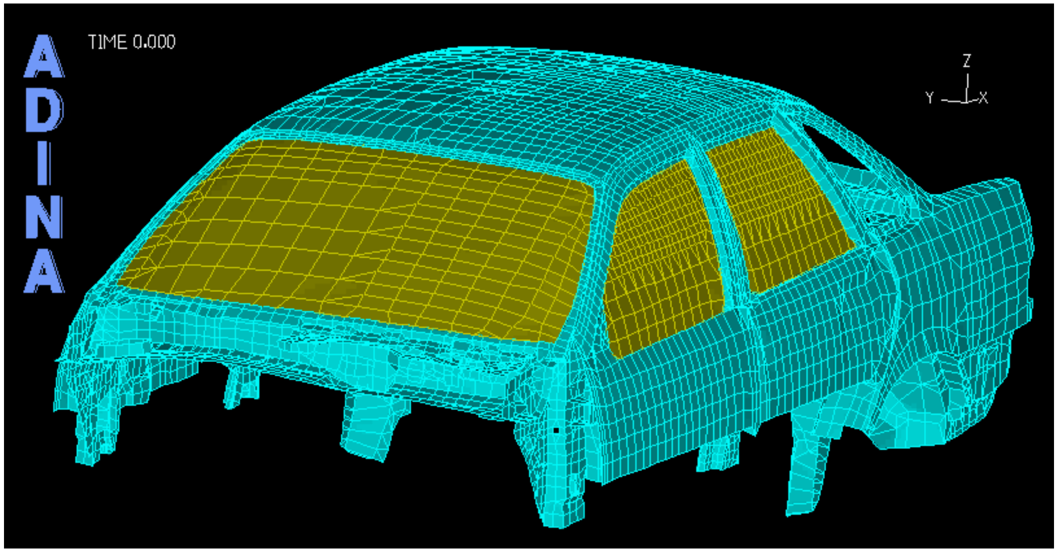
Mesh Generation



An area of interest

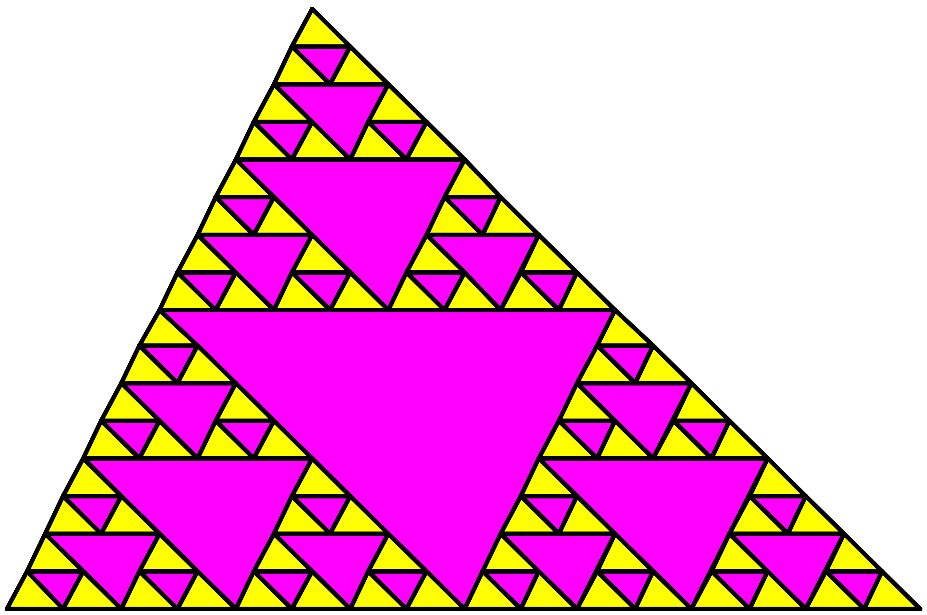
Step one in simulating flow around an airfoil is to generate a mesh and (say) estimate velocity at each mesh point.

Mesh Generation in 3D

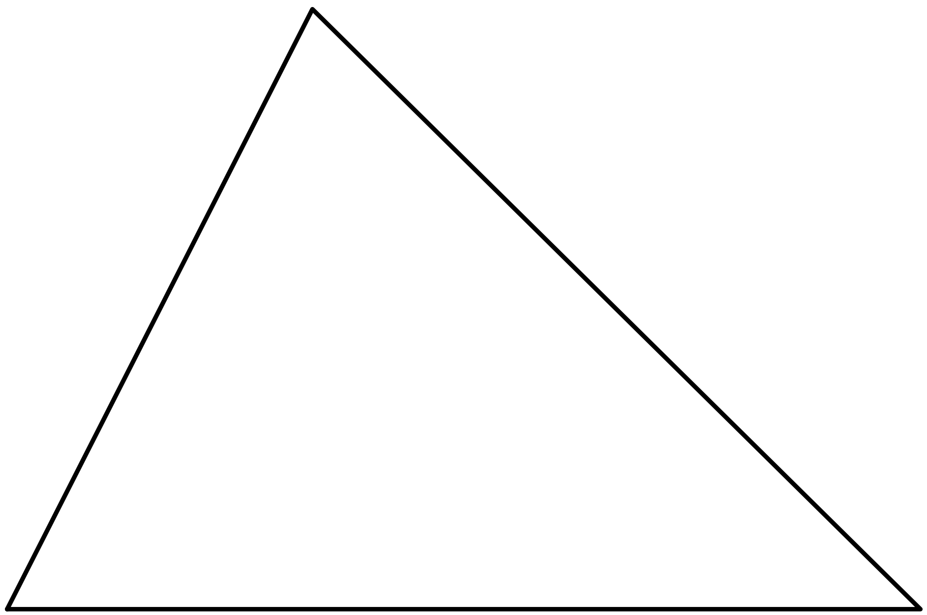


Why is mesh generation a divide-&-conquer process?

Let's draw this graphic

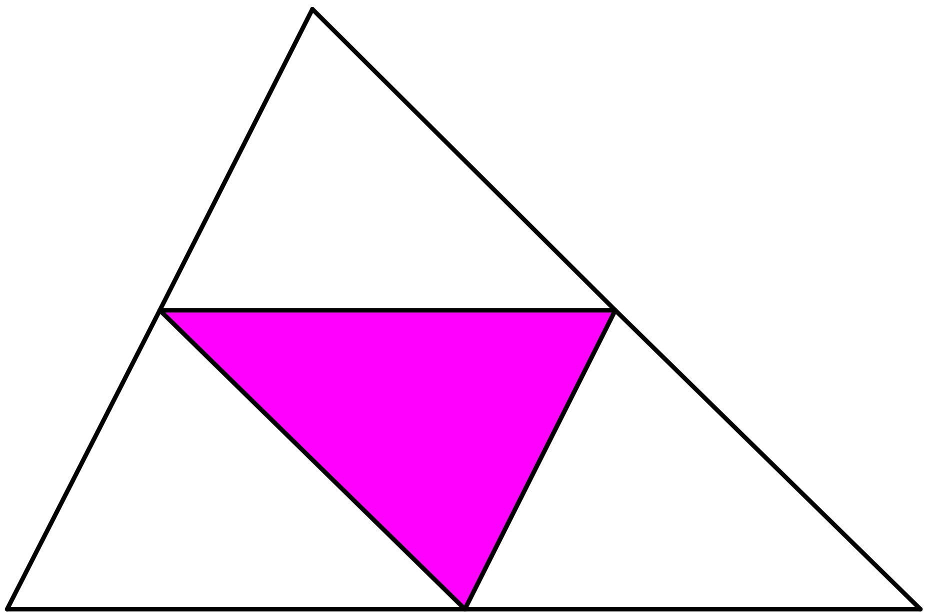


Start with a triangle



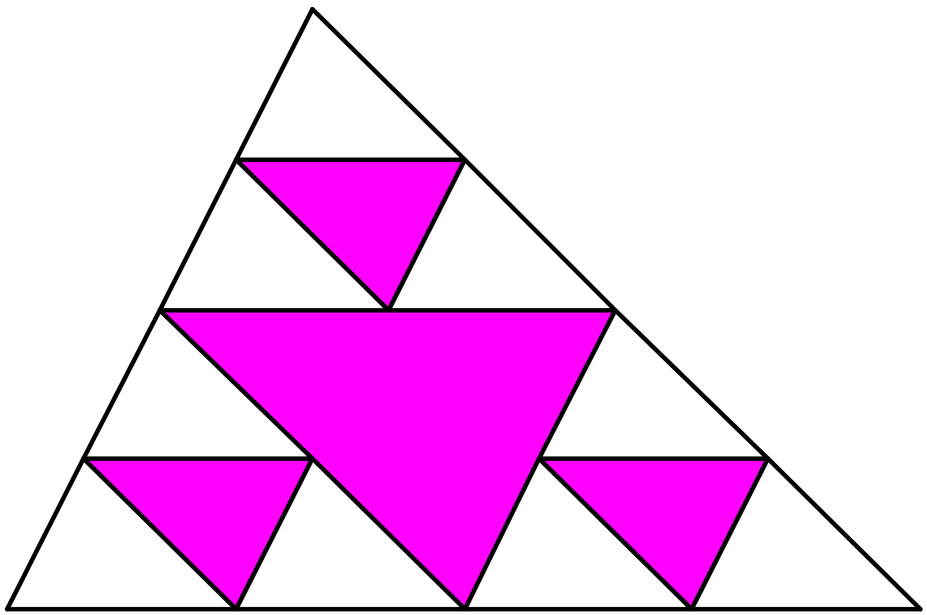
A "level-1" partition of the triangle

(obtained by connecting the midpoints of the sides of the original triangle)

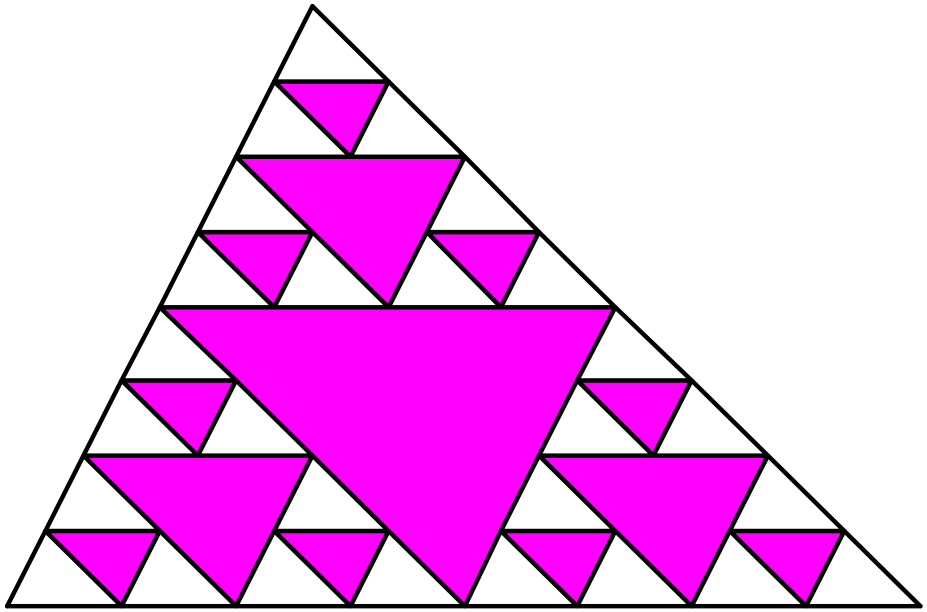


Now do the same partitioning (connecting midpts) on each corner (white) triangle to obtain the "level-2" partitioning

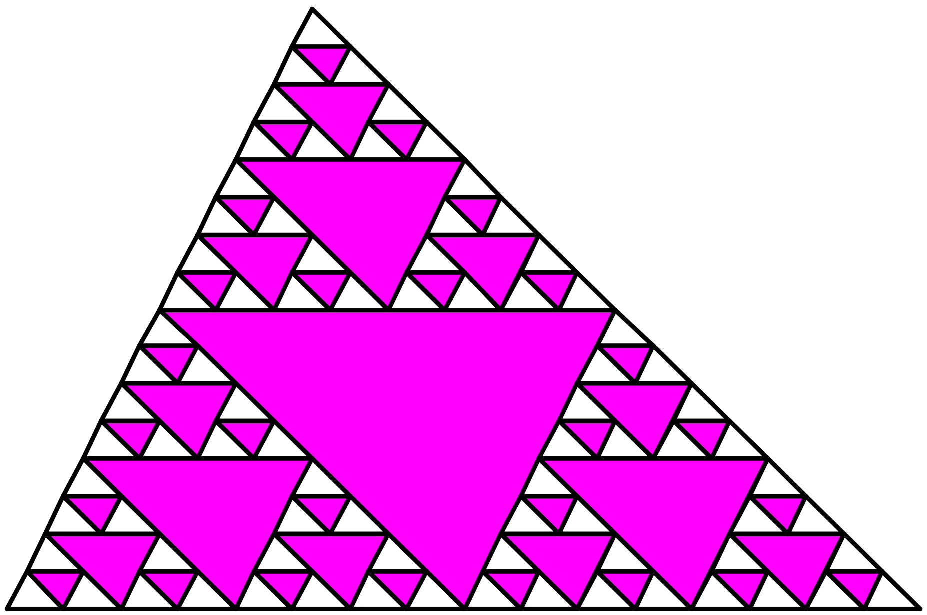
The “level-2” partition of the triangle



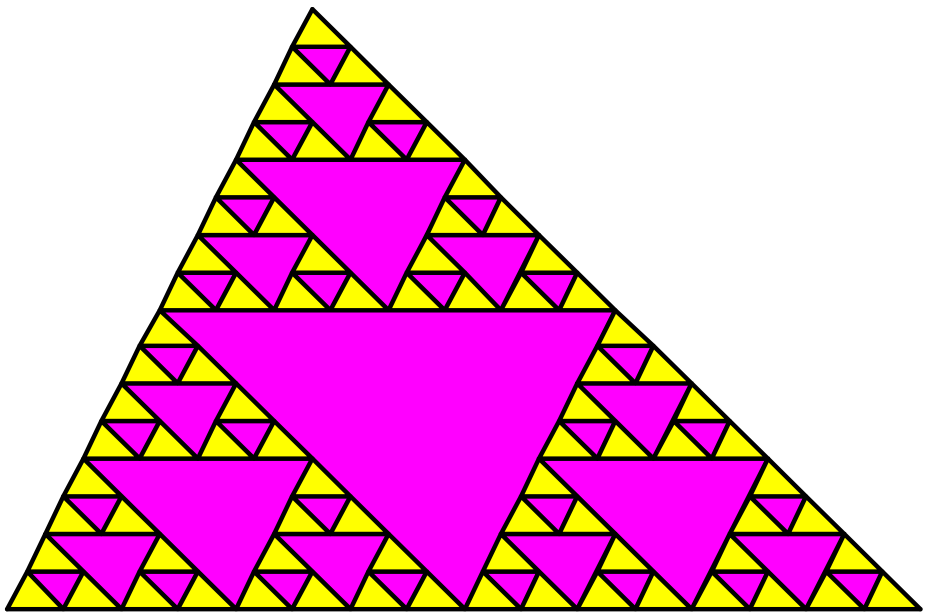
The “level-3” partition of the triangle



The “level-4” partition of the triangle



The "level-4" partition of the triangle

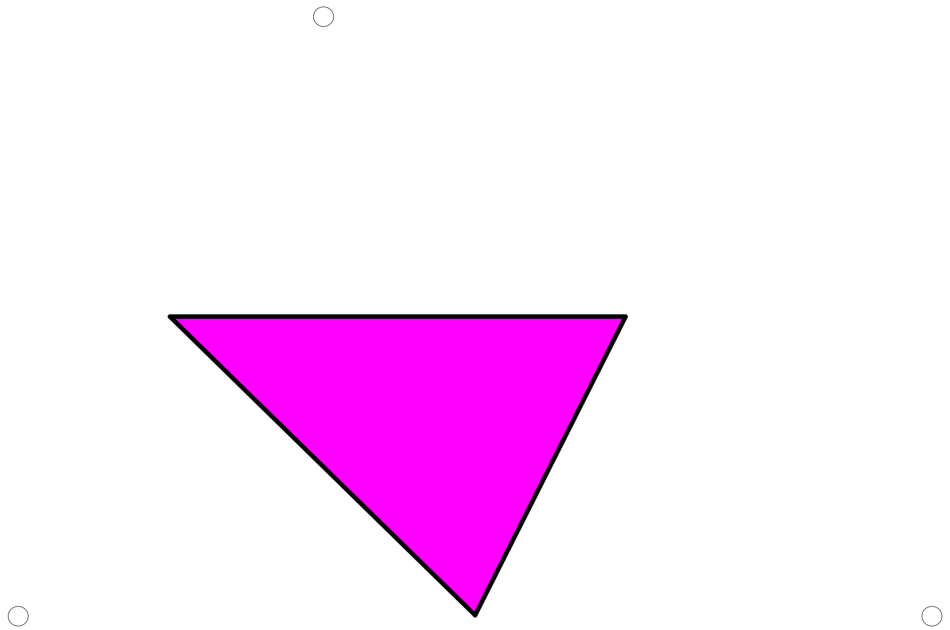


The basic operation at each level

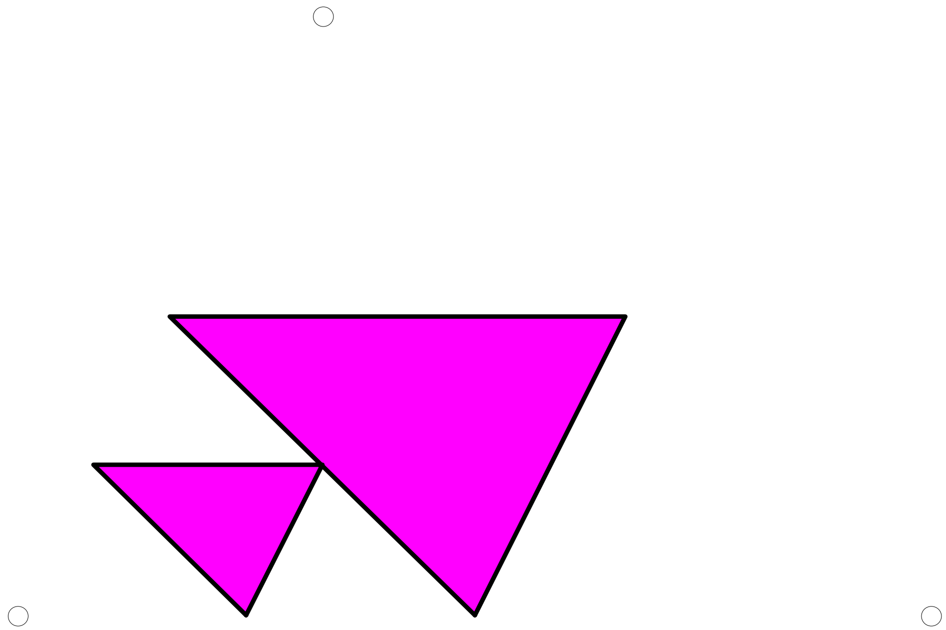
```
if    the triangle is small
      Don't subdivide and just color it yellow.
else
      Subdivide:
      Connect the side midpoints;
      color the interior triangle magenta;
      apply same process to each outer triangle.
end
```

Draw a level-4 partition of the triangle with these vertices

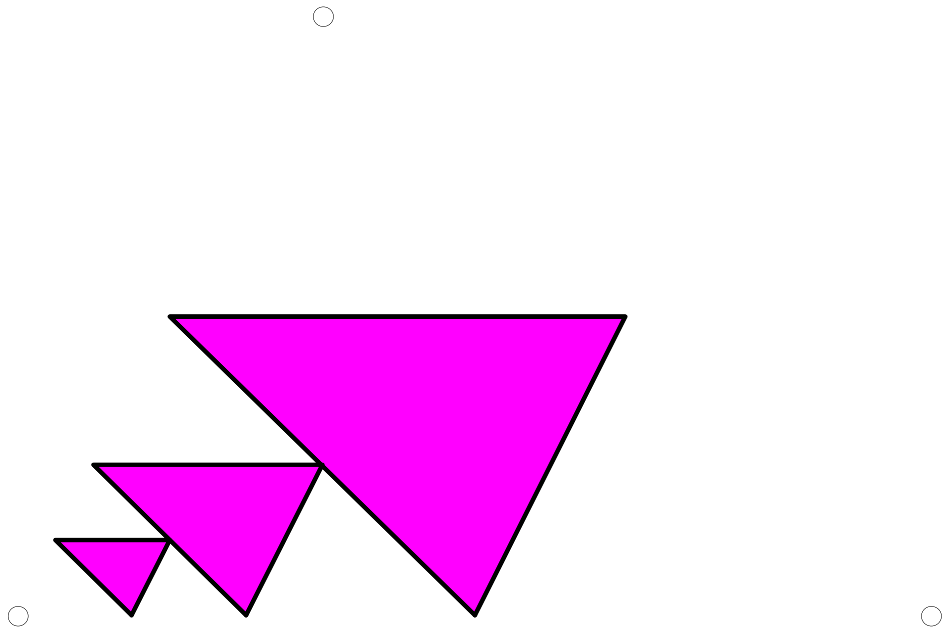
At the start…



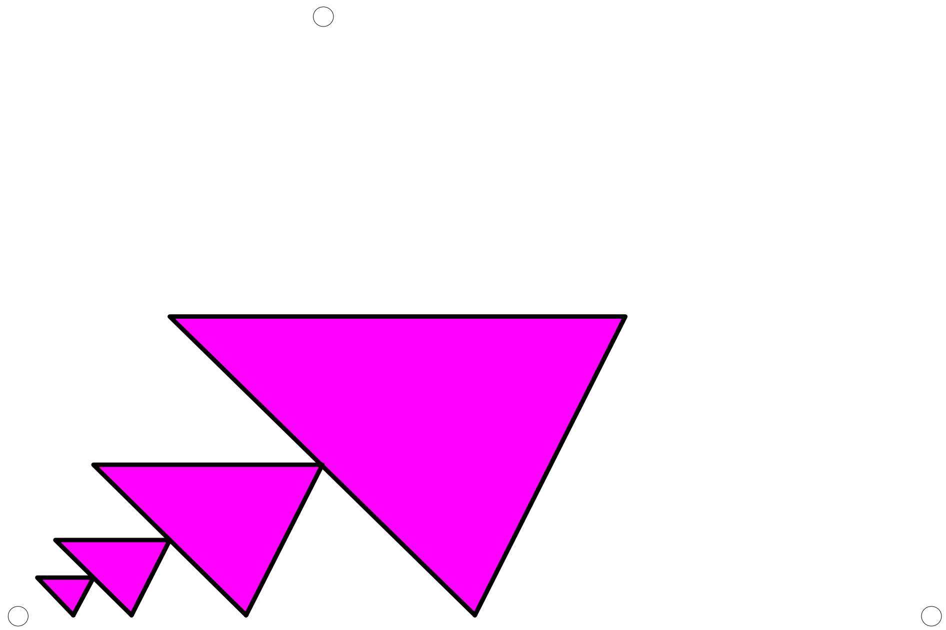
Recur: apply the same process on the lower left triangle



Recur again

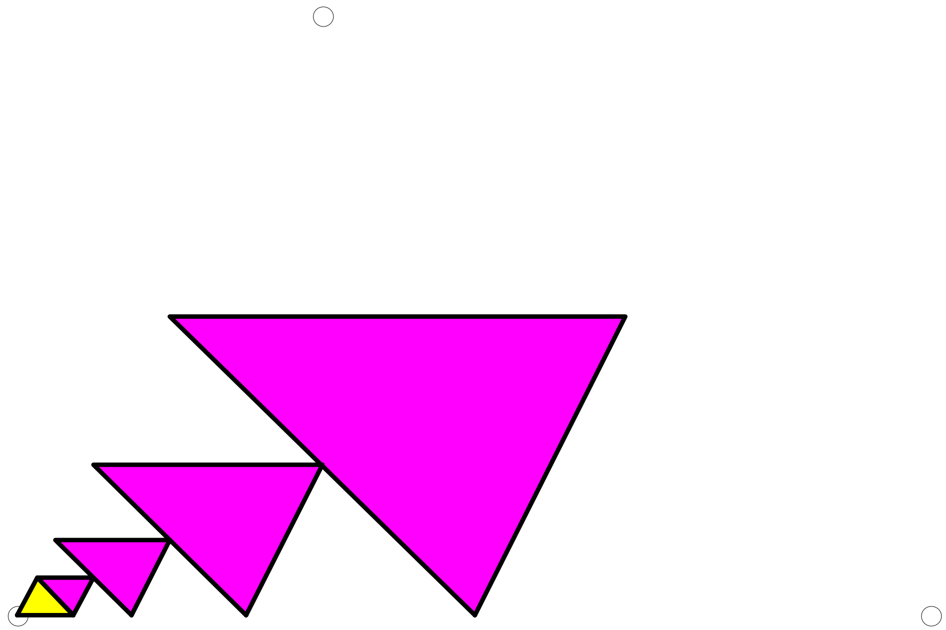


… and again

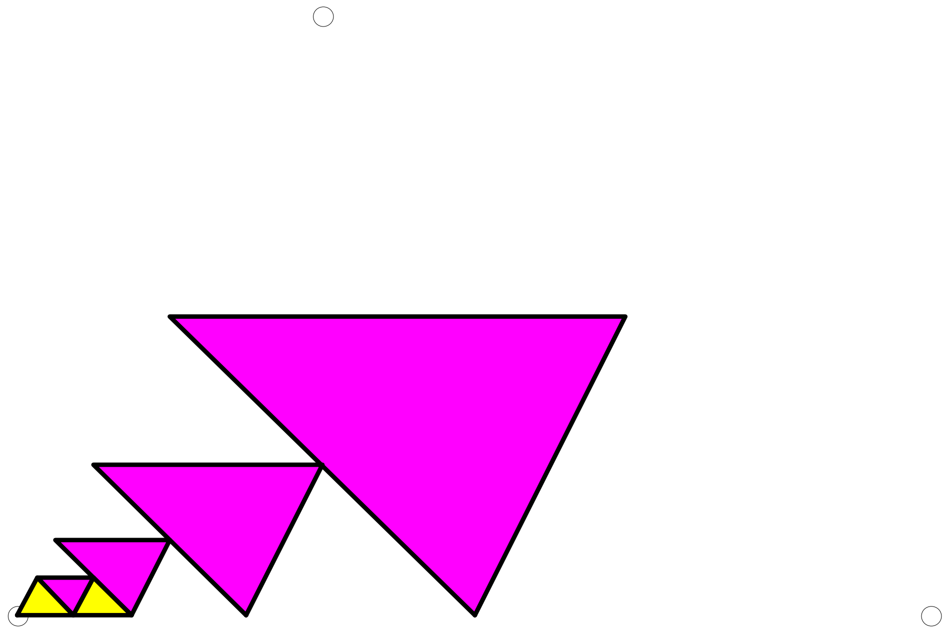


The next lower left corner triangle (white) is small—no more subdivision and just color it yellow.

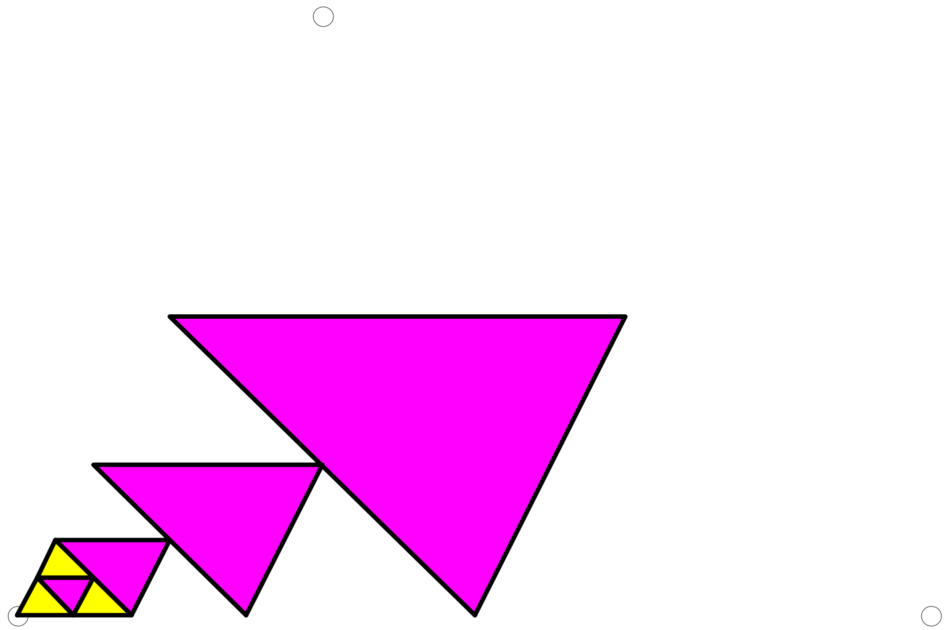
Now lower left corner triangle of the "level-4" partition is done. Continue with another corner triangle

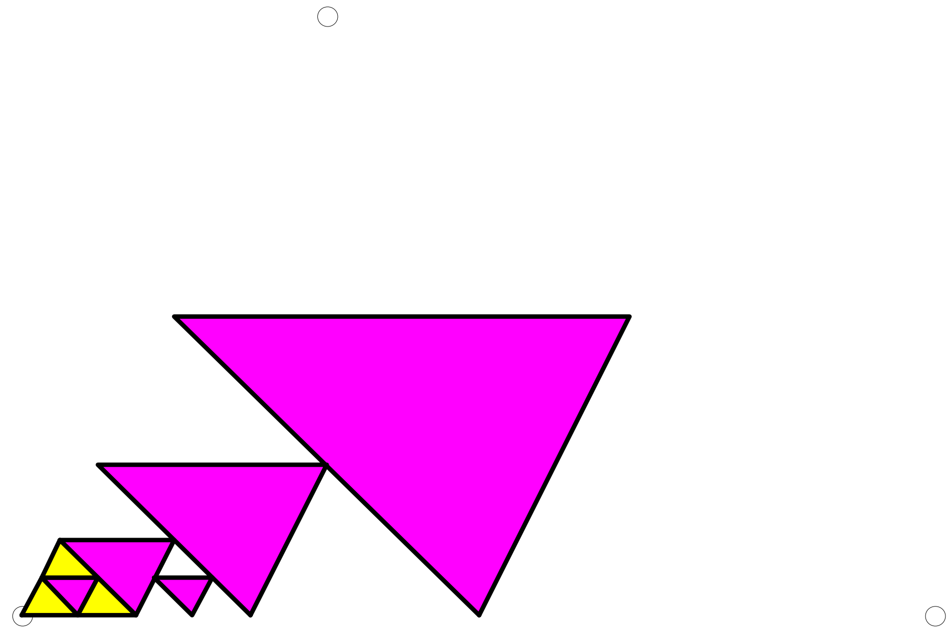


… and continue

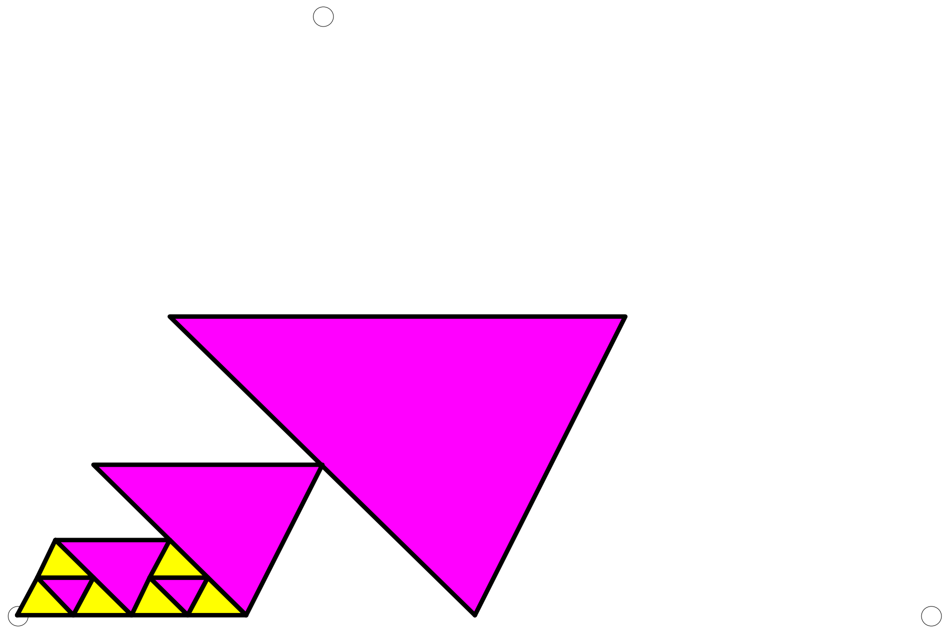


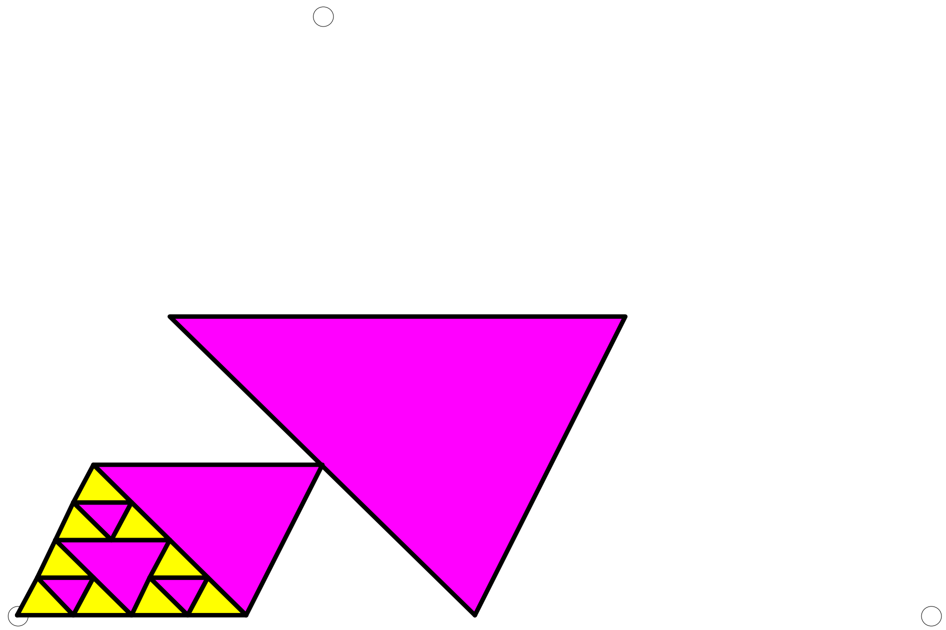
Now the lower left corner triangle of the "level-3" partition is done. Continue with another corner triangle…

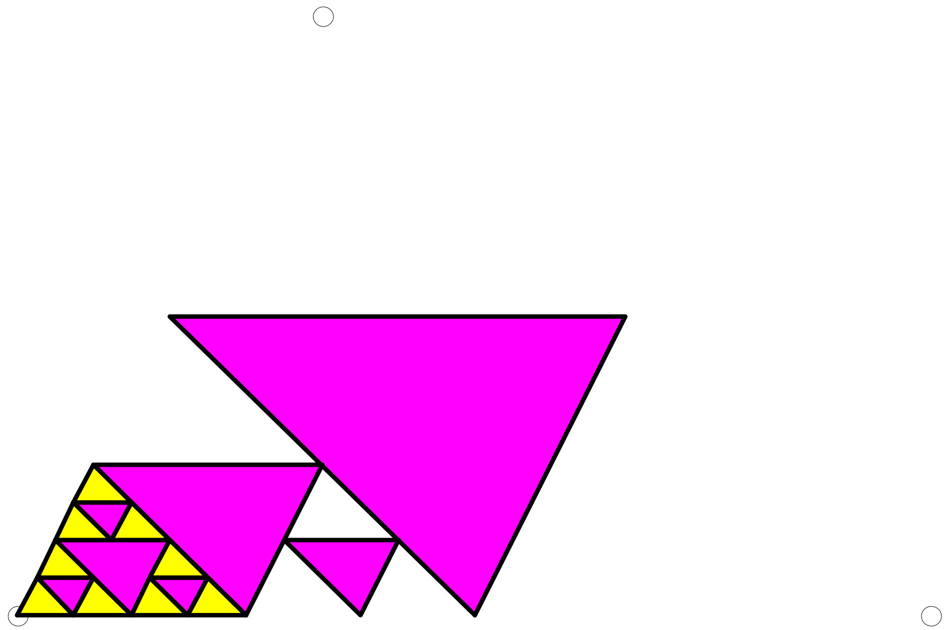


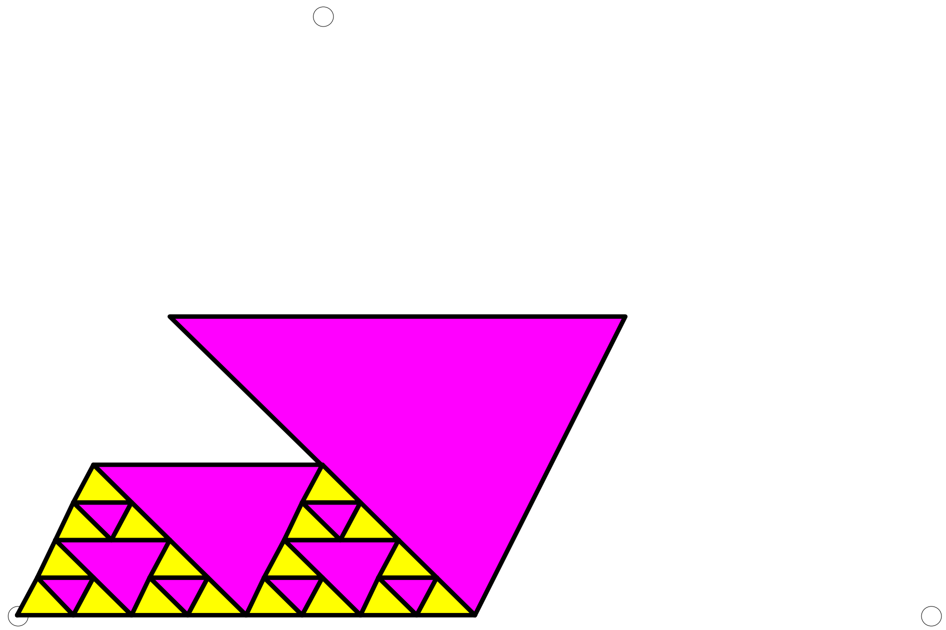


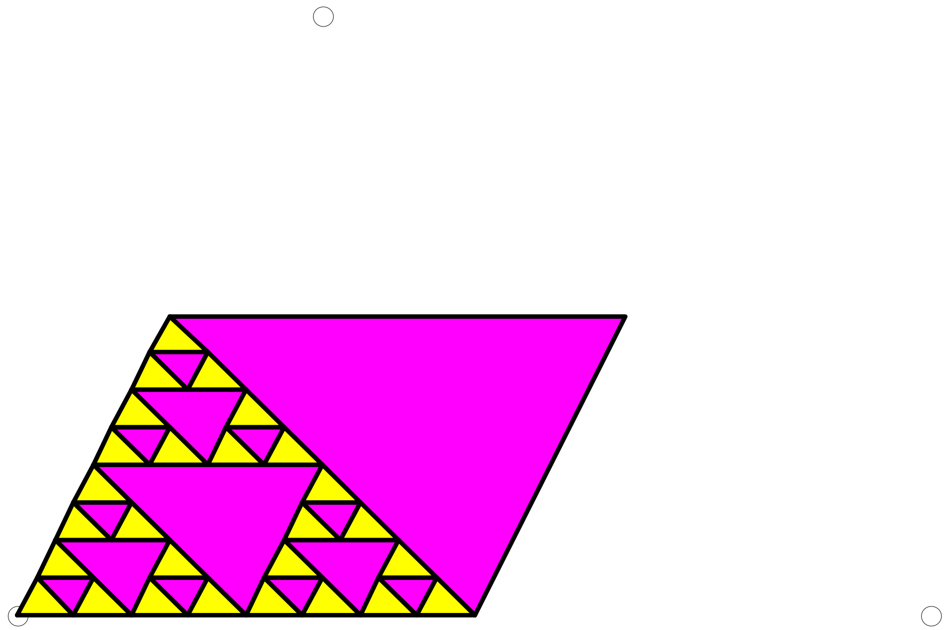
We're "climbing our way out" of the deepest level of partitioning

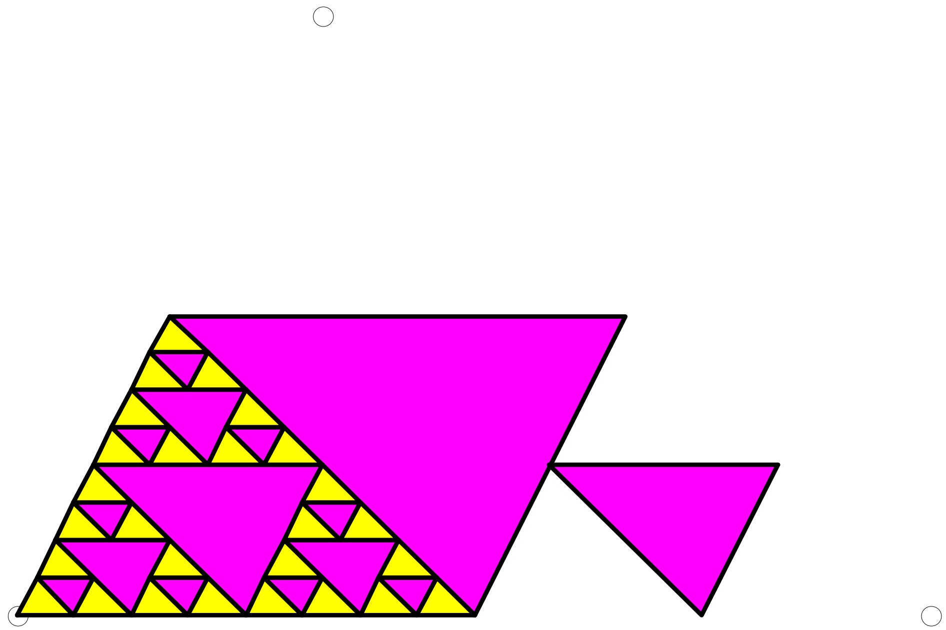


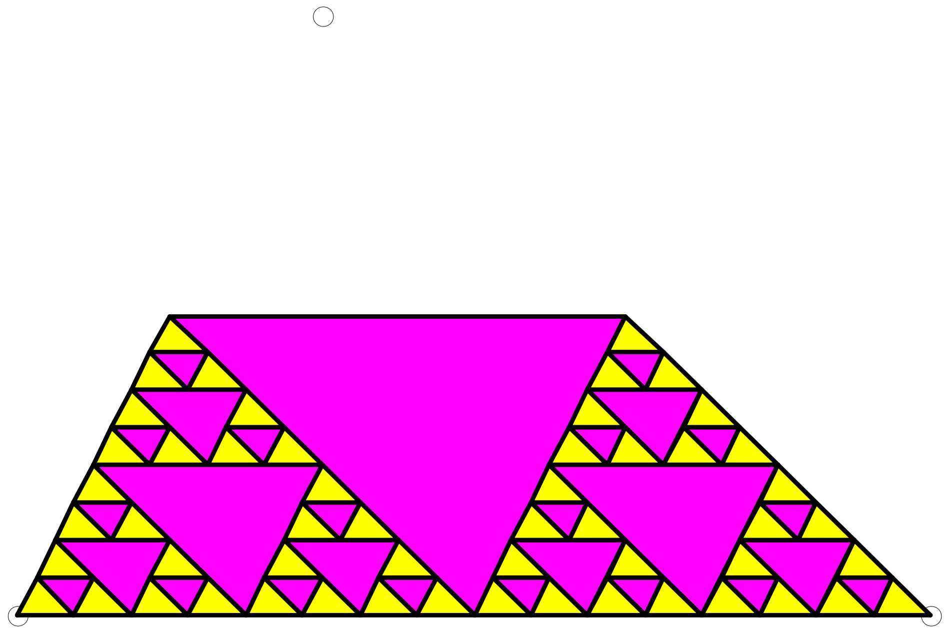


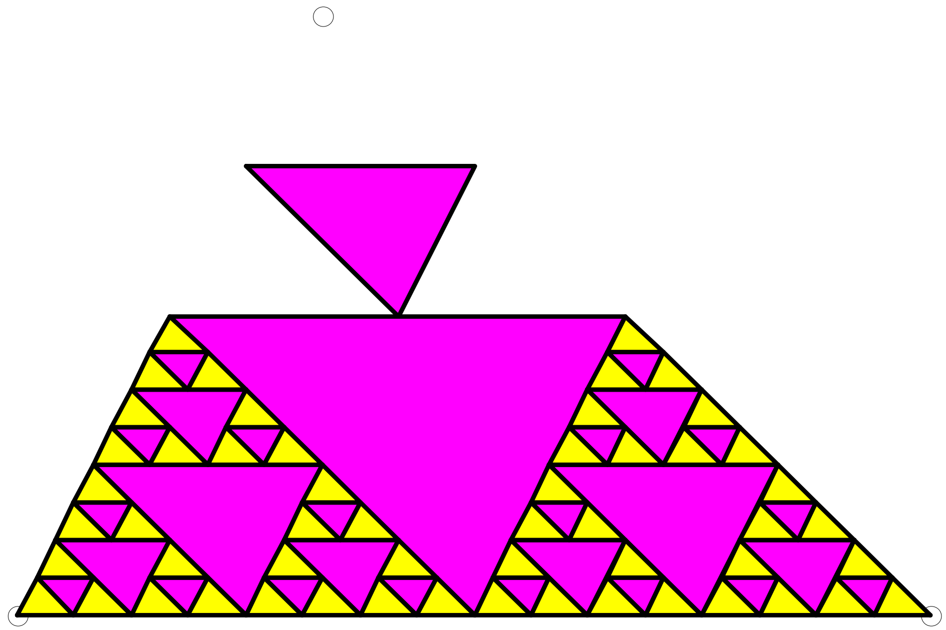




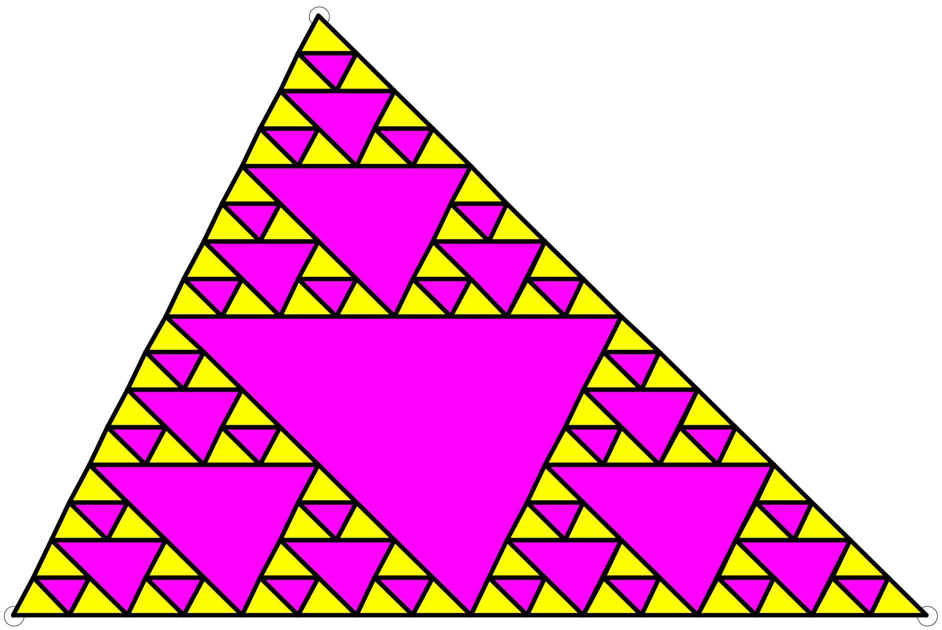








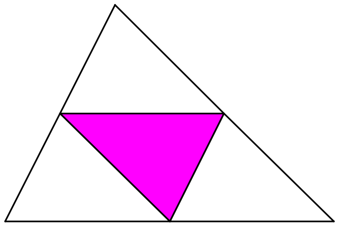
Eventually climb all the way out to get the final result



The basic operation at each level

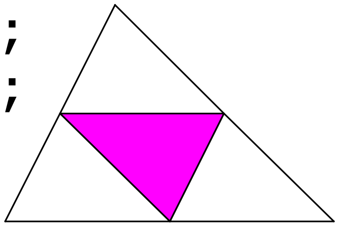
```
if    the triangle is small
      Don't subdivide and just color it yellow.
else
      Subdivide:
      Connect the side midpoints;
      color the interior triangle magenta;
      Apply same process to each outer triangle.
end
```

```
function MeshTriangle(x,y,L)
% x,y are 3-vectors that define the vertices of a triangle.
% Draw level-L partitioning.  Assume hold is on.

if L==0
  % Recursion limit reached; no more subdivision required.
    fill(x,y,'y')  % Color this triangle yellow

else
  % Need to subdivide:  determine the side midpoints; connect
  % midpts to get “interior triangle”; color it magenta.
```

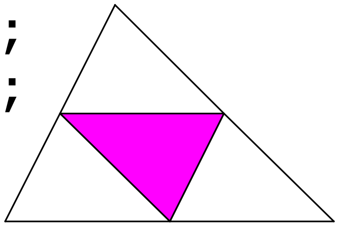


```
  % Apply the process to the three "corner" triangles...



end
```

```
function MeshTriangle(x,y,L)
% x,y are 3-vectors that define the vertices of a triangle.
% Draw level-L partitioning.  Assume hold is on.

if L==0
  % Recursion limit reached; no more subdivision required.
    fill(x,y,'y')  % Color this triangle yellow

else
  % Need to subdivide:  determine the side midpoints; connect
  % midpts to get “interior triangle”; color it magenta.
    a = [(x(1)+x(2))/2 (x(2)+x(3))/2 (x(3)+x(1))/2];
    b = [(y(1)+y(2))/2 (y(2)+y(3))/2 (y(3)+y(1))/2];
    fill(a,b,'m')

  % Apply the process to the three "corner" triangles...



end
```



```
function MeshTriangle(x,y,L)
% x,y are 3-vectors that define the vertices of a triangle.
% Draw level-L partitioning.  Assume hold is on.

if L==0
  % Recursion limit reached; no more subdivision required.
    fill(x,y,'y')  % Color this triangle yellow

else
  % Need to subdivide:  determine the side midpoints; connect
  % midpts to get “interior triangle”; color it magenta.
    a = [(x(1)+x(2))/2 (x(2)+x(3))/2 (x(3)+x(1))/2];
    b = [(y(1)+y(2))/2 (y(2)+y(3))/2 (y(3)+y(1))/2];
    fill(a,b,'m')

  % Apply the process to the three "corner" triangles...
    MeshTriangle([x(1) a(1) a(3)],[y(1) b(1) b(3)],L-1)
    MeshTriangle([x(2) a(2) a(1)],[y(2) b(2) b(1)],L-1)
    MeshTriangle([x(3) a(3) a(2)],[y(3) b(3) b(2)],L-1)
end
```



Quantifying Importance

How do you rank web pages for importance given that you know the link structure of the Web, i.e., the in-links and out-links for each web page?

A related question:

How does a deleted or added link on a webpage affect its "rank"?

Background

Index all the pages on the Web from 1 to n. (n is around ten billion.)

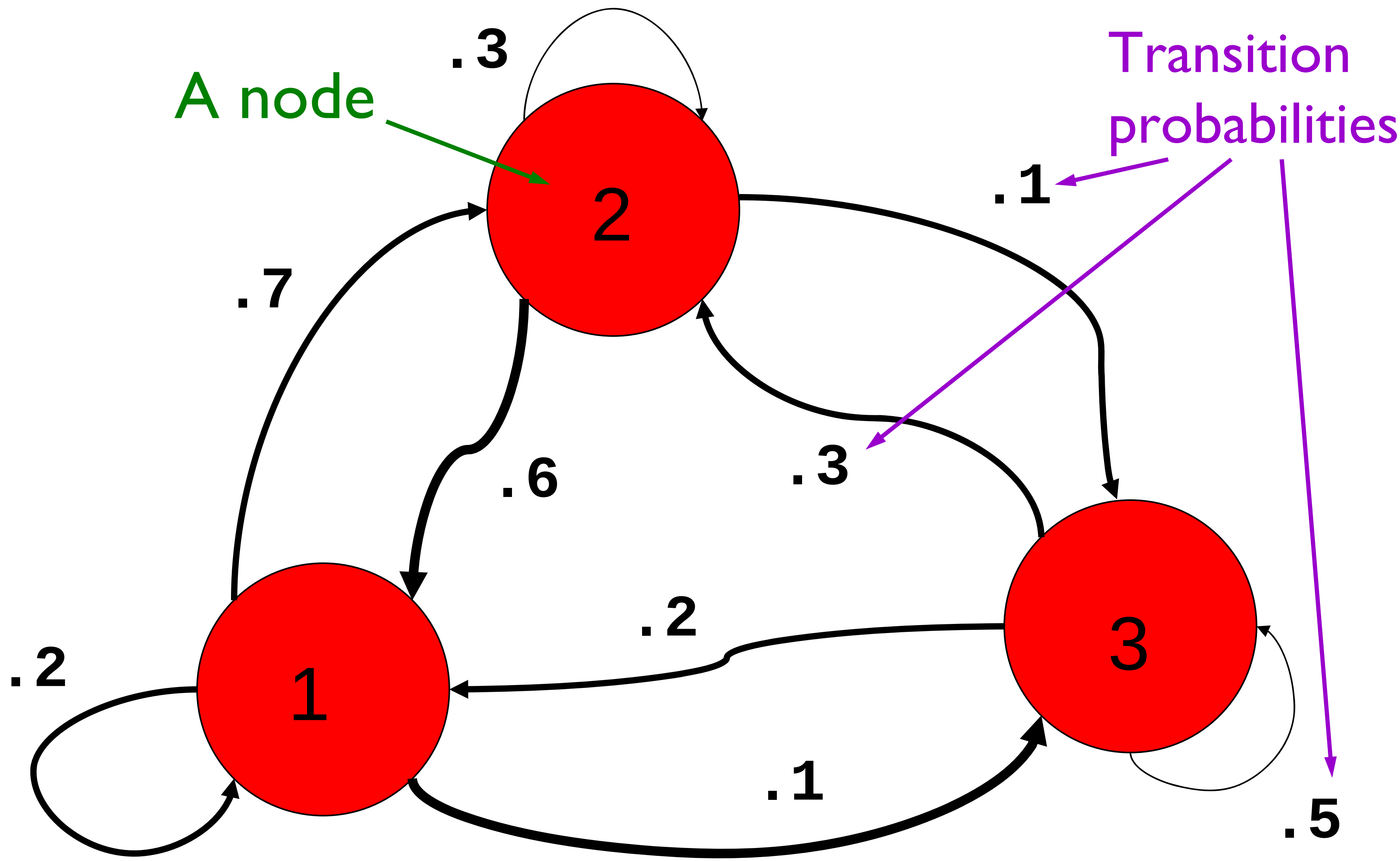
The PageRank algorithm orders these pages from "most important" to "least important."

It does this by analyzing links, not content.

Key ideas

- There is a random web surfer—a special random walk
- The surfer has some random "surfing" behavior—a transition probability matrix
- The transition probability matrix comes from the link structure of the web—a connectivity matrix
- Applying the transition probability matrix → Page Rank

A 3-node network with specified transition probabilities



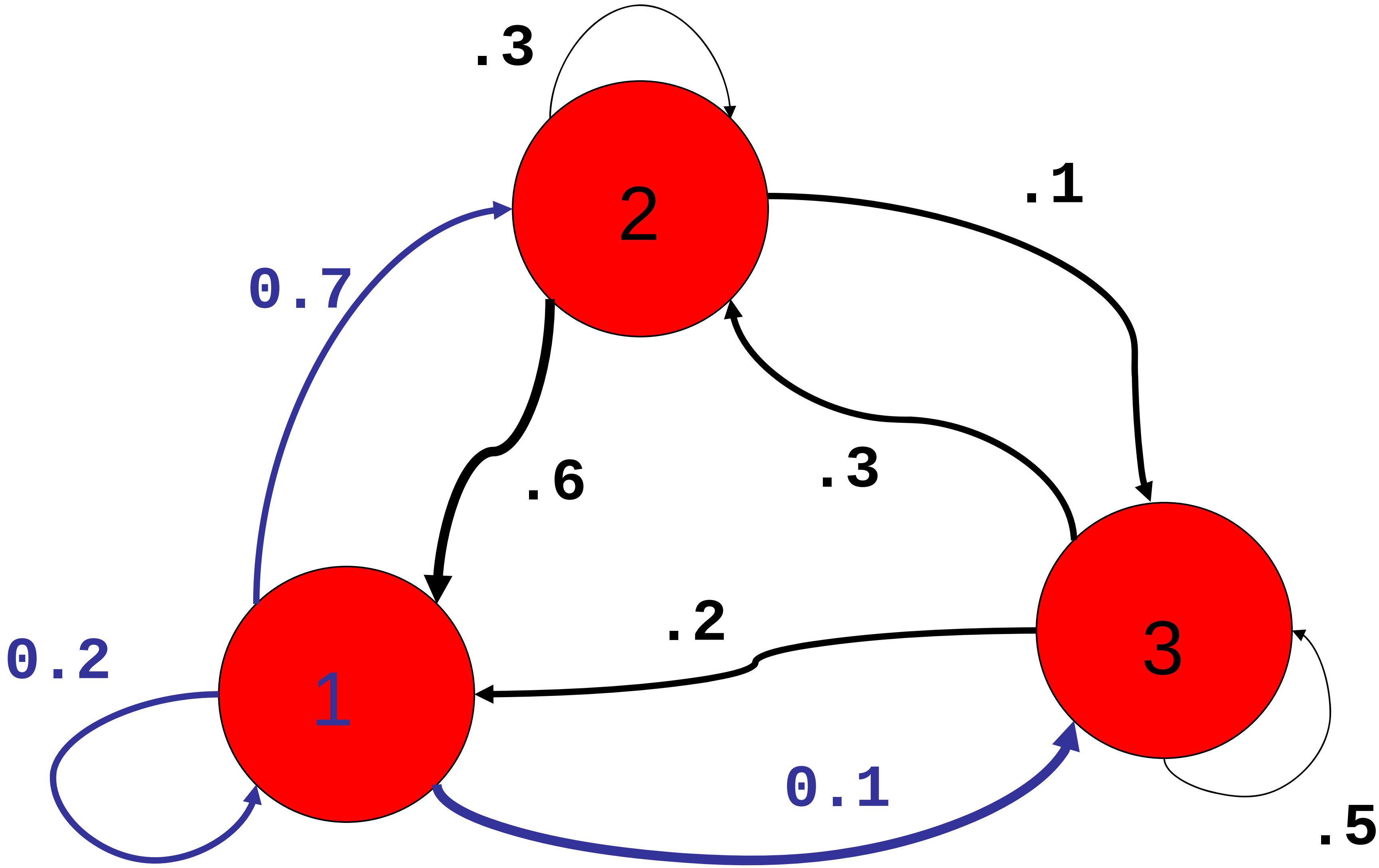
A special random walk

Suppose there are a 1000 people on each node.

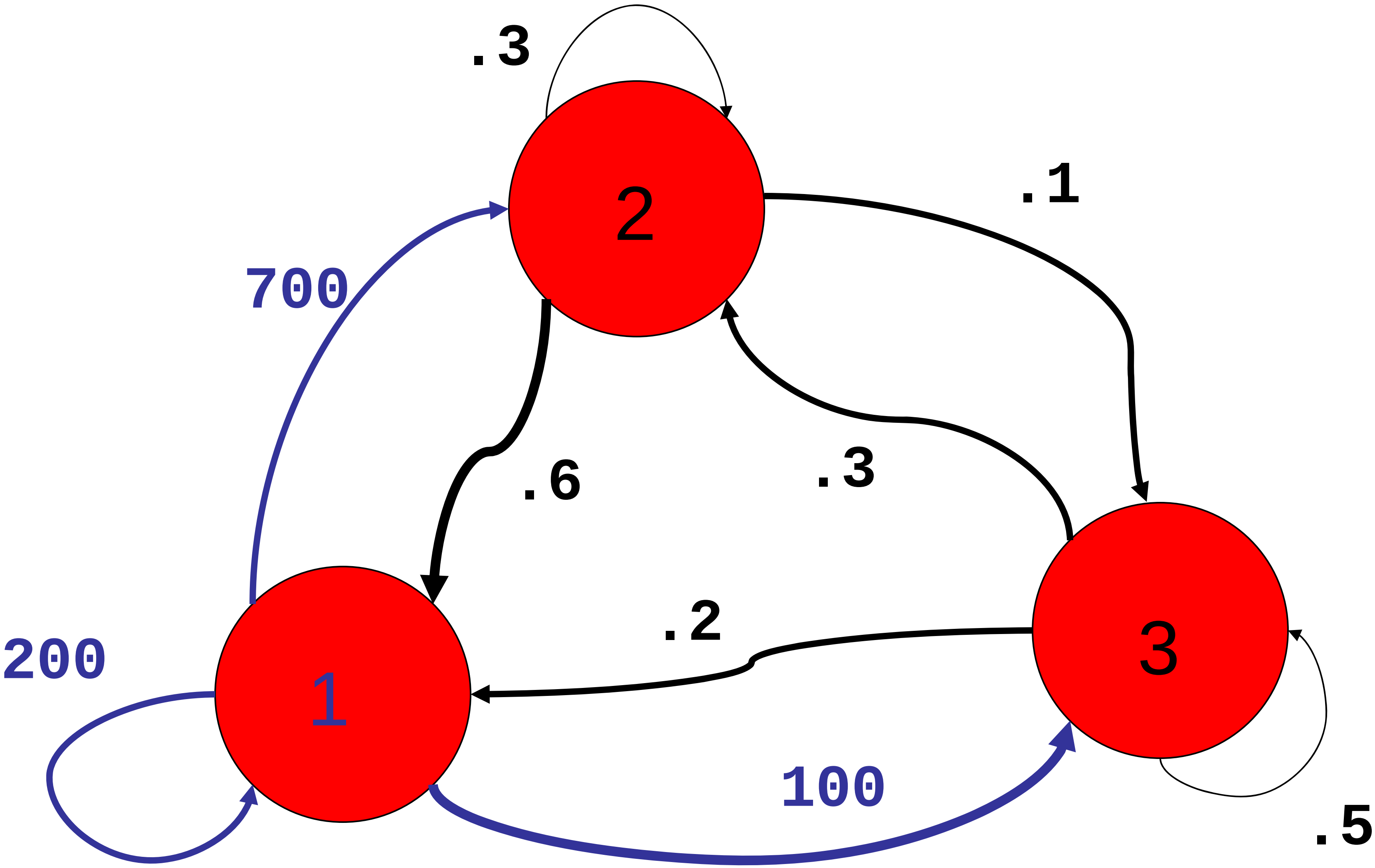
At the sound of a whistle they hop to another node in accordance with the "outbound" probabilities.

For now we assume we know these probabilities. Later we will see how to get them.

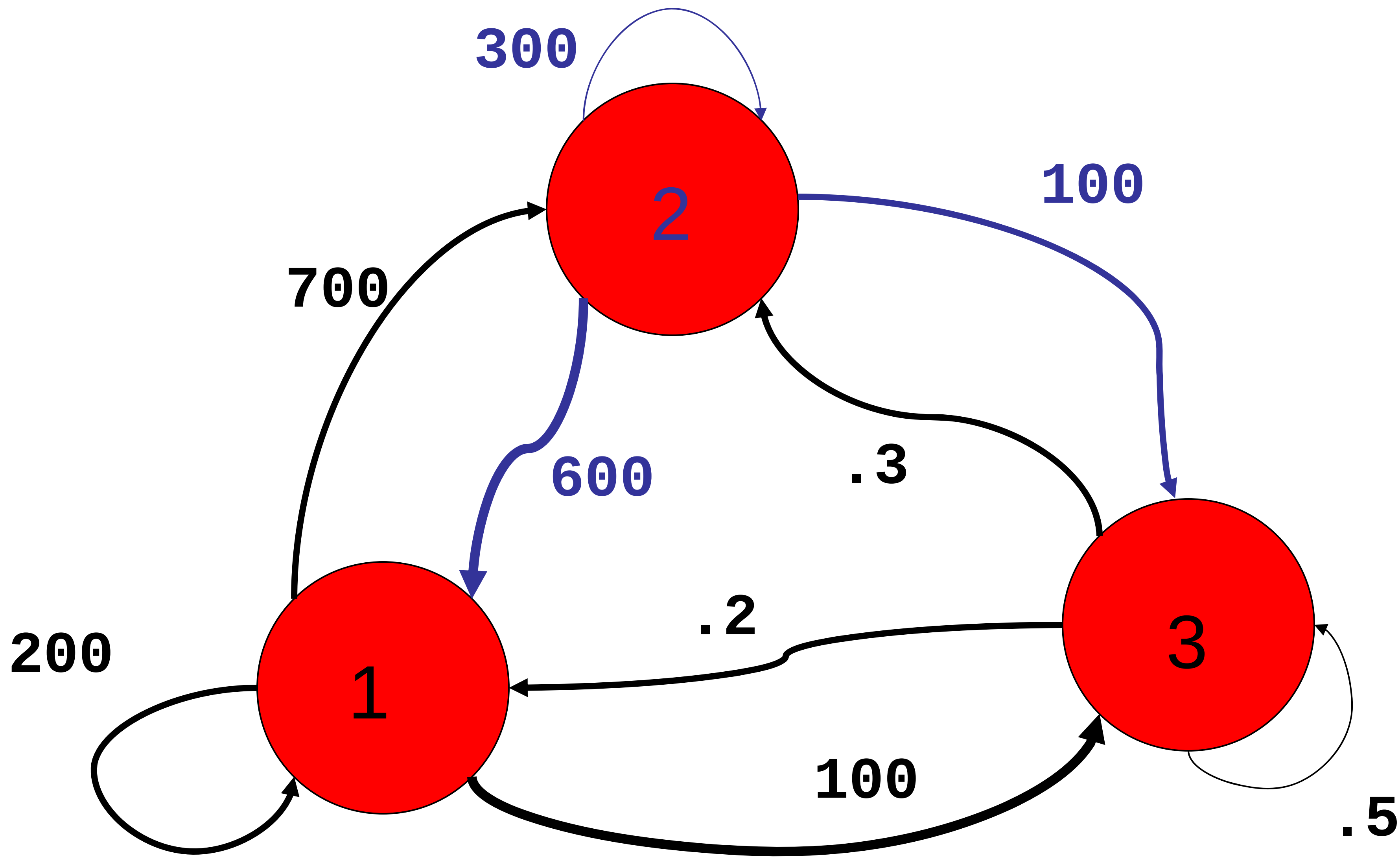
At Node 1



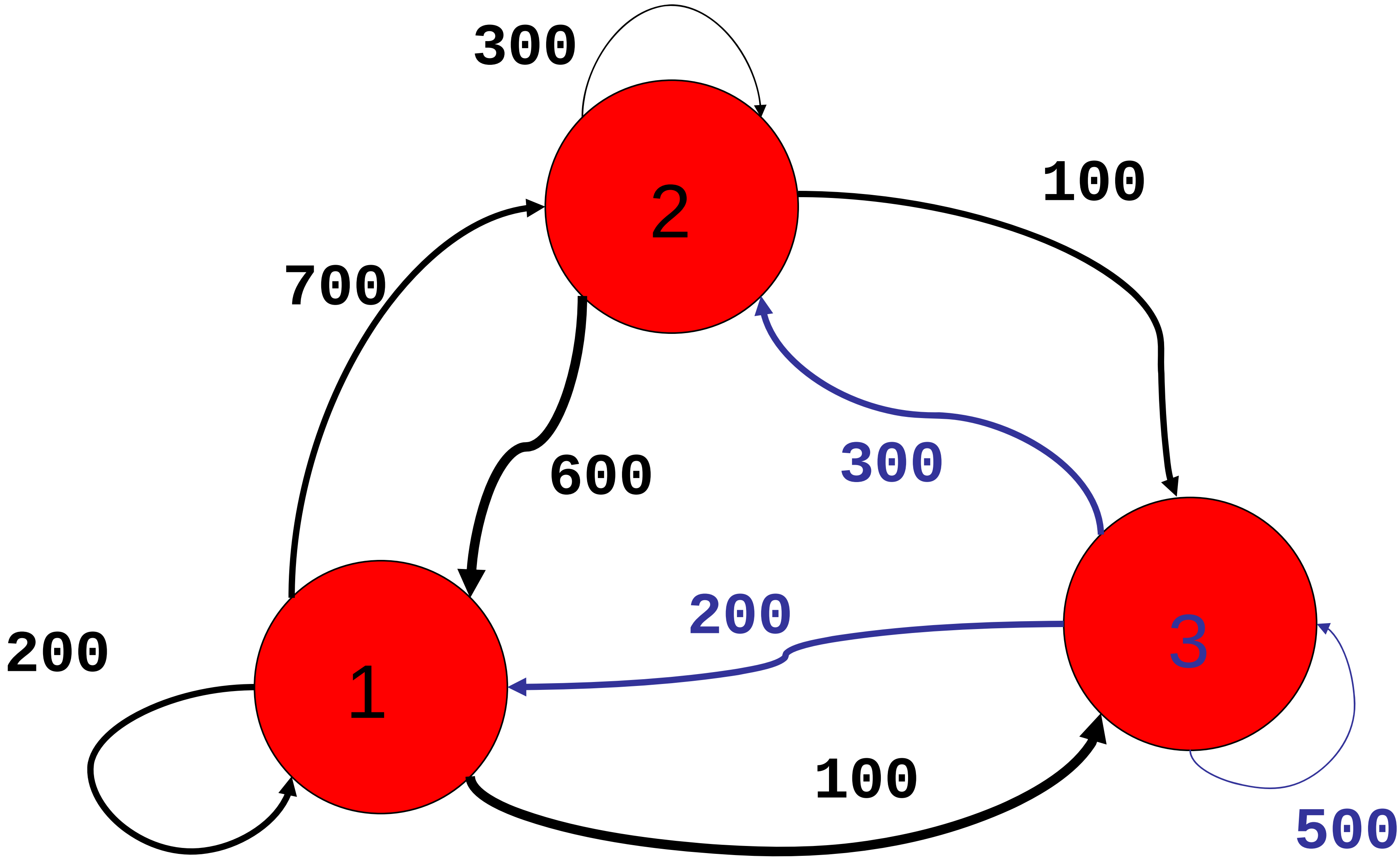
At Node 1



At Node 2



At Node 3



State Vector:
describes the state at each node at a specific time

T=0

$$\begin{pmatrix} 1000 \\ 1000 \\ 1000 \end{pmatrix}$$

T=1

$$\begin{pmatrix} 1000 \\ 1300 \\ 700 \end{pmatrix}$$

T=2

$$\begin{pmatrix} 1120 \\ 1300 \\ 580 \end{pmatrix}$$

After 100 iterations

	T=99	T=100
Node 1	1142.85	1142.85
Node 2	1357.14	1357.14
Node 3	500.00	500.00

Appears to reach a steady state

Call this the stationary vector

Transition Probability Matrix

P

.2	.6	.2
.7	.3	.3
.1	.1	.5

`P(i,j)` is the probability of hopping to node **`i`** from node **`j`**

Formula for the new state vector

P(i,j) is probability of hopping to node **i** from node **j**

P

.2	.6	.2
.7	.3	.3
.1	.1	.5

```
W(1) = P(1,1)*v(1) + P(1,2)*v(2) + P(1,3)*v(3)
W(2) = P(2,1)*v(1) + P(2,2)*v(2) + P(2,3)*v(3)
W(3) = P(3,1)*v(1) + P(3,2)*v(2) + P(3,3)*v(3)
```

v is the old state vector

w is the updated state vector

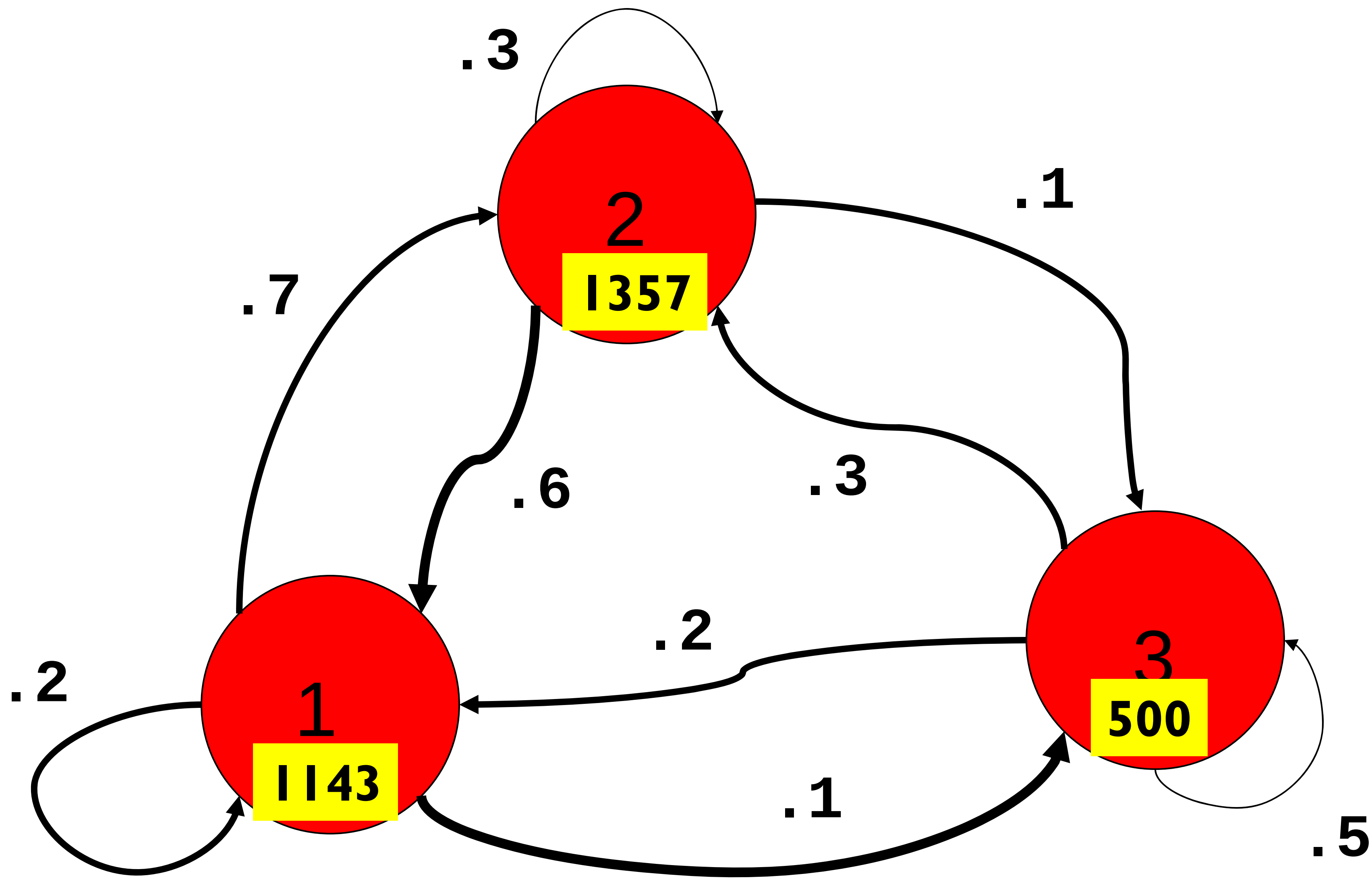
The general case

```
function w = Update(P,v)
% Update state vector v based on transition
% probability matrix P to give state vector w
n = length(v);
w = zeros(n,1);
for i=1:n
   for j=1:n
      w(i) = w(i) + P(i,j)*v(j);
   end
end
```

To obtain the stationary vector…

```
function [w,err]= StatVec(P,v,tol,kMax)
% Iterate to get stationary vector w
w = Update(P,v);
err = max(abs(w-v));
k = 1;
while k<kMax && err>tol
      v = w;
      w = Update(P,v);
      err = max(abs(w-v));
      k = k+1;
end
```

Stationary vector indicates importance: 2 1 3



A random walk on the web

Repeat:

You are on a webpage.

There are m outlinks, so choose one at random.

Click on the link.

Random island hopping

Repeat:

You are on an island.

According to the transitional probabilities,

go to another island.

Use the link structure of the web to figure out the transitional probabilities!

(Assume no dead ends for now; we deal with them later.)

Connectivity Matrix

G

0	0	0	0	0	0	1	1
1	0	0	1	0	0	0	0
1	0	1	0	0	1	0	1
0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	1
0	0	1	0	0	0	0	1
0	0	1	0	0	0	0	0
0	1	0	1	0	0	0	0

`G(i,j)` is **1** if there is a link on page `j` to page `i`.

(I.e., you can get to `i` from `j`.)

Connectivity Matrix

G

0	0	0	0	0	0	1	1
1	0	0	1	0	0	0	0
1	0	1	0	0	1	0	1
0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	1
0	0	1	0	0	0	0	1
0	0	1	0	0	0	0	0
0	1	0	1	0	0	0	0

Transition Probability Matrix

derived from Connectivity Matrix

P

0	0	0	0	0	0	?	?
?	0	0	?	0	0	0	0
?	0	?	0	0	?	0	?
0	0	0	0	?	0	0	0
?	0	?	0	0	0	0	?
0	0	?	0	0	0	0	?
0	0	?	0	0	0	0	0
0	?	0	?	0	0	0	0

Connectivity Matrix

G

0	0	0	0	0	0	1	1
1	0	0	1	0	0	0	0
1	0	1	0	0	1	0	1
0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	1
0	0	1	0	0	0	0	1
0	0	1	0	0	0	0	0
0	1	0	1	0	0	0	0

Transition Probability

P

0	0	0	0	0	0	?	?
?	0	0	?	0	0	0	0
?	0	?	0	0	?	0	?
0	0	0	0	?	0	0	0
?	0	?	0	0	0	0	?
0	0	?	0	0	0	0	?
0	0	?	0	0	0	0	0
0	?	0	?	0	0	0	0

A. 0

B. 1/8

C. 1/3

D. 1

E. rand(1)

Connectivity Matrix

G

0	0	0	0	0	0	1	1
1	0	0	1	0	0	0	0
1	0	1	0	0	1	0	1
0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	1
0	0	1	0	0	0	0	1
0	0	1	0	0	0	0	0
0	1	0	1	0	0	0	0

Transition Probability Matrix

derived from Connectivity Matrix

P

0	0	0	0	0	0	1	.25
.33	0	0	.50	0	0	0	0
.33	0	.25	0	0	1	0	.25
0	0	0	0	1	0	0	0
.33	0	.25	0	0	0	0	.25
0	0	.25	0	0	0	0	.25
0	0	.25	0	0	0	0	0
0	1	0	.50	0	0	0	0

Connectivity (G) → Transition Probability (P)

```
[n,n] = size(G);
P = zeros(n,n);
for j=1:n
   P(:,j) = G(:,j)/sum(G(:,j));
end
```

To obtain the stationary vector…

```
function [w,err]= StatVec(P,v,tol,kMax)
% Iterate to get stationary vector w
w = Update(P,v);
err = max(abs(w-v));
k = 1;
while k<kMax && err>tol
      v = w;
      w = Update(P,v);
      err = max(abs(w-v));
      k = k+1;
end
```

Stationary vector represents how "popular" the pages are → PageRank

statVec	sorted	idx	pR
0.5723	0.8911	6	4
0.8206	0.8206	2	2
0.7876	0.7876	3	3
0.2609	0.5723	1	6
0.2064	0.4100	8	8
0.8911	0.2609	4	1
0.2429	0.2429	7	7
0.4100	0.2064	5	5

```
[sorted, idx] = sort(-statVec);
for k= 1:length(statVec)
   j = idx(k);  % index of kth largest
   pR(j) = k;
end
```

statVec	sorted	idx	pR
0.5723	-0.8911	6	4
0.8206	-0.8206	2	2
0.7876	-0.7876	3	3
0.2609	-0.5723	1	6
0.2064	-0.4100	8	8
0.8911	-0.2609	4	1
0.2429	-0.2429	7	7
0.4100	-0.2064	5	5

The random walk idea gets the transitional probabilities from connectivity. So how to deal with dead ends?

Repeat:

You are on a webpage.

There are m outlinks.

Choose one at random.

Click on the link.

What if there are no outlinks?

The random walk idea gets transitional probabilities from connectivity. Can modify the random walk to deal with dead ends.

```
Repeat:
    You are on a webpage.
    If there are no outlinks
        Pick a random page and go there.
    else
        Flip an unfair coin.
        if heads
            Click on a random outlink and go there.
        else
            Pick a random page and go there.
        end
    end
```

In practice, an unfair coin with prob .85 heads works well.

This results in a different transitional probability matrix.

Quantifying Importance

How do you rank web pages for importance given that you know the link structure of the Web, i.e., the in-links and out-links for each web page?

A related question:

How does a deleted or added link on a webpage affect its "rank"?

What we learned…

- Develop/implement algorithms for problems
- Develop programming skills
  - Design, implement, document, test, and debug
- Programming "tool bag"
  - Control flow (if-else; loops)
  - Functions for reducing redundancy
  - Data structures
  - Graphics
  - File handling

What we learned… (cont'd)

- Applications and concepts
  - Image and sound
  - Sorting and searching—you should know the algorithms covered
  - Divide-and-conquer strategies
  - Approximation and error
  - Simulation
  - Computational effort and efficiency

Final Exam

- Mon 12/10, 9-11:30am, Barton West
- Covers entire course, but emphasizes material after Prelim 3
- Closed-book exam, no calculators
- Bring student ID card

- Check for announcements on webpage:
  - Study break office/consulting hours
  - Review questions
  - List of potentially useful functions

Final Exam

- Mon 12/10, 9-11:30am, Barton West
- Covers entire course, but emphasizes mat…
  after Prelim 3
- Closed-boo…

Best wishes
and
good luck with all your exams!

- …ge.
  - …iting hours
  - … questions
  - List of potentially useful functions