Supporting Text

PageRank Definition and Computation. Search engines use topological prestige measures to express the quality of pages, so that they can sort search results according to their importance along with various query relevance factors. One possibility is to weigh a web page according to the number of other pages that point to it, i.e., its in-degree. This represents the easiest option: the in-degree is a local quantity whose value can be updated with the least computational effort. On the other hand, just because the in-degree is a local criterion, it does not take into account the correlation between the quality of pages which point to one another. Two pages with the same in-degree would have the same prestige no matter how "good" the pages that point to them.

The prestige measure used by Google, called PageRank, takes into account this important aspect in the quality evaluation. The question is shifted from a local to a global perspective. One does not want to know how easy it is to reach a page from its neighbourhood, but rather how easy it is to reach the page if one randomly crawls through the Web. Google basically simulates a random walk through the (directed) links of the web graph. A simple random walk would have however two drawbacks:

- Pages with no incoming links would have a zero asymptotic probability to be visited.
- The walkers would concentrate in the sites without outgoing links (dangling links).

To eliminate these drawbacks, the inventors of Google introduced a probability q for the walker to jump at any time step from the page it is sitting on to a random page of the Web. PageRank is, therefore, the stationary probability of a mixed process, that consists of a random walk and a random scattering from a generic site to any other. Let $p(i)$ be the PageRank of the web page i. The vector p satisfies the following self-consistent system of relations:

$$p(i) = \frac{q}{N} + (1-q)\sum_{j=1}^{r} p(i_j)/c(i_j) \qquad i = 1, 2, \ldots, N \quad \mathbf{(9)}$$

where N is the total number of pages, $i_1, i_2, \ldots, i_r$ are the pages that point to i, and $c(j)$ the outdegree of page j. This is equivalent to solving the eigenvalue problem for the transition matrix $\mathcal{M}$, whose element $\mathcal{M}_{ij}$ is given by the following expression:

$$\mathcal{M}_{ij} = \frac{q}{N} + (1-q)\frac{1}{c(j)}A_{ji}, \qquad (\mathbf{10})$$

where A is the adjacency matrix of the web graph ($A_{ji} = 1$ if there is a link from j to i, otherwise $A_{ji} = 0$). In fact, p is just the principal eigenvector of $\mathcal{M}$. If $c(j) = 0$, the second term on the *rhs* of Eq. **10** would have no meaning. In this special case (dangling link), we drop the term for all $i \neq j$, and we add the probability $1-q$ when $i = j$. This is the most natural way to proceed: if the crawler reaches a dangling link j, it cannot reach any other site by following links, so it will stay trapped in j with probability $1-q$, or it will leap to a random page (possibly j itself) with probability q.

The stationary probability of the process described by $\mathcal{M}$ is given by its principal eigenvector. Its calculation is a standard problem of numerical analysis and can be achieved by repeteadly applying the matrix $\mathcal{M}$ to a generic vector p_0 not orthogonal to p. It is easy to show that $1 = \lambda_0 \geq \lambda_1 \geq ... \geq \lambda_N$ (λ's being the eigenvalues of $\mathcal{M}$) , and therefore $\lim_{n\to\infty} \mathcal{M}^n p_0 = p$.

In practical applications, it turns out that ≈ 100 iterations suffice to calculate the PageRank of a network with $10^7 - 10^8$ vertices.

The Vicious Cycle. To understand the potential danger of the popularity bias introduced by search engines, let us envision a scenario in which people search for information about the *minollo* (an imaginary animal). Imagine that there is an established site `minollo-recipes.com` about the minollo and its culinary qualities. Further imagine a newly developed site `save-the-minollo.org` holding the view that the minollo is an endangered species and it should no longer be hunted. Now, suppose a student is assigned the homework of creating a web page with a report on the minollo. The student will submit the query “minollo” to a search engine and browse the top 10 hits. Let’s say that `minollo-recipes.com` is the fifth hit and `save-the-minollo.org` is ranked 15th. The student will read the established site and write her report on minollo recipes. She will not read about the possible endangered status of the minollo. She will also diligently cite her source by adding a link from her new page to `minollo-recipes.com`.

As a result of this process, the more established site will have acquired a new link and increased its popularity (as measured by PageRank). The next time someone searches about the minollo, it will be more likely that the established site will be ranked even higher — fourth, say. The visibility of the less established site, on the other hand, will not increase. Our example is a practical representation of the vicious cycle illustrated in Fig. 7.

Hit List Size Distribution. We obtained the hit list size distribution from a log of 200,000 actual queries submitted to AltaVista in 2001 (Fig.5B). The data can be reasonably well fitted by a power law with an exponential cutoff due to the finite size of the AltaVista index. The exponent of the power law is $\delta \approx 1.1$. In our Monte Carlo simulations we neglected the exponential cutoff, and used the simple power law

$$S(h,N) = B(N)h^{-\delta} \qquad \textbf{(11)}$$

where the normalization constant $B(N)$ is just a function of N. The cutoff would affect only the part of the distribution $S(h,N)$ corresponding to the largest values of h, influencing a limited portion of the curve $t_S(R,N)$ and the click probability of the very top pages (compare with the scaling relation of Eq. **7**). As there are no real queries that return hit lists containing all pages,[*] we have that $h_M < 1$. To estimate h_M we divided the largest observed number of Google hits in our collection of AltaVista queries ($\approx 6.6 \times 10^8$) by the total number of pages reportedly indexed by Google ($\approx 8 \times 10^9$ at data collection time), yielding $h_M \approx 0.1$. The top-ranked $1/h_M \approx 10$ sites will have the same probability to be clicked.

[*] The policy of all search engines is to display at most 1,000 hits, and we took this into account in our simulations. This does not mean that $h \leq 1000/N$; the search engine scans all its database and can report millions of hits, but it will finally display only the top 1,000.

We then expect a flattening of the portion of $t_S(R,N)$ corresponding to the pages with the highest PageRank/in-degree. This flattening seems consistent with the pattern observed in the real data (Fig. 4C).

Scaling Behavior of Click Probability. We start from the *ansatz* of Eq. **7** for the function $t(R,N,h)$ and the power law form of the distribution $S(h,N)$ (Eq. **11**). If we perform the convolution of Eq. **8**, we have

$$t_S(R,N) = \int_{1/N}^{h_M} S(h,N)h\,A(N)F(Rh)dh, \qquad (\mathbf{12})$$

where we explicitly set $h_m = 1/N$ and $F(Rh)$ is the universal function of Eq. **7**. By plugging the explicit expression of $S(h,N)$ from Eq. **11** into Eq. **12** and performing the change of variable $z = hN$ within the integral we obtain

$$t_S(R,N) = \frac{A(N)B(N)}{N^{2-\delta}} \int_{1}^{h_M\,N} z^{1-\delta}\,F\left(\frac{R}{N}\,z\right)dz. \qquad (\mathbf{13})$$

The upper integration limit can be safely set to infinity because $h_M\,N$ is very large. The integral in Eq. **13** thus becomes a function of the ratio R/N. The additional explicit dependence on N, expressed by the term outside the integral, consists in a simple multiplicative factor $f(N)$ that does not affect the shape of the curve (compare with Fig. 8).

We finally remark that $t_S(R,N)$ represents the relation between the click probability and the global rank of a page as determined by the value of its PageRank. For a comparison with the empirical data of Fig. 4C we need a relation between click probability and in-degree. We can relate rank to in-degree by means of Eq. **1** between rank and PageRank and by exploiting the proportionality between PageRank and in-degree discussed in the main text.