NOTICE WARNING CONCERNING COPYRIGHT RESTRICTIONS:
The copyright law of the United States (title 17, U.S. Code) governs the making of photocopies or other reproductions of copyrighted material. Any copying of this document without permission of its author may be prohibited by law.

PROBABILISTIC ANALYSIS OF GRAPH ALGORITHMS

by

A. M. Frieze
Department of Mathematics
Carnegie Mellon University
Pittsburgh, PA 15213

Research Report No. 88-42

February 1989

Probabilistic Analysis of Graph Algorithms

by

A.M. Frieze

Department of Mathematics
Carnegie Mellon University
Pittsburgh, PA 15213
U.S.A.

February 1989

Abstract

We give a review of some of the results pertaining to the analysis of the average performance of graph algorithms.

1. INTRODUCTION

Graph theory is an important source of computational problems and as such has played a significant part in the development of a theory of algorithms and their analysis. We find here as elsewhere that the analysis of the execution times of algorithms has concentrated in the main on that of their worst-case. There is nevertheless a sizeable literature on the average case performance of algorithms.

The analysis assumes that the problem instances are randomly selected from some reasonable distribution of problems and an attempt is made to estimate the expected running time of algorithms for these problems. The analytical difficulties are compounded by the fact that algorithms condition their data quickly. Consequently, the statistical independence which is required by most common forms of probabilistic analysis is hard to come by. Probabilistic algorithm analysis has therefore necessitated the development of its own, often indirect, techniques.

In this paper we will try to review some cases where probabilistic analysis has something positive to say about the performance of algorithms. We will look in some detail at eight problems. The first four: the all pairs shortest path problem, the assignment problem, the matching problem in general graphs and the minimum spanning tree problem are all solvable in polynomial time in the worst-case. Nevertheless we will find that algorithms can be constructed whose average performance on natural distributions is significantly better than the worst-case of any known algorithm. The next three: the Hamilton cycle problem, the graph colouring problem and the graph bisection problem are all known to be NP-hard. We will, in spite of this, be able to describe polynomial time algorithms which have a high probability of finding solutions to these problems. Our final example will be that of graph

isomorphism whose exact complexity is at present unknown. Here we will find that a simple algorithm works with high probability. Thus, in these examples and many others, the average case is a long way from the worst-case.

In the next section we introduce some notation and state some basic results needed from probability theory. the next eight sections cover the problems we have mentioned above. Following this we will mention some results with a different flavour.

2. NOTATION AND BASIC PROBABILISTIC INEQUALITIES

We first define what we mean by a random graph. Let $V_n = \{1,2,\ldots,n\}$ and suppose $1 \le m = m(n) \le N = \binom{n}{2}$. The random graph $G_{n,m}$ has vertex set V_n and its edge set $E_{n,m}$ is a randomly chosen subset of m edges. Thus if G is a graph with vertex set V_n and m edges then $\Pr(G_{n,m} = G) = \binom{N}{m}^{-1}$.

There is a closely related model $G_{n,p}$ where $0 \le p = p(n) \le 1$. This has vertex set V_n and edge set $E_{n,p}$ where each of the N possible edges is independently included with probability p. Hence if G is a graph with vertex set V_n and m edges then $\Pr(G_{n,p} = G) = p^m(1-p)^{N-m}$. Observe that if $p = \frac{1}{2}$ then $\Pr(G_{n,\frac{1}{2}} = G) = 2^{-N}$ and so each graph with vertex set V_n is equally likely.

These models have been studied extensively since the pioneering work or Erdös and Renyi [ER1] - [ER4]. The book of Bollobás [Bo1] gives a systematic and extensive account of this subject. A gentler introduction is provided by Palmer [P].

When $m \approx Np$ (i.e. $\lim_{n\to\infty} \left|\frac{m}{Np} - 1\right| = 0$) the graphs $G_{n,m}$ and $G_{n,p}$ have similar properties. Indeed for any graph property $\mathcal{A}$ we have

$$(1.1)\qquad \Pr(G_{n,p} \in d) = \sum_{m=0}^{N} \Pr(G_{n,p} \in d \mid |E_{R,p}| = m)\Pr(|E_{n>p}| = m)$$

$$= \sum_{m=0}^{N} \Pr(G_{n,m} \in d)\ \Pr(|E_{n,p}| = m).$$

since $G_{n,p}$, given $|E_{n,p}| = m$, is precisely $G_{n,m}$. Now $|E_{n,p}|$ is distributed as the binomial random variable $B(N,p)$. So for example, if $m = fNpl$

$$(1.2)\qquad \Pr(G_{n,p} \in si) > \Pr(G_{n,m} \in si)\ \binom{N}{m} p^m (1-p)^{N-m}$$

$$\geq \tfrac{1}{2}\Pr(G_{n,m} \in si)\ (2\pi p(1-p)N)^{1/2}$$

on using Stirling's inequalities for factorials. (1.2) can often be used to show that $\Pr(G_{n,m} \in si)$ is small when $\Pr(G_{n,p} \in d)$ is small.

We are mainly concerned with asymptotic results in this paper and in all cases we will be concerned with what happens as $n \to \infty$. So let $\mathcal{S}^n$ be some event (dependent on n). We say that $\mathcal{S}^n$ occurs *with high* probability (whp) if

$$\lim_{n\to\infty} \Pr(\mathcal{S}^n) = 1.$$

Finally we will note the following bounds on the tails of the binomial

$$(1.3)\qquad \Pr(|B(n,p) - np| \geq \varepsilon np) \leq 2e^{-\frac{\varepsilon^2}{3} np}.$$

(See e.g. [Bo1]).

Thus if $np \to \infty$ and we take $\epsilon = (np)^{-\frac{1}{4}}$ then we see that $B(n,p) \approx np$ whp. By using this in (1.1) we can see that $G_{n,p}$ and $G_{n,\lceil Np \rceil}$ are "similar". We will refer to (1.3) as the Chernoff bound.

3. SHORTEST PATH PROBLEM

In this section we consider the problem of finding a shortest path between all pairs of nodes in a digraph $D= (V,A)$ with non-negative arc lengths $\ell(u)$ for $u \in A$. For notational convenience we assume that D is the complete digraph on V_n. The arc lengths are random and satisfy an "endpoint independence" condition. More precisely the lengths of arcs with different start vertices are independent and if for a given $v \in V_n$ we have $\ell(vw_1) \le \ell(vw_2 \le \ldots \le \ell(vw_{n-1})$ (ties broken randomly) then $w_1, w_2, \ldots, w_{n-1}$ is a random permutation of $V_n - \{v\}$.

We present here an algorithm of Moffat and Takaoka [MT1] which solves the problem in $O(n^2 \log n)$ time. This is to be contrasted with $O(n^3)$ for the best worst-case performance. (See for example Lawler [La] or Papadimitriou and Steiglitz [PS]). The algorithm in [MT1] proceeds as follows:

A: sort the arcs incident with each $v \in V_n$ into increasing order to create list $L(v)$.

B: for each $s \in V_n$ find a shortest path from s to every other vertex.

Since A requires $O(n^2 \log n)$ time we need only prove an $O(n \log n)$ expected time bound for each single source problem in B. The algorithm used is based on one originally attributable to Dantzig [D] and improved and

analysed in the average-case by Spira [Sp]. We first describe this version and then given the contribution of Moffat and Tokakoa.

The algorithm works with a set S. Initially $S = \{s\}$, finally $S = V_n$ and at any stage $v \in S$ means that a shortest path of length $D(v)$ has been found from s to v. If $v \notin S$ then $D(v)$ is an estimate of the shortest path length.

Suppose now that for $v \in S$, $w \notin S$ we let $\lambda(vw) = D(v) + \ell(vw)$ and $D(w) = \min\{\lambda(vw): v \in S\}$. It is easy to show that if $D(x) = \min\{D(w) : w \notin S\}$ then $D(x)$ is the length of a shortest path from s to x.

In the algorithm that follows we keep a priority queue AQ of items $(xy, \lambda(xy))$, one for each $x \in S$, ordered by increasing value of λ.

Algorithm SHORTPATH(s)
begin
    Initialise AQ with (st, λ(st)) where st is the first arc on L(s);
    D(s) := 0; S := {s};
L1: **while** $S \neq V_n$ **do**
    begin
L2: remove the first item (xy, λ(xy)) of AQ; add the item (xy', λ(xy'))
     to AQ where xy' succeeds xy on L(x);
L3: **if** $y \notin S$ **then do**
     begin
       S := S U {y}; D(y) := D(x) + ℓ(xy)
L4:   add (yz, λ(yz)) to AQ where yz is the first arc on L(y) with
       head not in S
     end
    end
end

The above algorithm spends too much time at L3 with $y \in S$. Building on an idea of Fredman (Fd) (rediscovered independently later by Frieze and Grimmett [FG]) Moffat and Takaoka [MT1] "clean up" AQ at line L1 when $|S|$ reaches $n - \frac{n}{2^k}$ for $k = 1,2,\ldots,L = \lfloor \lg\lg n \rfloor$. We shall use $\lg$ to denote $\log_2$ and reserve $\log$ for $\log_e$.

Procedure CLEANUP

begin

E := Ø;

for each $xy \in AQ$ **do**

begin

if $y \notin S$ **then** E := E U {xy}

C1: **else** E := E U {xy'} where xy' is the first arc after xy on L(x) with head not in S.

end

C2: rebuild AQ out of the arcs in E.

end

Analysis

Let Stage k run from $|S| = n - \frac{n}{2^{k-1}}$ to $|S| = n - \frac{n}{2^k}$ for $1 \leq k \leq L = \lfloor \log\log n \rfloor$ and let Stage L+1 denote the final part of the algorithm.

$1 \leq k \leq L$

Let T_k denote $V_n - S$ at the start of Stage k. The probability that $y \notin S$ at L2 of SHORTPATH is always at least $\frac{1}{2}$ since $y \in T_k$, $|V_n - S| \geq \frac{1}{2}|T_k|$ throughout Stage k and y is equally likely to be any member of T_k.

Since $\frac{n}{2^k}$ vertices are added to S in Stage k we expect to execute L3

and hence L2 at most $\frac{n}{2^{k-1}}$ times. Since L2 requires O(logn) time we have

(3.1) E(time spent at L2 in Stage k) = $O(\frac{n}{2^{k-1}} \log n)$ $1 \leq k \leq L$.

To choose z in L4 we expect to examine at most 2^k entries in the list of arcs leaving y. This is because $|V_n - S| \geq \frac{n}{2^k}$ throughout Stage k and the next vertex of y's list is equally likely to be any vertex not encountered so far on this list. Hence

(3.2) E(time spent at L4 in Stage k) = $O(\frac{n}{2^k} 2^k)$ $1 \leq k \leq L$

Now consider CLEANUP. At line C1 we expect to examine at most 2^k arcs before y' is found (some argument as for L4) and so

(3.3) E(time spent at C1 in Stage k) = $O(n2^k)$ $1 \leq k \leq L$

It takes O(n) time to rebuild AQ at C2 and so from (3.1), (3.2), (3.3) we obtain

E(time spent in first L stages)

$$= O\left(\sum_{k=1}^{L} \frac{n}{2^{k-1}} \log n + \sum_{k=1}^{L} n + \sum_{k=1}^{L} n2^k + \sum_{k=1}^{L} n \right)$$

$$= O(n \log n).$$

Let us now consider Stage L+1.

First consider L2. Vertex $y \in T_{L+1}$ and is equally likely to be any member of T_{L+1} that has not yet been examined on x's list. Suppose $|S| = n-s$ at some point. Then we expect to repeat L2 at most $\frac{|T_{L+1}|}{s} = \frac{n}{s \lg n}$ times before finding $y \notin S$. Hence

$$E(\text{time spent at L2 in Stage L+1}) = O(\frac{n}{\log n} \sum_{s=1}^{n/\lg n} \frac{1}{s} \log n)$$

$$= O(n \log n)$$

(the final logn factor is the time to delete the first element of AQ). Now consider L4 and suppose again that $|S| = n-s$. This time we expect to examine at most $\frac{n}{s}$ edges before finding z. Hence

$$E(\text{time spent at L4 in Stage L+1}) = O(n \sum_{s=1}^{n/\lg n} \frac{1}{s})$$

$$= O(n \log n)$$

We have thus shown that algorithm SHORTPATH runs in $O(n^2 \log n)$ expected time. In [MT2] Moffat and Takaoka gave another $O(n^2 \log n)$ expected time algorithm for the same problem. It is not known whether $o(n^2 \log n)$ expected time is achievable for this problem.

4. ASSIGNMENT PROBLEM

In this section we discuss the result of Karp [Kal] that the m x n assignment problem ($m \leq n$) can be solved in O(mn logn) expected time. The analysis can be applied when the matrix of costs ||c(i, j)|| is such that (i) the costs in different rows are independent and (ii) for each i, if $c(i,j_1) \leq c(i,j_2) \leq \ldots \leq c(i,j_n)$ then $j_1 j_2 \ldots j_n$ is a random permutation of {1,2,...,n}. (This is the endpoint independence condition of §3). The proposed algorithm starts with an empty matching and then uses shortest augmenting paths to increase it to size m. The idea of Edmonds and Karp [EK] and Tomizawa [To] is used to ensure that the shortest path problems that need to be solved have non-negative arc lengths.

Let G be the bipartite graph with vertex set V = X U Y where $X = \{x_1, x_2, \ldots, x_m\}$, $Y = \{y_1, y_2, \ldots, y_n\}$ and the cost of edge $x_i y_j$ is $c(i,j)$. We are looking for a minimum cost matching that covers X. If M is any matching of G let D(M) be the digraph with vertex set V and arcs

$x_i y_j$ whenever *edge* $x_i y_j \in M$ forward arc

$y_j x_i$ whenever *edge* $x_i y_j \in M$ backward arc.

Let A = A(M) (resp. B = B(M)) denote the vertices of X (resp. Y) not covered by M.

The following algorithm can be implemented to solve the assignment problem in $O(m^2 n)$ *worst-case* time (for a proof see [EK] or [To]).

Algorithm ASSIGN

begin

$M := \emptyset$

for $v \in V$ **do** $\alpha(v) = 0$ {α is the potential function used to keep arc lengths ≥ 0}

while $|M| < m$ **do**

begin

A. Find a shortest path P from A to B in D(M) where the arc-lengths are given by

$$\overline{\ell}(x_i y_j) = c(i,j) + \alpha(x_i) - \alpha(y_j) \qquad x_i y_j \notin M$$
$$\overline{\ell}(y_j x_i) = -c(i,j) + \alpha(y_j) - \alpha(x_i) \qquad x_i y_j \in M$$

{update M}

Use the alternating path P to alternately add and delete edges to and from M in the normal way.

{update α}

for $v \in V$ **do** $\alpha(v) := \alpha(v) + \tau(v)$

where $\tau(v)$ is the minimum of $\overline{\ell}(P)$ and the length of a shortest $\overline{\ell}$ path from s to v

end

end

To find the shortest paths in A we use a modification of algorithm SHORTPATH of § 3

Changes to SHORTPATH

We create adjacency lists $L(x_i)$, $x_i \in X$, sorted by increasing $c(i,\cdot)$. (For $y \in Y$ either $L(y) = \emptyset$ ($y \in B$) or $L(y)$ consists of the unique vertex of X matched with y by M). We only have time to do the sorting once for each $x \in X$ but on the other hand, at Statement A we need them sorted according to $\bar{\ell}$ and not c. Karp's solution to this problem is rather nice. Define

$$\ell^*(x_iy_j) = c(i,j) + \alpha(x_i) - \alpha^* \qquad x_iy_j \notin M$$

where $\alpha^* = \max\{\alpha(v): \ v \in V\}$.

Observe that

(4.1) $$\ell^*(x_iy_j) \leq \bar{\ell}(x_iy_j) \qquad x_iy_j \notin M$$

and

(4.2) $$\ell^*(x_iy_j) = \bar{\ell}(x_iy_j) \qquad y_j \in B(M).$$

When an item $(uv, D(u) + \bar{\ell}(uv))$ is added to AQ in L2 or L4 we also add a special item $(uv, D(u) + \ell^*(uv))$ unless $v \in B(M)$ or uv is a backward arc of $D(M)$. Also, if the item removed from AQ is special, then it is ignored and the next item of AQ is removed. The point is that we are not necessarily examining the arcs leaving a vertex $x \in X$ in increasing $\bar{\ell}$ order. We want to be sure that the "real" items get to the front of AQ in the order they would in the unmodified SHORTPATH algorithm. Thus we want to be sure that when an item $(xy, D(x) + \ell(xy))$ gets to the front it has a lower value than all competing arcs. But this follows from the fact that if this

item precedes $(uv, D(u) + \ell^*(uv))$ then

$$D(x) + \overline{\ell}(xy) \leq D(u) + \ell^*(uv) \leq D(u) + \overline{\ell}(uw)$$

for all $w \notin S$.

We can also make the simplification that yz in L4 is now to be the first item on $L(y)$. Finally, we will of course start each execution of SHORTPATH with $S = A$ and AQ made from the first items of $L(a)$, $a \in A$ and terminate when $S \cap B \neq \emptyset$.

Analysis

We say that xy is a *virgin* edge if it has not been selected in L2 in any execution of SHORTPATH. The key observation is that if the selection xy in L2 is a virgin edge then

$$\Pr(y \notin B) \geq \frac{|B|}{|Y|}. \tag{4.3}$$

This is because the virgin edges with start node x come to the head of AQ in their (original random) order on $L(x)$ and none of the non-virgin edges with start node x have an end node in B. For when $y \in B$ an augmentation is triggered which means that y gets covered by the new M.

Let Stage k denote the k'th execution of SHORTPATH and σ_k denote the number of virgin edge selections at L2 in Stage k. Then by (4.3) we have

$$E(\sigma_k) \leq \frac{n}{n-k+1}.$$

If an edge ceases to be virgin in Stage k then it can be selected at most 2(m-k+1) times altogether. Hence the expected number of executions of L2 overall is bounded above by

$$2n \sum_{k=1}^{m} \frac{m - k+1}{n - k+1} \leq 2m\, n \qquad \text{(since } m \leq n\text{)}$$

Each such selection requires O(logn) time. The cost of L2 selections and initial sorting dominates the execution time and Karp's result follows.

5. MATCHINGS IN SPARSE RANDOM GRAPHS

Karp and Sipser [KSp] analysed a simple heuristic algorithm for finding a large matching in $G_{n,p}$, $p = \frac{c}{n}$ for a constant $c > 0$. The algorithm runs in O(n) time and produces a near optimal matching **whp**. This is to be compared with the asymptotically most efficient $O(n^{1.5})$ algorithm of Micali and Vazirani [MV] which is much more complex. The analysis is difficult and we will only be able to outline what is going on. (Even so, our treatment is technically at variance in some places with what is said in [KSp]).

First the algorithm: here $\delta(G)$ is the minimum degree of graph G.

Algorithm MATCH

(i) Remove isolated vertices - if G is now empty, stop.

(ii) if $\delta(G) = 1$ choose a random degree 1 vertex v and let vw be its incident edge. Otherwise ($\delta(G) \geq 2$) let vw be a random edge of G.

(iii) add edge vw to the output matching M and then remove vertices v,w from G. Goto (i).

Phase 1 of the algorithm lasts until the first time that $\delta \geq 2$ in Step (ii) and Phase 2 constitutes the remainder of the algorithm.

Let a vertex be *lost* by the algorithm if it is deleted in Step (i) and so is not covered by M. Let $L_i(n,c)$ denote the number of vertices lost in Phase i, i = 1,2. Let $R(n,c)$ denote the number of vertices remaining after Phase 1.

Karp and Sipser prove the following:

Theorem 5.1

For every $\epsilon > 0$

(a) $\lim_{n\to\infty} \Pr\left(\left|\frac{L_1(n,c)}{n} - \alpha(c)\right| > \epsilon\right) = 0$

for some $\alpha(c) > 0$.

(b) $\lim_{n\to\infty} \Pr(L_2(n,c) \geq \epsilon n) = 0$

(c) $\Pr\left(\left| \frac{R(n,c)}{n} - p(c)\right| > \epsilon\right) = 0$

for some $p(c) \geq 0$.

Also $p(c) = 0$ iff $c \leq e = 2.71828...$

□

Now any maximum matching must leave at least $L_1(n,c)$ vertices isolated and so (b) above shows that M is usually of almost optimum size. The final property that $p(c) = 0$ iff $c \leq e$ (the *e-phenomenom*) is remarkable.

Analysis

Let $\mathcal{G}(n_1,n_2,m)$ denote the set of graphs (i) with vertex set $V \subseteq V_n$, (ii) with n_1 vertices of degree 1, (iii) n_2 vertices of degree ≥ 2 and (iv) m edges. Suppose that after first removing the isolated vertices of $G = G_{n,p}$ we have a graph in $\mathcal{G}(n_1,n_2,m)$. It is easy to see that each graph in $\mathcal{G}(n_1,n_2,m)$ is equally likely. (Each such graph arises from a unique $G_{n,p}$ with m edges). More importantly, if we stop the algorithm at the end of any Step (i) and observe the values of n_1,n_2,m then the graph we have is still equally likely to be any of $\mathcal{G}(n_1,n_2,m)$. This is proved inductively by showing that each $G \in \mathcal{G}(n_1,n_2,m)$ can arise from the same number of graphs in $\mathcal{G}(n_1',n_2',m')$ via a single execution of Steps (i)-(iii).

Knowing this, we examine the Markov chain with state space $\{(n_1,n_2,m)\colon n_1 + n_2 \leq n,\ n_1 + 2n_2 \leq m \leq \binom{n}{2},\ n_1,n_2 \geq 0\}$ with transition probabilities defined by the algorithm. Using this we can, for example, examine the length of Phase 1 by seeing how long it is before n_1 becomes zero.

Consider Phase 1. Let $n_1(t)$, $n_2(t)$, $m(t)$ denote the values of n_1,n_2,m at the start of the t^{th} iteration of the algorithm. If in Step (ii) w has degree k and k_i neighbours of degree i, i = 1,2 then we have

$$(5.1)\qquad \begin{aligned} n_1(t) - n_1(t+1) &= k_1 - k_2 + \delta_{k,1} \qquad [\text{Kronecker delta} \\ n_2(t) - n_2(t+1) &= k_2 + 1 - \delta_{k,1} \\ m(t) - m(t+1) &= k + 1 \end{aligned}$$

Now consider a period of time $t \in [\tau n, (\tau + \delta\tau)n]$. If $\delta\tau$ is small one imagines that whp the values $(n_1(t), n_2(t), m(t))$ will be close to some values $ny_1(\tau)$, $ny_2(\tau)$, $ny_3(\tau)$ where y_1,y_2,y_3 are functions of τ only. It would

also be reasonable to assume that whp.

$$n_1(\tau n + 6m) - n\hat{}Tn) \times nfr\hat{}T + 6r) - y_1(T))$$

$$\approx n\ \delta\tau\ y_1'(\tau) \approx \sum_{t=\tau n}^{(T+6T)n} 2\ (kj - kg + 6_{k,j})\ \pounds\ n\ 6T\ E(k_1 - k_2 + \delta_{k,1}).$$

In summary we expect that whp the Markov chain $(n\hat{}iu.m)$ closely follows a path $n(y_1(T),y_2(T),y_3(T))$ for $0 \leq T \leq T = \inf(T : y\hat{}T) = 0)$, where $y(T)$ satisfies

$$(5.2) \qquad y\hat{}(r) = u\hat{}T). \quad i = 1,2,3, \quad 0 \leq r \leq T$$

and the u_1 are the expected values of the RHS of (5.1) at $t = nT$. Furthermore

$$(5.3) \qquad y(0) = (ce^{c},\bar{}1 - e^{c},\bar{}-ce^{c},\bar{}c/2)$$

since the degree of a given vertex in $G_{n,c/n}$ is asymptotically Poisson with mean c.

The formal justification for (5.2) can be obtained by applying a theorem of Kurtz [Kz].

The next question is how to compute the u_i. Consider a graph G chosen randomly from $\hat{}(n\hat{}n\hat{}.m)$. Suppose we know that whp G has approximately v_i vertices of degree i, for $i = 1,2,...$ (thus $v_1 = n_1$ and $2u_1 = 2m$). The study of random graphs with a fixed degree sequence is most easily handled by the configuration model of Bollobas (see [Bol]: if vertex i is of degree d_i

then it gives rise to a set W_i of cardinality d_i. $W = \cup W_i$. A configuration F is a random partition of W into 2-element sets. From F we obtain a multigraph $\mu(F)$ by mapping $\{\alpha,\beta\} \in F$ to uv where $\alpha \in W_u$, $\beta \in W_v$. Conditional on $\mu(F)$ being simple each such graph with the given degree sequence is equally likely. Also if $|W|/(n_1 + n_2) = O(1)$ then the probability of being simple is bounded away from zero by a constant and so we can study random F in place of random $G \in \mathcal{G}(n_1,n_2,m)$.

Returning to the evaluation of u_1,u_2,u_3 in (5.2), we take any i such that $d_i = 1$ and pair the unique $x \in W_i$ with a random element in $W - W_i$. This yields

$$\Pr(k = 1) = \frac{\nu_1}{2m} \; ; \quad \Pr(k = t \geq 2) = \frac{\nu_t t}{2m}. \tag{5.4}$$

By similar reasoning we obtain

$$E(k_1|k) = 1 + (k - 1)\frac{\nu_1}{2m} \Rightarrow E(k_1) = 1 + \frac{\nu_1}{m}(E(k) - 1) \tag{5.5}$$

$$E(k_2|k) = (k - 1)\frac{2\nu_2}{2m} \Rightarrow E(k_2) = \frac{\nu_2}{m}(E(k) - 1) \tag{5.6}$$

$$E(k) = \frac{\mu_1}{2m} + \sum_{t=2}^{\infty} \frac{\nu_t t^2}{2m}. \tag{5.7}$$

Thus we can compute u_1,u_2,u_3 once we have a handle on $\nu_1,\nu_2,\ldots$. Now it is well known that in a random graph with n vertices and average degree d constant that the degree of vertex 1, say, is asymptotically Poisson with mean d. We should not be surprised that if we condition on minimum degree at

least $d_0 \leq d$ then the degree of vertex 1 is asymptotically *truncated* Poisson with parameter θ, i.e.

$$(5.8)\quad \Pr(\text{the degree of vertex 1 is } t \geq d_0) \approx \pi_{\theta,d_0}(t) = \frac{e^{-\theta}\theta^t}{t!}/(e^{-\theta}\sum_{k=d_0}^{\infty}\frac{\theta^k}{k!}).$$

θ must be chosen so that the average degree is still d (to get the number of edges correct) i.e.

$$(5.9)\quad \mu_{\theta,d_0} = \sum_{t=d_0}^{\infty} t\,\pi_{\theta,d_0}(t) = d.$$

(The proof of (5.8), (5.9) is rather long).

Now in our case we can show that, ignoring vertices of degree 1, the degree sequence of what remains is precisely that of a graph with $2m-n_1$ edges, n_2 vertices and minimum degree at least 2. So we now have enough to compute the u_i for (2). Unfortunately, these equations have not been solved explicitly, but at least Part (a) of Theorem 5.1 follows.

The analysis of Phase 2 is more complicated. There we define *clean* states to be those with $y_1 = 0$ and consider transitions from clean state to clean state so that each such transition corresponds to a sequence of iterations of MATCH in which all but the first iteration deletes vertices of degree 1. It is possible to establish differential equations as in (2), (3) which describe the process with high probability. We will not try to establish them here but instead aim to give the barest justification of part (b) of the Theorem.

This will be quite easy if we accept that **whp** a random graph in $\mathcal{G}(0,n_2,m)$, $m \leq cn$ satisfies

(5.10a) no two (*small*) cycles of length $\leq \sqrt{\log n}$ are within distance $\sqrt{\log n}$ of each other.

(5.10b) The number of vertices within distance $\sqrt{\log n}$ of a small cycle is $o(n)$.

It then follows that **whp** the number of lost vertices in a transition from a clean state to a clean state is $O\left(\frac{\#\text{ matching edges found}}{\sqrt{\log n}}\right)$. We leave the justification of this last remark to the reader and note that it implies part (b) of the theorem. We will not attempt to justify the e-phenomenon.

6. MINIMUM SPANNING FORESTS

In this section we consider the problem of finding a minimum weight spanning forest of a graph. Our model of randomness is $G_{n,m}$ with edge weights which when ordered define a random permutation of the edge-set. Remember that it is the edge weight order that defines the minimum weight forest.

Karp and Tarjan [KT] gave an $O(m+n)$ expected time algorithm for this problem based on an algorithm of Cheriton and Tarjan [CT]. This should be compared with the best deterministic algorithm which runs in $O(m\omega(m,n))$ time ([FT] and [GGS]). Here ω is a very slowly growing function of m and n which nevertheless tends to infinity with n. McDiarmid [M1] gave an alternative treatment of a key lemma. The algorithm of [KT] is in two stages:

Stage 1

Step 1a: construct a queue Q of n trees each consisting of a single vertex.

Step 1b: if the queue has at most $\sqrt{n}$ trees go to Stage 2, otherwise delete the first tree T from Q.

Step 1c: let vw be the unexamined (by Step 1c) edge of least weight with one endpoint, v say, in T. (If there are no such edges, go to Step 1b). If $w \in T$ then delete vw and restart Step 1c. Otherwise add vw to the minimum forest F_0 and go to Step 1d.

Step 1d: if the tree T′ containing w is *small* ($\leq \sqrt{n}$ vertices) then delete it from Q and merge T, T′ into a single tree T″. If T″ is small then add it to the rear of Q and go to Step 1b.

At the end of Stage 1 there are at most $2\sqrt{n}$ subtrees. In O(m+n) time we can contract each such tree to a single vertex and reduce the problem to that of finding a minimum forest on $\leq 2\sqrt{n}$ vertices. This requires $O((\sqrt{n})^2) = O(n)$ time. The validity of the algorithm follows, for example, from Lemma 5.2 of Aho, Hopcroft and Ullman [AHU]. The most interesting question from the view of probabilistic analysis is answered by

Lemma 6.1

$$\Pr(w \in T \text{ in Step 1c}) \leq \frac{1}{2}.$$

Proof

Suppose Q contains trees $T_1, T_2, \ldots, T_k$. A vertex $v \in T_i$ is *virgin* if it has never belong to a tree T in Step 1c. It is simple to show by induction that each tree T_i in Q contains exactly one virgin vertex v_i. Now if $v,w \in T \neq T_i$ then the lengths of vw, vv_i have not yet been compared directly or indirectly by the algorithm i.e. interchanging their lengths will not affect the course of the algorithm to this point. On the other hand $k > \sqrt{n} \geq |T|$ and the result follows.

□

The remainder of the analysis is mainly nonprobabilistic. The sets of vertices of the trees of Q are treated as the sets in the UNION-FIND problem in [AHU]. Each set is represented by a tree so that given edge xy say, where $x \in T$, $y \in T'$ it takes O(height (T')) to find out that $y \in T'$ and O(1) time to merge T,T' if $T \neq T'$. When merging, if height (T) $\geq$ height (T') we make the root of T' a child of the root of T and vice-versa. The sets of unexamined edges incident with trees in Q are represented as priority queues. Karp and Tarjan used binomial queues (Vuillemin [V]), but the analysis will be easier, if we use *bottom up skew heaps* from Sleator and Tarjan [ST]. Then if there are k unexamined edges indident with T then it takes O(logk) (amortized) time to remove the one of minimum weight and O(1) time to merge two queues.

The final concept is that of level. Initially imagine a marker placed at the back of Q. All trees (single vertices) are level zero. The marker continually moves to the front and then is placed at the back. If the marker has reached the front ℓ times then we say the trees behind it in Q are at level $\ell + 1$ and those in front are at level ℓ. The following are easy to justify inductively:

(a) A tree of level ℓ contains at least 2^{ℓ} vertices.

(b) Trees of same level are disjoint ($\rightarrow$ at most $\frac{n}{2^{\ell}}$ trees of level ℓ).

(c) height (T) $\leq$ level (T) for $T \in Q$.

Let us now bound the (expected amortized) running time of Phase 1

(i) Total time to find the tree containing w in Step 1c and merge trees in Step 1d.

$$O\left(\sum_{\ell=0}^{\infty} \frac{n}{2^{\ell}} (\ell+1)\right) = O(n)$$

(ii) Total time to find vw in Step 1c

$$O\left(\sum_{\ell=0}^{\infty} \sum_{\text{level}(T)=\ell} \log e(T)\right)$$

(where $e(T) = |\{$unexamined edges incident with T when it reaches front of $Q\}|$)

$$= O\left(\sum_{\ell=0}^{\infty} \frac{n}{2^{\ell}} \log\left(\frac{m2^{\ell}}{n}\right)\right) \quad \text{by concavity of log}$$

$$= O(m).$$

(iii) Total time to merge priority queues in Step 1d

$$= O(n).$$

One can show that it takes O(m) time to initialize the data structures and that amortized time (with a suitable potential function) is within O(m) of actual time. This completes the analysis of the algorithm.

7. HAMILTON CYCLES

Komlós and Szemerédi [KSz] prove the following

Theorem 7.1

Let $m = \frac{1}{2} n \log n + \frac{1}{2} n \log\log n + \frac{1}{2} c_n n$. Then

$$\lim_{n\to\infty} \Pr(G_{n,m} \text{ is Hamiltonian}) =$$

$$\lim_{n\to\infty} \Pr(\delta(G_{n,m}) \geq 2) = \begin{cases} 0 & c_n \to -\infty \\ e^{-e^{-c}} & c_n \to c \\ 1 & c_n \to +\infty \end{cases}$$

□

The aim of this section is to prove the result of Bollobás, Fenner and Frieze [BFF] that there exists an $O(n^{3+o(1)})$ time algorithm HAM satisfying

$$\lim_{n\to\infty} \Pr(\text{HAM finds a Hamilton cycle in } G_{n,m}) = \lim_{n\to\infty} \Pr(G_{n,m} \text{ is Hamiltonian}). \tag{7.1}$$

The most interesting case is where $c_n \to c$. For this we can reformulate (7.1) as

$$\lim_{n\to\infty} \Pr(\text{HAM finds a Hamilton cycle} \mid \delta(G_{n,m}) \geq 2) = 1. \tag{7.2}$$

The following idea has been used extensively: given a path $P = (v_1, v_2, \ldots, v_k)$ and an edge $e = v_i v_{i+1}$ where $1 \leq i \leq k-2$, we can create another path of length $k-1$ by deleting edge $v_i v_{i+1}$ and adding e. Thus let

$$\mathbf{ROTATE(P,e) = (v_1, v_2 \ldots v_i, v_{i+1}, v_{i+2} \ldots v_{i+1}).}$$

HAM proceeds in a sequence of stages. At the beginning of the k'th stage we have a path P_k of length k, with endpoints w_0 and w_j. Stage k ends when we have constructed a path of length k+1 or created a Hamilton cycle. We try to extend P_k from either w_0 or w_j. If we fail, but $w_0 w_j \in E(G_{n,m})$ then, assuming $G_{n,m}$ is connected, as it is **whp,** we can find a longer path than P_k. Failing this, we can create a set of paths of length k by constructing all possible paths of the form ROTATE(P_k,e). These paths at *rotation depth* 1 from P_k are then tested for extension or closure. If none of these yield a path of length k+1 or form a Hamilton cycle then we create all possible paths at rotation depth 2 and so on. The algorithm only gives up trying to find a path of length k+1 or close a Hamilton path (and fails) when it has created all paths at rotation depth 2T where T = [logn/(loglogn - loglogn)].

Now it can be shown that **whp** the number of distinct pairs of endpoints of the paths created grows by a factor of at least logn/1000 as we create each set of paths at a given rotation depth. Thus if HAM fails at any stage there will be a set of an^2 (a > 0 constant) pairs of vertices Z (the distinct pairs of endpoints of the paths created) which depends on the execution of the algorithm, such that if (v,w) € Z then vw € $E(G_{n,m})$.

The final part of the proof is rather unintuitive. It is based on a counting argument of Fenner and Frieze [FF]. In order to get the main idea across we will omit to mention certain technical conditions which hold **whp** and are required for the proof.

Suppose now that HAM fails on $G_{n,m}$ during Stage k. Now P_k is derived from P_0 (= vertex 1) by a sequence of at most 2nT rotations and

extensions. Let $W = W(G_{n,m})$ denote the set of at most $2nT + n$ edges which are involved in these operations.

Now consider the deletion of $w = [\log n]$ random edges X from G^*_m and the following events which are all made conditional on $\delta(G^*_{n,m}) \geq 2$.

$$\mathcal{E}_0 = \{\text{HAM fails to find a Hamilton cycle}\}$$

$$\mathcal{E}_1 = \{\text{HAM fails on } G_{n,m} - X \text{ in the same stage as on } G_{n,m}\}.$$

Observe first that

$$\Pr(\mathcal{E}_1 \mid \mathcal{E}_0) \geq \left(1 - \frac{3nT}{m}\right)^{\omega} \tag{7.3}$$

Since if X avoids $W(G_{n,m})$ then g_1 will occur. But on the other hand, for any fixed graph H with $m-w$ edges

$$\Pr(g_1 \mid G_{n,m} - X = H) \leq (1-a)^{f} \tag{7.4}$$

This is because given H, X is a random w-subset of $E(H)$ and in order that $i \leq$ occur, X must avoid $Z(H)$ which will be of size an^2. (7.3) and (7.4) together show $\Pr(\mathcal{E}_0) = o(1)$ which yields (7.2).

Modifications of these ideas have been used to find Hamilton cycles in sparse random graphs [F1], random directed graphs [F2] and to solve travelling salesman problems [F3].

8. GRAPH COLOURING

In this section we discuss an algorithm which tries to 3-colour graphs. If a graph is chosen uniformly at random from the set of 3-colourable graphs

with vertex set V_n then it succeeds *hp. Our discussion is based on work of **Dyer and Frieze [DF] and Turner [Tu].**

Before getting into this discussion it is as well to briefly state what is known about colouring random graphs in general, say for G ^ where each graph with vertex set V_n is equally likely. Now it has been shown by Bollobás [Bo2] that

$$\chi(G_{n,.5}) \approx \frac{n}{2\log_2 n} \quad * -$$

(See also Luczak [L] for the sparse graph case). In spite of a great deal of effort the best polynomial time algorithms tend to use roughly twice as many colours as are really needed (see e.g. Grimmett and McDiarmid [GM], Bollobás and Erdbs [BE], Shamir and Upfal [SU]).

Having explained this sorry situation we can turn to 3-colourable graphs. (Actually, the proposed methods extend to k-colourable graphs, k fixed - see [DF] or [Tu] for details). It is not obvious how to deal with the probability space ^(n: \ = 3) = the set of 3-colourable graphs with vertex set V_n. We must deal with it indirectly.

First consider a simple way of constructing a random 3-colourable graph. Suppose B_1, B_0, B_2 is a random partition of V into sets of size $\approx \frac{n}{3}$. For each e € ($B_1 \times B_0$ U ($B_1 \times B_2$ U ($B_2 \times B_3$) independently put in the coresponding edge with probability $p \approx \frac{1}{n}$. (We need to allow p close to $\frac{1}{2}r$ as well as $= \frac{1}{2}$). Call the resulting random graph G_1. Clearly G_1 is 3-colourable. Can we 3-colour G_1 **whp** without knowing $B_1.B_2.B_3$? The answer is yes. In the following algorithm Xj.Xg.Xg will (hopefully) denote a 3-colouring of G_1. We use the notation $d_S(v)$ to denote the number of neighbours of a vertex v in a set S.

Algorithm COLOUR

begin

for i: = 1 **to** 2 **do**

begin

X_i: = Ø ; Y_i: = $V_n - \bigcup_{j<i} X_i$

repeat

choose $v \in Y_i$ such that $d_{Y_i}(v)$ is minimal;

X_i: = $X_i \cup \{v\}$; Y_i := $Y_i - \{v\} - \Gamma(v)$

until Y_i = Ø

end

if $X_3 = V_n - \bigcup_{j=1}^{2} X_i$ is independent **then** X_1, X_2, X_3 is a 3-colouring

else COLOUR has failed.

end

(Replace 2 by k-1 and 3 by k to get an algorithm for colouring k-colourable graphs.)

Lemma 8.1

Pr (COLOUR fails) = o(1).

Proof

It is only necessary to show that the first repetition of the for-loop in COLOUR terminates with $X_1 = B_1$ or B_2 or B_3. If this is the case then we are effectively re-applying the algorithm having replaced 3 by 2.

Without loss of generality assume that the first $v \in Y_1$ is in B_1. Suppose inductively that $r \geq 1$ vertices have been selected in X_1 and

suppose $X_1 \subseteq B_1$. Note that $Y_1 = V_n - \Gamma(X_1) - X_1$. If $r \leq 3$ then we can show using the Chernoff bound that for any r-subset X of B_1, $|B_j - \Gamma(X)| \approx \frac{n}{3\times 2^r}$, $j = 2,3$, **whp** and if $v \in B_i$ then it has degree $\approx \frac{n}{3\times 2^{r+1}}$ in $B_j - \Gamma(X)$, $j \neq 1$. Similarly $v \in B_i$, $i \neq 1$ has degree $\approx \frac{n}{6}$ in B_1. Hence under these assumptions

$$d_{Y_1}(v) \approx 2\ \frac{n}{3\times 2^{r+1}}\ , \qquad V \in B_1$$

$$d_{Y_1}(v) \approx \frac{n}{6} + \frac{n}{3\times 2^{r+1}}\ , \qquad V \notin B_1$$

and so the next choice is also in B_1. For $r > 3$ we use the fact that **whp** $|(B_2 \cup B_3) \cap Y_1| \lesssim \frac{n}{12}$ while $v \notin B_1$ retains $\approx \frac{n}{6}$ neighbours in B_1. □

Now we have not yet proved that COLOUR works with high probability on graphs chosen uniformly at random from $\mathcal{G}(n: \chi = 3)$ and we do not have the space here to give all the details of how to "translate" the result of Lemma 8.1 to obtain this result. On the other hand it is easy to show that **whp** G_1 is uniquely 3-colourable, a fact which is of interest in its own right and vital to the "translation".

Lemma 8.2

Pr (G_1 is not uniquely 3-colourable) = o(1).

Proof (Outline)

Consider a vertex $v \in B_1$. **Whp** it has $\approx \frac{n}{6}$ neighbours $N_i \subseteq B_i$, $i = 1,2$. **Whp** $N_1 \cup N_2$ induces a connected, and hence uniquely 2-colourable, bipartite graph. But then **whp** each $w \in B_1 - \{v\}$ is adjacent to a vertex in

both N_1 and N_2 and so B_1 is determined as one colour class. Finally, whp $B_2 \cup B_2$ induces a connected bipartite graph which then determines B_2, B_3 uniquely.

□

We can now discuss the "translation" of Lemma 8.1. Let G_2 be the random graph in which we randomly choose $m \approx \frac{n^2}{6}$ edges from $(B_1 \times B_2) \cup (B_1 \times B_3) \cup (B_2 \times B_3)$ where B_1, B_2, B_3 are as for G_1. If we let $p = \frac{3m}{n^2} \approx \frac{1}{2}$ then we have

$$\Pr(\text{COLOUR fails on } G_1) \geq \Pr(\text{COLOUR fails on } G_1 \mid |E(G_1)| = m) \Pr(|E(G_1)| = m)$$

$$= \Pr(\text{COLOUR fails on } G_2) \Pr(|E(G_1)| = m).$$

Now if the calculations are made explicit in Lemma 8.1 then we can prove that, say,

$$\Pr(\text{COLOUR fails on } G_1) \leq e^{-\sqrt{n}}$$

and it is easy to see that

$$\Pr(|E(G_1)| = m) = \Omega(1/n) \tag{8.1}$$

for p, m close enough to $\frac{1}{2}, \frac{n^2}{6}$ respectively. Hence, with these caveats, one sees immediately that

$$\Pr(\text{COLOUR fails on } G_2) = o(1).$$

Similarly

(8.2) $\Pr(G_2$ is not uniquely 3-colourable) $= o(1)$.

Now some rather tedious calculations show that almost all graphs in $\mathcal{G}(n: \chi = 3)$ have $\approx \frac{n^2}{6}$ edges and have 3 colour classes of size $\approx \frac{n}{3}$ only, (the approximations here are good enough for (8.2) to hold). Thus we really only have to show that Lemma 8.1 can be translated to G_3 chosen uniformly from $\mathcal{G}'$ = the set of 3-colourable graphs with $m \approx \frac{n^2}{6}$ edges and a set of colour classes of size $n_1, n_2, n_3 \approx \frac{n}{3}$.

Now while G_3 is chosen uniformly from $\mathcal{G}'$, G_2 is chosen from $\mathcal{G}'$ with probability proportional to the number, $\nu(G_2)$ of different 3-colourings G_2 (with colour-classes of the appropriate size).

Now for any $\mathcal{A} \subseteq \mathcal{G}'$

$$\Pr(G_2 \in \mathcal{A}) = \sum_{G \in \mathcal{A}} \frac{\nu(G)}{\nu(\mathcal{G}')} \geq \frac{|\mathcal{A}|}{|\mathcal{G}'|} \cdot \frac{|\mathcal{G}'|}{\nu(\mathcal{G}')}$$

$$\geq (1 - o(1)) \quad \Pr(G_3 \in \mathcal{A})$$

since the result of Lemma 8.2 can be expressed as

$$\sum_{\nu(G) \geq 2} \frac{\nu(G)}{\nu(\mathcal{G}')} = o(1).$$

Thus

$$\Pr(G_3 \in \mathcal{A}) \leq (1 + o(1)) \quad \Pr(G_2 \in \mathcal{A}).$$

We obtain the result we want by taking $\mathcal{A} = \{G \in \mathcal{G}' : \text{COLOUR fails on } G\}$.

The failure probability of COLOUR is not quite small enough so that one has a polynomial expected time algorithm if one handles exceptional cases by enumeration. In [DF] we construct another algorithm COLOUR 2 to handle the exceptional cases of COLOUR. It has polynomial running time and failure probability $O(e^{-\Omega(n \log n)})$. Thus if both COLOUR and COLOUR1 fail we can then resort to enumeration of all possible 3-colourings and we will have a polynomial expected time algorithm for colouring, 3-colourable graphs.

9. GRAPH ISOMORPHISM

In this section we give the earliest and simplest result concerning the graph isomorphism problem for random graphs. It is due to Babai, Erdös and Selkow [BES]. Suppose we are given graphs $G_i = (V_n, E_i)$, $i = 1,2$ where G_1 is the random graph $G_{n,.5}$ and G_2 is any graph. Can we quickly tell whether or not $G_1 \cong G_2$ i.e. whether there exists a bijection $f: V_n \to V_n$ such that $vw \in E_1$ iff $f(v)f(w) \in E_2$. The answer in [BES] is that **whp** we can check this in $O(n^2)$ time.

The method is based on the fact that **whp** G_1 has the properties (9.1) and (9.2) below. Let the vertices of G_1 be relabelled so that $d(i) \geq d(i+1)$, $i = 1,2,\ldots,n-1$. Let $r = \lceil 3 \log_2 n \rceil$. Then **whp**

$$d(i) > d(i+1) \quad \text{for } 1 \leq i < r. \tag{9.1}$$

Next, for $i > r$ let $X_j = \{i: 1 \leq i < r \text{ and } ij \in E_1\}$. then **whp**

$$X_j \neq X_k \quad \text{for } j,k > r. \tag{9.2}$$

Thus we can relabel the vertices $r+1,\ldots,n$ so that

(9.3) X_i is lexicographically larger than X_{j+1}, $i = r+1,\ldots,n-1$.

Given (9.1) and (9.2) it is easy to check if G_1 and G_2 are isomorphic.

1. Compute the degree sequence of G_2. If the largest r degrees do not coincide with those of G_1 then G_1 and G_2 are not isomorphic. If they are then by (9.1) we can identify $f(1),\ldots,f(r)$ in any possible isomorphism. By relabelling vertices of G_2 we can assume $f(i) = i$ for $1 \le i \le r$.
2. For each vertex $v > r$ of G_2 compute $Y_v = \{i: 1 \le i \le r$ and $iv \in E(G_p)\}$. Sort these $n-r$ sets into lexicographic order. If there exists $i > r$ such that $Y_i \neq X_i$ then by (9.2) and (9.3) Gj and $G^{\wedge}$ are not isomorphic. Otherwise the only possible isomorphism is now $f(i) = i$.
3. Finally, check if $f(i) = i$ is an isomorphism i.e. check if now Gj = G_2.

The proof of (9.1) requires a lot of calculation. Babai, Erdbs and Selkow proved considerably more than this. They showed that

(9.4) $$d(i) - d(i+1) \ge n^{.03} \quad \text{for} \quad 1 \le i < n^{.15}.$$

For an even stronger result see Theorem III. 15 of Bollobás [Bol]. Given (9.4) it is quite easy to prove (9.2). If (9.4) holds and $X_i = X_j$ for some $i,j > r$ then i and j have the same set of neighbours among the r

largest vertices in $H_{ij} = G_1 - \{i,j\}$. Denote the this event by $\mathcal{E}_{ij}$. Now since the graph H_{ij} is independent of i,j the probability of $\mathcal{E}_{ij}$ is $(\frac{1}{2})^r$. Hence

$$\Pr(\exists\ i,j > r : X_i = X_j) \leq \sum_{i=1}^{n-1} \sum_{j=1}^{n} \Pr(\mathcal{E}_{ij}) + \Pr(\text{(9.4) fails})$$

$$\leq n^2 \cdot (\tfrac{1}{2})^r + o(1)$$

$$= o(1).$$

The result of [BES] has been strengthened by Karp [Ka2], Lipton [Li] and Babai and Kucera [BK]. In particular Babai and Kucera handle exceptional graphs in such a way that graph isomorphism can be tested in *linear expected time* on $G_{n,.5}$. For regular graphs, the above algorithm(s) would be particularly ineffective. However, Kucera [Ku] has recently devised an algorithm for regular graphs which runs in linear expected time, i.e. O(nd), assuming the degree d does not grow with n.

10. GRAPH BISECTION

Here we are given a graph $G = (V,E)$ with n vertices, n even, and the problem is to find the partition of V into two equal sized subsets S_1, S_2 so that the number of $S_1 : S_2$ edges is minimised. The minimum such number of edges is called the *bisection width* of G. The problem is useful in VLSI design problems (see Bhatt and Leighton [BL]), but is NP-hard (Garey, Johnson and Stockmeyer [GJS]).

If we take the graph $G_{n,m}$ as a model of random input then we find that

all relevant cuts have $\approx \frac{m}{2}$ edges whp provided m is sufficiently large e.g. $m = \Omega(n \log n)$. Finding the bisection width in these circumstances is still open.

Positive results can be obtained if we consider sampling uniformly from $\mathcal{G}(n,m,b)$, the set of graphs with vertex set $V_{n,m}$ edges and bisection width b. Basically, the idea is to have b significantly smaller than $\frac{m}{2}$ and then whp there will be a unique cut of size b which will be easy to find. Bui, Chaudhuri, Leighton and Sipser [BCLS] considered this approach for regular graphs, Dyer and Frieze [DF] considered $\mathcal{G}(n,m,b)$ with $m = \Omega(n^2)$ and Boppana [Bp] considered the case $m = \Omega(n \log n)$. We will outline Boppana's approach here.

First of all we remark that it is not easy to work directly with $\mathcal{G}(n,m,b)$. Instead one chooses a random partition of V_n into S_1, S_2 of equal size, and then add edges between S_1 and S_2 of equal size, and then adds edges between S_1 and S_2 with probability $q = 4b/n^2$ and within each S_i with probability $p = 4(m-b)/\left[\begin{matrix}\frac{1}{2}n\\ 2\end{matrix}\right]$. Results are proved for this "independent" model and then translated to $\mathcal{G}(n,m,b)$ - see §8 on colouring.

For $S \subseteq V_n$ we define $x = x(S) \in \mathbb{R}^n$ by $x_i = 1$, $i \in S$ and $= -1$ otherwise. Given $d \in \mathbb{R}^n$ we let $B = B(d) = A + D$ where A is the adjacency matrix of G and $D = \mathrm{diag}(d)$. Also let $\mathrm{sum}(B) = 2|E| + \sum_{i=1}^{n} d_i =$ the sum of the entries of B. Next let

$$f(G,D,x) = \sum_{(i,j)\in E} \frac{1 - x_i x_j}{2} - \frac{1}{4} \sum_{i=1}^{n} d_i (x_1^2 - 1)$$

$$= \frac{1}{4} (\mathrm{sum}(B) - x^T B x).$$

The significance of this function is that

(10.1) $$f(G,d,x(S)) = |(S: V_n - S)| \quad \text{for} \quad S \subseteq V_n.$$

Observe that $\|x(S)\| = \sqrt{n}$ (Euclidean norm) and $e^T x(S) = 0$ when $|S| = \frac{1}{2} n$. ($e^T = (1,1,\ldots,1)$). So from (10.1) it is natural to consider

$$g(G,d) = \min_{\substack{e^T x = 0 \\ \|x\| = \sqrt{n}}} f(G,d,x)$$

$$= \min_{\|x\| = \sqrt{n}} \frac{1}{4} (\text{sum}(B) - x^T \hat{B} x)$$

where $\hat{B} = \hat{B}(d) = (I - \frac{1}{n} ee^T)B$. (Observe that the matrix $I - \frac{1}{n} ee^T$ projects $\mathbb{R}^n$ onto $\{x \in \mathbb{R}^n: e^T x = 0\}$.

Bopanna's idea is that one can find d for which the x minimising $f(G,d,x)$ is $x(S)$ for a minimum bisection S.

Note that

(10,2) $$g(G,d) = \frac{1}{4}(\text{sum}(B) - n\lambda(\hat{B}))$$

where $\lambda(\hat{B})$ is the largest eigenvalue of $\hat{B}$.

Now $g(G,d)$ being the infimum of a collection of linear functions is concave in d and so

$$h(G) = \max_{d \in \mathbb{R}^n} g(G,d)$$

can be computed in polynomial time. (Grotschel, Lovasz and Schrijver (GLS)).

Since $g(G,d) \leq f(G,d,x(S))$ for $S \subseteq V_n$ we see that

$$h(G) \leq \text{bisection width of } G.$$

The nice probabilistic result of Boppana is that if G is sampled uniformly from $\mathcal{G}(n,m,b)$ and

$$0 \leq b \leq \frac{1}{2}m - 5\sqrt{mn \log n} \tag{10.3}$$

then **whp** the bisection width of G is $b = h(G) = g(G,d^*)$ and the eigenvector corresponding to $\lambda(\hat{B}(d^*))$ yields the minimum bisection. The proof of this result is as follows. First of all it is straightforward to show given (10.3), that in the independent model there is **whp** a unique minimum bisection of size b. Next let S_1,S_2 be a minimum bisection. For $i \in S_1$ let $d_i^* = d(i,S_2) - d(i,S_1)$ and for $i \in S_2$ let $d_i^* = d(i,S_1) - d(i,S_2)$ where $d(v,S) = |\{w \in S: vw \in E\}|$. Now $\text{sum}(B(d^*)) = 4b$ and so by (10.2) $g(G,d^*) = b$ iff $\lambda(\hat{B}(d^*)) = 0$. Observe also that $\hat{B}(d^*)\, x(S_1) = 0$ and so Boppana's result is reduced to showing that **whp** $\hat{B}(d^*)$ has a unique eigenvalue of zero and every other eigenvalue is negative.

Now we have $E(B) = M - \frac{1}{2}(p-q)nI$ where $M = E(A)$. Also $\hat{M}x(S_1) = 0$, $x(S_1)^T\hat{M} = 0$ and so if $\xi^T(\hat{B}-\hat{M})\xi \leq 0$ always then $\xi^T\hat{B}\xi \leq 0$ always, which is what we need. This follows *a fortiori* if $B-M$ has non-positive eigenvalues or equivalently if $B-E(B)$ has eigenvalues bounded above by $\frac{1}{2}(p-q)n$. Now

$$\lambda(B - E(B)) \leq \lambda(A - E(A)) + \lambda(D - E(D)).$$

The eigenvalues of $D - E(D)$ are precisely its diagonal entries and using Chernoff's bound we find that $\lambda(D - E(D)) \leq 5\sqrt{pn \log n}$.

Extimating $\lambda(A - E(A))$ is more difficult, but the eigenvalues of random matrices have been intensively studied. By modifying a result due to Furédi and Komlós [FuK] Boppana shows that $\lambda(A - E(A)) \leq 3\sqrt{pn}$ **whp** and so $\lambda(B - E(B)) \leq 6\sqrt{pn \log n}$ **whp**. The reader can now check that if (10.3) holds then $6\sqrt{pn \log n} < \frac{1}{2}(p-q)n$ as required.

The probability that Boppana's algorithm fails to work is not sufficiently small that exceptional cases can be dealt with more crudely and still yield a polynomial expected time algorithm which handles all graphs. For $m = \Omega(n^2)$ however, there is a polynomial expected time algorithm, [DF].

11. OTHER ASPECTS

We have concentrated here on positive results that arise in probabisistic analysis. This field also has its share of negative results. We mention three: Chvatal [C] showed that a certain class of approaches to finding the largest independent set in a graph took exponential time **whp**; McDiarmid [M] proved a similar result for graph colouring as did Ahn, Cooper, Cornuéjols and Frieze [ACCF] for finding a small dominating set.

There is an increasing interest in finding fast parallel algorithms. There are a few results here of interest to us: Frieze and Rudolph [FR] gave an $O(\log\log n)$ expected time parallel algorithm for the shortest path problem of §3; Frieze [F4] gave an $O((\log\log n)^2)$ expected time parallel algorithm for the Hamilton cycle problem in $G_{n,p}$, p constant; Frieze and Kucera [FrK] give a polylog expected time algorithm for colouring graphs; Coppersmith, Raghavan and Tompa [CRT] give polylog expected time algorithms for graph colouring, finding maximal independent sets and finding Hamilton cycles; Calkin and

Frieze [CF] deals with maximal independent sets.

Finally we mention an area of particular interest to probabilists. Given a weighted optimisation problem, determine the properties of the (random) optimal value. We first mention two similar results: consider the nxn assignment problem in which the costs $c(i, j)$ are independent uniform [0,1] random variables. Let W_n denote the minimal value of an assignment. Walkup [W] showed that, rather surprisingly, $E(W_n) < 3$ for all n. Karp [K] improved this to $E(W_n) < 2$ (see also Dyer, Frieze and McDiarmid [DFM]). A lower bound of $1 + e^{-1}$ was proved by Lazarus [Lz].

Consider next the minimum spanning tree problem where the edge weights are also independent uniform [0,1] random variables. Let L_n denote the minimum length of a spanning tree. We showed [F5] that

$$\lim_{n\to\infty} E(a_n) = f(3) = \sum_{k=1}^{\infty} k^{-3}.$$

(see also Steele [Stl] and Frieze and McDiarmid [FM]).

Probabilists have found Euclidean problems even more interesting. For example, suppose $X_n, X_0, \ldots, X_n$ are independetntly chosen uniformly at random in the unit square $[0,1]^2$. In a very important paper Beardwood, Halton and Hammersley [BHH] showed that if T_n is the minimum length of a travelling salesman tour through these points then there exists a constant $j5 > 0$ such that

$$\Pr\left(\lim_{n\to\infty} \frac{T_n}{\sqrt{n}} = j3\right) = 1.$$

Steele [St2] has generalised this result considerably and the paper by

Karp [Ka4] was very influential in generating interest in the probabilistic analysis of algorithms.

For a bibliography on topics related to this paper see Karp, Lenstra, McDiarmid and Rinnooy Kan [KLMR].

REFERENCES

[ACCF] S.Ahn, C.Cooper, G.Cornuejols and A.M.Frieze, 'Probabilistic analysis of a relaxation for the k-median problem', *Mathematics of Operations Research* **13** (1988) 1-31.

[AHU] A.Aho, J.E.Hopcroft and J.D.Ullman, '*The design and analysis of computer algorithms*', Addison-Wesley, Reading MA, 1974.

[BES] L.Babai, P.Erdos and S.M.Selkow, 'Random graph isomorphisms', *SIAM Journal on Computing* **9** (1980) 628-635.

[BHH] J.Beardwood, J.H.halton and J.M.Hammersley, 'The shortest path through many points', *Proceedings of the Cambridge Philosophical Society* **55** (1959) 299-327.

[Bo1] B.Bollobas, *Random graphs*, Academic Press, 1985.

[Bo2] B.Bollobas, 'The chromatic number of random graphs', *Combinatorica* **8** (1988) 49-55.

[BE] B.Bollobas and P.Erdos, 'Cliques in random graphs', *Mathematical Proceedings of the Cambridge Philisophical Society* **80** (1976) 419-427.

[BFF] B.Bollobas, T.I.Fenner and A.M.Frieze, 'An algorithm for finding Hamilton paths and cycles in random graphs', *Combinatorica* **7** (1987) 327-342.

[Bp] R.Boppana, 'Eigenvalues and graph bisection: an average case analysis', *Proceedings of the 28th Annual IEEE Symposium on the Foundations of Computer Science* (1987) 280-285.

[BK] L.Babai and L.Kucera, 'Canonical labelling of graphs in linear average time', *Proceedings of the 20th Annual IEEE Symposium on the Foundations of Computer Science* (1979) 39-46.

[BL] S.Bhatt and T.Leighton, 'A framework for solving VLSI graph problems', *Journal of Computer and System Sciences* **28** (1984) 300-343.

[BCLS] T.Bui, S.Chaudhuri, T,Leighton and M.Sipser, 'Graph bisection algorithms with good average case behaviour', *Combinatorica* **6**.

[CF] N.Calkin and A.M.Frieze, 'Probabilistic analysis of a parallel algorithm for finding a maximal independent set', to appear.

[C] V.Chvatal, 'Determining the stability number of a graph', *SIAM Journal on Computing* **6** (1977) 643-662.

[CT] D.Cheriton and R.E.Tarjan, 'Finding minimum spanning trees', *SIAM Journal on Computing* **5** (1976) 724-742.

[CRT] D.Coppersmith, P.Raghavan and M.Tompa, 'Parallel graph algorithms that are efficient on average', *Proceedings of the 28th Annual IEEE Symposium on the Foundations of Computer Science* (1987) 260-270.

[D] G.B.Dantzig, 'On the shortest route through a network', *Management Science* **6** (1960) 187-190.

[DF] M.E.Dyer and A.M.Frieze, 'The solution of some random NP-hard problems in polynomial expected time' to appear in *Journal of Algorithms*.

[DFM] M.E.Dyer, A.M.Frieze and C.J.H.McDiarmid, 'On linear programs with random costs', *Mathematical Programming* **35** (1986) 3-16.

[EK] J.Edmonds and R.M.Karp, 'Theoretical improvements in algorithmic efficiency for network flow problems', *Journal of the ACM* **19** (1972) 248-264.

[ER1] P.Erdos and A.Renyi, 'On random graphs 1', *Publ. Math. Debrecen* **6** (1959) 290-297.

[ER2] P.Erdos and A.Renyi, 'The evolution of a random graphs', *Publ. Math. Inst. Hungar. Acad. Sci.* **5** (1960) 17-61.

[ER3] P.Erdos and A.Renyi, 'On the strength of connectedness of a random graph', *Acta Math. Acad. Sci. Hungar.* **12** (1961) 261-267.

[ER4] P.Erdos and A.Renyi, 'On the existence of a factor of degree one of a connected random graph', *Acta Math. Inst. Acad. Sci. Hungar.* **17** (1966) 359-368.

[FF] T.I.Fenner and A.M.Frieze, 'On the existence of Hamilton cycles in a class of random graphs', *Discrete Mathematics* **45** (1983) 301-305.

[Fd] M.L.Fredman, 'On the decision tree complexity of the shortest path problem', *Proceedings of the 16th Annual IEEE Symposium on the Foundations of Computer Science* ((1985) 101-105.

[FT] M.L.Fredman and R.E.Tarjan, 'Fibonnacci heaps and their uses in network optimisation problems', *Proceedings of the 25th Annual IEEE Symposium on the Foundations of Computer Science* (1984) 338-346.

[F1] A.M.Frieze, 'Finding Hamilton cycles in sparse random graphs', *Journal of Combinatorial Theory* **B 44** (1988) 230-250.

[F2] A.M.Frieze, 'An algorithm for finding Hamilton cycles in random digraphs', *Journal of Algorithms* **9** (1988) 181-204.

[F3] A.M.Frieze, 'On the exact solution of random travelling salesman problems with medium sized integer costs', *SIAM Journal on Computing* **16** (1987) 1052-1072.

[F4] A.M.Frieze, 'Parallel algorithms for finding Hamilton cycles in random graphs', *Information Processing Letters* **25** (1987) 111-117.

[F5] A.M.Frieze, 'On the value of a random minimum spanning tree problem', *Discrete Applied Mathematics* **10** (1985) 47-56.

[FG] A.M.Frieze and G.R.Grimmett, 'The shortest path problem for graphs with random arc-lengths', *Discrete Applied Mathematics* **10** (1985) 57-77.

[FrK] A.M.Frieze and L.Kucera, 'Parallel colouring of random graphs', to appear in *Annals of Discrete Mathematics*.

[FM] A.M.Frieze and C.J.H.McDiarmid, 'On random minimum length spanning trees', to appear in *Combinatorica*.

[FR] A.M.Frieze and L.Rudolph, 'A parallel algorithm for all-pairs shortest paths in a random graph', *Proceedings of the 22nd Allerton Conference on Communication, Control and Computing* (1985) 663-670.

[FuK] Z.Furedi and M.Komlos, 'The eigenvalues of random symmetric matrices' *Combinatorica* **1** (1981) 233-241.

[GGS] H.N.Gabow, Z.Galil and T.Spencer, 'Efficient implementation of graph algorithms using contraction', *Proceedings of the 25th Annual IEEE Symposium on the Foundations of Computer Science* (1984) 347-357.

[GJS] M.R.Garey, D.S.Johnson and L.Stockmeyer, 'Some simplified NP-complete graph problems', *Theoretical Computer Science* **1** (1976) 237-267.

[GM] G.R.Grimmett and C.J.H.McDiarmid, 'On colouring random graphs', *Mathematical Proceedings of the Cambridge Philosophical Society* **77** (1975) 313-324.

[Ka1] R.M.Karp, 'An algorithm to solve the m x n assignment problem in expected time O(mnlogn)', *Networks* **10** (1980) 143-152.

[Ka2] R.M.Karp, 'Probabilistic analysis of a canonical numbering algorithm for graphs', *Proceedings of Symposia in Pure Mathematics*, Volume 34, 1979, American Mathematical Society, Providence, RI, 365-378.

[Ka3] R.M.Karp, 'An upper bound on the expected cost of an optimal assignment', *Discrete Algorithms and Complexity: Proceedings of the Japan-US Joint Seminar* (D.Johnson et al, eds.), 1-4, Academic Press, New York, 1987.

[Ka4] R.M.Karp, 'Probabilistic analysis of partitioning algorithms for the traveling salesman problem in the plane', *Mathematics of Operations Research* **2** (1977) 209-224.

[KLMR] R.M. Karp, J.K.Lenstra, C.J.H.McDiarmid and A.H.G.Rinnooy Kan', Probabilistic analysis of combinatorial algorithms: an annotated bibliography', in *Comminatorial Optimisation: Annotated Bibliographies*, (M.O'Heigeartaigh, J.K.Lenstra and A.H.G.Rinnooy Kan, eds.), John Wiley and sons, New York, 1984.

[KSp] R.M.Karp and M.Sipser, 'Maximum matchings in sparse random graphs', *Proceedings of the 22nd Annual IEEE Symposium on Foundations of Computer Science* (1981) 364-375.

[KT] R.M.Karp and R.E.Tarjan, 'Linear expected time algorithms for connectivity problems', *Journal of Algorithms* **1** (1980) 374-393.

[KSz] M.Komlos and E.Szemeredi, 'Limit distributions for the existence of Hamilton cycles in a random graph', *Discrete Mathematics* **43** (1983) 55-63.

[Ku] L.Kucera, 'Canonical labeling of regular graphs in linear expected time' *Proceedings of the 28th Annual IEEE Symposium on the Foundations of Computer Science* (1987) 271-279.

[Kz] T.G.Kurtz, 'Solutions of ordinary differential equations as limits of pure jump Markov processes', *Journal of Applied Probability* **7** (1970) 49-58.

[La] E.L.Lawler, *Combinatorial optimization: networks and matroids*, Holt, Rinehart and Winston, New York 1976.

[Lz] A.J. Lazarus, 'The assignment problem with uniform (0,1) cost matrix', B.A. Thesis, Department of Mathematics, Princeton University, Princton, N.J.

[Li] R.J.Lipton, 'The beacon set approach to graph isomorphism', Yale University, pre-print.

[Lu] T.Luczak, 'The chromatic number of random graphs', *Combinatorica* to appear.

[M1] C.J.H.McDiarmid, 'Determining the chromatic number of a graph', *SIAM Journal on Computing* **8** (1979) 1-14.

[M2] C.J.H.McDiarmid, 'On some conditioning results in the analysis of algorithms', *Discrete Applied Mathematics* **10** (1985) 197-201.

[MV] S.Micali and V.V.Vazirani, 'An $O(\sqrt{|V|}.|E|)$ algorithm for finding maximum matching in general graphs', *Proceedings of the 21st Annual IEEE Symposium on Foundations of Computer Science* (1980) 17-27.

[MT1] A.Moffat and T.Takaoka, 'An all pairs shortest path algorithm with expected running time $O(n^2 \log n)$', *Proceedings of the 26th Annual IEEE Symposium on the Foundations of Computer Science* (1985) 101-105.

[MT2] A.Moffat and T.Takaoka, 'An all pairs shortest path algorithm with expected time $O(n^2 \log n)$', *SIAM Journal on Computing* **16** (1987) 1023-1031.

[P] E.M.Palmer, *Graphical evolution*, Wiley-Interscience, 1985.

[PS] C.H.Papadimitriou and K.Steiglitz, *Combinatorial optimization: algorithms and complexity*, Prentice-Hall, 1982.

[SU] E.Shamir and E.Upfal, 'Sequential and distributed graph colouring algorithms with performance analysis in random graph spaces', *Journal of Algorithms* **5** (1984) 488-501.

[Sp] P.M.Spira, 'A new algorithm for finding all shortest paths in a graph of positive arcs in average time O(n log n)' *SIAM Journal on Computing* **2** (1973) 28-32.

[St2] J.M.Steele, 'On Frieze's $\zeta(3)$ limit for lengths of minimal spanning trees', *Discrete Applied Mathematics*.

[St2] J.M.Steele, 'Subadditive Euclidean functionals and non-linear growth in geometric probability', *Annals of Probability* **9** (1981) 365-376.

[ST] D.D.Sleator and R.E.Tarjan, 'Self-adjusting heaps', *SIAM Journal on Computing* **15** (1986) 52-69.

[To] N.Tomizawa, 'On some techniques useful for solution of transportation problems', *Networks* **1** (1972) 173-194.

[Tu] J.S.Turner, 'Almost all k-colorable graphs are easy to color', *Journal of Algorithms* **9** (1988) 63-82.

[V] J.Vuillemin, 'A data structure for manipulating priority queues', *Communications of the ACM* **21** (1978) 309-315.

[W] D.Walkup, 'On the expected value of a random assignment problem', *SIAM Journal on Computing* **8** (1979) 440-442.

Carnegie Mellon University Libraries
3 fi4fi£ D135b Ib55