Mathematics Behind the Internet

James H. Davenport

University of Bath

21 September 2009

“Google” — a new word?

“Google” — a new word?

I met this woman last night at a party and I came right home and googled her.

“Google” — a new word?

I met this woman last night at a party and I came right home and googled her.
2001 N.Y. Times 11 Mar. III. 12/3

“Google” — a new word?

I met this woman last night at a party and I came right home and googled her.
2001 N.Y. Times 11 Mar. III. 12/3

“Google” — a new word?

I met this woman last night at a party and I came right home and googled her.
2001 N.Y. Times 11 Mar. III. 12/3

Part of the *Oxford English Dictionary*’s definition of this verb.

Googol

$$10^{100} = 10,000,000,000,000,000,000,000,000,000,$$
$$000,000,000,000,000,000,000,000,000,000,000,000,$$
$$000,000,000,000,000,000,000,000,000,000,000,000$$

Googol

$$10^{100} = 10,000,000,000,000,000,000,000,000,000,$$
$$000,000,000,000,000,000,000,000,000,000,000,000,$$
$$000,000,000,000,000,000,000,000,000,000,000,000$$

The name "googol" was invented by a child (Dr. Kasner's nine-year-old nephew) who was asked to think up a name for a very big number, namely, 1 with a hundred zeros after it. *Oxford English Dictionary*

Googol

$$10^{100} = 10,000,000,000,000,000,000,000,000,000,$$
$$000,000,000,000,000,000,000,000,000,000,000,000,$$
$$000,000,000,000,000,000,000,000,000,000,000,000$$

The name "googol" was invented by a child (Dr. Kasner's nine-year-old nephew) who was asked to think up a name for a very big number, namely, 1 with a hundred zeros after it.   *Oxford English Dictionary*

We chose our system name, Google, because it is a common spelling of googol, or 10^{100} *and fits well with our goal of building very large-scale search engines.*

Googol

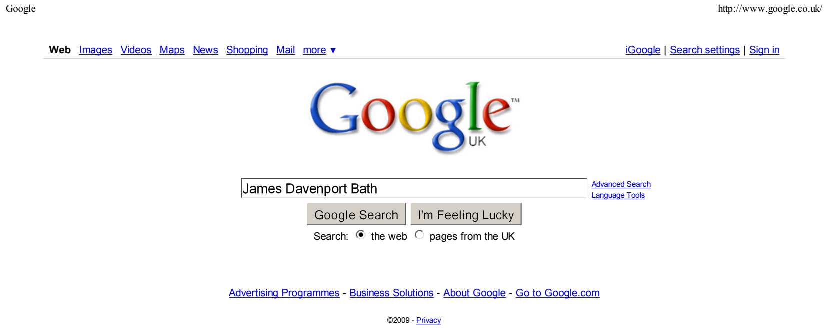
$$10^{100} = 10,000,000,000,000,000,000,000,000,000,$$
$$000,000,000,000,000,000,000,000,000,000,000,000,$$
$$000,000,000,000,000,000,000,000,000,000,000,000$$

The name "googol" was invented by a child (Dr. Kasner's nine-year-old nephew) who was asked to think up a name for a very big number, namely, 1 with a hundred zeros after it. *Oxford English Dictionary*

We chose our system name, Google, because it is a common spelling of googol, or 10^{100} *and fits well with our goal of building very large-scale search engines.*

The Anatomy of a Large-Scale Hypertextual Web Search Engine by Sergey Brin and Lawrence Page (1998).

How does Google choose what to show



“I’m feeling lucky” is often right

James Davenport

Davenport in the robes of a Cambridge PhD, wearing the Bronze Medal of the University of Helsinki (awarded 2001). Davenport lecturing at RISC (Austria) in 2007.

Professor James Davenport

Departments: Computer Science and Mathematical Sciences
Job Title: Hebron & Medlock Professor of Information Technology and (until 2005) University Director of Information Technology
Founding Editor-in-Chief LMS Journal of Computation and Mathematics: **submit** papers/queries here.
The first Ontario Research Chair in Computer Algebra
Former Royal Society Industrial Fellow.

Until June 2008, Director of Studies for undergraduates, and would still like them to speak English. He co-ordinates the Sun Campus Ambassador programme for the campus: the current ambassador is Anupriya Balikai, and the Bath group's pages are here. He represents the University on the Bristol Military Education Committee.

Works in Computer Algebra, where he is an author of a textbook, many papers and presentations. He has been Project Chair of the European OpenMath Project and its successor Thematic Network, with responsibilities for aligning OpenMath and MathML, where he gave (2/Oct/2008) a talk on the problems of differentiation, wrote a paper on conditions, and is producing Content Dictionaries and supervised a Reduce-based OpenMath/MathML translator. He is organising the 22nd OpenMath workshop. He was also Treasurer of the European Mathematical Trust.

He chairs the Research Committee's Working Party on Powerful Computing: report here. There was a training course run by NAG on 17-19 September: details here. A similar course is being run in Bristol 23-25 March: register here or contact Caroline Gardiner M.Sc. (Bath).

In July 2007 he visited Hagenberg im Muehlkreis, at a variety of meetings: his notes are here. In January/February 2008 he visited the Third Joining Educational Mathematics workshop in Barcelona. The slides of his talk are here, and his (partial) notes are here. On 18 February 2008 there was a special seminar in Bristol in honour of Clifford Cocks: his notes are here. In July 2008 he visited Birmingham (U.K.), at a variety of meetings: his notes are here.

Academic Year 2007/2008: in Semester 1 he taught CM30070: Computer Algebra and CM30078/50123: Advanced Networking. In Semester 2 he oversaw the teaching of CM30173/CM50210 Cryptography, coordinated CM50209 Security, and supervised various projects.

Academic Year 2008/2009: in Semester 1 he is teaching CM30070: Computer Algebra and CM30078/50123: Advanced Networking. In Semester 2 he is on sabbatical at the University of Waterloo. See some photographs here.

Academic Year 2009/20010: in Semester 1 he is teaching XX10190: Programming and Discrete Mathematics, CM30070: Computer Algebra and CM30078/50123: Advanced Networking. On Tuedays at 10.15 in 6E2.2, he is running a seminar series on cylindrical algebraic decomposition.

Whereas it has a lot to choose from

Web Images Videos Maps News Shopping Mail more ▾ Search settings | Sign in

Google [james davenport bath] Search Advanced Search

Search: the web pages from the UK

Web Show options... Results **1** - **10** of about **37,400** for **james davenport bath**. (**0.11** seconds)

James Davenport's Home Page
University of **Bath**. Computer Algebra, OpenMath Project, Mediated Learning Environments. Publications, resources.
people.**bath**.ac.uk/masjhd/ - Cached - Similar

[PDF] A Small OpenMath Type System **James Davenport Bath** 1.3.2c (Public)
File Format: PDF/Adobe Acrobat - View
A Small OpenMath Type System. **James Davenport**. **Bath**. 1.3.2c (Public) c© 1999 The OpenMath Consortium (24.969). Page 2. ESPRIT project 24969: OpenMath **...**
www.openmath.org/standard/sts.pdf - Similar
by J Davenport - Cited by 22 - Related articles - All 12 versions

[PDF] On Writing OpenMath Content Dictionaries **James Davenport Bath** 1.4 **...**
File Format: PDF/Adobe Acrobat - View
28 Jan 2000 **...** 2000-03-09. On Writing OpenMath Content Dictionaries. **James Davenport**. **Bath**. 1.4.5 (Restricted) c 2002 The OpenMath Consortium (24.969) **...**
www.openmath.org/documents/writingCDs.pdf - Similar
by J Davenport - Cited by 7 - Related articles - All 7 versions

[PDF] **James** H. **Davenport** Dept. Mathematical Sciences University of **Bath** **...**
File Format: PDF/Adobe Acrobat - View
11 Feb 2000 **...** On Writing OpenMath Content Dictionaries. **James** H. **Davenport**. Dept. Mathematical Sciences. University of **Bath**. **Bath** BA2 7AY. England **...**
staff.**bath**.ac.uk/masjhd/OpenMath/omcd2.pdf - Similar
by JH Davenport - Related articles

[PDF] The Computational Challenges of public-key cryptography **James** **...**
File Format: PDF/Adobe Acrobat - View
James Davenport. Dept. {Computer,Mathematical} Science{,s}. University of **Bath**. J.H.**Davenport@bath**.ac.uk. Public Key Cryptography. Two main methods: **...**
staff.**bath**.ac.uk/masjhd/BICS-paper.pdf - Similar
by PK Cryptography - Related articles - All 2 versions

How do we decide which pages to choose

(It isn't luck!)

How do we decide which pages to choose

(It isn't luck!)
The basic idea is obvious,

How do we decide which pages to choose

(It isn't luck!)
The basic idea is obvious, with hindsight.
Choose the page with more links to it.

How do we decide which pages to choose

(It isn't luck!)

The basic idea is obvious, with hindsight.

Choose the page with more links to it.

$$
\begin{array}{ccc}
A & & B \\
\downarrow & \searrow & \downarrow \\
C & & D
\end{array}
$$

How do we decide which pages to choose

(It isn't luck!)
The basic idea is obvious, with hindsight.
Choose the page with more links to it.

$$\begin{tikzcd} A \arrow[d] \arrow[dr] & B \arrow[d] \\ C & D \end{tikzcd}$$

Obviously D is more popular than C.

But the Web is much more complicated!

But the Web is much more complicated!

$$
\begin{tikzcd}
A \arrow[d] \arrow[dr] & B \arrow[d] \\
C \arrow[d] & D \arrow[d] \\
E \arrow[d] & F \arrow[d] \\
G & H
\end{tikzcd}
$$

But the Web is much more complicated!

$$
\begin{array}{ccc}
A & & B \\
\downarrow & \searrow & \downarrow \\
C & & D \\
\downarrow & & \downarrow \\
E & & F \\
\downarrow & & \downarrow \\
G & & H
\end{array}
$$

E and F each have only one link to them, but, since D is more popular than C, we should regard F as more popular than E (and H as more popular than G).

But the Web is much more complicated!

And constantly changing.

But the Web is much more complicated!

And constantly changing.

$$
\begin{tikzcd}
A \arrow[d] \arrow[dr] & B \arrow[d] \\
C \arrow[d] & D \arrow[d] \arrow[dl] \\
E \arrow[d] & F \arrow[d] \\
G & H
\end{tikzcd}
$$

But the Web is much more complicated!

And constantly changing.

$$\begin{array}{ccc} A & & B \\ \downarrow & \searrow & \downarrow \\ C & & D \\ \downarrow & \swarrow & \downarrow \\ E & & F \\ \downarrow & & \downarrow \\ G & & H \end{array}$$

Now E is more popular than F.

But the Web is much more complicated!

And constantly changing.

$$
\begin{tikzcd}
A \arrow[d] \arrow[dr] & B \arrow[d] \\
C \arrow[d] & D \arrow[d] \arrow[dl] \\
E \arrow[d] & F \arrow[d] \\
G & H
\end{tikzcd}
$$

Now E is more popular than F. And G is more popular than H,

But the Web is much more complicated!

And constantly changing.

$$\begin{array}{ccc} A & & B \\ \downarrow & \searrow & \downarrow \\ C & & D \\ \downarrow & \swarrow & \downarrow \\ E & & F \\ \downarrow & & \downarrow \\ G & & H \end{array}$$

Now E is more popular than F. And G is more popular than H, even though nothing has changed for G itself.

But the Web is much much more complicated!

But the Web is much much more complicated!

1. The real Web contains (lots of) loops.

But the Web is much much more complicated!

1. The real Web contains (lots of) loops.
2. The real Web is utterly massive — no-one, not even Google, really knows how big.

But the Web is much much more complicated!

1. The real Web contains (lots of) loops.
2. The real Web is utterly massive — no-one, not even Google, really knows how big.
3. The real Web keeps changing.

But the Web is much much more complicated!

1. The real Web contains (lots of) loops.
2. The real Web is utterly massive — no-one, not even Google, really knows how big.
3. The real Web keeps changing.
4. The real Web is commercially valuable, so there are incentives to manipulate it.

The real Web contains loops

The real Web contains loops

Nevertheless, we could, *in principle* write down a set of (linear) equations for the popularity of each page,

The real Web contains loops

Nevertheless, we could, *in principle* write down a set of (linear) equations for the popularity of each page, which would depend on the popularity of the pages which linked to it,

The real Web contains loops

Nevertheless, we could, *in principle* write down a set of (linear) equations for the popularity of each page, which would depend on the popularity of the pages which linked to it, which would depend on the popularity of the pages which linked to it . . . .

The real Web contains loops

Nevertheless, we could, *in principle* write down a set of (linear) equations for the popularity of each page, which would depend on the popularity of the pages which linked to it, which would depend on the popularity of the pages which linked to it . . . .
Then we could solve these equations.

The real Web contains loops

Nevertheless, we could, *in principle* write down a set of (linear) equations for the popularity of each page, which would depend on the popularity of the pages which linked to it, which would depend on the popularity of the pages which linked to it . . . .
Then we could solve these equations.
These equations have a name: they are the equations for the **principal eigenvector** of the connectivity matrix of the Web.

The real Web contains loops

Nevertheless, we could, *in principle* write down a set of (linear) equations for the popularity of each page, which would depend on the popularity of the pages which linked to it, which would depend on the popularity of the pages which linked to it . . . .
Then we could solve these equations.
These equations have a name: they are the equations for the **principal eigenvector** of the connectivity matrix of the Web.
The genius of Brin and Page was to realise that these equations *could* be solved,

The real Web contains loops

Nevertheless, we could, *in principle* write down a set of (linear) equations for the popularity of each page, which would depend on the popularity of the pages which linked to it, which would depend on the popularity of the pages which linked to it . . . .
Then we could solve these equations.
These equations have a name: they are the equations for the **principal eigenvector** of the connectivity matrix of the Web.
The genius of Brin and Page was to realise that these equations *could* be solved, and in a distributed and iterative manner.

The real Web contains loops

Nevertheless, we could, *in principle* write down a set of (linear) equations for the popularity of each page, which would depend on the popularity of the pages which linked to it, which would depend on the popularity of the pages which linked to it . . . .
Then we could solve these equations.
These equations have a name: they are the equations for the **principal eigenvector** of the connectivity matrix of the Web.
The genius of Brin and Page was to realise that these equations *could* be solved, and in a distributed and iterative manner. It's known as the "Page Rank" algorithm.

The real Web contains loops

Nevertheless, we could, *in principle* write down a set of (linear) equations for the popularity of each page, which would depend on the popularity of the pages which linked to it, which would depend on the popularity of the pages which linked to it ....
Then we could solve these equations.
These equations have a name: they are the equations for the **principal eigenvector** of the connectivity matrix of the Web.
The genius of Brin and Page was to realise that these equations *could* be solved, and in a distributed and iterative manner. It's known as the "Page Rank" algorithm.
Solving these equations is what makes Google work!

The real Web contains loops

Nevertheless, we could, *in principle* write down a set of (linear) equations for the popularity of each page, which would depend on the popularity of the pages which linked to it, which would depend on the popularity of the pages which linked to it . . . .
Then we could solve these equations.
These equations have a name: they are the equations for the **principal eigenvector** of the connectivity matrix of the Web.
The genius of Brin and Page was to realise that these equations *could* be solved, and in a distributed and iterative manner. It's known as the "Page Rank" algorithm.
Solving these equations is what makes Google work!
So it's not really "I'm feeling lucky", it's "I believe in eigenvectors"!

Flow in the Internet

Assume the routers R_1 and R_2 have total capacity 1 each.

$$\begin{tikzcd} & A_1 \arrow[d] & B_1 \arrow[d] & \\ C_1 \arrow[r] & R_1 \arrow[r] \arrow[d] & R_2 \arrow[r] \arrow[d] & C_2 \\ & A_2 & B_2 & \end{tikzcd}$$

Flow in the Internet

Assume the routers R_1 and R_2 have total capacity 1 each.

$$\begin{array}{ccccccc} & & A_1 & & B_1 & & \\ & & \downarrow & & \downarrow & & \\ C_1 & \to & R_1 & \to & R_2 & \to & C_2 \\ & & \downarrow & & \downarrow & & \\ & & A_2 & & B_2 & & \end{array}$$

What is the best way of allocating bandwidth to the various flows $A_1 \to A_2$, $B_1 \to B_2$ and $C_1 \to C_2$?

Flow in the Internet

Assume the routers R_1 and R_2 have total capacity 1 each.

$$\begin{array}{ccccccc} & & A_1 & & B_1 & & \\ & & \downarrow & & \downarrow & & \\ C_1 & \rightarrow & R_1 & \rightarrow & R_2 & \rightarrow & C_2 \\ & & \downarrow & & \downarrow & & \\ & & A_2 & & B_2 & & \end{array}$$

What is the best way of allocating bandwidth to the various flows $A_1 \rightarrow A_2$, $B_1 \rightarrow B_2$ and $C_1 \rightarrow C_2$?
Of course, it all depends what you mean by “best”.

Network Most Efficient

Network Most Efficient

A and B each get 1, and C nothing.

Network Most Efficient

A and B each get 1, and C nothing.

$$\begin{array}{ccccccc} & & A_1 & & B_1 & & \\ & & \downarrow 1 & & \downarrow 1 & & \\ C_1 & \xrightarrow{0} & R_1 & \xrightarrow{0} & R_2 & \xrightarrow{0} & C_2 \\ & & \downarrow 1 & & \downarrow 1 & & \\ & & A_2 & & B_2 & & \end{array}$$

Total flow 2, but C might feel aggrieved.

Max–min Fairness

Max–min Fairness

The worst-off person gets as much as possible.

Max–min Fairness

The worst-off person gets as much as possible.
Each flow gets $1/2$.

$$
\begin{array}{ccccccc}
 & & A_1 & & B_1 & & \\
 & & \downarrow 1/2 & & \downarrow 1/2 & & \\
C_1 & \xrightarrow{1/2} & R_1 & \xrightarrow{1/2} & R_2 & \xrightarrow{1/2} & C_2 \\
 & & \downarrow 1/2 & & \downarrow 1/2 & & \\
 & & A_2 & & B_2 & &
\end{array}
$$

Max–min Fairness

The worst-off person gets as much as possible.
Each flow gets $1/2$.

$$
\begin{array}{ccccccc}
 & & A_1 & & B_1 & & \\
 & & \downarrow 1/2 & & \downarrow 1/2 & & \\
C_1 & \xrightarrow{1/2} & R_1 & \xrightarrow{1/2} & R_2 & \xrightarrow{1/2} & C_2 \\
 & & \downarrow 1/2 & & \downarrow 1/2 & & \\
 & & A_2 & & B_2 & &
\end{array}
$$

Total flow 1.5, *but* C is getting twice as much routing done for him as A and B are.

Max–min Fairness

The worst-off person gets as much as possible.
Each flow gets $1/2$.

$$\begin{array}{ccccccc}
 & & A_1 & & B_1 & & \\
 & & \downarrow 1/2 & & \downarrow 1/2 & & \\
C_1 & \xrightarrow{1/2} & R_1 & \xrightarrow{1/2} & R_2 & \xrightarrow{1/2} & C_2 \\
 & & \downarrow 1/2 & & \downarrow 1/2 & & \\
 & & A_2 & & B_2 & &
\end{array}$$

Total flow 1.5, *but* C is getting twice as much routing done for him as A and B are.
A and B might feel aggrieved.

Proportional Fairness

Proportional Fairness

Each flow gets the same amount of effort from the routers.

Proportional Fairness

Each flow gets the same amount of effort from the routers.
A and B each get $2/3$, and C gets $1/3$.

$$\begin{array}{ccccccc}
 & & A_1 & & B_1 & & \\
 & & \downarrow 2/3 & & \downarrow 2/3 & & \\
C_1 & \xrightarrow{1/3} & R_1 & \xrightarrow{1/3} & R_2 & \xrightarrow{1/3} & C_2 \\
 & & \downarrow 2/3 & & \downarrow 2/3 & & \\
 & & A_2 & & B_2 & &
\end{array}$$

Proportional Fairness

Each flow gets the same amount of effort from the routers.
A and B each get $2/3$, and C gets $1/3$.

$$\begin{array}{ccccccc} & & A_1 & & B_1 & & \\ & & \downarrow 2/3 & & \downarrow 2/3 & & \\ C_1 & \xrightarrow{1/3} & R_1 & \xrightarrow{1/3} & R_2 & \xrightarrow{1/3} & C_2 \\ & & \downarrow 2/3 & & \downarrow 2/3 & & \\ & & A_2 & & B_2 & & \end{array}$$

Total flow is now $\frac{5}{3} \approx 1.66$, better than max-min, but not as good as the flow where C gets nothing.

But in the real world

But in the real world

- Routers and links have widely different capacities

But in the real world

- Routers and links have widely different capacities
- The network is **much** more complicated, and always changing

But in the real world

- Routers and links have widely different capacities
- The network is **much** more complicated, and always changing
- No-one has overall knowledge of the flows.

But in the real world

- Routers and links have widely different capacities
- The network is **much** more complicated, and always changing
- No-one has overall knowledge of the flows.

But in the real world

- Routers and links have widely different capacities
- The network is **much** more complicated, and always changing
- No-one has overall knowledge of the flows.

Nevertheless, the **purely local** algorithm devised by van Jacobsen (earlier; published 1988) was shown in 1997 to converge to proportional fairness.

Numbers rather than Padlocks (I)

A wishes to send x to B.

Numbers rather than Padlocks (I)

A wishes to send x to B.
A and B each think of a random number, say a and b.

Numbers rather than Padlocks (I)

A wishes to send x to B.
A and B each think of a random number, say a and b.

A's action	Message	B's action
multiply x by a		
	xa $\searrow$	
		multiply message by b
	$xba = xab$ $\swarrow$	
divide message by a		
	xb $\searrow$	
		divide message by b

Numbers rather than Padlocks (I)

A wishes to send x to B.
A and B each think of a random number, say a and b.

A's action	Message	B's action
multiply x by a		
	xa	
	$\searrow$	
		multiply message by b
	$xba = xab$	
	$\swarrow$	
divide message by a		
	xb	
	$\searrow$	
		divide message by b

In practice, to avoid guessing, and numerical errors, x, a and b are whole numbers modulo some *large* prime p.

Numbers rather than Padlocks (I) — snag

A's action	Message	B's action
multiply x by a		
	xa	
	$\searrow$	
		multiply message by b
	$xba = xab$	
	$\swarrow$	
divide message by a		
	xb	
	$\searrow$	
		divide message by b

Numbers rather than Padlocks (I) — snag

A's action	Message	B's action
multiply x by a		
	xa	
		multiply message by b
	$xba = xab$	
divide message by a		
	xb	
		divide message by b

Eavesdropper computes $\dfrac{xa \cdot xb}{xab}$

Numbers rather than Padlocks (I) — snag

A's action	Message	B's action
multiply x by a		
	xa $\searrow$	
		multiply message by b
	$xba = xab$ $\swarrow$	
divide message by a		
	xb $\searrow$	
		divide message by b

Eavesdropper computes $\dfrac{xa \cdot xb}{xab} = x$.

Numbers rather than Padlocks (I) — snag

A's action	Message	B's action
multiply x by a		
	xa	
	$\searrow$	
		multiply message by b
	$xba = xab$	
	$\swarrow$	
divide message by a		
	xb	
	$\searrow$	
		divide message by b

Eavesdropper computes $\dfrac{xa \cdot xb}{xab} = x$.
So replacing the padlocks by numbers has given the eavesdropper the chance of doing arithmetic.

Numbers rather than Padlocks (II)

Let's be more subtle.

Numbers rather than Padlocks (II)

Let's be more subtle.

A's action	Message	B's action
raise x to power a		
	x^a	
	$\searrow$	
		raise message to power b
	$(x^b)^a = (x^a)^b$	
	$\swarrow$	
take ath root of message		
	x^b	
	$\searrow$	
		take bth root of message

Numbers rather than Padlocks (II)

Let's be more subtle.

A's action	Message	B's action
raise x to power a		
	x^a	
	$\searrow$	
		raise message to power b
	$(x^b)^a = (x^a)^b$	
	$\swarrow$	
take ath root of message		
	x^b	
	$\searrow$	
		take bth root of message

Surely this frustrates the eavesdropper?

But what about logarithms?

A's action	Message	B's action
raise x to power a		
	x^a $\searrow$	
		raise message to power b
	$(x^b)^a = (x^a)^b$ $\swarrow$	
take ath root of message		
	x^b $\searrow$	
		take bth root of message

Eavesdropper computes
$$\frac{\log(x^a) \cdot \log(x^b)}{\log(x^{ab})}$$

But what about logarithms?

A's action	Message	B's action
raise x to power a		
	x^a $\searrow$	
		raise message to power b
	$(x^b)^a = (x^a)^b$ $\swarrow$	
take ath root of message		
	x^b $\searrow$	
		take bth root of message

Eavesdropper computes

$$\frac{\log(x^a) \cdot \log(x^b)}{\log(x^{ab})} = \frac{a \log(x) \cdot b \log(x)}{ab \log(x)}$$

But what about logarithms?

A's action	Message	B's action
raise x to power a		
	$x^a \searrow$	
		raise message to power b
	$(x^b)^a = (x^a)^b \swarrow$	
take ath root of message		
	$x^b \searrow$	
		take bth root of message

Eavesdropper computes

$$\frac{\log(x^a)\cdot\log(x^b)}{\log(x^{ab})} = \frac{a\log(x)\cdot b\log(x)}{ab\log(x)} = \log(x).$$

But what about logarithms?

A's action	Message	B's action
raise x to power a		
	$x^a \searrow$	
		raise message to power b
	$(x^b)^a = (x^a)^b \swarrow$	
take ath root of message		
	$x^b \searrow$	
		take bth root of message

Eavesdropper computes

$$\frac{\log(x^a) \cdot \log(x^b)}{\log(x^{ab})} = \frac{a \log(x) \cdot b \log(x)}{ab \log(x)} = \log(x).$$

Essentially the same trick as before, but with logarithms!

Do logarithms exist?

Do logarithms exist?

Remember that we are working modulo a *large* prime p.

Do logarithms exist?

Remember that we are working modulo a *large* prime p. For simplicity, I will take $p = 41$, since it's small enough, and logs base 5, so that $\log(5) = 1$.

Do logarithms exist?

Remember that we are working modulo a *large* prime p. For simplicity, I will take $p = 41$, since it's small enough, and logs base 5, so that $\log(5) = 1$.

1	2	3	4	5	6	7	8	9	10
0				1					
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40

Do logarithms exist?

Remember that we are working modulo a *large* prime p. For simplicity, I will take $p = 41$, since it's small enough, and logs base 5, so that $\log(5) = 1$.

1	2	3	4	5	6	7	8	9	10
0				1					
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
				2					
31	32	33	34	35	36	37	38	39	40

Do logarithms exist?

Remember that we are working modulo a *large* prime p. For simplicity, I will take $p = 41$, since it's small enough, and logs base 5, so that $\log(5) = 1$.

1	2	3	4	5	6	7	8	9	10
0				1					
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
				2					
31	32	33	34	35	36	37	38	39	40

So $\log(125) = 3$, but $125 = 3 \cdot 41 + 2$

Do logarithms exist?

Remember that we are working modulo a *large* prime p. For simplicity, I will take $p = 41$, since it's small enough, and logs base 5, so that $\log(5) = 1$.

1	2	3	4	5	6	7	8	9	10
0				1					
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
				2					
31	32	33	34	35	36	37	38	39	40

So $\log(125) = 3$, but $125 = 3 \cdot 41 + 2 \equiv 2$ since we are working modulo 41.

Do logarithms exist?

Remember that we are working modulo a *large* prime p. For simplicity, I will take $p = 41$, since it's small enough, and logs base 5, so that $\log(5) = 1$.

1	2	3	4	5	6	7	8	9	10
0	3			1					
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
				2					
31	32	33	34	35	36	37	38	39	40

Do logarithms exist?

Remember that we are working modulo a *large* prime p. For simplicity, I will take $p = 41$, since it's small enough, and logs base 5, so that $\log(5) = 1$.

1	2	3	4	5	6	7	8	9	10
0	3			1					
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
				2					
31	32	33	34	35	36	37	38	39	40

And we can fill in: $10 = 2 \cdot 5$, so $\log(10) = 4$.

Do logarithms exist?

Remember that we are working modulo a *large* prime p. For simplicity, I will take $p = 41$, since it's small enough, and logs base 5, so that $\log(5) = 1$.

1	2	3	4	5	6	7	8	9	10
0	3			1					
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
				2					
31	32	33	34	35	36	37	38	39	40

And we can fill in: $10 = 2 \cdot 5$, so $\log(10) = 4$. Also $4 = 2 \cdot 2$ so $\log(4) = 3 + 3 = 6$.

Do logarithms exist?

Remember that we are working modulo a *large* prime p. For simplicity, I will take $p = 41$, since it's small enough, and logs base 5, so that $\log(5) = 1$.

1	2	3	4	5	6	7	8	9	10
0	3		6	1			9		4
11	12	13	14	15	16	17	18	19	20
					12				7
21	22	23	24	25	26	27	28	29	30
				2					
31	32	33	34	35	36	37	38	39	40
	15								

Do logarithms exist?

Remember that we are working modulo a *large* prime p. For simplicity, I will take $p = 41$, since it's small enough, and logs base 5, so that $\log(5) = 1$.

1	2	3	4	5	6	7	8	9	10
0	3		6	1			9		4
11	12	13	14	15	16	17	18	19	20
					12				7
21	22	23	24	25	26	27	28	29	30
				2					
31	32	33	34	35	36	37	38	39	40
	15								

$40 = 2 \cdot 20$, so $\log(40) = \log(2) + \log(20) = 3 + 7 = 10$.

Do logarithms exist?

Remember that we are working modulo a *large* prime p. For simplicity, I will take $p = 41$, since it's small enough, and logs base 5, so that $\log(5) = 1$.

1	2	3	4	5	6	7	8	9	10
0	3		6	1			9		4
11	12	13	14	15	16	17	18	19	20
					12				7
21	22	23	24	25	26	27	28	29	30
				2					
31	32	33	34	35	36	37	38	39	40
	15								10

Do logarithms exist?

Remember that we are working modulo a *large* prime p. For simplicity, I will take $p = 41$, since it's small enough, and logs base 5, so that $\log(5) = 1$.

1	2	3	4	5	6	7	8	9	10
0	3		6	1			9		4
11	12	13	14	15	16	17	18	19	20
					12				7
21	22	23	24	25	26	27	28	29	30
				2					
31	32	33	34	35	36	37	38	39	40
	15								10

$80 = 2 \cdot 40$, so $\log(80) = 13$, but $80 \equiv 39$, and so on

Do logarithms exist?

Remember that we are working modulo a *large* prime p. For simplicity, I will take $p = 41$, since it's small enough, and logs base 5, so that $\log(5) = 1$.

1	2	3	4	5	6	7	8	9	10
0	3		6	1			9		4
11	12	13	14	15	16	17	18	19	20
					12				7
21	22	23	24	25	26	27	28	29	30
				2					
31	32	33	34	35	36	37	38	39	40
	15	19				16		13	10

Do logarithms exist?

Remember that we are working modulo a *large* prime p. For simplicity, I will take $p = 41$, since it's small enough, and logs base 5, so that $\log(5) = 1$.

1	2	3	4	5	6	7	8	9	10
0	3		6	1			9		4
11	12	13	14	15	16	17	18	19	20
					12				7
21	22	23	24	25	26	27	28	29	30
				2					
31	32	33	34	35	36	37	38	39	40
	15	19				16		13	10

But $2 \cdot 33 = 66 \equiv 25$, so we deduce that $\log 25$ ought to be 22.

Logs aren't as simple as we thought!

Logs aren't as simple as we thought!

If we continue this process, we find that we have logarithms of only half the numbers, but each one has two values, e.g. 25 seems to be 2 and 22.

Logs aren't as simple as we thought!

If we continue this process, we find that we have logarithms of only half the numbers, but each one has two values, e.g. 25 seems to be 2 and 22.
A fatal snag?

Logs aren't as simple as we thought!

If we continue this process, we find that we have logarithms of only half the numbers, but each one has two values, e.g. 25 seems to be 2 and 22.

A fatal snag? Not really.

- There's a workround, which is messy, but not really difficult.

Logs aren't as simple as we thought!

If we continue this process, we find that we have logarithms of only half the numbers, but each one has two values, e.g. 25 seems to be 2 and 22.

A fatal snag? Not really.

- There's a workround, which is messy, but not really difficult.
- If we'd chosen a different base, say 7, then we would have logarithms of every non-zero number.

Logs aren't as simple as we thought!

If we continue this process, we find that we have logarithms of only half the numbers, but each one has two values, e.g. 25 seems to be 2 and 22.

A fatal snag? Not really.

- There's a workround, which is messy, but not really difficult.
- If we'd chosen a different base, say 7, then we would have logarithms of every non-zero number.

Logs aren't as simple as we thought!

If we continue this process, we find that we have logarithms of only half the numbers, but each one has two values, e.g. 25 seems to be 2 and 22.

A fatal snag? Not really.

- There's a workround, which is messy, but not really difficult.
- If we'd chosen a different base, say 7, then we would have logarithms of every non-zero number.

However, for *suitable* p, computing "discrete" logarithms is sufficiently hard that we can be sure of the safety of this scheme.

But it takes three messages

But it takes three messages

Can we do better?

But it takes three messages

Can we do better? Let x be a **public** number.

But it takes three messages

Can we do better? Let x be a **public** number.
Again, A and B choose random numbers a and b.

But it takes three messages

Can we do better? Let x be a **public** number.
Again, A and B choose random numbers a and b.

A's action	Message	B's action
raise x to power a		raise x to power b
	$x^a x^b$	
raise message to power a		raise message to power b
$(x^b)^a$		$(x^a)^b$

But it takes three messages

Can we do better? Let x be a **public** number.
Again, A and B choose random numbers a and b.

A's action	Message	B's action
raise x to power a		raise x to power b
	$x^a x^b$	
raise message to power a		raise message to power b
$(x^b)^a$		$(x^a)^b$

Now they are *both* in possession of $(x^a)^b = (x^b)^a$, which can be used as the key for any standard cipher.

But it takes three messages

Can we do better? Let x be a **public** number.
Again, A and B choose random numbers a and b.

A's action	Message	B's action
raise x to power a		raise x to power b
	$x^a \; x^b$	
raise message to power a		raise message to power b
$(x^b)^a$		$(x^a)^b$

Now they are *both* in possession of $(x^a)^b = (x^b)^a$, which can be used as the key for any standard cipher.
This is *one* reason why secure websites display a padlock: to assure you that they have gone through this process between *your* browser and the web site.

A few lessons

A few lessons

1. Always check for the padlock, which indicates that the data should be secure *between* you and the far end.

A few lessons

1. Always check for the padlock, which indicates that the data should be secure *between* you and the far end.
2. If possible, use *your* browser — your laptop/ BlackBerry/ whatever is safer than a browser in an Internet cafe.

A few lessons

1. Always check for the padlock, which indicates that the data should be secure *between* you and the far end.
2. If possible, use *your* browser — your laptop/ BlackBerry/ whatever is safer than a browser in an Internet cafe.
3. If you do use an Internet cafe, make sure you reboot the machine afterwards — not a guarantee, but definitely safer.