Information Theory

http://www.inf.ed.ac.uk/teaching/courses/it/

Week 9
Hashes and lossy memories

Iain Murray, 2010
School of Informatics, University of Edinburgh

Course overview

Source coding / compression:
— Losslessly representing information compactly
— Good probabilistic models → better compression

Noisy channel coding / error correcting codes:
— Add redundancy to transmit without error
— Large psuedo-random blocks approach theory limits
— Decoding requires large-scale inference (cf Machine learning)

Other topics in information theory
— Cryptography: not covered here
— Over capacity: using fewer bits than info. content
  — Rate distortion theory
  — Hashing

Rate distortion theory (taster)

Q. How do we store N bits of information with $N/3$ binary symbols (or N uses of a channel with $C = 1/3$)?

A. We can't without a non-negligible probability of error. But what if we were forced to try?

Idea 1:
— Drop $2N/3$ bits on the floor
— Transmit $N/3$ reliably
— Let the receiver guess the remaining bits

Expected number of errors: $2N/3 \cdot 1/2 = N/3$

Can we do better?

Reversing a block code

Swap roles of encoder and decoder for $[N, K]$ block code

E.g., Repetition code R_3

Put message through decoder first, transmit, then encode

110111010001000 → 11000 → 111111000000000

111 and 000 sent without error. Other six blocks lead to one error. Error rate $= 6/8 \cdot 1/3 = 1/4$, which is $< 1/3$

Slightly more on MacKay p167-8, much more in Cover and Thomas.

Rate distortion theory plays little role in practical lossy compression systems for (e.g.) images. It's a challenge to find practical coding schemes that respect perceptual measures of distortion.

Hashing

Hashes reduce large amounts of data into small values

(obviously the info. content of a source is not preserved in general)

Computers, humans and other animals can do amazing things, very quickly, based on tiny amounts of information.

Understanding how to use hashes can make progress in cognitive science and practical information systems.

Some of this is long-established computer science

A surprising amount is fertile research ground

Hashing motivational examples:

Many animals can do amazing things. While: `http://www.google.com/technology/pigeonrank.html` was a hoax. The paper on the next slide and others like it are not.

It isn't just pigeons. *Amazingly* humans can do this stuff too. Paul Speller demonstrated that humans can remember to distinguish similar pictures of pigeons over many minutes(!). `http://www.webarchive.org.uk/wayback/archive/20100223122414/http://www.oneandother.co.uk/participants/PaulSpeller`

How can we build systems that rapidly recall arbitrary labels attached to large numbers of rich but noisy media sources? YouTube has recently done this on a *very* large scale for copyright enforcement.

Some web browsers rapidly prove that a website isn't on a malware black-list without needing to access an external server, or needing an explicit list of all black-listed sites. (False positives can be checked with a request to an external server.)

Journal of Experimental Psychology:
Animal Behavior Processes
1984, Vol. 10, No. 2, 256–271

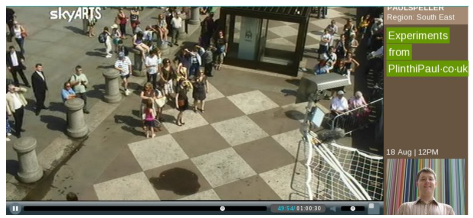
Pigeon Visual Memory Capacity

William Vaughan, Jr., and Sharon L. Greene
Harvard University

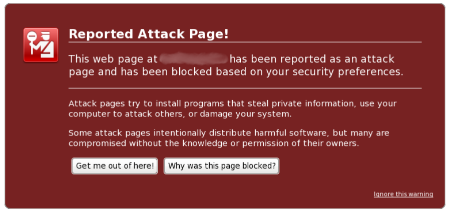
This article reports on four experiments on pigeon visual memory capacity. In the first experiment, pigeons learned to discriminate between 80 pairs of random shapes. Memory for 40 of those pairs was only slightly poorer following 490 days without exposure. In the second experiment, 80 pairs of photographic slides were learned; 629 days without exposure did not significantly disrupt memory. In the third experiment, 160 pairs of slides were learned; 731 days without exposure did not significantly disrupt memory. In the fourth experiment, pigeons learned to respond appropriately to 40 pairs of slides in the normal orientation and to respond in the opposite way when the slides were left-right reversed. After an interval of 751 days, there was a transient disruption in discrimination. These experiments demonstrate that pigeons have a heretofore unsuspected capacity with regard to both breadth and stability of memory for abstract stimuli and pictures.

Remembering images

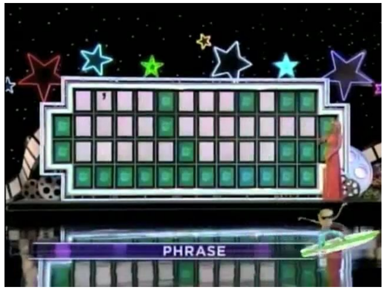
Remembering images



'Safe browsing'



Information retrieval



Wheel of Fortune, Nov 2010

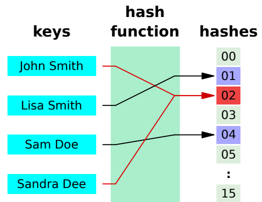
Information retrieval

Information retrieval

Hash functions

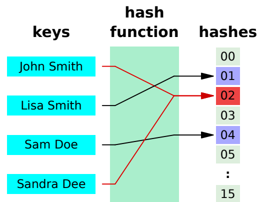
A common view:

file $\to$ b bit string (maybe like random bits)



Many uses: e.g., integrity checking, security, communication with feedback (`rsync`), indexing for information retrieval

Hash Tables



Hash indexes table of pointers to data

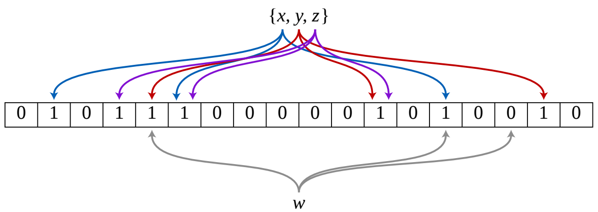
When hash table is empty at index, can *immediately* return `'Not found'`

Need to resolve conflicts. Ways include:

— List of data at each location. Check each item in list.
— Put pointer to data in next available location. Deletions need 'tombstones', rehash when table is full
— 'Cuckoo hashing': use > 1 hash and recursively move pointers out of the way to alternative locations.

Bloom Filters

Hash files multiple times (e.g., 3)
Set (or leave) bits equal to 1 at hash locations



Immediately know haven't seen w: ≥ 1 bits are zero

Notes on Bloom filters

Probability of false negative is zero

Probability of false positive depends on number of memory bits, M, and number of hash functions, K.

For fixed large M the optimal K (ignoring computation cost) turns out to be the one that sets $\approx 1/2$ of the bits to be on. This makes sense: the memory is less informative if sparse.

Other things we've learned are useful too. One way to get a low false positive rate is to make K small but M huge. This would have a huge memory cost... except we could compress the sparse bit-vector. This can potentially perform better than a standard Bloom filter (but the details will be more complicated).

Google Chrome uses (or at least used to use) a Bloom filter with $K = 4$ for its safe web-browsing feature.

Hashing in Machine Learning

A couple of example research papers

Semantic Hashing (Salakhutdinov & Hinton, 2009)

— Hash bits are "latent variables" underlying data
— 'Semantically' close files $\to$ close hashes
— Very fast retrieval of 'related' objects

Feature Hashing for Large Scale Multitask Learning, (Weinberger et al., 2009)

— 'Hash' large feature vectors without (much) loss in (spam) classification performance.
— Exploit multiple hash functions to give millions of users personalized spam filters at only about twice the cost (time and storage) of a single global filter(!).