Small-World Phenomena and Decentralized Search

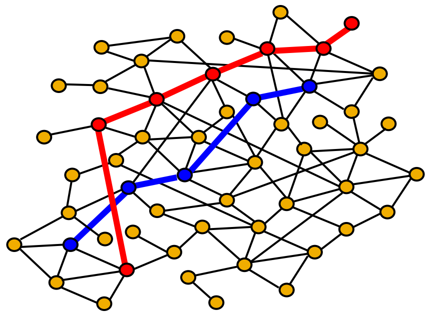
CS224W: Social and Information Network Analysis

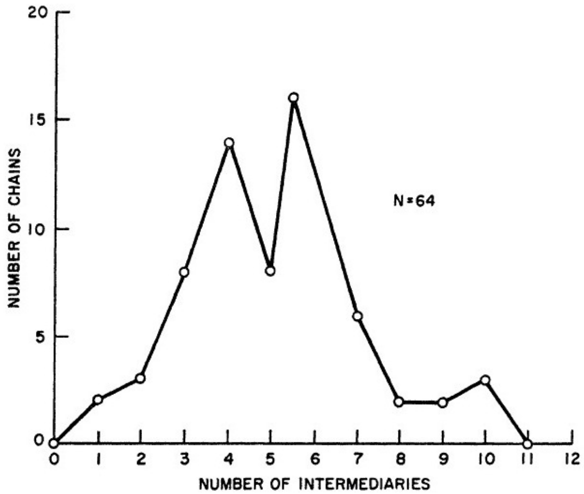
Jure Leskovec, Stanford University

http://cs224w.stanford.edu

The Small-World Experiment

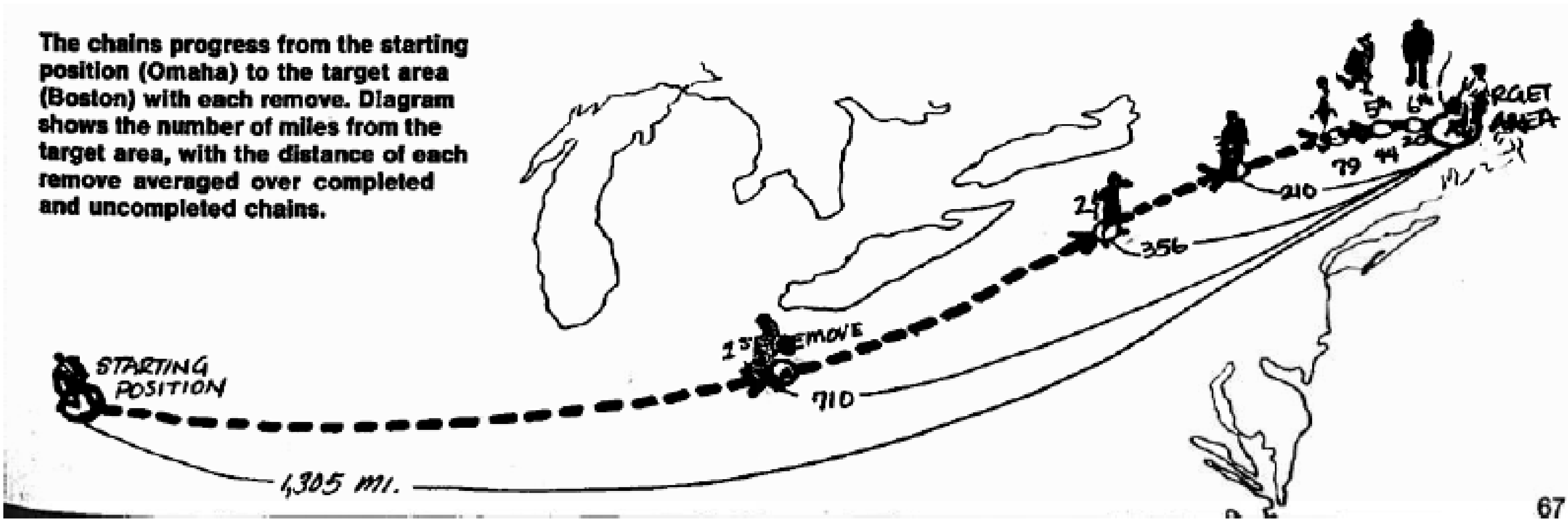
- What is the typical shortest path length between any two people?
  - Experiment on the global soc. network
    - Can’t measure, need to probe explicitly
- Small-world experiment [Milgram ’67]
  - Picked 300 people in Omaha, Nebraska and Wichita, Kansas
  - Task: Get a letter to a Boston stock-broker by passing it through friends
- **How many steps did it take?**
  - It took 6.2 steps on the average, thus **“6 degrees of separation”**





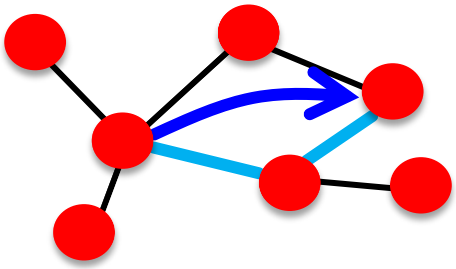
Two Questions

- **(1) What is the structure of a social network?**
- (2) Which mechanisms do people use to route and find the target?

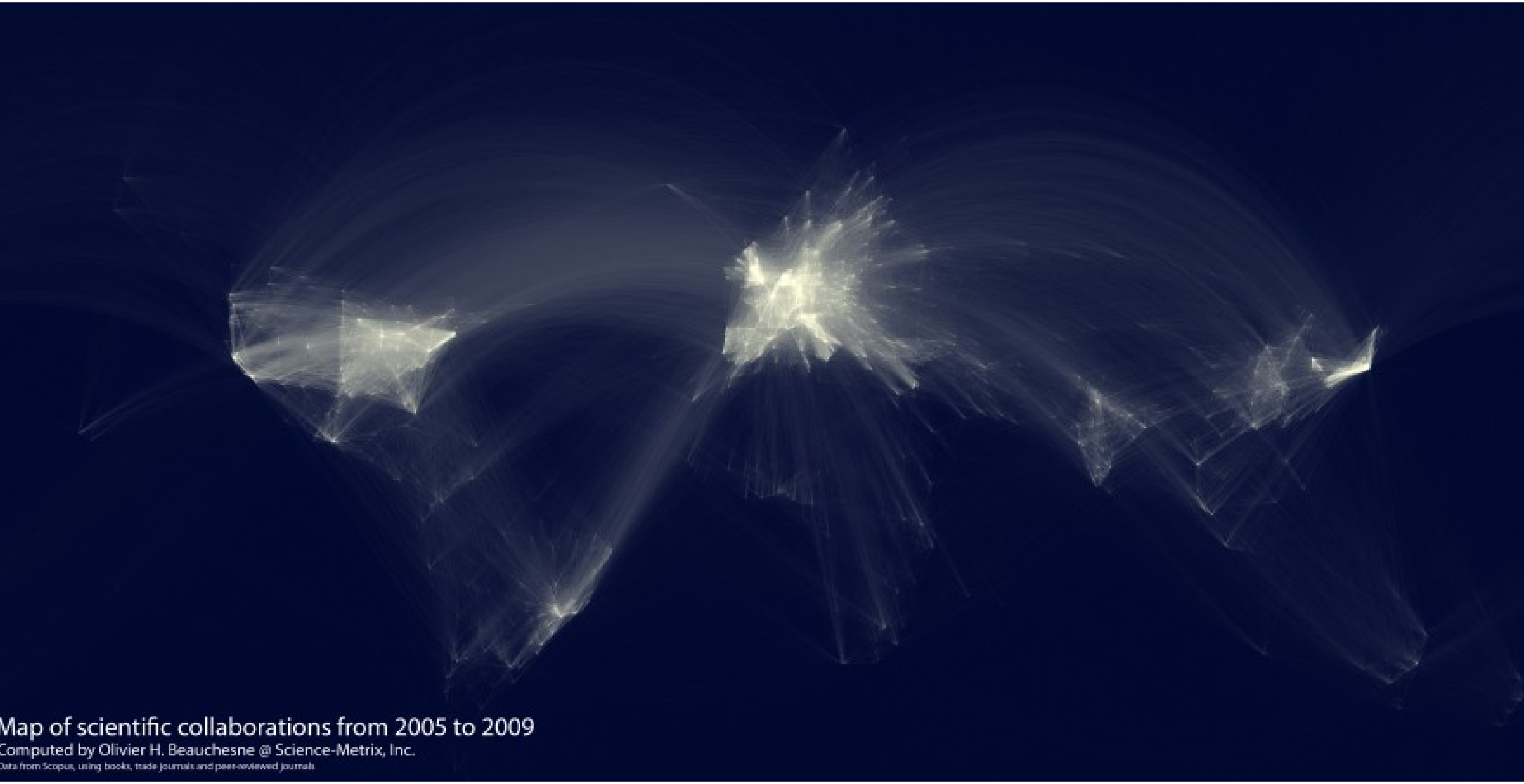


6-Degrees: Should We Be Surprised?

- Assume each human is connected to 100 other people. **Then:**
  - Step 1: reach 100 people
  - Step 2: reach 100*100 = 10,000 people
  - Step 3: reach 100*100*100 = 1,000,000 people
  - Step 4: reach 100*100*100*100 = 100M people
  - **In 5 steps we can reach 10 billion people**
- **What's wrong here?**
  - **92% of new FB friendships are to a friend-of-a-friend**



Scientific Collaborations



Clustering Implies Edge Locality

- **MSN network has 7 orders of magnitude larger clustering than the corresponding G_{np}!**
- **Other examples:**

Actor Collaborations (IMDB): 225,226 nodes, avg. degree k=61
Electrical power grid: 4,941 nodes, k=2.67
Network of neurons 282 nodes, k=14

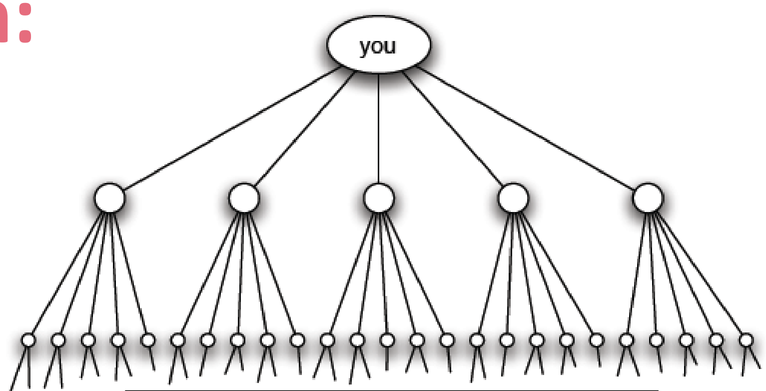
Table 1 Empirical examples of small-world networks

	L_{actual}	L_{random}	C_{actual}	C_{random}
Film actors	3.65	2.99	0.79	0.00027
Power grid	18.7	12.4	0.080	0.005
C. elegans	2.65	2.25	0.28	0.05

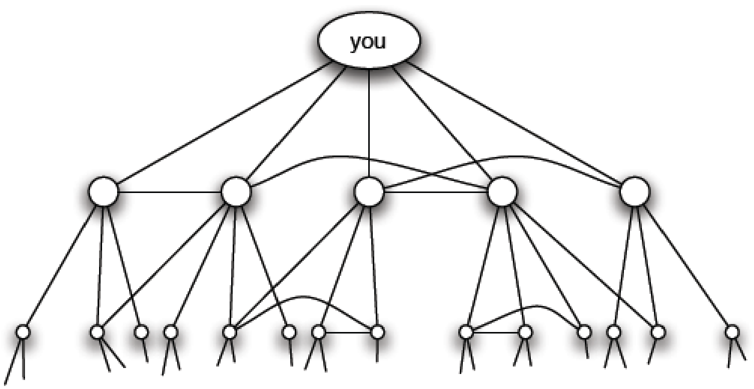
L ... Average shortest path length
C ... Average clustering coefficient

Back to the Small-World

- **Consequence of expansion:**
  - Short paths: $O(\log n)$
    - This is the "best" we can do if the graph has constant degree and n nodes
- **But networks have local structure:**
  - Triadic closure:
    Friend of a friend is my friend
- **How can we have both?**



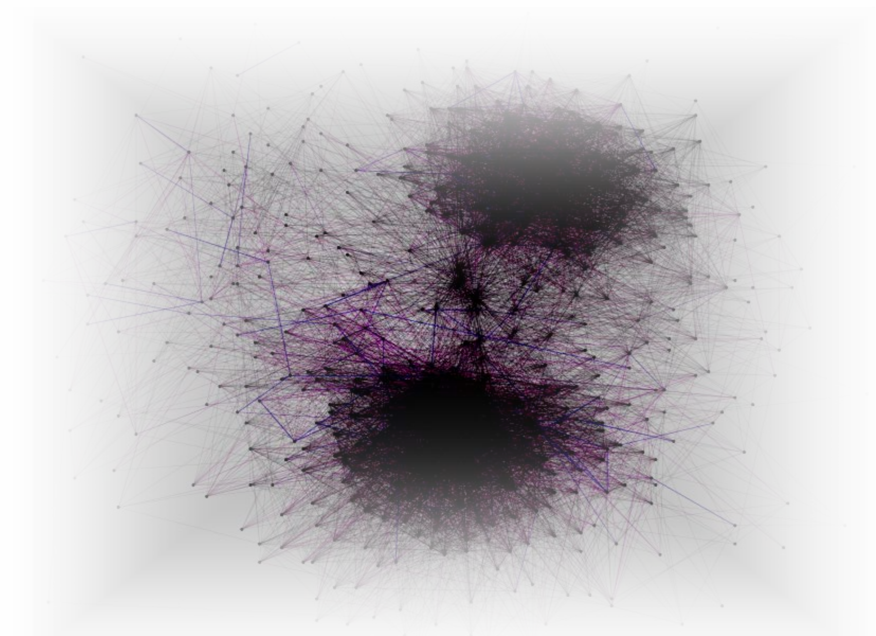
Pure exponential growth



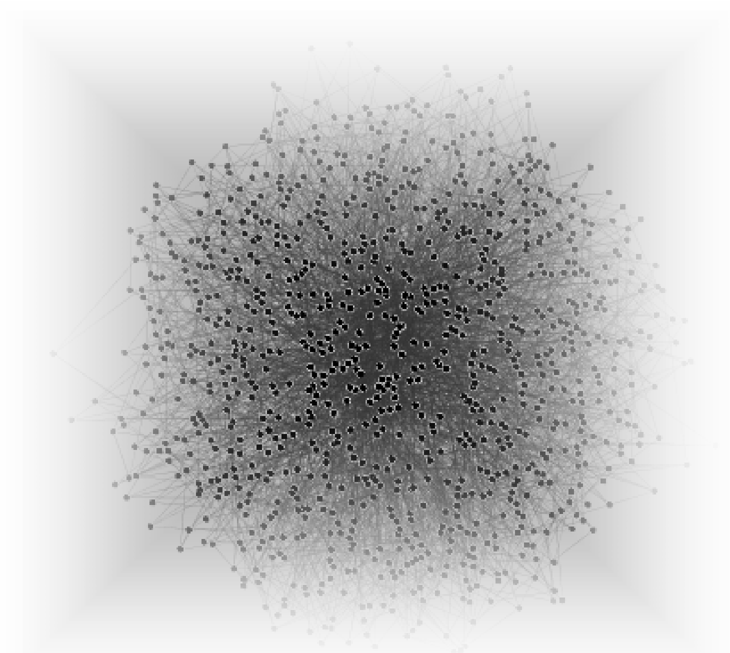
Triadic closure reduces growth rate

Clustering vs. Randomness

Where should we place social networks?



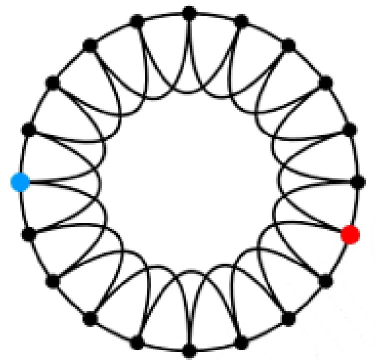
Clustered?



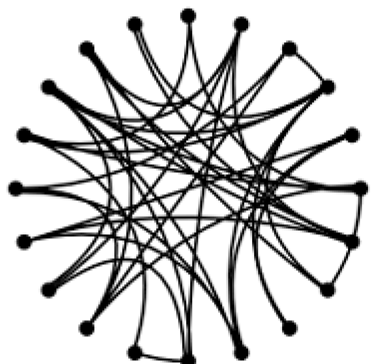
Random?

Small-World: How?

- **Could a network with high clustering be at the same time a small world?**
  - How can we at the same time have **high clustering** and **small diameter?**



High clustering
High diameter



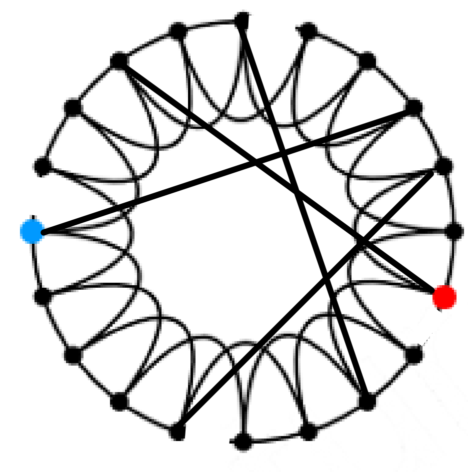
Low clustering
Low diameter

  - Clustering implies edge "locality"
  - Randomness enables "shortcuts"

Solution: The Small-World Model

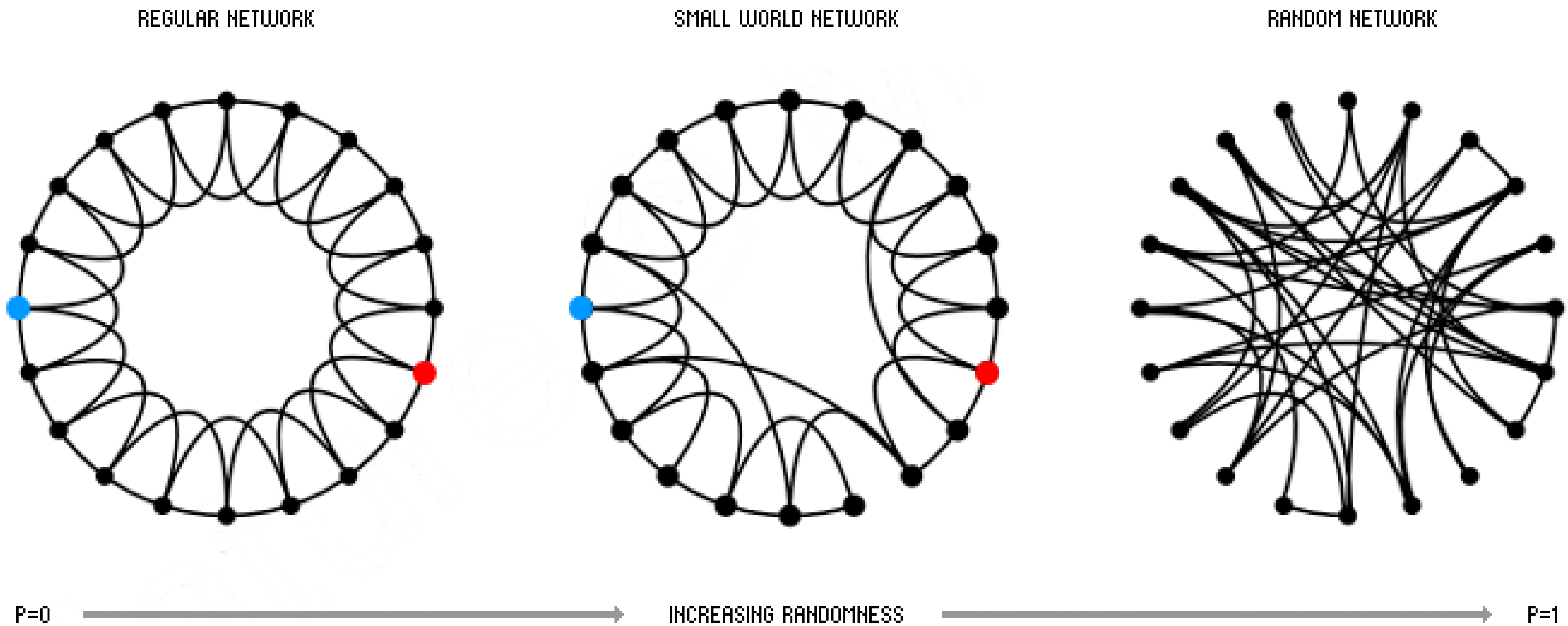
Small-world Model [Watts-Strogatz '98]:
2 components to the model:

- **(1)** Start with a **low-dimensional regular lattice**
  - Has high clustering coefficient
- Now introduce randomness ("shortcuts")
- **(2)** **Rewire:**
  - Add/remove edges to create shortcuts to join remote parts of the lattice
  - For each edge with prob. p move the other end to a random node



[Watts-Strogatz Nature '98]

The Small-World Model



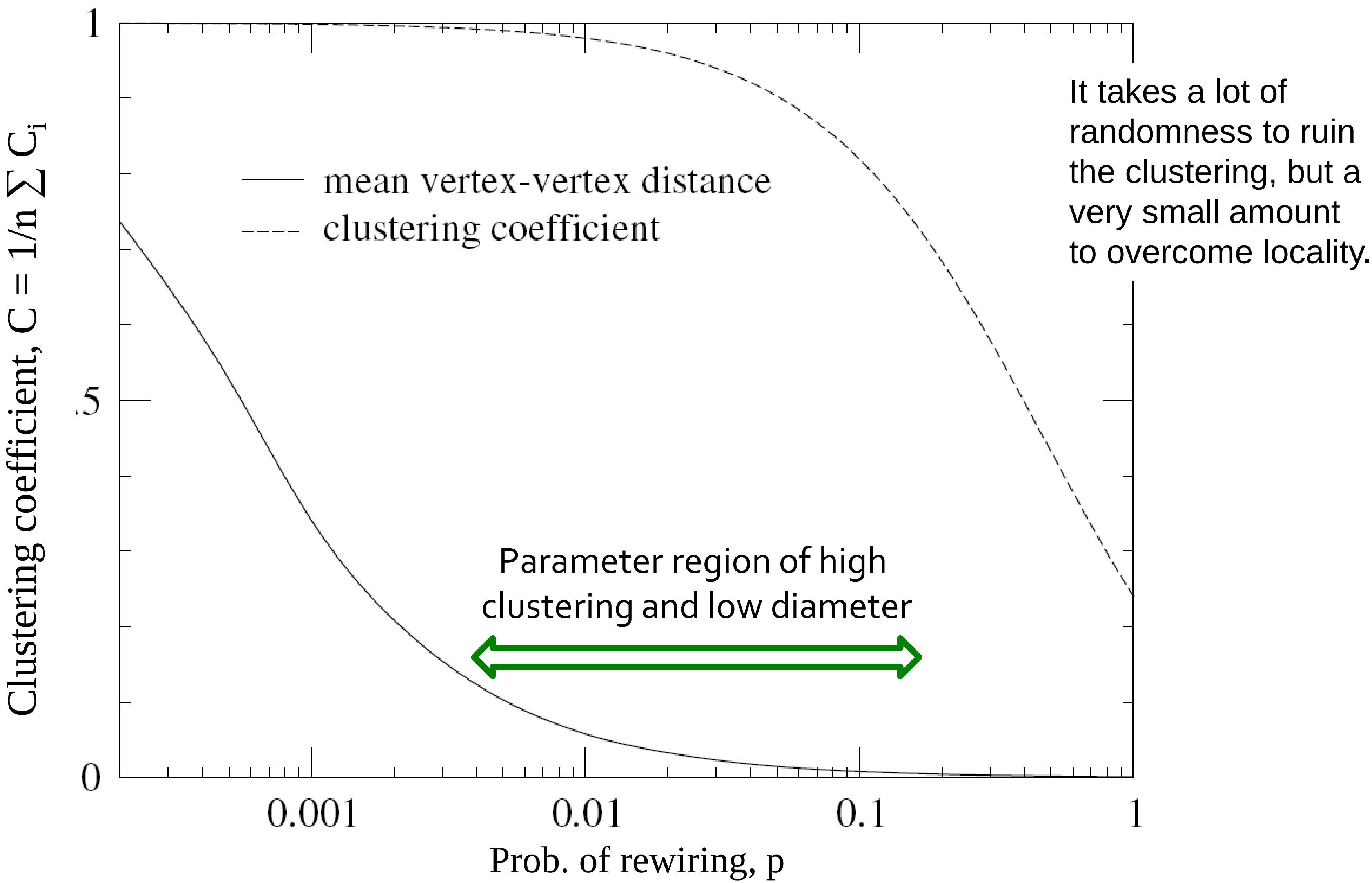
High clustering High diameter	High clustering Low diameter	Low clustering Low diameter

$$h = \frac{N}{2\bar{k}} \qquad C = \frac{3}{4}$$

$$h = \frac{\log N}{\log \alpha} \qquad C = \frac{\bar{k}}{N}$$

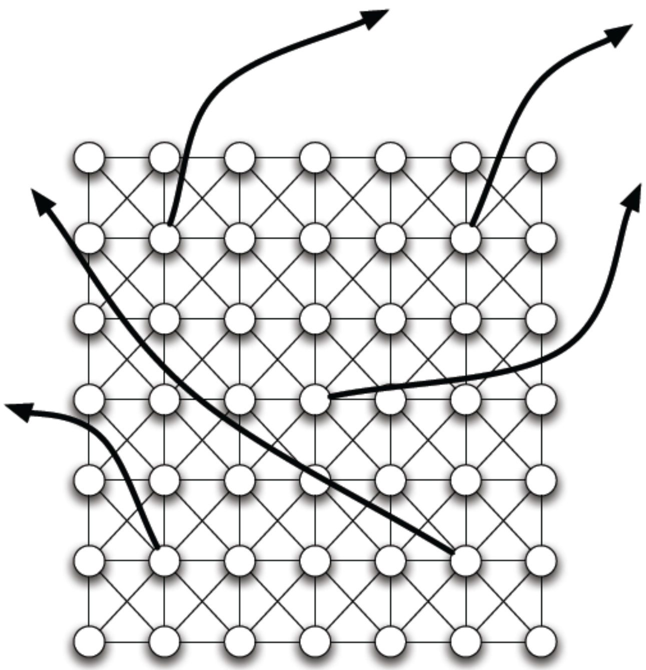
Rewiring allows us to interpolate between regular lattice and a random graph

The Small-World Model



Diameter of the Watts-Strogatz

- **Alternative formulation of the model:**
  - Start with a square grid
  - Each node has 1 random long-range edge
    - Each node has 1 spoke. Then randomly connect them.



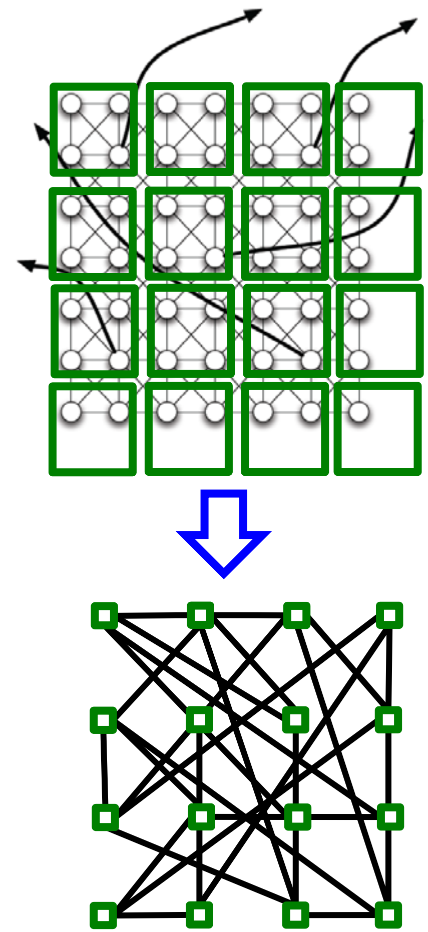
$C_i \geq 2*12/(8*7) \geq 0.43$

What's the diameter?
It is *log(n)*
Why?

Diameter of the Watts-Strogatz

- Proof:
  - Consider a graph where we contract 2x2 subgraphs into supernodes
  - Now we have 4 edges sticking out of each supernode
    - **4-regular random graph!**
  - From Thm. we have short paths between super nodes
  - We can turn this into a path in a real graph by adding at most 2 steps per hop

$\Rightarrow$ **Diameter of the model is** $O(2 \log n)$ i.e. **short paths exist!**



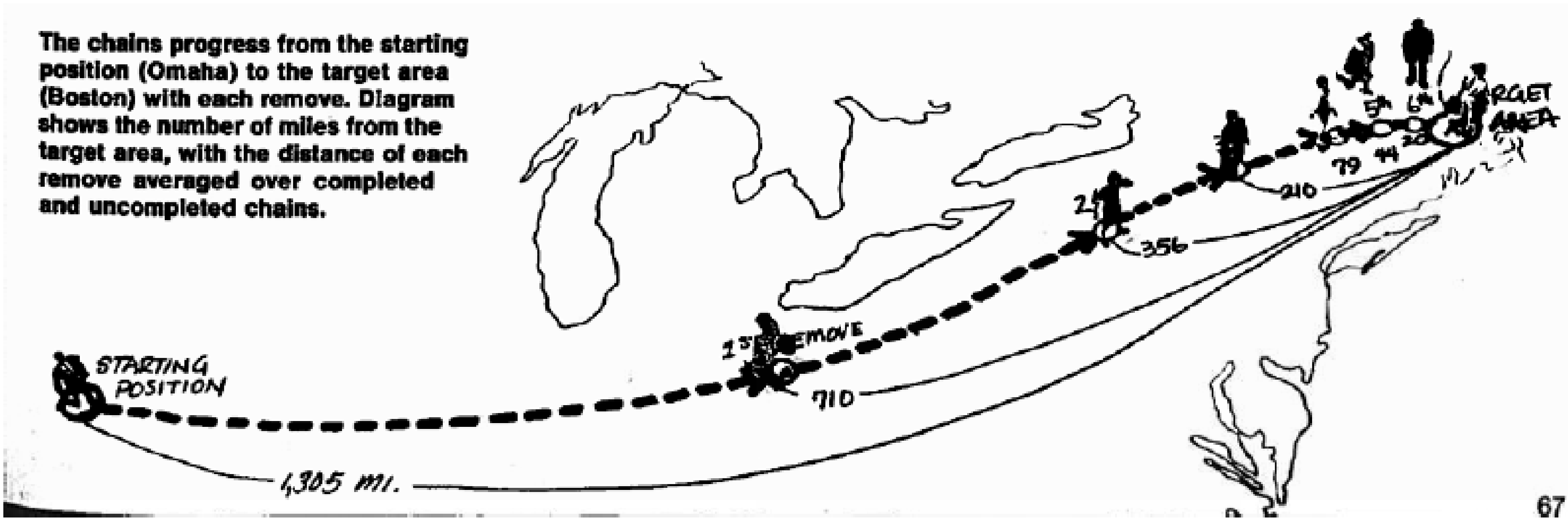
4-regular random graph

Small-World: Summary

- **Could a network with high clustering be at the same time a small world?**
  - Yes. You don’t need more than a few random links.
- **The Watts Strogatz Model:**
  - Provides insight on the interplay between clustering and the small-world
  - Captures the structure of many realistic networks
  - Accounts for the high clustering of real networks
  - Does not lead to the correct degree distribution
  - Does not enable **navigation** (next lecture)

How to Navigate the Network?

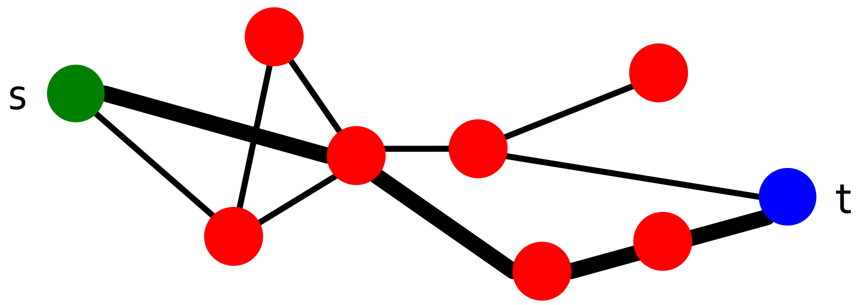
- (1) What is the structure of a social network?
- **(2) What strategies do people use to route and find the target?**



How would you go about finding the path?

Decentralized Search

- *s* only knows locations of its friends and location of the target *t*
- *s* does not know links of anyone but itself
- **Geographic Navigation:** *s* navigates to the node closest to *t*
- **Search time T:** Number of steps to reach *t*



Overview of the Results

Searchable

Search time:

$$O((\log n)^{\beta})$$

Kleinberg's model

$$O((\log n)^{2})$$

Not searchable

$$O(n^{\alpha})$$

Watts-Strogatz

$$O(n^{\frac{2}{3}})$$

Erdős–Rényi

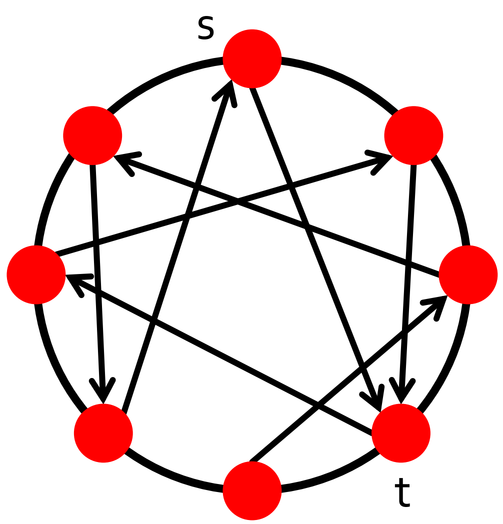
$$O(n)$$

Navigation in Watts-Strogatz

- **Model:** 2-dim grid where each node has one random edge
  - This is a small-world
- **Fact:** A decentralized search algorithm in Watts-Strogatz model needs $N^{2/3}$ steps to reach t in expectation
  - **Note:** even though paths of $O(\log N)$ steps exist

Navigation in Watts-Strogatz

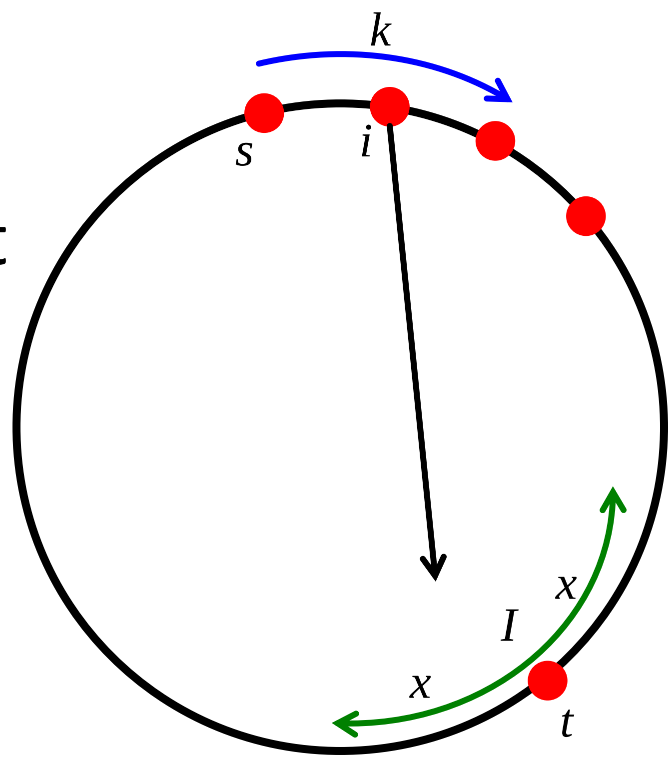
- **Let's do the proof for 1-dimensional case**
- **About the proof:**
  - Setting: n nodes on a ring plus one random directed edge per node.
  - Search time is now $O(n^{1/2})$
    - For d-dim. case: $\sim n^{d/(d+1)}$
  - Proof strategy: **Principle of deferred decision**
    - Doesn't matter when a random decision is made if you haven't seen it yet
    - Assume random long range links are only created once you get to them



Proof: Search time is $\geq n^{1/2}$

- **Claim:**
  - Expected search time is $\geq n^{1/2}$
- **Let:** E_i = event that long link out of node i points to some node in interval I of width $2x$ nodes
- **Then:** $P(E_i) = 2x/n$
  (haven't seen node i yet, but can assume random edge generation)
- **Let:** E =event that any of first k nodes you see has a link to I:
- **Then:**

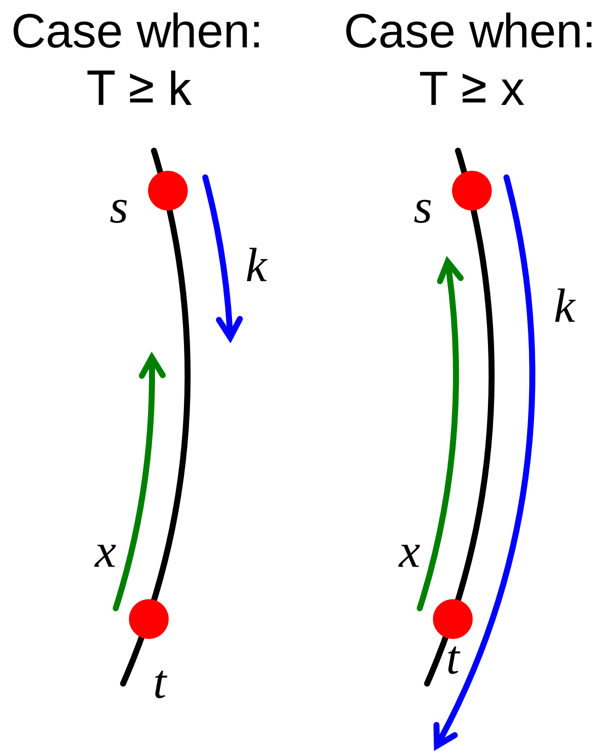
$$P(E) = P\left(\bigcup_{i}^{k} E_i\right) \leq \sum_{i}^{k} P(E_i) = \frac{2kx}{n}$$



Proof: Search time is ≥ $n^{1/2}$

- Prob. of link to I: $P(E) \le \dfrac{2kx}{n}$
- **Need** k, x s.t. $\dfrac{2kx}{n} < 1$
- **Choose:** $k = x = \frac{1}{2}\sqrt{n}$

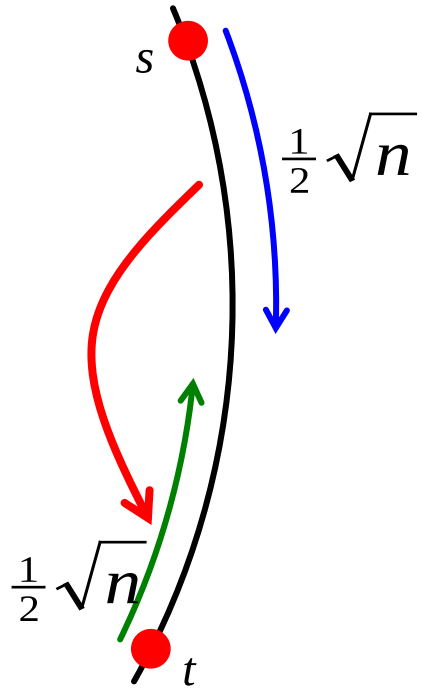
  So, $P(E) = 2\dfrac{\left(\frac{1}{2}\sqrt{n}\right)^2}{n} = \dfrac{1}{2}$
- **Suppose** initial s is outside I and E does not happen. **Then** the search algorithm must take $\geq min(k, x)$ steps to get to t



Proof: Search time is ≥ $n^{1/2}$

- **Claim:** Getting from s to t takes $\geq k = \frac{1}{2}\sqrt{n}$ steps
  - If we don't take a long-range link, we must traverse $\geq \frac{1}{2}\sqrt{n}$ steps to get in t
  - Expected time to get to t:

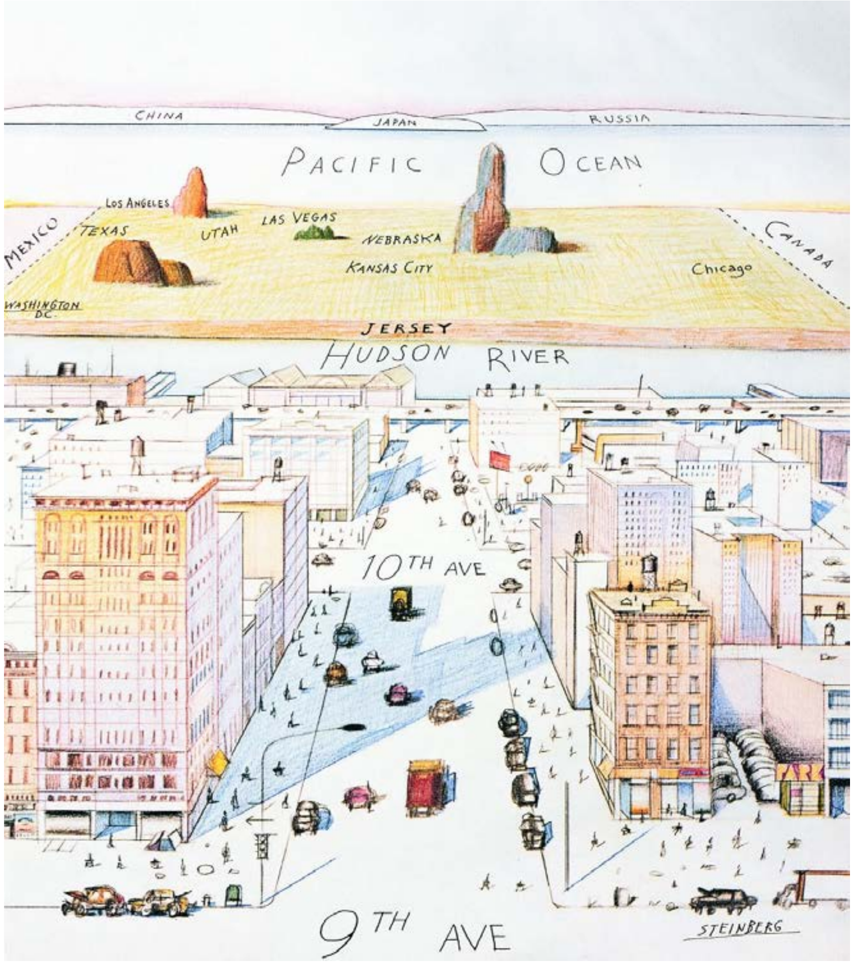
$$\geq \left(\frac{1}{2}k + \frac{1}{2}x\right) P(E\ occurs) + \frac{1}{2}\sqrt{n}\, P(E\ doesn't\ occur) = \frac{1}{2}\sqrt{n}$$



- **Algorithm:**
  - Walk in the direction of t
  - With prob. $\frac{1}{\sqrt{n}}$ we have a link to I
  - It takes $O(\sqrt{n})$ steps on average to find such link
  - After that need another $O(\sqrt{n})$ steps to walk towards t

Navigable Small-World Graph?

- Watts-Strogatz graphs are **not searchable**
- **How do we make a searchable small-world graph?**
- Intuition:
  - Our long range links are not random
  - **They follow geography!**



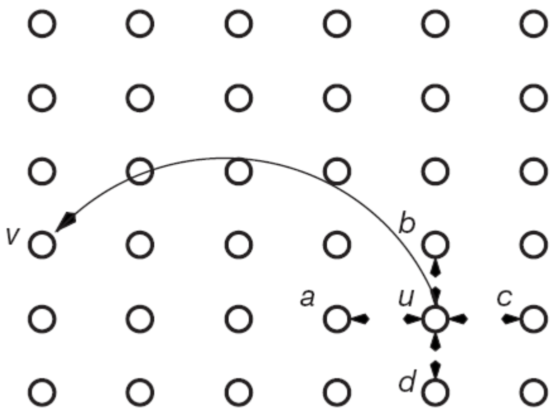
Saul Steinberg, "View of the World from 9th Avenue"

Variation of the Model

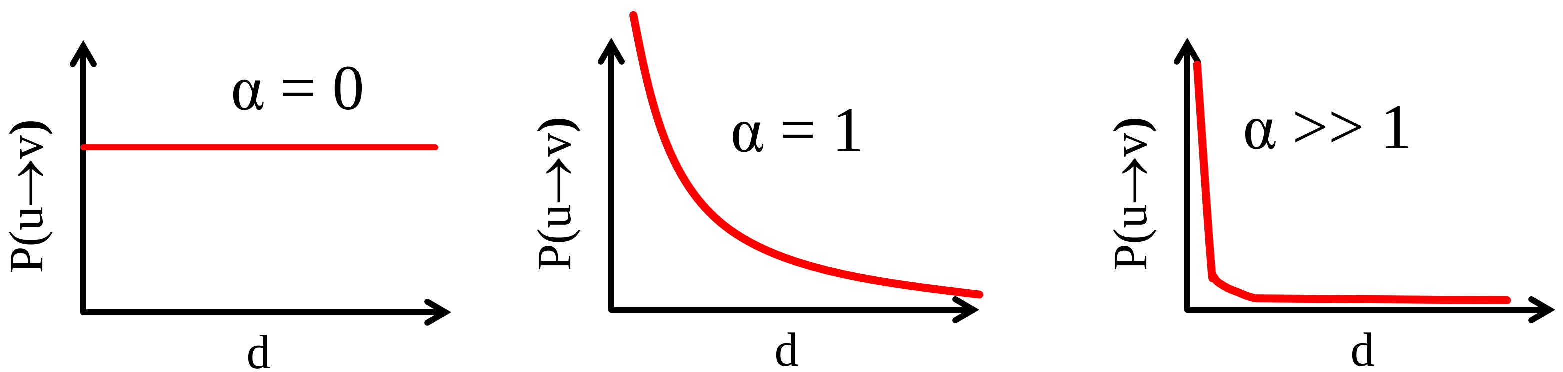
- **Model** [Kleinberg, Nature '01]
  - Nodes still on a grid
  - Node has one long range link
  - Prob. of long link to node v:

$$P(u \to v) \sim d(u,v)^{-\alpha}$$

    - $d(u,v)$ … grid distance between u and v
    - α … parameter ≥ 0



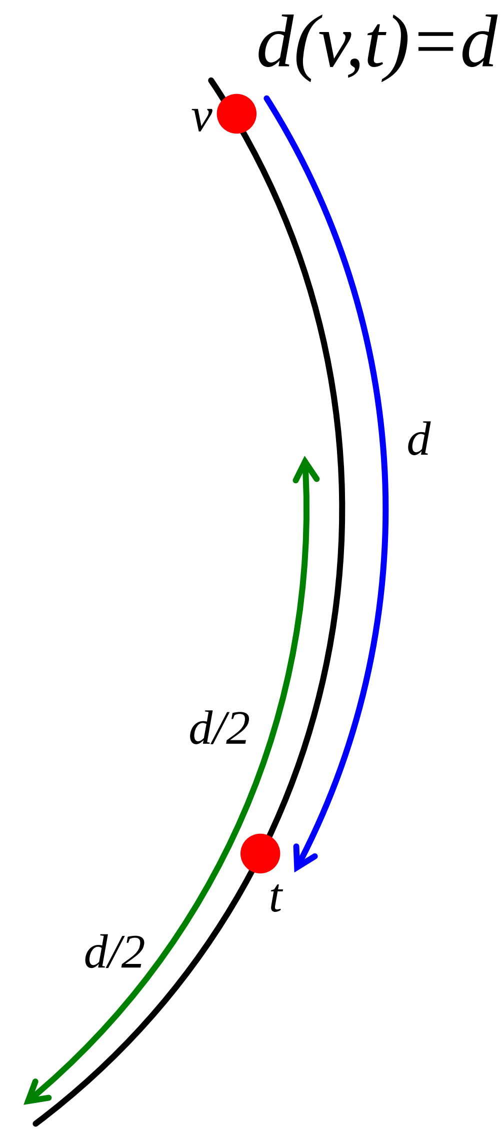
$$P(u \to v) = \frac{d(u,v)^{-\alpha}}{\sum_{w \neq u} d(u,w)^{-\alpha}}$$



Kleinberg's Model in 1 Dimension

1-dim case:

- **Claim:** For $\alpha=1$ we can get from s to t in $O(\log(n)^2)$ steps
  - Set: $I = \frac{d}{2}$
  - Now we want $P\begin{pmatrix}\text{long range link} \\ \text{from } v \text{ points} \\ \text{to a node in } I\end{pmatrix}$



Kleinberg's Model in 1D

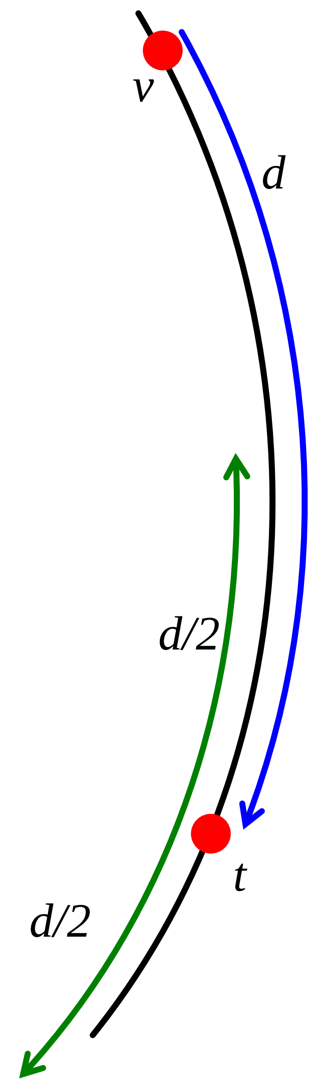
- We need to calculate:

$$P(v \to w) = \frac{d(v,w)^{-1}}{\sum_{u \neq v} d(v,u)^{-1}}$$

- What is the normalizing const?

$$\sum_{u \neq v} d(u,v)^{-1} = \sum_{\substack{\text{all possible} \\ \text{distances d} \\ \text{from } 1 \to n/2}} 2d^{-1} = 2\sum_{d=1}^{n/2} \frac{1}{d} \leq 2 \ln n$$

$$\sum_{d=1}^{n/2} \frac{1}{d} \leq 1 + \int_{1}^{n/2} \frac{dx}{x} = 1 + \ln(\frac{n}{2}) = \ln n$$



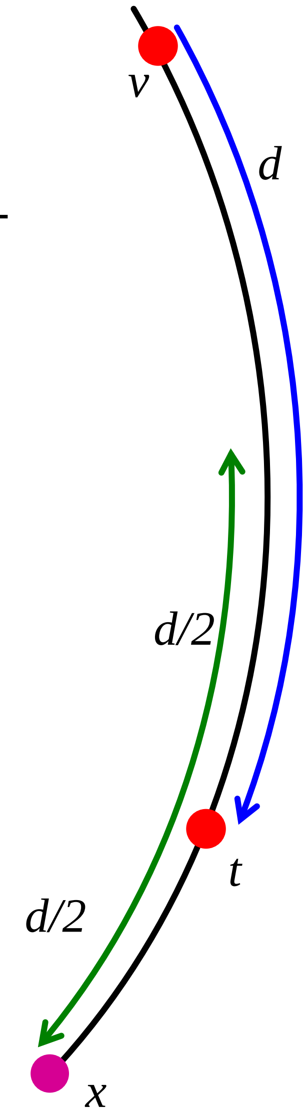
Kleinberg's Model in 1D

- We need ***P(v* points to *I)*=**

$$P(v\,\text{points to}\,I) = \sum_{w \in I} \mathrm{P}(\mathrm{v} \to \mathrm{w}) \geq \sum_{w \in I} \frac{\mathrm{d}(\mathrm{v},\mathrm{w})^{-1}}{2\ln n}$$

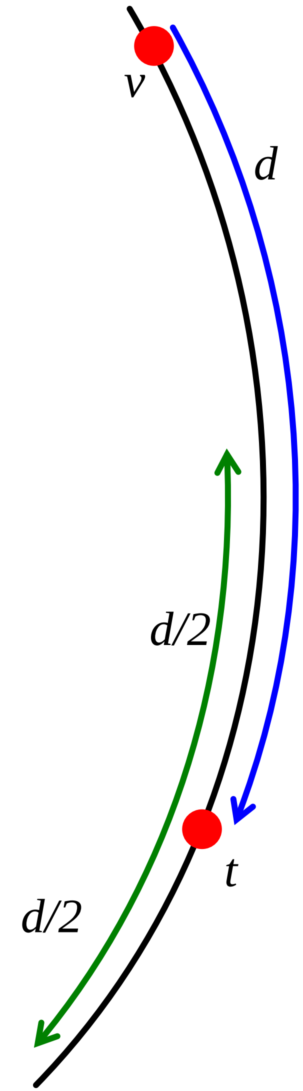
$$= \frac{1}{2\ln n} \sum_{w \in I} \frac{1}{\underbrace{d(v,w)}_{\text{All terms } \geq 2/(3d)}} \geq \frac{1}{2\ln n} d \frac{2}{3d} = \frac{1}{3\ln n}$$

Note:
d(v,x)=3d/2



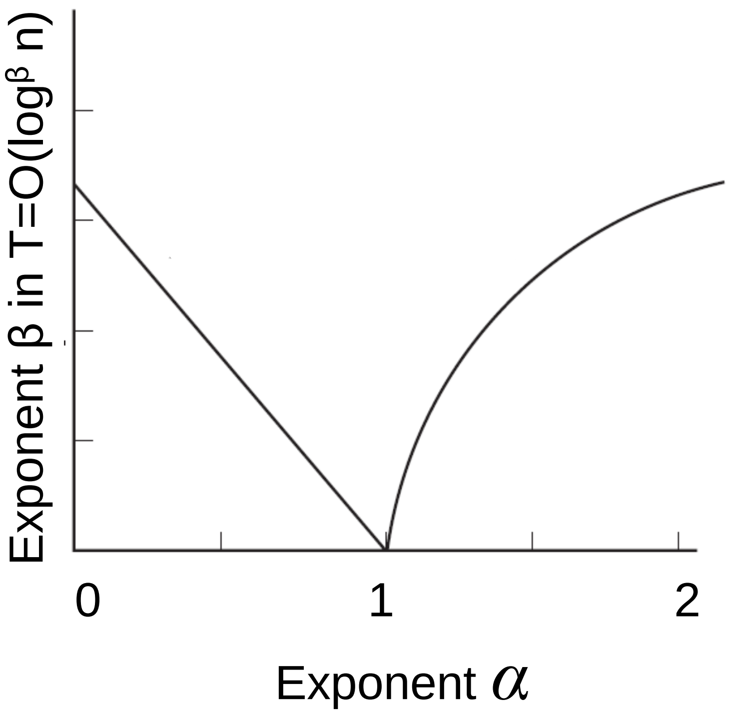
Kleinberg's Model in 1D

- We have:
  - I ... interval of $d/2$ around t (where d=d(s,t))
  - P(long link of v points to I)=1/ln(n)
- In expected # of steps $\leq \ln(n)$ you get into I, and you thus halve the distance to t
- Distance can be halved at most $\log_2(n)$ times, so expected time to reach t: $O(\ln(n)\cdot\log_2(n)) = \mathbf{O(\log(n)^2)}$

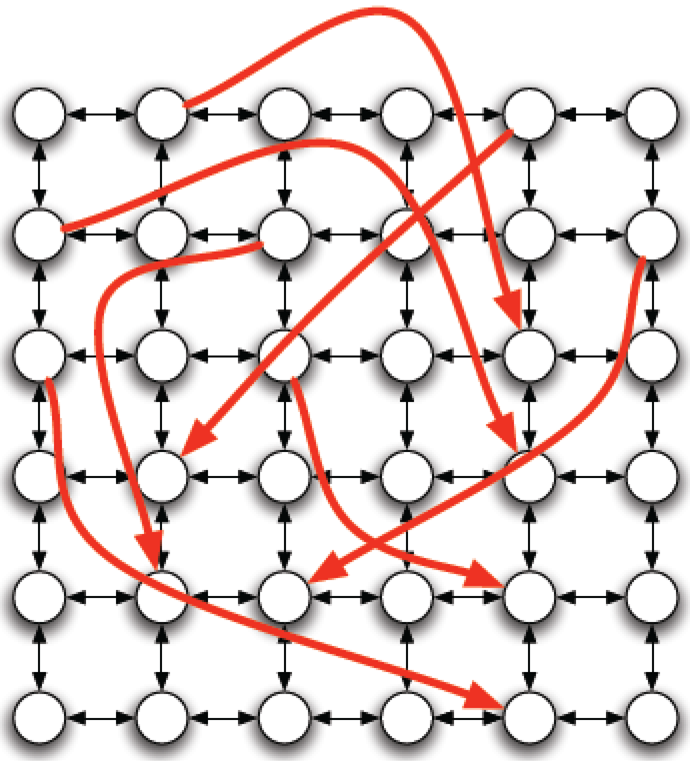


Kleinberg's Model: Search Time

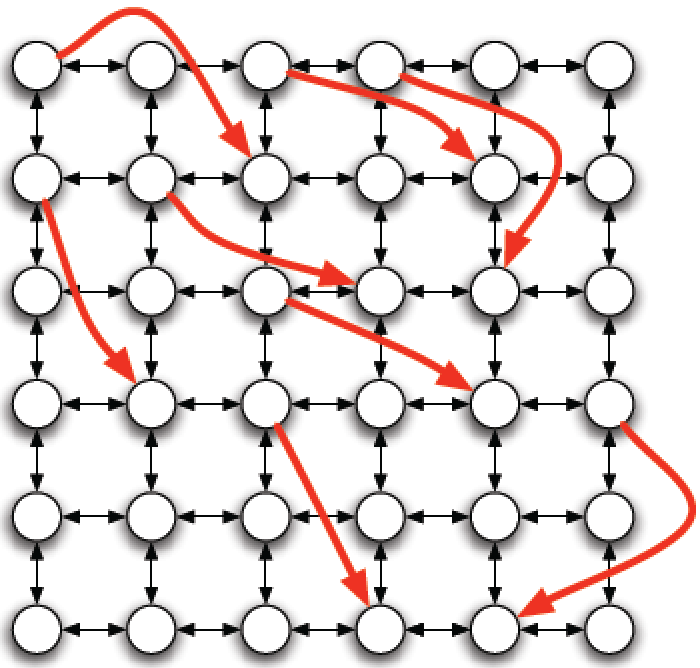
- We know:
  - $\alpha=0$ (i.e., Watts-Strogatz): we need $\sqrt{n}$ steps
  - $\alpha=1$: we need $T=O(log(n)^2)$ steps



Intuition: Why Search Takes Long



Small α: too many long links



Big α: too many short links

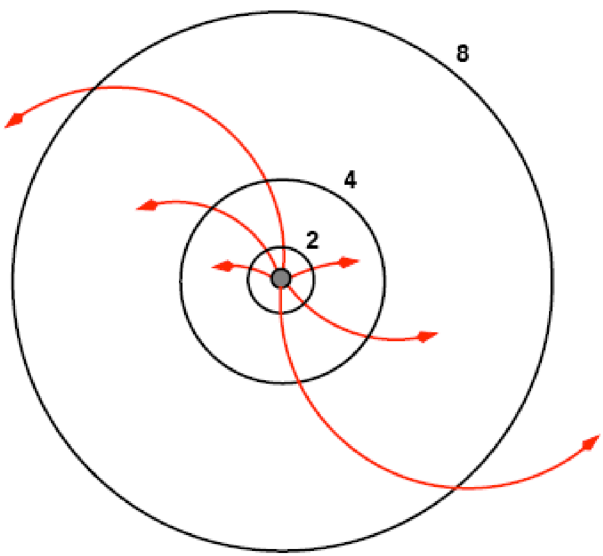
Demo: http://projects.si.umich.edu/netlearn/NetLogo4/SmallWorldSearch.html

Why Does It Work?

- **How does the argument change for 2-d grid:**
  - $P(u \to v) > 1/Z \quad \cdot \quad size(I) \quad \cdot \quad Prob\ on\ each\ node$

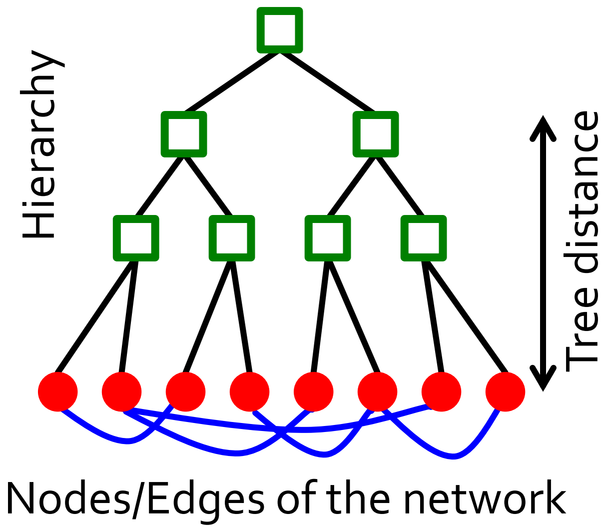
$$\log n \qquad d^2 \qquad d^{-2} \qquad \Rightarrow \alpha = 2$$

- **Why $P(u \to v) \sim d(u,v)^{-dim}$ works?**
  - Approx uniform over all "scales of resolution"
  - # points at distance d grows as d^{dim}, prob. d^{-dim} of each edge → const. prob. of a link, independent of d



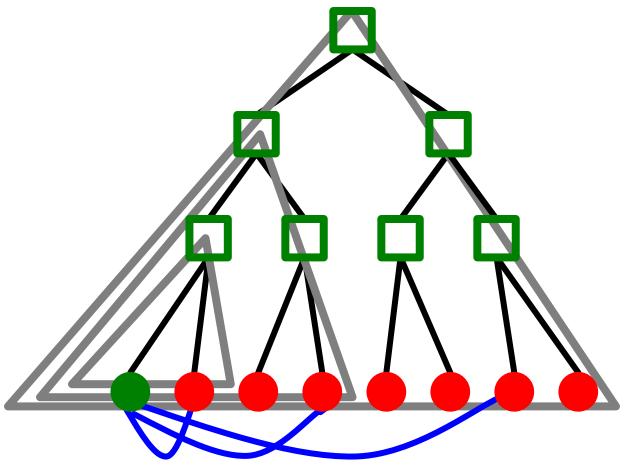
Different Model: Hierarchies

- $h(u,v)$ = tree-distance (height of the least common ancestor)
- $P(u \to v) \sim b^{-\alpha\, h(u,v)}$
- $P(u \to v)$ is approx uniform at all scales of resolution
- **How many nodes are at dist. h?** $(b-1)b^{h-1} \sim b^h$
  - So we need b^{-h} to cancel, as we wanted for distance independence
- Start at s, want to go to t
  - Only see out links of node you are at
  - Have knowledge of where t is in the tree



Different Model: Hierarchies

- **Nodes are in the leaves of a tree:**
  - Departments, topics, …
- Create k edges out of a node
  - Create i-th ($i=1…k$) edge out of v by choosing $v \rightarrow w$ with prob. $\sim b^{-h(v,w)}$
- Claim 1:
  - For any direct subtree T' one of v's links points to T'
- Claim 2:
  - Claim 1 guarantees efficient search
- **You will prove C1 & C2 in HW1**

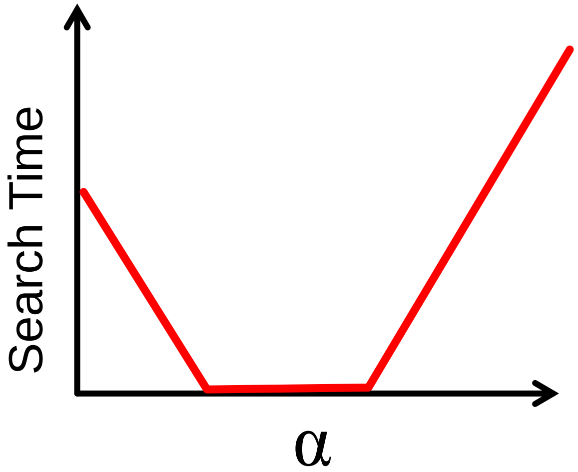


Node has 1 link to each direct subtree

[Watts-Dodds-Newman '02]

Different Model: Hierarchies

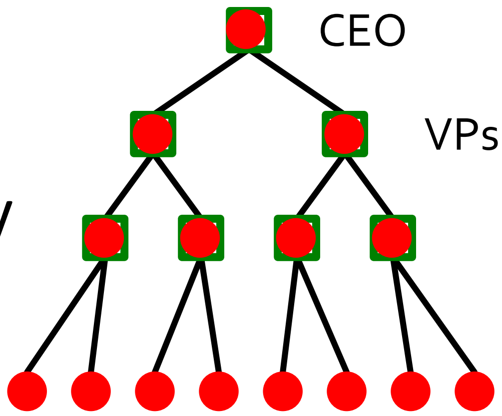
- **Extension:**
  - Multiple hierarchies – geography, profession, …
  - Generate separate random graph in each hierarchy
  - Superimpose the graphs
  - Search algorithm:
    - Choose a link that gets closest in any hierarchy
- **Q: How to analyze the model?**
  - **Simulations:**
    - Search works for a range of alphas
    - Biggest range of searchable alphas for 2 or 3 hierarchies
      - Too many hierarchies hurts



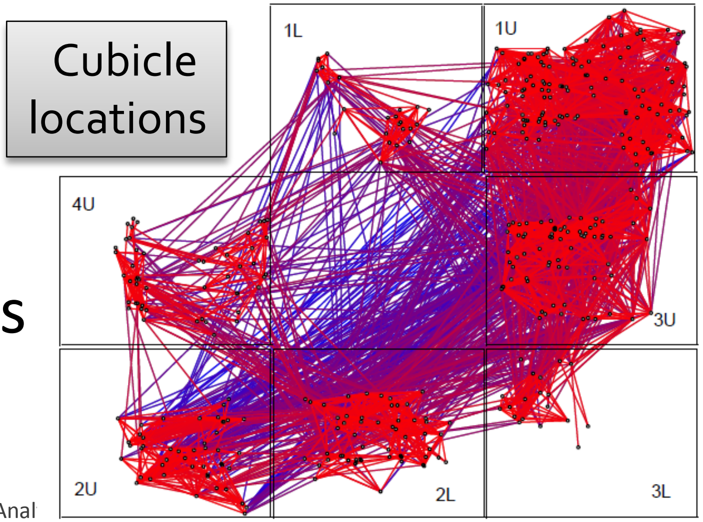
Empirical Studies of Navigation in Small-World Networks

Small-World in HP Labs

- **Adamic-Adar 2005:**
  - HP Labs email logs (436 people)
  - Link if u,v exchanged >5 emails each way
  - Map of the organization hierarchy
    - How many edges cross groups?
    - Finding:
      $P(u \rightarrow v) \sim 1 / (\text{social distance})^{3/4}$



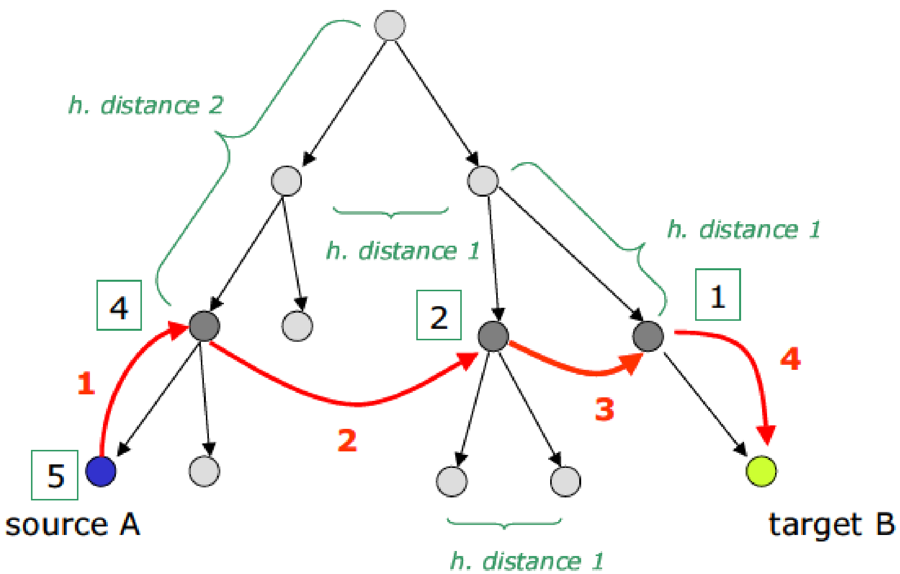
- Differences from the hierarchical model:
  - Data has weighted edges
  - Data has people on non-leaf nodes
  - Data not *b*-ary or uniform depth

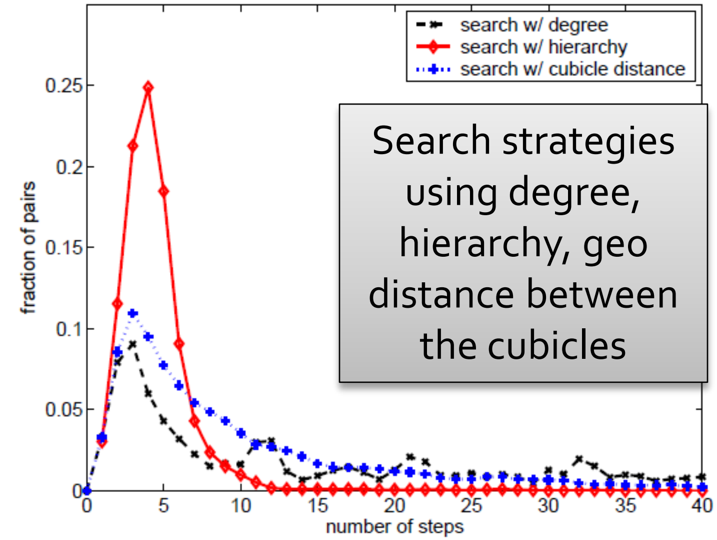


[Adamic-Adar 2005]

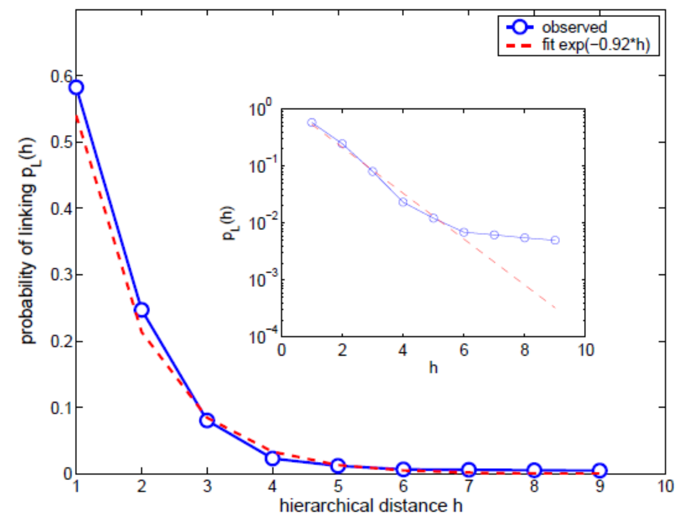
Small-World in HP Labs

- **Generalized hierar. model:**
  - Arbitrary tree defines "groups" = rooted subtrees
  - $P(u \rightarrow v) \sim 1\ /$ (smallest group containing u,v)





Search strategies using degree, hierarchy, geo distance between the cubicles

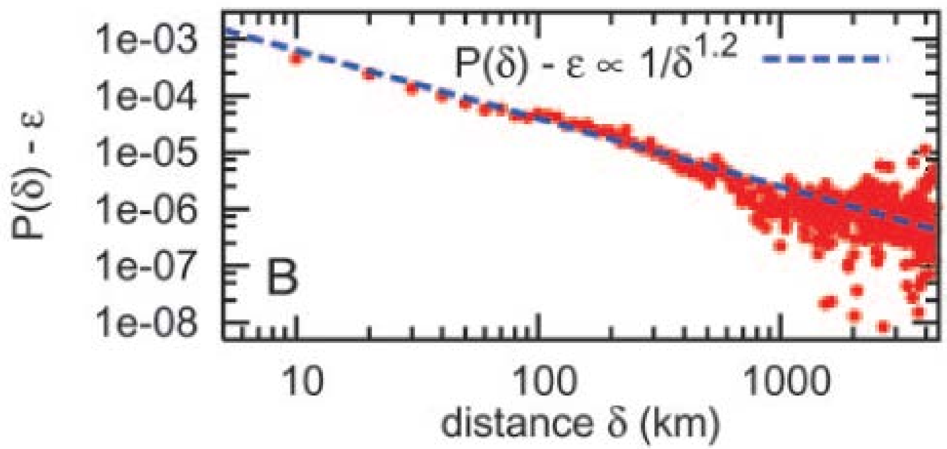


Prob. of link vs. distance in the hierarchy

Small-World in LiveJournal

Liben-Nowell et al. '05:

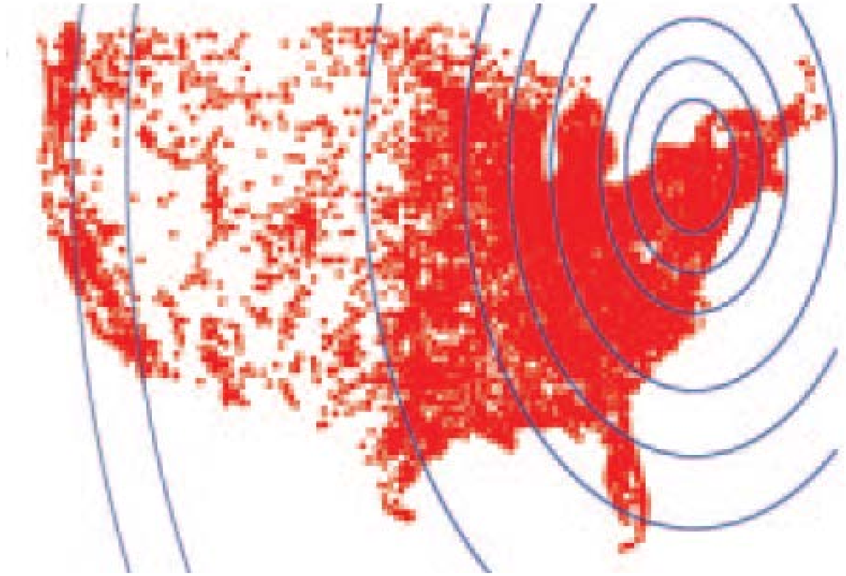
- LiveJournal data
  - Blogers + zip codes

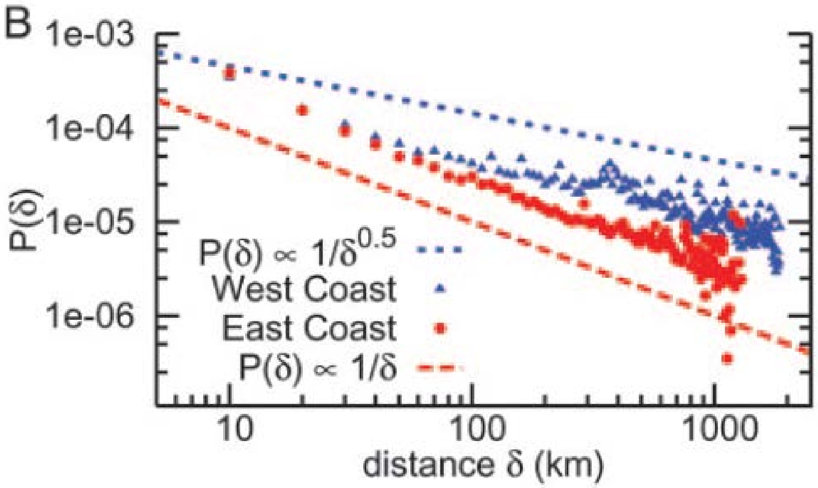


Link length in a network of bloggers (0.5 million bloggers, 4 million links)

- Link prob.: $P(u,v)=\delta^{\alpha}$
- **α =?**
- **Problem:**
  - Not uniform population density
- **Solution: Rank based friendship**

Improved model





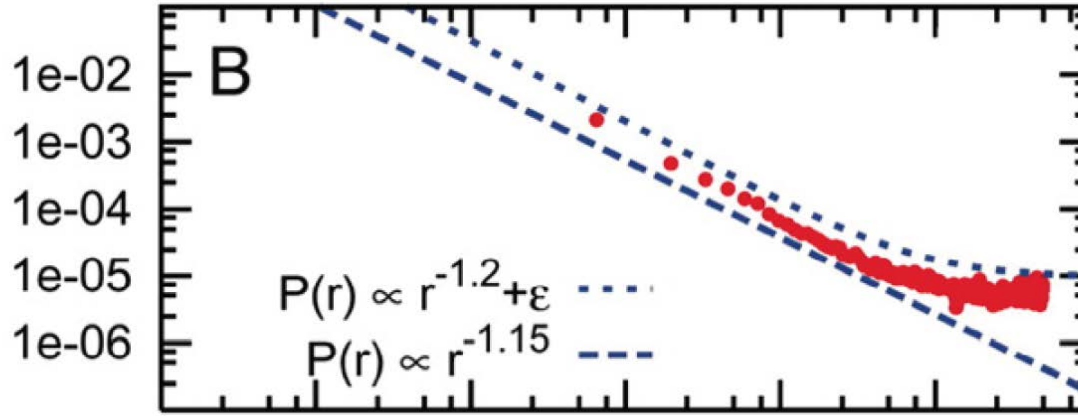
$$rank_u(v) := \left|\{w : d(u,w) < d(u,v)\}\right|$$

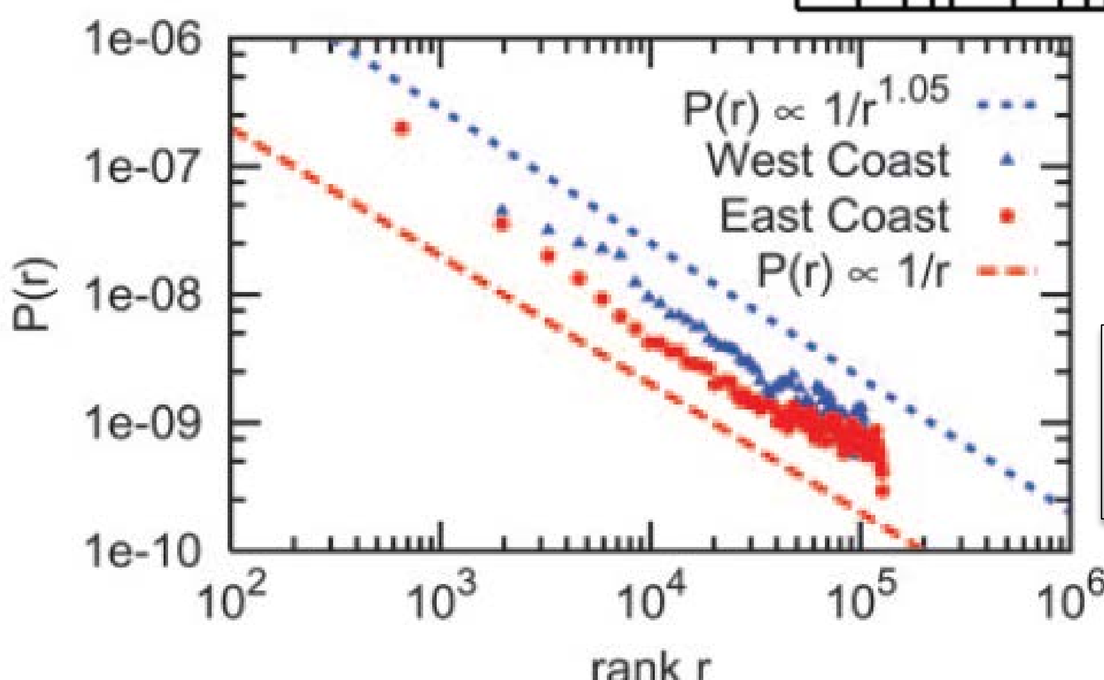
- $P(u \rightarrow v) = rank_u(v)^{-\alpha}$
- What is best α?
  - For equally spaced pairs: α=dim. of the space
  - In this special case α=1 is best for search

[Liben-Nowell et al. '05]

Rank based friendships

- Close to theoretical optimum of -1

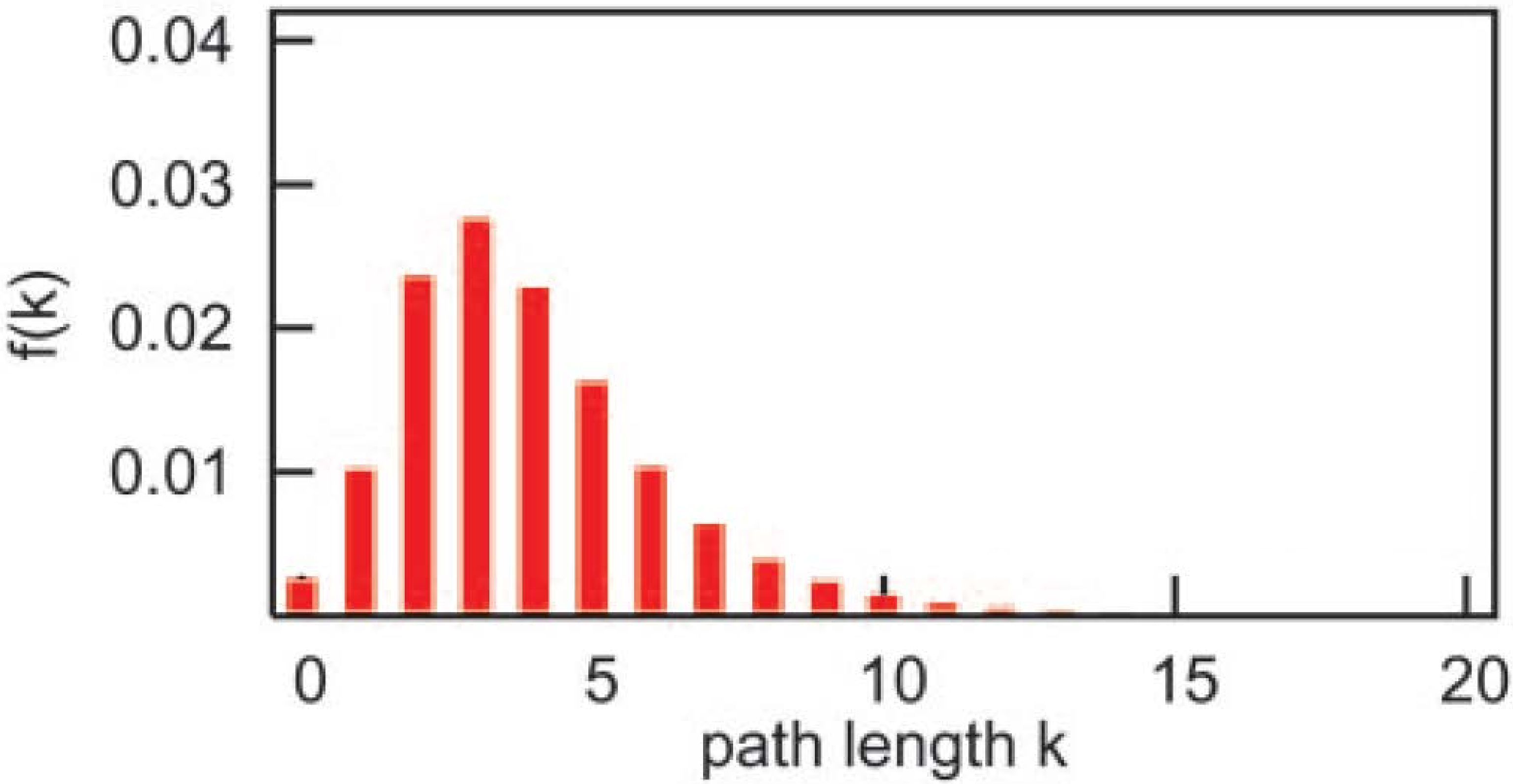




The difference between the East and West coast disappears

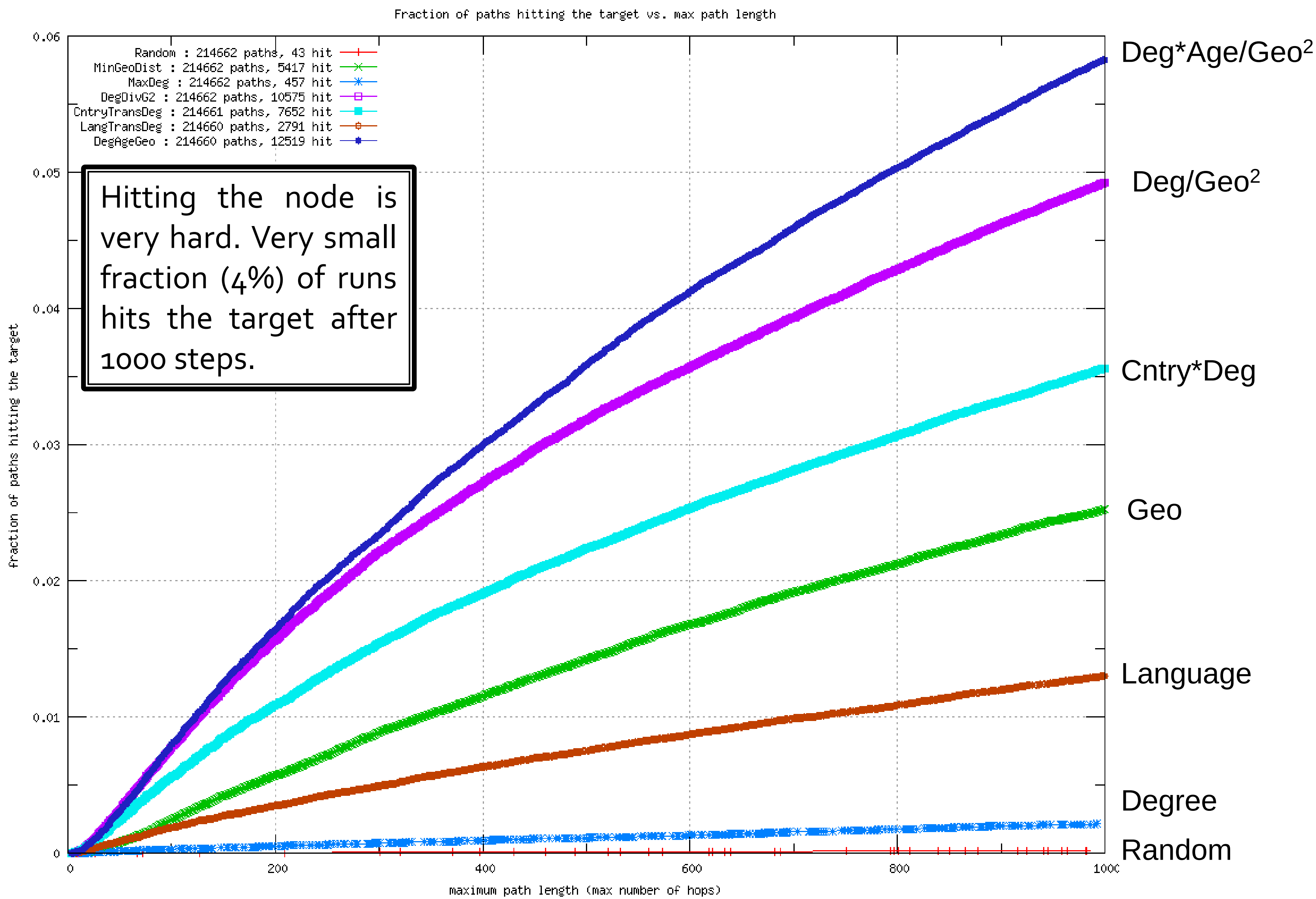
[Liben-Nowell et al. '05]

Geographic Navigation

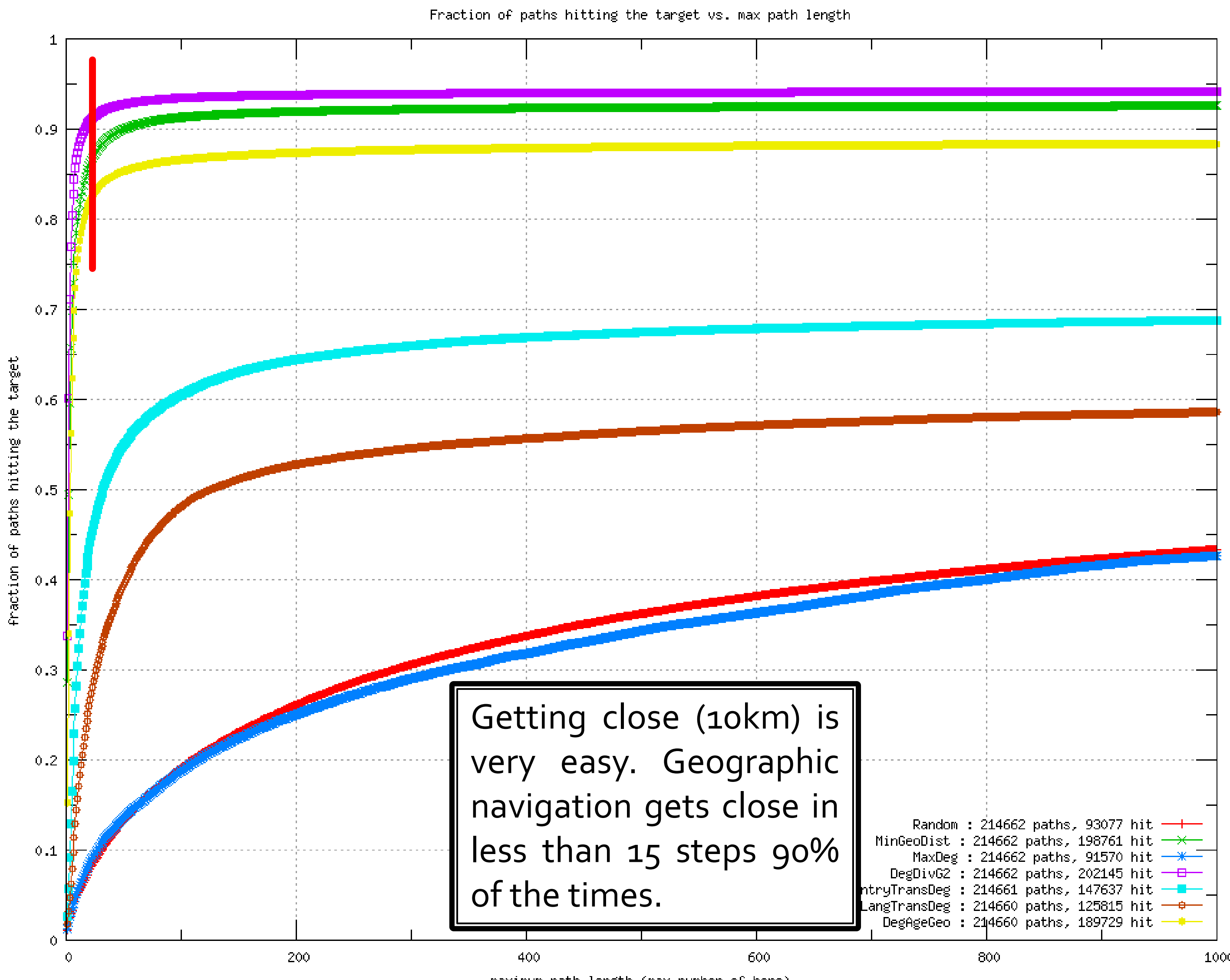


- **Decentralized search in a LiveJournal network**
  - 12% searches finish, average 4.12 hops

Messenger: Prob. of Hitting Target

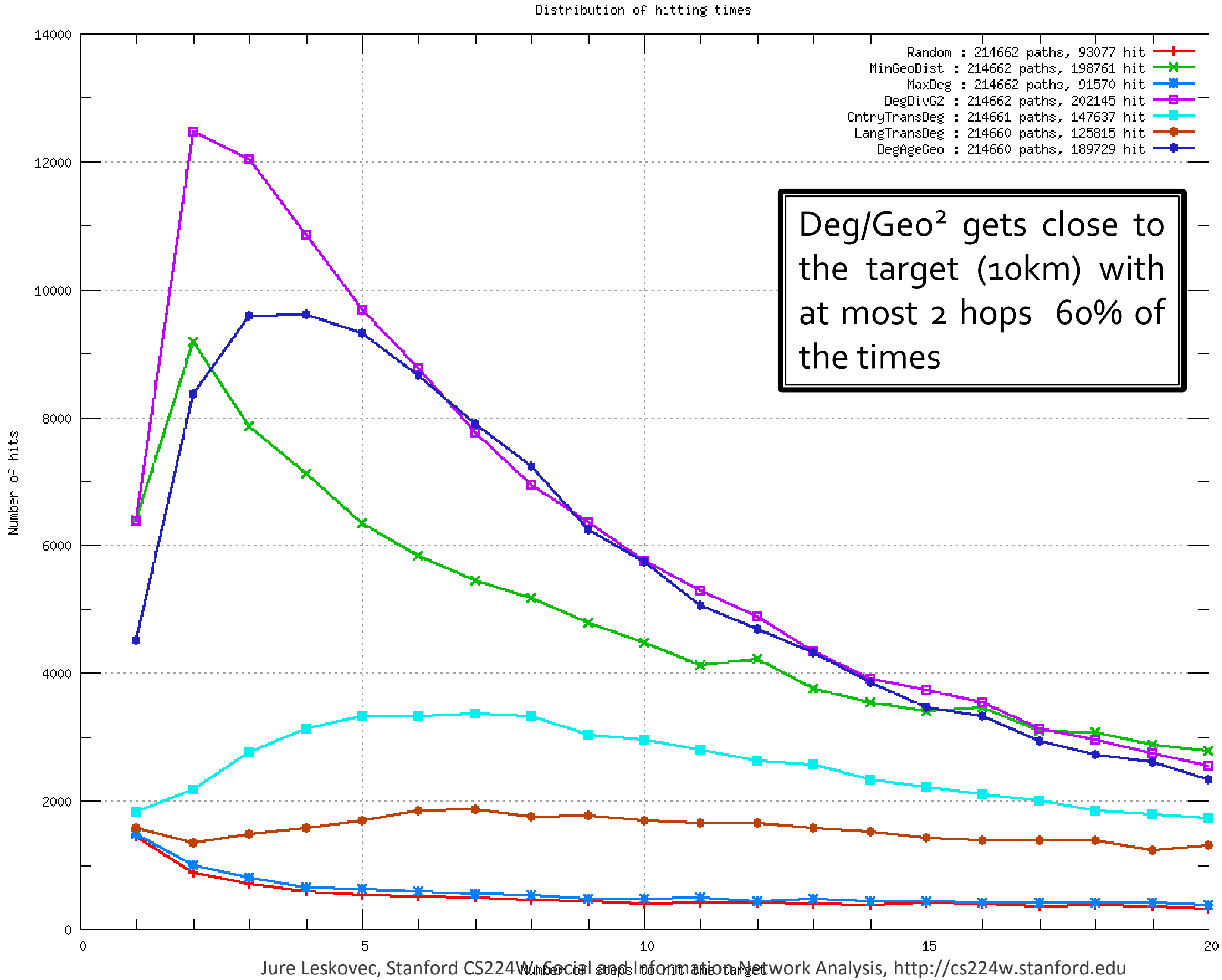


Prob. of Getting Close to Target *t*



Getting close (10km) is very easy. Geographic navigation gets close in less than 15 steps 90% of the times.

Distribution of Getting-Close Times



Deg/Geo² gets close to the target (10km) with at most 2 hops 60% of the times

Q: Why do searchable networks arise?

- **Why is rank exponent close to -1?**
  - Why in any network? Why online?
  - How robust/reproducible?
- Mechanisms that get α=1 purely through local "rearrangements" of links
- **Conjecture [Sandbeng-Clark 2007]:**
  - Nodes on a ring with random edges
  - Process of morphing links:
    - Update step: Randomly choose s, t, run decentr. search alg.
    - Path compression: each node on path updates long range link to go directly to t with some small prob.
  - Conjecture from simulation: $P(u \rightarrow v) \sim dist^{-1}$

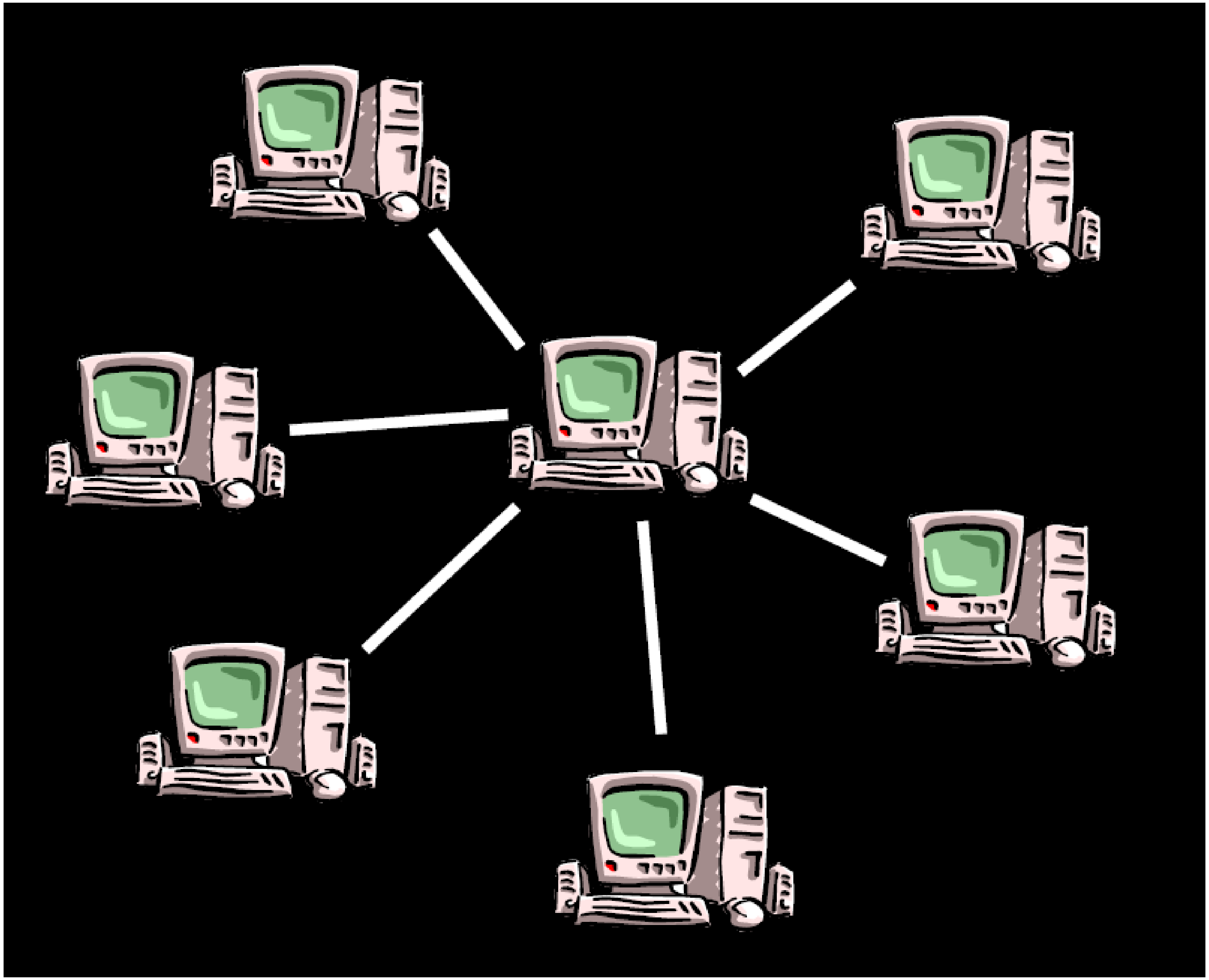
EXTRA MATERIAL: Search in P2P Networks

Extra Material:

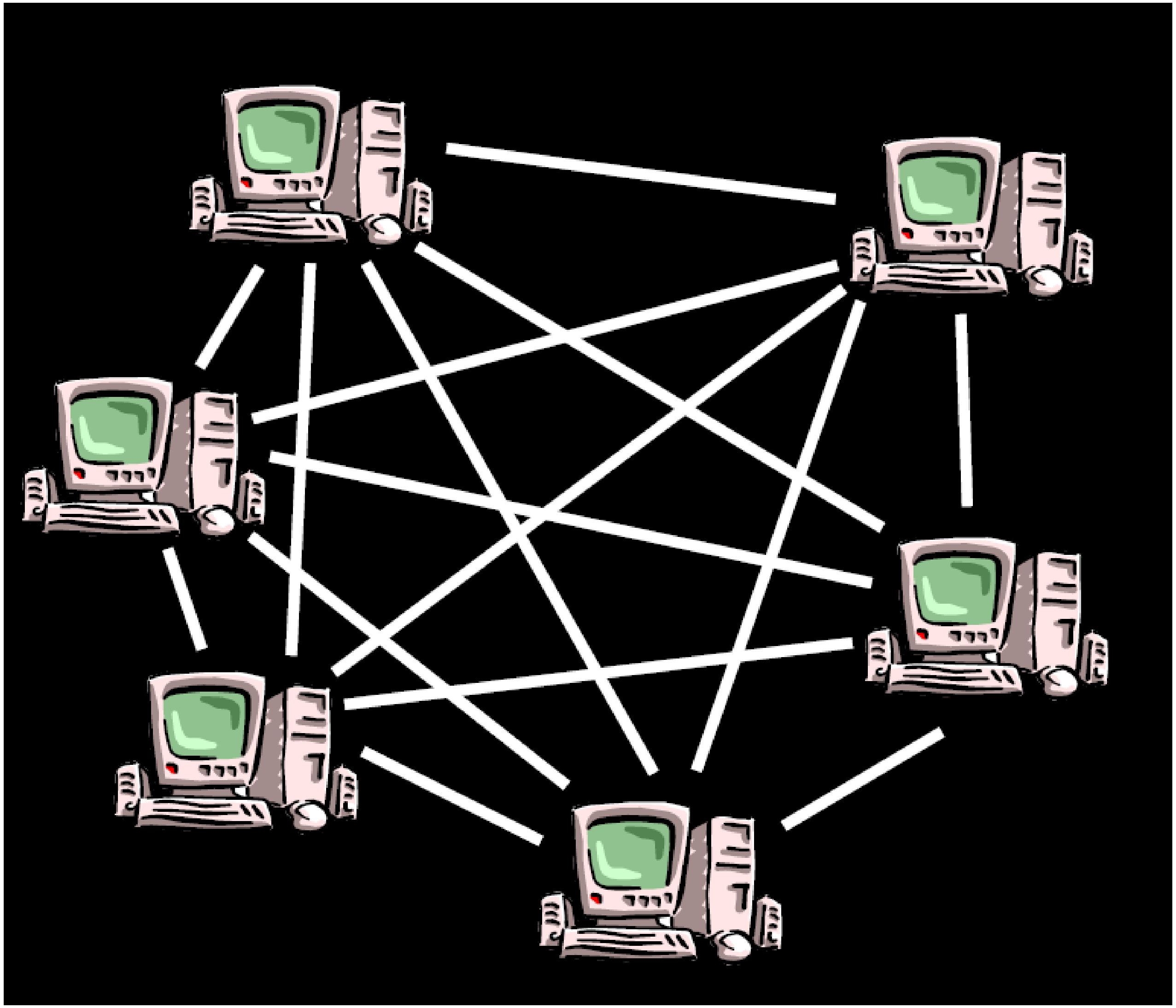
Algorithmic consequence of small-world:

How to find files in Peer-to-Peer networks?

Client – Server



P2P: Only Clients



Napster

- Napster existed from June ‘99 and July ‘01
- Hybrid between P2P and a centralized network
- Once lawyers got the central server to shut down the network fell apart

True P2P networks

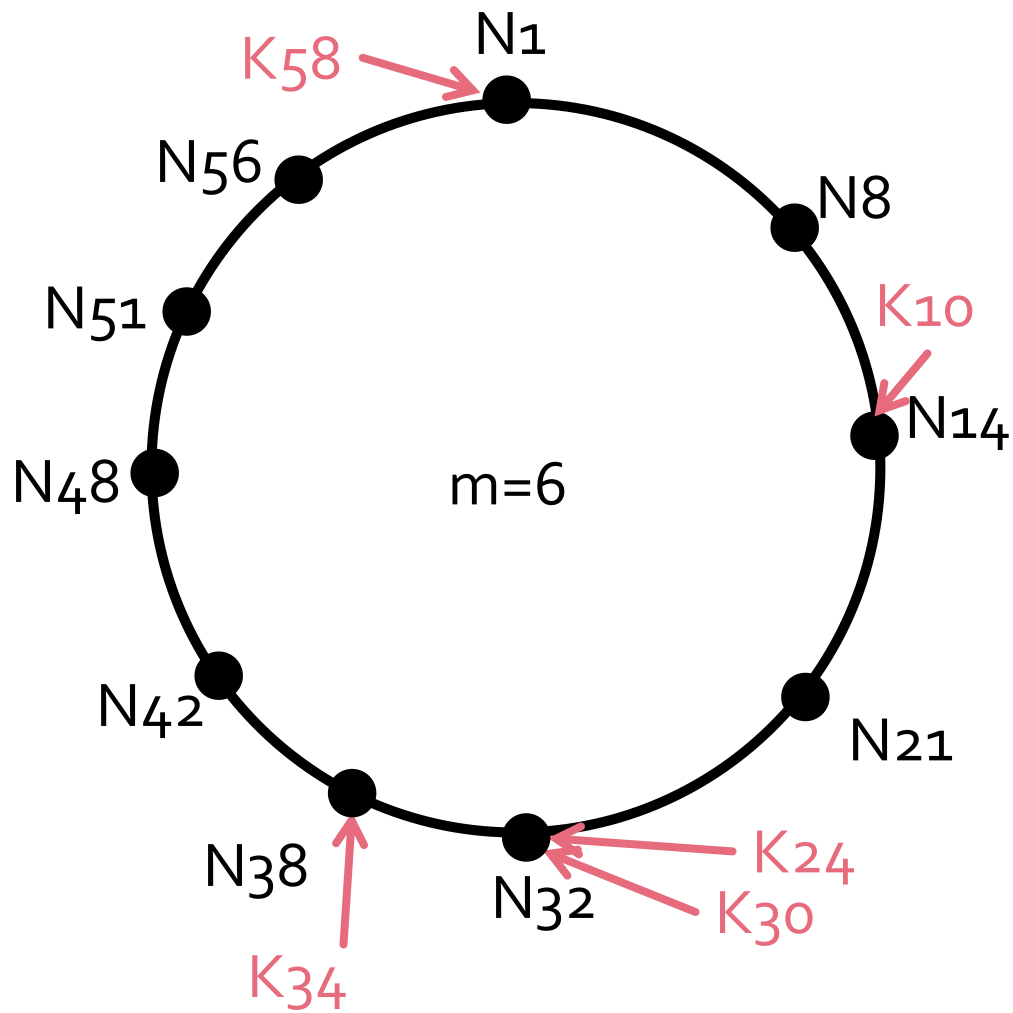
- Networks that can't be turned "off"
  - BitTorrent, ML-donkey, Kazaa, Gnutella
- **Q: How to find a file in a network without a central server?**
- **First attempt: Freenet**
  - Random graph of peers who know each other
  - **Query:** Find a file with key x, $x \in [0,2^{64}]$
  - **Algorithm:**
    - If node has it, done
    - Forward query to node with a file having key y as close to x as possible: $\min_y |x-y|$
    - If can't forward, then backtrack.
    - Cut off after some # of steps.
    - Copy the key x along the path (path compression)

Protocol Chord

- Protocol **Chord** consistently maps key (filename) to a node:
  - Keys are files we are searching for
  - Computer that keeps the key can then point to the true location of the file
- Keys and nodes have m-bit IDs assigned to them:
  - Node ID is a hash-code of the IP address
  - Key ID is a hash-code of the file

Chord on a Cycle

- Cycle with node ids 0 to 2^{m-1}
- File (key) k is assigned to a node $a(k)$ with ID $\geq k$

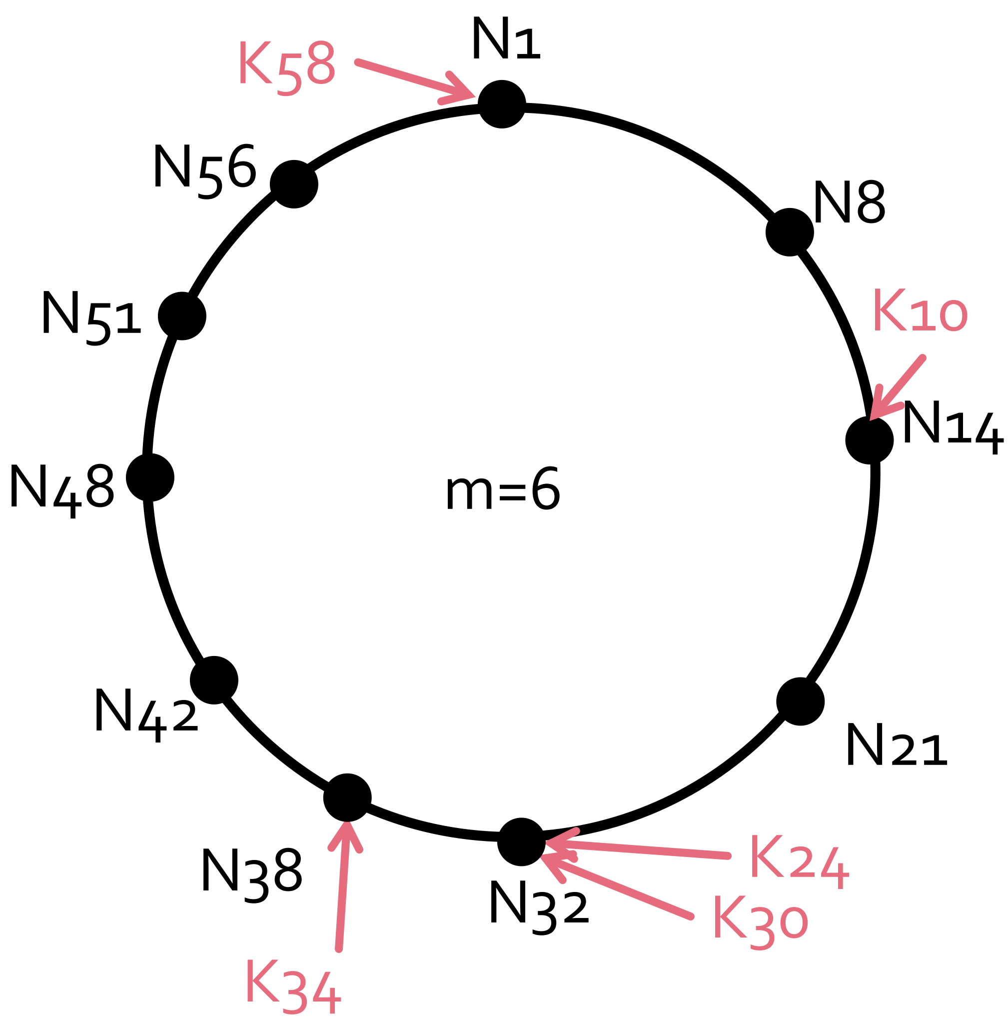


Basics

- Assume we have N nodes and K keys (files)
  How many keys has each node?
- When a node joins/leaves the system it only needs to talk to its immediate neighbors
  - When $N+1$ nodes join or leave, then only O(K/N) keys need to be rearranged
- Each node know the IP address of its immediate neighbor

Searching the network

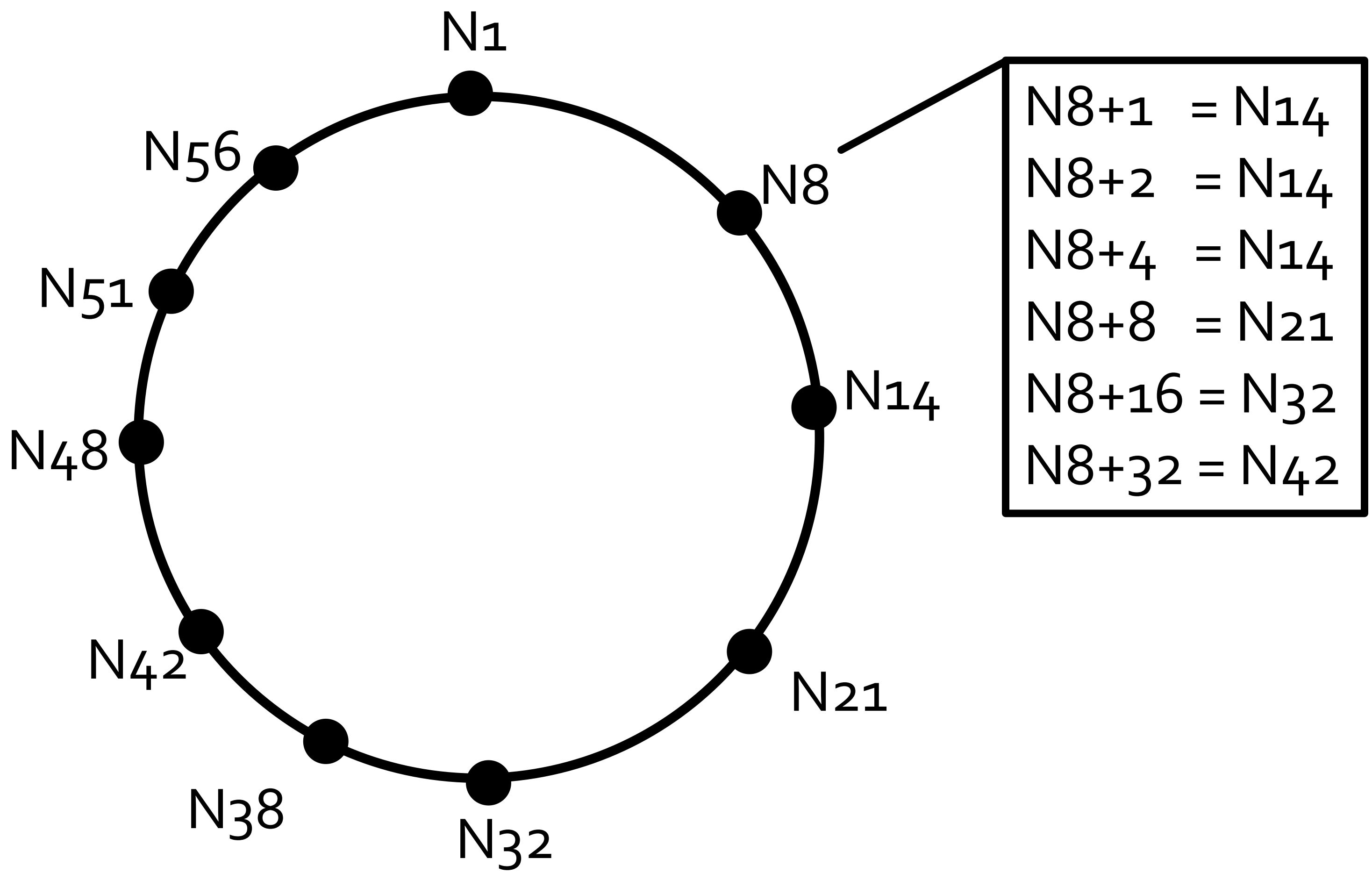
- If every node knows its immediate neighbor then use sequential search



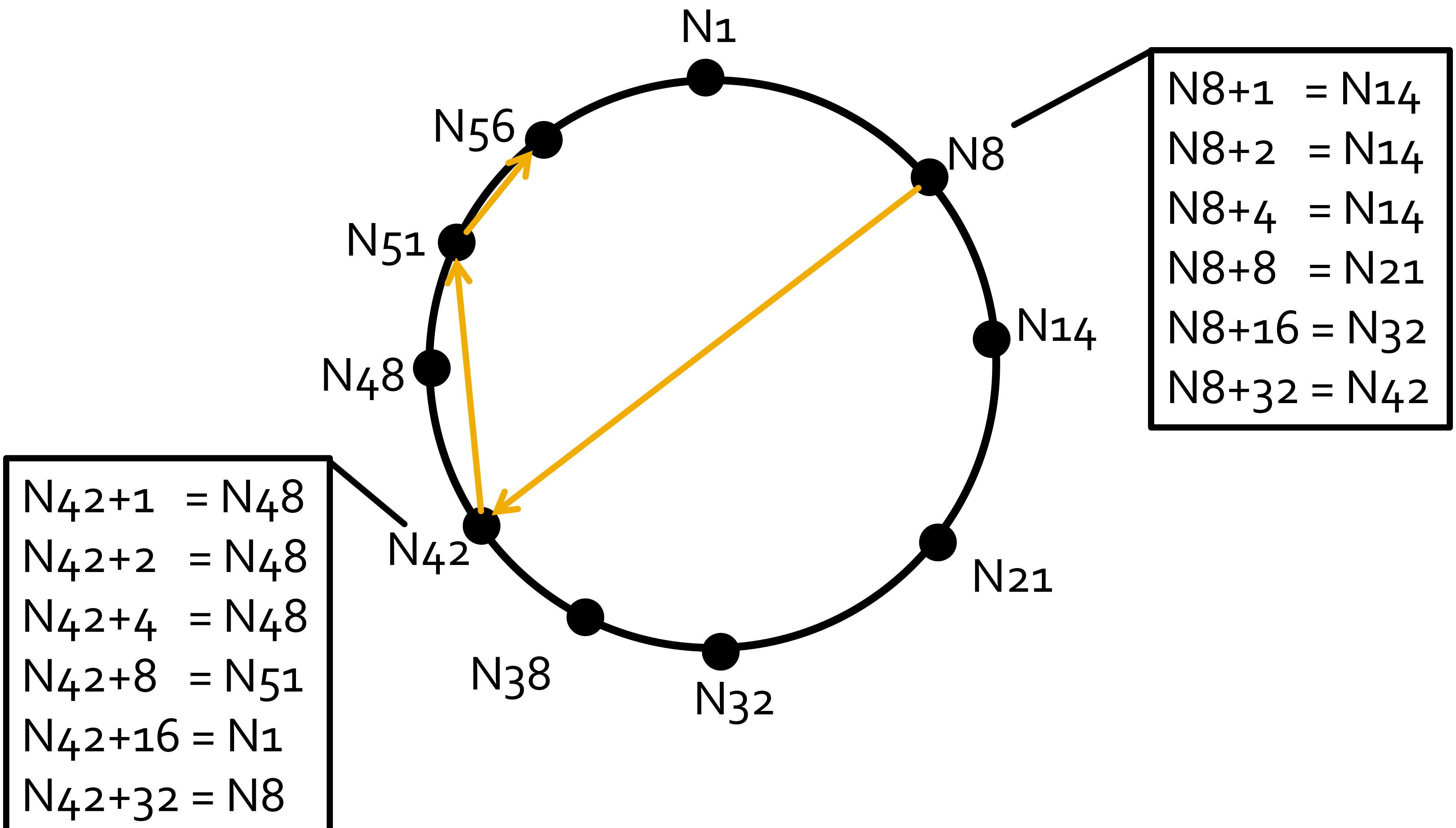
Faster search

- A node maintains a table of $m=log(N)$ entries
- i-th entry of a node n contains the address of $(n+2^i)$-th neighbor
  - **Problem:** When a node joins we violate long range pointers of all other nodes
    - Many papers about how to make this work
- **Search algorithm:**
  - Take the longest link that does not overshoot
    - This way with each step we half the distance to the target

i-th entry of N has the address of $(N+2^i)$-th node



Find key with ID 54



How long does it take to find a key?

- Search for a key in the network of N nodes visits O(log N) nodes

- Assume that node n queries for key k
- Let the key k reside at node t

- How many steps do we need to reach t?

Proof

- We start the search at node n
- Let i be a number such that t is contained in interval $[n+2^{i-1}, n+2^{i}]$
- Then the table at node n contains a pointer to node $n+2^{i-1}$ – the smallest node f from the interval
- Claim: f is closer to t than n
- **So, in one step we halved the distance to *t***
- We can do this at most $log\ N$ times
- <u>Thus, we find t in $O(log\ N)$ steps</u>