FAST PROBABILISTIC ALGORITHMS FOR HAMILTONIAN CIRCUITS AND MATCHINGS*

Dana Angluin and Leslie G. Valiant
Department of Computer Science
University of Edinburgh
Edinburgh, Scotland.

1. Introduction

The main purpose of this paper is to give techniques for analysing the probabilistic performance of certain kinds of algorithms, and hence to suggest some fast algorithms with provably desirable probabilistic behaviour. The particular problems we consider are: finding Hamiltonian circuits in directed graphs (DHC), finding Hamiltonian circuits in undirected graphs (UHC), and finding perfect matchings in undirected graphs (PM). We show that for each problem there is an algorithm that is extremely fast ($O(n(\log n)^c)$ for DHC and UHC, and $O(n \log n)$ for PM), and which with probability tending to one finds a solution in randomly chosen graphs of sufficient density. These results contrast with the known NP-completeness of the first two problems [2,12] and the best worst-case upper bound known of $O(n^{2.5})$ for the last [9].

As in [8], we consider two different distributions for random graphs: (i) $G_{n,p}$: graphs of n nodes where each edge is present with probability p, independent of other edges, and (ii) $G_{n,N}$: graphs of n nodes and exactly N edges, each graph with equal probability. By a simple argument we show that if in a certain sense an algorithm works for one model then it also works for the other. Hence it suffices to prove our main results for just one of them.

The form of our results is the following: if $N \geq cn \log n$ then given a graph from $G_{n,N}$ the algorithm will find a solution with probability $1 - O(n^{-\alpha})$. Furthermore α can be arbitrarily increased at the expense of increasing c and the constant multiplicative factor of the run-time. It is known, however, that if $N < (\frac{1}{2}-\varepsilon)n \log n$ for any $\varepsilon > 0$, then $G_{n,N}$ will have isolated points with probability tending to one [6]. Since graphs with isolated points cannot have Hamiltonian circuits or perfect matchings, it follows that the edge density N required for our results cannot be improved by more than a constant factor.

We are concerned here primarily with the computational problem of efficiently finding instances of solutions to the given problems. This type of question appears in general more difficult than the problem of merely determining the existence of a solution. For example for the clique problem in $G_{n,p}$ with $p = \frac{1}{2}$, it is known that a clique of size $(2-\varepsilon)\log_2 n$ almost surely exists, but no polynomial time algorithm is known for almost surely finding one [11]. In a non-probabilistic context this dichotomy between finding solutions and determining their existence is present in extreme forms in problems where the existence of a solution is certain and hence requires no computation. In some cases, of course, the proof of certainty does yield a fast algorithm (e.g. Dirac's proof [4] of the existence of an UHC in graphs where all nodes have degree at least $n/2$ implies an $O(n^2)$ algorithm for finding one.) In others, however, this is not the case (e.g. the problem of finding the prime factors of an n-digit integer.) Note, however, that for such NP-complete problems as DHC and UHC it is known that the worst-case complexities of detecting the existence of a solution and of finding one are polynomially related.

The proof of almost sure existence in $G_{n,N}$ (for N about $cn \log n$) was shown by Erdös and Rényi [7] in the case of PM and by Pósa [17] in the case of UHC. (In these proofs there is considerable emphasis on determining the constant c and even lower order terms [14]. We do not follow this here.) Note that our result for DHC does, of course, imply almost sure existence of a DHC in directed graphs of this density. This appears to be new, and does not follow from the result for UHC. We give an easy reduction in the other direction (i.e. the existence results for PM, UHC, and DHC are successively more powerful.)

The stimulus for the present work was the observation of Karp [13] that Pósa's argument [17] can be adapted to obtain a polynomial time algorithm for finding UHCs almost surely. As in other existential proofs, Posa relies on global properties of the graph that need to be proved only for the initial undisturbed graph under the simple assumptions about probabilities and independence that are allowed by the input distribution. The techniques we develop here are more appropriate for algorithmic analysis in that they can keep track of the complex ways in which the progress of the algorithm can condition the examined graph.

Perhaps our main result is our $O(n(\log n)^2)$ algorithm for finding DHCs almost surely in random graphs of sufficient density. By an easy reduction an algorithm for UHC of the same complexity follows. As a separate result we give a different algorithm for UHC that has a similar complexity. [N.B. The latter algorithm involves a variant of the Pósa-Karp construction. However, it does not appear

* Supported by a research grant from the Science Research Council. Part of this work was done at the University of Leeds.

that their techniques could be used to prove a time bound approaching ours.] For the PM problem we can improve the above bounds by a logarithmic factor. This result also has, as a consequence, an $O(n \log n)$ expected time algorithm for determining (with certainty) the existence of PMs in such graphs.

The reader will note that the algorithms as described in §3 are all randomized (i.e. make random decisions [18]). As we show in §9, however, they can be translated into deterministic algorithms that simulate randomness by abstracting the necessary random bits from unused parts of the random input graph. This translation appears primarily of theoretical interest since for practical use the randomized versions are preferable.

We wish to express the run-times of our algorithms so as to reflect their performance on random access computers of the kind in conventional present-day use. Because of the apparent absence in the literature of a model suitable for our purpose we have chosen to define a new class of models. We believe that this may be a convenient one on which to standardize in general for expressing results in low-level complexity.

In conclusion we note the desirability and regrettable absence of available results in the probabilistic analysis of combinatorial problems, that relate not only to specific problems but have significant wider application. It is therefore worth mentioning that a technique introduced by Levin [16,19] can be used to show that for a wide class of computational problems of the kind we consider here, and for most machine models, one can construct for each time complexity class an algorithm that has at least as good probabilistic behaviour as any algorithm in a slightly smaller complexity class.

2. Preliminaries

A graph G is a pair (V,E) where V is the set of nodes and E the set of edges. We always assume that there are n nodes, labelled by integers $1,2,\ldots,n$. G is a directed graph if $E \subset \{(v,w) \mid v \neq w;\ v,w \in V\}$. It is an undirected graph if $E \subset \{\{v,w\} \mid v \neq w;\ v,w \in V\}$. In either case we denote the cardinality of E by N.

A Hamiltonian path (HP) from u to v in a directed graph G is a directed path from u to v that visits every node of G exactly once. In the case $u = v$ such a path is a Hamiltonian circuit (HC). Hamiltonian paths and circuits are defined analogously in the undirected case. A perfect matching (PM) in an undirected graph with $n = 2k$ nodes is a set of k edges $e_1, e_2, \ldots, e_k$ such that $e_1 \cup e_2 \cup \ldots \cup e_k = V$.

Following [8] we introduce random variables $G_{n,p}$ and $G_{n,N}$ ($D_{n,p}$ and $D_{n,N}$) that range over undirected (directed) graphs with n nodes. $G_{n,p}$ is defined by the properties that (i) $\Pr(\{u,v\} \in G_{n,p}) = p$ and ii) these probabilities for each of the $\binom{n}{2}$ possible edges are mutually independent. $G_{n,N}$ is defined (i) to take values that have exactly N edges such that (ii) all graphs with n nodes and N edges have the same probability. The definitions for $D_{n,p}$ and $D_{n,N}$ are identical except that here there are $n(n-1)$ possible edges.

For brevity we will frequently denote a sequence "$x_1, x_2, \ldots, x_i$" by "x^i" schematically. Unless otherwise stated all logarithms are to the base e.

We shall use the following inequalities:

Fact 1 For all real x, $1+x \leq \exp(x)$.

Fact 2 For all n,k $(1 \leq k \leq n)$,

$$\binom{n}{k} \leq \left(\frac{ne}{k}\right)^k$$

Fact 3 $\exists K > 0\ \forall n,p$ with $n > 0$, $1 > p > 0$, and np is an integer,

$$\binom{n}{np} p^{np}(1-p)^{n-np} \geq Kn^{-\frac{1}{2}}$$

Fact 4 $\forall n,p,\beta$ with $0 \leq p \leq 1$ and $0 \leq \beta < 1$

$$\text{(i)} \quad \sum_{k=0}^{(1-\beta)np} \binom{n}{k} p^k (1-p)^{n-k} < \exp(-\beta^2 np/2)$$

$$\text{(ii)} \quad \sum_{k=(1+\beta)np}^{n} \binom{n}{k} p^k (1-p)^{n-k} < \exp(-\beta^2 np/3)$$

Facts 2 and 3 follow from Stirling's approximation for $n!$, and Fact 4 from Chernoff's bound ([8] pp. 17-18).

We call our new model of computation a random access computer (RAC). The memory of a RAC consists of a set of words. In any computation the words are all of the same finite size. RAC programs resemble programs for RAMs [3] except that they may involve a much richer set of instructions. Each instruction is executed on a whole word and takes unit time.

The underlying idea of the model is to make all of the space used in any computation addressable from one word; hence the word size has to be at least the logarithm of the space used. It follows that the word size of the machine depends on the size of the input under consideration. (Note that this has the advantage of not introducing the word size as a further parameter of the model.) We could follow the above idea through and make the word size dependent on the algorithm being executed (i.e. by relating it directly to the space used.) However, we can avoid this complication and retain the main idea, for polynomial space computations, by the following:

Definition The time complexity of an algorithm is $\leq f(n)$ if for some constant c and for all $n > 0$ the algorithm takes $\leq f(n)$ steps on a RAC with word size $c \log n$ for all n-word inputs.

Any set I of instructions defines a particular model, which is denoted by RAC_I. For the purposes of this paper we are not concerned with minimizing I and shall therefore assume informally that I contains all the standard instructions with appropriate conventions e.g. $+$, $-$, $*$, $\div$, shifting, bit-wise Boolean operations, indirect addressing, jumping on zero. We assume that the input is already present in memory.

Given a particular I we distinguish between a deterministic RAC and a randomized RAC (r-RAC). The latter is distinguished from the former by having a random element, i.e. there is an extra

instruction to generate a random word (of c log n bits.)

3. Algorithms and Main Results

In this section we give a high-level description of our three principal algorithms and state the main results about their probabilistic performance on $D_{n,N}$ or $G_{n,N}$. The algorithms all work equally well on $D_{n,p}$ or $G_{n,p}$. We show this by giving some general conditions under which the two models can be proved equivalent (Propositions 1,2,2'). The algorithms described are randomized. In §9 we describe how to make them deterministic, and in §10 we analyse the run-times of implementations of them on r-RACs and RACs.

The algorithms all use the following procedure for selecting an edge randomly from a given node u and deleting it.

```
procedure SELECT(u)
  begin if ∀ v ε V (u,v) ∉ G then return "*"
        else select at random with equal
             probabilities one of the edges
             (u,v) ε G , delete (u,v) from
             G, and return the value v;
  end
```

(N.B. For the case of undirected graphs we replace (u,v) by {u,v} in the above.)

DHC Algorithm

As input the algorithm takes a directed graph G with n nodes, together with two specified nodes of G, s and t (which may be the same). The algorithm attempts to find a directed HP from s to t. If it succeeds, it returns "success". Otherwise it returns "failure". During execution it maintains a partial path P which consists either of a (simple) directed path, or of the disjoint union of a (simple) directed path and a (simple) directed cycle. In either case the terminal endpoint of the simple path is kept in the variable ndp (for next departure point) and the algorithm attempts to extend P by calling SELECT to explore edges at random out of the node ndp.

```
procedure DHC(G,s,t)
begin
1. ndp ← s, P ← ∅ .
2. (comment: P consists of a directed path
   from s to ndp.)
   (a) If P includes every node of G
       (except t, in the case s ≠ t) and
       if we previously explored and deleted
       (ndp,t) from G then add (ndp,t) to P
       and return "success".
   (b) v ← SELECT(ndp)
      (i) If v = "*" then return "failure".
     (ii) If v ≠ t and v is not in P then
          add (ndp,v) to P, ndp ← v, and go
          to 2.
    (iii) If v ≠ s and v is in P and there
          are at least n/2 nodes in P between
          v and ndp (inclusive) then
          begin
            u ← predecessor of v in P,
            delete (u,v) from P,
            add (ndp,v) to P,
            ndp ← u,
            go to 3.
          end
     (iv) Otherwise (i.e. if v = s or v = t
          or cycle is too small) go to 2.
3. (comment: P consists of a directed path
   from s to ndp and a disjoint directed
   cycle of at least n/2 nodes.)
   (a) v ← SELECT(ndp)
      (i) If v = "*" then return "failure".
     (ii) If v ≠ t and v is not in P then
          begin add (ndp,v) to P, ndp ← v,
          and go to 3 end.
    (iii) If v is in the cycle part of P
          then
          begin
            u ← predecessor of v in P,
            delete (u,v) from P,
            add (ndp,v) to P,
            ndp ← u,
            go to 2.
          end
     (iv) Otherwise go to 3
end
```

DHC Theorem $\forall\alpha\ \exists M\ \exists c\ \forall N \geq cn\log n$ $\forall s,t\ (1 \leq s,t \leq n)$ the probability that $DHC(D_{n,N},s,t)$ returns "success" before SELECT has been called $Mn\log n$ times is at least $1 - O(n^{-\alpha})$.

The theorem is proved in §4 and §5. In §9 and §10 an $O(n\log^2 n)$ time bound is deduced for an implementation on a RAC.

PM Algorithm

The algorithm takes as input an undirected graph G with n nodes, where n is an even integer. It attempts to find a perfect matching in G. It returns "success" if it succeeds, and "failure" otherwise. It maintains a partial matching P (i.e. a perfect matching on a subset of the nodes) and tries randomly to augment it by exploring new edges until it either fails for lack of edges, or succeeds in finding a PM.

```
procedure PM(G)
begin
1. P ← ∅.
2. (a) If P includes all the nodes of G then
       return "success".
   (b) Otherwise ndp ← least-numbered node not
       in P, and go to 3.
3. v ← SELECT(ndp)
   (a) If v = "*" then return "failure".
   (b) If v is not in P then add {ndp,v}
       to P and go to 2.
   (c) If v is in P then
```

```
        begin
          u ← unique node such that {u,v} ε P,
          delete {u,v} from P,
          add {ndp,v} to P,
          ndp ← u,
          go to 3.
        end
end
```

PM Theorem $\forall\alpha\ \exists M\ \exists c\ \forall N \geq cn\log n$ the probability that $PM(G_{n,N})$ returns "success" before SELECT has been called $Mn\log n$ times is at least $1 - O(n^{-\alpha})$.
The proof of the theorem is indicated in §6 and §7. In §9 and §10 an $O(n\log n)$ time bound is deduced for an implementation on a r-RAC, $O(n\log^2 n)$ on a RAC.

Corollary There is an algorithm PM' with the following properties:
(a) Given any undirected graph it outputs a perfect matching if one exists, and "failure" otherwise, and

(b) $\exists K\ \exists c\ \forall n\ \forall N \geq cn\log n$, the expected run-time of $PM'(G_{n,N})$ is less than $Kn\log n$ on a r-RAC, or $Kn\log^2 n$ on a RAC.

Proof: PM' consists of first running PM and if this fails, then running a worst-case $O(n^k)$ time algorithm for the same problem (e.g. [5,9]) If K and c are chosen large enough that the probability of failure is $O(n^{-\alpha})$ where $\alpha \geq k-1$ then PM' will have the claimed expected run-time. □

UHC Algorithm

As input the algorithm takes an undirected n node graph G and two specified nodes of G, s and t (which may be the same). The algorithm attempts to find a HP from s to t. If it succeeds then it returns "success". Otherwise it returns "failure". It maintains a partial path P which consists of a simple path with s as one endpoint. The other endpoint is kept in the variable ndp. The algorithm attempts to extend P by calling SELECT to explore edges at random out of ndp.

```
procedure UHC(G,s,t)
begin
1. ndp ← s, P ← ∅.
2. (comment: P consists of a path from s to
   ndp.)
  (a) If P includes every node of G (except
      t, in the case s ≠ t), and if we
      previously deleted {ndp,t} from G,
      then we add {ndp,t} to P and return
      "success".
  (b) v ← SELECT(ndp)
    (i) If v = "*" then return "failure".
   (ii) If v ≠ t and v is not in P then
        begin
          add {ndp,v} to P,
          ndp ← v,
          go to 2.
        end
  (iii) If v ≠ t and v is in P then
        begin
          u ← the neighbour of v in P
              which is closer to ndp,
          delete {u,v} from P,
          add {ndp,v} to P,
          ndp ← u,
          go to 2.
        end
   (iv) Otherwise (i.e. if v = t) go to 2.
end
```

UHC Theorem $\forall\alpha\ \exists M\ \exists c\ \forall N \geq cn\log n$ $\forall s,t (1 \leq s,t \leq n)$ the probability that $UHC(G_{n,N},s,t)$ returns "success" before $Mn\log n$ calls of SELECT is at least $1 - O(n^{-\alpha})$.

The proof of this theorem is indicated in §6 and §8. In §9 and §10 an $O(n\log^2 n)$ time bound is deduced for a RAC.

The above theorems are all stated (and proved) in the fixed N model. However, as the next propositions show, they also hold in the fixed p model. We will see that under relatively weak assumptions a predicate which holds for one model also holds for the other. The final proposition of this section gives a probabilistic reduction of UHC to DHC.

A mapping f from the set of all undirected graphs into the interval [0,1] will be called a graph function. f is a graph predicate iff f is a graph function which takes no values other than 0 and 1. A graph function f is monotone iff whenever G = (V,E) and G'= (V,E') are such that $E \subset E'$ then $f(G) \leq f(G')$.

Examples:

$$f_1(G) = \begin{cases} 1 & \text{if } G \text{ has a HC} \\ 0 & \text{otherwise} \end{cases}$$

is a monotone graph predicate, while

$$f_2(G) = \Pr(UHC(G,1,1) \text{ returns "success" before calling SELECT } Mn\log n \text{ times})$$

is a graph function which is not in general monotone.

Let f be a graph function. Let X be a random variable taking values in the set of undirected graphs. Consider the experiment of drawing a graph G according to the distribution for X and then selecting 1 with probability f(G) and 0 with probability 1 - f(G). The event that 1 is selected as the outcome of this experiment will be denoted by f'(X) = 1.

Propositions 1 and 2 will show that the fixed N and fixed p models are intertranslatable for graph functions even without requiring monotonicity. Proposition 2' shows that Proposition 2 can be slightly strengthened if we do assume monotonicity.

Obvious analogues of the results hold for directed graphs.

Proposition 1 If N(n) is any function such that $\log n/N(n) \to 0$ as $n \to \infty$ and f is any graph function such that

$$N \geq N(n) \Rightarrow \Pr(f'(G_{n,N}) = 1) = 1 - O(n^{-\alpha}) \quad (1)$$

then

$$p \geq p(n) \Rightarrow \Pr(f'(G_{n,p}) = 1) = 1 - O(n^{-\alpha})$$

where $p(n) = (N(n)/\binom{n}{2})(1 + 2\beta(n))$
and $\beta(n) = (2\alpha \log n/\binom{n}{2}(n))^{\frac{1}{2}}$.

Proof The conditions on p imply that
$N(n) \leq \binom{n}{2}p(1 + 2\beta(n))^{-1} \leq (1 - \beta(n))\binom{n}{2}p$
for large enough n. Hence by Fact 4(i)
$\Pr(G_{n,p}$ has $\leq N(n)$ edges$) \leq \exp(-\beta(n)^2\binom{n}{2}p/2) \leq n^{-\alpha}$ (2)
But (abbreviating "$G_{n,p}$ has N edges" by Q_N):

$$\Pr(f'(G_{n,p}) = 1) \geq \sum_{N \geq N(n)} \Pr(f'(G_{n,p}) = 1 | Q_N).\Pr(Q_N) \quad (3)$$

and it is clear in general that
$\Pr(f'(G_{n,p}) = 1|Q_N) = \Pr(f'(G_{n,N}) = 1)$. (4)
Substituting (1),(2), and (4) into (3) gives the desired result. □

Proposition 2 If f is a graph function such that
$p \geq p(n) \Rightarrow \Pr(f'(G_{n,p}) = 1) = 1 - O(n^{-\alpha})$ (5)
then
$N \geq \binom{n}{2}p(n) \Rightarrow \Pr(f'(G_{n,N}) = 1) = 1 - O(n^{1-\alpha})$.

Proof Given $N \geq \binom{n}{2}p(n)$, let $p = N/\binom{n}{2}$.
By Fact 3 there exists a $K > 0$, independent of n, N, such that
$\Pr(G_{n,p}$ has N edges$) \geq K/n$ (6)
But from (4):
$\Pr(f'(G_{n,N}) \neq 1) \leq$
$\Pr(f'(G_{n,p}) \neq 1)/\Pr(G_{n,p}$ has N edges$)$.
Hence by (5) and (6)
$\Pr(f'(G_{n,N}) \neq 1) \leq (n/K)cn^{-\alpha} = O(n^{1-\alpha})$. □

The loss of a factor of n can be avoided if we consider only monotone graph functions and restrict N more. The following generalizes Theorem 3 in [17] .

Proposition 2' Let f be a monotone graph function and suppose $p(n)$ is such that $\log n/(\binom{n}{2}p(n)) \to 0$ as $n \to \infty$. Then if
$p = p(n) \Rightarrow \Pr(f'(G_{n,p}) = 1) = 1 - O(n^{-\alpha})$
then
$N \geq N(n) \Rightarrow \Pr(f'(G_{n,N}) = 1) = 1 - O(n^{-\alpha})$
where $N(n) = (1 + \beta(n))\binom{n}{2}p(n)$
and $\beta(n) = (3\alpha \log n/(\binom{n}{2}p(n)))^{\frac{1}{2}}$

Proof: Define random variables X and Y as follows: select a graph $G = (V,E)$ according to $G_{n,p}$, and let $X = G$. If G has more than N edges, let $Y = *$. Otherwise, add $N - |E|$ edges to G at random and let Y take the resulting value.

From Fact 4(ii) and the assumptions about N, $\Pr(Y = *) = O(n^{-\alpha})$.
By construction,

$$\Pr(f'(G_{n,N}) = 1) = \Pr(f'(Y) = 1 | Y \neq *)$$
$$\geq \Pr(f'(X) = 1 | Y \neq *),$$

by the monotonicity of f . But

$$\Pr(f'(X) = 1 | Y \neq *)$$
$$\geq 1 - \Pr(f'(X) \neq 1) - \Pr(Y = *)$$
$$= 1 - O(n^{-\alpha}).$$ □

Finally we observe that the following reduction can be used to derive from the described DHC algorithm an UHC algorithm alternative to the one described. Note that this reduction assumes the availability of random elements for probabilities p and $\frac{1}{2}$ (see §9 and §10).

Proposition 3 If we select G according to the distribution $G_{n,p}$ and perform the following randomized procedure on it to produce G', then G' will be distributed according to $D_{n,p/2}$:

(i) If $\{i,j\}$ is an edge of G then with probability
$\frac{1}{2} - p/4$ set $(i,j) \in G'$ and $(j,i) \notin G'$
$\frac{1}{2} - p/4$ set $(i,j) \notin G'$ and $(j,i) \in G'$
$p/4$ set $(i,j) \in G'$ and $(j,i) \in G'$
$p/4$ set $(i,j) \notin G'$ and $(j,i) \notin G'$

(ii) If $\{i,j\}$ is not an edge of G then set $(i,j) \notin G'$ and $(j,i) \notin G'$.

Proof It is easy to verify that for any pair $\{i,j\}$, $1 \leq i \neq j \leq n$, we have

$\Pr((i,j) \in G'$ and $(j,i) \notin G') = p(1-p/2)/2$
$\Pr((i,j) \notin G'$ and $(j,i) \in G') = p(1-p/2)/2$
$\Pr((i,j) \in G'$ and $(j,i) \in G') = p^2/4$
$\Pr((i,j) \notin G'$ and $(j,i) \notin G') = (1 - p/2)^2$

Furthermore, these probabilities are independent of the outcomes for all other pairs. From this it can be deduced that for any G_o with N edges
$\Pr(G' = G_o) = (p/2)^N(1 - p/2)^{n(n-1)-N}$
Since any HC in the constructed G' can be made into a HC in the original G by undirecting its edges, a randomized reduction from the UHC problem to the DHC problem is implied. □

4. Basic Lemmas

We call u a neighbour of v in G if G is undirected and contains the edge $\{v,u\}$, or if G is directed and contains the edge (v,u). Assuming particular values of n and d given by context we define the set $S(d)$ to be $\{G|G$ is a graph of n nodes such that every node of G has at least d neighbours$\}$. The implied graphs are directed or undirected according to context. Also, for brevity, we shall denote either of the events $G_{n,N} \in S(K\log n)$ and $D_{n,N} \in S(K\log n)$ simply by $Z(K)$.

Sociability Lemma $\forall\alpha\ \forall K\ \exists c\ \forall N \geq cn\log n$
$\Pr(Z(K)) = 1 - O(n^{-\alpha})$
for both $G_{n,N}$ and $D_{n,N}$.

Proof By Proposition 2 it suffices to prove the above for $G_{n,p}$ with $p \geq c\log n/n$. Given α,K we let $c = 2(\alpha + K + 1)$. If v is any node, the probability that v has fewer than $K\log n$ neighbours can be bounded by $n^{-(\alpha+1)}$ by applying Fact 4(i). The result follows by summing over the n choices of v . □

The following result on conditional probabilities we shall also use widely. It can be verified by induction on n and k.

Bottleneck Lemma Suppose $x_1,x_2,\ldots,x_n$ are random variables taking values in a finite set S.

Let Q be any event. Let
$U = \{s^n \mid s_i \in S \text{ for } 1 \leq i \leq n\}$
Suppose $Y \subset^i U$ has the property that there exist probabilities $p_1, p_2, \ldots, p_k$ such that for each $y^n \in Y$ there exist distinct integers $i_1 < i_2 < \ldots < i_k$ such that
$\Pr(x_{i_j} \in C_Y(y^{i_j-1}) \mid x^{i_j-1} = y^{i_j-1} \wedge Q) \leq p_j$
where

$$C_Y(z^i) = \{s \in S \mid \exists y^n \in Y \text{ with } y^i = z^i \wedge y_{i+1} = s\}$$

(i.e. $C_Y(z^i)$ is the set of continuations of z^i that keep it in Y).

Then $\Pr(x^n \in Y \mid Q) \leq \prod_{j=1}^{k} p_j$. □

Note

Here, as in the remainder of the paper, we could have avoided conditioning on the empty event by adding an appropriate further hypothesis. For brevity, however, we have omitted this throughout.

5. Proof of the DHC Theorem

For brevity we omit unnecessary references to n,N,s, and t which will be assumed to be fixed by context, and will denote $D_{n,N}$ by D. We introduce random variables $T_1, T_2, \ldots, T_{n(n-1)}$ to record the random selection of edges made in one run of DHC on input (D,s,t). Thus $T_i = *$ if DHC returns before calling SELECT i^{1} times, and T_i = the value returned by the i^{th} call of SELECT otherwise.

The "simple events" [10] we consider are the $(n(n-1) + 1)$ - tuples $\langle G_0, w_1, \ldots, w_{n(n-1)}\rangle$ such that $(D = G_0) \wedge (T_1 = w_1) \wedge \cdots \wedge (T_{n(n-1)} = w_{n(n-1)})$ is a possible outcome of the experiment of running DHC on input (D,s,t). "Compound events", or simply "events" are sets of simple events.

Suppose that the event $(D = G_0) \wedge (T_1 = w_1) \wedge \ldots \wedge (T_k = w_k)$ for some $k \leq n(n-1)$ is nonempty. We may simulate DHC with G_0,s, and t using w_i as the result of the i^{th} call of SELECT to determine whether it returns "success" or "failure" before the $(k+1)^{th}$ call, or, otherwise, what the values of ndp and G are at the $(k+1)^{th}$ call of SELECT. Accordingly we have

$\Pr(T_{k+1} = w_{k+1} \mid (D = G_0) \wedge (T_1 = w_1) \wedge \ldots \wedge (T_k = w_k))$

= 1/d if $(D = G_0) \wedge (T^k = w^k)$ gives at least $(k+1)$ calls and at the $(k+1)^{th}$ call (i) out-degree(ndp) = d and (ii) (ndp, w_{k+1}) is an edge of G.

1 if $w_{k+1} = *$ and either the event returns before the $(k+1)^{th}$ call, or ndp has no neighbours at the $(k+1)^{th}$ call of SELECT.

0 otherwise

Note that the probability of any simple event may be calculated from this by applying a suitable generalization of the following identity ([10] p.116):

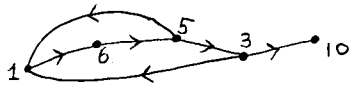
$\Pr(A \wedge B \wedge C) = \Pr(A) \cdot \Pr(B \mid A) \cdot \Pr(C \mid A \wedge B)$.

Furthermore, given a nonempty event $T^k = w^k$ (but not the value of D) the action of DHC and the successive values of P and ndp are uniquely determined up to the $(k+1)^{th}$ call of SELECT or up to the return from the DHC procedure if that occurs first. If "failure" has not occurred, then any value of D containing all the edges explored for $T^k = w^k$ is still possible. However the relative probabilities of these possible values will in general have been conditioned by the event.

We define w^k to be an expedition iff $T^k = w^k$ is a nonempty event. Let w^k be an expedition and suppose that we simulate DHC using w_i as the value returned by the i^{th} call of SELECT.[1] Then w^k is unfinished iff DHC does not return before the $(k+1)^{th}$ call of SELECT. In that case we define $P(w^k)$ and $ndp(w^k)$ to be the values of P and ndp respectively at the $(k+1)^{th}$ call of SELECT. $G(G_0, w^k)$ is the value of G at the $(k+1)^{th}$ call of SELECT if $G = G_0$ initially (provided that $(D = G_0) \wedge (T^k = w^k)$ is a nonempty event).

Let j be the largest integer not exceeding k such that $w_j \neq *$ in the expedition w^k. Determine the nodes $v_1, v_2, \ldots, v_j$ that are the arguments of SELECT at the first[j] j calls of it. Then for $1 \leq i \leq j$ we have the following definitions: (i) (v_i, w_i) is the i^{th} step of w^k; (ii) w^k departs from v at step i iff $v = v_i$; (iii) w^k arrives at v at step i iff $v = w_i$; (iv) w^k makes v the departure point at step $(i-1)$ iff either $i = 1$ and $v = v_1$, or $i > 1$ and $v = v_i$ and $v \neq_k v_{i-1}$ ("step 0"[1] is a slight anomaly here); (v) w^{i-1} has a chance to close at step i if w^{i-1} is unfinished and $P(w^{i-1})$ is a directed path containing every node (except t, in the case $s \neq t$). We say that w^k departs from v m times iff $|\{i \mid w^k \text{ departs from } v \text{ at step } i \leq k\}| = m$, and use analogous terminology for (iii),(iv), and (v).

Example: n = 10, N = 20, s = 1, t = 10. Then $w^8 = \langle 6,5,1,3,10,1,*,*\rangle$ is an expedition. The steps of w^8 are (1,6),(6,5),(5,1),(5,3),(3,10), (3,1). w^8 makes node 5 the departure point once (at step 2) and departs from node 5 twice (at steps 3 and 4). Given $T^8 = w^8$ we know that node 3 has only two neighbours, 1 and 10, and DHC fails because it exhausts the edges out of node 3.



We next give a fundamental lemma that states that the random selection of edges by DHC makes every possible continuation of an expedition that has not exhausted the edges out of its next departure point equally likely.

Equiprobability (EP) Lemma Let d be a positive integer such that $dn \leq N$. Suppose that w^k is an unfinished expedition with $u = ndp(w^k)$ and suppose that w^k departs from u exactly b times, where $b < d$. Then for any node z such that (u,z) is not a step of w^k

$$\Pr(T_{k+1} = z \mid T^k = w^k \wedge D \in S(d)) = (n-b-1)^{-1}$$

Proof By symmetry. (N.B. In contrast to the undirected case, where conditioning problems arise (see §6) the result here is immediate.)

For simplicity we shall denote $T^{n(n-1)}$ and $w^{n(n-1)}$ by T and w respectively in the rest of this section. Let

$F_M = \{w | w$ is an expedition which does not return "success" within the first $Mn\log n$ steps $\}$.

The DHC theorem we need to prove is therefore equivalent to the following

Theorem $\forall\alpha \quad \exists c \quad \exists M \quad \forall N \geq cn\log n$
$\Pr(T \in F_M) = O(n^{-\alpha})$.

Proof First note that $\forall K_1$
$\Pr(T \in F_M) \leq (1 - \Pr(Z(K_1))) + \Pr(T \in F_M | Z(K_1))$

The sociability lemma can be used to bound the first term. To bound the second we introduce the following sets of expeditions which we define in terms of n, M, K_1, K_2, and K_3 (here regarded as general variables).

$A = \{w|$ for $i = Mn\log n$ w^i departs from some node at least $K_1 \log n$ times$\}$

$B = \{w| w \notin A$ and for $i = K_2 n\log n$ w^i does not arrive at every node other than $s\}$

$C = \{w| w \notin A, w \notin B, w \in F_M$, and w has fewer than $K_3 n\log n$ chances to close within the first $Mn\log n$ steps$\}$

$E = F_M - (A \cup B \cup C)$

Clearly $F_M \subset A \cup B \cup C \cup E$ for all n, M, K_1, K_2, K_3 and therefore

$$\Pr(T \in F_M | Z(K_1)) \leq \Pr(T \in A | Z(K_1)) + \Pr(T \in B | Z(K_1)) + \Pr(T \in C | Z(K_1)) + \Pr(T \in E | Z(K_1)).$$

For each of the quantities on the r.h.s. we prove a lemma to bound it.

Lemma E $\forall\alpha \quad \exists K_3 \quad \forall K_1 \quad \forall K_2 \quad \forall M \geq K_3$
$\Pr(T \in E | Z(K_1)) = O(n^{-\alpha})$.

Lemma B $\forall\alpha \quad \exists K_2 \quad \forall K_1 \quad \forall M \geq K_2$
$\Pr(T \in B | Z(K_1)) = O(n^{-\alpha})$.

Hence for any α we can find K_2 and K_3 satisfying the above two lemmas simultaneously for all K_1 and for all $M \geq K_2, K_3$.

Lemma C $\forall\alpha \quad \forall K_2 \quad \forall K_3 \quad \exists M \geq K_2, K_3 \quad \forall K_1$
$\Pr(T \in C | Z(K_1)) = O(n^{-\alpha})$.

Hence we can find a suitable $M(\alpha, K_2, K_3)$ also.

Lemma A $\forall\alpha \quad \forall M \quad \exists K_1$
$\Pr(T \in A | Z(K_1)) = O(n^{-\alpha})$.

Hence we can find an appropriate $K_1(\alpha, M)$. Finally, by the sociability lemma, we can choose a $c(\alpha, K_1)$ such that $N \geq cn\log n$ guarantees
$\Pr(Z(K_1)) = 1 - O(n^{-\alpha})$.

This establishes the theorem. It only remains to prove the four lemmas. □

Proof of Lemma A

Define the following two sets of expeditions:

$H = \{w|$ for some $i \leq Mn\log n$ w^i departs from each node $< K_1 \log n$ times, and makes some node the departure point $\geq K\log n$ times $\}$.

$J = \{w|$ for some $i \leq Mn\log n$ w^i makes each node the departure point $\leq K\log n$ times and departs from some node $\geq K_1 \log n$ times$\}$.

The reader can easily verify that for all K, K_1, and M $A \subset (H \cup J)$. We therefore need the following results:

Sublemma H $\forall\alpha \quad \forall M \quad \exists K \quad \forall K_1$
$\Pr(T \in H | Z(K_1)) = O(n^{-\alpha})$.

Sublemma J $\forall\alpha \quad \forall M \quad \forall K \quad \exists K_1$
$\Pr(T \in J | Z(K_1)) = O(n^{-\alpha})$.

Clearly for all α and M we can find a $K(\alpha, M)$ from sublemma H and hence a $K_1(\alpha, M)$ from sublemma J such that both are satisfied simultaneously. Lemma A therefore follows. □

Proof of Sublemma H

Let $X = \{Y | Y \subset \{0, 1, \ldots, Mn\log n\}$ and $|Y| = K\log n\}$. For $v \in V$ and $Y \in X$ define $H(v,Y)$ to be $\{w|$ $w \in H$ and for each $i \in Y$, w makes v the departure point at step i, and w^i departs from each node $\leq K_1 \log n$ times$\}$. Then

$$H = \bigcup_{\substack{v \in V \\ Y \in X}} H(v,Y) \quad \text{and hence}$$

$$\Pr(T \in H | Z(K_1)) \leq \sum_{\substack{v \in V \\ Y \in X}} \Pr(T \in H(v,Y) | Z(K_1))$$

Fix v, Y, $w \in H(v,Y)$ and $i \in Y$ $(i > 0)$. Clearly w^{i-1} is unfinished and there are two possibilities:

(i) Assume $v \notin P(w^{i-1})$. Then if v is made the departure point at step i then w must arrive at v at step i (i.e. $T_i = v$). Hence by the EP lemma

$\Pr(T_i = v | T^{i-1} = w^{i-1} \wedge Z(K_1)) \leq (n - K_1 \log n - 1)^{-1}$.

(ii) Assume $v \in P(w^{i-1})$. Then there is a unique node y such that $(v,y) \in P(w^{i-1})$, and if v is to be made the departure point at step i then it must be that $T_i = y$. Hence by the EP lemma

$\Pr(T_i = y | T^{i-1} = w^{i-1} \wedge Z(K_1)) \leq (n - K_1 \log n - 1)^{-1}$.

In either case the probability of a continuation of w^{i-1} remaining in $H(v,Y)$ is at most $(n - K_1 \log n - 1)^{-1}$. Since this holds for each $i \in Y (i > 0)$ it follows from the bottleneck lemma that:

$\Pr(T \in H(v,Y) | Z(K_1)) \leq (n - K_1 \log n - 1)^{-K\log n + 1}$.

Summing over all $v \in V$ and $Y \in X$, and then appealing to Fact 2 gives

$$\Pr(T \in H | Z(K_1)) \leq n \binom{Mn\log n + 1}{K\log n} (n - K_1 \log n - 1)^{-K\log n + 1} \leq n^2 \left(\frac{Me(1 + o(1))}{K} \right)^{K\log n} \quad \text{for any } K_1$$

Choosing $K > \max\{\alpha + 2, Me^2\}$ gives the result. □

Proof of Sublemma J

Let $X = \{< l_1, \ldots, l_r > \mid 1 \leq r \leq K\log n$ each $l_i \geq 1$, and $\Sigma l_i = K_1 \log n\}$.

For any $v \in V$ and $Y \in X$ we let $J(v,Y)$ be $\{w \in J | \exists i_1, i_2, \ldots, i_r$ such that for each $j, 1 \leq j \leq r$, $\overleftarrow{w}$ makes v the departure point at step i_j and departs from v at steps $i_j + 1, \ldots, i_j + l_j$. Furthermore, $w^{i_r + l_r - 1}$

departs from each node $< K_1 \log n$ times.} Then

$$J = \bigcup_{\substack{v \in V \\ Y \in X}} J(v,Y) \quad \text{and thus}$$

$$\Pr(T \in J | Z(K_1)) \leq \sum_{\substack{v \in V \\ Y \in X}} \Pr(T \in J(v,Y) | Z(K_1)) .$$

Fix $v \in V$, $Y \in X$, and $w \in J(v,Y)$. The first $K_1 \log n$ departures of w from v appear in $r \leq K \log n$ groups:

$$i_1+1,\ldots,i_1+l_1;\ i_2+1,\ldots,i_2+l_2;\ i_r+1,\ldots,i_r+l_r;$$

Fix j $(1 \leq j \leq r)$ and suppose that $l_j > 1$. Let i be such that $i_j + 1 \leq i < i_j + l_j$. Then w^{i-1} is unfinished and departs from each node fewer than $K_1 \log n$ times. There are two cases:

(i) Suppose $P(w^{i-1})$ consists of a simple path from s to v. Let $F = \{y | y = t$ or y is in $P(w^{i-1})$ and there are fewer than $n/2$ nodes between y and v in $P(w^{i-1})\}$. Clearly $|F| \leq n/2 + 1$ and we will depart from v again at step $i + 1$ iff $T_i \in F$. By the EP lemma

$$\Pr(T_i \in F | T^{i-1} = w^{i-1} \wedge Z(K_1)) \leq (1 + n/2)/(n - K_1 \log n - 1).$$

(ii) Suppose $P(w^{i-1})$ consists of a path from s to v and a directed cycle on $\geq n/2$ points. Let $F = \{ y | y = t$ or y is on the path from s to $v \}$. Then $|F| \leq n/2 + 1$ and we depart again from v at step $i + 1$ iff $T_i \in F$. By the EP lemma

$$\Pr(T_i \in F | T^{i-1} = w^{i-1} \wedge Z(K_1)) \leq (1 + n/2)/(n - K_1 \log n - 1).$$

Thus in either case the probability of a continuation of w^{i-1} which remains in $J(v,Y)$ is $\leq (1+o(1))/2$. Since there are $\geq (K_1 - K)\log n$ distinct values of i it follows by the bottleneck lemma that

$$\Pr(T \in J(v,Y) | Z(K_1)) \leq ((1 + o(1))/2)^{(K_1 - K)\log n}$$

Summing over $v \in V$ and $Y \in X$, and noting that, provided $K_1 \geq 2K$, $|X| \leq K \log n \binom{K_1 \log n}{K \log n}$ and that this can be bounded by Fact 2, we have

$$\Pr(T \in J | Z(K_1)) \leq K n \log n \left(\frac{2K_1 e}{K} \right)^{K \log n} \left(\frac{1 + o(1)}{2} \right)^{K_1 \log n}$$

This is $O(n^{-\alpha})$ if K_1 is chosen large enough with respect to K and α. □

Proof of Lemma B

Given α choose $K_2 > \alpha + 1$ and suppose that $M \geq K_2$ and K_1 is arbitrary. For $v \in V (v \neq s)$ let $B(v) = \{w^1 \in B |$ v is the least numbered node other than s such that w^1 does not arrive at v for any $i \leq K_2 \log n \}$. Clearly $B = \bigcup_{v \in V} B(v)$. Fix $v \in V (v \neq s)$ and $w \in B(v)$. For $1 \leq i \leq K_2 \log n$, w^i clearly cannot return "success". Also, we may ignore the possibility that w^i returns "failure", for then $w \notin A$ and $M \geq K_2$ imply $\Pr(T = w | Z(K_1)) = 0$. Therefore we may assume w^{i-1} is unfinished and hence by the EP lemma

$$\Pr(T_i \neq v | T^{i-1} = w^{i-1} \wedge Z(K_1)) \leq 1 - 1/n.$$

Thus the probability of a continuation of w^{i-1} remaining in $B(v)$ is $\leq 1 - 1/n$ for each i and hence, by the bottleneck lemma

$$\Pr(T \in B(v) | Z(K_1)) \leq (1 - 1/n)^{K_2 n \log n}$$

Summing over v and using Fact 1 gives

$$\Pr(T \in B | Z(K_1)) \leq n e^{-K_2 \log n} = O(n^{-\alpha}). \quad \square$$

Proof of Lemma C

Let $X = \{Y \subset \{1,2,\ldots,(M - K_2)n\log n\} | |Y| = K_3 n \log n \}$. For $Y \in X$ let $C(Y) = \{w \in C |$ for each i $(1 \leq i \leq (M - K_2)n\log n)$ if w has a chance to close at step $i + K_2 n \log n$ then $i \in Y\}$. Clearly $C = \bigcup_{Y \in X} C(Y)$.

Fix $Y \in X$ and $w \in C(Y)$. Let $W = \{1,2, \ldots (M - K_2)n\log n\} - Y$ and suppose that $(i + 1^2 - K_2 n \log n) \in W$. Now $C \subset F_M$ ensures that w^{i+1} does not return "success". $w \notin A$ ensures that the case of it returning "failure" can be ignored (since then $\Pr(T = w | Z(K_1)) = 0$). $w \notin B$ ensures that w^1 arrives at every node other than s. By the choice of i, w does not have a chance to close at step $i + 1$, which implies that $P(w^i)$ consists of a path and a cycle. To analyse the probability of $P(w^i)$ consisting of a path and a cycle we consider two cases:

(i) Assume that $P(w^{i-1})$ consists of a simple path. If F is the set of all nodes y in $P(w^{i-1})$ such that there are $\geq n/2$ nodes between y and $ndp(w^{i-1})$, then it follows from the EP lemma that

$$\Pr(T_i \in F | T^{i-1} = w^{i-1} \wedge Z(K_1)) \leq (n/2)/(n - K_1 \log n - 1).$$

(ii) Assume that $P(w^{i-1})$ consists of a path and a cycle of $\geq n/2$ nodes. If F is the set of all the nodes that are not in the cycle then it follows from the EP lemma that

$$\Pr(T_i \in F | T^{i-1} = w^{i-1} \wedge Z(K_1)) \leq (n/2)/(n - K_1 \log n - 1).$$

Thus in either case the probability of a continuation within $C(Y)$ is at most $(1 + o(1))/2$ (for any K_1) for each of the $(M - K_2 - K_3)n\log n$ values of i such that $(i + 1 - K_2 n \log n)^3 \in W$. By applying the bottleneck lemma, summing over $Y \in X$, and then using Fact 2,

$$\Pr(T \in C | Z_1(K_1)) \leq \binom{(M - K_2)n\log n}{K_3 n \log n} ((1 + o(1))/2)^{(M - K_2 - K_3)n\log n}$$

$$\leq \left(\frac{2(M - K_2)e}{K_3} \right)^{K_3 n \log n} \left(\frac{1 + o(1)}{2} \right)^{(M - K_2)n\log n}$$

which is $O(n^{-\alpha})$ for M sufficiently large with respect to α, K_2 and K_3 (for any K_1). □

Proof of Lemma E

Let α be given and choose $K_3 \geq \alpha$. Let K_1, K_2, and $M \geq K_3$ be given. Suppose $w \in E$. Then $w \notin C$ ensures that for some set $\{i_1,\ldots,i_r\}$ such that $1 \leq i_1 < i_2 \ldots < i_r \leq Mn\log n$ where $r = K_2 n \log n$, w^{i_j} has a chance to close at each step i_j. Also $w \in F_M$ and $w \notin A$ ensure that we need only consider the case when $w^{i_j - 1}$ is unfinished. If $u = ndp(w^{i_j - 1})$ then (u,t) cannot be a step of w^{i_j} for otherwise w^{i_j} would return "success". Then, by the EP lemma

$$\Pr(T_{i_j} \neq t | T^{i_j - 1} = w^{i_j - 1} \wedge Z(K_1)) \leq 1 - 1/n$$

and by the bottleneck lemma, and Fact 1:

$\Pr(T \in E|Z(K_1)) \leq (1 - 1/n)^{K_3 n \log n} \leq n^{-\alpha}$

by our choice of K_3. □

6. Basic Lemmas for Undirected Analysis

We shall first define a more general notion of "expedition" that allows us to discuss PM and UHC (as well as DHC) in the same framework. We then prove the Almost Equiprobability (AEP) Lemma which serves as an analogue of the EP Lemma in the undirected case.

A function f is a next departure rule iff for every k-tuple $(w_1,w_2,\ldots,w_k)$ with $k \geq 0$ and each $w_i \in V$, $f(w_1,w_2,\ldots,w_k)$ is defined and has value in $V \cup \{+\}$. (In particular, $f(\ )$ is the first node to be chosen for exploration.) For such an f we define the following algorithm U_f:

```
begin   i ← 0 ;
  loop: ndp ← f(w^i) ;
        if ndp = "+" then return "success";
        i ← i + 1 ;
        w_i ← SELECT(ndp) ;
        if w_i = "*" then return "failure" ;
        go to loop ;
end
```

It should be clear that we could define an appropriate f (independent of the input graph) to cast each of DHC, PM, and UHC into this form.

Henceforth we restrict attention to undirected graphs, and denote $G_{n,N}$ by G. The random variables T_i, simple events, and expeditions are defined as in §5 except that "$n(n-1)$" is now replaced by "$n(n-1)/2$", D by G, and the DHC algorithm by U_f.

Let w^k be an expedition. Then it is unfinished iff $(w_k \neq * \text{ or } k = 0)$ and $f(w^k) \neq +$. It returns success iff for some $i \leq k$ $w_i \neq *$ and $f(w^{i-1}) = +$. It returns failure iff for some $i \leq k$, w^{i-1} is unfinished, $f(w^{i-1}) \neq +$ and $w_i = *$. If j is the largest integer $\leq k$ such that $w_j \neq *$ then we define $v_i = f(w^i)$ for $0 \leq i \leq j$. Then (v_i,w_i) is the i^{th} step of w^k. At the i^{th} step w^k departs from v iff $v = v_i$, arrives at v iff $w_i = v$, and visits v iff $v_i = v$ or $w_i = v$. Finally, if $(G = G_0) \wedge (T^k = w^k)$ is nonempty and j is the largest integer $\leq k$ such that $w_j \neq *$ then $G(G_0,w^k)$ is the graph left when the edges $\{v_1,w_1\}$, $\ldots,\{v_j,w_j\}$ are deleted from G_0, where $v_i = f(w^{i-1})$.

If $(G = G_0) \wedge (T^k = w^k)$ is nonempty then

$\Pr(T_{k+1} = w_{k+1} \mid G = G_0 \wedge T^k = w^k)$

$= 1/d$ if w^k is unfinished and w_{k+1} is one of the d neighbours of $f(w^k)$ in $G(G_0,w^k)$.

1 if $w_{k+1} = *$ and either w^k is unfinished but $f(w^k)$ has no neighbours in $G(G_0,w^k)$, or w^k is not unfinished.

0 otherwise.

As in the directed case this allows us to calculate the probability of any simple event. The EP lemma, however, does not hold since edges can now be explored in either direction and the results of explorations from v will in general condition the probability of arriving at v.

The following bounds the decrease in probability of a particular expedition in a graph when the degree of one of its nodes is increased by one. (Straightforward generalizations to larger perturbations can also be derived easily.)

Perturbation Lemma Suppose that G_1 and G_2 are graphs, that w^k is a possible expedition on both G_1 and G_2, that for some node v degree(v in G_2) = 1 + degree(v in G_1) = 1 + d and degree(w in G_2) $\leq$ degree(w in G_1) $\forall\, w \neq v$, and that w^k visits v at most b times. Then

$\Pr(T^k = w^k \mid G = G_1) \leq K \cdot \Pr(T^k = w^k \mid G = G_2)$

where $K = \exp(b/(d - b + 1))$.

Proof The initial relations between the degrees of nodes in G_1 and G_2 are preserved since the same edges are explored and deleted in both. If w^{i-1} is unfinished and $f(w^{i-1}) = v$ has current degree a (in $G(G_1,w^{i-1})$), then

$\Pr(T_i = w_i \mid T^{i-1} = w^{i-1} \wedge G = G_1) = 1/a$ and

$\Pr(T_i = w_i \mid T^{i-1} = w^{i-1} \wedge G = G_2) = 1/(a+1)$.

For all other values of $f(w^{i-1})$

$\Pr(T_i = w_i \mid T^{i-1} = w^{i-1} \wedge G = G_1)$

$\leq \Pr(T_i = w_i \mid T^{i-1} = w^{i-1} \wedge G = G_2)$.

Hence we have

$\Pr(T^k = w^k \mid G = G_1) = \frac{1}{d_1} \cdots \frac{1}{d_r} Q_1$,

$\Pr(T^k = w^k \mid G = G_2) = \frac{1}{d_1+1} \cdots \frac{1}{d_r+1} Q_2$,

if w^k departs from v r times and $d_1,d_2,\ldots,d_r$ are the degrees of v in G_1 at the successive departures. Since also $Q_1 \leq Q_2$, the ratio of the two probabilities is bounded by

$(1 + 1/d_1) \ldots (1 + 1/d_r) \leq \exp(r/d_r)$.

Since $r \leq b$ and $d_r \geq d - b + 1$ the result follows. □

Almost Equiprobability (AEP) Lemma Suppose w^k is an unfinished expedition, with $f(w^k) = u$, which visits each node at most $\text{rlog } n$ times and $z(\neq u)$ is such that neither (u,z) nor (z,u) is a step of w^k. Then

$$\frac{1-\varepsilon}{(1+\varepsilon)n} \leq \Pr(T_{k+1}=z \mid T^k=w^k \wedge Z(K)) \leq \frac{1+\varepsilon}{n-r\log n-1}$$

if $K > r$, $c \geq 2K$, $N \geq cn\log n$ and n is sufficiently large, where

$1 + \varepsilon = (\exp(r/(K - r)))(c/(c - K))$.

Proof We have to bound the relative probabilities of arrival at z_1 and z_2 where these are any two of the $\geq n - r\log n - 1$ possible nodes. Let $R = (\Pr(T^k = w^k \wedge Z(K))^{-1}$ and let $g(x,G_0) = \Pr(T_{k+1} = x \wedge T^k = w^k \wedge G = G_0)$. Then for $i = 1,2$

$$\Pr(T_{k+1} = z_i \mid T^k = w^k \wedge Z(K)) = R \sum_{G_0 \in A_i} g(z_i,G_0) \quad (1)$$

where $A_i = \{G_0 \in Z(K) \mid \Pr(T^k = w^k \wedge G = G_0) > 0$ and $\{u,z_i\}$ is an edge of $G_0\}$.

Let $S_0 = A_1 \wedge A_2$, $S_1 = A_1 - A_2$ and $S_2 = A_2 - A_1$. Clearly for each $G_0 \in S_0$

$$\Pr(T_{k+1} = z_1 \mid T^k = w^k \wedge G = G_0) = \Pr(T_{k+1} = z_2 \mid T^k = w^k \wedge G = G_0).$$

Hence $$\sum_{G_0 \in S_0} g(z_1,G_0) = \sum_{G_0 \in S_0} g(z_2,G_0) \qquad (2)$$

Let $B_1 = \{G_0 \in S_1 \mid \text{degree}(z_1) \text{ in } G_0 \text{ is exactly } K\log n\}$, and let $C_1 = S_1 - B_1$. Now each member G_0 of C_1 can be mapped into a member G_0^* of S_2 by replacing the edge $\{u,z_1\}$ by $\{u,z_2\}$. Since this mapping of G_0 to G_0^* satisfies the perturbation lemma, if $K_1 = \exp(r/(K - r))$, then for each $G_0 \in C_1$,

$$\Pr(T^k = w^k \mid G = G_0) \leq K_1 \Pr(T^k = w^k \mid G = G_0^*).$$

Also, $$\Pr(T_{k+1} = z_1 \mid T^k = w^k \wedge G = G_0) = \Pr(T_{k+1} = z_2 \mid T^k = w^k \wedge G = G_0^*).$$

Hence $$\sum_{G_0 \in C_1} g(z_1,G_0) \leq K_1 \cdot \sum_{G_0 \in S_2} g(z_2,G_0). \qquad (3)$$

It remains to bound the contributions of $G_0 \in B_1$. For $G_0 \in B_1$ define

$Q(G_0) = \{(v,x) \mid \{v,x\} \in G_0$; (v,x) and (x,v) are not steps of w^k ; degree(x in G_0) $\geq K\log n + 1$; $x \neq z_1,z_2$; $\{v,z_1\} \notin G_0\}$ and

$F(G_0) = \{G_0^* \mid G_0^*$ is G_0 with $\{u,z_1\}$ replaced by $\{u,z_2\}$ and $\{v,x\}$ by $\{v,z_1\}$ for some $(v,x) \in Q(G_0)\}$

Then F is a multi-valued mapping from B_1 into S_2 that satisfies the perturbation lemma. Hence it follows (as in (3) above) that $\forall\, G_0 \in B_1$, $\forall\, G_0^* \in F(G_0)$,

$$g(z_1,G_0) \leq K_1 g(z_2,G_0^*).$$

We shall now show that for all $G_0 \in B_1$, $G^* \in S_2$, and large enough n

$$|F(G_0)| \geq (c - K)n\log n - K^2\log^2 n \qquad (4)$$

and $$|F^{-1}(G^*)| \leq (n - K\log n)K\log n . \qquad (5)$$

It then follows that

$$((c - K)n\log n - K^2\log^2 n) \cdot \sum_{G_0 \in B_1} g(z_1,G_0) \leq K_1 \cdot \sum_{G_0 \in B_1} \sum_{G^* \in F(G_0)} g(z_2,G^*)$$

$$\leq K_1(n - K\log n)K\log n \sum_{G^* \in S_2} g(z_2,G^*). \qquad (6)$$

(Proof of (4) : w^k uses fewer than $K\log n$ edges from each node. Hence $K\log n$ edges can be set aside at each node, including all those explored by w^k, and still at least $(c - K)n\log n$ edges remain. Of these at most $(K\log n + 2)$ can have both end-points in the set $\{z_1,z_2$, neighbours of z_1 in $G_0\}$. This gives a lower bound for $|Q(G_0)|$. Since clearly $|Q(G_0)| = |F(G_0)|$ result (4) follows.

Proof of (5): The only way of recovering from G^* a value $G_0 \in F^{-1}(G^*)$ is the following: (i) select one of the $K\log n$ nodes v such that $\{v,z_1\} \in G^*$, (ii) select one of the $\leq n - K\log n$ nodes x which is not a neighbour of v in G^*, (iii) replace $\{u,z_2\}$ by $\{u,z_1\}$ and $\{v,z_1\}$ by $\{v,x\}$.)

We now combine results (2),(3), and (6) and use the facts that $A_2 = S_0 \cup S_2$ and $A_1 = S_0 \cup B_1 \cup C_1$ to give

$$\sum_{G_0 \in A_1} g(z_1,G_0) \leq K_1(c/(c - K)) \sum_{G_0 \in A_2} g(z_2,G_0).$$

Applying (1) shows that all the $\geq (n - r\log n - 1)$ possible outcomes have probabilities differing from each other by a factor of at most $(1 + \varepsilon)$. The result follows. □

7. & 8. Proofs of PM and UHC Theorems

Proofs of the main theorems for PM and UHC are omitted because the techniques needed have all been introduced in the DHC proof. The proofs are, however, rather easier than the DHC proof because the algorithms analysed are simpler. The proof strategy is the same: the event of the algorithm failing is decomposed into a number of subevents, and the probabilities of each of these is bounded by the same techniques as those used in the lemmas of §5. The main difference is that the AEP lemma must be used instead of the EP lemma. The definitions introduced in §6 suffice.

9. Derandomization

The foregoing analysis was greatly facilitated by the fact that we were dealing with randomized algorithms. The purpose of this section is to show that all the main algorithms can, in principle, be "derandomized" so as to become deterministic algorithms. The idea is simply to use some otherwise unused parts of the random graph to provide the necessary random decisions in the algorithm. This derandomization procedure appears to be of theoretical interest only.

We shall discuss the derandomization of DHC for $D_{n,p}$ with $(c\log n)/n \leq p \leq 1 - (c\log n)/n$. (The case of the remaining extremely dense graphs will be treated at the end of this section.) The same method can easily be applied to PM and UHC. By invoking Propositions 1 and 2 the derandomized results for all three problems for $D_{n,N}$ or $G_{n,N}$ follow immediately.

The deterministic DHC procedure works as follows: Let $L = \{1,2,\ldots,n/2\}$ and $U = \{n/2 + 1, \ldots,n\}$. Let $t_1 = \max\{j \in L \mid j \neq 1$ and $(j,n) \in G\}$ and $t_2 = \min\{j \in U \mid j \neq n$ and $(j,1) \in G\}$. (Return "failure" in the unlikely event that either of these sets is empty.) Simulate DHC twice, once on the subgraph of G induced by L with $s = 1$ and $t = t_1$, and once on the subgraph of G induced by U with $s = n$ and $t = t_2$. The success of both calls clearly guarantees a HC in G. Edges of G going between $L' = L - \{1\}$ and $U' = U - \{n\}$ are clearly independent of the two subgraphs; We show how to use these edges as a source of the random bits required in the two simulations of DHC.

Considering just the edges in $L' \times U'$, we can regard the source as kn^2 probability p Bernoulli trials [10,p.146]. If we divide them up into pkn^2 segments of about $1/p$ bits then with large probability at least a fixed fraction γ of the segments will contain exactly one 1 and the position of the 1 in the segment will provide about $\log_2(1/p)$ random bits (i.e. probability $\frac{1}{2}$ Bernoulli trials). Hence at least $\gamma kc\log^2 n$ random bits can be expected, and, with large probability this is enough for $c\log n$ edge selections provided c is large with respect to ε. Note that if $p = (c\log n)/n$ then $O(n\log n\log\log n)$ random bits are sufficient. (The above refers to the case $p \leq \frac{1}{2}$. In the alternative case one proceeds similarly, but looking for 0's rather than 1's.)

We formalize these claims in the following lemmas, which can be verified easily by using Facts 2 and 4. The first lemma ensures that the algorithm can be specified without knowing p. It is sufficient to use as an approximation $\hat{p}$, which is defined as the ratio of the number of edges of G in $U' \times L'$ to $|U' \times L'|$. (N.B. The edges in $U' \times L'$ are independent of those in $L' \times U'$).

Lemma 1 $\forall \alpha \quad \exists d > 0$ such that

$Pr(|p - \hat{p}| \geq d(\text{plog } n)^{\frac{1}{2}}/n) = O(n^{-\alpha})$.

We choose the segments of the source $(L' \times U')$ to be of length $\lfloor 1/\hat{p} \rfloor$.

Lemma 2 $\exists \gamma > 0 \quad \forall d$

$|p - \hat{p}| < d(\text{plog } n)^{\frac{1}{2}}/n \Rightarrow$ the probability that any given segment has exactly one 1, and will therefore provide $\log_2 \lfloor 1/\hat{p} \rfloor$ random bits is at least γ, for all sufficiently large n.

Lemma 3 $\exists \delta > 0 \quad \forall \alpha \quad \forall d$

$|p - \hat{p}| < d(\text{plog } n)^{\frac{1}{2}}/n \Rightarrow$
$Pr(\text{fewer than } \delta n^2 \hat{p}\log(1/\hat{p}) \text{ random bits can be generated from the source by the implied method}) = O(n^{-\alpha})$.

Since the quantity $\hat{p}\log(1/\hat{p})$ is minimized for $\hat{p} \in [(c\log n)/n - d(\text{plog } n)^{\frac{1}{2}}/n, \frac{1}{2} + d(\text{plog } n)^{\frac{1}{2}}/n]$ at the lower endpoint, for n sufficiently large, the claimed $\delta' cn\log^2 n$ random bits can be expected with appropriate probability. We leave to §10 the task of analysing the complexity of extracting these random bits.

The only case we have not considered is that of very dense graphs. If $p > 1 - (c\log n)/n$ then we may not have enough "randomness" in the input graph. However, the graph is so dense that just about any sensible deterministic algorithm will work with appropriate probability. This is because we can afford to test for the existence of named edges (e.g. (3,7)) with little likelihood of failure. One such algorithm does the following: (i) first establish a long simple path in G, (ii) make this into a large cycle by testing for edges from the endpoint to near the beginning, and (iii) enlarge this to full size by doing the following: for each out-of-cycle node, z, look for a consecutive pair of nodes (u,v) (consecutive in the cycle,) which will permit z to be inserted in the cycle between them. With probability $1 - O(n^{-\alpha})$ this procedure will find a HC within $K(\alpha)\cdot n$ edge-tests, provided $p(n)$ is bounded below by some positive constant, for all sufficiently large n. On a RAC this would require time $O(n)$ for adjacency matrix representations, and $O(n\log n)$ for adjacency list representations of the input graph.

10. Implementation

We shall now outline possible implementations of the three main algorithms, first for r-RACs, and then, with the aid of derandomization, for RACs. We must emphasize that for practical purposes we would recommend the randomized versions of the algorithms in conjunction with a random number generator. Indeed, some preliminary experimental tests in this direction appear encouraging. The deterministic implementations are only for the theoretical purpose of making direct comparisons with other deterministic algorithms possible.

Representation of Inputs

In the directed case we assume that $G = (V,E)$ is given in adjacency list form. Each list $L_v = \{w \mid (v,w) \in E\}$ is stored in increasing order in the RAC memory. Also the value of $n = |V|$ is available in a word, and there are two length n vectors $SA[v]$ and $LEN[v]$ giving respectively the start address and length of L_v for $i = 1,2,\ldots,n$.

In the undirected case the representation is similar, with lists $L_v = \{w \mid \{v,w\} \in E\}$. However the two copies of $\{v,w\}$ (i.e. in L_v and L_w) are doubly-linked so that they can be easily deleted together.

Implementation on a r-RAC

Whenever an edge is deleted the corresponding item(s) in the adjacency lists are made zero. A vector $CLEN[v]$ is maintained to record the current number of non-zero entries in each list. When calling SELECT(u), we fail if $CLEN[u] = 0$. Otherwise we extract the number x represented by the first $\lceil \log_2(LEN[u]) \rceil$ bits of a random word supplied by the r-RAC. (This word will always be sufficiently long, by the definition of r-RACs). In this way we generate a succession of values x until we find one such that $1 \leq x \leq LEN[u]$ and the x^{th} entry of L_u is a non-zero value, say w. We then delete the edge (u,w) (or $\{u,w\}$). By choosing the input density constant c large enough we ensure that in $Mn\log n$ calls of SELECT we shall call on the random element more than (say) $\frac{1}{4}Mn\log n$ times with probability only $O(n^{-\alpha})$.

For the DHC and UHC algorithms we need to represent paths and cycles. For the DHC algorithm we need the following operations: (i) adding a node to the end of a list, (ii) determining the ordinal position of a node in a list, (iii) splitting a list (to form a path and a cycle, or to rejoin a path and a cycle into a path), and (iv) concatenating lists. Each of these operations can be carried out in $O(\log n)$ time if balanced trees (e.g. 2-3 trees [1]) are used to represent the paths and cycles. For the UHC algorithm we need (i), (iii), and (iv), and, in addition, the operation of list reversal. If "reversible 2-3 trees" are used then this set of operations still requires only $O(\log n)$ steps per operation [15]. We therefore conclude that:

Theorem 10.1 $\forall \alpha \quad \exists M \quad \exists c \quad \forall N \geq cn\log n$
$\forall s,t (1 \leq s,t \leq n)$ the probabilities that $DHC(D_{n,N},s,t)$ and $UHC(G_{n,N},s,t)$ return "success" within time bound $Mn\log^2 n$ on a r-RAC is in both cases at least $1 - O(n^{-\alpha})$.
Proposition 1 ensures that the theorem holds for the corresponding $D_{n,p}$ and $G_{n,p}$.

For the PM algorithm we represent the "partial matching" P as an n-vector $P[v]$ where $P[v] = w$ if $\{v,w\} \in P$ and $P[v] = 0$ otherwise. It is clear that after each call of SELECT the updating of P that must be performed in the remainder of step 3 can be done in constant time. Furthermore, since the values of ndp chosen by successive executions of step 2(b) have to be monotonically increasing, it is sufficient to remember the last such value and scan P forwards from that point at each execution. Hence the total time spent in step 2 is $O(n)$. We can therefore conclude the following, as well as its analogue for $G_{n,p}$.

Theorem 10.2 $\forall\alpha\ \exists M\ \exists c\ \forall N \geq cn\log n$ the probability that $PM(G_{n,N})$ returns "success" within time $Mn\log n$ on a r-RAC is at least $1 - O(n^{-\alpha})$.

Implementation on a RAC

It remains to analyse the complexity of the algorithms in the case that the random element of the RAC is replaced by a supply of random bits obtained from derandomization.

The derandomization procedure is implemented by searching the appropriate parts of the adjacency lists linearly. For successive segments of length $h = \lfloor 1/\hat{p} \rfloor$ of the integers we can thus count the number of edges that occur in the segment, identifying those which will generate $\lfloor \log_2 h \rfloor$ random bits according to the derandomization procedure. (Note that for $\hat{p} \in [(\log n)/n, \frac{1}{2}]$, $k\log_2 n \geq \lfloor \log_2 h \rfloor \geq 1$.) By applying Fact 4(ii) to bound the number of edges searched, it follows that with probability $1 - O(n^{-\alpha})$ the time spent in derandomization is $O(n\log n)$ if $p = O(n^{-\varepsilon})$ for some $\varepsilon > 0$, and $O(n\log^2 n)$ if $p = \frac{1}{2}$. By means of further applications of Propositions 1 and 2 we may conclude the following, as well as its analogue for $D_{n,p}$ and $G_{n,p}$:

Theorem 10.3 Theorem 10.1 holds for RACs.

For PM with the given derandomization procedure, the complexity of the latter will dominate for dense graphs. Hence

Theorem 10.4 Theorem 10.2 holds for RACs provided either that the additional restriction $N = O(n^{2-\varepsilon})$ is added or that the time bound is increased to $Mn\log^2 n$.

Finally we note that the algorithm for UHC that is obtained by reducing it via Proposition 3 to DHC also has a derandomized version with the same time bound. The original undirected graph is broken into two, and the connecting edges used as a random source both for generating the directed graphs (edge by edge, on demand) and for running DHC on them. It is clear that the source can supply, at constant cost, random bits not only of probability $\frac{1}{2}$ as already shown, but also of probability exactly p, as is required.

11. Conclusion

We have shown that in the particular context of Hamiltonian circuits and matchings in random graphs rigorous probabilistic analysis is possible for a family of algorithms that are related and not completely trivial. Since the proof techniques formalize some fairly intuitive notions, we hope that they will be of wider application.

Concerning the particular problems we have considered several lines of enquiry remain open: (i) It is possible that a rich variety of algorithms, other than the ones given, work also and can be proved to do so. (ii) It is likely that the algorithms can be made more efficient in practice by making various "heuristic" modifications. Some preliminary experiments suggest that it may be better not to delete edges as they are explored. (iii) Because of our imprecision about constant factors it is probable that the given algorithms perform much better than the analysis would suggest. There remains considerable scope for clarifying the relationships between the constant parameters by both analysis and experimentation.

References

1. A.V. Aho, J.E. Hopcroft, and J.D. Ullman, The Design and Analysis of Computer Algorithms, Addison-Wesley, Reading, Massachusetts (1974).
2. S.A. Cook, The complexity of theorem proving procedures, Proc. of 3rd Annual ACM Symposium on Theory of Computing, 151-158 (1971).
3. S.A. Cook and R.A. Reckhow, Time-bounded random access machines, JCSS 7, 354-375 (1973).
4. G.A. Dirac, Some theorems on abstract graphs, Proc.London Math.Soc. 2, 69-81 (1952).
5. J. Edmonds, Paths, trees, and flowers, Canad. J. Math. 17, 449-467 (1965).
6. P. Erdős and A. Rényi, On random graphs I, Publicationes Mathematicae 6, 290-297 (1959).
7. P. Erdős and A. Rényi, On the existence of a factor of degree one of connected random graphs, Acta Math.Acad.Sci.Hung. 17, 359-379 (1966).
8. P. Erdős and J. Spencer, Probabilistic Methods in Combinatorics, Academic Press, New York (1974).
9. S. Even and O. Kariv, An $O(n^{2.5})$ algorithm for maximum matching in general graphs, Proc. IEEE 16th Symp. on FOCS, 100-112 (1975).
10. W. Feller, An Introduction to Probability Theory and its Applications, Vol. I, Third Edition, John Wiley and Sons, Inc., New York (1968).
11. G.R. Grimmett and C.J.H. McDiarmid, On colouring random graphs, Math.Proc.Camb. Phil.Soc. 77. 313-324 (1975).
12. R.M. Karp, Reducibility among combinatorial problems, in Complexity of Computer Computations, R.E. Miller and J.W.Thatcher,eds., Plenum Press, New York, 85-104 (1972).
13. R.M. Karp, The probabilistic analysis of some combinatorial search algorithms, in Algorithms and Complexity, J.F. Traub, ed., Academic Press, New York (1976).
14. J. Komlós and E. Szemerédi, private communication.
15. G. Lev, Reversible concatenable lists, manuscript, Univ. of Edinburgh, Dept. of Comp.Sci.
16. L.A. Levin, Universal combinatorial problems (Russian), Problemi Peredaci Informacii, Vol. IX, 115-116 (1973).
17. L. Pósa, Hamiltonian circuits in random graphs, Discrete Mathematics 14, 359-364 (1976).
18. M.O. Rabin, Probabilistic algorithms, in Algorithms and Complexity, J.F. Traub, ed., Academic Press, New York, (1976).
19. C.P. Schnorr, Optimal algorithms for self-reducible problems, Proc. Third International Colloquium on Automata, Languages and Programming, S. Michaelson and R. Milner, eds., Edinburgh University Press, 322-337 (1976).