The Structure of Sparse Random Bipartite Graphs

Reinhard Kutzelnigg*

Workshop on Discrete Mathematics
Vienna, November 22, 2008
based on joint work with Michael Drmota

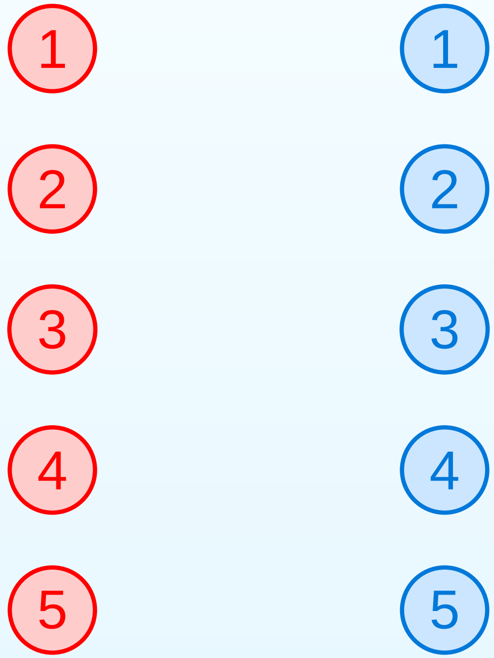
*Research supported by the Austrian Science Foundation FWF, project S9604 and by the EU FP6-NEST-Adventure Programme, Contract number 028875 (NEMO).

Random Bipartite Graphs

- We consider multigraphs with two types of labelled nodes.
- Each labelled edge connects nodes of different types and is chosen uniformly at random.
- We concentrate on (relatively) sparse graphs.

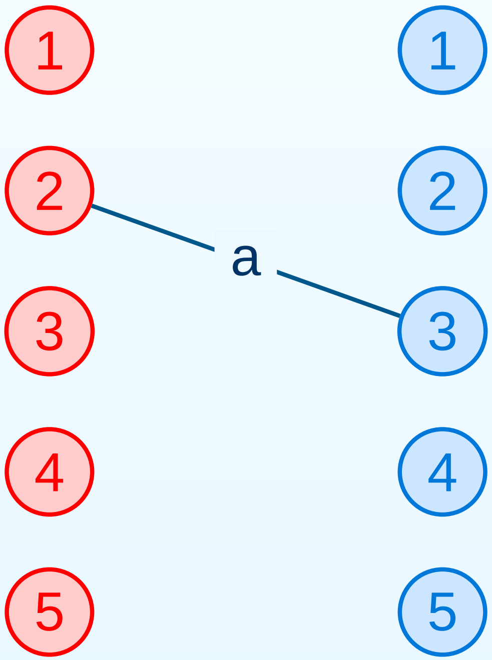
Random Bipartite Graphs

- We consider multigraphs with two types of labelled nodes.
- Each labelled edge connects nodes of different types and is chosen uniformly at random.
- We concentrate on (relatively) sparse graphs.



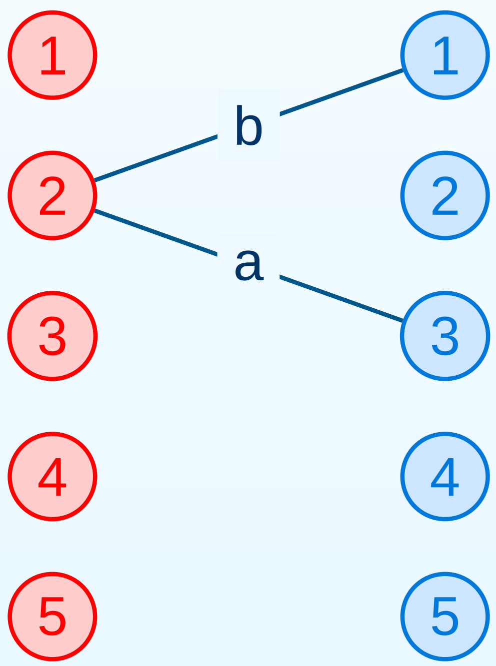
Random Bipartite Graphs

- We consider multigraphs with two types of labelled nodes.
- Each labelled edge connects nodes of different types and is chosen uniformly at random.
- We concentrate on (relatively) sparse graphs.



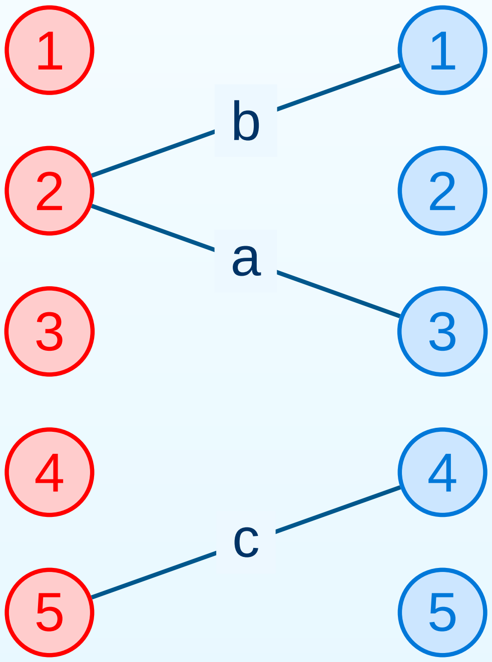
Random Bipartite Graphs

- We consider multigraphs with two types of labelled nodes.
- Each labelled edge connects nodes of different types and is chosen uniformly at random.
- We concentrate on (relatively) sparse graphs.



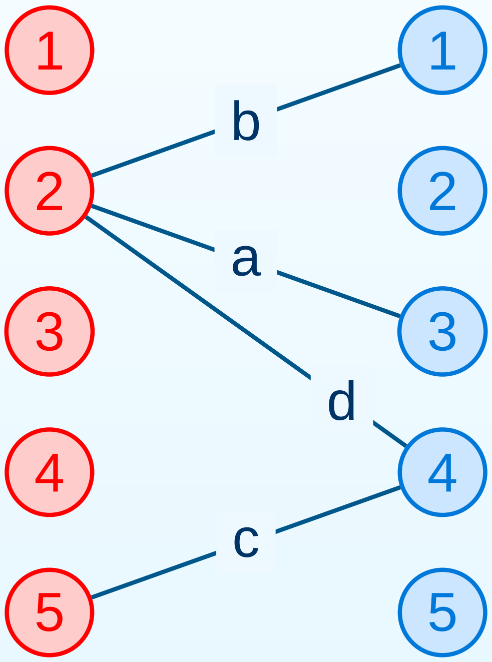
Random Bipartite Graphs

- We consider multigraphs with two types of labelled nodes.
- Each labelled edge connects nodes of different types and is chosen uniformly at random.
- We concentrate on (relatively) sparse graphs.



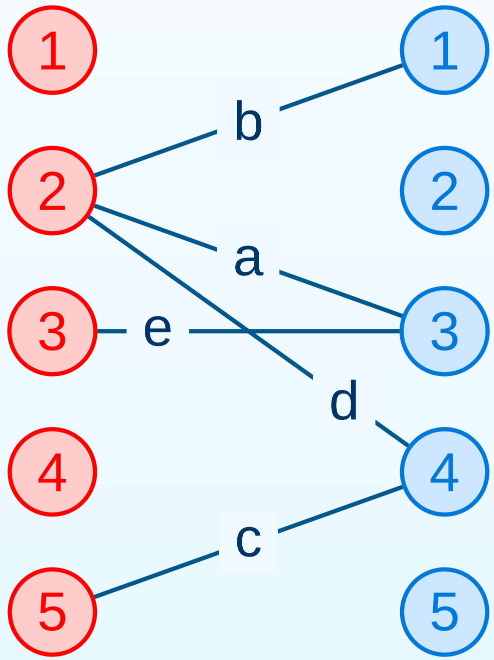
Random Bipartite Graphs

- We consider multigraphs with two types of labelled nodes.
- Each labelled edge connects nodes of different types and is chosen uniformly at random.
- We concentrate on (relatively) sparse graphs.



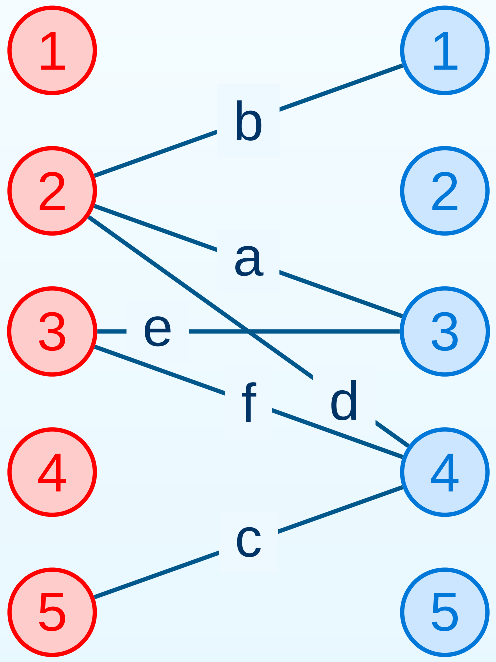
Random Bipartite Graphs

- We consider multigraphs with two types of labelled nodes.
- Each labelled edge connects nodes of different types and is chosen uniformly at random.
- We concentrate on (relatively) sparse graphs.



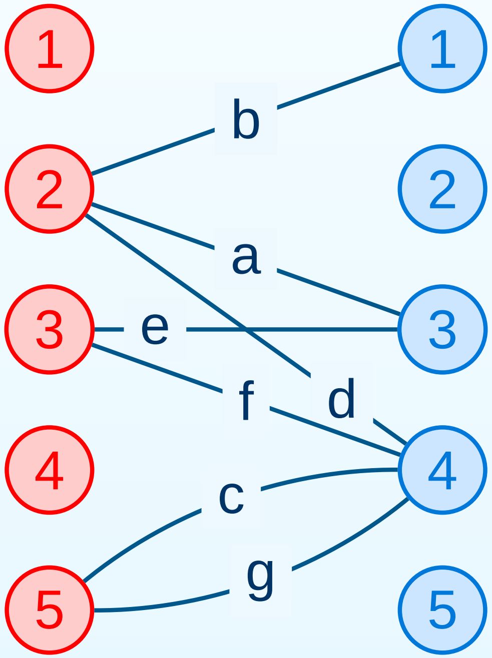
Random Bipartite Graphs

- We consider multigraphs with two types of labelled nodes.
- Each labelled edge connects nodes of different types and is chosen uniformly at random.
- We concentrate on (relatively) sparse graphs.



Random Bipartite Graphs

- We consider multigraphs with two types of labelled nodes.
- Each labelled edge connects nodes of different types and is chosen uniformly at random.
- We concentrate on (relatively) sparse graphs.



First Results

Consider a random bipartite graph G, consisting of m nodes of each type and n edges.

Theorem 1. *The probability, that G contains only tree and unicyclic components, provided that $n = \lfloor (1-\varepsilon)m \rfloor$ and $\varepsilon \in (0,1)$, equals*

$$1 - \frac{(2\varepsilon^2 - 5\varepsilon + 5)(1-\varepsilon)^3}{12(2-\varepsilon)^2\varepsilon^3}\frac{1}{m} + \mathcal{O}\left(\frac{1}{m^2}\right).$$

Theorem 2. *Assume m equals n. The probability, that G contains only tree and unicyclic components, equals $\sqrt{2/3} + o(1)$.*

- Devroye and Morin showed $1 - O(1/m)$ in 2001.
- The analytic structure of generating functions for bipartite random graphs is more difficult than that of usual random graphs.
- Nevertheless the results look the same, cf. Janson et al. 1993. Thus, one can expect that most properties of random graphs have a counterpart in random bipartite graphs (birth of giant component etc.).

Generating Functions

Bipartite Trees

- We use the generating functions $t_1(x, y)$ and $t_2(x, y)$ for bipartite rooted trees, where the root is contained in first respectively second subset of nodes.
- This generating functions are given by

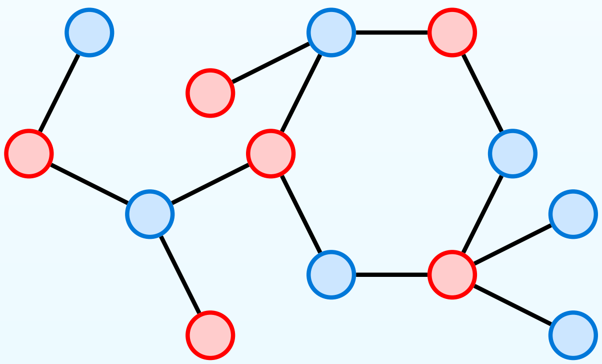
$$t_1(x, y) = xe^{t_2(x,y)}, \quad t_2(x, y) = ye^{t_1(x,y)}.$$

- Let $\tilde{t}(x, y)$ denote the generating function of unrooted bipartite trees.
- Furthermore, one can show:

$$\tilde{t}(x, y) = t_1(x, y) + t_2(x, y) - t_1(x, y)t_2(x, y)$$

Cyclic Components

- Of course, a cycle has to have an even number of nodes, say $2k$, where k nodes are of type 1 and the other k nodes of type 2.
- A cyclic node of type 1 can be considered as the root of a rooted tree of type 1 and similarly, for type 2.



- The product of the generating functions of this trees is divided by $2k$, to account for cyclic order and change of orientation.

- Consequently the generating function of a connected graph with exactly one cycle is given by

$$c(x,y) = \sum_{k\geq 1} \frac{1}{2k} t_1(x,y)^k t_2(x,y)^k$$
$$= \frac{1}{2} \log \frac{1}{1 - t_1(x,y)t_2(x,y)}.$$

Trees and Unicycles

- Let $g^\circ(x,y)$ denote the generating function of bipartite graphs consisting only of tree and unicyclic components.
- Note that such a graph possesses exactly $2m-n$ tree components.
- Hence, we get

$$g^\circ(x,y) = \frac{1}{(2m-n)!}\frac{\tilde{t}(x,y)^{2m-n}}{\sqrt{1-t_1(x,y)t_2(x,y)}}.$$

- We are interested in $[x^m y^m]g^\circ(x,y)$.

Asymptotic Analysis

$$[x^m y^m] g^\circ(x,y) = \frac{-(m!)^2}{4\pi(2m-n)!} \int_{|x|=x_0} \int_{|y|=y_0} \frac{\tilde{t}(x,y)^{2m-n}}{\sqrt{1-t_1(x,y)t_2(x,y)}} \frac{dx\,dy}{(xy)^{m+1}}$$

This is in fact an integral that can be asymptotically evaluated with help of a (double) saddle point method. It turns out, that if $n=(1-\varepsilon)m$ and $\varepsilon \in (0,1)$ is fixed, the saddle point is given by

$$x_0 = y_0 = \frac{n}{m} e^{-\frac{n}{m}} = (1-\varepsilon)e^{\varepsilon-1} < \frac{1}{e}.$$

The Saddle Point Method

Lemma 1. *$f(x,y)$ and $g(x,y)$ analytic functions in a ball around $(0,0)$ (+ technical assumptions):*

$$[x^{m_1}y^{m_2}]g(x,y)f(x,y)^k = \frac{g(x_0,y_0)f(x_0,y_0)^k}{2\pi x_0^{m_1}y_0^{m_2}k\sqrt{\Delta}}\left(1+\frac{h}{24\Delta^3}\frac{1}{k}+O\left(\frac{1}{k^2}\right)\right),$$

where x_0 and y_0 are uniquely defined by

$$\frac{m_1}{k}=\frac{x_0}{f(x_0,y_0)}\left[\frac{\partial}{\partial x}f(x,y)\right]_{(x_0,y_0)}$$

$$\frac{m_2}{k}=\frac{y_0}{f(x_0,y_0)}\left[\frac{\partial}{\partial y}f(x,y)\right]_{(x_0,y_0)}.$$

(m_1, m_2, and k have to be of the same order of magnitude)

Generally, let the cummulants κ_{ij} and $\overline{\kappa}_{ij}$ be

$$\kappa_{ij} = \left[\frac{\partial^i}{\partial u^i}\frac{\partial^j}{\partial v^j}\log f(x_0 e^u, y_0 e^v)\right]_{(0,0)}$$

$$\overline{\kappa}_{ij} = \left[\frac{\partial^i}{\partial u^i}\frac{\partial^j}{\partial v^j}\log g(x_0 e^u, y_0 e^v)\right]_{(0,0)}.$$

Further let $\Delta = \kappa_{20}\kappa_{02} - \kappa_{11}^2$, then h is a constant depending on $\kappa_{02}, \kappa_{11}, \kappa_{20}, \kappa_{03}, \kappa_{12}, \kappa_{21}, \kappa_{30}, \overline{\kappa}_{01}, \overline{\kappa}_{10}, \overline{\kappa}_{02}, \overline{\kappa}_{11}$, and $\overline{\kappa}_{20}$.

The “critical” case

- Here, we consider the special case $\varepsilon = 0$.
- The proof of Theorem 2 follows the same idea.
- However, the saddle point $x_0 = y_0 = 1/e$ coalesces with the singularity of the denominator of

$$\frac{\tilde{t}(x,y)^{2m-n}}{\sqrt{1 - t_1(x,y)t_2(x,y)}}.$$

- We use the following series of $\#G^{\circ}_{m,m,m}$ and consider each summand separately,

$$(m!)^2 \sum_{k\geq 0} \binom{2k}{k} \frac{1}{4^k} [x^m y^m] \tilde{t}(x,y)^m t_1(x,y)^k t_2(x,y)^k.$$

- Using Lagrange's Inversion Theorem, we get

$$[x^m y^m]\tilde{t}(x,y)^m t_1(x,y)^k t_2(x,y)^k = \frac{1}{m}[u^m y^m] f(u,y)^m l(u,y) h(u,y).$$

Hereby, we use the following functions:

$$f(u,y) = (u + ye^u(1-u)) \exp(ye^u),$$
$$l(u,y) = u^k (ye^u)^k,$$
$$h(u,y) = u\frac{mu - mye^u u^2 + ku + kye^u + ku^2 - ku^2 ye^u}{u(u + ye^u(1-u))}.$$

- The saddle point now equals $u_0 = 1, y_0 = 1/e$.
- We have to handle the integral

$$\int_0^\infty s e^{-\frac{2}{3}s^3 + \frac{2\zeta}{\sqrt[3]{m}} ks} dt\, ds.$$

- This function is related to the Lommel function of second kind, that is a solution of the inhomogeneous Bessel differential equation

$$x^2 \frac{d^2y}{dx^2} + x\frac{dy}{dx} + (x^2 - \nu^2)y = x^{\mu+1}.$$

The Component Structure

Number of Cycles

Suppose that $\varepsilon \in (0,1)$ is fixed and that $n = \lfloor (1-\varepsilon)m \rfloor$. Then a labelled random bipartite multigraph with $2 \times m$ vertices and n edges satisfies the following properties:

- The number of unicyclic components with cycle length $2k$ has in limit a Poisson distribution $Po(\lambda_k)$ with parameter

$$\lambda_k = \frac{1}{2k} (1-\varepsilon)^{2k},$$

and the number of unicyclic components has in limit a Poisson distribution $Po(\lambda)$, too, with parameter

$$\lambda = -\frac{1}{2} \log\left(1 - (1-\varepsilon)^2\right).$$

Trees with fixed size

- Denote the number of tree components with k vertices by t_k. Mean and variance of this random variable are asymptotically equal to

$$m\mu = 2m\frac{k^{k-2}(1-\varepsilon)^{k-1}e^{k(\varepsilon-1)}}{k!},$$

respectively

$$m\sigma^2 = m\mu - \frac{2me^{2k(\varepsilon-1)}k^{2k-4}(1-\varepsilon)^{2k-3}(k^2\varepsilon^2+k^2\varepsilon-4k\varepsilon+2)}{(k!)^2}.$$

Furthermore t_k satisfies a central limit theorem of the form

$$\frac{t_k-\mu}{\sigma} \to N(0,1).$$

Nodes in all cyclic Components

- Furthermore, the expected value of the number of nodes in unicyclic components is asymptotically given by

$$\frac{(1-\varepsilon)^2}{\varepsilon\left(1-(1-\varepsilon)^2\right)},$$

and its variance by

$$\frac{(1-\varepsilon)^2(\varepsilon^2-3\varepsilon+4)}{\varepsilon^2\left(1-(1-\varepsilon)^2\right)^2}.$$

Remarks

- Because of Theorem 1, it is sufficient to consider graphs that contain tree and unicylic components only. Recall the corresponding generating function

$$g^{\circ}(x,y) = \frac{\tilde{t}(x,y)^{2m-n}}{(2m-n)!}\exp(c(x,y)),$$

where $c(x,y)$ denotes the generating function of an unicyclic component.

- Similar results hold for “usual” random graphs too.

Proof: Step 1

Introduce a “new” Variable w to mark the Parameter of interest:

- Number of cycles

$$g_1^\circ(x,y,w) = \frac{\tilde{t}(x,y)^{2m-n}}{(2m-n)!}\exp(wc(x,y))$$

- Trees possessing k nodes

$$g_2^\circ(x,y,w) = \frac{\left(\tilde{t}(x,y) + (w-1)\tilde{t}_k(x,y)\right)^{2m-n}}{(2m-n)!}\exp(c(x,y))$$

- Nodes in all cyclic Components

$$g_3^\circ(x,y,w) = \frac{\tilde{t}(x,y)^{2m-n}}{(2m-n)!}\exp(c(wx,wy))$$

Proof: Step 2

Calculate the l-th factorial Moment

$$\mathcal{M}_l = \frac{[x^m y^m]\left[\frac{\partial^l}{\partial w^l} g^\circ(x,y,w)\right]_{w=1}}{[x^m y^m] g_t^\circ(x,y,1)},$$

or the characteristic function

$$\phi(s) = \frac{[x^m y^m] g^\circ(x,y,e^{is})}{[x^m y^m] g_t^\circ(x,y,1)}.$$

The calculation itself is again performed using the saddle point method.

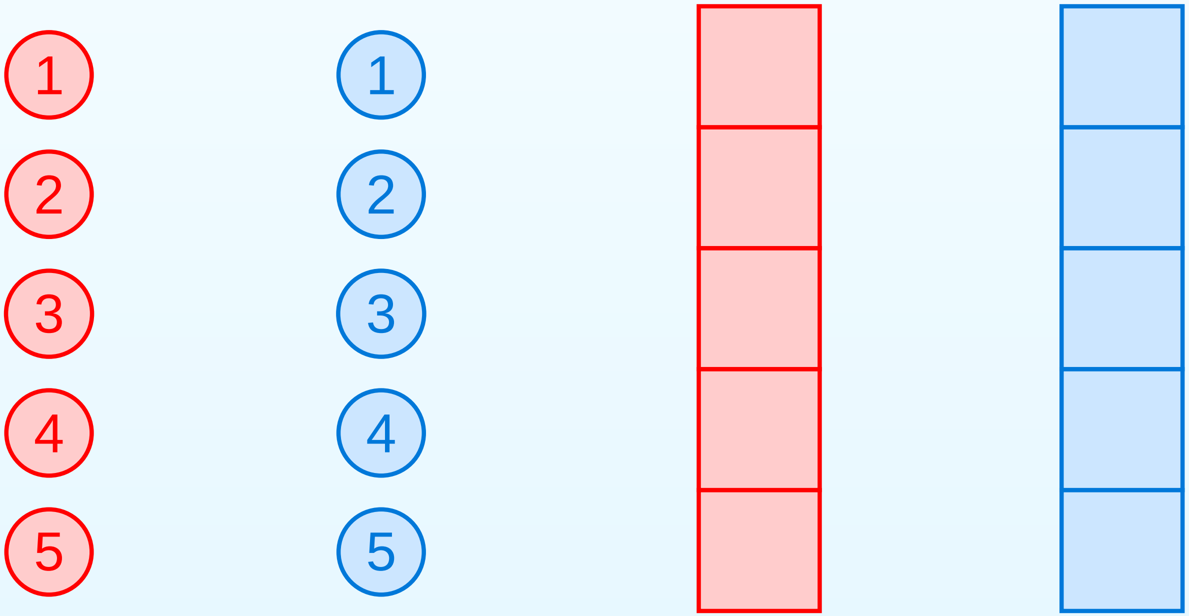
Application

Cuckoo Hashing

- Hash table data structure introduced by Pagh and Rodler in 2001.
- Offers constant worst case search time.
- Uses two tables and two different hash functions h_1 and h_2, both determine a unique position in each table.
- Resolve conflicts by rearranging keys.
- Algorithm can be modelled by a random bipartite graph.

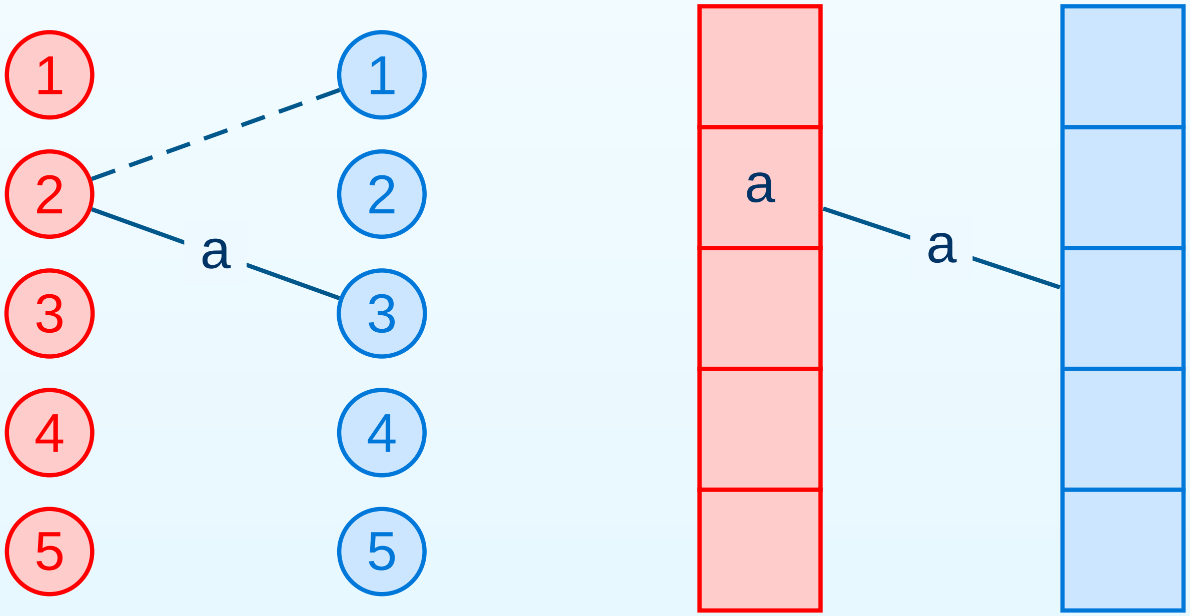
Cuckoo Hashing

- Hash table data structure introduced by Pagh and Rodler in 2001.
- Offers constant worst case search time.
- Uses two tables and two different hash functions h_1 and h_2, both determine a unique position in each table.
- Resolve conflicts by rearranging keys.
- Algorithm can be modelled by a random bipartite graph.



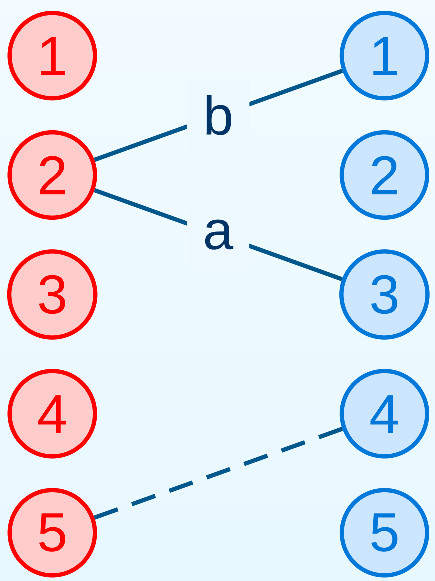
Cuckoo Hashing

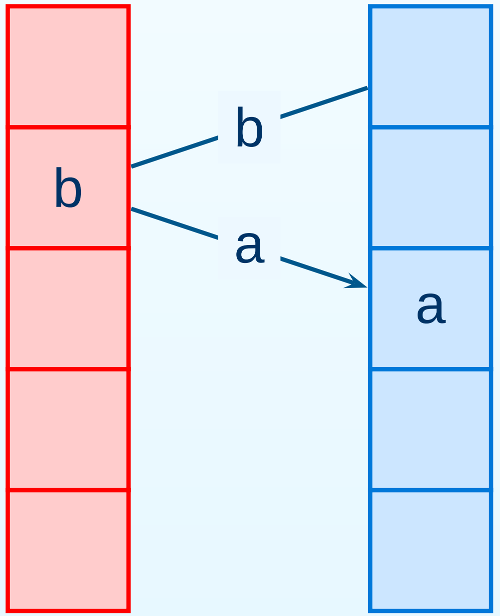
- Hash table data structure introduced by Pagh and Rodler in 2001.
- Offers constant worst case search time.
- Uses two tables and two different hash functions h_1 and h_2, both determine a unique position in each table.
- Resolve conflicts by rearranging keys.
- Algorithm can be modelled by a random bipartite graph.



Cuckoo Hashing

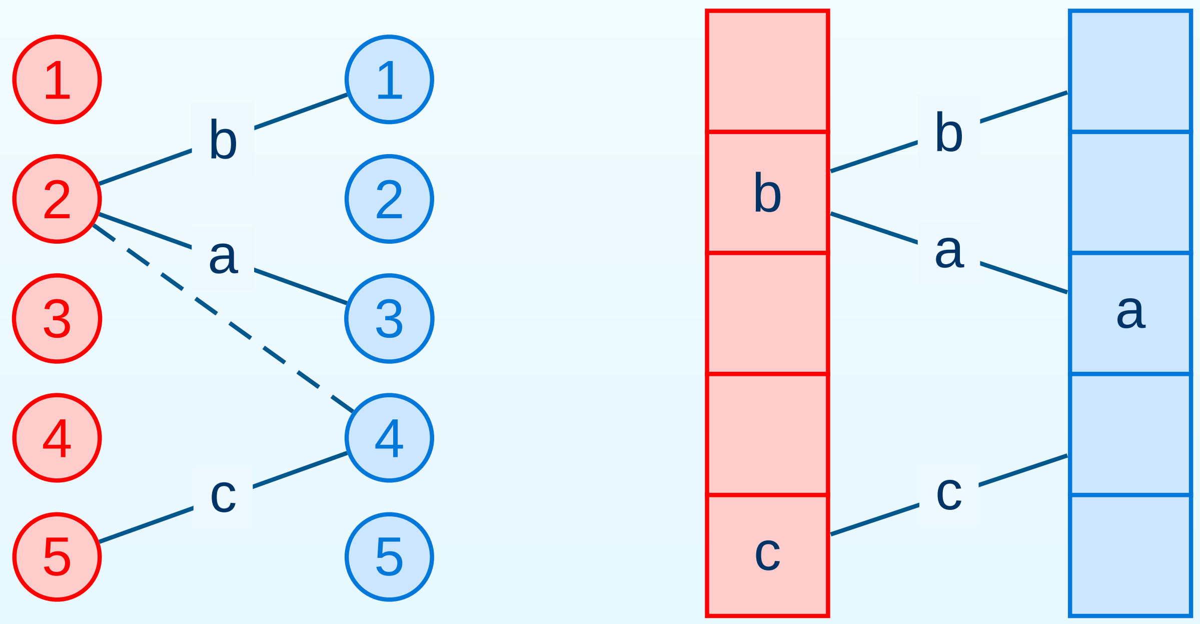
- Hash table data structure introduced by Pagh and Rodler in 2001.
- Offers constant worst case search time.
- Uses two tables and two different hash functions h_1 and h_2, both determine a unique position in each table.
- Resolve conflicts by rearranging keys.
- Algorithm can be modelled by a random bipartite graph.





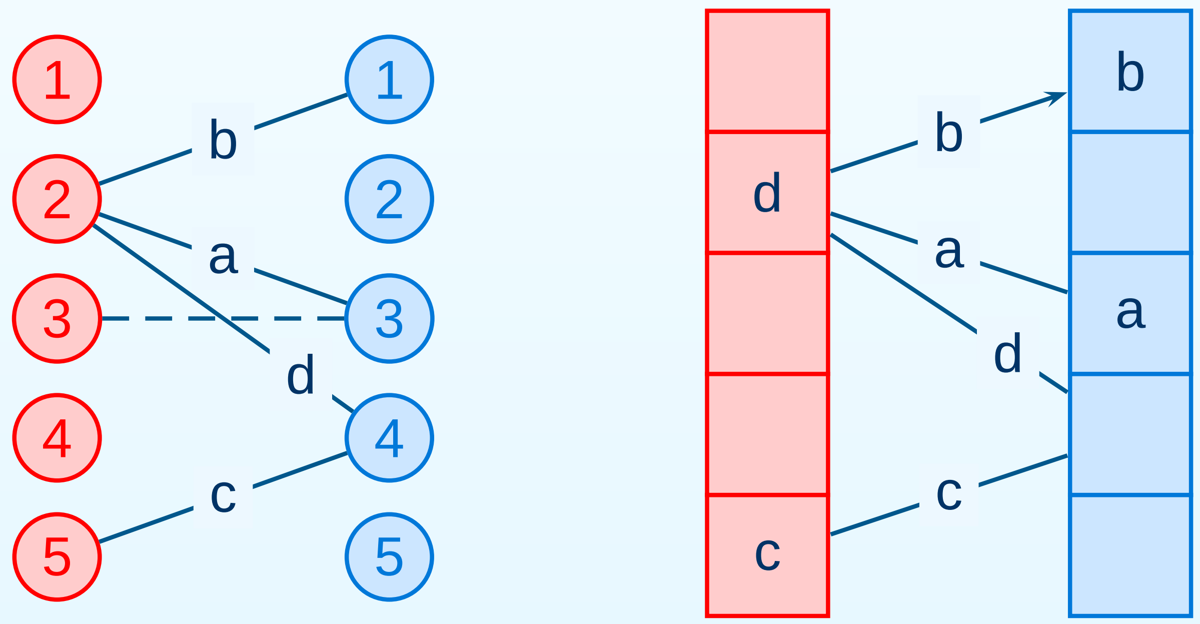
Cuckoo Hashing

- Hash table data structure introduced by Pagh and Rodler in 2001.
- Offers constant worst case search time.
- Uses two tables and two different hash functions h_1 and h_2, both determine a unique position in each table.
- Resolve conflicts by rearranging keys.
- Algorithm can be modelled by a random bipartite graph.



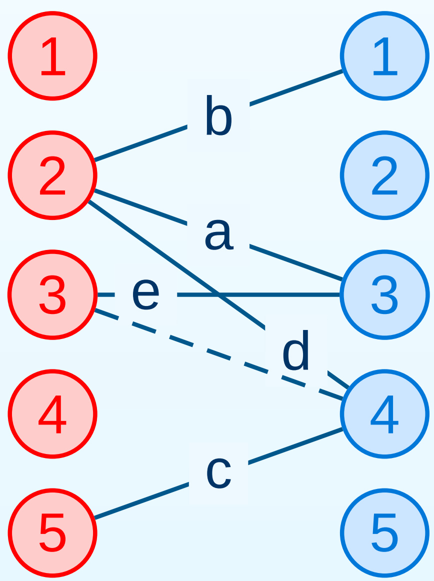
Cuckoo Hashing

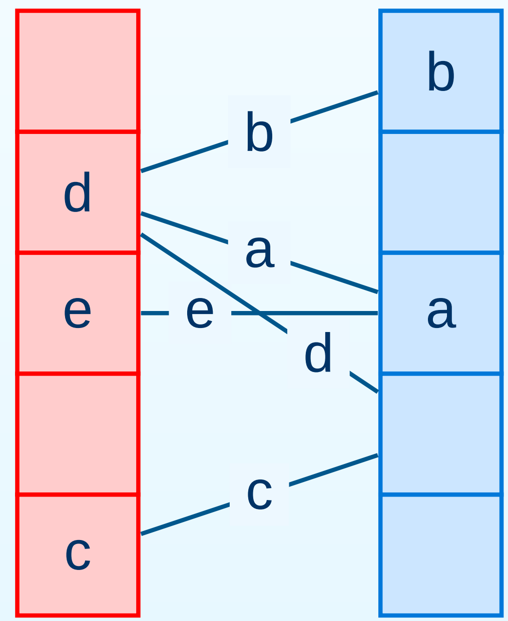
- Hash table data structure introduced by Pagh and Rodler in 2001.
- Offers constant worst case search time.
- Uses two tables and two different hash functions h_1 and h_2, both determine a unique position in each table.
- Resolve conflicts by rearranging keys.
- Algorithm can be modelled by a random bipartite graph.



Cuckoo Hashing

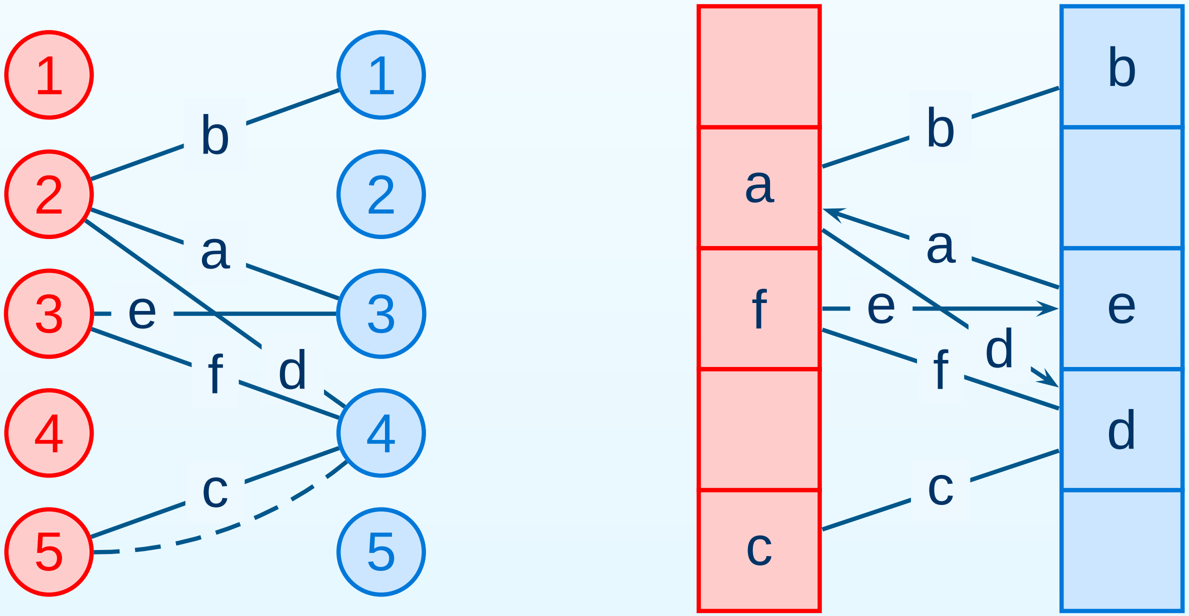
- Hash table data structure introduced by Pagh and Rodler in 2001.
- Offers constant worst case search time.
- Uses two tables and two different hash functions h_1 and h_2, both determine a unique position in each table.
- Resolve conflicts by rearranging keys.
- Algorithm can be modelled by a random bipartite graph.





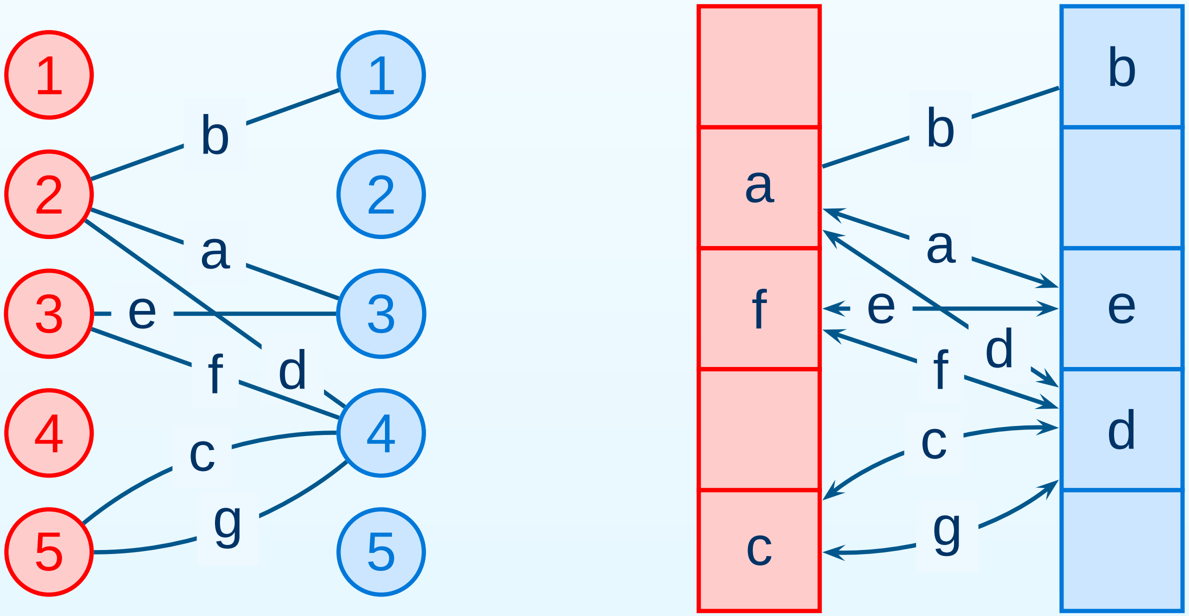
Cuckoo Hashing

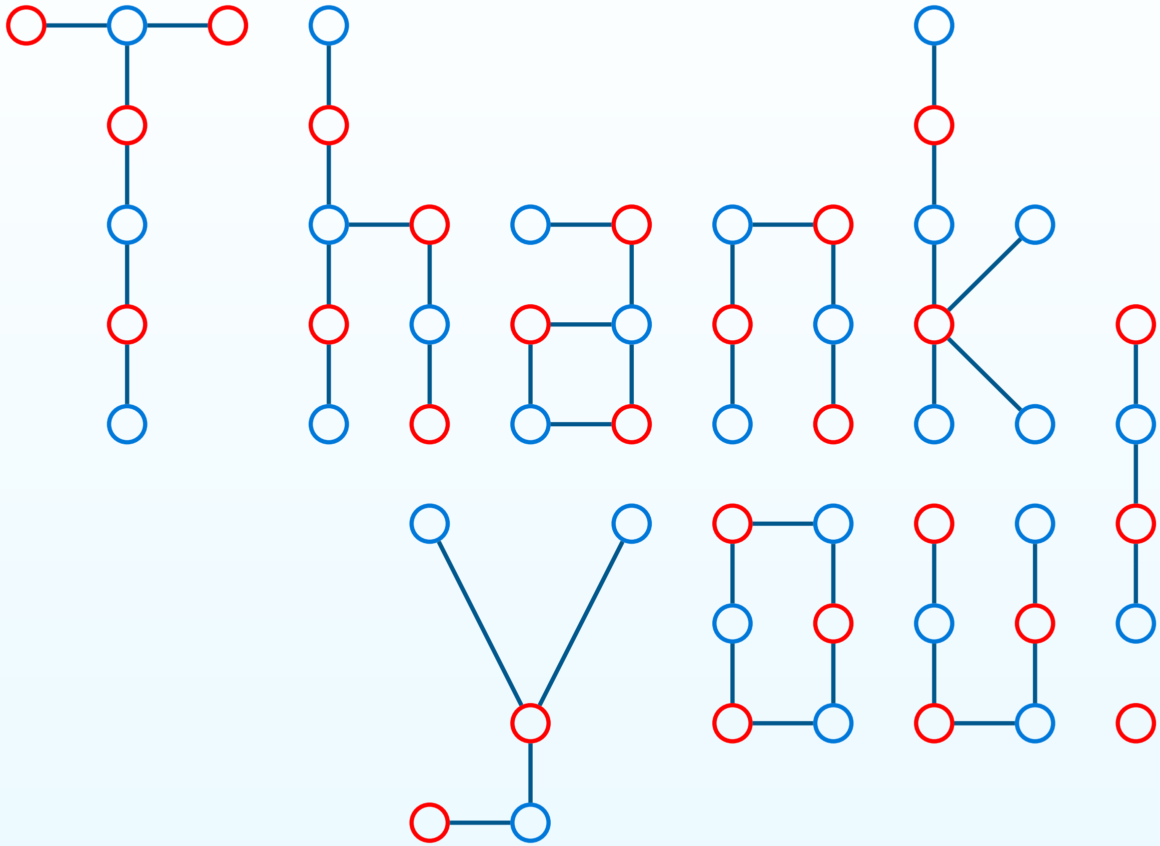
- Hash table data structure introduced by Pagh and Rodler in 2001.
- Offers constant worst case search time.
- Uses two tables and two different hash functions h_1 and h_2, both determine a unique position in each table.
- Resolve conflicts by rearranging keys.
- Algorithm can be modelled by a random bipartite graph.



Cuckoo Hashing

- Hash table data structure introduced by Pagh and Rodler in 2001.
- Offers constant worst case search time.
- Uses two tables and two different hash functions h_1 and h_2, both determine a unique position in each table.
- Resolve conflicts by rearranging keys.
- Algorithm can be modelled by a random bipartite graph.




Thank you!

References (1)

L. Devroye and P. Morin. Cuckoo hashing: Further analysis. *Information Processing Letters*, 86(4):215–219, 2003.

Michael Drmota. A bivariate asymptotic expansion of coefficients of powers of generating functions. *European Journal of Combinatorics*, 15(2):139–152, 1994.

Dimitris Fotakis, Rasmus Pagh, Peter Sanders, and Paul G. Spirakis. Space efficient hash tables with worst case constant access time. *Theory Comput. Syst.*, 38(2):229–248, 2005.

Svante Janson, Donald E. Knuth, Tomasz Łuczak, and Boris Pittel. The birth of the giant component. *Random Structures and Algorithms*, 4(3):233–359, 1993.

I. B. Kalugin. The number of components of a random bipartite graph. *Discrete Mathematics and Applications*, 1(3):289–299, 1991.

References (2)

Donald E. Knuth. *The Art of Computer Programming, Volume III: Sorting and Searching*. Addison-Wesley, Boston, second edition, 1998.

R. Kutzelnigg. Bipartite random graphs and cuckoo hashing. In *Proceedings of the 4th Colloquium on Mathematics and Computer Science*, DMTCS, pages 403–406, 2006.

R. Kutzelnigg. An improved version of cuckoo hashing: Average case analysis of construction cost and search operations. In *Proceedings of the 19th IWOCA*, pages 253–266, 2008.

Reinhard Kutzelnigg. *Random Bipartite Graphs and their Application to Cuckoo Hashing*. PhD thesis, Vienna University of Technology, 2008.

R. Pagh and F. F. Rodler. Cuckoo hashing. *Journal of Algorithms*, 51(2):122–144, 2004.