Chapter 9
Some Graph Theory and Randomized Algorithms

Aldous - Fill

September 1 1999

Much of the theory of algorithms deals with algorithms on graphs; conversely, much of the last twenty years of graph theory research pays attention to algorithmic issues. Within these large fields random walks play a comparatively small role, but they do enter in various quite interesting and diverse ways, some of which are described in this chapter. One theme of this chapter is properties of random walks on expander graphs, introduced in sections 1.1 and 1.2. Some non-probabilistic properties of graphs can be explained naturally (to a probabilist, anyway!) in terms of random walk: see section 2. Section 3 reviews the general idea of randomized algorithms, and in section 4 we treat a diverse sample of randomized algorithms based on random walks. Section 5 describes the particular setting of *approximate counting*, giving details of the case of self-avoiding walks. (xxx details not written in this version).

For simplicity let's work in the setting of regular graphs. Except where otherwise stated, G is an n-vertex r-regular connected graph,

$$p_{vw} := r^{-1} 1_{((v,w) \text{ is an edge})}$$

is the transition matrix for discrete-time random walk on G (so $P = r^{-1}A$ for the adjacency matrix A) and $1 = \lambda_1 > \lambda_2 \geq \ldots \geq \lambda_n \geq -1$ are its eigenvalues, and $\tau_2 = 1/(1 - \lambda_2)$.

1 Expanders

1.1 Definitions

The Cheeger time constant τ_c discussed in Chapter 4 section 5.1 (yyy 10/11/94 version) becomes, for a r-regular n-vertex graph,

$$\tau_c = \sup_A \frac{r\ |A|\ |A^c|}{n\ |\mathcal{E}(A, A^c)|}$$

where $\mathcal{E}(A, A^c)$ is the set of edges from a proper subset A of vertices to its complement A^c. Our version of Cheeger's inequality is (Chapter 4 Corollary 37 and Theorem 40) (yyy 10/11/94 version)

$$\tau_c \le \tau_2 \le 8\tau_c^2. \tag{1}$$

Definition. An *expander family* is a sequence G_n of r-regular graphs (for some fixed $r > 2$), with $n \to \infty$ through some subsequence of integers, such that

$$\sup_n \tau_c(G_n) < \infty$$

or equivalently (by Cheeger's inequality)

$$\sup_n \tau_2(G_n) < \infty.$$

One informally says "expander" for a generic graph G_n in the family. The expander property is stronger than the *rapid mixing* property exemplified by the d-cube (Chapter 5 Example 15) (yyy 4/23/96 version). None of the examples in Chapter 5 is an expander family, and indeed there are no known elementary examples. Certain random constructions of regular graphs yield expanders: see Chapter 30 Proposition 1 (yyy 7/9/96 version). Explicit constructions of expander families, in particular the celebrated *Ramanujan graphs*, depend on group- and number-theoretic ideas outside our scope: see the elegant monograph of Lubotzky [40].

Graph parameters like τ_c are more commonly presented in inverted form (i.e. like $1/\tau_c$) as *coefficients of expansion* such as

$$h := \inf_A \frac{|\mathcal{E}(A, A^c)|}{r\ \min(|A|, |A^c|)}. \tag{2}$$

A more familiar version ([17] page 26) of Cheeger's inequality in graph theory becomes, on regular graphs,

$$h^2/2 \leq 1 - \lambda_2 \leq 2h. \tag{3}$$

Since trivially $\tau_c \leq 1/h \leq 2\tau_c$ the two versions agree up to factors of 2. Inequalities involving coefficients of expansion are often called *isoperimetric inequalities*. Expanders and isoperimetric inequalities have been studied extensively in graph theory and the theory of algorithms, e.g. Chung [17] Chapters 2 and 6, the conference proceedings [26], and the introduction of Lubotzky [40].

One algorithmic motivation for Cheeger-type inequalities concerns computational complexity of calculating parameters like τ_c amd h. Using the definition directly requires exponential (in n) time; but because eigenvalues can be calculated in polynomial time, these general inequalities imply that at least crude bounds can be computed in polynomial time.

1.2 Random walk on expanders

If we don't pay attention to numerical constants, then general results about reversible chains easily give us the orders of magnitude of other hitting and mixing time parameters for random walks on expanders.

Theorem 1 *For random walk on an expander family, as $n \to \infty$*

$$\begin{aligned} \tau_1 &= \Theta(\log n) && (4)\\ \tau_0 &= \Theta(n) && (5)\\ \tau^* &= \Theta(n) && (6)\\ \sup_v E_v C &= \Theta(n \log n) && (7) \end{aligned}$$

Proof. Recall the general inequality between τ_1 and τ_2 (Chapter 4 Lemma 23) (yyy 10/11/94 version), which on a regular graph becomes

$$\tau_1 \leq \tau_2(1 + \tfrac{1}{2}\log n). \tag{8}$$

This immediately gives the upper bound $\tau_1 = O(\log n)$. For the lower bound, having bounded degree obviously implies that the diameter Δ of the graph satisfies $\Delta = \Omega(\log n)$. And since the mean distance between an initial vertex v and the position X_T of the walk at a stopping time T is at most ET, the definition of $\tau_1^{(2)}$ implies $d(v,w) \leq 2\tau_1^{(2)}$ for any pair of vertices,

that is $\tau_1^{(2)} \geq \Delta/2$. This establishes (4). The general Markov chain fact $\tau_0 = \Omega(n)$ is Chapter 3 Proposition 14 (yyy 1/26/93 version). Chapter 4 Lemma 25 gives $\tau_0 \leq 2n\tau_2$. Combining these and the obvious inequality $\tau_0 \leq \tau^*/2$ establishes (5,6). Finally, the lower bound in (7) follows from the general lower bound in Chapter 6 Theorem 31 (yyy 10/31/94 version), while the upper bound follows from the upper bound on τ^* combined with Chapter 2 Theorem 36 (yyy 8/18/94 version). ■

In many ways the important aspect of Theorem 1 is that τ_1-type mixing times are of order $\log n$. We spell out some implications below. These hold for arbitrary regular graphs, though the virtue of expanders is that τ_2 is bounded.

Proposition 2 *There exists constants K_1, K_2 such that the following inequalities hold on any regular graph.*

(i) For each vertex v there exists a stopping time T_v such that $P_v(X(T_v) \in \cdot)$ is uniform and $E_v T_v \leq K_1 \tau_2 \log n$.

(ii) For lazy random walk $\widetilde{X}_t$ (with hold-probability 1/2)

$$P_v(\widetilde{X}_t = w) \geq \tfrac{1}{n}(1 - \tfrac{1}{2^j}) \text{ for all } t \geq jK_2\tau_2 \log n \text{ and all vertices } v, w.$$

Proof. Part (i) is just the definition of $\tau_1^{(2)}$, combined with (8) and the fact $\tau_1^{(2)} = O(\tau_1)$.

yyy relate (ii) to Chapter 4 section 3.3 ■

Repeated use of (i) shows that we can get independent samples from π by sampling at random times $T_1, T_2, T_3, \ldots$ with $E(T_{j+1} - T_j) \leq K_1 \tau_2 \log n$. Alternatively, repeated use of (ii) shows that we can get almost independent samples from π by examining the lazy chain at deterministic times, as follows.

Corollary 3 *Fix j and let $t_0 \geq jK_2\tau_2 \log n$. Write $(Y_1, \ldots, Y_L) = (X_{t_0}, \ldots, X_{Lt_0})$. Then*

$$P(Y_1 = y_1, \ldots, Y_L = y_L) \geq n^{-L}(1 - \tfrac{L}{2^j}), \quad \text{for all } L, y_1, \ldots, y_L$$

the variation distance between $\mathrm{dist}(Y_1, \ldots, Y_L)$ and $\pi \times \ldots \times \pi$ is at most $L/2^j$.

Examining the lazy chain at deterministic times means sampling the original walk at random times, but at *bounded* random times. Thus we can get L precisely independent samples using (i) in mean number $K_1 L \tau_2 \log n$ of steps, but without a deterministic upper bound on the number of steps. Using Corollary 3 we get almost independent samples (up to variation distance ε) in a number of steps deterministically bounded by $K_2 L \log(L/\varepsilon) \tau_2 \log n$.

1.3 Counter-example constructions

Constructions with expanders are often useful in providing counter-examples to conjectures suggested by inspecting properties of random walk on the elementary examples of graphs in Chapter 5. For example, consider upper bounds on τ_0 in terms of τ_2 and n, in our setting of regular graphs. From general results for reversible chains in Chapter 4 (10/11/94 version: Lemma 24 and below (9))

$$\max(\tau_2, \tfrac{(n-1)^2}{n}) \le \tau_0 \le (n-1)\tau_2.$$

The examples in Chapter 5 are consistent with a conjecture

$$\tau_0 =^{?} O(\max(n, \tau_2) \log n) \tag{9}$$

where the $\log n$ term is needed for the 2-dimensional torus. We now outline a counter-example.

Take m copies on the complete graph on m vertices. Distinguish one vertex v_i from each copy i. Add edges to make the (v_i) the vertices of a r-regular expander. For this graph G_m we have, as $m \to \infty$ with fixed r,

$$n = m^2;\ \tau_2 = \Theta(m^2);\ \tau_0 = \Theta(m^3)$$

contradicting conjecture (9). We leave the details to the reader: the key point is that random walk on G_m may be decomposed as random walk on the expander, with successive steps in the expander separated by sojourns of times $\Theta(m^2)$ within a clique.

2 Eigenvalues and graph theory

Our treatment of the relaxation time τ_2 in Chapter 4 emphasized probabilistic interpretations in the broad setting of reversible Markov chains. Specializing to random walk on unweighted graphs, there are a range of non-probabilistic connections between eigenvalues of the adjacency matrix and other graph-theoretic properties. Such *spectral graph theory* is the subject of Chung [17]: we shall just give a few results with clear probabilistic interpretations.

2.1 Diameter of a graph

Implicit in the proof of Theorem 1 is that, on a regular graph, the diameter Δ satisfies $\Delta = O(\tau) = O(\tau_2 \log n)$. By being a little careful we can produce numerical constants.

Proposition 4 $\Delta/2 \le \left\lceil \frac{1+\frac{1}{2}\log n}{\log \frac{3-\lambda_2}{1+\lambda_2}} \right\rceil$.

Proof. The discrete analog $\tau_1^{\rm disc}$ of variation threshold satisfies

$$\tau_1^{\rm disc} \ge \Delta/2, \tag{10}$$

because obviously

$$\text{if } d(v,w) = \Delta \text{ and } t < \Delta/2 \text{ then } ||P_v(X_t \in \cdot) - P_w(X_t \in \cdot)|| = 1. \tag{11}$$

Chapter 4 Lemma 26 (yyy 10/11/94 version) specializes to

$$\tau_1^{\rm disc} \le \left\lceil \frac{1 + \frac{1}{2}\log n}{\log 1/\beta} \right\rceil, \quad \beta := \max(\lambda_2, -\lambda_n). \tag{12}$$

We can remove dependence on λ_n by the trick of introducing artificial holds. (yyy tie up with Chapter 4 section 3.3). The chain with transition matrix $P' := \theta I + (1-\theta)P$ has eigenvalues $\lambda_i' = \theta + (1-\theta)\lambda_i$. Choosing $\theta = (1-\lambda_2)/(3-\lambda_2)$, (this being the value making $\lambda_2' = -\lambda_n'$ in the worst case $\theta_n = -1$), we have

$$\tfrac{1+\lambda_2}{3-\lambda_2} = \beta' = \lambda_2' \ge -\lambda_n'.$$

Since (10) still holds for the Markov chain P', combining (10) and (12) with β' establishes the Proposition.

2.2 Paths avoiding congestion

Upper bounding τ_2 via the distinguished paths technique (Chapter 4 section 4.3) (yyy 10/11/94 version) is a valuable theoreticial technique: the essence is to choose paths which "avoid congestion". In the opposite direction, one can use upper bounds on mixing times to show existence of paths which avoid congestion. Here's the simplest result of this type. The first part of the proof repeats an idea from Chapter 4 Lemma 21 (yyy new part of Lemma to be added).

Proposition 5 *Let $v(1), v(2), \ldots, v(n)$ be any ordering of the vertices $1, 2, \ldots, n$ of a r-regular graph. Then there exists, for each $1 \le i \le n$, a path from i to $v(i)$ such that, writing N_{vw} for the number of times the directed edge (v,w) is traversed in all the paths,*

$$\max_{(v,w)} N_{vw} \le 7 \max\left(e\tau_1^{(2)}/r, \log n\right).$$

So on an expander the bound is $O(\log n)$, using (4).

Proof. By definition of $\tau_1^{(2)}$, for each vertex i there is a segment of the chain $i = X_0^{(i)}, X_1^{(i)}, \ldots, X_{U_i}^{(i)}$ such that $EU_i \le \tau_1^{(2)}$ and $X_{U_i}^{(i)}$ has uniform distribution. Take these segments independent as i varies. Write $\tilde{N}_{vw}$ for the (random) number of times that (v, w) is traversed by all these random paths. By considering a uniform random start, by (yyy tie up with Chapter 2 Proposition 3)

$$\tfrac{1}{n} E\tilde{N}_{vw} = \tfrac{1}{rn} \sum_i \tfrac{1}{n} EU_i.$$

In particular, $E\tilde{N}_{vw} \le \tau_1^{(2)}/r := \kappa$. By erasing loops we may contract each path to a path in which no directed edge is traversed twice. For fixed (v, w) let p_i be the chance that (v, w) is traversed by the contracted path from vertex i and let N'_{vw} be the total number of traversals. By independence of paths as i varies,

$$\begin{aligned} P(N'_{vw} \ge m) &\le \left(\sum_i p_i \right)^m / m! \quad \text{(expand the sum)} \\ &\le \kappa^m / m! \le (e\kappa/m)^m. \end{aligned}$$

Choosing $m = \lceil 3 \max(e\kappa, \log n) \rceil$ makes the bound less that $1/(2n^2)$ and so

$$P\left(\max_{(v,w)} N'_{vw} \ge m \right) < 1/2.$$

Now repeat the entire construction to define another copy $(Y_t^{(i)}, 0 \le t \le V_i)$ of chain segments with traversal counts N''_{vw}. Since $X_{U_i}^{(i)}$ and $Y_{V_i}^{(v(i))}$ have the same uniform distribution, for each i we can construct the chain segments jointly such that $X_{U_i}^{(i)} = Y_{V_i}^{(v(i))}$. Concatenating paths gives a (non-Markov) random path from i to $v(i)$. Then

$$P\left(\max_{(v,w)} N'_{vw} + N''_{vw} \ge 2m \right) < 1$$

and so paths with the maximum $\le 2m - 1$ must exist. ■

Broder et al. [13] give a more elaborate algorithm for constructing edge-disjoint paths between specified pairs $\{(a_i, b_i), 1 \le i \le k\}$ of distinct vertices on an expander, for $k = n^{1-o(1)}$. The essential idea is to first pick a set S of $4k$ vertices at random, then use a greedy algorithm to construct (as in the

proof above) paths from each a_i and b_i to some $\tilde{a}_i$ and $\tilde{b}_i$ in S, then for each i construct a bundle of random walk paths from $\tilde{a}_i$ to $\tilde{b}_i$, and finally show that one path may be selected from each bundle so that the set of paths is edge-disjoint.

3 Randomized algorithms

3.1 Background

Here we give some background for the mathematician with no knowledge of the theory of algorithms. Typically there are many different possible algorithms for a particular problem; the theory of algorithms seeks "optimal" algorithms according to some notion of "cost". Cost is usually "time", i.e. number of computational steps, but sometimes involves criteria such as (storage) space or simplicity of coding. The phrase *randomized algorithm* refers to settings where the problem itself does not involve randomness but where randomness is introduced into the running of the algorithm. Why this is useful is best seen by example; the textbook of Motwani and Raghavan [41] provides a comprehensive range of examples and classifications of types of problems where randomized algorithms have proved useful. We give three standard examples below (not using Markov chains) and then proceed to talk about algorithms using random walks on graphs.

Example 6 *Statistical sampling.*

Consider a population of n individuals. Suppose we wish to know the proportion q of the population with some attribute, i.e. who answer "Yes" to some Yes/No question. To calculate q exactly we need to question all n individuals. But if we can sample uniformly at random from the population, then we can estimate q approximately and can bound the size of error in probability. To do this, we sample independently k random individuals, question them, and calculate the empirical proportion $\bar{q}$ of Yes answers. Use $\bar{q}$ as our estimate of q, and theory gives error probabilities

$$P(|\bar{q} - q| > 2(\bar{q}(1-\bar{q}))^{1/2} k^{-1/2}) \approx 5\%.$$

Such 95% confidence intervals are discussed in every freshman statistics course. Classical statististical sampling is conceptually a bit different from algorithms, in that the "cost" k here refers to real-world costs of interviewing human individuals (or experimenting on individual rats or whatever) rather

than to computational cost. However, the key insight in the formula above is that, for prescribed allowable error ε, the cost of this *simple random sampling* is $O(\varepsilon^{-2})$ and this cost does not depend on the "problem size" (i.e. population size) n. The next example is a slightly more subtle use of sampling in a slightly more algorithmic context.

Example 7 *Size of a union of sets.*

It's fun to say this as a word problem in the spirit of Chapter 1. Suppose your new cyberpunk novel has been rejected by all publishers, so you have published it privately, and seek to sell copies by mailing advertizements to individuals. So you need to buy mailing lists (from e.g. magazines and specialist bookshops catering to science fiction). Your problem is that such mailing lists might have much overlap. So before buying L lists $A_1, \ldots, A_L$ (where A_i is a set of $|A_i|$ names and addresses) you would like to know roughly the size $|\cup_i A_i|$ of their union. How can you do this without knowing what the sets A_i are (the vendors won't give them to you for free)? Statistical sampling can be used here. Suppose the vendors will allow you to randomly sample a few names (so you can check accuracy) and will allow you to "probe" whether a few specified names are on their list. Then you can sample k times from each list, and for each sampled name X_{ij} probe the other lists to count the number $m(X_{ij}) \geq 1$ of lists containing that name. Consider the identity

$$\begin{aligned} |\cup_i A_i| &= \sum_i |A_i| \times |A_i|^{-1} \sum_{a \in A_i} 1/m(a) \\ &= \sum_i |A_i| \; E(1/M_i) \end{aligned}$$

where M_i is the number of lists containing a uniform random name from A_i. You can estimate $E(1/M_i)$ by $k^{-1} \sum_{j=1}^{k} 1/m(X_{ij})$, and the error has standard deviation $\leq k^{-1/2}$, and the resulting estimate of $|\cup_i A_i|$ has error

$$\pm O((\sum_i |A_i|^2/k)^{1/2}) = \pm O(k^{-1/2} L \max_i |A_i|).$$

As in the previous example, the key point is that the cost of "approximately counting" $\cup_i A_i$ to within a small relative error does not depend on the size of the sets.

Example 8 *Solovay-Strassen test of primality [43].*

We can't improve on the concise description given by Babai [10].

> Let $n > 1$ be an odd integer. Call an integer w a Solovay-Strassen *witness* (of compositeness of n) if $1 \leq w \leq n-1$ and either g.c.d.$(w, n) > 1$ or $w^{(n-1)/2} \not\equiv (\frac{w}{n}) \bmod n$, where $(\frac{w}{n})$ is the Jacobi symbol (computed via quadratic reciprocity as easily as g.c.d.'s are computed via Euclid's algorithm). Note that no S-S witness exists if n is prime. On the other hand (this is the theorem) if n is composite, then *at least half* of the integers $1, 2, \ldots, n-1$ are S-S witnesses.
>
> Suppose now that we want to decide whether or not a given odd 200-digit integer n is prime. Pick k integers $w_1, \ldots, w_k$ independently at random from $\{1, 2, \ldots, n-1\}$. If any one of the w_i turns out to be a witness, we know that n is composite. If none of them are, let us conclude that n is prime. Here we may err, but for any n, the probability that we draw the wrong conclusion is at most $\varepsilon = 2^{-k}$. Setting $k = 500$ is perfectly realistic, so we shall have proven the mathematical statement "n is prime" beyond the shade of doubt.

3.2 Overview of randomized algorithms using random walks or Markov chains

Our focus is of course on randomized algorithms using random walks or Markov chains. We will loosely divide these into three categories. *Markov chain Monte Carlo* seeks to simulate a random sample from a (usually non-uniform) given probability distribution on a given set. This is the central topic of Chapter 11. In section 4 below we give a selection of miscellaneous graph algorithms. Into this category also falls the idea (Chapter 6 section 8.2) (yyy 10/31/94 version; details to be written) of using random walk as a "undirected graph connectivity" algorithm, and the idea (end of section 2.2) of using random walk paths as an ingredient in constructing edge-disjoint paths in an expander graph. A third, intermediate category is the specific topic of *approximate counting via Markov chains*, to be discussed in section 5.

4 Miscellaneous graph algorithms

4.1 Amplification of randomness

In practice, Monte Carlo simulations are done using deterministic pseudo-random number generators. Ideally one would prefer some physical device which generated "truly random" bits. Presumably any such physical random number generator would be rather slow compared to the speed of arithmetical calculations. This thinking has led to an area of theory in which the cost of a randomized algorithm is taken to be the number of truly random bits used.

Recall the Solovay-Strassen test of primality in Example 8. Philosophically, there is something unsettling about using a deterministic pseudo-random number generator in this context, so we regard this as a prototype example where one might want to use a hypothetical source of truly random bits. To pick a uniform random integer from $\{1, 2, \ldots, n\}$ requires about $\log_2 n$ random bits, so the cost of the algorithm as presented above is about $k \log_2 n = (\log_2 1/\varepsilon)\ (\log_2 n)$ bits, where ε is the prescribed allowable error probability. But one can use the existence of explicit expanders and results like Lemma 12 to devise an algorithm which requires fewer truly random bits. Suppose we have a n-vertex r-regular expander, and label the vertices $\{1, 2, \ldots, n\}$. To simulate a uniform random starting vertex and t steps of the random walk requires about $\log_2 n + t \log_2 r$ bits. The chance that such a walk never hits the set A of witnesses is, by Lemma 12, at most $\exp(-\frac{t}{2\tau_2})$. To make this chance $\leq \varepsilon$ we take $t = 2\tau_2\ \log(1/\varepsilon)$, and the cost becomes $\log_2 n + 2\tau_2(\log_2 r)\ \log(1/\varepsilon)$. Thus granted the existence of expanders on which we can efficiently list neighbors of any specified vertex in order to simulate the random walk, the method of simulating (dependent) integers (w_i) via the random walk (instead of independently) reduces the number of truly random bits required from $O((\log n) \times (\log 1/\varepsilon))$ to $O(\max(\log n,\ \log 1/\varepsilon))$.

The idea of using random walks on expanders for such algorithmic purposes is due to Ajtai et al [2]. Following Impagliazzo and Zuckerman [29] one can abstract the idea to rather general randomized algorithms. Suppose we are given a randomized algorithm, intended to show whether an object $x \in \mathcal{X}$ has a property $\mathcal{P}$ by outputting "Yes" or "No", and that for each x the algorithm is correct with probability $\geq 2/3$ and uses at most b random bits. Formally, the algorithm is a function $A : \mathcal{X} \times \{0,1\}^b \to \{\text{Yes,No}\}$ such that

$$\text{if } x \in \mathcal{P} \text{ then } 2^{-b} |\{\mathbf{i} \in \{0,1\}^b : A(x, \mathbf{i}) = \text{YES}\}| \geq 2/3$$

$$\text{if } x \notin \mathcal{P} \text{ then } 2^{-b}|\{\mathbf{i} \in \{0,1\}^b : A(x,\mathbf{i}) = \text{YES}\}| \leq 1/3$$

where $\mathcal{P} \subset \mathcal{X}$ is the subset of all objects possessing the property. To make the probability of incorrect classification be $\leq \varepsilon$ we may simply repeat the algorithm m times, where $m = \Theta(\log 1/\varepsilon)$ is chosen to make

$$P(\text{Binomial}(m, 2/3) \leq m/2) \leq \varepsilon,$$

and output Yes or No according to the majority of the m individual outputs. This requires $bm = \Theta(b \log 1/\varepsilon)$ random bits. But instead we may take $\{0,1\}^b$ as the vertices of a degree-r expander, and simulate a uniform random starting vertex and m steps of random walk on the expander, using about $b + m \log_2 r$ random bits. For each of the $m+1$ vertices of $\{0,1\}^b$ visited by the walk $(Y_i, 0 \leq i \leq m)$, compute $A(x, Y_i)$, and output Yes or No according to the majority of the $m+1$ individual outputs. The error probability is at most

$$\max_B P_\pi \left(\tfrac{N_{m+1}(B)}{m+1} - \pi(B) \geq \tfrac{1}{3} \right)$$

where $N_{m+1}(B)$ is the number of visits to B by the walk $(Y_i, 0 \leq i \leq m)$. By the large deviation bound for occupation measures (Theorem 11, yyy to be moved to other chapter) this error probability is at most

$$(1 + c_1 m/\tau_2) \ \exp(-c_2 m/\tau_2)$$

for constants c_1 and c_2. To reduce this below ε requires $m = 0(\tau_2 \log(1/\varepsilon))$. Thus the existence of (bounded-degree) expanders implies that the number of random bits required is only

$$b + m \log_2 r = O(\max(b, \ \log 1/\varepsilon))$$

compared to $O(b \log(1/\varepsilon))$ using independent sampling.

4.2 Using random walk to define an objective function

In Chapter 6 section 8.2 (yyy currently at end of this Chapter; to be moved) we gave a standard use of the probabilistic method Here is a less standard use, from Aldous [4], where we use the sample path of a random walk to make a construction.

Consider a function h defined on the vertices of a n-vertex graph G. Constrain h to have no local minima except the global minimum (for simplicity, suppose the values of h are distinct). We seek algorithms to find

the vertex v at which $h(v)$ is minimized. Any deterministic "descent" algorithm will work, but it might work slowly. Could there be some more sophisticated algorithm which always works quickly? One idea is *multi-start descent*. Pick $n^{1/2}$ vertices uniformly at random; from these, choose the vertex with minimum h-value, and follow the greedy descent algorithm. On a degree-d graph, the mean time is $O(dn^{1/2})$. Now specialize to the case where G is the d-cube. One can give examples where single-start (from a uniform random start) descent has mean time $\Omega(2^{(1-\varepsilon)d})$, so from a worst-case mean-time viewpoint, multi-start is better. The next theorem shows that (again from a worst-case mean-time viewpoint), one cannot essentially improve on multi-start descent. Consider random walk on the d-cube started at a uniform random vertex U and let H(v) be the first hitting time on v. Then H is a random function satisfying the constraint, minimized at $v = U$, but

Theorem 9 ([3]) *Every algorithm for locating U by examining values $H(v)$ requires examining a mean number $\Omega(2^{d/2-\varepsilon})$ of vertices.*

The argument is simple in outline. As a preliminary calculation, consider random walk on the d-cube of length $t_0 = O(2^{d/2-\varepsilon})$, started at $\mathbf{0}$, and let L_v be the time of the last visit to v, with $L_v = 0$ if v is not visited. Then

$$EL_v \le \sum_{t=1}^{t_0} tP_{\mathbf{0}}(X(t) = v) = O(1) \tag{13}$$

where the $O(1)$ bound holds because the worst-case v for the sum is $v = \mathbf{0}$ and, switching to continuous time,

$$\int_0^{2^{d/2}} tP_{\mathbf{0}}(\tilde{X}(t) = \mathbf{0})\ dt = \int_0^{2^{d/2}} t\ \left(\frac{1 + e^{-2t/d}}{2}\right)^d\ dt = O(1).$$

Now consider an algorithm which has evaluated $H(v_1), \ldots, H(v_m)$ and write $t_0 = \min_{i \le m} H(v_i) = H(v^*)$ say. It does no harm to suppose $t_0 = O(2^{d/2-\varepsilon})$. Conditional on the information revealed by $H(v_1), \ldots, H(v_m)$, the distribution of the walk $(X(t); 0 \le t \le t_0)$ is specified by
(a) take a random walk from a uniform random start U, and condition on $X(t_0) = v^*$;
(b) condition further on the walk not hitting $\{v_i\}$ before time t_0.
The key point, which of course is technically hard to deal with, is that the conditioning in (b) has little effect. If we ignore the conditioning in (b), then

by reversing time we see that the random variables $(H(v^*) - H(v))^+$ have the same distribution as the random variables L_v (up to vertex relabeling). So whatever vertex v the algorithm chooses to evaluate next, inequality (13) shows that the mean improvement $E(H(v^*) - H(v))^+$ in objective value is $O(1)$, and so it takes $\Omega(2^{d/2-\varepsilon})$ steps to reduce the objective value from $2^{d/2-\varepsilon}$ to 0.

4.3 Embedding trees into the d-cube

Consider again the d-cube $I = \{0,1\}^d$ with Hamming distance $d(\mathbf{i},\mathbf{j})$. Let $\mathcal{B}$ be the vertices of a M-vertex binary tree. For an *embedding*, i.e. an arbitrary function $\rho : \mathcal{B} \to I$, define

$$\text{load} = \max_{\mathbf{i} \in I} \ |\{v \in \mathcal{B} : \rho(v) = \mathbf{i}\}|$$

$$\text{dilation} = \max_{\text{edges } (v,w) \text{ of } \mathcal{B}} d(\rho(v), \rho(w)).$$

How can we choose an embedding which makes both load and dilation small? This was studied by Bhatt and Cai [11], as a toy model for parallel computation. In the model I represents the set of processors, $\mathcal{B}$ represents the set of tasks being done at a particular time, the tree structure indicating tasks being split into sub-tasks. To assign tasks to processors, we desire no one processor to have many tasks (small load) and we desire processors working on tasks and their sub-tasks to be close (small dilation) to facilitate communication. As the computation proceeds the tree will undergo local changes, as tasks are completed and new tasks started and split into sub-tasks, and we desire to be able to update the embedding "locally" in response to local changes in the tree. Bhatt and Cai [11] investigated the natural *random walk embedding*, where the root of $\mathcal{B}$ is embedded at $\mathbf{0}$, and recursively each child w of v is embedded at the vertex $\rho(w)$ found at step L (for even L) of a random walk started at $\rho(v)$. So by construction, dilation $\leq L$, and the mathematical issue is to estimate load. As before, the details are technically complicated, but let us outline one calculation. Clearly load $= \Omega(\max(1, M/2^d))$, so we would like the mean number of vertices of $\mathcal{B}$ embedded at any particular vertex $\mathbf{i}$ to be $O(\max(1, M/2^d))$. In bounding this mean, because $p_{\mathbf{0i}}(t) \leq p_{\mathbf{00}}(t)$ for even t (Chapter 7 Corollary 3) (yyy 1/31/94 version) we see that the worst-case $\mathbf{i}$ is $\mathbf{0}$, and then because $p_{\mathbf{00}}(t)$ is decreasing in t we see that the worst-case M-vertex binary tree is

a maximally balanced tree. Thus we want

$$\sum_{k=0}^{\log_2 M} 2^k p_{\mathbf{00}}(kL) = O(\max(1, M/2^d)). \tag{14}$$

From the analysis of random walk on the d-cube (Chapter 5 Example 15) (yyy 4/23/96 version) one can show

$$p_{\mathbf{00}}(k \log d) = O(\max(d^{-k}, 2^{-d})), \text{ uniformly in } k, d \geq 1.$$

It follows that (14) holds if we take $L = \lceil \log d \rceil$.

Of course to bound the load we need to consider the maximally-loaded vertex, rather than a typical vertex. Considering $M = 2^d$ for definiteness, if the M vertices were assigned independently and uniformly, the mean load at a typical vertex would be 1 and classical arguments show the maximal load would be $\Theta(\frac{d}{\log d})$. We have shown that with tree-embedding the mean load at a typical vertex is $O(1)$, so analogously one can show the maximal load is $O(d/\log d)$. However, [11] shows that by locally redistributing tasks assigned to the same processor, one can reduce the maximal load to $O(1)$ while maintaining the dilation at $O(\log d)$.

4.4 Comparing on-line and off-line algorithms

Here we describe work of Coppersmith et al [19]. As in Chapter 3 section 2 (yyy 9/2/94 version) consider a weighted graph on n vertices, but now write the edge-weights as $c_{ij} = c_{ji} > 0$ and regard them as a matrix $\mathbf{C}$ of *costs*. Let $\mathbf{P}$ be the transition matrix of an irreducible Markov chain whose only transitions are along edges of the graph. For each i and j let $m(i,j)$ be the mean cost of the random walk from i to j, when traversing an edge (v,w) incurs cost c_{vw}. Define the *stretch* $s(\mathbf{P}, \mathbf{C})$ to be the smallest s such that there exists $a < \infty$ such that, for arbitrary $v_0, v_1, \ldots, v_k$

$$\sum_{i=0}^{k-1} m(v_i, v_{i+1}) \leq a + s \sum_{i=0}^{k-1} c_{v_i v_{i+1}}. \tag{15}$$

Note that $c(\mathbf{P}, \mathbf{C})$ is invariant under scaling of $\mathbf{C}$.

Proposition 10 ([19]) *(a)* $s(\mathbf{P}, \mathbf{C}) \geq n - 1$.

(b) If $\mathbf{P}$ *is reversible and* $\mathbf{C}$ *is the matrix of mean commute times* $E_i T_j + E_j T_i$ *then* $s(\mathbf{P}, \mathbf{C}) = n - 1$.

(c) For any cost matrix $\tilde{\mathbf{C}}$ there exists a reversible transition matrix $\mathbf{P}$ with matrix $\mathbf{C}$ of mean commute times such that, for some constant α,

$$c_{ij} \le \alpha \tilde{c}_{ij}$$

$$c_{ij} = \alpha \tilde{c}_{ij} \ when \ p_{ij} > 0.$$

So from (b) and invariance under scaling, $s(\mathbf{P}, \tilde{\mathbf{C}}) = n - 1$.

We shall prove (a) and (b), which are just variations of the standard theory of mean hitting times developed in Chapters 2 and 3. The proof of part (c) involves "convex programming" and is rather outside our scope. The algorithmic interpretations are also rather too lengthy to give in detail, but are easy to say in outline. Imagine a problem where it is required to pick a minimum-cost path, where the cost of a path consists of costs of traversing edges, together with extra costs and constraints. There is some optimal off-line solution, which may be hard to calculate. In such a problem, one may be able to use Proposition 10(c) to show that the algorithm which simply picks a random sample path (with transition matrix $\mathbf{P}$ from (c)) has mean cost not more than $n-1$ times the cost of the optimal path.

Proof. Write π for the stationary distribution of $\mathbf{P}$. Write $m^+(v,v)$ for the mean cost of an excursion from v to v, and write $\bar{c} = \sum_v \sum_w \pi_v p_{vw} c_{vw}$. Then $m^+(v,v) = \bar{c}/\pi_v$ by the ergodic argument (Chapter 2 Lemma 30) (yyy 8/18/94 version). and so

$$\begin{aligned} n\bar{c} &= \sum_v \pi_v m^+(v,v) \\ &= \sum_v \pi_v \sum_w p_{vw}(c_{v,w} + m(w,v)) \\ &= \bar{c} + \sum_v \sum_w \pi_v p_{vw} m(w,v). \end{aligned}$$

In other words,

$$\sum_v \sum_w \pi_w p_{wv} m(v,w) = (n-1)\bar{c}. \tag{16}$$

Now apply the definition (15) of $s(\mathbf{P}, \mathbf{C})$ to the sequence of states visited by the stationary time-reversed chain $\mathbf{P}^*$; by considering the mean of each step,

$$\sum_v \sum_w \pi_v p^*_{vw} m(v,w) \le s(\mathbf{P}, \mathbf{C}) \sum_v \sum_w \pi_v p^*_{vw} c_{vw}. \tag{17}$$

But the left sides of (16) and (17) are equal by definition of $\mathbf{P}^*$, and the sum in the right of (17) equals $\bar{c}$ by symmetry of $\mathbf{C}$, establishing (a). For (b), first note that the definition (15) of stretch is equivalent to

$$s(\mathbf{P}, \mathbf{C}) = \max_{\sigma} \frac{\sum_i m(v_i, v_{i+1})}{\sum_i c_{v_i, v_{i+1}}} \tag{18}$$

where σ denotes a cycle $(v_1, v_2, \ldots, v_m, v_1)$. Write $t(v, w) = E_v T_w$. Fix a cycle σ and write $\mu = \sum_i t(v_i, v_{i+1})$ for the mean time to complete the cyclic tour. By the ergodic argument (Chapter 2 Lemma 30) (yyy 8/18/94 version). the mean number of traversals of an edge (v, w) during the tour is $\mu \pi_y p_{vw}$, and hence the ratio in (18) can be written as

$$\frac{\sum_i t(v_i, v_{i+1})}{\sum_i c_{v_i, v_{i+1}}} \times \sum_v \sum_w \pi_v p_{vw} c_{vw}. \tag{19}$$

Now the hypothesis of (b) is that $\mathbf{P}$ is reversible and $c_{vw} = t(v, w) + t(w, v)$. So the second term of (19) equals $2(n-1)$ by Chapter 3 Lemma 6 (yyy 9/2/94 version) and the first term equals $1/2$ by the cyclic tour property Chapter 3 Lemma 1 (yyy 9/2/94 version). So for each cycle σ the ratio in (18) equals $n-1$, establishing (b).

5 Approximate counting via Markov chains

For a finite set S, there is a close connection between

(a) having an explicit formula for the size $|S|$

(b) having a bounded-time algorithm for generating a uniform random element of S.

As an elementary illustration, we all know that there are $n!$ permutations of n objects. From a proof of this fact, we could write down an explicit $1-1$ mapping f between the set of permutations and the set $A = \{(a_1, a_2, \ldots, a_n) : 1 \le a_i \le i\}$. Then we could simulate a uniform random permutation by first simulating a uniform random element a of A and then computing $f(a)$. Conversely, given an algorithm which was guaranteed to produce a uniform random permutation after $k(n)$ calls to a random number generator, we could (in principle) analyze the working of the algorithm in order to calculate the chance p of getting the identity permutation. Then we can say that number of permutations equals $1/p$.

A more subtle observation is that, in certain settings, having an algorithm for generating an *approximately* uniform random element of S can

be used to estimate *approximately* the size $|S|$. The idea is to estimate successive ratios by sampling. Suppose we can relate S to smaller sets

$$S = S_L \supset S_{L-1} \supset \ldots \supset S_2 \supset S_1 \tag{20}$$

where $|S_1|$ is known, the ratios $p_i := |S_{i+1}|/|S_i|$ are bounded away from 0, and where we can sample uniformly from each S_i. Then take k uniform random samples from each S_i and find the sample proportion W_i which fall into S_{i-1}. Because $|S| = |S_1| \prod_{i=2}^{L} |S_i|/|S_{i-1}|$, we use

$$\hat{N} := |S_1| \prod_{i=2}^{L} W_i^{-1}$$

as an estimate of $|S|$. To study its accuracy, it is simpler to consider $|S|/\hat{N} = \prod_{i=2}^{L} W_i/p_i$. Clearly $E(|S|/\hat{N}) = 1$, and we can calculate the variance by

$$\begin{aligned} \text{var } \left(\frac{|S|}{\hat{N}}\right) &= \text{var } \left(\prod_{i=2}^{L} \frac{W_i}{p_i}\right) \\ &= \prod_{i=2}^{L} (1 + \text{var } (W_i/p_i)) \quad - 1 \\ &= \prod_{i=2}^{L} (1 + \tfrac{1-p_i}{p_i k}) \quad - 1. \end{aligned}$$

The simplest case is where we know a theoretical lower bound p_* for the p_i. Then by taking $k = O(\varepsilon^{-2} L/p_*)$ we get

$$\text{var } \left(\frac{|S|}{\hat{N}}\right) \leq \exp(\tfrac{L}{p_* k}) \ - 1 = O(\varepsilon^2).$$

In other words, with a total number

$$Lk = O(\varepsilon^{-2} L^2/p_*) \tag{21}$$

of random samples, we can statistically estimate $|S|$ to within a factor $1 \pm O(\varepsilon)$.

The conceptual point of invoking intermediate sets S_i is that the overall ratio $|S_1|/|S|$ may be exponentially small in some size parameter, so that trying to estimate this ratio directly by sampling from S would involve order $|S|/|S_1|$, i.e. exponentially many, samples. If we can specify the intermediate

sets with ratios p_i bounded away from 0 and 1 then $L = O(\log(|S|/|S_1|))$ and so the number of samples required in (21) depends on $\log(|S|/|S_1|)$ instead of $|S|/|S_1|$.

The discussion so far has not involved Markov chains. From our viewpoint, the interesting setting is where we cannot directly get uniform random samples from a typical S_i, but instead need to use Markov chain Monte Carlo. That is, on each S_i we set up a reversible Markov chain with uniform stationary distribution (i.e. a chain whose transition matrix is symmetric in the sense $p_{vw} \equiv p_{wv}$) Assume we have a bound τ_1 on the τ_1-values of all these chains. Then as a small modification of Corollary 3, one can get m samples from the combined chains whose joint distribution is close (in variation distance) to the the distribution of independent samples from the uniform distributions in $O(\tau_1 m \log m)$ steps. As above, if we can specify the intermediate sets with ratios p_i bounded away from 0 and 1 then we need $m = O(\varepsilon^{-2} \log^2(|S|/|S_1|))$ samples, and so (ignoring dependence on ε) the total number of steps in all the chains is $O(\tau_1 \log^{2+o(1)}(|S|/|S_1|)$.

In summary, to implement this method of approximate counting via Markov chains, one needs

- a way to specify the intermediate sets (20)
- a way to specify Markov chains on the S_i whose mixing times can be rigorously bounded.

Two particular examples have been studied in detail, and historically these examples provided major impetus for the development of technical tools to estimate mixing times. Though the details are too technical for this book, we outline these examples in the next two sections, and then consider in detail the setting of self-avoiding walks.

5.1 Volume of a convex set

Given a closed convex set K in R^d, for large d, how can we algorithmically calculate the volume of K? Regard K as being described by an *oracle*, that is for any $x \in R^d$ we can determine in one step whether or not $x \in K$. Perhaps surprisingly, there is no known deterministic algorithm which finds $\text{vol}(K)$ approximately (i.e. to within a factor $1 \pm \varepsilon$) in a polynomial in d number of steps. But this problem is amenable to "approximate counting via Markov chains" technique. This line of research was initiated by Dyer et al [24, 25] who produced an algorithm requiring $O(d^{23+o(1)})$ steps. A

long sequence of papers (see the Notes) studied variants of both the Markov chains and the analytic techniques in order to reduce the polynomial degree. Currently the best bound is $O(d^{5+o(1)})$, due to Kannan et al [35].

To outline the procedure in this example, suppose we know $B(1) \subset K \subset B(r)$, where $B(r)$ is the ball of radius $r = r(d)$. (It turns out that one can transform any convex set into one satisfying these constraints with $r = O(d^{3/2})$.) We specify an increasing sequence of convex subsets

$$B(1) = K_0 \subset K_1 \subset \ldots \subset K_L = K$$

by setting $K_i := B(2^{i/d}) \cap K$. This makes the ratios of successive volumes bounded by 2 and requires $L = O(d \log d)$ intermediate sets. So the issue is to design and analyze a chain on a typical convex set K_i whose stationary distribution is uniform. Various Markov chains have been used: simple random walk on a fine discrete lattice restricted to K_i, or spherically symmetric walks. The analysis of the chains has used Cheeger inequalities for chains and the refinement of classical isoperimetric inequalities for convex sets. Recent work of Bubley et al [15] has successfully introduced coupling methods, and it is a challenging problem to refine these coupling methods. There is a suggestive analogy with theoretical study of Brownian motion in a convex set – see Chapter 13 section 1.3 (yyy 7/29/99 version).

5.2 Matchings in a graph

For a finite, not necessarily regular, graph G_0 let $\mathcal{M}(G_0)$ be the set of all matchings in G_0, where a *matching* M is a subset of edges such that no vertex is in more than one edge of M. Suppose we want to count $|\mathcal{M}(G_0)|$ (for the harder setting of counting *perfect* matchings see the Notes). Enumerate the edges of G_0 as $e_1, e_2, \ldots, e_L$, where L is the number of edges of G_0. Write G_i for the graph G with edges $e_1, \ldots, e_i$ deleted. A matching of G_i can be identified with a matching of G_{i-1} which does not contain e_i, so we can write

$$\mathcal{M}(G_{L-1}) \subset \mathcal{M}(G_{L-2}) \subset \ldots \subset \mathcal{M}(G_1) \subset \mathcal{M}(G_0).$$

Since G_{L-1} has one edge, we know $|\mathcal{M}(G_{L-1})| = 2$. The ratio $|\mathcal{M}(G_{i+1})|/|\mathcal{M}(G_i)|$ is the probability that a uniform random matching of G_i does not contain the edge e_{i+1}. So the issue is to design and analyze a chain on a typical set $\mathcal{M}(G_i)$ of matchings whose stationary distribution is uniform. Here is a natural such chain. From a matching M_0, pick a uniform random edge e of G_0, and construct a new matching M_1 from M_0 and e as follows.

If $e \in M_0$ then set $M_1 = M_0 \setminus \{e\}$.

If neither end-vertex of e is in an edge of M_0 then set $M_1 = M_0 \cup \{e\}$.

If exactly one end-vertex of e is in an edge (e' say) of M_0 then set $M_1 = \{e\} \cup M_0 \setminus \{e'\}$.

This construction (the idea goes back to Broder [12]) yields a chain with symmetric transition matrix, because each possible transition has chance $1/L$. An elegant analysis by Jerrum and Sinclair [32], outlined in Jerrum [30] section 5.1, used the distinguished paths technique to prove that on a n-vertex L-edge graph

$$\tau_2 = O(Ln).$$

Since the number of matchings can be bounded crudely by $n!$,

$$\tau_1 = O(\tau_2 \log n!) = O(Ln^2 \log n). \tag{22}$$

5.3 Simulating self-avoiding walks

xxx to be written

6 Notes on Chapter 9

Section 1.1. Modern interest in expanders and their algorithmic uses goes back to the early 1980s, e.g. their use in parallel sorting networks by Ajtai et al [1], and was increased by Alon's [5] graph-theoretic formulation of Cheeger's inequality. The conference proceedings [26] provides an overview.

Edge-expansion, measured by parameters like h at (2), is more relevant to random walk than vertex-expansion. Walters [45] compares definitions. What we call "expander" is often called *bounded-degree expander*.

Section 1.2. Ajtai et al [2], studying the "amplification of randomness" problems in section 4.1, was perhaps the first explicit use of random walk on expanders. In Theorem 1, the upper bounds on τ^* and EC go back to Chandra et al [16].

Section 1.3. With the failure of conjecture (9), the next natural conjecture is: on a r-regular graph

$$\tau_0 =^? O(\max(n, \tau_2) \max(\log n, r)).$$

It's not clear whether such conjectures are worth pursuing.

Section 2. More classical accounts of spectral graph theory are in Cvetkovic et al [21, 20].

On a not-necessarily-regular graph, Chung [17] studies the eigenvalues of the matrix $\mathcal{L}$ defined by

$$\begin{aligned} \mathcal{L}_{vw} &= 1, \quad w = v \\ &= -(d_v d_w)^{-1/2} \text{ for an edge } (v, w) \\ &= 0 \text{ else.} \end{aligned} \tag{23}$$

In the regular case, $-\mathcal{L}$ is the Q-matrix of transition rates for the continuous-time random walk, and so Chung's eigenvalues are identical to our continuous-time eigenvalues. In the non-regular case there is no simple probabilistic interpretation of $\mathcal{L}$ and hence no simple probabilistic interpretation of results involving the relaxation time $1/\lambda_2$ associated with $\mathcal{L}$.

Section 2.1. Chung [17] Chapter 3 gives more detailed results about diameter and eigenvalues. One can slightly sharpen the argument for Proposition 4 by using (11) and the analog of Chapter 4 Lemma 26 (yyy 10/11/94 version) in which the threshold for τ_1^{disc} is set at $1 - \varepsilon$. Such arguments give bounds closer to that of [17] Corollary 3.2: if G is not complete then

$$\Delta \leq \left\lceil \frac{\log(n-1)}{\log \frac{3-\lambda_2}{1+\lambda_2}} \right\rceil .$$

Section 2.2. Chung [17] section 4.4 analyzes a somewhat related *routing problem.* Broder et al [14] analyze a dynamic version of path selection in expanders.

Section 3.1. Example 7 (union of sets) and the more general *DNF counting problem* were studied systematically by Karp et al [36]; see also [41] section 11.2.

The Solovay-Strassen test of primality depends on a certain property of the Jacobi symbol: see [41] section 14.6 for a proof of this property.

Section 4.1. Several other uses of random walks on expanders can be found in Ajtai et al [2], Cohen and Wigderson [18], Impagliazzo and Zuckerman [29].

Section 4.4. Tetali [44] discussions extensions of parts (a,b) of Proposition 10 to nonsymmetric cost matrices.

Section 5. More extensive treatments of approximate counting are in Sinclair [42] and Motwani and Raghavan [41] Chapter 12.

Jerrum et al [33] formalize a notion of *self-reducibility* and show that, under this condition, approximate counting can be performed in polynomial time iff approximately uniform sampling can. See Sinclair [42] section 1.4 for a nice exposition.

Abstractly, we are studying randomized algorithms which produce a random estimate $\hat{a}(d)$ of a numerical quantity $a(d)$ (where d measures the "size" of the problem) together with a rigorous bound of the form

$$P((1-\varepsilon)a(d) \leq \hat{a}(d) \leq (1+\varepsilon)a(d)) \geq 1-\delta.$$

Such a scheme is a *FPRAS* (fully polynomial randomized approximation scheme) if the cost of the algorithm is bounded by a polynomial in d, $1/\varepsilon$ and $\log 1/\delta$. Here the conclusion involving $\log 1/\delta$ is what emerges from proofs using large deviation techniques.

Section 5.1. Other papers on the volume problem and the related problem of sampling from a log-concave distribution are Lovász and Simonovits [38], Applegate and Kannan [8], Dyer and Frieze [23], Lovász and Simonovits [39] and Frieze et al [27].

Section 5.2. In the background is the problem of approximating the *permanent*

$$\mathrm{per}A := \sum_{\sigma} \prod_{i=1}^{n} A_{i\sigma(i)}$$

of a $n \times n$ non-negative matrix, where the sum is over all permutations σ. When A is the adjacency matrix of a $n + n$ bipartite graph, per(A) is the

number of *perfect matchings*. Approximate counting of perfect matchings is in principle similar to approximate counting of all matchings; one seeks to use the chain in section 5.2 restricted to $\mathcal{M}_i \cup \mathcal{M}_{i-1}$, where $\mathcal{M}_i$ is the set of matchings with exactly i edges. But successful analysis of this chain requires that we have a *dense* graph, with minimum degree $> n/2$. Jerrum and Sinclair [31] gave the first analysis, using the Cheeger inequality and estimating expansion via distinguished paths. Sinclair [42] Chapter 3 and Motwani and Raghavan [41] Chapter 11 give more detailed expositions. Subsequently it was realized that using the distinguished paths technique directly to bound τ_2 was more efficient. A more general setting is to seek to sample from the non-uniform distribution on matchings M

$$\pi(M) \propto \lambda^{|M|}$$

for a parameter $\lambda > 1$. The distinguished paths technique [32, 30] giving (22) works in this setting to give

$$\tau_1 = O(Ln^2\lambda \log(n\lambda)).$$

References

[1] M. Ajtai, J. Komlós, and E. Szemerédi. Sorting in $c \log n$ parallel steps. *Combinatorica*, 3:1–19, 1983.

[2] M. Ajtai, J. Komlós, and E. Szemerédi. Deterministic simulation in logspace. In *Proc. 19th ACM Symp. Theory of Computing*, pages 132–140, 1987.

[3] D.J. Aldous. Some inequalities for reversible Markov chains. *J. London Math. Soc. (2)*, 25:564–576, 1982.

[4] D.J. Aldous. Minimization algorithms and random walk on the d-cube. *Ann. Probab.*, 11:403–413, 1983.

[5] N. Alon. Eigenvalues and expanders. *Combinatorica*, 6:83–96, 1986.

[6] N. Alon, U. Feige, A. Wigderson, and D. Zuckerman. Derandomized graph products. *Comput. Complexity*, 5:60–75, 1995.

[7] N. Alon and J. H. Spencer. *The Probabilistic Method.* Wiley, 1992.

[8] D. Applegate and R. Kannan. Sampling and integration of log-concave functions. In *Proc. 23rd ACM Symp. Theory of Computing*, pages 156–163, 1991.

[9] L. Babai. Local expansion of vertex-transitive graphs and random generation in finite groups. In *Proc. 23rd ACM Symp. Theory of Computing*, pages 164–174, 1991.

[10] L. Babai. Probably true theorems, cry wolf? *Notices Amer. Math. Soc.*, 41:453–454, 1994.

[11] S. Bhatt and J-Y Cai. Taking random walks to grow trees in hypercubes. *J. Assoc. Comput. Mach.*, 40:741–764, 1993.

[12] A. Broder. How hard is it to marry at random? (on the approximation of the permanent). In *Proc. 18th ACM Symp. Theory of Computing*, pages 50–58, 1986.

[13] A. Z. Broder, A. M. Frieze, and E. Upfal. Existence and construction of edge disjoint paths on expander graphs. *SIAM J. Comput.*, 23:976–989, 1994.

[14] A. Z. Broder, A. M. Frieze, and E. Upfal. Static and dynamic path selection on expander graphs: a random walk approach. *Random Struct. Alg.*, 14:87–109, 1999.

[15] R. Bubley, M. Dyer, and M. Jerrum. An elementary analysis of a procedure for sampling points in a convex body. *Random Struct. Alg.*, 12:213–235, 1998.

[16] A.K. Chandra, P. Raghavan, W.L. Ruzzo, R. Smolensky, and P. Tiwari. The electrical resistance of a graph captures its commute and cover times. *Comput. Complexity*, 6:312–340, 1996/7. Extended abstract originally published in *Proc. 21st ACM Symp. Theory of Computing* (1989) 574-586.

[17] F.R.K. Chung. *Spectral Graph Theory*, volume 92 of *CBMS Regional Conference Series in Mathematics*. Amer. Math. Soc., 1997.

[18] A. Cohen and A. Widgerson. Dispensers, deterministic amplification, and weak random sources. In *Proc. 30'th IEEE Symp. Found. Comp. Sci.*, pages 14–19, 1989.

[19] D. Coppersmith, P. Doyle, P. Raghavan, and M. Snir. Random walks on weighted graphs and applications to on-line algorithms. *J. Assoc. Comput. Mach.*, 40:421–453, 1993.

[20] D.M. Cvetkovic, M. Doob, I. Gutman, and A. Torgasev. *Recent Results in the Theory of Graph Spectra*. North-Holland, 1988. Annals of Discrete Math. 36.

[21] D.M. Cvetkovic, M. Doob, and H. Sachs. *Spectra of Graphs*. Academic Press, 1980.

[22] I.H. Dinwoodie. A probability inequality for the occupation measure of a reversible Markov chain. *Ann. Appl. Probab.*, 5:37–43, 1995.

[23] M. Dyer and A. Frieze. Computing the volume of convex bodies: A case where randomness provably helps. In B. Bollobás, editor, *Probabilistic Combinatorics And Its Applications*, volume 44 of *Proc. Symp. Applied Math.*, pages 123–170. American Math. Soc., 1991.

[24] M. Dyer, A. Frieze, and R. Kannan. A random polynomial time algorithm for approximating the volume of convex bodies. In *Proc. 21st ACM Symp. Theory of Computing*, pages 375–381, 1989.

[25] M. Dyer, A. Frieze, and R. Kannan. A random polynomial time algorithm for approximating the volume of convex bodies. *J. Assoc. Comput. Mach.*, 38:1–17, 1991.

[26] J. Friedman, editor. *Expanding Graphs*. Amer. Math. Soc., 1993. DIMACS volume 10.

[27] A. Frieze, R. Kannan, and N. Polson. Sampling from log-concave distributions. *Ann. Appl. Probab.*, 4:812–837, 1994.

[28] D. Gillman. A Chernoff bound for random walks on expander graphs. *SIAM J. Comput.*, 27:1203–1220, 1998.

[29] R. Impagliazzo and D. Zuckerman. How to recycle random bits. In *Proc. 30'th IEEE Symp. Found. Comp. Sci.*, pages 248–253, 1989.

[30] M. Jerrum. Mathematical foundations of the Markov chain Monte Carlo method. In *Probabilistic Methods for Algorithmic Discrete Mathematics*, number 16 in Algorithms and Combinatorics, pages 116–165. Springer–Verlag, 1998.

[31] M. Jerrum and A. Sinclair. Approximating the permanent. *SIAM J. Comput.*, 18:1149–1178, 1989.

[32] M. Jerrum and A. Sinclair. The Markov chain Monte Carlo method: an approach to approximate counting and integration. In D. Hochbaum, editor, *Approximation Algorithms for NP-Hard Problems*, pages 482–520, Boston MA, 1996. PWS.

[33] M.R. Jerrum, L.G. Valiant, and V.V. Vazirani. Random generation of combinatorial structures from a uniform distribution. *Theor. Computer Sci.*, 43:169–188, 1986.

[34] N. Kahale. Large deviation bounds for Markov chains. *Combin. Probab. Comput.*, 6:465–474, 1997.

[35] R. Kannan, L. Lovász, and M. Simonovits. Random walks and an $O^*(n^5)$ volume algorithm for convex bodies. *Random Struct. Alg.*, 11:1–50, 1997.

[36] R.M. Karp, M. Luby, and N. Madras. Monte Carlo approximation algorithms for enumeration problems. *J. Algorithms*, 10:429–448, 1989.

[37] P. Lezaud. Chernoff-type bound for finite Markov chains. *Ann. Appl. Probab.*, 8:849–867, 1998.

[38] L. Lovász and M. Simonovits. The mixing rate of Markov chains, an isoperimetric inequality, and computing the volume. In *Proc. 31st IEEE Symp. Found. Comp. Sci.*, pages 346–355, 1990.

[39] L. Lovász and M. Simonovits. Random walks in a convex body and an improved volume algorithm. *Random Struct. Alg.*, 4:359–412, 1993.

[40] A. Lubotzky. *Discrete Groups, Expanding Graphs and Invariant Measures*. Birkhauser, 1994. Progress in Mathematics, vol. 125.

[41] R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.

[42] A. J. Sinclair. *Algorithms for Random Generation and Counting*. Birkhauser, 1993.

[43] R. Solovay and V. Strassen. A fast Monte-Carlo test for primality. *SIAM J. Comput.*, 6:84–85, 1977.

[44] P. Tetali. Design of on-line algorithms using hitting times. Bell Labs, 1994.

[45] I.C. Walters. The ever expanding expander coefficients. *Bull. Inst. Combin. Appl.*, 17:79–86, 1996.

7 Material belonging in other chapters

7.1 Large deviation bounds

yyy Somewhere in the book we need to discuss the results on explicit large deviation bounds for occupation measure / empirical averages: [28, 22, 34, 37]. In section 4.1 we used the following bound from Gillman [28] Theorem 2.1.

Theorem 11

$$P_\mu \left(N_n(B)/n \ - \pi(B) > \gamma \right) \leq \left(1 + \frac{\gamma n}{10 \tau_2} \right) \ \sqrt{\sum_i \mu_i^2 / \pi_i} \ \exp\left(\frac{-\gamma^2 n}{20 \tau_2} \right).$$

7.2 The probabilistic method in combinatorics

yyy This is to be moved to Chapter 6, where we do the "universal traversal sequences" example.

Suppose one wants to show the existence of a combinatorial object with specified properties. The most natural way is to give an explicit construction of an example. There are a variety of settings where, instead of a giving an explicit construction, it is easier to argue that a randomly-chosen object has a non-zero chance of having the required properties. The monograph by Alon and Spencer [7] is devoted to this topic, under the name *the probabilistic method*. One use of this method is below. Two more example occur later in the book: random construction of expander graphs (Chapter 30 Proposition 1) (yyy 7/9/96 version), and the random construction of an objective function in an optimization problem (Chapter 9 section 4.2) (yyy this version).

7.3 Move to Chapter 4 section 6.5

(yyy 10/11/94 version) Combining Corollary 31 with (62) gives the continuous time result below. Recasting the underlying theory in discrete time establishes the discrete-time version.

Lemma 12

$$\begin{aligned} \textit{(continuous time)} \quad & P_\pi(T_A > t) \ \leq \exp(-t\pi(A)/\tau_2), \ t \geq 0 \\ \textit{(discrete time)} \quad & P_\pi(T_A \geq t) \ \leq (1 - \pi(A)/\tau_2)^t, \ t \geq 0. \end{aligned}$$

Notes on this section. In studying bounds on T_A such as Lemma 12 we usually have in mind that $\pi(A)$ is small. One is sometimes interested in exit times from a set A with $\pi(A)$ small, i.e. hitting times on A^c where $\pi(A^c)$ is near 1. In this setting one can replace inequalities using τ_2 or τ_c (which parameters involve the whole chain) by inequalities involving analogous parameters for the chain restricted to A and its boundary. See Babai [9] for uses of such bounds.

On several occasions we have remarked that for most properties of random walk, the possibility of an eigenvalue near -1 (i.e. an almost-bipartite graph) is irrelevant. An obvious exception arises when we consider lower bounds for $P_\pi(T_A > t)$ in terms of $|A|$, because in a bipartite graph with bipartition $\{A, A^c\}$ we have $P(T_A > 1) = 0$. It turns out (Alon et al [6] Proposition 2.4) that a corresponding lower bound holds in terms of $\tau_n \equiv 1/(\lambda_n + 1)$.

$$P_\pi(T_A > t) \geq \left(\max(0, 1 - \tfrac{|A|}{n\tau_n})\right)^t .$$