Computational Complexity. Lecture 16

Expanders and PRGs

Alexandra Kolla

Today

- The use of PRGs in randomized algorithms.
- Random walks on expanders and Impagliazzo-Zuckerman PRG.
- Quasi-random properties of expanders, expander mixing lemma.

Why Study PRGs?

- Pseudo-random number generators take a seed which is presumably random and generate a long string of random bits that are supposed to act random.
- Why would we want a PRG?
  - Random bits are scarce (eg low-order bits of temperature of the processor in computer is random, but not too many such random bits). Randomized algorithms often need many random bits.
  - Re-run an algorithm for debugging, convenient to use same set of random bits. Can only do that by re-running the PRG with the same seed, but not with truly random bits.

What Type of PRGs?

- Standard PRGs are terrible (e.g. rand in C). Often produce bits that behave much differently than truly random bits.

- One can use cryptography to produce such bits, but much slower

Repeating an Experiment

- Consider wanting to run the same randomized algorithm many times.
- Let A be the algorithm, which returns "yes"/"no" and is correct 99% of the time (correctness function of the random bits)
- Boost accuracy by running A t times and taking majority vote
- Use truly random bits the first time we run A and then with the PRG we will see that every new time we only need 9 random bits.
- If we run t times, probability that majority answer is wrong is exponential in t.

The Random Walk Generator

- Let r be the number of bits out algorithm needs for each run: space of random bits is $\{0,1\}^r$
- Let X⊆ $\{0,1\}^r$ be the settings of random bits on which algorithm gives wrong answer
- Let Y =$\{0,1\}^r \backslash X$ be the settings on which algorithm gives the correct answer

The Random Walk Generator: Expander Graphs

- Our PRG will use a random walk on a d-regular G with vertex set $\{0,1\}^r$, and degree d = constant.
- We want G to be an expander in the following sense: If A_G is G's adjacency matrix and $d = \alpha_1 > \alpha_2 \geq \cdots \geq \alpha_n$ its eigenvalues then we require that

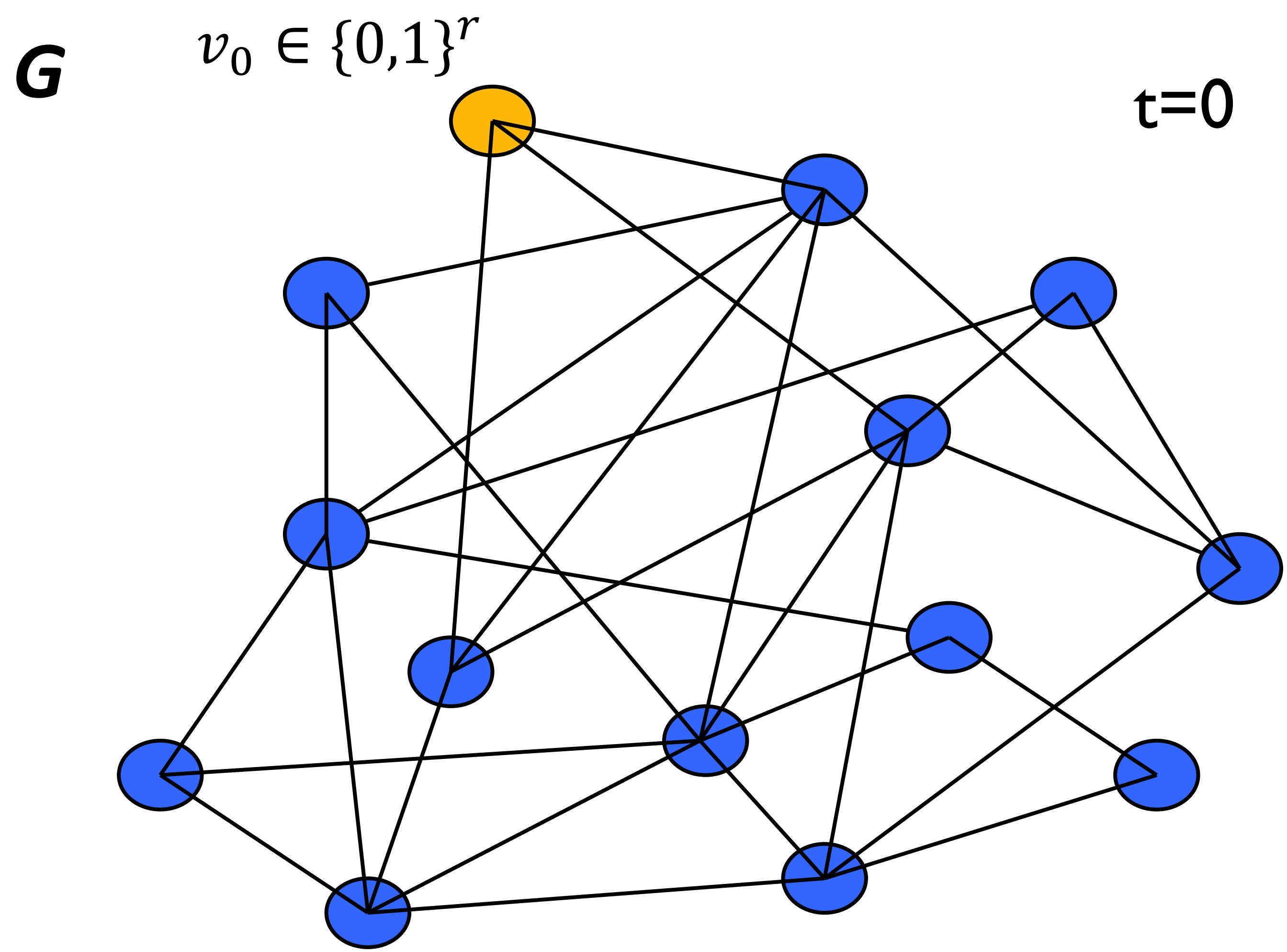
$$\frac{|\alpha_i|}{d} \leq \frac{1}{10}$$

Such graphs exist with d=400 (next lectures)

The Random Walk Generator

- For the first run of algorithm, we require r truly random bits. Treat those bits as vertex of expander G.
- For each successive run, we choose a random neighbor of the present vertex and feed the corresponding bits to our algorithm.
- I.e, choose random i between 1 and 400 and move to the i-th neighbor of present vertex. Need log(400) ~ 9 random bits.
- Need concise description, don't want to store the whole graph (e.g. see hypercube)

The Random Walk Generator

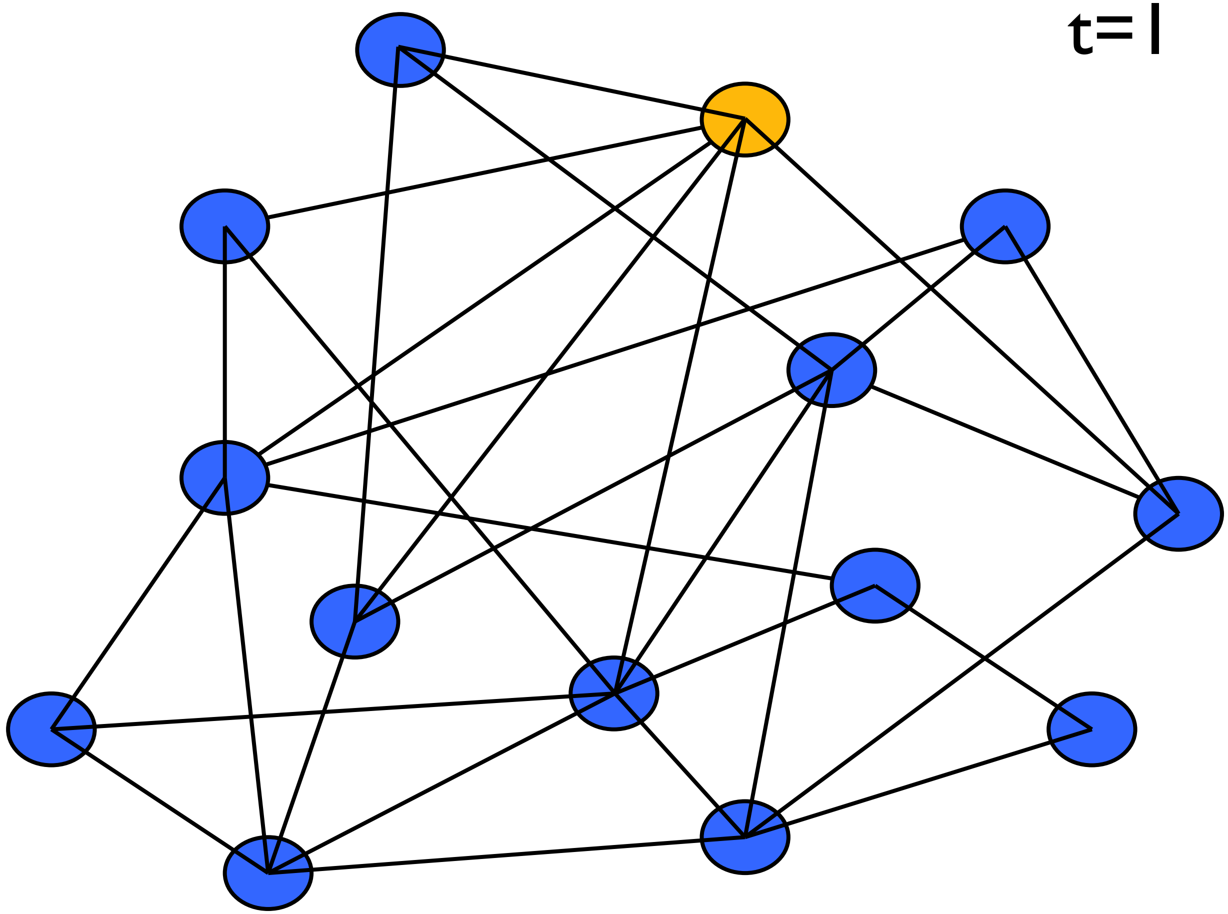


The Random Walk Generator

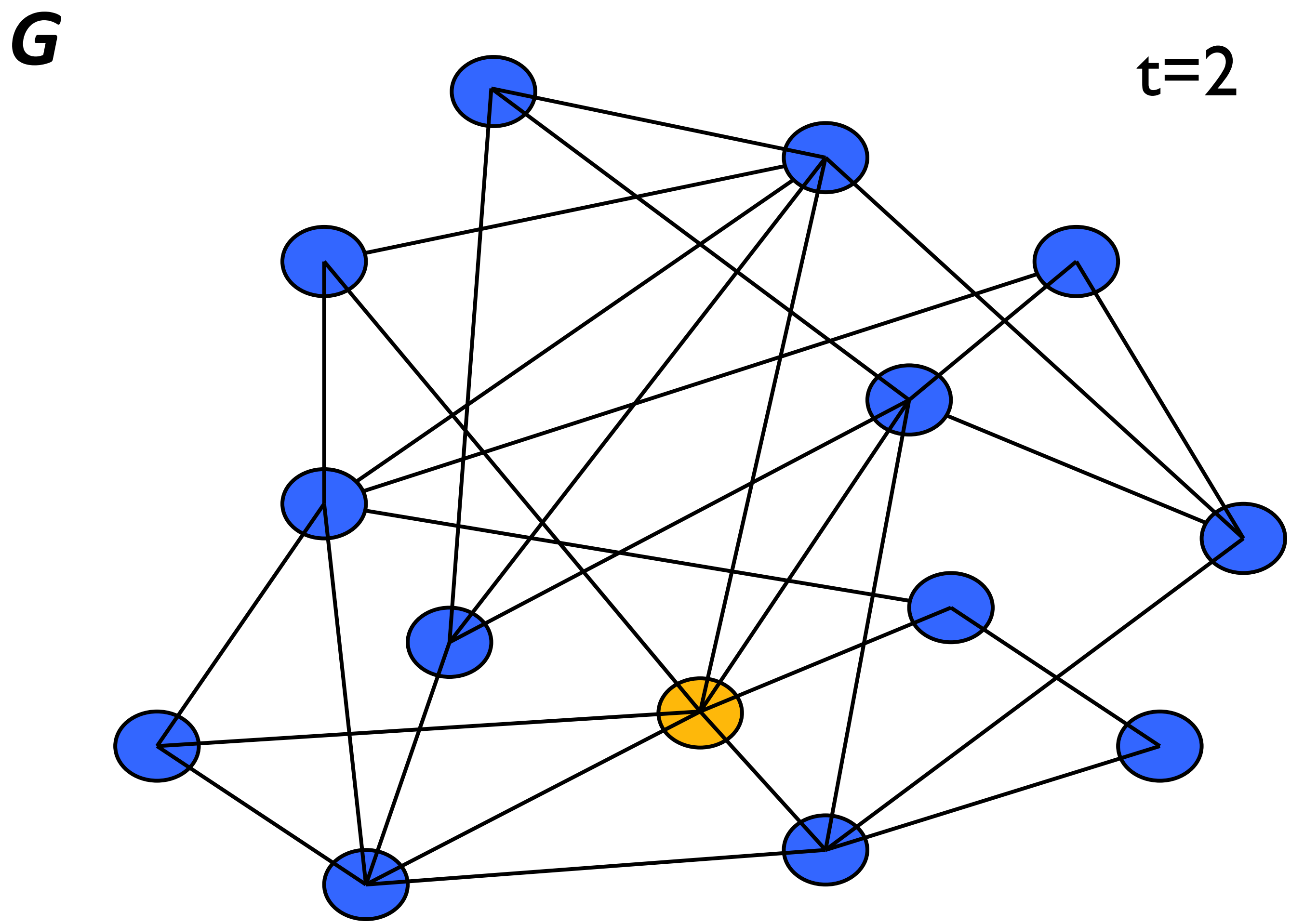
$v_1 \in N(v_0)$

G

t=1

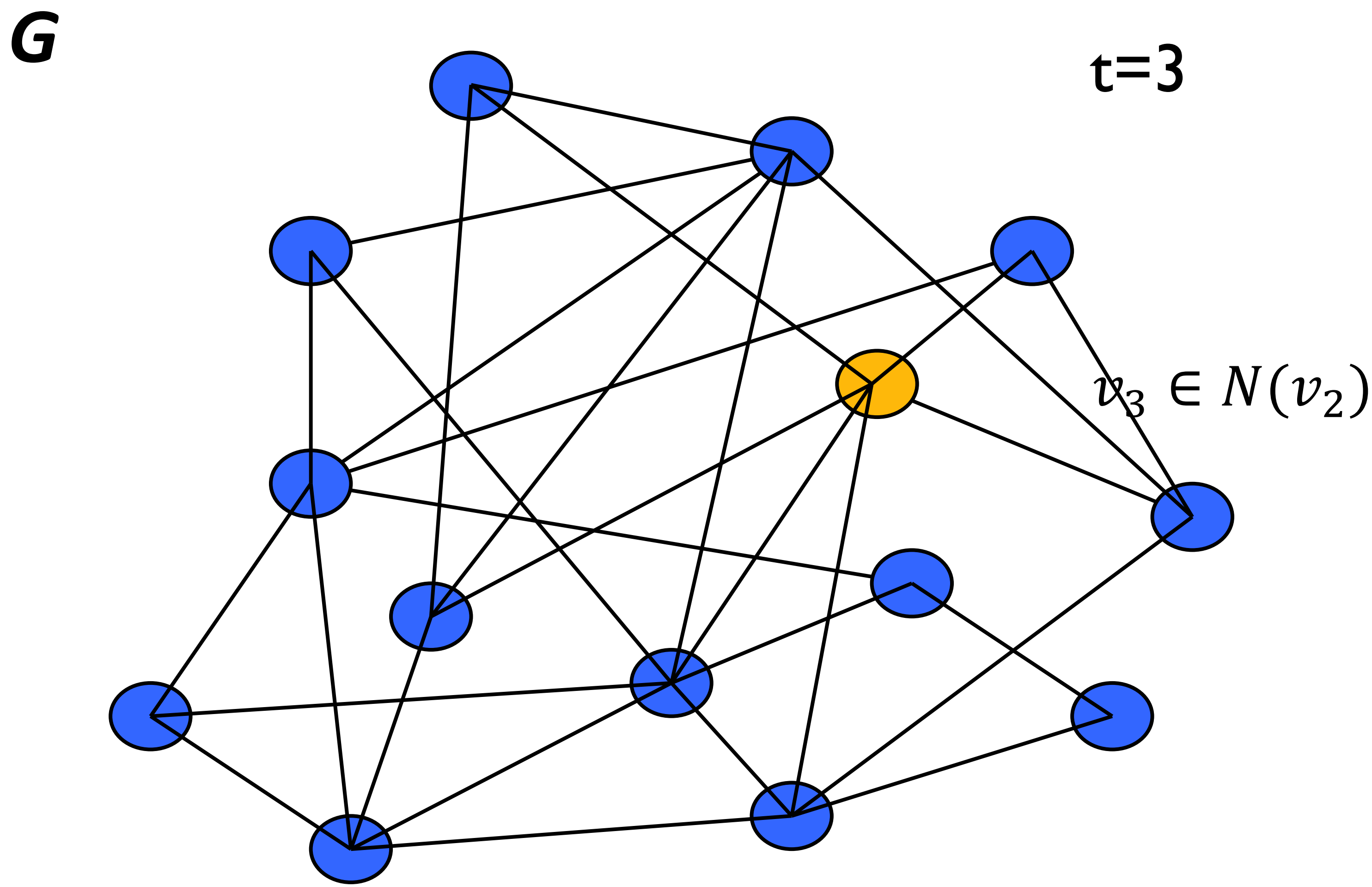


The Random Walk Generator

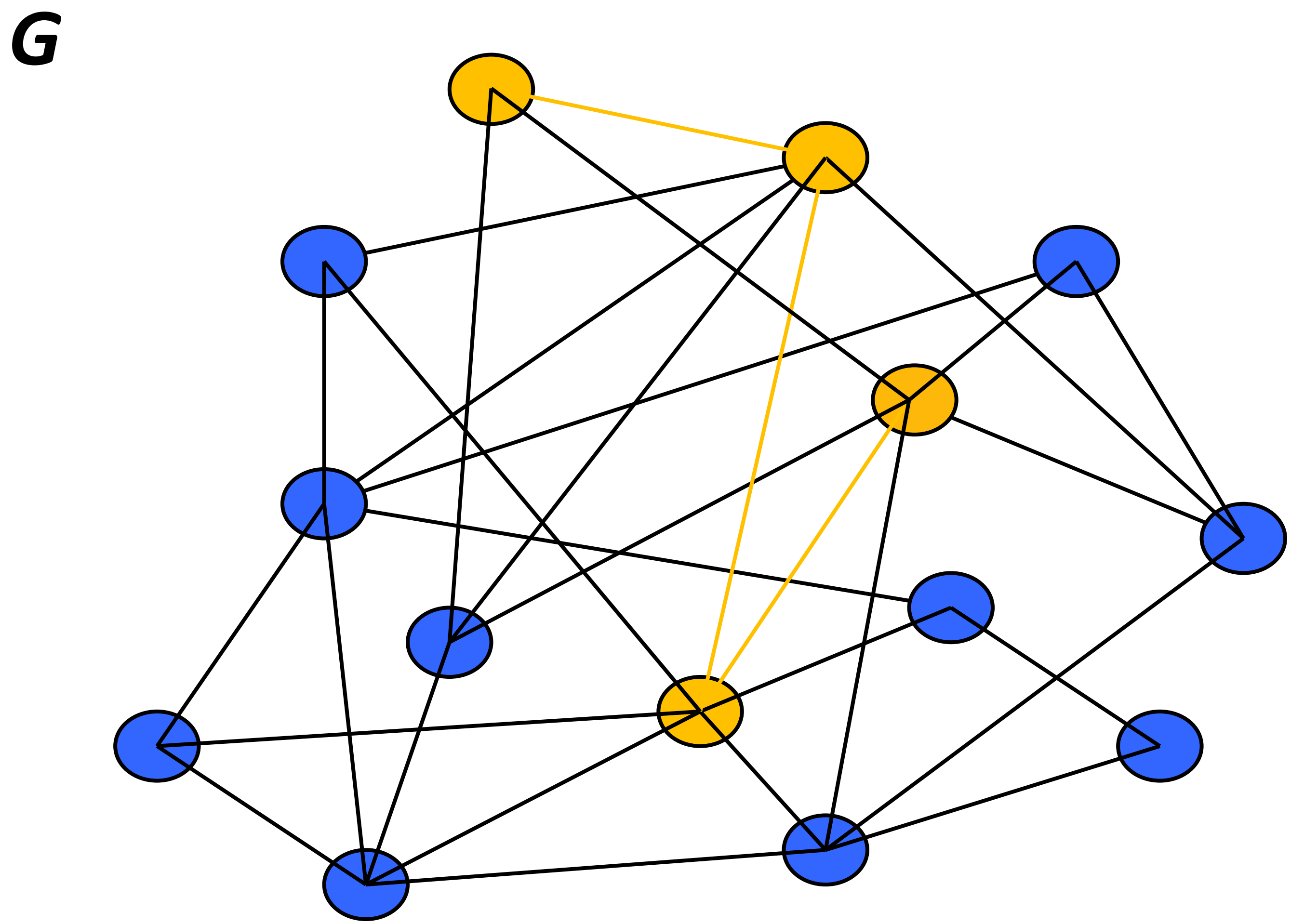


$v_2 \in N(v_1)$

The Random Walk Generator



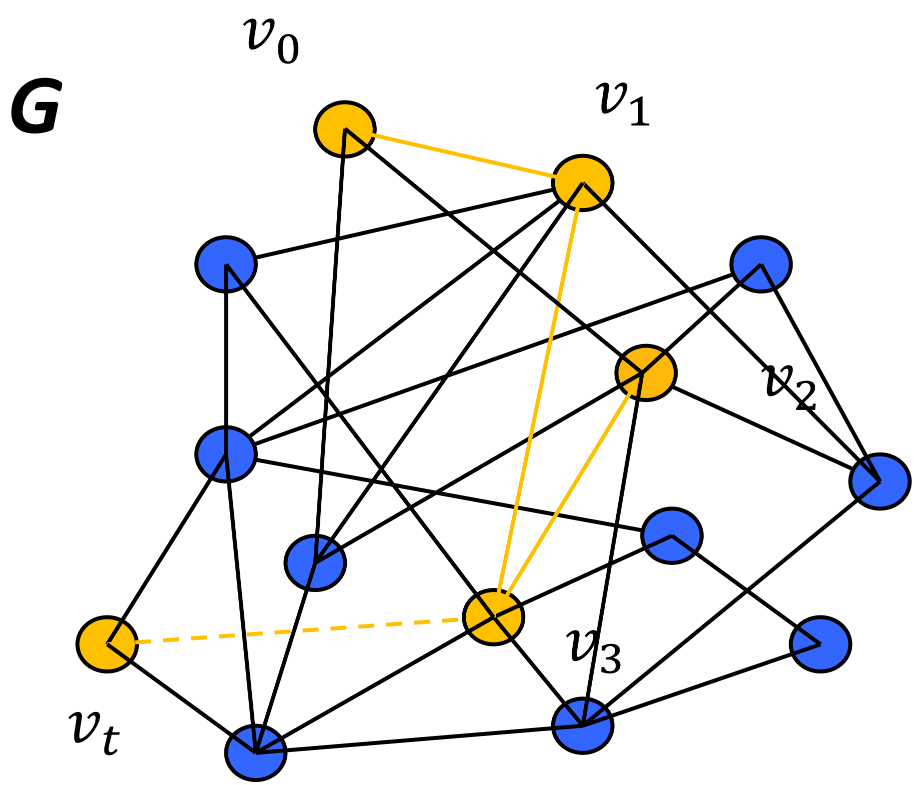
The Random Walk Generator



Formalizing the Problem

- Assume we will run the algorithm t+1 times. Start with truly random vertex u and take t random walk steps.
- Recall that X is the set of vertices on which the algorithm is not correct, we assume that $|X| \leq \frac{2^r}{100}$ (algorithm correct 99% of time)
- If at the end, we report the majority of the t+1 runs of algorithm, then we will return the correct answer as along as the random walk is inside X less than half the time.

The Random Walk Generator



T={o,…,t} time steps

S={i: $v_i \in X$}

We will show that

$$\Pr[|S| > t/2] \leq (\frac{2}{\sqrt{5}})^{t+1}$$

Formalizing the Problem

- Initial distribution is uniform (start with truly random string): $\boldsymbol{p_0} = \mathbf{1}/n$
- Let χ_X and χ_Y the characteristic vectors of X and Y.
- Let $D_X = diag(X)$ and $D_Y = diag(Y)$
- Let $W = \frac{1}{d}A$ (not lazy) random walk matrix, with eigenvalues ω_1,…, ω_n such that $\omega_i \leq \frac{1}{10}$ by the expansion requirement.
- For $|X| \leq \frac{2^r}{100}$,

S={i:$v_i \in X\}$ (time steps that the walk is in X)

we want to show $\Pr[|S| > t/2] \leq (\frac{2}{\sqrt{5}})^{t+1}$

Expander Graphs

- Generally, we defined expander graphs to be d-regular graphs whose adjacency matrix eigenvalues satisfy

$$|\alpha_i| \leq \epsilon d$$

for i>1, and some small ϵ.

Quasi-Random Properties of Expander Graphs

- Expanders act like random graphs in many ways.
- We saw that with random walk on expander, we can boost the error probability like we could do with random walk on a random graph (or truly random stings, Chernoff bound)
- In fact, a random d-regular graph is expander w.h.p.

Quasi-Random Properties of Expander Graphs

- All sets of vertices in expander graph act like random sets of vertices.
- To see that, consider creating a random set S$\subseteq V$ by including every vertex in S independently w.p. a.
- For every edge (u,v) the probability that each end point is in S is a. Probability that both end points are in S is a^2.
- So, we expect a^2 fraction of the edges to go between vertices in S.
- We show that this is true for all sufficiently large sets in an expander.

Quasi-Random Properties of Expander Graphs: EML

- We show something stronger (expander mixing lemma), for two sets S and T.
- Include each vertex in S w.p. a and each vertex in T w.p. b. We allow vertices to belong to both S and T. We expect that for ab fraction of ordered pairs (u,v) we have u in S and v in T.

Expander Mixing Lemma

- For graph G=(V,E) define the ordered set of pairs
  $\overrightarrow{E(S,T)} = \{(u,v): u \in S, v \in T, (u,v) \in E\}$

- When S, T disjoint $\overrightarrow{|E(S,T)|}$ is the number of edges between S and T.
- $\overrightarrow{|E(S,S)|}$ counts every edge inside S twice.

Expander Mixing Lemma, simplified

- **Theorem** (Beigel, Margulis, Spielman'93, Alon, Chung '88)

Let G=(V,E) a d-regular graph with $|\alpha_i| \leq (\epsilon - \frac{1}{n-1})d$, for i>1. Then, for every S⊆V, T⊆V with |S|=an, |T|=bn

$$||\overrightarrow{E(S,T)}| - d\frac{|S||T|}{n}| \leq \epsilon d\sqrt{|S||T|} \Rightarrow$$

$$||\overrightarrow{E(S,T)}| - dabn| \leq \epsilon dn\sqrt{ab}$$