HHS Public Access

Author manuscript

IEEE Internet Comput. Author manuscript; available in PMC 2019 March 01.

Published in final edited form as:

IEEE Internet Comput. 2018 ; 22(2): 32–41. doi:10.1109/MIC.2018.112102542.

Perfectly Secure and Efficient Two-Party Electronic-Health-Record Linkage

Feng Chen[1,*,¶], **Xiaoqian Jiang**[1,¶], **Shuang Wang**[1,¶], **Lisa M. Schilling**[2], **Daniella Meeker**[3], **Toan Ong**[2], **Michael E. Matheny**[4,5], **Jason N. Doctor**[6], **Lucila Ohno-Machado**[1], and **Jaideep Vaidya**[7]

[1]Health System Department of Biomedical Informatics, UC San Diego, La Jolla, CA, 92093

[2]Department of Medicine, University of Colorado, Anschutz Medical Campus, CO, 80045

[3]Keck School of Medicine, University of Southern California, Los Angeles, CA 90089

[4]Geriatric Research Education and Clinical Care Service, Tennessee Valley Healthcare System VA, Nashville, TN 37212

[5]Departments of Biomedical Informatics, Medicine, and Biostatistics, Vanderbilt University Medical Center, Nashville, TN 37235

[6]Schaeffer Center for Health Policy & Economics, University of Southern California, Los Angeles, CA 90033

[7]Management Science & Information Systems Department, Rutgers University, Newark, NJ 07102

Abstract

Patient health data are often found spread across various sources. However, precision medicine and personalized care requires access to the complete medical records. The first step towards this is to enable the linkage of health records spread across different sites. Existing record linkage solutions assume that data is centralized with no privacy/security concerns restricting sharing. However, that is often untrue. Therefore, we design and implement a portable method for privacy-preserving record linkage based on garbled circuits to accurately and securely match records. We also develop a novel approximate matching mechanism that significantly improves efficiency.

Keywords

EHR linkage; Privacy preserving; Secure multi-party computation; Garbled circuit

Introduction and Significance

Initiatives such as the Patient Centered Outcomes Research Institute's Clinical Data Research Network [1], [2] and the National Institutes of Health eMERGE [3] share the

*To whom correspondence should be addressed. f4chen@ucsd.edu.

¶share the first authorship

[4]Intel® Software Guard Extensions (Intel® SGX): https://software.intel.com/en-us/isa-extensions/intel-sgx

challenge that a single entity rarely holds all of the information necessary to conduct research or to provide clinical care. Patient health data are naturally spread across different sources due to visits at various clinics, hospitals, and pharmacies that often do not share the same information system (e.g. EHR implementations) or exchange data. Linking patient-level data from various sources allows physicians and researchers to have more comprehensive data for clinical decision-making and research. However, due to valid data privacy concerns, health entities may be reluctant to exchange identifiers.

Creating a collection of patient-level records with geographically distributed and longitudinal data from multiple entities that can be used for research is a significant challenge. In recent years the importance of this problem has been realized resulting in innovative solutions. However, existing methods still suffer from security challenges. For example, Kho et al. [4] proposed a seeded hashing on pre-determined fields. Essentially, all parties convert the record fields used for linkage into corresponding random-looking strings that are then matched by a central party. This provides protection against external attackers, since the random-looking strings cannot be easily reversed into the actual record values. However, such solutions that rely on a third party (honest broker) are often not secure [5], [6]. If the trusted third party has access to hash data and the shared encryption key, the hashed value of the linkage variables would be susceptible to different types of attacks (e.g., dictionary attack, frequency analysis), especially if the domain of the information is small or well-known[7]. To address this issue, alternative methods using a semi-trusted third party [8] or commutative encryption [9] have also been proposed, but they still make weaker security assumptions. For example, Durham et al.[10] use Bloom filters and are robust to some well-known attacks for Bloom filters such as frequency-based cryptanalysis, but a third party is often required for coordinating field weight and linkage. Vatsalan et al.[11] do not rely on a third party, but the framework requires optimizing some critical parameters, which are difficult in practice. Yakout et al.[12] propose a two-step linkage method that does not rely on a third party, however, some privacy may be leaked during the first step, which is used to get the approximate linkage.

In this work, we propose a completely secure deterministic record linkage method for two parties. Our method guarantees that no extra information is revealed to the two parties beyond the results of the matching, and no information is revealed to any other party (with 80-bit security level). We provide a basic approach based on a classical implementation of garbled circuits and a computationally more efficient approach using a filtering strategy. Unlike methods requiring Honest Broker intermediaries, our approach does not require an external third party. The significance of this work is that it provides a proof of concept for secure two-party record linkage using state-of-the-art garbled circuit methods that can be readily adopted in small-to-medium scale linkage tasks with a strong security guarantee.

Adversary Model

We assume that the parties want to collaborate and will adhere to terms of use. This “honest-but-curious” model assumes that one of the parties may inspect (or be unintentionally exposed to) private data, while carrying out agreed upon algorithms faithfully, i.e., parties are semi-honest, which is typical in healthcare settings. Individuals in this scenario should

be protected from accidental exposure to protected health information. On the other hand, if trust levels are low or data are posted publicly, assuming that both parties are malicious may be required to obtain desired levels of security. Our work can be extended to such settings too, though with much greater computation cost.

Related Methods

We follow the deterministic approach followed by Kho *et al.* [4] which has been shown to perform well in terms of both accuracy and efficiency with real data from 6 institutions in the Chicago metropolitan area. Kho's method proceeds as follows: For two records (A and B) fromdifferent sites, the following four match criteria are used:

- **a.** Seeded HashID of (First Name + Last Name + Date of Birth),
- **b.** Seeded HashID of (Date of Birth + SSN),
- **c.** Seeded HashID of (Last Name + SSN), or
- **d.** *Seeded HashID of (Three Letters First Name + Three Letters Last Name + Soundex*[13] *First Name + Soundex Last Name + Date of Birth + SSN).*

For flexibility, each criterion is assigned a weight (Kho *et al.* use a weight of 1.01 for each criterion). If two records A and B provide the same output for the same rule, then this rule's weight will be added to the matching score, i.e., $s = \sum_{i=1}^{4} \left(r_i(A) = = r_i(B)\right) w_i$, where the $r_i(\cdot)$is the i-th rule mapping function, and w_i is the i-th weight. The matching score, s, is used to determine if A and B are a match based on the following criteria:

$$\{s > T, \ match; \ s \leq T, no \ match\}$$

Here T is the given threshold (set to 1 by [4]). Note that in the actual implementation, the comparison of the rule outputs is done by an independent third party, with the seeded HashID provides security, except in the case of collusion.

Garbled Circuits

Garbled circuits (GC) [14] is a cryptographic technique that enables the computation of a function where the inputs are held by two different parties in a completely secure manner. It is a general technique that can be used to compute *any* function representable in the form of a circuit. Although restricted to two parties, it supports a rich class of operations.

The basic idea is to represent the function to be computed as a basic logic circuit, which is then interactively computed by the two parties. One party creates a garbled circuit corresponding to the original circuit taking into account all of the potential inputs of the other party, while the other party evaluates this garbled circuit using its actual input. Thus, the first party plays the role of the (circuit) generator, while the second plays the role of the (circuit) evaluator. Essentially, the generator designs a garbled circuit to compute the required function while encoding its input into the circuit itself. This is then sent to the evaluator, which first retrieves the evaluation key (garbled input) corresponding to its actual

input from the generator and then evaluates the circuit using the retrieved keys. Figure 1 depicts this process. Without loss of generality, we assume that Site 1 plays the role of the generator while Site 2 plays the role of the evaluator. We now explain this in more detail.

To securely compute a function $f(x, y)$, where x is the input from the generator and y is the input from the evaluator, the generator will first translate the function $f(x, y)$ into a Boolean circuit, which is composed of basic Boolean gates (e.g., AND/OR/XOR). For each input and output of each gate, the generator creates a pair of keys (garbled inputs), one to be used if the value of the input/output is 1 while the other to be used if the value of the input/output is 0. Now, given the truth table (input/output relationship) for each gate, the generator creates a garbled truth table by using the appropriate input keys to encrypt the corresponding output key as per the truth table. This truth table is called garbled since it only contains encryptions for the output which cannot be used to determine the input. The generator can send all of the garbled truth tables along with its garbled inputs to the evaluator. However, the evaluator still needs to get the key (garbled input) corresponding to its actual input, without telling the generator what the input is. This can be done using a standard cryptographic primitive known as Oblivious Transfer (OT) [15], which allows a party (in this case the evaluator) to retrieve a single message (the corresponding key) from the other party (the generator) without revealing anything to the generator. Once the evaluator has retrieved its garbled inputs, it can then evaluate the circuit.

For example, assume that x and y are both a single bit (0 or 1) and the function $f(x, y)$ returns 0 if the two are equal and 1 if they are not (i.e., $f(x, y)= x\hat{}y$). Such a function can be easily encoded as a single XOR gate since an XOR gate 1 only if its inputs are different. Figure 2(a) depicts this simple inequality circuit. Figure 2(b) shows the garbled inputs (keys) that are created by the generator for the XOR gate. As can be seen from its truth table (in Figure 2(c)), the XOR gate returns the correct output based on the comparison of its single bit inputs. Figure 2(d) gives the corresponding garbled truth table created.

Each row is simply the appropriate encryption of the corresponding output key. For example, if x and y were both 0, since the actual output should be 0, the corresponding row of the garbled truth table contains the output key corresponding to 0 (k_{z0}) encrypted by the keys corresponding to its input (k_{x0} and k_{y0}). Note that in actuality the rows of the garbled truth table are permuted to ensure that the evaluator cannot identify what inputs correspond to each row.

Now, as mentioned above, the generator transfers the garbled truth tables for all of the gates in the circuit to the evaluator, along with the generator's garbled input (the keys corresponding to its input). For example, when the generator's input (X) is 1, it sends the key k_{x1} to the evaluator. Note that since the keys k_{x0} and k_{x1} cannot be distinguished, the evaluator cannot figure out what the generator's input is from the key it sees. The evaluator then retrieves its garbled input using Oblivious Transfer. Suppose the evaluator's input (Y) is 0. Then, the evaluator would obliviously retrieve the key k_{y0} from the generator.

Now, the evaluator uses the garbled inputs (keys) corresponding to the inputs of both the generator and the evaluator to evaluate the circuit. This is done by decrypting each row of

the garbled truth table – note that only one row will be correctly decrypted giving the key corresponding to the correct output, even though the evaluator does not know what the output is. Thus, in our example only the row corresponding to keys k_{x1} and k_{y0} is decrypted giving the key k_{z1}. This continues for all of the gates with the outputs of the corresponding steps used to decode the following garbled truth tables until the evaluator receives the final output of the circuit. For the final gate, the generator also tells the evaluator what bits the output keys correspond to. Thus, for our simple inequality circuit above, the generator tells the evaluator that k_{z0} corresponds to 0, and k_{z1} corresponds to 1. With this information, the evaluator can now figure out the answer of the circuit. Thus, in the example above, since the evaluator receives k_{z1}, it can figure out that the output of $f(x, y)=1$. The detailed proof of security can be found in the manuscript by Yao [14]. While this approach only works for the honest-but-curious model, there are refinements which allow it to work in the malicious model too.

Methods

We implemented and tested the garbled circuit approach[2] by building a multi-bit circuit that encodes all of the 4 match criteria checked by Kho et al. There is no need to compute the seeded HashID since no data is externally visible. For each matching rule, we convert the required fields into sufficiently long bit strings, concatenate them, and build a corresponding bit length comparator circuit for matching. For example, the first rule checks if the combination of first name, last name, and date of birth are the same. The first name is encoded using 96 bits (sufficient for any name under 12 ascii characters). The last name is encoded using 88 bits (sufficient for any name with up to 11 characters). The date of birth is encoded using 64 bits (sufficient for 2-digit day, 2-digit month, and 4-digit year). This gives us a total of 248 bits, which was sufficient for testing, but could be adjusted based on the usage environment. Computationally, the smallest possible encoding is preferable since the comparison circuit cost increases on a per-bit basis. Now, a 248-bit inequality circuit is built, similar to the 2-bit inequality circuit shown in Figure 3. The real circuit includes 248 XOR gates to check for equality among the 248 bits and multiple OR gates to coalesce the result. Similarly, circuits are also built to check the remaining match criteria.

Finally, the output of the 4 circuits is coalesced to give a single equality/inequality result. Now, for each record from Party A, and each record from Party B, the entire garbled circuit will be generated and evaluated. The output of this tells the parties only if the two records match or not. Since every pair of records is independently compared, this process can be completely parallelized.

Approximate matching to improve computational efficiency

To improve the efficiency of the basic garbled circuit approach, we designed a simple but efficient approximation strategy which is motivated by the intuition that a record only has one other true match in any other database, and it is unnecessary to compare it against many other records (if they are obviously not going to match). Thus, most of the comparisons

[2]The software can be found at: https://github.com/achenfengb/RecordLinkage.git

between the pairs of records are unsuccessful – while there are ways to speed up comparisons by using other mechanisms such as stopping the comparison as soon as an inequality is detected, these typically compromise on security since the timing can lead to indications of what is the same or different across the records. Instead, what we want is a fully secure technique that still takes significantly less time than the basic garbled circuit strategy.

The key observation is as follows: if two records do not match on any of the 4 criteria mentioned above, they also will not match in terms of simpler predicates. For example, suppose the match parameters are simply the first and last name. Instead of directly checking these, we check if their length matches for both records, and select the matching records as potential candidates. Thus, for example, instead of directly matching “Emily Abbott” with “Barbara Mcklewski” we first check “56” with “79”. Since they do not match, there is no need to check the full names. The benefit of this is that the comparison is carried out over much smaller sizes (for example, 2 digits as opposed to over 20 characters). While there can be spurious matches in the first step (for example “Emily Abbott” and “Sofia Potter” which both give “66”), they will be likely to be filtered out when we consider other attributes. Even though this incurs a potential risk of introducing false positives (records do not actually match), the approximate comparison can be significantly faster than comparing the raw records in a pairwise manner. It is also possible to create a more complicated circuit that compares the potential matches to output only true matches.

Instead of using ad-hoc predicates such as length, the actual implementation uses hashing. First, the records are hashed using a SHA-256 hash. Next the first few bits of the hash are compared as the preliminary comparison step (small number of bits correspond to faster comparison but higher risks of introducing false positives).

Figure 4 illustrates how the whole predicate matching is converted into an approximate comparison based on their hashed values. Note that the use of a good hash function is critical. A good hash function will uniformly distribute all of the records, thus automatically giving us a reduction in potential records to match. For example, if we use only 3-bit of the hash, then each 3-bit pattern should have approximately $1/8^{th}$ of the records. In general, if s bits of the hash are used, we expect to have approximately $1/2^s$ of the records matching a particular pattern. Clearly, larger s is better to filter more records. However, larger s also results in more expensive calculation (and more memory cost), thus giving us a parameter that can be suitably tuned for optimum performance based on the dataset. While we do not provide bounds on the number of false positives, it may be possible to formally analyze this from a bloom filter perspective, which we plan to explore, in the future. As the experiments show, even using the initial s bits gives very good performance.

Note that the above filtering strategy does not negatively impact security, since the output for every pair of records is still only whether they match or not. When the records do not match, the computation is swift, however, it still does not leak any information regarding what part of the records do not match, etc., due to the use of the one-way hash function.

Results and Conclusion

We implemented the proposed algorithms in Java in a two-party distributed setting. The code is completely portable and has been successfully tested on Windows, MacOS, and CentOS, with only some variations in performance (running time) due to different Java interpreter implementations. The results below are on a Windows 10 with Intel i7-6820HK 2.7 GHz CPU, a 48 GB memory, and 1.0 Gbps network adaptor. The synthetic datasets used were generated using the Mockaroo synthetic data generator[3]. The data in each field in these datasets were randomly corrupted (10% to 20%) to emulate basic typographical errors.

Note that the accuracy of the exact matching algorithm is exactly the same as that of [4]. Therefore, we compare the loss in accuracy due to the filtering strategy while showing the performance improvement. Figure 5(a) reports the total running time for several datasets of different sizes. For the cases of 5k × 5k and 10k × 10k, we cannot run the exact matching using one gigantic circuit due to memory limitations, and therefore, estimate the time (i.e., from multiple runs of single execution of 500 × 5k and 500 × 10k linkage). Note that the two parties do not need to have the same number of records. The figure clearly demonstrates the running time of using approximate matching based on the filtering mechanism is significantly faster (time is depicted in log-scale). Note that the improvement in computation time is as expected, since smaller input sizes for the comparison circuit automatically lead to smaller computation time. Indeed, the different hash sizes are somewhat comparable in efficiency, though they all provide over a 10-fold improvement in efficiency over the base method. However, the price paid is in terms of accuracy, since smaller hash sizes lead to more collisions (false positives in terms of the match), as discussed below.

Table 1 shows the linkage results. Using the full fields the same results are achieved as the underlying matching algorithm. Approximate matching is much faster but can introduce false positives (FP). It is clear that all of the algorithms obtain accurate results for the two smallest datasets. This is due to the fact that no hash collisions occur even for smaller hash sizes. However, for larger datasets, shorter hashes will result in more collisions thus leading to more false positives. For example, for 10,000 × 10,000 size, the 24-bit hashes input results in 26 FP cases beyond the baseline. If we increase the hash inputs just 4 bits to 28 bits, only 1 additional FP arises. With 36-bit hashes, there are no additional FPs beyond the underlying method.

Figure 5(b) shows the maximum memory cost (MMC) of approximate matching. As expected, the MMC grows when the size of patient linkages increase. However, for the same linkage size, increasing hash bit length does not increase the MMC significantly. Furthermore, it can be seen that the base method (without the use of hashing) requires significantly larger memory. Indeed, for the two largest cases (5K × 5K and 10K × 10K), 48GB was insufficient and therefore in order to get the running time, we had to carry out the experiment over partitions of the data instead of the entire dataset all at once. For the memory usage, we have estimated the exact memory that would be required if the entire dataset were to be used in a single pass. Again, this demonstrates the viability of the hash

[3]Available at: www.mockaroo.com

based filtering, since the memory usage is the primary constraint in carrying out such computation for larger dataset sizes.

To conclude, using garbled circuits provides perfect security, and when coupled with approximate matching can be reasonably efficient, though restricted to only two parties. In fact, multi-party matching can be implemented using a pairwise strategy, where comparisons are done in parallel using a binary tree, which will results in an additional log-scale factor on complexity, though we leave the analysis of this to future work.

In addition, the computational complexity of garbled circuit framework is much higher than that of the insecure plaintext method. For our laptop with 4 cores and 48 GB memory, to guarantee at least 80-bit security, the size 10K × 10K with 36-bit hash requires almost 3 hours. Bigger data sizes require more memory or powerful workstations. Alternatively, a secure hardware based method such Intel SGX 4 can also be leveraged, which we plan to explore in the future.

Acknowledgments

Research reported in this publication was supported by the National Institutes of Health under awards U54HL108460, U01EB023685, R01GM118574, R01GM124111, R01HG008802, R01GM114612, R21LM012060, R01GM118609, R00HG008175, by the National Science Foundation under awards CNS-1422501 and CNS-1564034 and by the Patient-Centered Outcomes Research Institute (PCORI) under Contract CDRN-1306-04819. The content is solely the responsibility of the authors and does not necessarily represent the official views of the agencies funding the research.

References

1. Fleurence RL, Curtis LH, Califf RM, Platt R, Selby JV, Brown JS. Launching PCORnet, a national patient-centered clinical research network. J Am Med Informatics Assoc. Jul; 2014 21(4):578–582.
2. Ohno-Machado L, et al. pSCANNER: patient-centered Scalable National Network for Effectiveness Research. J Am Med Informatics Assoc. Jul; 2014 21(4):621–626.
3. Gottesman O, et al. The Electronic Medical Records and Genomics (eMERGE) Network: past, present, and future. Genet Med. Oct; 2013 15(10):761–771. [PubMed: 23743551]
4. Kho AN, et al. Design and implementation of a privacy preserving electronic health record linkage tool in Chicago. J Am Med Informatics Assoc. Sep; 2015 22(5):1072–1080.
5. Boyd AD, et al. The University of Michigan Honest Broker: a Web-based service for clinical and translational research and practice. J Am Med Inform Assoc. 2009; 16(6):784–91. [PubMed: 19717803]
6. Dhir R, et al. A multidisciplinary approach to honest broker services for tissue banks and clinical data. Cancer. Oct; 2008 113(7):1705–1715. [PubMed: 18683217]
7. Ajmani, S., Morris, R., Liskov, B. A trusted third-party computation service. 2001.
8. Lazrig I, Moataz T, Ray I, Ong T. Privacy Preserving Record Matching Using Automated Semi-trusted Broker. Data Appl Secur Priv XXIX. 2015; 9149:103–118.
9. Adam N, White T, Shafiq B, Vaidya J, He X. Privacy preserving integration of health care data. Annu Symp proceedings AMIA Symp. Oct.2007 2007:1–5.
10. Durham EA, Kantarcioglu M, Xue Y, Toth C, Kuzu M, Malin B. Composite Bloom Filters for Secure Record Linkage. IEEE Trans Knowl Data Eng. Dec; 2014 26(12):2956–2968. [PubMed: 25530689]
11. Vatsalan, D., Christen, P. An Iterative Two-party Protocol for Scalable Privacy-preserving Record Linkage. Proceedings of the Tenth Australasian Data Mining Conference; 2012. p. 127-138.
12. Yakout M, Atallah MJ, Elmagarmid A. Efficient and Practical Approach for Private Record Linkage. J Data Inf Qual. Aug; 2012 3(3):5:1–5:28.

13. Stephenson C. The Methodology of Historical Census Record Linkage: a User's Guide To the Soundex. J Fam Hist. Mar; 1980 5(1):112–115.
14. Yao, AC. Protocols for secure computations. 23rd Annual Symposium on Foundations of Computer Science (sfcs 1982); 1982; p. 160-164.
15. Naor, M., Pinkas, B. Efficient Oblivious Transfer Protocols. Proceedings of the Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms; 2001; p. 448-457.

Biographies

Feng Chen (M'15) received his B.S. degree, M.S. degree and Ph.D. degree from China University of Geosciences-Beijing, Beihang University and the University of OklahomaTulsa, respectively. He was a postdoctoral researcher at DBMI, UCSD. His research interests include secure computation over EHR/genome data and pure mathematical problems.

Xiaoqian Jiang (S'06–M'10) received the Ph.D. degree in computer science from Carnegie Mellon University. He is currently an Associate Professor with the Department of Biomedical Informatics, University of California at San Diego. He is an Associate Editor of BMC Medical Informatics and Decision Making and serves as an Editorial Board Member of the Journal of American Medical Informatics Association. He was involved primarily in health data privacy and predictive models in biomedicine. He was a recipient of the Distinguished Paper Award from American Medical Informatics Association, Clinical Research Informatics Summit in 2012 and 2013, respectively.

Shuang Wang (S'08–M'12) received the B.S. degree in applied physics and the M.S. degree in biomedical engineering from the Dalian University of Technology, China, and the Ph.D. degree in electrical and computer engineering from the University of Oklahoma, OK, USA, in 2012. He was worked as a postdoc researcher with the Department of Biomedical Informatics (DBMI), University of California, San Diego (UCSD), CA, USA, 2012 – 2015. Currently, he is an assistant professor at the DBMI, UCSD. His research interests include machine learning, and healthcare data privacy/security. He has published more than 60 journal/conference papers, 1 book and 2 book chapters. He was awarded a NGHRI K99/R00 career grant. Dr. Wang is a senior member of IEEE.

Lisa M. Schilling, MD, MSPH is a practicing, board-certified internist. She is also board certified in clinical informatics. She is co-director of the Data Science to Patient Value Initiative at the University of Colorado School of Medicine. She focuses on clinical and research informatics, with an interest in making clinical data valuable for secondary use, including issues of interoperability, data modeling, and data quality.

Daniella Meeker, PhD is an Assistant Professor at the USC Keck School of Medicine and Directs the Informatics Program for the Southern California Clinical and Translational Science Institute. She is an Information Scientist at RAND Corporation.

Toan Ong is an Assistant Professor at the University of Colorado Anschutz Medical Campus – School of Medicine. He has a PhD in Computer Science and Information Systems. He has been involved in national projects funded by AHRQ and PCORI such as SAFTINet,

PEDSNet or pSCANNER. He has experience with linking, designing, harmonizing and loading large-scale healthcare datasets. In pSCANNER, Dr. Ong has been involved in developing secure centralized and distributed identity linkage methods. In collaborations with other team members, he has been working on designing and incorporating different identify linkage solutions into the pSCANNER data infrastructure. Dr. Ong's other research interests are schema mapping, record linkage, data mining and natural language processing.

Michael E. Matheny, MD, MS, MPH, is a practicing general internist and medical informatician at Vanderbilt University and TVHS Veteran's Administration. He received a B.S. in Chemical Engineering and an M.D from the University of Kentucky, completed Internal Medicine residency training at St. Vincent's, Indianapolis, IN, and was an NLM Biomedical Informatics Fellow at Decision Systems Group at Brigham & Women's Hospital, Boston, MA during which time he completed a Master's in public health at Harvard University as well as a master's of science in biomedical informatics at MIT. His key focus areas include natural language processing, data mining and population health analytics as well as health services research in acute kidney injury, diabetes, and device safety in interventional cardiology. He is currently funded by VA HSR&D, PCORI, NHLBI, NHGRI, and AstraZeneca.

Jason N. Doctor is Associate Professor and Chair of the Department of Health Policy and Management at the University of Southern California's Price School of Public Policy. He also holds the Norman Topping Chair in Medicine and Public Policy and is the Director of Health Informatics at the USC Leonard D. Schaeffer Center for Health Policy & Economics. His research program centers on decision-making in healthcare and health informatics. Dr. Doctor specializes in behavioral economics and the use of choice architecture to affect policy in health and medicine.

Lucila Ohno-Machado, MD, MBA, PhD received her medical degree from the University of São Paulo and her doctoral degree in medical information sciences and computer science from Stanford. She is Associate Dean for Informatics and Technology, and the founding chair of the Health System Department of Biomedical Informatics at UCSD, where she leads a group of faculty with diverse backgrounds in medicine, nursing, informatics, and computer science. Prior to her current position, she was faculty at Brigham and Women's Hospital, Harvard Medical School and at the MIT Division of Health Sciences and Technology. Dr. Ohno-Machado is an elected fellow of the American College of Medical Informatics, the American Institute for Medical and Biological Engineering, and the American Society for Clinical Investigation. She serves as editor-in-chief for the Journal of the American Medical Informatics Association since 2011.

Jaideep Vaidya is the RBS Dean's Research Professor in the Management Science and Information Systems department at Rutgers University. He received the B.E. degree in Computer Engineering from the University of Mumbai, the M.S. and Ph.D. degree in Computer Science from Purdue University. His general area of research is in data mining, data management, security, privacy, and digital government. He has published over 140 technical papers in peer-reviewed journals and conference proceedings, and has received

several best paper awards. He is an ACM distinguished scientist and an IEEE senior member.

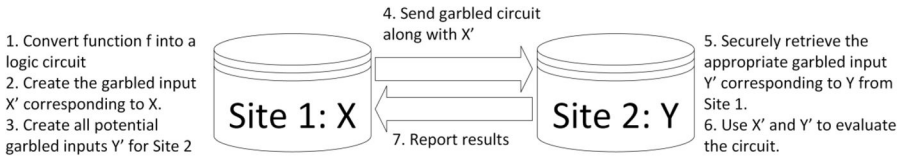

Figure 1.
Overall process flow

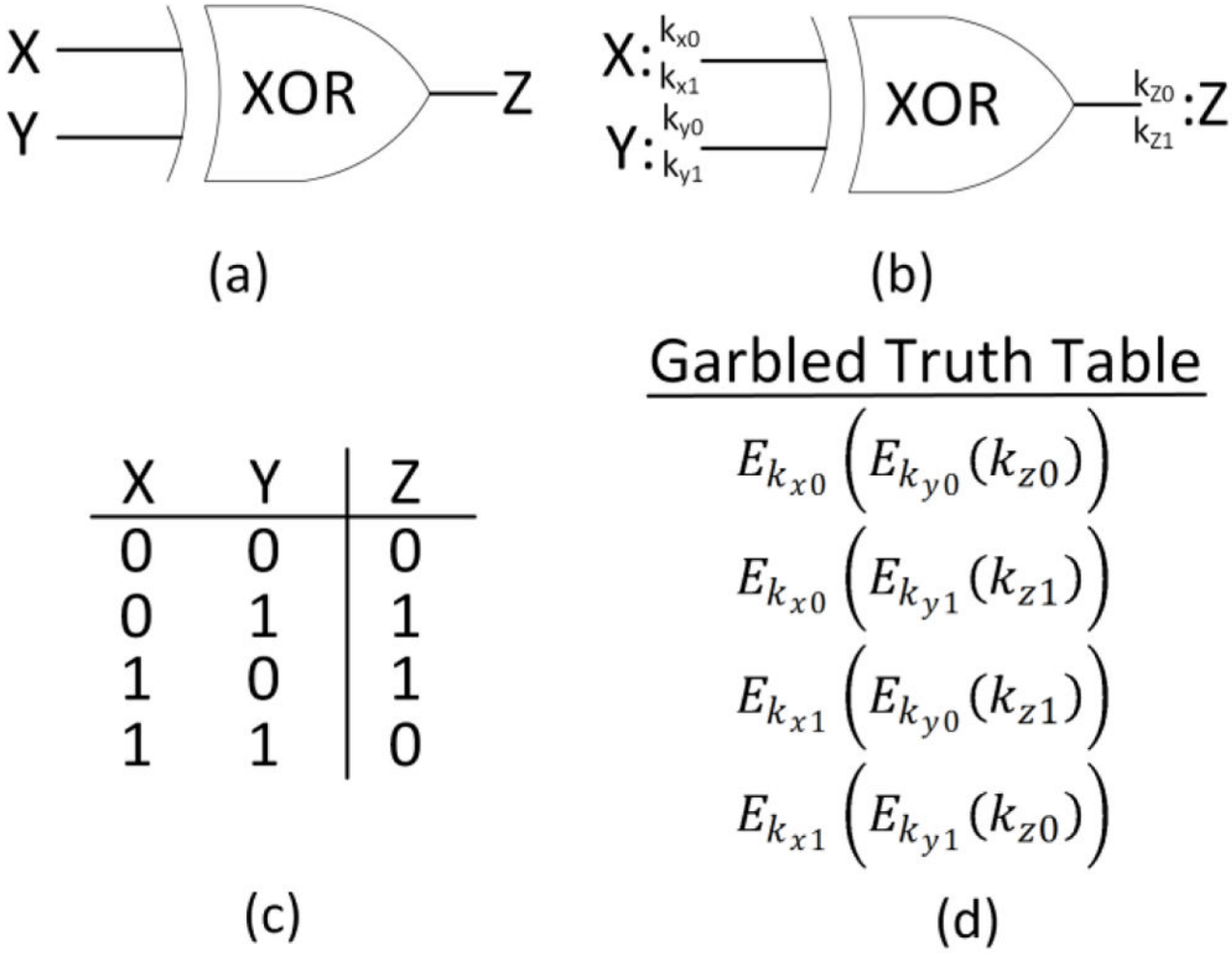

Figure 2.
Garbled Circuit Example

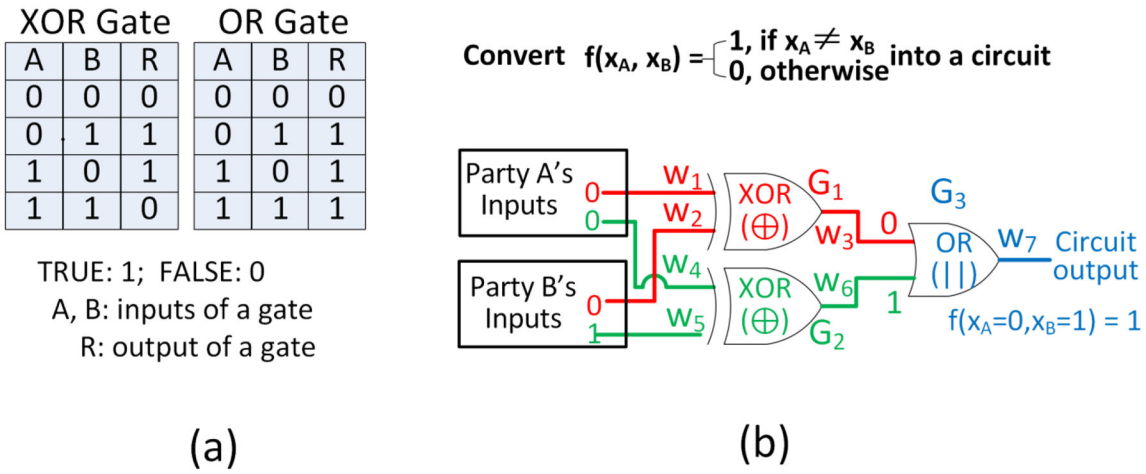

Figure 3.
The inequality circuit

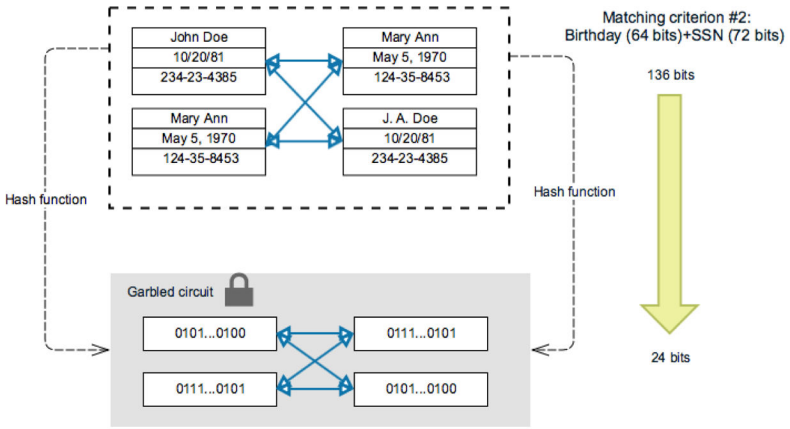

Figure 4.
Predicate matching through hashing

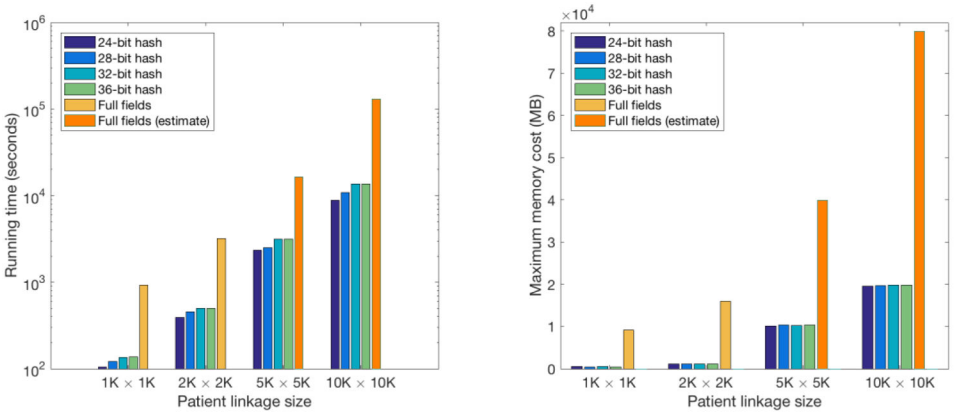

Figure 5.
Results for (a) computation time and (b) memory usage

Table 1

Patient linkage results over different dataset sizes.

size	# of linkages				
	24-bit hash	28-bit hash	32-bit hash	36-bit hash	Full fields
1K×1K	987	987	987	987	987
2K×2K	1,986	1,986	1,986	1,986	1,986
5K×5K	4,978	4,974	4,974	4974	4,974
10K×10K	6,026	6,001	6,001	6,000	6,000