Networks and Markets — Lecture 6: Graphs/Networks Introduction + Algorithms

Nikhil Garg, Cornell Tech

1 Motivation: social networks

One recurring empirical observation in social data is:

> "On average, your friends have more friends than you do."

This is the *friendship paradox*, and it is a good entry point for network thinking.

2 Graphs as models of networks

A network is represented by a graph $G = (V, E)$:

- V: nodes (vertices),
- E: edges (links/interactions).

Examples.

- Road or subway systems.
- Friendship/follow networks.
- The web/Internet.

Graphs can be:

- **Undirected**: edge $\{u, v\}$ means a symmetric relation.
- **Directed**: edge (u, v) means a one-way relation (e.g., "u follows v").

2.1 Basic graph properties

Degree. For an undirected graph, the degree of node i is

$$d_i = \#\{\text{neighbors of } i\}.$$

In directed graphs:

- in-degree = number of incoming links,
- out-degree = number of outgoing links.

Path and connectivity. A path is a sequence of nodes connected by edges. Two nodes are connected if there is a path between them.

Connected components and giant component. A connected component is a maximal set of mutually reachable nodes. Large social networks typically have a *giant component* containing most nodes.

Distance. Distance between two nodes is the length of the shortest path. This leads to notions like radius/diameter (how far nodes can be from each other).

2.2 Worked toy graph

Consider the following graph

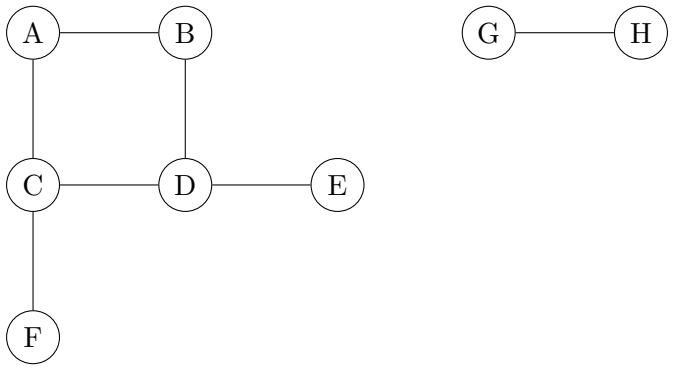

Figure 1: Toy graph used for degree, distance, components, and BFS.

Degrees.

$$d_A = 2,\ d_B = 2,\ d_C = 3,\ d_D = 3,\ d_E = 1,\ d_F = 1,\ d_G = 1,\ d_H = 1.$$

Distances.

$$\mathrm{dist}(A, E) = 3, \qquad \mathrm{dist}(B, F) = 3.$$

Small-world intuition. Classic experiments (e.g., "small world" letter-passing studies) suggested that many human social graphs have surprisingly short path lengths between random pairs of people. The slogan is often "six degrees of separation".

Connected components. There are two components:

$$\{A, B, C, D, E, F\} \quad \text{and} \quad \{G, H\}.$$

3 Friendship paradox

Let d_i be degrees in an undirected graph with n nodes.

Average degree.

$$\bar{d} = \frac{1}{n} \sum_{i=1}^{n} d_i.$$

Average degree of a random friend. If we pick a random edge endpoint (equivalently, a random friend of a random person), then node i is selected with probability

$$\Pr(i) = \frac{d_i}{\sum_j d_j},$$

since node i contributes d_i edge endpoints, and there are $\sum_j d_j$ total endpoints. Therefore,

$$\bar{d}_{\text{friend}} = \sum_i \Pr(i)\, d_i = \sum_i \frac{d_i}{\sum_j d_j}\, d_i = \frac{\sum_i d_i^2}{\sum_j d_j}.$$

Since high-degree nodes appear more often as neighbors, this quantity is typically larger. By Cauchy–Schwarz (or Jensen),

$$\frac{\sum_i d_i^2}{\sum_i d_i} \geq \frac{\sum_i d_i}{n} = \bar{d}.$$

So, on average, your friends have more friends than you do.

3.1 Star-graph example

Suppose there are 5 people in a star graph: one center with degree 4 and four leaves with degree 1.

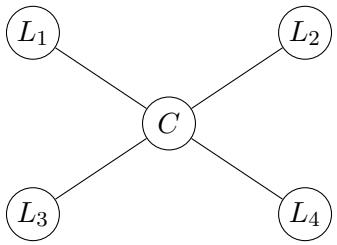

Figure 2: A 5-node star graph: center C and four leaves.

- Average degree:
$$\bar{d} = \frac{4+1+1+1+1}{5} = \frac{8}{5} = 1.6.$$
- Average friend degree:
$$\bar{d}_{\text{friend}} = \frac{4^2+1^2+1^2+1^2+1^2}{4+1+1+1+1} = \frac{20}{8} = 2.5.$$

3.2 Inspection-paradox analog: class-size paradox

If class sizes are unequal, a random *student* is more likely to be in a large class than a small one. So "the average class size reported by students" can be much larger than "the simple average over classes".

Example: two classes with sizes 100 and 2.

- Average over classes: $(100+2)/2 = 51$.

- Average experienced by students: if class sizes are $s_1, \ldots, s_m$, then

$$\bar{s}_{\text{student}} = \sum_{k=1}^{m} \frac{s_k}{\sum_\ell s_\ell} s_k = \frac{\sum_k s_k^2}{\sum_k s_k}.$$

Here,

$$\bar{s}_{\text{student}} = \frac{100^2 + 2^2}{100 + 2} = \frac{10004}{102} \approx 98.1.$$

4 Recommendations from graphs

4.1 Triadic closure

If u is connected to v and v is connected to w, then u and w are more likely to connect. This motivates:

- "People you may know"
- "Who to follow"

by counting mutual neighbors or weighted variants.

Common link-prediction scores. For a candidate non-edge (u, v) with neighbor sets $N(u), N(v)$:

$$\text{CN (Common Neighbors)}(u, v) = |N(u) \cap N(v)|,$$

$$\text{Jaccard}(u, v) = \frac{|N(u) \cap N(v)|}{|N(u) \cup N(v)|},$$

$$\text{AA (Adamic–Adar)}(u, v) = \sum_{w \in N(u) \cap N(v)} \frac{1}{\log d_w}.$$

Using the toy graph above:

- Candidate (A, D): common neighbors $\{B, C\}$, so

$$\text{CN} = 2, \quad \text{Jaccard} = \frac{2}{3}, \quad \text{AA} = \frac{1}{\log 2} + \frac{1}{\log 3}.$$

- Candidate (B, E): common neighbor $\{D\}$, so

$$\text{CN} = 1, \quad \text{Jaccard} = \frac{1}{2}, \quad \text{AA} = \frac{1}{\log 3}.$$

All three scores rank (A, D) above (B, E).

4.2 User–item graphs and collaborative filtering

Represent people and items (movies/products) as a bipartite graph. Edges indicate interactions (view, click, rating, purchase).

One family of methods uses local graph neighborhoods:

- **User-based CF:** find users similar to you (overlap in consumed items), then recommend what they liked.
- **Item-based CF:** find items similar to what you consumed (co-consumption links), then recommend those.

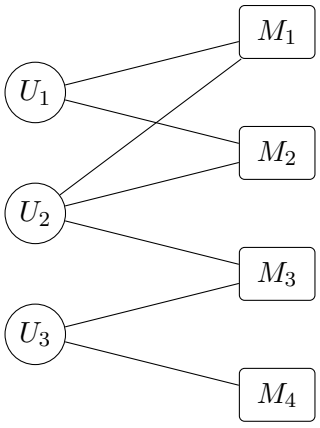

Figure 3: Bipartite user–item graph for a collaborative-filtering example.

Mini-example. Suppose:

$$U_1 : \{M_1, M_2\}, \quad U_2 : \{M_1, M_2, M_3\}, \quad U_3 : \{M_3, M_4\}.$$

For user U_1, a neighborhood method sees U_2 as the closest user (shared M_1, M_2), so it recommends M_3.

When neighborhood methods work well. Local graph similarity is especially effective when local structure is informative (for example, clear communities or strong co-consumption patterns). It can also preserve topic diversity: if a user connects to multiple local neighborhoods, recommendations can draw from each neighborhood rather than a single global direction.

How this differs from matrix factorization. Matrix factorization does not only use immediate overlap counts. It learns latent vectors for users/items,

$$\hat{r}_{ui} \approx p_u^\top q_i,$$

from the whole interaction matrix. So it can capture broader patterns (e.g., hidden "genre/taste" dimensions) and may recommend an item even when direct neighbor overlap is weak. In short:

- neighborhood CF = local graph similarity,
- matrix factorization = global latent-structure fit.

5 Algorithms on graphs

5.1 Breadth-first search (BFS)

BFS explores a graph from a start node by distance: first all nodes at distance 1, then all nodes at distance 2, and so on. It uses a queue and marks nodes as visited so each node is processed once. This gives shortest-path distances in unweighted graphs and naturally defines the layer decomposition below.

BFS layers from above graph, starting at A.

$$L_0 = \{A\},\ L_1 = \{B, C\},\ L_2 = \{D, F\},\ L_3 = \{E\}.$$

Nodes G, H are unreachable from A.

It answers queries like:

- Which nodes are reachable from node A?
- How many nodes are within 2 hops?
- Is node B within k hops?

On unweighted graphs, BFS also computes shortest-path distances from the source.

5.2 PageRank intuition

PageRank (Sergey Brin + Larry Page) scores nodes by recursive importance: a node is important if important nodes link to it. This can be interpreted as a random-surfer process on a directed graph.

Short history. PageRank was developed in the late 1990s by Larry Page and Sergey Brin at Stanford. The "Google" systems paper is *The Anatomy of a Large-Scale Hypertextual Web Search Engine* (1998): `https://snap.stanford.edu/class/cs224w-readings/Brin98Anatomy.pdf` and a dedicated PageRank technical report is *The PageRank Citation Ranking: Bringing Order to the Web* (Stanford InfoLab, 1998): `http://ilpubs.stanford.edu:8090/422/1/1999-66.pdf`

Random-surfer view. Imagine a user repeatedly clicking links:

- with high probability α (often around 0.85; in this example we use 0.8), follow a random outgoing link;
- with probability $1 - \alpha$, "teleport" to a random page.

Pages visited more often in the long run get higher PageRank.

Update equation. If $R_t(i)$ is the score of node i at iteration t, then

$$R_{t+1}(i) = \frac{1-\alpha}{n} + \alpha \sum_{j \to i} \frac{R_t(j)}{\text{outdeg}(j)}.$$

So each node passes its current score evenly across outgoing links. Iterating this update converges to stable scores, which are then used for ranking.

Worked example. Consider the directed graph with edges

$$A \to C, \quad B \to C, \quad B \to D, \quad C \to A, \quad D \to C.$$

With $\alpha = 0.8$ and $n = 4$, the teleport term is $(1 - \alpha)/n = 0.05$. The fixed-point equations are

$$R_A = 0.05 + 0.8\, R_C, \qquad R_B = 0.05,$$

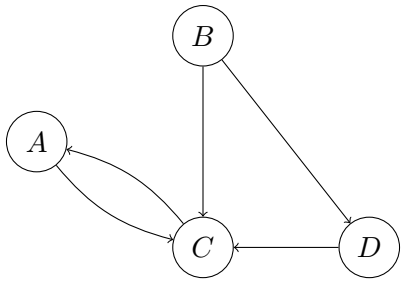

Figure 4: A small directed graph for a PageRank calculation.

$$R_C = 0.05 + 0.8\left(R_A + \frac{R_B}{2} + R_D\right), \qquad R_D = 0.05 + 0.8\,\frac{R_B}{2}.$$

Solving (equivalently, iterating until convergence) gives the ranking C first, then A, then D, then B. This also shows "same in-degree, different importance": nodes A and D both have in-degree 1, but A has much higher PageRank because its incoming link is from higher-ranked C, while D receives only half of B's outflow.

Why this helps. Raw in-degree counts links, but PageRank also weights *who* links to you. Links from highly ranked nodes contribute more than links from low-ranked nodes.

Note: Dangling nodes. If a node j has $\text{outdeg}(j) = 0$, the fraction above is undefined. Standard fixes treat dangling nodes as linking uniformly to all nodes (or equivalently fold them into teleportation), which keeps the process well-defined and convergent.

6 Takeaways

- Graphs are the core abstraction for many social/market systems.
- Recommendation systems heavily rely on graph structure.
- BFS and related graph methods power many practical queries.