Some methods for blindfolded record linkage

Tim Churches[1]*, Peter Christen[2]

[1] Centre for Epidemiology and Research, Population Health Division, New South Wales Department of Health, Locked Mail Bag 961, North Sydney NSW 2059, Australia

[2] Department of Computer Science, Australian National University, Canberra ACT 0200, Australia

Appendix 2 – An impractical protocol for the minimum-knowledge calculation of the Jaro and Winkler metrics

The following protocol for minimum-knowledge calculation of the Jaro and Winkler string similarity (described in detail below) is known to be infeasible. We include it in the hope of stimulating further work on the blindfolded calculation of these comparators.

The Jaro comparator and the Winkler modification of it are widely used in current record linkage systems – the Winkler comparator is used by both the US Bureau of the Census and Statistics Canada in their Canlink/GRLS (Generalised Record Linkage System) software[i,ii,iii]. Using the notation of Winkler, the Jaro comparator is defined as:

$$\Phi = W_A \bullet c / d_A + W_B \bullet c / d_B + W_\tau \bullet (c-\tau)/c$$

where

W_A = weight associated with characters in the first of two strings
W_B = weight associated with characters in the second of two strings
W_τ = weight associated with transcriptions
d_A = length of first string
d_B = length of second string
τ = number of transpositions of characters in common, and
c = number of characters in common in the pair of strings

If c= 0, then $\Phi = 0$. Two characters are considered in-common if and only if they are no further than g character positions apart, where $g = \max(d_A,d_B)/2 - 1$. Characters in-common are flagged as “assigned”; remaining characters as “unassigned”. The number of transpositions is computed by comparing the first assigned character on one string with the first assigned character on the other string. If the characters are not the same, half of a

i Winkler WE: **Record linkage software and methods for merging administrative lists.** *Statistical Research Report Series No. RR2001/03.* Washington DC: US Bureau of the Census; 2001. Available at http://www.census.gov/srd/papers/pdf/rr2001-03.pdf

ii US Bureau of the Census: *Record Linkage Software User Documentation.* Washington DC; 1999. Available at http://www.alw.nih.gov/Other_resources/amgtech/dsc/directory/docs/US%20Census%20Bureau%20Record%20Linkage%20SW%20user%20Documentation.pdf

iii Statistics Canada: *Canlink (formerly GRLS) (software)*. Ottawa: Statistics Canada; 2001.

transposition has occurred. Subsequently assigned characters are compared in a similar fashion, and the total number of transpositions is accumulated until there are no more assigned characters to compare. Note that each string has the same number of assigned characters, by definition. The values of W_A, W_B and W_τ are typically set to 1/3.

The Winkler comparator modifies the Jaro metric if the first few characters of the two strings agree. For i = 1,2,3,4:

$$\Phi_W = \Phi + i \bullet 0.1 \bullet (1-\Phi) \text{ if the first } i \text{ characters agree}$$

The following is a description of a minimal-knowledge version of the Jaro and Winkler comparators, using Rivest's concept of “chaffing and winnowing” to hide information []. The actors are the same as in the previous protocols. The notation H(k,x) denotes the HMAC keyed hash digest of value x using key k.

The protocol consists of the following steps:

1. As in step 1 in Protocols 1 and 2, Alice and Bob mutually agree on: a secret random key, K_{AB}, which they share only with each other; a keyed hash transformation function (HMAC); and a standard protocol for pre-processing strings to render them in a standard form.

2. Alice pre-processes (normalises) the values in attribute A.a in the agreed manner and for each character, j, with position i in the normalised string of length d, she prepares a “wheat” tuple of the form:

   $$(\{A.record\_key\_digest, R_A\}_{PublicKeyD}, H(K_{AB}, j), i_A, d_A, R_A, H(K_{AD}, R_A))$$

   For each character, j, Alice also prepares a set of “chaff” tuples, of the form:

   $$(\{A.record\_key\_digest, R_A\}_{PublicKeyD}, H(K_{AB}, \forall j'), i_A, d_A, R_A, H(K_{rand}, R_A))$$

   where $\forall j'$ is each character in the alphabetic complement of j (for example, if j is “f” then $\forall j'$ takes the value of every other letter except “f”), and both R_A and K_{rand} are arbitrary, but different random keys which are used by Alice only once (in other words, they are nonces). Alice sends this set of wheat and chaff tuples, in randomised order, to Carol.

3. Bob does the same with the values of his attribute B.a, sending Carol a randomised mixture of tuples of the following forms:

   $$(\{B.record\_key\_digest, R_B\}_{PublicKeyD}, H(K_{AB}, j), i_B, d_B, R_B, H(K_{BD}, R_B))$$
   $$(\{B.record\_key\_digest, R_B\}_{PublicKeyD}, H(K_{AB}, \forall j'), i_B, d_B, R_B, H(K_{rand}, R_B))$$

4. For each of the tuples received from Alice, Carol locates “in common” tuples in the set she has received from Bob – that is, any tuples from Bob in which the second element, $H(K_{AB}, j)$, matches the second element of the tuple from Alice which is currently under

consideration, and in which the character position i_B is within g characters of i_A. Tuples which meet these criteria are flagged as “assigned”. Note that Carol is unable to determine the original value of character j from which $H(K_{AB}, j)$ was derived, nor is she able to distinguish “wheat” tuples from “chaff” tuples.

5. Carol performs the same actions for each of the tuples she has received from Bob, with respect to the set of tuples from Alice. Carol now has two sets of “assigned” tuples, each set containing the same number of elements.

6. Carol performs an inner join of these two sets, using i_A and i_B as the join key – in other words, Carol forms the Cartesian product of all assigned tuples which share the same character position in their respective source strings.

7. From each of the resulting joined tuples, Carol extracts the following elements and sends them as a tuple to David:

   $(\{R_A, A.record\_key\_digest\}_{PublicKeyD}, \{R_B, B.record\_key\_digest\}_{PublicKeyD}, agreement\_flag, i, d_A, d_B, R_A, R_B, H(K_{AD} \mid K_{rand}, R_A), H(K_{BD} \mid K_{rand}, R_B))$

   where agreement_flag has a value of 0.5 if the pair of $H(K_{AB}, j)$ values agree, or 0 if they do not, and $K_{AD} \mid K_{rand}$ .means either K_{AD} or K_{rand} (Carol is unable to tell which).

8. For each of the tuples he receives, David uses his knowledge of K_{AD} and K_{BD} to compute $H(K_{AD}, R_A)$ and $H(K_{BD}, R_B)$, and discards those tuples which contain chaff or a mixture of wheat and chaff.

9. David uses his private key to decrypt the values of A.record_key_digest and B.record_key_digest. For each distinct pairing of A.record_key_digest and B.record_key_digest in the remaining tuples, David obtains the number of transpositions, τ, by summing the values of agreement_flag, and the number of characters in common, c, by counting the number of tuples within each key digest pairing. The values of d_A and d_B will be the same for all tuples within each record key digest pairing. David now has all the information he needs to compute the value of the Jaro and/or Winkler metrics for each record key digest pair.

In this protocol, Carol learns the number of characters in each of Alice's and Bob's original values, but nothing more. Due to the chaffing, she is unable to determine what the actual values are, nor is she able to determine which of Alice's values have characters in common with Bob's values, or *vice versa*. Unfortunately, the chaffing also means that in step 5 of the protocol, from Carol's perspective, each tuple from Alice will appear to be assigned to a large number of tuples from Bob, and vice versa. The reason is that there is either a wheat or chaff tuple for every possible letter in each position of each original string. In step 6, a combinatorial explosion results in a huge number of tuples – for example, several million from only ten original values. Unfortunately this renders the protocol impractical. It may be possible to reduce the amount of chaffing to improve the efficiency of the protocol, without unduly

reducing its security. The other major flaw is that it is completely vulnerable to collusion between or simultaneous compromise of Carol and David.