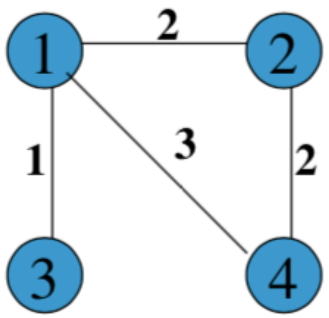
CLUSTERING

MARKOV CLUSTERING ALGORITHM

Markov Clustering

Some of the content in this lesson is taken from the publication:



Van Dongen, S. (2000)

Graph Clustering by Flow Simulation.

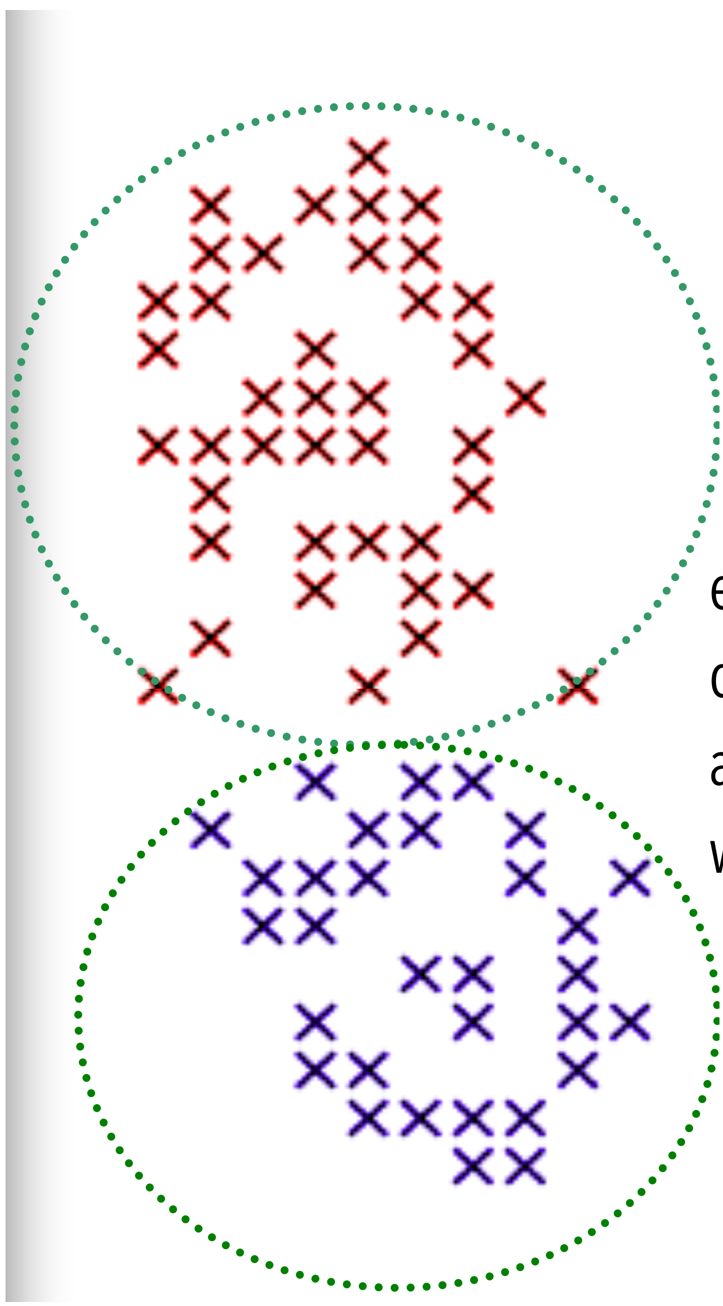
PhD Thesis, University of Utrecht, The Netherlands.

The MCL software can be downloaded from http://www.micans.org/mcl/

Markov Clustering

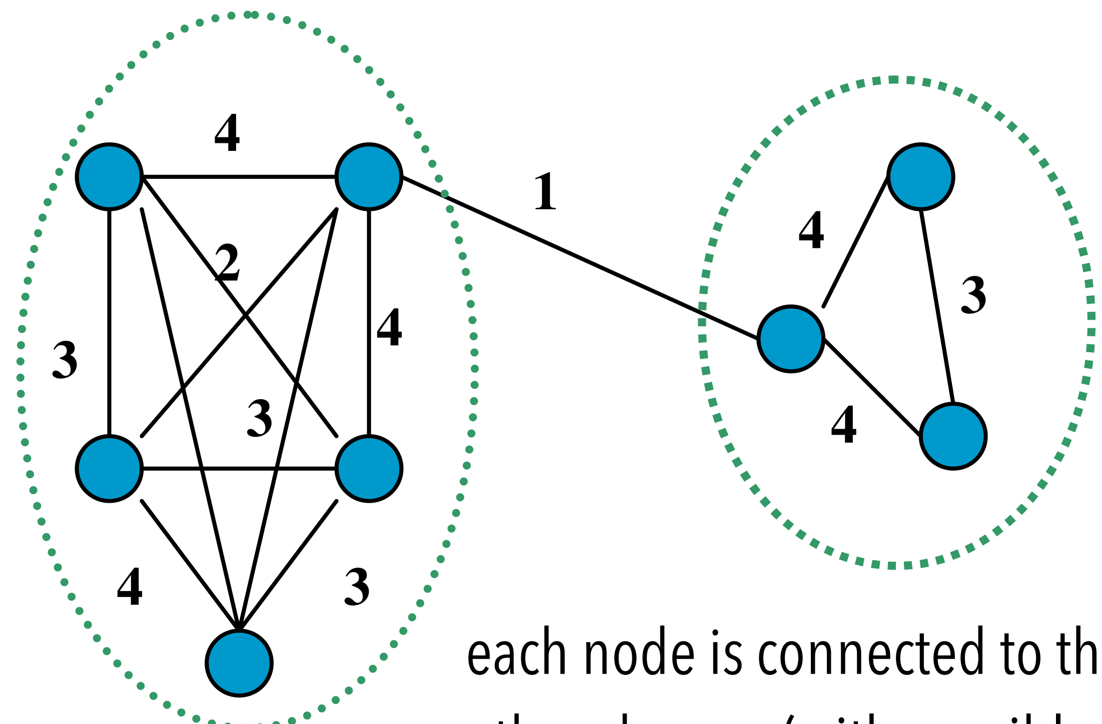
- It is a graph-based clustering algorithm used in bioinformatics.

Vector clustering



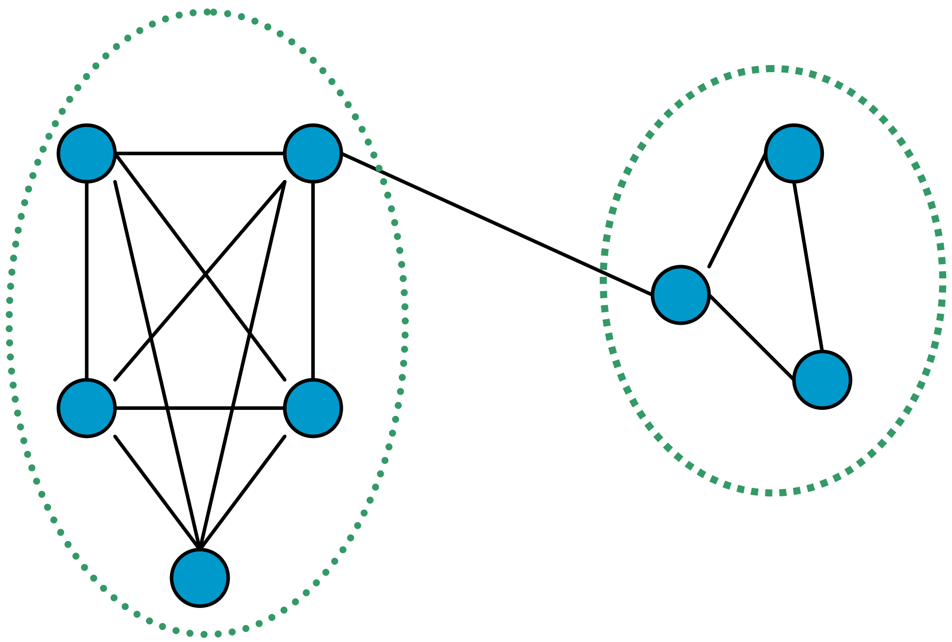
each point has coordinates (x, y) and a class (color) to which it belongs

Graph clustering



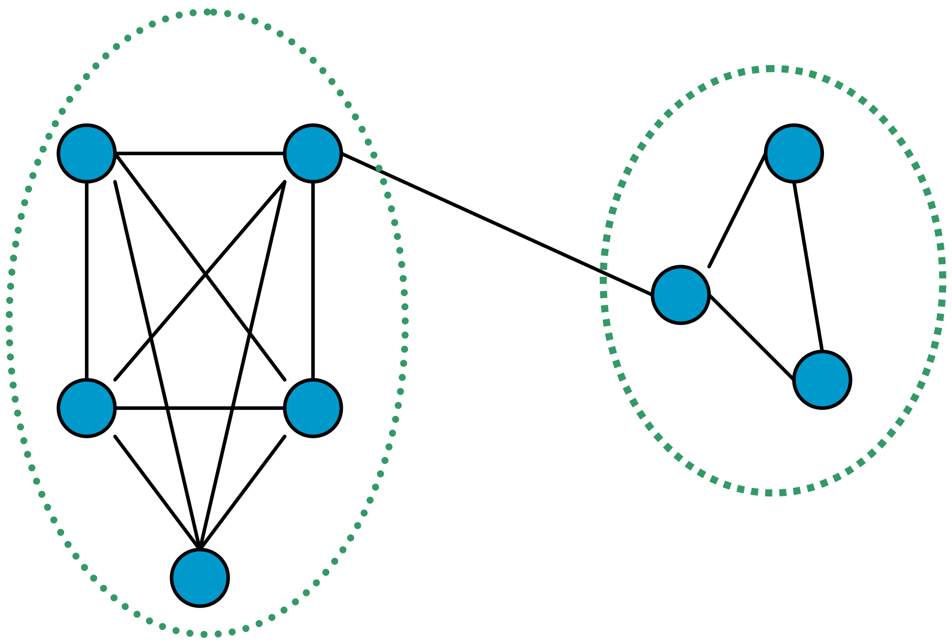
each node is connected to the others by arcs (with possible weights)

Random Walks



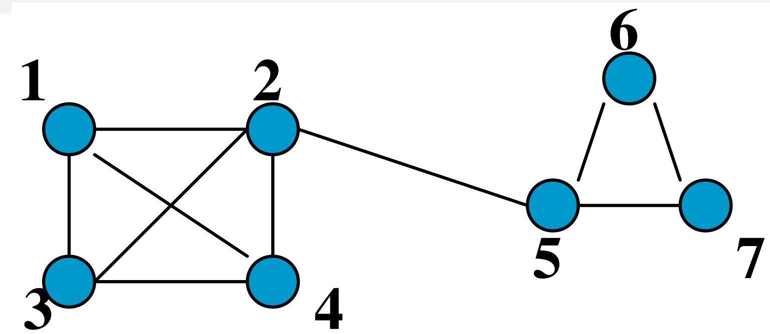
- Considering a graph, there will be many links within a cluster and few between clusters.
- This means that if you start from a node and follow a random path to another connected node, you are more likely to stay within a cluster than to cross it to reach the other node.
- This is the concept on which the MCL algorithm is based.

Random Walks



- Following *random walks* in the graph, it is possible to find out where the flows (paths) tend to converge, and therefore where are the clusters.
- The Random Walks on a graph are calculated by means of the "Markov Chains".

Random Walks



Let's see a working example.

- At a first step, a random *walker* starting at node 1 has a 33% probability of going to nodes 2, 3 and 4 and 0% probability of going to nodes 5, 6 or 7.
- On the other hand, starting from node 2, it has a 25% probability of reaching nodes 1, 3, 4, 5 and 0% towards nodes 6 and 7.
- The corresponding transition matrix (paths on columns) will then be:

	1	*2*	*3*	*4*	*5*	*6*	*7*
1	0	.25	.33	.33	0	0	0
2	.33	0	.33	.33	.33	0	0
3	.33	.25	0	.33	0	0	0
4	.33	.25	.33	0	0	0	0
5	0	.25	0	0	0	.5	.5
6	0	0	0	0	.33	0	.5
7	0	0	0	0	.33	.5	0

- each column has sum 1
- it can therefore be seen like a *probability matrix*

Markov Chain

A simpler example.

$$\begin{array}{c|cc} & 1 & 2 \\ \hline 1 & .6 & .2 \\ 2 & .4 & .8 \end{array}$$

- *the transitions materialize in the (repeated) product of the probability matrices*

- Let's valuate the steps at times $t_0 \rightarrow t_1 \rightarrow t_2$

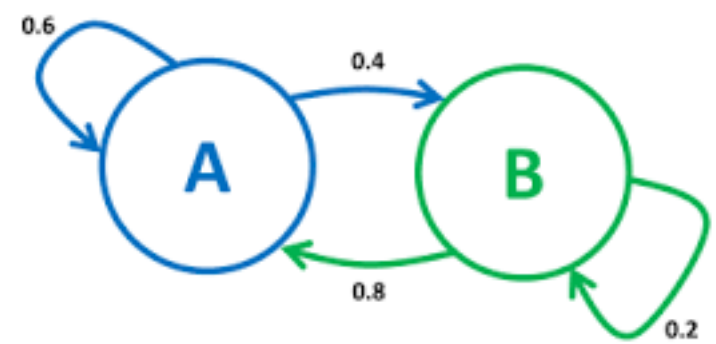
$$.6 \times .6 + .2 \times .4 = .44$$

$$\cdots$$

$$.4 \times .2 + .8 \times .8 = .72$$

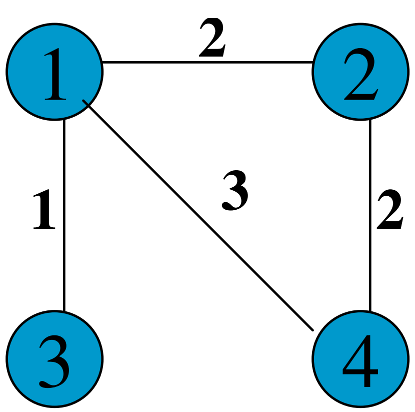
$$\begin{pmatrix} .6 & .2 \\ .4 & .8 \end{pmatrix} \cdot \begin{pmatrix} .6 & .2 \\ .4 & .8 \end{pmatrix} = \begin{pmatrix} .44 & .28 \\ .56 & .72 \end{pmatrix} \longrightarrow \begin{pmatrix} .35 & .32 \\ .65 & .68 \end{pmatrix} \longrightarrow \begin{pmatrix} .34 & .33 \\ .67 & .67 \end{pmatrix}$$

Markov Chain



- A sequence of variables X_1, X_2, X_3, ... (in our case probability matrices) in which, given the present state, the past and future states are independent.
- A "Markov Chain" (Markovian process) therefore has no memory.
- The probabilities for the next time step depend only on the current probabilities.
- A Random Walk is an example of a Markov Chain, using probability transition matrices.

Weighted Graphs (grafi pesati)



- To transform a weighted graph into a (transition of) probability matrix we need to normalize the columns:

$$\begin{pmatrix} 0 & 2 & 1 & 3 \\ 2 & 0 & 0 & 2 \\ 1 & 0 & 0 & 0 \\ 3 & 2 & 0 & 0 \end{pmatrix}$$

$$\downarrow$$

$$\begin{pmatrix} 0 & 1/2 & 1 & 3/5 \\ 1/3 & 0 & 0 & 2/5 \\ 1/6 & 0 & 0 & 0 \\ 1/2 & 1/2 & 0 & 0 \end{pmatrix}$$

- *a column is normalized by dividing each element by the sum of its elements;*
- *at the end of the transformation the matrix is no longer symmetric!*

Markov Clustering

Self Loops

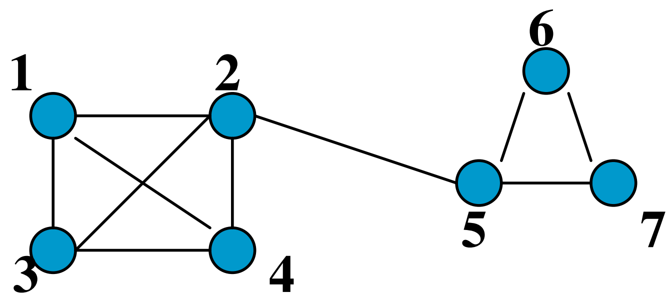
- Small simple paths with loops can complicate things.
  - There is a strong effect given that odd powers of expansion obtain their mass from simple paths of odd length, as well as for those of even length.
  - This makes the transition probabilities dependent on the equality of the lengths of the simple paths.
- Adding self-loop arcs on each node solves this problem.
  - A self-loop adds a small path of length 1 so that the mass does not appear only in the odd (or even) powers of the matrix.

$$\begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix} \longrightarrow \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{pmatrix}$$

Markov Clustering

Cluster structure of a Markov chain

Example.



0	.25	.33	.33	0	0	0
.33	0	.33	.33	.33	0	0
.33	.25	0	.33	0	0	0
.33	.25	.33	0	0	0	0
0	.25	0	0	0	.5	.5
0	0	0	0	.33	0	.5
0	0	0	0	.33	.5	0

eventually →

.15	.15	.15	.15	.15	.15	.15
.2	.2	.2	.2	.2	.2	.2
.15	.15	.15	.15	.15	.15	.15
.15	.15	.15	.15	.15	.15	.15
.15	.15	.15	.15	.15	.15	.15
.1	.1	.1	.1	.1	.1	.1
.1	.1	.1	.1	.1	.1	.1

- Note that, in the initial steps, before the flow shuffles, the cluster structure is already evident in the matrix.
- This is not a coincidence and the MCL algorithm uses this feature, by modifying the random walk process, to further emphasize the separation between clusters in the matrix.

The MCL algorithm

- Flow is easier through dense regions than through scattered boundaries; however, in the long run this effect fades.
- During the initial powers of the Markov Chain, the weights of the arcs will be **larger** in the links *within* clusters, and smaller in the links *between* clusters.
- This means that there is a correspondence between the distribution of weights on the columns and the clustering.

The MCL algorithm

- MCL deliberately increases this effect:
  - by first breaking the chain;
  - then modifying the transitions through the columns.
- For each node, the transition values are changed so that:
  - the "strong" neighbors are further strengthened;
  - "weaker" neighbors are demoted.
- This change can be made by raising a single column to a non-negative power, and then re-normalizing it.
- This operation is called "*Inflation*".
- Raising the matrix to a power is called "*Expansion*".

MCL Inflation

- Example of order 2 inflation (elevation squared):

Column i Column j

$$\begin{pmatrix} 0 \\ 1/2 \\ 0 \\ 1/6 \\ 1/3 \end{pmatrix} \quad \begin{pmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \\ 0 \end{pmatrix}$$

$$\downarrow$$

$$\begin{pmatrix} 0 \\ 9/14 \\ 0 \\ 1/14 \\ 4/14 \end{pmatrix} \quad \begin{pmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \\ 0 \end{pmatrix}$$

- *Elevates a column squared, then normalizes it*

MCL Inflation

Definition. *Given a matrix* $M \in \mathbb{R}^{k \times l}, M \geq 0$ *, and a real non-negative number* $r, \Gamma_r M$ *is the matrix obtained by raising each column of* M *to the power* $r; \Gamma_r\ T$ *is called the* ***inflation*** *operator with power coefficient* r.
Formally, the operation of $\Gamma_r : \mathbb{R}^{k \times l}$ is defined as follows:

$$(\Gamma_r M)_{pq} = \frac{(M_{pq})^r}{\sum_{i=1}^{k} (M_{iq})^r}$$

q indicates the vertex (column) attracted by vertex p

If r is omitted, the power coefficient is equal to 2.

- The inflation operator is responsible for both strengthening and weakening flows (increases the strength of already strong flows, decreases the strength of already weak flows).
- The inflation parameter r controls the speed of this process.
- This eventually affects the *granularity* of the obtained clusters.

The MCL algorithm

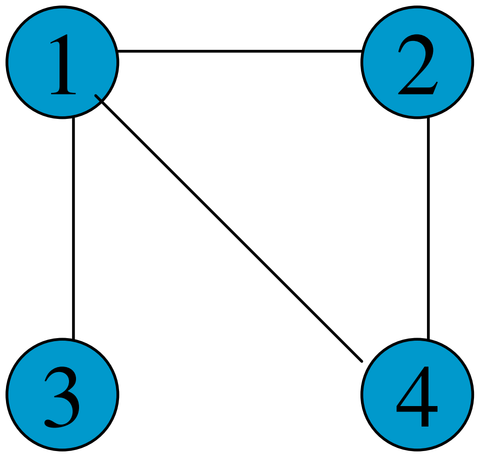
- The following two processes alternate repeatedly in the MCL algorithm:
  - **Expansion** (raise the transition matrix to a power);
  - **Inflation.**
- The expansion operator allows the flow to connect different regions of the graph.
- The inflation operator is responsible for the increase (intra-cluster) and decrease (inter-cluster) of the flow.

The MCL algorithm

1. Input: undirected (bidirectional) graph g, expansion parameter e, inflation parameter r.
2. Create the matrix M associated with the graph.
3. Add (possibly) self-loops to the nodes.
4. Normalize the M matrix.
5. Expand by raising the matrix M to the power e, obtaining M'.
6. Inflate the matrix M' obtained in the previous step by applying the parameter r.
7. Repeat steps 5 and 6 until convergence [$M(i+1) = M(i)$] is achieved.
8. Interpret the resulting matrix to discover the clusters.

The MCL algorithm

$e = 2$
$r = 2$



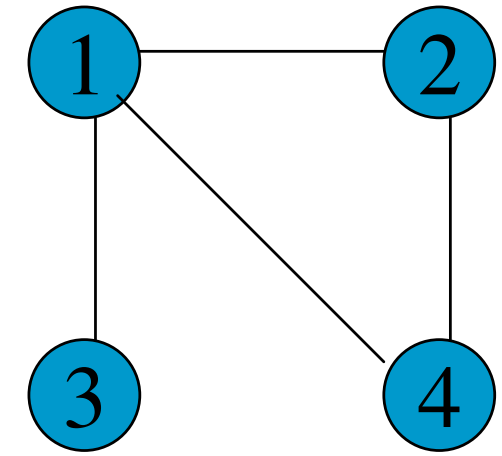
1. **Input: undirected (bidirectional) graph *g*, expansion parameter *e*, inflation parameter *r*.**
2. Create the matrix *M* associated with the graph.
3. Add (possibly) self-loops to the nodes.
4. Normalize the *M* matrix.
5. Expand by raising the matrix *M* to the power *e*, obtaining *M*'.
6. Inflate the matrix *M*' obtained in the previous step by applying the parameter *r*.
7. Repeat steps 5 and 6 until convergence [$M(i+1) = M(i)$] is achieved.
8. Interpret the resulting matrix to discover the clusters.

The MCL algorithm

1. Input: undirected (bidirectional) graph *g*, expansion parameter *e*, inflation parameter *r*.
2. **Create the matrix *M* associated with the graph.**
3. Add (possibly) self-loops to the nodes.
4. Normalize the *M* matrix.
5. Expand by raising the matrix *M* to the power *e*, obtaining *M*'.
6. Inflate the matrix *M*' obtained in the previous step by applying the parameter *r*.
7. Repeat steps 5 and 6 until convergence [*M*(*i*+1) = *M*(*i*)] is achieved.
8. Interpret the resulting matrix to discover the clusters.

$e = 2$

$r = 2$



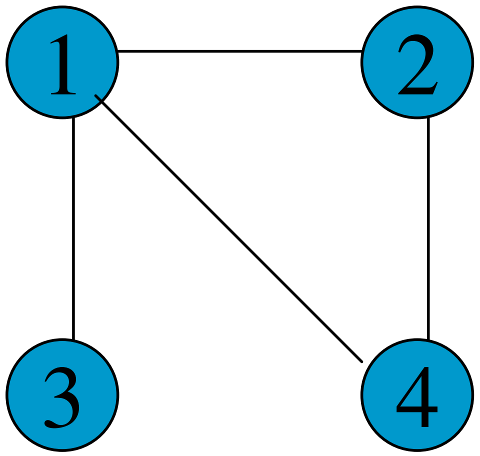
0	1	1	1
1	0	0	1
1	0	0	0
1	1	0	0

The MCL algorithm

1. Input: undirected (bidirectional) graph *g*, expansion parameter *e*, inflation parameter *r*.
2. Create the matrix *M* associated with the graph.
3. **Add (possibly) self-loops to the nodes.**
4. Normalize the *M* matrix.
5. Expand by raising the matrix *M* to the power *e*, obtaining *M'*.
6. Inflate the matrix *M'* obtained in the previous step by applying the parameter *r*.
7. Repeat steps 5 and 6 until convergence [$M(i+1) = M(i)$] is achieved.
8. Interpret the resulting matrix to discover the clusters.

$e = 2$

$r = 2$



$$\begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix}$$

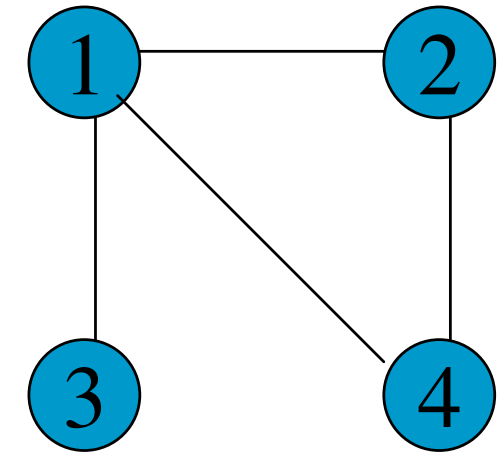
$$\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{pmatrix}$$

Markov Clustering

The MCL algorithm

1. Input: undirected (bidirectional) graph *g*, expansion parameter *e*, inflation parameter *r*.
2. Create the matrix *M* associated with the graph.
3. Add (possibly) self-loops to the nodes.
4. **Normalize the *M* matrix.**
5. Expand by raising the matrix *M* to the power *e*, obtaining *M*'.
6. Inflate the matrix *M*' obtained in the previous step by applying the parameter *r*.
7. Repeat steps 5 and 6 until convergence [*M*(*i*+1) = *M*(*i*)] is achieved.
8. Interpret the resulting matrix to discover the clusters.

***e* = 2**
***r* = 2**



$$\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{pmatrix}$$

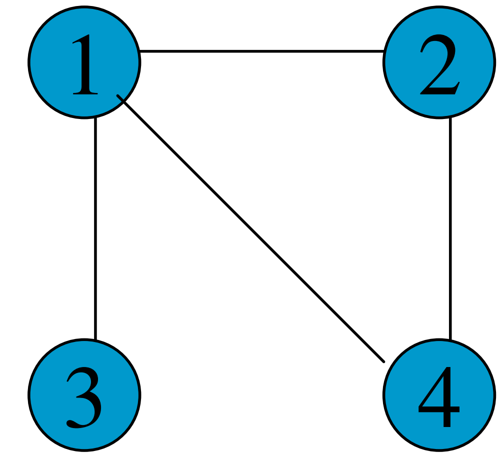
$$\begin{pmatrix} 1/4 & 1/3 & 1/2 & 1/3 \\ 1/4 & 1/3 & 0 & 1/3 \\ 1/4 & 0 & 1/2 & 0 \\ 1/4 & 1/3 & 0 & 1/3 \end{pmatrix}$$

The MCL algorithm

1. Input: undirected (bidirectional) graph *g*, expansion parameter *e*, inflation parameter *r*.
2. Create the matrix *M* associated with the graph.
3. Add (possibly) self-loops to the nodes.
4. Normalize the *M* matrix.
5. **Expand by raising the matrix *M* to the power *e*, obtaining *M*'.**
6. Inflate the matrix *M*' obtained in the previous step by applying the parameter *r*.
7. Repeat steps 5 and 6 until convergence [$M(i+1) = M(i)$] is achieved.
8. Interpret the resulting matrix to discover the clusters.

$e = 2$

$r = 2$

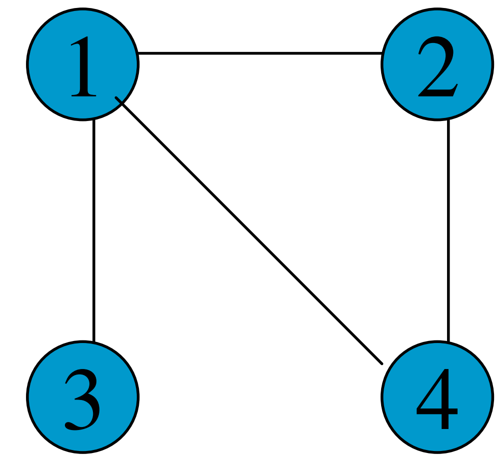


$$\begin{pmatrix} 1/4 & 1/3 & 1/2 & 1/3 \\ 1/4 & 1/3 & 0 & 1/3 \\ 1/4 & 0 & 1/2 & 0 \\ 1/4 & 1/3 & 0 & 1/3 \end{pmatrix} \cdot \begin{pmatrix} 1/4 & 1/3 & 1/2 & 1/3 \\ 1/4 & 1/3 & 0 & 1/3 \\ 1/4 & 0 & 1/2 & 0 \\ 1/4 & 1/3 & 0 & 1/3 \end{pmatrix} = \begin{pmatrix} .35 & .31 & .38 & .31 \\ .23 & .31 & .13 & .31 \\ .19 & .08 & .38 & .08 \\ .23 & .31 & .13 & .31 \end{pmatrix}$$

The MCL algorithm

$e = 2$

$r = 2$



1. Input: undirected (bidirectional) graph *g*, expansion parameter *e*, inflation parameter *r*.
2. Create the matrix *M* associated with the graph.
3. Add (possibly) self-loops to the nodes.
4. Normalize the *M* matrix.
5. Expand by raising the matrix *M* to the power *e*, obtaining *M'*.
6. **Inflate the matrix *M'* obtained in the previous step by applying the parameter *r*.**
7. Repeat steps 5 and 6 until convergence [$M(i+1) = M(i)$] is achieved.
8. Interpret the resulting matrix to discover the clusters.

$$\begin{pmatrix} .35 & .31 & .38 & .31 \\ .23 & .31 & .13 & .31 \\ .19 & .08 & .38 & .08 \\ .23 & .31 & .13 & .31 \end{pmatrix}$$

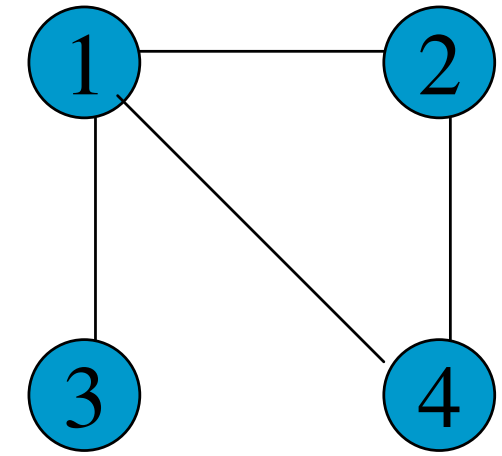
$$\begin{pmatrix} .13 & .09 & .14 & .09 \\ .05 & .09 & .02 & .09 \\ .04 & .01 & .14 & .01 \\ .05 & .09 & .02 & .09 \end{pmatrix}$$

$$\begin{pmatrix} .47 & .33 & .45 & .33 \\ .20 & .33 & .05 & .33 \\ .13 & .02 & .45 & .02 \\ .20 & .33 & .05 & .33 \end{pmatrix}$$

The MCL algorithm

$e = 2$

$r = 2$



1. Input: undirected (bidirectional) graph *g*, expansion parameter *e*, inflation parameter *r*.
2. Create the matrix *M* associated with the graph.
3. Add (possibly) self-loops to the nodes.
4. Normalize the *M* matrix.
5. Expand by raising the matrix *M* to the power *e*, obtaining *M*'.
6. Inflate the matrix *M*' obtained in the previous step by applying the parameter *r*.
7. **Repeat steps 5 and 6 until convergence [*M*(*i*+1) = *M*(*i*)] is achieved.**
8. Interpret the resulting matrix to discover the clusters.

$$\begin{pmatrix} .70 & .33 & .49 & .33 \\ .12 & .33 & .01 & .33 \\ .05 & .02 & .49 & \text{--} \\ .12 & .33 & .01 & .33 \end{pmatrix}$$

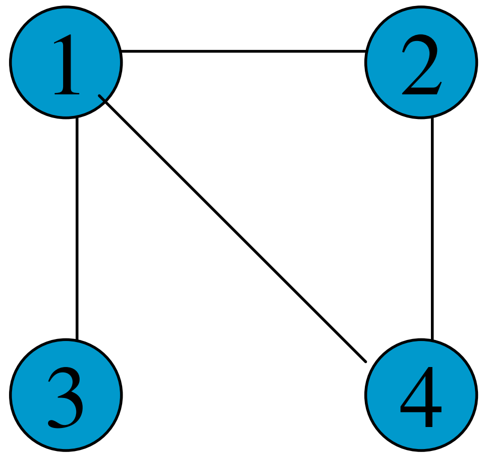
$$\begin{pmatrix} .94 & .33 & .50 & .33 \\ .03 & .33 & \text{--} & .33 \\ .01 & \text{--} & .50 & \text{--} \\ .13 & .33 & \text{--} & .33 \end{pmatrix}$$

$$\begin{pmatrix} 1 & .33 & .50 & .33 \\ \text{--} & .33 & \text{--} & .33 \\ \text{--} & \text{--} & .50 & \text{--} \\ \text{--} & .33 & \text{--} & .33 \end{pmatrix}$$

The MCL algorithm

1. Input: undirected (bidirectional) graph g, expansion parameter e, inflation parameter r.
2. Create the matrix M associated with the graph.
3. Add (possibly) self-loops to the nodes.
4. Normalize the M matrix.
5. Expand by raising the matrix M to the power e, obtaining M'.
6. Inflate the matrix M' obtained in the previous step by applying the parameter r.
7. Repeat steps 5 and 6 until convergence [$M(i+1) = M(i)$] is achieved.
8. **Interpret the resulting matrix to discover the clusters.**

$e = 2$
$r = 2$



later...

MCL Algorithm Convergence

- It is not proved that the algorithm converges [$M(i+1) = M(i)$]; in the doctoral thesis the author shows its convergence only experimentally...
- In practice, the algorithm *almost always* converges to a "doubly idempotent" matrix (steady state, equal values in the columns):

$$
\begin{pmatrix}
1.000 & -- & -- & -- & -- & 1.000 & 1.000 & -- & -- & 1.000 & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & 1.000 & 1.000 & -- & 1.000 & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & 0.500 & -- & -- & -- & 0.500 & 0.500 & -- & 0.500 & 0.500 \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & 0.500 & -- & -- & -- & 0.500 & 0.500 & -- & 0.500 & 0.500 \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & --
\end{pmatrix}
$$

$$M_{mcl}^{\infty}$$

MCL Algorithm Convergence

- It is shown that when the matrix is about to become "doubly idempotent", the algorithm converges quadratically.
- However, the final steady state can sometimes be cyclic and consist of a sequence of identically repeating matrices.
  - *In some cases, expansion and inflation behave as the inverse of each other.*
  - *This usually occurs in the absence of self-loops in bipartite graphs due to the odd length of the paths.*
  - *To overcome this, it is sufficient to add the self-loops and make a slight modification to the parameters.*

MCL Algorithm Convergence

$$
\begin{pmatrix}
0.200 & 0.250 & -- & -- & -- & 0.333 & 0.250 & -- & -- & 0.250 & -- & -- \\
0.200 & 0.250 & 0.250 & -- & 0.200 & -- & -- & -- & -- & -- & -- & -- \\
-- & 0.250 & 0.250 & 0.200 & 0.200 & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & 0.250 & 0.200 & -- & -- & -- & 0.200 & 0.200 & -- & 0.200 & -- \\
-- & 0.250 & 0.250 & -- & 0.200 & -- & 0.250 & 0.200 & -- & -- & -- & -- \\
0.200 & -- & -- & -- & -- & 0.333 & -- & -- & -- & 0.250 & -- & -- \\
0.200 & -- & -- & -- & 0.200 & -- & 0.250 & -- & -- & 0.250 & -- & -- \\
-- & -- & -- & 0.200 & 0.200 & -- & -- & 0.200 & 0.200 & -- & 0.200 & -- \\
-- & -- & -- & 0.200 & -- & -- & -- & 0.200 & 0.200 & -- & 0.200 & 0.333 \\
0.200 & -- & -- & -- & -- & 0.333 & 0.250 & -- & -- & 0.250 & -- & -- \\
-- & -- & -- & 0.200 & -- & -- & -- & 0.200 & 0.200 & -- & 0.200 & 0.333 \\
-- & -- & -- & -- & -- & -- & -- & -- & 0.200 & -- & 0.200 & 0.333
\end{pmatrix}
$$

$$M$$

$$
\begin{pmatrix}
0.380 & 0.087 & 0.027 & -- & 0.077 & 0.295 & 0.201 & -- & -- & 0.320 & -- & -- \\
0.047 & 0.347 & 0.210 & 0.017 & 0.150 & 0.019 & 0.066 & 0.012 & -- & 0.012 & -- & -- \\
0.014 & 0.210 & 0.347 & 0.056 & 0.150 & -- & 0.016 & 0.046 & 0.009 & -- & 0.009 & -- \\
-- & 0.027 & 0.087 & 0.302 & 0.062 & -- & -- & 0.184 & 0.143 & -- & 0.143 & 0.083 \\
0.058 & 0.210 & 0.210 & 0.056 & 0.406 & -- & 0.083 & 0.046 & 0.009 & 0.019 & 0.009 & -- \\
0.142 & 0.017 & -- & -- & -- & 0.295 & 0.083 & -- & -- & 0.184 & -- & -- \\
0.113 & 0.069 & 0.017 & -- & 0.062 & 0.097 & 0.333 & 0.012 & -- & 0.147 & -- & -- \\
-- & 0.017 & 0.069 & 0.175 & 0.049 & -- & 0.016 & 0.287 & 0.143 & -- & 0.143 & 0.083 \\
-- & -- & 0.017 & 0.175 & 0.012 & -- & -- & 0.184 & 0.288 & -- & 0.288 & 0.278 \\
0.246 & 0.017 & -- & -- & 0.019 & 0.295 & 0.201 & -- & -- & 0.320 & -- & -- \\
-- & -- & 0.017 & 0.175 & 0.012 & -- & -- & 0.184 & 0.288 & -- & 0.288 & 0.278 \\
-- & -- & -- & 0.044 & -- & -- & -- & 0.046 & 0.120 & -- & 0.120 & 0.278
\end{pmatrix}
$$

$$\Gamma_2 M^2,$$

MCL Algorithm Convergence

$$\begin{pmatrix}
0.448 & 0.080 & 0.023 & -- & 0.068 & 0.426 & 0.359 & -- & -- & 0.432 & -- & -- \\
0.018 & 0.285 & 0.228 & 0.007 & 0.176 & 0.006 & 0.033 & 0.005 & -- & 0.007 & -- & -- \\
0.005 & 0.223 & 0.290 & 0.022 & 0.173 & -- & 0.010 & 0.017 & 0.003 & 0.001 & 0.003 & 0.001 \\
-- & 0.018 & 0.059 & 0.222 & 0.040 & -- & 0.001 & 0.187 & 0.139 & -- & 0.139 & 0.099 \\
0.027 & 0.312 & 0.314 & 0.028 & 0.439 & 0.005 & 0.054 & 0.022 & 0.003 & 0.010 & 0.003 & 0.001 \\
0.116 & 0.007 & 0.001 & -- & 0.004 & 0.157 & 0.085 & -- & -- & 0.131 & -- & -- \\
0.096 & 0.040 & 0.013 & -- & 0.037 & 0.083 & 0.197 & 0.001 & -- & 0.104 & -- & -- \\
-- & 0.012 & 0.042 & 0.172 & 0.029 & -- & 0.002 & 0.198 & 0.133 & -- & 0.133 & 0.096 \\
-- & 0.001 & 0.015 & 0.256 & 0.009 & -- & -- & 0.266 & 0.326 & -- & 0.326 & 0.346 \\
0.290 & 0.021 & 0.002 & -- & 0.017 & 0.323 & 0.260 & -- & -- & 0.316 & -- & -- \\
-- & 0.001 & 0.015 & 0.256 & 0.009 & -- & -- & 0.266 & 0.326 & -- & 0.326 & 0.346 \\
-- & -- & 0.001 & 0.037 & 0.001 & -- & -- & 0.039 & 0.069 & -- & 0.069 & 0.112
\end{pmatrix}$$

$\Gamma_2(\Gamma_2 M^2 \cdot \Gamma_2 M^2)$

$$\begin{pmatrix}
0.807 & 0.040 & 0.015 & -- & 0.034 & 0.807 & 0.807 & -- & -- & 0.807 & -- & -- \\
-- & 0.090 & 0.092 & -- & 0.088 & -- & -- & -- & -- & -- & -- & -- \\
-- & 0.085 & 0.088 & -- & 0.084 & -- & -- & -- & -- & -- & -- & -- \\
-- & 0.001 & 0.001 & 0.032 & 0.001 & -- & -- & 0.032 & 0.031 & -- & 0.031 & 0.031 \\
-- & 0.777 & 0.798 & -- & 0.786 & -- & 0.001 & -- & -- & -- & -- & -- \\
0.005 & -- & -- & -- & -- & 0.005 & 0.005 & -- & -- & 0.005 & -- & -- \\
0.003 & 0.001 & -- & -- & 0.001 & 0.003 & 0.003 & -- & -- & 0.003 & -- & -- \\
-- & -- & 0.001 & 0.024 & -- & -- & -- & 0.024 & 0.024 & -- & 0.024 & 0.024 \\
-- & -- & 0.002 & 0.472 & 0.001 & -- & -- & 0.472 & 0.472 & -- & 0.472 & 0.472 \\
0.185 & 0.005 & 0.001 & -- & 0.004 & 0.185 & 0.184 & -- & -- & 0.185 & -- & -- \\
-- & -- & 0.002 & 0.472 & 0.001 & -- & -- & 0.472 & 0.472 & -- & 0.472 & 0.472 \\
-- & -- & -- & 0.001 & -- & -- & -- & 0.001 & 0.001 & -- & 0.001 & --
\end{pmatrix}$$

$(\Gamma_2 \circ Squaring)$ iterated four times on M

MCL Algorithm Convergence

$$
\begin{pmatrix}
1.000 & -- & -- & -- & -- & 1.000 & 1.000 & -- & -- & 1.000 & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & 1.000 & 1.000 & -- & 1.000 & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & 0.500 & -- & -- & -- & 0.500 & 0.500 & -- & 0.500 & 0.500 \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & 0.500 & -- & -- & -- & 0.500 & 0.500 & -- & 0.500 & 0.500 \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & --
\end{pmatrix}
$$

$$M_{mcl}^{\infty}$$

Interpretation of MCL Clusters

- To interpret clusters, vertices are divided into two types:
  - **attractors** *(attracting other vertices);*
  - *the attracted (vertices that are attracted by the former).*
- Attractors have at least one positive flux value within their corresponding row (of the stationary matrix).
- Each attractor attracts vertices that have positive values within its row.

$$
\begin{pmatrix}
1.000 & -- & -- & -- & -- & 1.000 & 1.000 & -- & -- & 1.000 & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & 1.000 & 1.000 & -- & 1.000 & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & 0.500 & -- & -- & -- & 0.500 & 0.500 & -- & 0.500 & 0.500 \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & 0.500 & -- & -- & -- & 0.500 & 0.500 & -- & 0.500 & 0.500 \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & --
\end{pmatrix}
$$

$$M_{mcl}^{\infty}$$

Interpretation of MCL Clusters

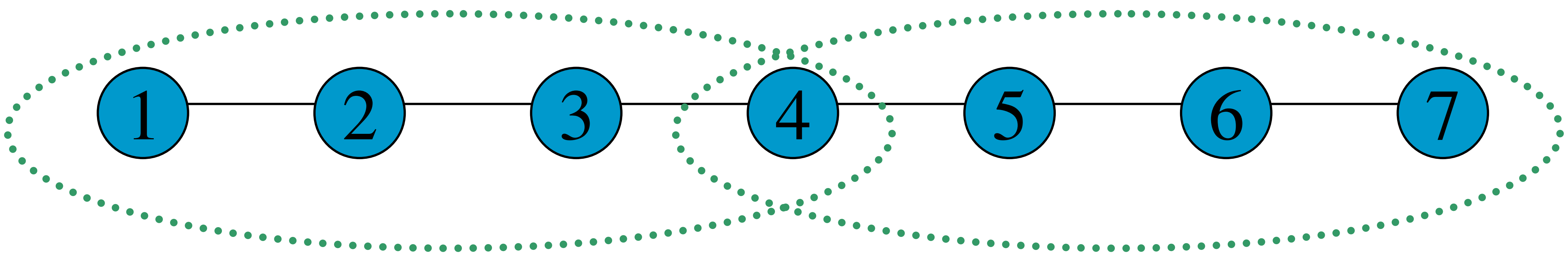
$$
\begin{pmatrix}
1.000 & -- & -- & -- & -- & 1.000 & 1.000 & -- & -- & 1.000 & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & 1.000 & 1.000 & -- & 1.000 & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & 0.500 & -- & -- & -- & 0.500 & 0.500 & -- & 0.500 & 0.500 \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & 0.500 & -- & -- & -- & 0.500 & 0.500 & -- & 0.500 & 0.500 \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & --
\end{pmatrix}
$$

$$M_{mcl}^{\infty}$$

- The attractors and the elements attracted by them are bound together in the same cluster.
- In the example above, the clusters are:
  - C_1 = {1, 6, 7, 10} (row 1)
  - C_2 = {2, 3, 5} (row 5)
  - C_3 = {4, 8, 9, 11, 12} (row 9 = row 11).

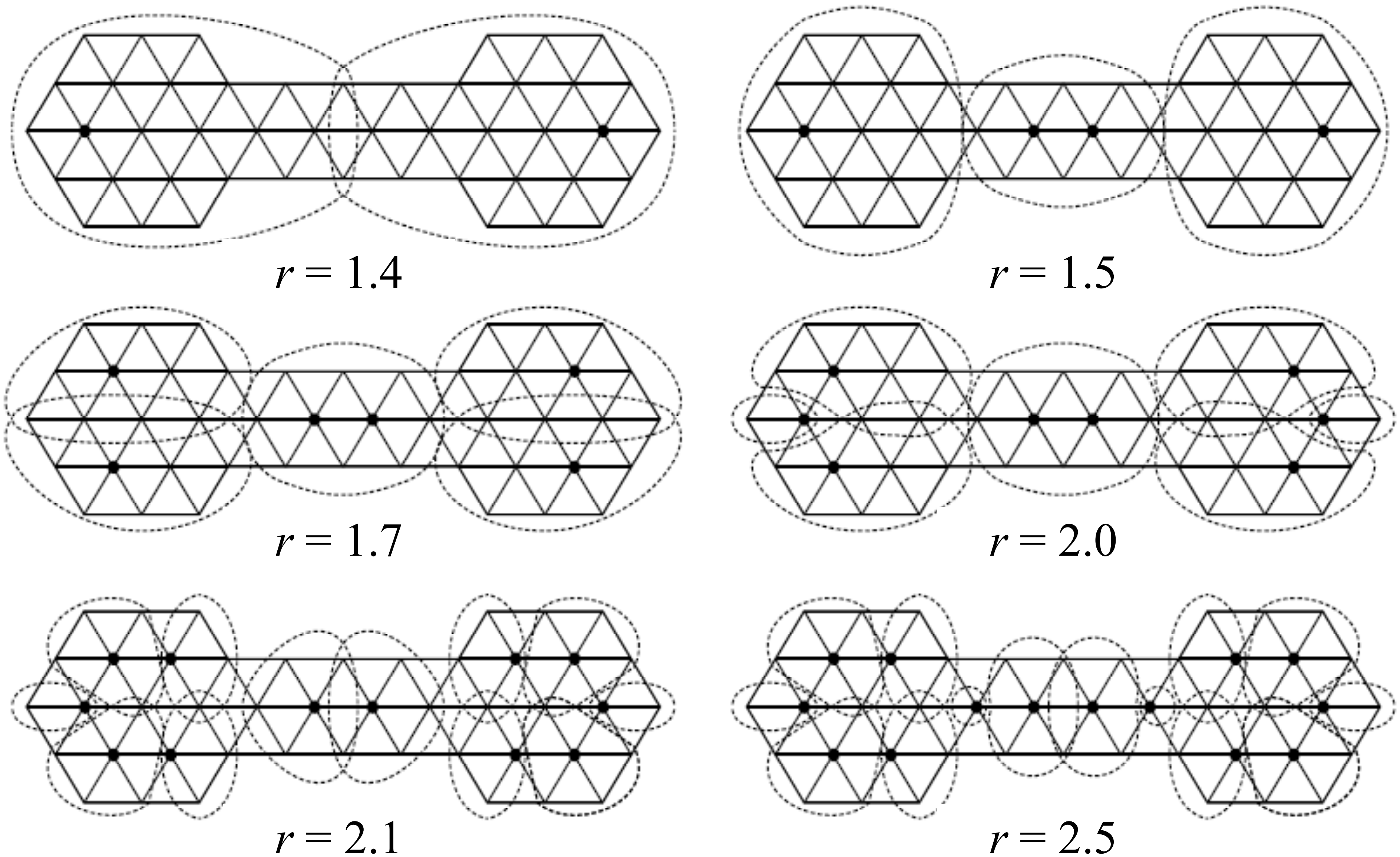
Interpretation of MCL Clusters

- In general, overlapping clusters (where one or more nodes belong to more than one cluster) result only in some special cases of symmetrical graphs:
  - only when a vertex (node of the graph) is attracted in an *exactly* equal way by more than one cluster;
  - this happens only when both clusters are isomorphic:



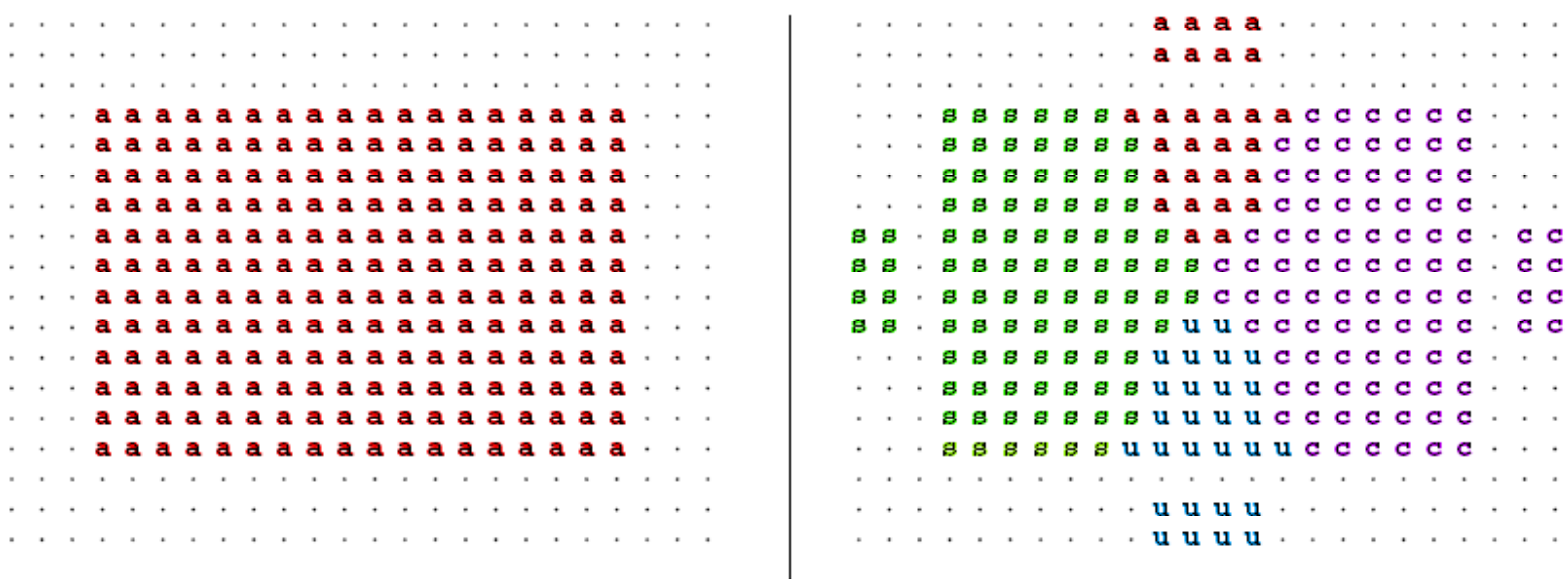
MCL Clusters

- The inflation parameter *r* affects the granularity of the clusters.
- In the following example, the weight of the self-loops is 1.



MCL Clusters

- For large diameter clusters, MCL has problems.
- Distributing the flow across clusters needs high expansion and low inflation (otherwise the cluster splits).
- This involves many iterations and makes MCL sensitive to small perturbations in the graph.
- Adding small diameter clusters disrupts clustering, as the low inflation parameter will cause them to disproportionately inflate the surrounding probabilities.



MCL Algorithm Analysis

- Processing time proportional to N^3, where N is the number of vertices:
    - N^3 is the cost of multiplying two matrices of order N;
    - inflation can be performed in a time proportional to N2;
    - the number of iterations required for convergence of the algorithm is not proven, but has been shown experimentally to be $\sim 10 \div 100$ steps, for most concerning scattered matrices after the first few steps.
- Processing speed can be improved by removing (*pruning*) unnecessary values:
    - by examining the matrix it is possible to set to zero the values that are small enough (it is assumed that they would become so at a certain step);
    - the algorithm works well when the diameter of the clusters is small (non-homogeneous distribution of weights).

MCL Algorithm Analysis

- Scales well as the size of the graph increases.
- Operates with weighted and unweighted graphs.
- Produces good clustering results.
- Robust with respect to the presence of noise in the graph data.
- Number of clusters not initially specified, but it is possible to adjust the granularity of clusters with parameters *e* and *r*.
- Generally unable to detect overlapping clusters.
- Not suitable for large diameter clusters.