3

PageRank: The Technology That Launched Google

> The Star Trek computer doesn't seem that interesting. They ask it random questions, it thinks for a while. I think we can do better than that.
>
> —LARRY PAGE (Google cofounder)

Architecturally speaking, the garage is typically a humble entity. But in Silicon Valley, garages have a special entrepreneurial significance: many of the great Silicon Valley technology companies were born, or at least incubated, in a garage. This is not a trend that began in the dot-com boom of the 1990s. Over 50 years earlier—in 1939, with the world economy still reeling from the Great Depression—Hewlett-Packard got underway in Dave Hewlett's garage in Palo Alto, California. Several decades after that, in 1976, Steve Jobs and Steve Wozniak operated out of Jobs' garage in Los Altos, California, after founding their now-legendary Apple computer company. (Although popular lore has it that Apple was founded in the garage, Jobs and Wozniak actually worked out of a bedroom at first. They soon ran out of space and moved into the garage.) But perhaps even more remarkable than the HP and Apple success stories is the launch of a search engine called Google, which operated out of a garage in Menlo Park, California, when first incorporated as a company in September 1998.

By that time, Google had in fact already been running its web search service for well over a year—initially from servers at Stanford University, where both of the cofounders were Ph.D. students. It wasn't until the bandwidth requirements of the increasingly popular service became too much for Stanford that the two students, Larry Page and Sergey Brin, moved the operation into the now-famous Menlo Park garage. They must have been doing something right,

because only three months after its legal incorporation as a company, Google was named by *PC Magazine* as one of the top 100 websites for 1998.

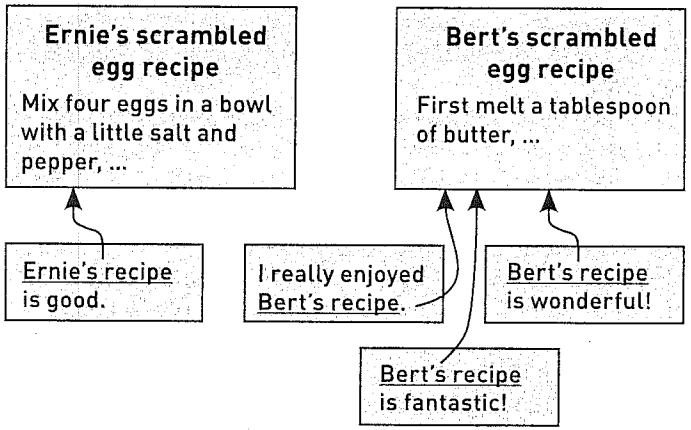
And here is where our story really begins: in the words of *PC Magazine*, Google's elite status was awarded for its "uncanny knack for returning extremely relevant results." You may recall from the last chapter that the first commercial search engines had been launched four years earlier, in 1994. How could the garage-bound Google overcome this phenomenal four-year deficit, leapfrogging the already-popular Lycos and AltaVista in terms of search quality? There is no simple answer to this question. But one of the most important factors, especially in those early days, was the innovative algorithm used by Google for ranking its search results: an algorithm known as *PageRank*.

The name "PageRank" is a pun: it's an algorithm that ranks web pages, but it's also the ranking algorithm of Larry Page, its chief inventor. Page and Brin published the algorithm in 1998, in an academic conference paper, "The Anatomy of a Large-scale Hypertextual Web Search Engine." As its title suggests, this paper does much more than describe PageRank. It is, in fact, a complete description of the Google system as it existed in 1998. But buried in the technical details of the system is a description of what may well be the first algorithmic gem to emerge in the 21st century: the PageRank algorithm. In this chapter, we'll explore how and why this algorithm is able to find needles in haystacks, consistently delivering the most relevant results as the top hits to a search query.

THE HYPERLINK TRICK

You probably already know what a hyperlink is: it is a phrase on a web page that takes you to another web page when you click on it. Most web browsers display hyperlinks underlined in blue so that they stand out easily.

Hyperlinks are a surprisingly old idea. In 1945 — around the same time that electronic computers themselves were first being developed — the American engineer Vannevar Bush published a visionary essay entitled "As We May Think." In this wide-ranging essay, Bush described a slew of potential new technologies, including a machine he called the *memex*. A memex would store documents and automatically index them, but it would also do much more. It would allow "associative indexing, ... whereby any item may be caused at will to select immediately and automatically another"—in other words, a rudimentary form of hyperlink!



The basis of the hyperlink trick. Six web pages are shown, each represented by a box. Two of the pages are scrambled egg recipes, and the other four are pages that have hyperlinks to these recipes. The hyperlink trick ranks Bert's page above Ernie's, because Bert has three incoming links and Ernie only has one.

Hyperlinks have come along way since 1945. They are one of the most important tools used by search engines to perform ranking, and they are fundamental to Google's PageRank technology, which we'll now begin to explore in earnest.

The first step in understanding PageRank is a simple idea we'll call the *hyperlink trick*. This trick is most easily explained by an example. Suppose you are interested in learning how to make scrambled eggs and you do a web search on that topic. Now any real web search on scrambled eggs turns up millions of hits, but to keep things really simple, let's imagine that only two pages come up—one called "Ernie's scrambled egg recipe" and the other called "Bert's scrambled egg recipe." These are shown in the figure above, together with some other web pages that have hyperlinks to either Bert's recipe or Ernie's. To keep things simple (again), let's imagine that the four pages shown are the *only* pages on the entire web that link to either of our two scrambled egg recipes. The hyperlinks are shown as underlined text, with arrows to show where the link goes to.

The question is, which of the two hits should be ranked higher, Bert or Ernie? As humans, it's not much trouble for us to read the pages that link to the two recipes and make a judgment call. It seems that both of the recipes are reasonable, but people are much more enthusiastic about Bert's recipe than Ernie's. So in the absence of any other information, it probably makes more sense to rank Bert above Ernie.

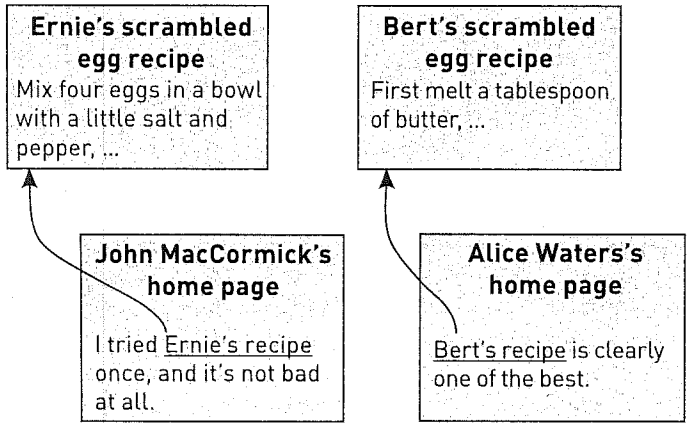
Unfortunately, computers are not good at understanding what a web page actually means, so it is not feasible for a search engine to examine the four pages linking to the hits and make an assessment of how strongly each recipe is recommended. On the other hand, computers are excellent at counting things. So one simple approach is to simply *count* the number of pages that link to each of the recipes—in this case, one for Ernie, and three for Bert—and rank the recipes according to how many incoming links they have. Of course, this approach is not nearly as accurate as having a human read all the pages and determine a ranking manually, but it is nevertheless a useful technique. It turns out that, if you have no other information, the number of incoming links that a web page has can be a helpful indicator of how useful, or "authoritative," the page is likely to be. In this case, the score is Bert 3, Ernie 1, so Bert's page gets ranked above Ernie's when the search engine's results are presented to the user.

You can probably already see some problems with this "hyperlink trick" for ranking. One obvious issue is that sometimes links are used to indicate *bad* pages rather than good ones. For example, imagine a web page that linked to Ernie's recipe by saying, "I tried Ernie's recipe, and it was awful." Links like this one, that criticize a page rather than recommend it, do indeed cause the hyperlink trick to rank pages more highly than they deserve. But it turns out that, in practice, hyperlinks are more often recommendations than criticisms, so the hyperlink trick remains useful despite this obvious flaw.

THE AUTHORITY TRICK

You may already be wondering why all the incoming links to a page should be treated equally. Surely a recommendation from an expert is worth more than one from a novice? To understand this in detail, we will stick with the scrambled eggs example from before, but with a different set of incoming links. The figure on the following page shows the new setup: Bert and Ernie each now have the same number of incoming links (just one), but Ernie's incoming link is from my own home page, whereas Bert's is from the famous chef Alice Waters.

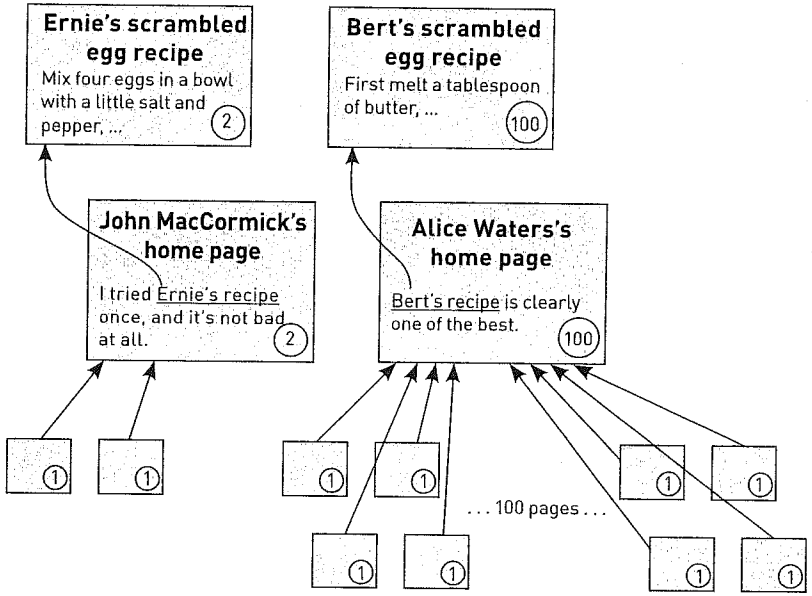
If you had no other information, whose recipe would you prefer? Obviously, it's better to choose the one recommended by a famous chef, rather than the one recommended by the author of a book about computer science. This basic principle is what we'll call the "authority trick": links from pages with high "authority" should result in a higher ranking than links from pages with low authority.



The basis for the authority trick. Four web pages are shown: two scrambled egg recipes and two pages that link to the recipes. One of the links is from the author of this book (who is *not* a famous chef) and one is from the home page of the famous chef Alice Waters. The authority trick ranks Bert's page above Ernie's, because Bert's incoming link has greater "authority" than Ernie's.

This principle is all well and good, but in its present form it is useless to search engines. How can a computer automatically determine that Alice Waters is a greater authority on scrambled eggs than me? Here is an idea that might help: let's combine the hyperlink trick with the authority trick. All pages start off with an authority score of 1, but if a page has some incoming links, its authority is calculated by adding up the authority of all the pages that point to it. In other words, if pages X and Y link to page Z, then the authority of Z is just the authority of X plus the authority of Y.

The figure on the next page gives a detailed example, calculating authority scores for the two scrambled egg recipes. The final scores are shown in circles. There are two pages that link to my home page; these pages have no incoming links themselves, so they get scores of 1. My home page gets the total score of all its incoming links, which adds up to 2. Alice Waters's home page has 100 incoming links that each have a score of 1, so she gets a score of 100. Ernie's recipe has only one incoming link, but it is from a page with a score of 2, so by adding up all the incoming scores (in this case there is only one number to add), Ernie gets a score of 2. Bert's recipe also has only one incoming link, valued at 100, so Bert's final score is 100. And because 100 is greater than 2, Bert's page gets ranked above Ernie's.



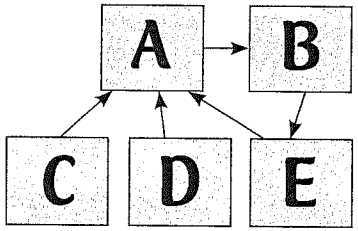
A simple calculation of "authority scores" for the two scrambled egg recipes. The authority scores are shown in circles.

THE RANDOM SURFER TRICK

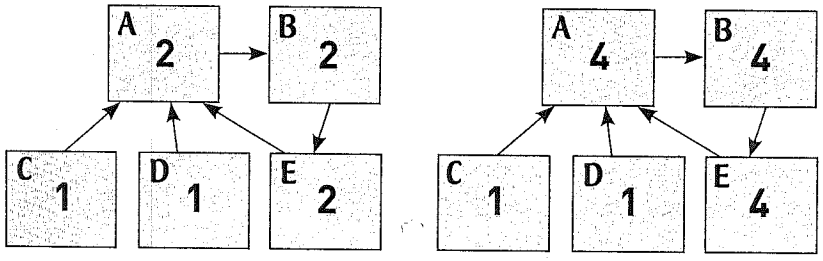
It seems like we have hit on a strategy for automatically calculating authority scores that really works, without any need for a computer to actually understand the content of a page. Unfortunately, there can be a major problem with the approach. It is quite possible for hyperlinks to form what computer scientists call a "cycle." A cycle exists if you can get back to your starting point just by clicking on hyperlinks.

The figure on the following page gives an example. There are 5 web pages labeled A, B, C, D, and E. If we start at A, we can click through from A to B, and then from B to E—and from E we can click through to A, which is where we started. This means that A, B, and E form a cycle.

It turns out that our current definition of "authority score" (combining the hyperlink trick and the authority trick) gets into big trouble whenever there is a cycle. Let's see what happens on this particular example. Pages C and D have no incoming links, so they get a score of 1. C and D both link to A, so A gets the sum of C and D, which is $1 + 1 = 2$. Then B gets the score 2 from A, and E gets 2 from B. (The situation so far is summarized by the left-hand panel of the



An example of a cycle of hyperlinks. Pages *A*, *B*, and *E* form a cycle because you can start at *A*, click through to *B*, then *E*, and then return to your starting point at *A*.

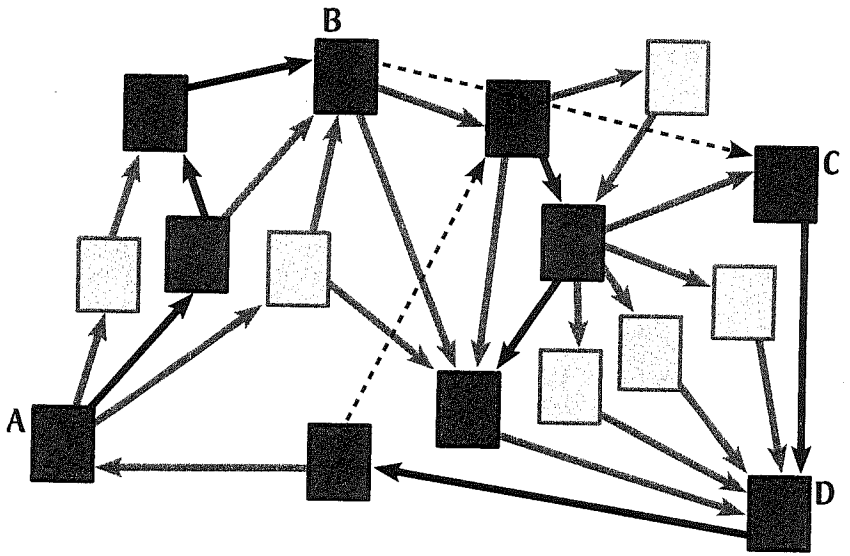


The problem caused by cycles. *A*, *B*, and *E* are always out of date, and their scores keep growing forever.

figure above.) But now *A* is out of date: it still gets 1 each from *C* and *D*, but it also gets 2 from *E*, for a total of 4. But now *B* is out of date: it gets 4 from *A*. But then *E* needs updating, so it gets 4 from *B*. (Now we are at the right-hand panel of the figure above.) And so on: now *A* is 6, so *B* is 6, so *E* is 6, so *A* is 8, ... You get the idea, right? We have to go on forever with the scores always increasing as we go round the cycle.

Calculating authority scores this way creates a chicken-and-egg problem. If we knew the true authority score for *A*, we could compute the authority scores for *B* and *E*. And if we knew the true scores for *B* and *E*, we could compute the score for *A*. But because each depends on the other, it seems as if this would be impossible.

Fortunately, we can solve this chicken-and-egg problem using a technique we'll call the *random surfer trick*. Beware: the initial description of the random surfer trick bears no resemblance to the hyperlink and authority tricks discussed so far. Once we've covered the basic mechanics of the random surfer trick, we'll do some analysis to uncover its remarkable properties: it combines the desirable features of the hyperlink and authority tricks, but works even when cycles of hyperlinks are present.



The random surfer model. Pages visited by the surfer are darkly shaded, and the dashed arrows represent random restarts. The trail starts at page *A* and follows randomly selected hyperlinks interrupted by two random restarts.

The trick begins by imagining a person who is randomly surfing the internet. To be specific, our surfer starts off at a single web page selected at random from the entire World Wide Web. The surfer then examines all the hyperlinks on the page, picks one of them at random, and clicks on it. The new page is then examined and one of its hyperlinks is chosen at random. This process continues, with each new page being selected randomly by clicking a hyperlink on the previous page. The figure above shows an example, in which we imagine that the entire World Wide Web consists of just 16 web pages. Boxes represent the web pages, and arrows represent hyperlinks between the pages. Four of the pages are labeled for easy reference later. Web pages visited by the surfer are darkly shaded, hyperlinks clicked by the surfer are colored black, and the dashed arrows represent random restarts, which are described next.

There is one twist in the process: every time a page is visited, there is some fixed *restart probability* (say, 15%) that the surfer does *not* click on one of the available hyperlinks. Instead, he or she restarts the procedure by picking another page randomly from the whole web. It might help to think of the surfer having a 15% chance of getting bored by any given page, causing him or her to follow a new chain of links instead. To see some examples, take a closer look at the figure above. This particular surfer started at page *A* and followed three

random hyperlinks before getting bored by page *B* and restarting on page *C*. Two more random hyperlinks were followed before the next restart. (By the way, all the random surfer examples in this chapter use a restart probability of 15%, which is the same value used by Google cofounders Page and Brin in the original paper describing their search engine prototype.)

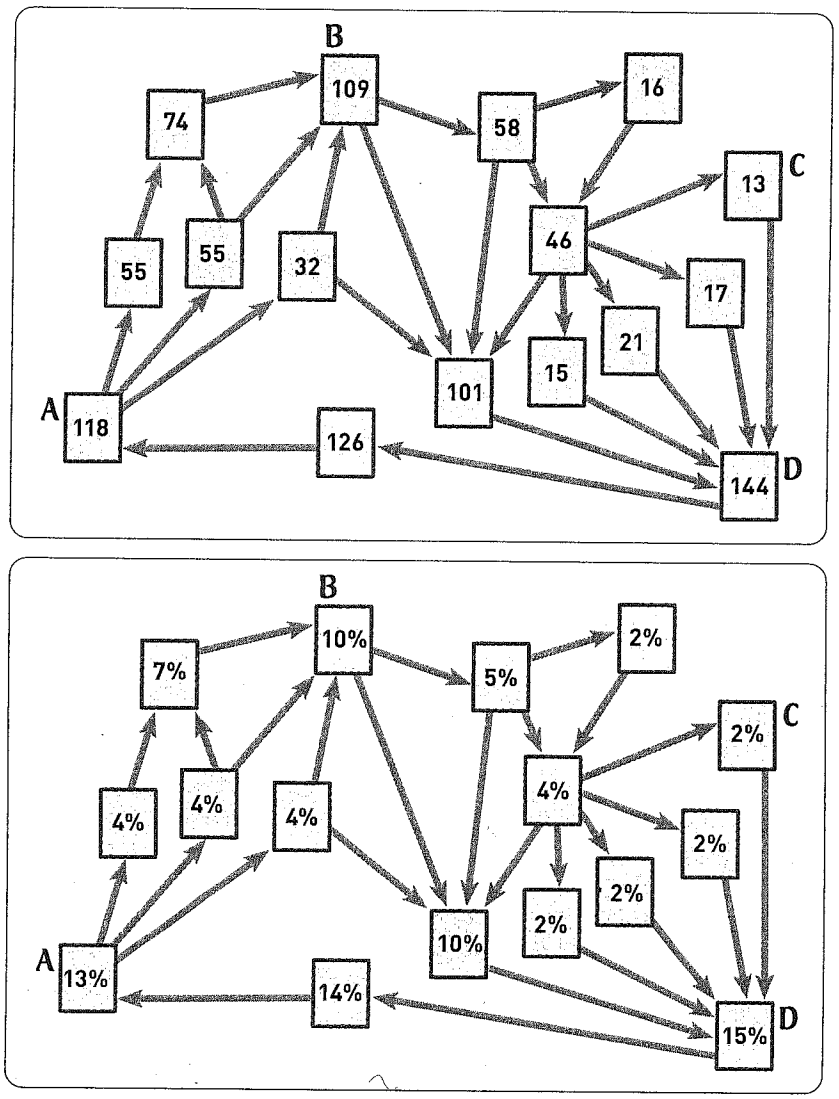
It's easy to simulate this process by computer. I wrote a program to do just that and ran it until the surfer had visited 1000 pages. (Of course, this doesn't mean 1000 distinct pages. Multiple visits to the same page are counted, and in this small example, all of the pages were visited many times.) The results of the 1000 simulated visits are shown in the top panel of the figure on the next page. You can see that page *D* was the most frequently visited, with 144 visits.

Just as with public opinion polls, we can improve the accuracy of our simulation by increasing the number of random samples. I reran the simulation, this time waiting until the surfer had visited one million pages. (In case you're wondering, this takes less than half a second on my computer!) With such a large number of visits, it's preferable to present the results as percentages. This is what you can see in the bottom panel of the figure on the facing page. Again, page *D* was the most frequently visited, with 15% of the visits.

What is the connection between our random surfer model and the authority trick that we would like to use for ranking web pages? The percentages calculated from random surfer simulations turn out to be exactly what we need to measure a page's authority. So let's define the *surfer authority score* of a web page to be the percentage of time that a random surfer would spend visiting that page. Remarkably, the surfer authority score incorporates both of our earlier tricks for ranking the importance of web pages. We'll examine these each in turn.

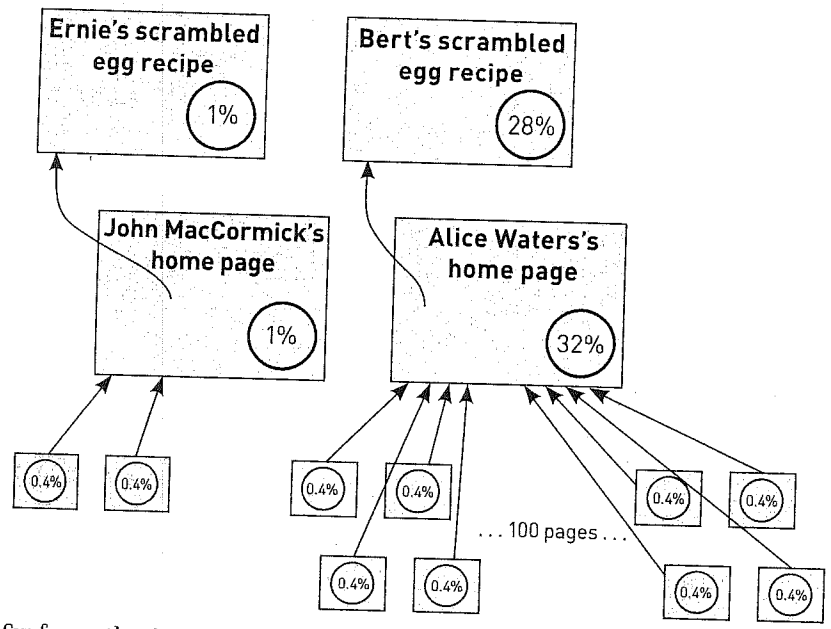
First, we had the hyperlink trick: the main idea here was that a page with many incoming links should receive a high ranking. This is also true in the random surfer model, because a page with many incoming links has many chances to be visited. Page *D* in the lower panel on the next page is a good example of this: it has five incoming links—more than any other page in the simulation—and ends up having the highest surfer authority score (15%).

Second, we had the authority trick. The main idea was that an incoming link from a highly authoritative page should improve a page's ranking more than an incoming link from a less authoritative page. Again, the random surfer model takes account of this. Why? Because an incoming link from a popular page will have more opportunities to be followed than a link from an unpopular page. To



Random surfer simulations. Top: Number of visits to each page in a 1000-visit simulation. Bottom: Percentage of visits to each page in a simulation of one million visits.

see an instance of this in our simulation example, compare pages *A* and *C* in the lower panel above: each has exactly one incoming link, but page *A* has a much higher surfer authority score (13% vs. 2%) because of the quality of its incoming link.

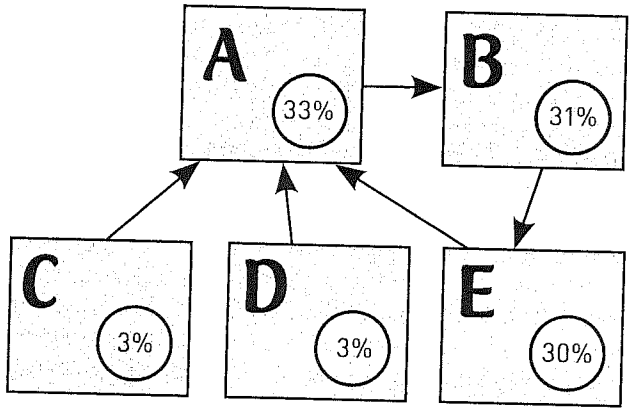


Surfer authority scores for the scrambled egg example on page 29. Bert and Ernie each have exactly one incoming link conferring authority on their pages, but Bert's page will be ranked higher in a web search query for "scrambled eggs."

Notice that the random surfer model simultaneously incorporates both the hyperlink trick and authority trick. In other words, the quality and quantity of incoming links at each page are all taken into account. Page *B* demonstrates this: it receives its relatively high score (10%) due to three incoming links from pages with moderate scores, ranging from 4% to 7%.

The beauty of the random surfer trick is that, unlike the authority trick, it works perfectly well whether or not there are cycles in the hyperlinks. Going back to our earlier scrambled egg example (page 29), we can easily run a random surfer simulation. After several million visits, my own simulation produced the surfer authority scores shown in the figure above. Notice that, as with our earlier calculation using the authority trick, Bert's page receives a much higher score than Ernie's (28% vs. 1%)—despite the fact that each has exactly one incoming link. So Bert would be ranked higher in a web search query for "scrambled eggs."

Now let's turn to the more difficult example from earlier: the figure on page 30, which caused an insurmountable problem for our



Surfer authority scores for the earlier example with a cycle of hyperlinks (page 30). The random surfer trick has no trouble computing appropriate scores, despite the presence of a cycle ($A \rightarrow B \rightarrow E \rightarrow A$).

original authority trick because of the cycle of hyperlinks. Again, it's easy to run a computer simulation of random surfers, producing the surfer authority scores in the figure above. The surfer authority scores determined by this simulation tell us the final ranking that would be used by a search engine when returning results: page *A* is highest, followed by *B*, then *E*, with *C* and *D* sharing last place.

PAGERANK IN PRACTICE

The random surfer trick was described by Google's cofounders in their now-famous 1998 conference paper, "The Anatomy of a Large-scale Hypertextual Web Search Engine." In combination with many other techniques, variants of this trick are still used by the major search engines. There are, however, numerous complicating factors, which mean that the actual techniques employed by modern search engines differ somewhat from the random surfer trick described here.

One of these complicating factors strikes at the heart of PageRank: the assumption that hyperlinks confer legitimate authority is sometimes questionable. We already learned that although hyperlinks can represent criticisms rather than recommendations, this tends not to be a significant problem in practice. A much more severe problem is that people can abuse the hyperlink trick to artificially inflate the ranking of their own web pages. Suppose you run a website called BooksBooksBooks.com that sells (surprise, surprise) books. Using automated technology, it's relatively easy to create a large

number—say, 10,000—of different web pages that all have links to BooksBooksBooks.com. Thus, if search engines computed PageRank authorities exactly as described here, BooksBooksBooks.com might undeservedly get a score thousands of times higher than other bookstores, resulting in a high ranking and thus more sales.

Search engines call this kind of abuse *web spam*. (The terminology comes from an analogy with e-mail spam: unwanted messages in your e-mail inbox are similar to unwanted web pages that clutter the results of a web search.) Detecting and eliminating various types of web spam are important ongoing tasks for all search engines. For example, in 2004, some researchers at Microsoft found over 300,000 websites that had *exactly* 1001 pages linking to them—a very suspicious state of affairs. By inspecting these websites manually, the researchers found that the vast majority of these incoming hyperlinks were web spam.

Hence, search engines are engaged in an arms race against web spammers and are constantly trying to improve their algorithms in order to return realistic rankings. This drive to improve PageRank has spawned a great deal of academic and industrial research into other algorithms that use the hyperlink structure of the web for ranking pages. Algorithms of this kind are often referred to as *link-based ranking algorithms*.

Another complicating factor relates to the efficiency of PageRank computations. Our surfer authority scores were computed by running random simulations, but running a simulation of that kind on the entire web would take far too long to be of practical use. So search engines do not compute their PageRank values by simulating random surfers: they use mathematical techniques that give the same answers as our own random surfer simulations, but with far less computational expense. We studied the surfer-simulation technique because of its intuitive appeal, and because it describes *what* the search engines calculate, not *how* they calculate it.

It's also worth noting that commercial search engines determine their rankings using a lot more than just a link-based ranking algorithm like PageRank. Even in their original, published description of Google back in 1998, Google's cofounders mentioned several other features that contributed to the ranking of search results. As you might expect, the technology has moved on from there: at the time of writing, Google's own website states that "more than 200 signals" are used in assessing the importance of a page.

Despite the many complexities of modern search engines, the beautiful idea at the heart of PageRank—that authoritative pages can confer authority on other pages via hyperlinks—remains valid.

It was this idea that helped Google to dethrone AltaVista, transforming Google from small startup to king of search in a few heady years. Without the core idea of PageRank, most web search queries would drown in a sea of thousands of matching, but irrelevant, web pages. PageRank is indeed an algorithmic gem that allows a needle to rise effortlessly to the top of its haystack.