Where is Knowledge? Discovering Semantic Proximity using User Access

Matthew Merzbacher

Department of Mathematics and Computer Science
Mills College
Oakland, California

matthew@mills.edu

1 October, 1999

Goal: Build Smarter Systems

Need: Organized Information

- Fast
- Convenient
- Scalable
- Error Tolerant
- Dynamic
-
-

Kinds of Information

- Facts
- Relationships
- Specific Rules
- General Rules/Patterns
-
-

Information Sources

- Humans
  - Domain Experts
  - End Users
  - Systems Engineers
  -
- Automated
  - Observation
  - Induction
  - Deduction
  -
- Semi-Automated

Fundamental Problem:
Want the skill and discrimination of a human with the speed and consequent scalability??? of a machine.

Structuring the Domain

Limit ourselves to a querying system
(DBMS, Web server, etc.)

Where might we find interesting patterns?

- Data
  - correlations
  - distributions
  - access patterns
- Schema (if there is one)
- Queries
- Results
- Updates
-

What about Time?

Things in temporal proximity may well be related

- Results of a Single Query
- Sequence of Queries
- Results of the Sequence
- Repeated Behavior
-

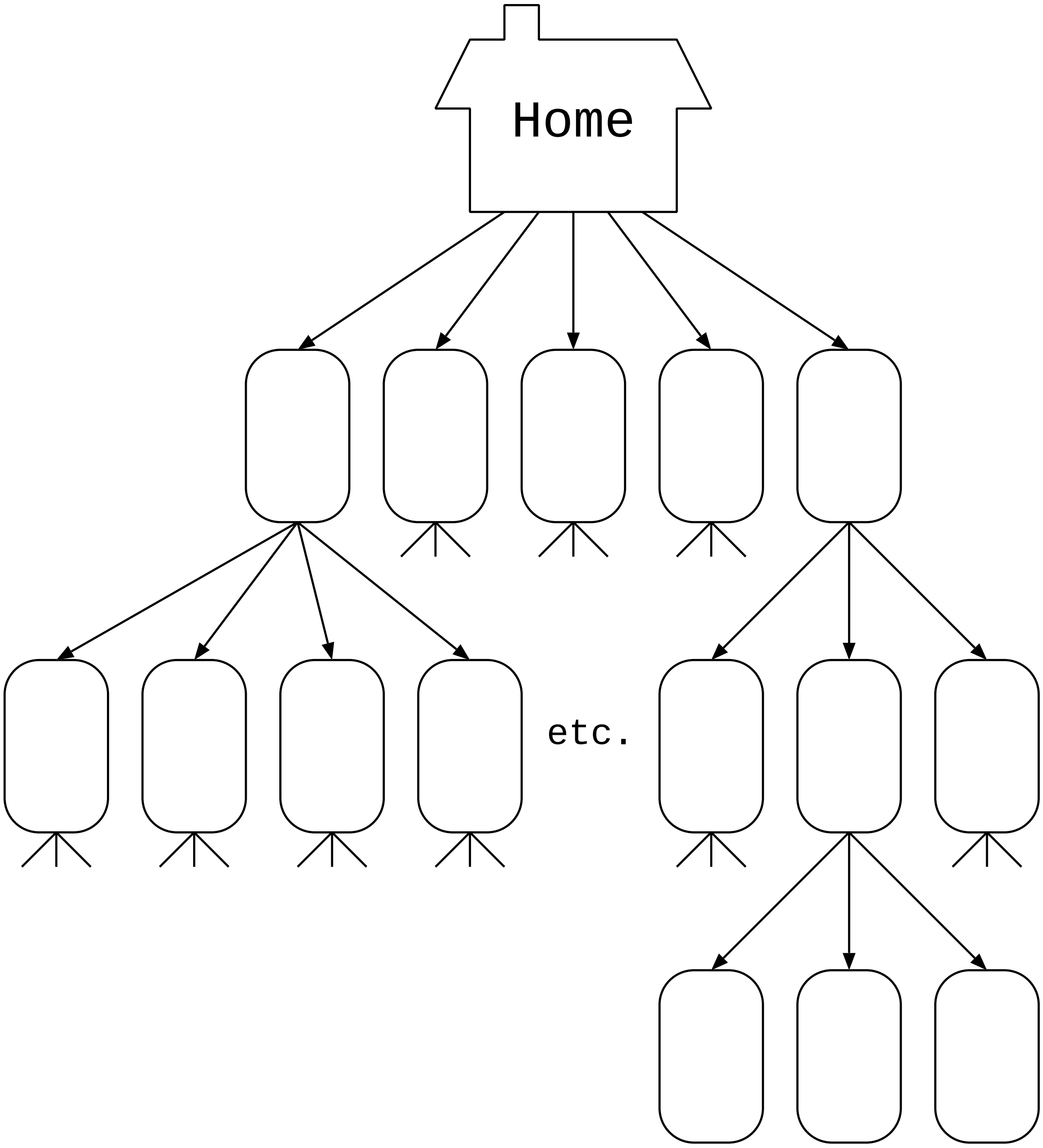
The Carrot and the Stick

Most systems look for undiscovered nuggets
- the carrot (or, perhaps, carat)

Repeated user behavior suggests disatisfaction
- the stick
- users stop doing something when satisfied

Today's Example: A Web Site

A Typical? Web Site



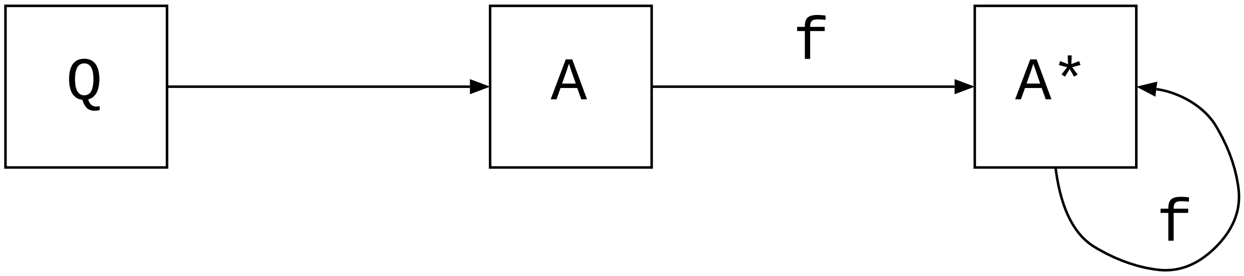
Problems

Finding information on a web site is hard

- several hops from entry to destination
- may take wrong turns along way
- new pages not properly integrated
- single-site search engines limited
- difficult to determine if desired information is *not* even available (the dog that doesn't bark)

Partial Solution:
Take advantage of well-traveled paths

Browsing



Used for:

– library databases

– web

The function f is a *metric*

Metrics

Measure distance/nearness between objects

- Tabular

	A	B	C	D
A	100	50	72	8
B	50	100	3	12
C	72	3	100	20
D	8	12	20	100

- Computational

$$F(O_1, O_2) = \sqrt{(O_1[A] - O_2[A])^2 + ...}$$

Characteristics of Metrics

- Tabular
  - simple
  - fast
  - large (O(N^2))
- Computational
  - small
  - potentially slower
  - hard/impossible to construct

Goal:
simplicity of tabular metrics without the size

What about the Web?

Need metric indicating proximity between pages

Proposed Metric: Are the pages linked?
- Binary metric
  + no partial or weighted links
- Non-commutative
- Typically few links per page

Task:
automatically find new useful links from each page

Some Related Work

Adaptive Web Sites (Washington)

WebWatcher (Carnegie–Mellon)

Avanti (GMD)

Popular approaches use classification
- based on content

Our Approach:
use historical access patterns

Dynamic Nearness

Find shortcuts for common paths

Advantages:

- can be run off-line (and infrequently)
- dynamically evolving
  - reflects changes in access patterns
- efficiently computable
- suitable for web sites
- allows automatic update of web pages
  - if desired

Definitions

Transaction –
sequence of pages accessed on a web site by a user over a short time span

Dynamic Nearness (DN) –
probable frequency that two pages will appear in same transaction

If DN is high, pages should be linked

DN Algorithm

```
initialize all counts to zero

for each transaction T
  for each pair of pages U and V in T
    increment count between U and V
```

Can only count links from earlier pages to later pages in transaction

Example

- T1 visits A and C
- T2 visits A and D
- T3 visits all four pages
- T4 visits A, B and D

Result

	A	B	C	D
A	4	2	2	3
B	2	2	1	2
C	2	1	2	1
D	3	2	1	3

Converting Counts to Links

The potential new neighbors of a page:

- N pages with highest counts
- pages having count above threshold V
  - absolute for all pages
  - varying per page

Problems:

- Count Table is still O(P^2)
- All counts are non-zero
  - due to T3

Solutions

Can calculate separate table for each page
- slows algorithm
- saves space

Experimental Result:
Large transactions are rare

Discard large transactions
- may lose information

Experiments

- Coded using Java
- Run on Mills monthly logs
- Only HTML files (no JPEG, CGI, etc.)
- 11000 distinct URLs
  - includes URL tags within a file
  - 60M possible pairs

Transaction identified as:
- multiple accesses
- short time
- same user == same IP address

Experimental Questions

Is the counts table sparse?
- hope so

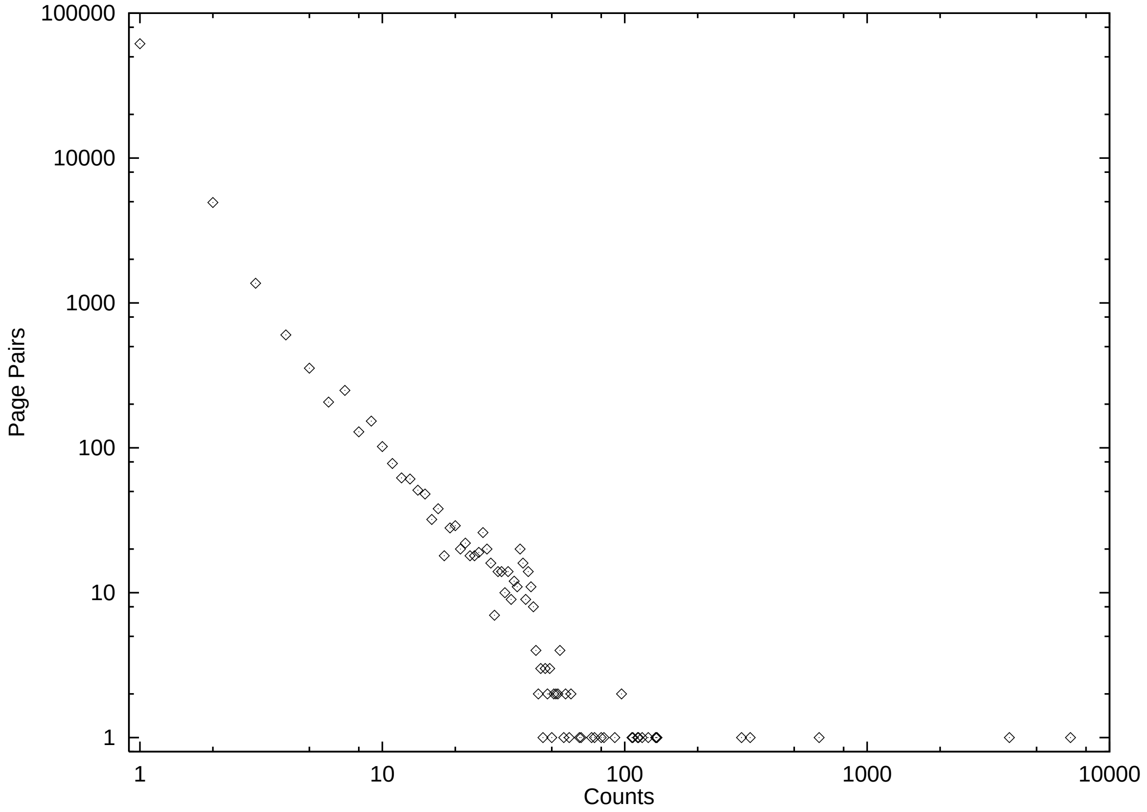
What is the distribution of transactions?
- hope for no large transactions

What is the impact of ignoring large transactions?
- hope for no effect

Is the Counts Table Sparse?

Of 11M possible pairs of pages:

70K pairs with non-zero count

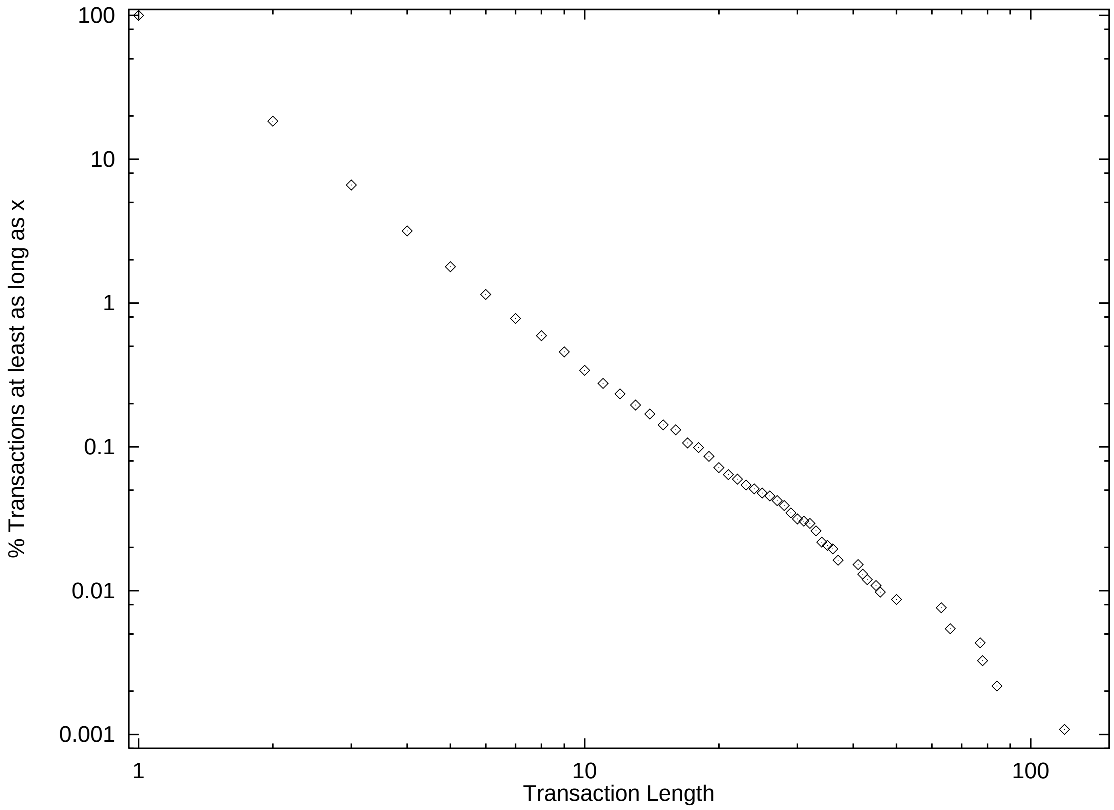


Of pairs with non-zero count

- avg. count/page is low (1.67)
- two outliers
- otherwise exponential drop
  + few pages have high count

Conclusion: Counts Table is Sparse

Transaction Size Distribution



Transactions
- most are very short
  + less than 1% are longer than six URLs
- usually focussed on single goal
  + linking results is reasonable
- almost all transactions under 100 URLs

Conclusion: Long Transactions are Rare

Discussion

DN algorithm determines related pages

Quadratic worst-case, but near linear in practice
- scalable

Algorithm output can be used to create dynamic list of top unlinked destinations for each page

Parting Questions

Is there a useful (to us) distinction between knowledge and information?

Is it possible to behave "intelligently", even if we don't know what we're doing?

Is it OK to do that?

Should we focus on understanding the pieces before trying to mesh them together?

Should we use Metric or English measurements when sending probes to Mars?