Classifying special interest groups in web graphs.

Colin Cooper*

July 9, 2002

Abstract

We consider the problem of classifying special interest groups in web graphs. There is a secret society of blue vertices which link preferentially to each other. The other vertices, which are red, are unaware of the distinction between vertex colours and link to vertices arbitrarily. Each new vertex directs m edges towards the existing graph on joining it. The colour of the vertices is unknown. We give an algorithm which **whp** classifies all blue vertices, and all red vertices of high degree correctly. We also give an upper bound for the number of mis-classified red vertices.

1 Introduction

1.1 Definition of the problem

We assume we have a graph which models the link structure of the world wide web. The vertices of this graph have a colour, either red or blue. The colour of a vertex is initially unknown to us. For example, this colour may indicate the response to some query, such as 'find all web pages which have property P'. Thus blue pages are those which have property P, and red pages are those which do not have this property.

In the *classification problem*, we attempt to determine which vertices are red and which vertices are blue. There are several versions of the classification problem. In some versions the colour can be decided by inspecting the page or the address of the page (eg. a vertex is blue if the url is `xxx.ac.uk`). In other cases a vertex would be blue if the page contained enough key words of the required type (eg. `graph, web, colour`).

*Department of Mathematical and Computing Sciences, Goldsmiths College, London SW14 6NW, UK. e-mail `c.cooper@gold.ac.uk`

The problem we consider here, is the problem of *secret societies*. The blue vertices form a special interest group, and link preferentially to each other. The red vertices are unaware of the colour distinction and can link to vertices of either colour. A blue vertex would never admit to being blue, ie. there is nothing on the page to allow a vertex colour to be discovered by inspection. The problem is to try to find the colours of the vertices using the structural properties of the graph.

Thus the problem is akin to the problem of finding a hidden vertex partition in a graph. Problem of this type have been studied in the context of standard random graphs (eg. $G_{n,p}$) by many authors. See for example [16] for a comprehensive discussion of this problem and an extensive bibliography. The analysis for web graphs differs from these cases, in that a web graph is not a standard random graph, web graphs are extremely sparse, continuously growing so that the partition is augmented at each step, and the edge probability of a vertex in a web graph is dependent on its age.

In a belief propagation model [8, 9] the perceived colour (or probability of colour) of each vertex is iteratively updated in a Bayesian manner, based on the opinions of neighbours. Such an algorithm requires an initial classification of vertex colours in which each node of the graph is (independently) assigned classification (perceived colour) based on some rule. This estimate is then improved on the basis of the classification of neighbours, next neighbours and so on, until some sort of stable value is (hopefully) arrived at.

The algorithm (CLASSIFY) which we describe here, is a simple two pass algorithm which processes the vertices sorted by decreasing degree. The algorithm first makes an initial classification (based on vertex degree). This is followed by a restricted one pass belief propagation, in which the classification of some vertices is updated based on the perceived colour of in-neighbours.

We claim that the algorithm can work well for classifying vertices of high degree (hub-authority vertices) and quite well for vertices of low degree. For the simple web-graph model on which we evaluate the algorithm, a more precise statement of performance is given in Theorem 1.

The performance of the algorithm depends on a parameter η_A, ($A =$ red , blue). η_A is a simple function of the probability p that a vertex is red. We assume that p is known, or could be estimated. The maximum in-degree of the red (resp. blue) vertices is about t^{η_R} (resp. t^{η_B}). As $p \to 1$, $\eta_R \to 1/2$ and $\eta_B \to 2/3$. This difference in the degree sequences of the red and blue vertices is exploited by the algorithm.

A web graph is a sparse connected graph designed to capture some properties of the www. Studies of the graph structure of the www were made by [4] and [7] among others. There are many models of web graphs designed to capture the structure of the www found in the studies given above. For example see references [1], [2], [3], [5], [6], [10], [11], [12], [14] and [15] for various models.

In the simple model we consider, each new vertex directs m edges towards existing vertices according to the degree of existing vertices (copy model). With probability p the new vertex takes the colour red, and with probability $q = 1 - p$ it takes the colour blue.

The new vertex v_t selects m vertices of $G(t-1)$ as neighbours, and directs edges from itself to these vertices. We require that the colour of these neighbours depends on the colour of v_t as follows: In the model we consider, the blue vertices are *biased*, so a new blue vertex always chooses existing blue vertices as neighbours. The red vertices are *arbitrary* in their choice of neighbours, in the sense that they do not discriminate on vertex colour, or are even perhaps unaware of the distinction. We model this arbitrariness by allowing a red vertex to choose a mix of red and blue vertices according to a binomial process, $Bin(m, p)$.

At each step the decision to add a red vertex is made by an independent Bernoulii $Be(p)$ process, so that the expected number of red vertices added in steps $1, ..., t$ is pt. Thus, provided $t \to \infty$, $|R(t)| \sim pt$, so that $p \sim |R(t)|/|V(t)|$, and p good approximation for the probability that a red vertex is selected when a neighbour vertex is chosen arbitrarily.

Once the number of red and blue neighbours of the new vertex has been decided, then the actual neighbour vertices are selected. Suppose, for example, that at step t, a blue vertex is to be selected as the i-th neighbour ($1 \le i \le m$) of v_t. Let $d(B, t-1)$ be the total degree in $G(t-1)$ of the set $B(t-1)$ of blue vertices existing at step $t-1$. Let $w \le t-1$ be a blue vertex of $G(t-1)$ of total degree $d(w, t-1)$, then

$$\mathbf{Pr}(v_t \text{ chooses } w) = \frac{d(w, t-1)}{d(B, t-1)}.$$

This sampling is repeated independently and with replacement for all edges directed out of v_t towards blue vertices. The sampling of red neighbours occurs similarly.

We suppose the initial graph $G(0)$ upon which $G(t)$ is built, is of constant size, is connected, and consists of some non-trivial mixture of red and blue vertices. We do not assume that the initial graph $G(0)$ is acyclic, but we require that all vertices of $G(0)$ have out-degree m.

Let $V(t)$ denote the vertex set of $G(t)$. Thus $V(t) = V(G(0)) \cup \{1, ..., t\}$. At time t, $G(t)$ has $\widehat{t} = t + |G(0)|$ vertices. For simplicity we assume $t \to \infty$ so that $\widehat{t}$ is well approximated by t.

We assume that we know the edges of $G(t)$, including their direction. The vertices are labelled in some arbitrary order which gives no information about their age. Each vertex v has exactly m out-edges, directed away from v, arising when v was added to G. Any other edges incident with v are directed towards v.

We suppose that do not know the colours of the vertices of $G(t)$. At first glance, there is not much about the structure of $G(t)$ which allows us to determine the colour of the vertices by inspecting the graph.

We describe a deterministic algorithm (algorithm CLASSIFY) which classifies the vertices of $G(t)$ into two sets $\boldsymbol{R}$, $\boldsymbol{B}$. The set $\boldsymbol{R}$ contains the vertices which we suppose to be red, and $\boldsymbol{B}$ those we suppose to be blue.

The algorithm classifies the vertices using the sorted degree sequence of the underlying graph.

The only other information the algorithm uses is the directed adjacency structure of the vertices.

Let $\eta_R = \frac{p}{(1+p)}$, $\eta_B = \frac{(1+p)}{(2+p)}$. Let $\alpha = \eta_R^2/(\eta_R + m(1-\eta_B)[(1-\epsilon)^2\eta_B - \eta_R])$ where $\epsilon > 0$ is a constant (eg $\epsilon = 0.01$). Let $d_0 = t^\alpha \ \log t$.
Let λ be the smallest non-negative solution of $\lambda = (q + \lambda p)^m$.

Theorem 1. **whp** *algorithm* CLASSIFY

i) *Correctly classifies all blue vertices.*

ii) *Correctly classifies all red vertices of degree at least* d_0.

iii) *Provided* $q > \lambda$, *correctly classifies all red vertices of degree at least* $K \log t$.

(iv) *Mis-classifies at most a limiting proportion* λ *of red vertices.*

The time complexity of the algorithm is dominated by sorting the vertices of $G(t)$ into decreasing degree sequence, and building an adjacency list structure for $G(t)$.

A red vertex which chooses m blue vertices (an event of probability q^m) can be considered to be re-coloured *purple*. Red vertices which choose a mixture of blue and purple vertices also become purple. A major problem for the algorithm will be to correctly classify purple vertices.

We make the convention that $\omega = \omega(t)$ is a generic function of t which tends slowly to infinity.

2 Algorithm CLASSIFY-I

At the end of algorithm CLASSIFY-I we claim that **whp**

(i) All blue vertices are correctly classified.

(ii) All red vertices of degree at least d_0 are correctly classified.

(iii) All non-purple (distinctly red) vertices are correctly classified.

begin-algorithm-CLASSIFY-I

Let $\boldsymbol{R}$ be the set of vertices classified as red and let $\boldsymbol{B}$ be the set of vertices classified as blue. Initially $\boldsymbol{R}, \boldsymbol{B}$ are empty.
Let $U = V \setminus (\boldsymbol{R} \cup \boldsymbol{B})$ denote the set of unclassified vertices. Initially $U = V$.
Any vertex which is added to $\boldsymbol{R} \cup \boldsymbol{B}$ by the algorithm is simultaneously removed from U.

A1 Put all vertices of degree at least $\Delta_R = (t^{\frac{p}{(1+p)}}\omega \log t)$ into $\boldsymbol{B}$.

A2 If any vertex $v \in \boldsymbol{B}$ has an edge pointing to any w in U, when w is classified blue and added to $\boldsymbol{B}$, until this no longer occurs.

A3 Let d be the maximum degree of U.
Let $\delta(d) = (d/(\omega \log t))^{(1-\epsilon)\eta_B/\eta_R}$.
Let $K(d) = \{u \in \boldsymbol{B} : d(u) \geq \delta(d)\}$.
Pick a vertex $v \in U$ of degree d.
Let $F^+(v)$ be the fan-out of v within U.
If $F^+(v)$ contains a directed cycle C, add the vertices of C to $\boldsymbol{R}$.

A4 For each $w \in F^+(v)$ we consider only the m edges pointing out of w.
Choose $w \in F^+(v)$ such that all out-edges of w point to $\boldsymbol{R} \cup \boldsymbol{B}$.
If any out-edge of any w points to $\boldsymbol{R}$ add w, to $\boldsymbol{R}$.
If any w points to $K(d)$ with all m out-edges, add w to $\boldsymbol{R}$.
Otherwise add w to $\boldsymbol{B}$.

A5 Repeat Steps A3-A4 until all vertices are classified.

end-algorithm-CLASSIFY-I

Analysis of Algorithm CLASSIFY-I

The main problem with analyzing the algorithm is that the analysis has to be made using the time when a vertex was born, whereas the only information we have about the vertices is their degree. The distribution of degree $d(s,t)$ of any vertex s in $G(t)$ is a function of its age and colour. The following facts are established in the Appendix.

Lemma 1. *Let $d_A(s,t)$ be the degree after step t of the vertex born at step s, then*

i) *The expected degree at step t of an A-coloured vertex ($A = B, R$) born at step s is*

$$\mu_A(s,t) = m\left(\frac{t}{s}\right)^{\eta_A}(1+o(1)),$$

where $\eta_R = \frac{p}{(1+p)}$, $\eta_B = \frac{(1+p)}{(2+p)}$. We note that $\eta_B > \eta_R$ for all $p \in [0,1]$.

ii)

$$\mathbf{Pr}(d(s,t) \leq a) \leq e^{-h\left(c\left(\frac{t}{s}\right)^{\eta(1-\epsilon)}-a\right)},$$

where $c, \epsilon > 0$ constant and $h = h(\epsilon)$, constant.

iii)

$$\mathbf{Pr}\left(d(s,t) \geq \left(\frac{t}{s}\right)^{\eta} \omega \log t\right) \leq t^{-K},$$

where $\omega \to \infty$ arbitrarily slowly.

Step A1 This initialization step classifies all vertices of degree at least $\Delta_R(t^{\frac{p}{(1+p)}} \omega \log t)$ as blue. This is correct **whp** , as by Lemma 1 no red vertex can have degree this large.

Step A2 This ensures that edges point from U to $\boldsymbol{B} \cup \boldsymbol{R}$. The algorithm maintains this at all future steps by processing the fan-out $F^+(v)$ of the selected vertex v.

Step A3 If $F^+(v)$ contains a directed cycle C then $C \subseteq G(0)$, as $G(t)$ is acyclic apart from (possibly) $G(0)$. Of necessity, all vertices of this cycle have the same colour, which must be red, as all the blue vertices of $G(0)$ have been moved into $\boldsymbol{B}$ during A1.

As the fan-out $F^+(v)$ is acyclic, we can always find a *fringe* vertex w with all m out-edges pointing towards classified vertices, that is $\mathrm{Out}(w) \cap F^+(v) = \emptyset$. We reduce the fan-out from the fringe inwards to v.

There is a technical point here. The fan out of v, $F^+(v)$, may contain vertices w of degree $d(w,t)$ less than d. However, the correct classification of these vertices is assured. If w is blue, the analysis for A4 assures us that w cannot point only to $K(d)$. If w is red, then w was born before v, and can only point to red vertices of $\boldsymbol{R}$ or blue vertices of $K(d)$.

Step A4 Given a red vertex v of degree d, the latest possible time for this vertex to have been born is $T = T_R(d)$, where

$$T_R(d) = t\left(\frac{\omega \log t}{d}\right)^{1/\eta_R}.$$

At time t the lowest possible degree of a blue vertex born at or before time T is

$$\delta(d) = c\left(\frac{t}{T}\right)^{\eta_B(1-\epsilon)} = c\left(\frac{d}{\omega \log t}\right)^{\eta_B(1-\epsilon)/\eta_R}.$$

The latest possible birth time of a blue vertex of degree $\delta(d)$ is $T' = T_B(\delta(d))$ where

$$T_B(\delta(d)) = t\left(\frac{\omega \log t}{\delta(d)}\right)^{1/\eta_B} = t\frac{(\omega \log t)^{1/\eta_B + (1-\epsilon)/\eta_R}}{c^{1/\eta_B} d^{(1-\epsilon)/\eta_R}}.$$

The birth time v of a blue vertex of degree d lies in the interval $I(d)$ where

$$I(d) = \left(t\left(\frac{c}{d}\right)^{1/((1-\epsilon)\eta_B)}, t\left(\frac{\omega \log t}{d}\right)^{1/\eta_B}\right).$$

Let $B(T')$ denote blue vertices born at or before T'. Given v is blue, let $\pi(v)$ be the probability that v only chooses its neighbours from $B(T')$.

At time T' the degree of $B(T')$ was $\theta_B T'(1+o(1))$, where $\theta_B = mq(2+p)$. At time v the degree of $B(T')$ is at most

$$d(B(T'), v) = \theta_B T' \left(\frac{v}{T'}\right) \omega \log v.$$

Thus

$$\pi(v) = (1+o(1)) \left(\frac{d(B(T'), v)}{\theta_B v}\right)^m \leq \left(\omega \log v \left(\frac{T'}{v}\right)^{1-\eta_B}\right)^m.$$

Now $d \geq c(t/v)^{(1-\epsilon)\eta_B}$, so

$$\pi(v) \leq \left((\omega \log t)^{\xi} \left(\frac{v}{t}\right)^{\lambda}\right)^m,$$

where $\xi = (1+(1-\eta_B)(1/\eta_B + 1/\eta_R)$ and $\lambda = (1-\eta_B)(1-\epsilon)^2\eta_B/\eta_R - 1)$.

We will choose v_0 such that $\sum_{v \leq v_0} \pi(v) = o(1)$. Now

$$\sum_{v \leq v_0} \pi(v) \leq \frac{(\omega \log t)^{\xi} v^{m\lambda+1}}{t^{m\lambda}}.$$

Let

$$v_0 = \left(\frac{o(1)t^{m\lambda}}{(\omega \log t)^{\xi}}\right)^{\frac{1}{1+m\lambda}}, \qquad d_0 = \log t \left(\frac{t}{v_0}\right)^{\eta_R},$$

then v_0, d_0 are respectively the birth time and degree of the last red vertex which is **whp** guaranteed correctly classified by CLASSIFY-I. After v_0, purple vertices may appear.

3 Algorithm-CLASSIFY-II

The purpose of algorithm-CLASSIFY-II is to re-classify those purple vertices of degree between d_0 and $K \log t$ which have been mis-classified as blue. We regard all vertices of degree d_0 as correctly classified by algorithm-CLASSIFY-I.

We give an upper bound on the proportion of mis-classified red vertices of degree at most $K \log t$.

Let $\zeta_B = 1/(1+p), \zeta_R = \lambda/(q+\lambda p)$ where λ is the smallest non-negative solution of $\lambda = (q+\lambda p)^m$. We assume $\zeta_B > \zeta_R$.

begin-algorithm-CLASSIFY**-II**
Initially all vertices of $\boldsymbol{B}$ of degree at most d_0 are *unprocessed.*
Pick an unprocessed vertex v of maximum degree d. Mark v as processed.
If $d_B^-(v) > \frac{1}{2}(\zeta_B + \zeta_R)(d(v) - m)$ return v to $\boldsymbol{B}$, else place v into $\boldsymbol{R}$.
Any vertex which points to a red vertex is classified red.
end-algorithm-CLASSIFY**-II**

Some care is needed in defining what we mean by a purple vertex in the context of the algorithm. It may well be that every red vertex of $G(0)$ points only to blue vertices, so that every red vertex of $G(t)$ is (in this sense) purple. However we are assured no red vertex of degree at least d_0 is purple. Thus a purple vertex is a red vertex classified into $\boldsymbol{B}$.

As an upper bound on the probability a vertex is purple, we can consider the extinction probability β of a binomial branching process $Bin(m, p)$. The process becomes extinct if all branches eventually have zero progeny (eventually only blue vertices are reached). The value of β is the smallest non-negative root of $g(x) = (q + px)^m - x$. The value of β is an upper bound on the limiting proportion $\lambda(t)$ of purple vertices in $G(t)$ for several reasons.

(i) $G(t)$ is finite so $\beta(t) \leq \beta$. This follows because extinction probabilities are monotone non-decreasing as the number of levels in the branching increases.

(ii) If the branching process hits a red vertex of degree at least d^*, it cannot become extinct.

(iii) Red neighbours are selected based on degree, so the process is biased towards red vertices of degree at least d^*.

We can imagine our branching process modified as follows. After the number, j say, of red descendants of vertex $t+1$ is sampled, there is a second sampling process, based on degree, to determine the labels of the descendants. If the descendant τ has degree $d(\tau) \geq d^*$ then the process is halted without extinction. When a label τ is selected then further branching can only occur to labels less than τ. The second sampling process occurs with replacement, so there is a possibility that a vertex may be hit more than once during a branching. However this is also biased towards vertices of larger degree.

Let $t \geq \tau$. Let $d_P(t)$ denote the total degree in $G(t)$ of purple vertices, then

$$\begin{aligned} d_P(t+1) &= d_P(t) + 1\{c(t+1) = R\} \times \\ & \sum \binom{m}{j} q^{m-j} p^j \left(m \left(\frac{d_P(t)}{d_R(t)} \right)^j + \sum \binom{j}{k} k \left(\frac{d_P(t)}{d_R(t)} \right)^k \left(1 - \frac{d_P(t)}{d_R(t)} \right)^{j-k} \right). \end{aligned}$$

Now $\mathbf{E}\, d_P(t) \geq 2mpq^m(t-\tau)$ and $\mathbf{E}\, d_P(t) < \mathbf{E}\, d_R(t)$ as there is a positive probability $(1-\beta)$ of a red vertex surviving in the worst case binomial branching. Thus $\mathbf{E}\, d_P(t) = c(t)t$ for $t \gg \tau$.

The values $d_P(t), d_R(t)$ are sharply concentrated within $O(\sqrt{t}\log t)$ of their expected values **whp** (see [10]), so for $1 \leq j \leq m$, $\mathbf{E}\,(d_P(t)/d_R(t))^j$ is well approximated by $f(t) = (\mathbf{E}\,d_P(t)/\mathbf{E}\,d_R(t))^j$. Thus

$$\theta f(t+1) \sim \theta f(t) + p \sum \binom{m}{j} q^{m-j} p^j (m(f(t))^j + jf(t)).$$

Noting that $\theta_R = mp(1+p)$ we obtain

$$(1+p)f(t+1) = (1+p)f(t) + (q + f(t)p)^m + pf(t).$$

If $f(t)$ ever increases to $f(t) = \lambda t$ where λ is the smallest non-negative root of

$$\lambda = (q + \lambda p)^m. \tag{1}$$

We should in principle be able to discriminate between a blue vertex v and a purple vertex w of the same degree d, based on the number of edges directed into v (resp. w) from $\boldsymbol{B}$ (vertices classified blue/purple) and from $\boldsymbol{R}$ (vertices classified red).

Lemma 2. *Let $\zeta_B = 1/(1+p)$, $\zeta_P = \lambda/(q+\lambda p)$. Provided $\zeta_B > \zeta_P$* **whp** *algorithm* CLASSIFY-*II can correctly re-classify all purple vertices in $\boldsymbol{B}$ of degree at least $K \log t$ into $\boldsymbol{R}$.*

Proof

Let $d_A^-(s,t)$ denote the in-degree after step t of an A-coloured vertex s, from A-coloured vertices τ, $(s < \tau \leq t)$.

$$\begin{aligned}
\mathbf{E}\; d_B^-(s,t) \mid d(s,t), c(s) = B &= \frac{1}{1+p}(d(s,t) - m)(1 + o(1)) \\
\mathbf{E}\; d_P^-(s,t) \mid d(s,t), c(s) = P &= \frac{\lambda}{q + \lambda p}(d(s,t) - m)(1 + o(1)).
\end{aligned}$$

The random variable $d_A^-(s,t)$ is the sum of $l = d(s,t) - m$ independent $\{0,1\}$-indicator variables.

We first consider the case where s is blue. What is the probability that s will be chosen by vertex τ?

$$\mathbf{Pr}(\tau \text{ chooses } s, c(\tau) = A \mid c(s) = B) = \begin{cases} pmq\frac{d(s,\tau)}{d_B(\tau)}(1 + o(1)) & c(\tau) = R \\ mq\frac{d(s,\tau)}{d_B(\tau)}(1 + o(1)) & c(\tau) = B \end{cases}.$$

Thus

$$\mathbf{Pr}(c(\tau) = B \mid \tau \text{ chooses } s, c(s) = B) = (1 + o(1))\frac{1}{1+p},$$

and

$$\mathbf{E}\, d^-_B(s,t) \mid d(s,t), c(s) = B = (1+o(1))\frac{1}{1+p}(d(s,t)-m).$$

We next consider the case where the colour of s is purple (P).

$$\mathbf{Pr}(\tau \text{ chooses } s \mid c(s) = P) = mp^2\frac{d(s,\tau)}{d_R(\tau)}(1+o(1)).$$

$$\begin{aligned}\mathbf{Pr}(\tau \text{ chooses } s,\ c(\tau) = P) &= p\sum \binom{m}{j} p^j \left(\frac{d_P(\tau)}{d_R(\tau)}\right)^{j-1} q^{m-j} j\frac{d(s,\tau)}{d_R(\tau)}(1+o(1)) \\ &= (1+o(1))mp^2(q+\lambda p)^{m-1}\frac{d(s,\tau)}{d_R(\tau)}.\end{aligned}$$

Thus

$$\mathbf{Pr}(c(\tau) = P \mid \tau \text{ chooses } s, c(s) = P) = (q+\lambda p)^{m-1}(1+o(1)) = \frac{\lambda}{q+\lambda p}(1+o(1)).$$

By the Hoeffding inequality [13]

$$\mathbf{Pr}(d^-_A(s,t) \notin [(1-\epsilon)\mathbf{E}\, d^-{}_A(s,t), (1+\epsilon)\mathbf{E}\, d^-{}_A(s,t)]) \le 2\exp\left(-\frac{\epsilon^2}{3}\zeta_A d(s,t)\right).$$

If v is an un-processed vertex of $\boldsymbol{B}$ of degree $d(v)$ the algorithm classifies v as blue if

$$d^-_B(v) > \frac{1}{2}(\zeta_B + \zeta_P)(d(v) - m),$$

which corresponds to the value of $\epsilon_B = \frac{1}{2}(\zeta_B - \zeta_P)/\zeta_B$, and $\epsilon_P = \frac{1}{2}(\zeta_B - \zeta_P)/\zeta_P$.

Thus the probability that a vertex of degree ω is mis-classified is $O(e^{-c\omega})$. Choosing $\omega = K\log t$ where K is a large constant, Theorem 1(ii) follows. □

References

[1] D. Achlioptas, A. Fiat, A.R. Karlin and F. McSherry, Web search via hub synthesis, *Proceedings of the 42nd Annual IEEE Symposium on Foundations of Computer Science* (2001) 500-509.

[2] M. Adler and M. Mitzenmacher, *Toward Compressing Web Graphs*, To appear in the 2001 Data Compression Conference.

[3] W. Aiello, F. Chung and L. Lu, Random evolution in massive graphs, *Proceedings of the 42nd Annual IEEE Symposium on Foundations of Computer Science* (2001) 510-519.

[4] R. Albert, A. Barabasi and H. Jeong. *Diameter of the world wide web.* Nature 401:103-131 (1999) see also `http://xxx.lanl.gov/abs/cond-mat/9907038`

[5] B. Bollobás, O. Riordan and J. Spencer, *The degree sequence of a scale free random graph process*, to appear.

[6] B. Bollobás and O. Riordan, *The diameter of a scale free random graph*, to appear.

[7] A. Broder, R. Kumar, F.Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins and J. Wiener. *Graph structure in the web.* `http://gatekeeper.dec.com/pub/DEC/SRC /publications/stata/www9.htm`

[8] A. Broder, R. Krauthgamer and M. Mitzenmacher. *Improved classification via connectivity information.*

[9] S. Chakrabarti, B. Dom and P. Indyk. *Enhanced hypertext categorization using hyperlinks.* Proceedings of ACM SIGMOD Conference on management of Data (1998) p307-318.

[10] C. Cooper and A.M. Frieze, A general model of web graphs, *Proceedings of ESA 2001*, 500-511.

[11] E. Drinea, M. Enachescu and M. Mitzenmacher, *Variations on random graph models for the web.*

[12] M.R. Henzinger, A. Heydon, M. Mitzenmacher and M. Najork, Measuring Index Quality Using Random Walks on the Web, *WWW8 / Computer Networks* 31 (1999) 1291-1303.

[13] W. Hoeffding, Probability inequalities for sums of bounded random variables, *Journal of the American Statistical Association* 58 (1963) 13-30.

[14] R. Kumar, P. Raghavan, S. Rajagopalan, D. Sivakumar, A. Tomkins and E. Upfal. *The web as a graph.* `www.almaden.ibm.com`

[15] R. Kumar, P. Raghavan, S. Rajagopalan, D. Sivakumar, A. Tomkins and E. Upfal. *Stochastic models for the web graph.* `www.almaden.ibm.com`

[16] F. McSherry. *Spectral partitioning of random graphs* FOCS 2001

4 Appendix: Distribution of vertex degree

Let A can mean either red or blue, and B is the opposite colour (local change of notation). Let $N(t)$ be the number of edges directed out of vertex t which are incident with A-coloured vertices other than t. Let p_A be the probability that vertex t is A-coloured, and let $q_A = 1-p_A$ be the probability that vertex t is B-coloured. Let $\widehat{p}$ be the probability of an A-coloured vertex choosing an A-coloured neighbour, and $\widetilde{q}$ be the probability of a B-coloured vertex choosing an A-coloured neighbour. The distribution of $N(t)$ is given by

$$N(t) \sim 1_A Bin(m, \widehat{p}) + 1_B Bin(m, \widetilde{q}), \tag{2}$$

where $Bin(m, p)$ is a Binomial random variable with parameters m, p and $1_A = 1\{c(t) = A\}$ is the indicator for the event that vertex t is A-coloured.

In the notation of the main body of this paper, when A denotes red, then $p_A = p,\ q_A = q = 1-p,\ \widehat{p} = p,\ \widetilde{q} = 0$, and when A denotes blue, then $p_A = q,\ \widehat{p} = 1,\ \widetilde{q} = q$.

Let $d(A, t)$ be the total degree of the A-coloured vertices of $G(t)$. Then

$$\mathbf{E}\, d(A, t) = t(pm + pm\widehat{p} + qm\widetilde{q}) = \theta_A t,$$

say. Thus in red, blue notation $\theta_R = mp(1 + p)$ and $\theta_B = mq(2 + p)$.

The value of $d(A, t)$ is the sum of t independent $\{0, ..., 2m\}$-random variables and is sharply concentrated in t. The inequality

$$d(A, t) = \theta_A t \left(1+O\left(\sqrt{\tfrac{K \log t}{\theta t}}\right)\right) \tag{3}$$

holds simultaneously with probability $1 - {t_0}^{-K+1}$ for all $t \geq t_0$ and $t_0 > 0$.

Let $X_t = X_t(v)$ denote $d(v, t)$ the degree of vertex v at time t.

Expected value of vertex degree.
Let $\eta_A = m(p_A\widehat{p} + q_A\widetilde{q})/\theta_A$. Let v be A-coloured. For any $v \leq \tau \leq t$ let $d(v, \tau)$ be given, then **whp**

$$\mathbf{E}\, d(v, t) = d(v, \tau) \left(\frac{t}{\tau}\right)^{\eta_A} \left(1+O\left(\sqrt{\tfrac{\log \tau}{\tau}}\right)\right). \tag{4}$$

From (2) $X_{t+1} = X_t + Y_{t+1}$ where

$$Y_{t+1} = \sum_j Bin\left(j, \frac{X_t}{d(A, t)}\right) \left(1_A \tbinom{m}{j}\, \widehat{p}^j\, \widehat{q}^{m-j} + 1_B \tbinom{m}{j}\, \widetilde{q}^j\, \widetilde{p}^{m-j}\right).$$

Thus

$$\mathbf{E}\, X_{t+1} = X_t \left(1 + \frac{m(p\widehat{p} + q\widetilde{q})}{d(A, t)}\right).$$

The result (4) follows from iterating the expectation of this identity and applying (3).

Lower bound on vertex degree.
Let $X_t = X_t(s)$. The lower bound will follow from

$$\mathbf{Pr}(X_t \le a) \le e^{ha}\mathbf{E}\left(e^{-hX_t}\right).$$

As before, let $X_t = X_{t-1} + Y_t$. Let $Z_t = Z_{t-1} + U_t$ where U_t is a Bernoulii random variable $Be(\eta' Z_{t-1}/(t-1))$ and $\eta' = \eta_t = \eta(1 - O(\sqrt{\log t/t}))$. Thus Y_t is a Binomial random variable with the same parameter as U_t. Hence X_t stochastically dominates Z_t, and thus

$$\mathbf{E}\left(e^{-hX_t}\right) \le \mathbf{E}\left(e^{-hZ_t}\right).$$

We now prove that

$$\mathbf{E}\left(e^{-hZ_t}\right) \le e^{-hc_s m\left(\frac{t}{s}\right)^{(1-\epsilon)\eta}},$$

where $c_s \ge 1$ is a constant depending on s and $\epsilon = \epsilon(h)$ is such that $1 - e^{-h} \ge (1-\epsilon)h$.

$$\begin{aligned}
\mathbf{E}\left(e^{-hU_{t+1}} \mid Z_t\right) &= 1 - \frac{\eta' Z_t}{t}(1 - e^{-h}) \\
&\le 1 - \frac{\eta' Z_t}{t}(1-\epsilon)h \\
&\le \exp\left(-hZ_t\frac{\eta'(1-\epsilon)}{t}\right).
\end{aligned}$$

Thus

$$\begin{aligned}
\mathbf{E}\left(e^{-hZ_t}\right) &\le \mathbf{E}\exp\left(-h\left(1 + \tfrac{\eta'(1-\epsilon)}{t-1}\right)Z_{t-1}\right) \\
&\le \exp\left(-hm\prod_{\tau=s\ldots t-1}\left(1 + \tfrac{\eta'(1-\epsilon)}{\tau-1}\right)\right) \\
&\le \exp\left(-hm\left(\frac{t}{s}\right)^{(1-\epsilon)\eta}\left(1+O\left(\sqrt{\tfrac{\log s}{s}}\right)\right)\right).
\end{aligned}$$

The result follows on choosing $c_s = \left(1+O\left(\sqrt{\frac{\log s}{s}}\right)\right)$.

Upper bound on vertex degree.
This will follow from

$$\mathbf{Pr}(X_t \ge a) \le e^{-ha}\mathbf{E}\left(e^{hX_t}\right),$$

on choosing

$$h = \frac{1}{\omega}\left(\frac{s}{t}\right)^{\eta} \qquad\qquad a = \left(\frac{t}{s}\right)^{\eta}\omega K \log t.$$

Let $\lambda = X_t/d(A,t)$ then from (2)

$$\mathbf{E}\ \left(e^{hY_{t+1}} \mid X_t\right) = p(1+\widehat{p}\lambda(e^h-1))^m + q(1+\widetilde{q}\lambda(e^h-1))^m.$$

Now, for $h \leq 1$, $e^h - 1 \leq h + h^2$ and $\lambda h = O(h)$ as $h \to 0$. Thus

$$\begin{aligned}
\mathbf{E}\ \left(e^{hY_{t+1}} \mid X_t\right) &\leq 1 + m(p\widehat{p} + q\widetilde{q})\lambda(h+h^2) + O((\lambda h)^2) \\
&= 1 + hX_t\frac{\eta}{t}(1+O(h))\left(1 + O(\sqrt{(\log t)/t})\right).
\end{aligned}$$

Thus

$$\mathbf{E}\ e^{h_tX_t} \leq \mathbf{E}\ e^{h_{t-1}X_{t-1}},$$

where

$$h_{t-1} = h_t\left(1 + \frac{\eta}{t-1}(1+O(h_t))\left(1 + O\left(\sqrt{\tfrac{\log t-1}{t-1}}\right)\right)\right).$$

Given $h = (1/\omega)(s/t)^\eta$ we claim that, for $\tau \leq t$

$$h_\tau = c_\tau h\left(\frac{t}{\tau}\right)^\eta, \qquad c_\tau = \Omega(1).$$

We note first that

$$\begin{aligned}
h_{\tau-1} &= c_\tau h\left(\frac{t}{\tau}\right)^\eta\left(1 + \tfrac{\eta}{\tau-1} + \tfrac{O(h_\tau)}{\tau-1} + \tfrac{O(\log\tau-1)^{1/2}}{(\tau-1)^{3/2}}\right) \\
&= h\left(\frac{t}{\tau-1}\right)^\eta c_\tau\left(1 + \tfrac{\eta}{\tau(\tau-1)} + \tfrac{O(1)}{\tau^2} + \tfrac{O(h_\tau)}{\tau-1} + \tfrac{O(\log\tau-1)^{1/2}}{(\tau-1)^{3/2}}\right),
\end{aligned}$$

where the second line follows from expanding $(\tau-1)/\tau)^\eta$ and multiplying out.

Now, certainly $c_t = 1$ and suppose $c_t, ..., c_\tau$ are $\Omega(1)$. Let $c^* = \max(c_t, ..., c_\tau)$.

$$\begin{aligned}
c_{\tau-1} &= \prod_{r=\tau\ldots t}\left(1 + \frac{O(1)}{r^2} + \frac{O(c^*ht^\eta)}{r^{1+\eta}} + \frac{O(\log r)^{1/2}}{r^{3/2}}\right) \\
&\leq \exp\left(O(1)\left(\sum\frac{1}{r^2} + hc^*\left(\frac{t}{\tau}\right)^\eta + \frac{O(1)\sqrt{\log\tau}}{\sqrt{\tau}}\right)\right) \\
&\leq \exp\left(O\left(1 + \sqrt{\tfrac{\log s}{s}}\right)\right).
\end{aligned}$$