“Winnowing: Local Algorithms for Document Fingerprinting”

Alexander Aiken
Review of Text Matching Algorithms

- **Hashes/Checksums**
  - Compute once, compare many times

- **String Algorithms**
  - do exact substring to text match

- **LCS**
  - find the longest common subsequence with gaps

- **Regular Expression (Reg Exps)**
  - Exact Substrings, Wild Cards, Or constructs

Source: "Winnowing: Local Algorithms for Document Fingerprinting"
New Problem

- “Accurately measuring copying, including small partial copies, within a large set of documents.”
  - Hashes can be defeated by altering a single character or even bit
  - LCS can be defeated by simply reordering text
    - the LCS will be the single largest “atomic” block
  - String Matching and Reg Exps too expensive
    - Break a document into pieces
    - match pieces against all other documents
    - How can you deal with substrings of different lengths?

Source: "Winnowing: Local Algorithms for Document Fingerprinting"
Document Fingerprinting

• Given a set of documents, we want to find the substring matches between them that satisfy two properties
  – If there is a substring match at least as long as the guarantee threshold, \( t \), then this match is detected
  – No detection of matches shorter than noise threshold \( k \).

• Note: \( k \leq t \) and as \( k \) increases
  – Matches less likely due to coincidence
  – Limit sensitivity to document component reordering
  – Parameterization

Source: "Winnowing: Local Algorithms for Document Fingerprinting"
Document Fingerprint Example

A do run run run, a do run run
(a) Some text.

adorunrunrunadorunrun
(b) The text with irrelevant features removed.

• Step 1:
  – Eliminate context irrelevant features.
  – In text – Spaces, Capitalization, and punctuation.
Document Fingerprint Example

- Step 2:
  - Choose appropriate $t$ and $k$ values
  - $t = 8$ // Detect any substring of at least 8
  - $K = 5$ // Don't care about substrings less than len 5
Document Fingerprint Example

A do run run run, a do run run
(a) Some text.

adorunrunrunadorunrun
(b) The text with irrelevant features removed.

adoru dorun orunr runru unrun nrun r unru
unrun nruna runad unado nador adoru dorun
orunr runru unrun
(c) The sequence of 5-grams derived from the text.

77 74 42 17 98 50 17 98 8 88 67 39 77 74 42
17 98
(d) A hypothetical sequence of hashes of the 5-grams.

• Step 3:
  – Compute a rolling Rabin-Karp “style hash” for every 5 character substring
Document Fingerprint Example

(a) Some text.

adorunrunrunadorunrun
(b) The text with irrelevant features removed.

adoru doru runr runr runr runru unru runu runad undo ador dorun orunrunrun
(c) The sequence of 5-grams derived from the text.

77 74 42 17 98 50 17 98 8 88 67 39 77 74 42
17 98
(d) A hypothetical sequence of hashes of the 5-grams.

    (77, 74, 42, 17)  (74, 42, 17, 98)
    (42, 17, 98, 50)  (17, 98, 50, 17)
    (98, 50, 17, 98)  (50, 17, 98, 8)
    (17, 98, 8, 88)  (98, 8, 88, 67)
    (8, 88, 67, 39)  (88, 67, 39, 77)
    (67, 39, 77, 74)  (39, 77, 74, 42)
    (77, 74, 42, 17)  (74, 42, 17, 98)
(e) Windows of hashes of length 4.

    17 17 8 39 17
(f) Fingerprints selected by winnowing.

    [17, 3] [17, 6] [8, 8] [39, 11] [17, 15]
(g) Fingerprints paired with 0-base positional information.

Figure 2: Winnowing sample text.

- Hashing strings len 5 but care about strings len 8?
- The pattern “adorunru” is covered by (77,74,42,17)
- Hash and positional info from start of string (77,0) (74,1)...(17,3)
- We'll only really need to keep one from this “Window”
  - How do we choose?
A do run run run, a do run run
(a) Some text.

adorunrunrunadorunrun
(b) The text with irrelevant features removed.

adoru dorun orunr runru unrun nrunr runru
unrun nruna runad unado nador adoru dorun
orunr runru unrun
(c) The sequence of 5-grams derived from the text.

77 74 42 17 98 50 17 98 8 88 67 39 77 74 42
17 98
(d) A hypothetical sequence of hashes of the 5-grams.

\( (77, 74, 42, 17) \quad (74, 42, 17, 98) \)
\( (42, 17, 98, 50) \quad (17, 98, 50, 17) \)
\( (98, 50, 17, 98) \quad (50, 17, 98, 8) \)
\( (17, 98, 8, 88) \quad (98, 8, 88, 67) \)
\( (8, 88, 67, 39) \quad (88, 67, 39, 77) \)
\( (67, 39, 77, 74) \quad (39, 77, 74, 42) \)
\( (77, 74, 42, 17) \quad (74, 42, 17, 98) \)
(e) Windows of hashes of length 4.

17 17 8 39 17
(f) Fingerprints selected by winnowing.

[17, 3] [17, 6] [8, 8] [39, 11] [17, 15]
(g) Fingerprints paired with 0-base positional information.

Figure 2: Winnowing sample text.

Given a set of hashes \( h_1 \ldots h_n \) representing a document
Let the window size \( w = t - k + 1 \)
Each position \( 1 \leq i \leq n - w + 1 \) in this sequence defines a
window of hashes \( h_i \ldots h_{(i + w - 1)} \)
To maintain the guarantee of detection of all matches of length \( \geq t \)
At least one of the \( h_i \) hashes must be chosen

Source: "Winnowing: Local Algorithms for Document Fingerprinting"
Document Fingerprint Example

A do run run run, a do run run
(a) Some text.

adorunrunrunadorunrun
(b) The text with irrelevant features removed.

adoru dorun orunr runru unrun nrunr runru unrun nruna runad unado nador adoru dorun orunr runru unrun
(c) The sequence of 5-grams derived from the text.

77 74 42 17 98 50 17 98 8 88 67 39 77 74 42 17 98
(d) A hypothetical sequence of hashes of the 5-grams.

(77, 74, 42, 17) (74, 42, 17, 98)
(42, 17, 98, 50) (17, 98, 50, 17)
(98, 50, 17, 98) (50, 17, 98, 8)
(17, 98, 8, 88) (98, 8, 88, 67)
(8, 88, 67, 39) (88, 67, 39, 77)
(67, 39, 77, 74) (39, 77, 74, 42)
(77, 74, 42, 17) (74, 42, 17, 98)
(e) Windows of hashes of length 4.

17 17 8 39 17
(f) Fingerprints selected by winnowing.

[17, 3] [17, 6] [8, 8] [39, 11] [17, 15]
(g) Fingerprints paired with 0-base positional information.

Figure 2: Winnowing sample text.

- **Definition 1 (Winnowing).** In each window select the minimum hash value. If there is more than one hash with the minimum hash value, select the rightmost occurrence. Now save all the selected hashes as the fingerprints of the document.

Source: "Winnowing: Local Algorithms for Document Fingerprinting"
Why is this Selection Algorithm good?

• Assume a hashes are random numbers
• Each window overlaps the previous window and only adds one new random number
• The probability is low that when adding a new random number to a list that number is smaller than the ones already in that list
• Thus, in practice need to store less than 1 value from each window

Source: "Winnowing: Local Algorithms for Document Fingerprinting"
Formal Definition of a Local Algorithm

Definition 2 (LOCAL ALGORITHMS). Let $S$ be a selection function taking a $w$-tuple of hashes and returning an integer between zero and $w-1$ inclusive. A fingerprinting algorithm is local with a selection function $S$, if, for every window $h_i, ..., h_{(i+w-1)}$ the hash at position $i + S(h_i, ..., h_{(i+w-1)})$ is selected as a fingerprint.

Note: This is a greedy heuristic, not a greedy property.

Source: "Winnowing: Local Algorithms for Document Fingerprinting"
Correctness of Local Algorithms

Any matching pair of substrings of length at lest t is found by any local algorithm.

Proof:
Assume in two different documents there exists a common textstring $TS$ with $\text{length}(TS) \geq t$.
The hashes generated by the $k$-grams will span at least one window, $W$, whose fixed element length is $w = t - k + 1$. Since the selection function $S$ is a function of the contents of a single window, $S$ will select the same fingerprint in both documents.

Example: If we had "abcdefghij" with $k = 5$, $t = 8$, and thus $w = 4$ our $k$-grams would be abcde bcdef cdefg defgh efghi fghij.

A notional set of hashes could be 47 56 60 45 70 23

With $w = 4$ six hashes would have to be split into at least two windows.
Cost Analysis

- Cost here is measured in “Density” – Function of Space

- “Density is the expected fraction of fingerprints selected from among all the hash values computed, given random input.”

Any local algorithm with noise threshold $k$ and guarantee $t = w + k - 1$

has a lower bound density $d \geq \frac{1.5}{(w+1)}$.

Winnowing has a density of $d \geq \frac{2}{(w+1)}$, which is with 33% of the theoretical lower bound.

Source: "Winnowing: Local Algorithms for Document Fingerprinting"
Comparing two Documents

- Compute sets of fingerprints for documents
  - Each fingerprint is a hash and the location of the hash in the actual document
- Intersection of hashes becomes list of “potential” hits
  - Potential because of hash collisions
  - Like in Rabin Karp must compare the actual text's to validate hits

Source: "Winnowing: Local Algorithms for Document Fingerprinting"