Small-World Phenomena and Decentralized Search
Could a network with high clustering be at the same time a small world?

<table>
<thead>
<tr>
<th>P=0</th>
<th>INCREASING RANDOMNESS</th>
<th>P=1</th>
</tr>
</thead>
<tbody>
<tr>
<td>High clustering</td>
<td>High clustering</td>
<td>Low clustering</td>
</tr>
<tr>
<td>High diameter</td>
<td>Low diameter</td>
<td>Low diameter</td>
</tr>
</tbody>
</table>

For a regular network:

\[ h = \frac{N}{2k} \quad C = \frac{3}{4} \]

For a small world network:

\[ h = \frac{\log N}{\log \alpha} \quad C = \frac{k}{N} \]
The Small-World Model

Parameter region of high clustering and low path length

Clustering coefficient, $C = 1/n \sum C_i$

mean vertex-vertex distance

clustering coefficient

Intuition: It takes a lot of randomness to ruin the clustering, but a very small amount to create shortcuts.
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.

There are already 12 triangles in the grid and the long-range edge can only close more.

What's the diameter?

It is $O(\log(n))$

Why?

$$C_i = \frac{2 \cdot e_i}{k_i(k_i - 1)} \geq \frac{2 \cdot 12}{9 \cdot 8} \geq 0.33$$
Proof:

- Consider a graph where we contract 2x2 subgraphs into supernodes.
- Now we have 4 random edges sticking out of each supernode.
  - 4-regular random graph!
- From Thm. we have short paths between super nodes (due to 4 random edges).
- We can turn this into a path in a real graph by adding at most 2 steps per long range edge (by having to traverse internal nodes).

⇒ Diameter of the model is $O(2 \log n)$
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)
How to Navigate the Network?

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

How would you go about finding the path?
The setting:

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

Searchable
Search time $T$:

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

Kleinberg’s model

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

Not searchable
Search time $T$:

$O(n^\alpha)$

Watts-Strogatz

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

Erdős–Rényi

$O(n)$

**Note:** We know these graphs have diameter $O(\log n)$. So in Kleinberg’s model search time is polynomial in $\log n$, while in Watts-Strogatz it is exponential (in $\log n$).
Model: 2-dim grid where each node has 1 random edge
- This is a small-world!
  - (Small-world = diameter $O(\log n)$)

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

Note: All our calculations are asymptotic, i.e., we are interested in what happens as $n \to \infty$
Let’s do the proof for 1-dimensional case
Want to show Watts-Strogatz is NOT searchable
  - Bound the search time from below
About the proof:
  - **Setting:** $n$ nodes on a ring plus one random directed edge per node.
  - Search time is $T \geq O(\sqrt{n})$
    - For $d$-dim. lattice: $T \geq O(n^{d/(d+1)})$
  - **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 link is only created once you get to the node
Overview of the proof:

- **Reason about event \( E \)**
  
  \( E = \) event that any of the first \( k \) nodes search algorithm visits has a link to \( I \) of width

  \( 2 \cdot x \) nodes (for some \( x \)) around target \( t \)

- **If** \( k = x = \frac{1}{2} \sqrt{n} \) **then**

  \( P(E) = P(\text{in } \frac{1}{2} \sqrt{n} \text{ steps we jump inside } \frac{1}{2} \sqrt{n} \text{ of } t) \leq \frac{1}{2} \)

- **Search time \( \geq P(E) \times \#\text{steps} + P(\text{not } E) \times k \)**

  \( \geq \frac{1}{2} \times k = \frac{1}{2} \times \frac{1}{2} \sqrt{n} = O(\sqrt{n}) \)

(next 4 slides give a detailed proof)
Proof: Search time is $\geq O(n^{1/2})$

- We reason about the time needed to get into interval $I$.
- Let: $E_i =$ event that long link out of node $i$ points to some node in interval $I$ of width $2 \cdot x$ nodes (for some $x$) around target $t$.
- Then: $P(E_i) = \frac{2x}{n-1} \approx \frac{2x}{n}$ (in the limit of large $n$)
  (haven’t seen node $i$ yet, but can assume random edge generation)
Proof: Search time is $\geq O(n^{1/2})$

- $E =$ event that any of the first $k$ nodes search algorithm visits has a link to $I$

- Then: $P(E) = P\left(\bigcup_{i=1}^{k} E_i\right) \leq \sum_{i} P(E_i) = k \frac{2x}{n}$

- Let’s choose $k = x = \frac{1}{2} \sqrt{n}$

Then:

$$P(E) \leq 2 \left(\frac{1}{2} \sqrt{n}\right)^2 \frac{1}{n} = \frac{1}{2}$$

Note: Our alg. is deterministic and will choose to travel via a long- or short-range links using some deterministic rule.

The principle of deferred decision tells us that it does not really matter how we reached node $i$. Its prob. of linking to interval $I$ is: $2x/n$. 
Proof: Search time is $\geq O(n^{1/2})$

$P(E) = P(\text{in } \frac{1}{2}\sqrt{n} \text{ steps we jump inside } \frac{1}{2}\sqrt{n} \text{ of } t) \leq \frac{1}{2}$

- **Suppose** initial $s$ is outside $I$ and event $E$ does not happen (first $k$ visited nodes don’t point to $I$)
- **Then** the search algorithm must take $T \geq \min(k, x)$ steps to get to $t$
  - (1) Right after we visit $k$ nodes a good long-range link may occur
  - (2) $x$ and $k$ “overlap”, due to $E$ not happening we have to walk at least $x$ steps
Proof: Search time is $\geq O(n^{1/2})$

- **Claim:** Getting from $s$ to $t$ takes $\geq \frac{1}{4} \sqrt{n}$ steps
- **Search time** $\geq P(E)*(\#steps) + P(\text{not } E)*\min(x,k)$
- **Proof:** We just need to put together the facts

  - We already showed that for $x = k = \frac{1}{2} \sqrt{n}$
    - $E$ does not happen with prob. $\frac{1}{2}$
    - If $E$ does not happen,
      we must traverse $\geq \frac{1}{2} \sqrt{n}$ steps to get to $t$

  - The expected time to get to $t$ is then
    $$\geq P(E \text{ doesn't occur}) \cdot \min\{x, k\} =$$
    $$= \frac{1}{2} \frac{1}{2} \sqrt{n} = \frac{1}{4} \sqrt{n}$$
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”
Model [Kleinberg, Nature ‘01]

- Nodes still on a grid
- Node has one long range link
- Prob. of long link to node $v$:

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

- $d(u,v)$ ... grid distance between $u$ and $v$
- $\alpha$ ... parameter $\geq 0$

$$P(u \rightarrow v) = \frac{d(u,v)^{-\alpha}}{\sum_{w \neq u} d(u,w)^{-\alpha}}$$
Why Does It Work?
We analyze 1-dim case:

- **Claim**: For $\alpha = 1$ we can get from $s$ to $t$ in $O(\log(n)^2)$ steps in expectation.

- **Assume**: For some node $v$: $d(v, t) = d$

- **Set interval**: $I = d$

- **Fact**: (next two slides give a proof of this fact)

$$P\left(\begin{array}{l}
\text{Long range link from } v \\
\text{points to a node in } I
\end{array}\right) = O\left(\frac{1}{\ln n}\right)$$

**Why is this cool?** As $d$ gets bigger, $I$ gets wider, but the prob. is independent of $d$. 
First we need: \( P(\nu \text{ points to } \nu) = \)

\[
P(\nu \rightarrow \nu) = \frac{d(\nu, \nu)^{-1}}{\sum_{\nu \neq v} d(\nu, \nu)^{-1}}
\]

What is the normalizing const?

\[
\sum_{\nu \neq v} d(\nu, \nu)^{-1} = \sum_{\text{all possible distances } d} \frac{1}{d} = 2 \sum_{d=1}^{n/2} \frac{1}{d} \leq 2 \ln n
\]

Note:

\[
\sum_{d=1}^{n/2} \frac{1}{d} \leq 1 + \int_{1}^{n/2} \frac{dx}{x} = 1 + \ln\left(\frac{n}{2}\right) = \ln n
\]
Next we need: \( P(\nu \text{ points to } I) = \)

\[
P(\nu \text{ points to } I) = \sum_{w \in I} P(\nu \rightarrow w) \geq \sum_{w \in I} \frac{d(\nu, w)^{-1}}{2 \ln n}
\]

\[
= \frac{1}{2 \ln n} \sum_{w \in I} \frac{1}{d(\nu, w)} \geq \frac{1}{2 \ln n} d \frac{2}{3d} = \frac{1}{3 \ln n}
\]

What’s the smallest of these terms?

All terms \( \geq 2/(3d) \)

Note:
\( d(\nu, x) = 3d/2 \)
So, we have:

- $I$ ... interval of $d/2$ around $t$ (where $d = d(v, t)$)
- $P$(long link of $v$ points to $I) = 1/\ln(n)$

In expected # of steps $\leq \ln(n)$ you get into $I$, and thus you halve the distance to $t$

How many times do we have to walk $\ln(n)$ steps?

- Distance can be halved at most $\log_2(n)$ times
- So expected time to reach $t$: $O(\log_2(n)^2)$
Overview of the Results

**Searchable**

Search time $T$: \( O((\log n)^\beta) \)

Kleinberg’s model

\( O((\log n)^2) \)

**Not searchable**

Search time $T$: \( O(n^\alpha) \)

Watts-Strogatz

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

Erdős–Rényi

\( O(n) \)

**Note:** We know these graphs have diameter \( O(\log n) \).

So in Kleinberg’s model search time is polynomial in \( \log n \), while in Watts-Strogatz it is exponential (in \( \log n \)).
Kleinberg’s Model: Search Time

We know:

- $\alpha = 0$ (i.e., Watts-Strogatz): We need $O(\sqrt{n})$ steps
- $\alpha = 1$: We need $O(\log(n)^2)$ steps
Intuition: Why Search Takes Long

Small $\alpha$: too many long links

Big $\alpha$: too many short links
**Why Does It Work?**

- **How does the argument change for 2-d grid:**
  
  \[ P(u \text{ points to } I) > \frac{1}{Z} \cdot \text{size}(I) \cdot P(u \rightarrow v) \]

  \[
  \ln n \quad d^2 \quad d^{-2} \Rightarrow \alpha = 2
  \]

- **Why \( P(u \rightarrow v) \sim d(u,v)^{-\text{dim}} \) works?**

  - Approx uniform over all “scales of resolution”
  - # points at distance \( d \) grows as \( d^{\text{dim}} \), prob. \( d^{-\text{dim}} \) of each edge \( \Rightarrow \) const. prob. of a link, independent of \( d \)

  Number of nodes is \( \propto d^2 \)

  Prob. of linking each is \( \propto d^{-2} \)
Different Model: Hierarchies

- **Nodes are in the leaves of a tree:**
  - Departments, topics, ...
- **Create $k$ edges out of a node**
  - Create $i$-th edge out of $v$ by choosing $v \rightarrow w$ with prob. $\sim b^{-h(v,w)}$
    - $h(u,v) =$ tree-distance (height of the least common ancestor)
- **Start at $s$, want to go to $t$**
  - Only see out links of the current node
  - But you know the hierarchy
- **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!**
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
Search in P2P Networks
Algorithmic consequence of small-world:

How to find files in Peer-to-Peer networks?
Client – Server
P2P: Only Clients
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
Protocol Chord 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 (32-bit)
- Key ID is a hash-code of the file
Cycle with node ids 0 to $2^{m-1}$

File (key) $k$ is assigned to a node $a(k)$ with ID $\geq k$
Assume we have $N$ nodes and $K$ keys (files)

How many keys does each node have?

When a node joins/leaves the system it only needs to talk to its immediate neighbors

- When node $N+1$ joins or leaves, then only $O(K/N)$ keys need to be rearranged

Each node knows the IP address of its immediate neighbors
Searching the Network

- If every node knows its immediate neighbor then use sequential search
- Search time is $O(N)$!
Faster Search:

- A node maintains a table of $m = \log(N)$ entries
- $i$-th entry of a node $n$ contains the address of $(2^i)$-th neighbor
  - $i$-th entry points to first node with ID $\geq n + 2^i$
- **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
  - With each step we **halve** the distance to the target!
i-th entry of N has the address of \((N+2^i)\)-th node

\[
\begin{align*}
N8+1 & = N14 \\
N8+2 & = N14 \\
N8+4 & = N14 \\
N8+8 & = N21 \\
N8+16 & = N32 \\
N8+32 & = N42
\end{align*}
\]
Start at N8, find key with ID 54

N8+1 = N14
N8+2 = N14
N8+4 = N14
N8+8 = N21
N8+16 = N32
N8+32 = N42

N42+1 = N48
N42+2 = N48
N42+4 = N48
N42+8 = N51
N42+16 = N1
N42+32 = N14
How Long Does It Take to Find a Key?

- **Claim**: Search for any 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 \( t \)?**
We start the search at node $n$

Let $i$ be a number such that $t$ is contained in interval $[n+2^{i-1}, n+2^i]$ (for some $i$)

Then the table at node $n$ contains a pointer to node $x$ that is the first node past node id $n+2^{i-1}$

**Claim:** Node $x$ is closer to $t$ than $n$

So, in one step we halved the distance to $t$

We can do this at most $\log_2 N$ times

Thus, we find $t$ in $O(\log_2 N)$ steps
Empirical Studies of Navigation in Small-World Networks
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:
    \[ P(u \rightarrow v) \sim 1 / (\text{org. hierarchy distance})^{3/4} \]

Differences from the hierarchical model:
- Data has weighted edges
- Data has people on non-leaf nodes
- Data not b-ary or uniform depth
Generalized hierar. model:
- Arbitrary tree defines “groups” = rooted subtrees
- \( P(u \rightarrow v) \sim \frac{1}{(\text{size of the smallest group containing } u,v)} \)

Search strategies using degree, hierarchy, geo distance between the cubicles
Liben-Nowell et al. ’05:
- LiveJournal data
  - Bloggers + zip codes
- Link prob.: $P(u,v) = \delta^{-\alpha}$
- $\alpha = ?$

Problem:
- Non-uniform population density

Solution: Rank based friendship
Improved Model

\[ P(u \rightarrow v) = rank_u(v)^{-\alpha} \]

- What is best \( \alpha \)?
  - For equally spaced pairs: \( \alpha = \text{dim. of the space} \)
  - In this special case \( \alpha = 1 \) is best for search
Rank Based Friendships

- Close to theoretical optimum of $\alpha = -1$

The difference between the East and West coast disappears!

Liben-Nowell et al. '05
Decentralized search in a LiveJournal network

- 12% searches finish, average 4.12 hops
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 \( \alpha = 1 \) purely through local “rearrangements” of links**
- **Conjecture [Sandbeng-Clark]**
  - Nodes on a ring with random edges
  - Process of morphing links:
    - **Update step:** Randomly choose \( s, t \), run decentral. search alg.
    - **Path compression:** each node on path updates long range link to go directly to \( t \) with some small prob.
- **Conjecture from simulation:** \( P(u \rightarrow v) \sim \text{dist}^{-1} \)
# How the Class Fits Together

## Observations
- Small diameter, Edge clustering
- Patterns of signed edge creation
- Viral Marketing, Blogosphere, Memetracking
- Scale-Free
- Densification power law, Shrinking diameters
- Strength of weak ties, Core-periphery

## Models
- Erdös-Renyi model, Small-world model
- Structural balance, Theory of status
- Independent cascade model, Game theoretic model
- Preferential attachment, Copying model
- Microscopic model of evolving networks
- Kronecker Graphs

## Algorithms
- Decentralized search
- Models for predicting edge signs
- Influence maximization, Outbreak detection, LIM
- PageRank, Hubs and authorities
- Link prediction, Supervised random walks
- Community detection: Girvan-Newman, Modularity