Efficient and Scalable Infrastructure Support for Dynamic Coalitions

Michael T. Goodrich
Department of Computer Science
University of California, Irvine
goodrich@acm.org

Roberto Tamassia
Department of Computer Science
Brown University
rt@cs.brown.edu

Abstract

We overview our work on the subject of authenticated data structures, with applications to the development of a trust infrastructure for dynamic coalitions. We summarize our theoretical and practical contributions and discuss future research directions.

1. Introduction

This is an exciting and challenging time for information technology. Computers have moved beyond their early use as computational engines to now become information processors, with applications to every other discipline. Modern computers have processors and storage units powerful enough to take advantage of interesting data structures and algorithms. Moreover, computers bring together dynamic coalitions of entities connected to a vast network, the Internet, which has brought about new paradigms and modalities for computer applications to society, commerce, and defense. Because of this permeation of computing in society, modern computer applications must take into account the reality that they will be subject to both beneficial and malicious uses.

Unfortunately, the potential for malicious use is now a major limiting factor in the growth of many applications for dynamic coalitions, such as electronic software delivery, online document distribution, video/audio streaming, and electronic commerce. Technology is not holding these applications back—lack of trust is. That is, content providers and authors in dynamic coalitions are reluctant to utilize online content delivery, largely out of fear that hackers will find a way around any content protection schemes they employ. The trust problem is pervasive and persistent.

The aim of this project therefore is to develop algorithms for providing trust in computer applications for dynamic coalitions. In particular, we are interested in solutions that are fast and efficient as well as secure. History has taught us that users find ways around security schemes that are costly, cumbersome, or slow. Such schemes therefore provide only a false sense of security. Thus, we desire security solutions that have low latency and storage requirements, small communication costs, real-time interaction, and limited trust assumptions, even as we scale the agents in the environment, such as clients and servers. Our philosophy can be summed up with an adaptation of a popular automobile advertisement to cyber-security systems—*we believe that speed is a safety feature.*

The trustworthiness of information is essential in many applications. Indeed, there are several contexts where a violation of the integrity of information could have significant adverse consequences, including digital certificate revocation, financial data reporting, system access control, and mobile device authentication. We are therefore interested in the construction of systems that can provide data integrity services over a large network, such as the Internet.

There is a major bottleneck to building an online trusted data repository, however. The trust essential to the authentication process often times rests with a single entity, such as a certificate authority, financial institution, system administrator, or authentication office. Thus, in order for critical information to be communicated to the entities who need it most, the trusted entity must disseminate it in an authenticated way. But having the trusted authority online during client querying is often infeasible or inefficient. An authority is a trusted a source of information, but is not necessarily a large-scale query processor for hundreds or millions of data clients. Also, if the trusted authority is placed online, it could become the target of a denial-of-service attack. Therefore, we are interested in information authentication schemes that adhere to a simple philosophy:

Distribute computations, conserve trust.

That is, we should provide a large number of online locations, called *responders*, that store a replica of the repository and answer critical queries on the content of the repository. Even so, we should let the trust in the answers given by the responders be derived entirely from the information source. That is, we view the responders as untrusted parties that are

Proceedings of the DARPA Information Survivability Conference and Exposition (DISCEX'03)
0-7695-1897-4/03 $17.00 © 2003 IEEE

nevertheless able to provide authentication services on behalf of the trusted source. Such a framework is known in the cryptography literature as an *authenticated dictionary*.

We are working on a comprehensive project focused on the design, analysis and implementation of *authenticated data structures*, with special emphasis on applications to dynamic coalitions. Our research themes are therefore focused on establishing trust through the use of efficient verification schemes. Put another way, they provide a cybersecurity analogue to the Cold War philosophy of President Reagan: "Trust but verify."

The rest of this paper is organized as follows. In Section 2, we review our data structure authentication model and summarize previous work on authenticated data structures. Section 3 overviews the main theoretical and applied results our project. Directions for future work are discussed in Section 4.

2. Authenticated Data Structures

Verifying information that at first appears authentic is an often neglected task in data structure and algorithm usage. Fortunately, there is a growing literature on correctness checking that aims to rectify this omission. Following early work on program checking (see, e.g, [4, 42, 43]), efficient schemes have been developed for checking the results of various data structures and algorithms (see, e.g., [2, 5, 6, 7, 16, 17, 18, 27, 29, 33, 34]). The original motivation for this work was mainly to defend the user against an inadvertent error made during implementation.

With the advent of Web services and content delivery networks (e.g., Akamai), data structures and algorithms are no longer being used just by a single user on a single machine. The computer responding to a user's query could be unknown to both the data structure author and its user. Thus, we want to guard against an agent hosting a data structure who could deliberately falsify query responses to users.

This project studies a new dimension in data structure and algorithm checking—how can we design sophisticated data structures and algorithms so that their responses can be verified as accurately as if they were coming from their author, even when the response is coming from an untrusted host? Digital signatures can be used to verify simple static documents, but are inefficient for dynamic data structures. We therefore are interested in new techniques for authenticating data structures.

2.1. Model

Our data structure authentication model, illustrated in Figure 1, involves three parties: a trusted *source*, an untrusted *responder*, and a *user*. The *source* holds a structured collection S of objects, where we assume that a repertoire

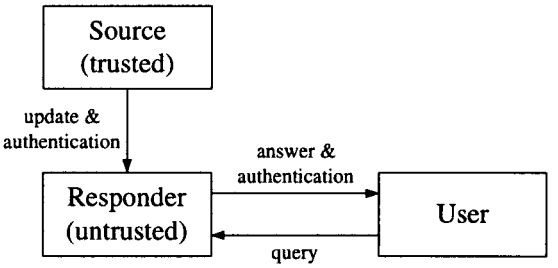

Figure 1. Authenticated data structure model.

of *query operations* are defined over S. If S is fixed over time, we say that it is *static*. Otherwise, we say that S is *dynamic* and assume that a repertoire of *update operations* are defined that modify S.

For example, S could be a network whose vertices and edges store data items and on which the following two query operations are defined: a *connectivity query* on S asks whether two given vertices of S are in the same connected component and a *path query* returns a path, if it exists, between two given vertices. We could also define update operations of S that add and/or remove vertices and edges. As a second example, S could be a collection of line segments in the plane forming a polygonal chain, where an *intersection query* returns all the segments intersected by a given query line. In this case we could define update operations that insert and/or remove segments.

The *responder* maintains a copy of the collection S together with *structure authentication information*, which consists of time-stamped statements about S signed by the source. The *user* performs queries on S but instead of contacting the source directly, it queries a responder. The responder provides the user with an answer to the query together with *answer authentication information*, which yields a cryptographic proof of the answer when used in conjunction with the structure authentication information. The user then verifies this proof. If S is dynamic, the responder receives together with each update *update authentication information*, which consists of signed time-stamped statements about the update and the current state of S.

The data structures used by the source and the responder to store collection S, together with the protocols and algorithms for queries, updates, and verifications executed by the various parties, form what we call an *authenticated data structure* [23, 32, 37]. In a practical deployment of an authenticated data structure, there would be various instances of geographically distributed responders. Such a distribution scheme reduces latency, allows for load balancing, and reduces the risk of denial-of-service attacks. Scalability is achieved by simply increasing the number of responders.

Indeed, since the responders are not trusted parties, they do not require physical security, such as brick enclosures, guards, etc.

2.2. Previous Work

Previous work related to authenticated data structures (initially motivated by its applications to the *certificate revocation* problem in public key infrastructure) is mostly concerned with *authenticated dictionaries*, which are authenticated structures for data sets on which membership queries are performed. The *hash tree* scheme introduced by Merkle [35, 36] can be used to implement a static authenticated dictionary. A hash tree T for a set S stores hashes of the elements of S at the leaves of T and a value $L(v)$ at each internal node v, which is the result of computing a one-way hash function on the values of its children. The authenticated dictionary for S consists of the hash tree T plus the signature of the value $L(r)$ stored at the root r of T. An element e is proven to belong in S by reporting the values stored at the nodes on the path in T from the node storing e to the root, together with the values of all nodes that have siblings on this path. With this approach, space is linear, and the query authentication information and the query and verification time are logarithmic in the size of the set S. Kocher [28] also advocates a static hash tree approach for realizing an authenticated dictionary, but simplifies somewhat the processing done by the user to verify that an item is not in the set S, by storing intervals instead of individual elements. A dynamic authenticated dictionary based on hash trees is described in [37]. Other schemes based on variations of hash trees have been proposed in [8, 19].

A first step towards the design of more general authenticated data structures (beyond dictionaries) is made in [15] with the authentication of operations *select*, *project* and *join* in a relational database and multidimensional orthogonal range queries. A study of authenticated queries beyond tree structures and skip lists is presented in [32]. It is shown that a class of data structures such that (i) the links of the structure form a directed acyclic graph of bounded degree and with a single source node and (ii) queries correspond to a traversal of a subdigraph of G, can be authenticated by means of a hashing scheme that digests the entire G into a hash value at its source. This technique is applied to the design of static authenticated data structures for pattern matching in tries and for orthogonal range searching in a multidimensional set of points. This paper also contains an initial treatment of authenticating fractional cascading structures, but only for a special and simple case, where catalogs are arranged as unions in a tree.

Additional related work on data authentication appears in [9, 30, 31], and related work on the authentication of XML documents appears in [14].

3. Accomplishments

Our research project has made several contributions to the theory and practice of authenticating data structures.

3.1. Theoretical Contributions

In [23], we present a data structure for an authenticated dictionary based on skip lists [40]. We introduce the notion of commutative hashing and show how to embed in the nodes of a skip list a computational DAG (directed acyclic graph) of cryptographic computations based on commutative hashing. This data structure matches the asymptotic performance of the Naor-Nissim approach [37], while simplifying the details of an actual implementation of a dynamic authenticated dictionary.

The software architecture and implementation of an authenticated dictionary based on skip lists is presented in [25]. In [1], the notion of *persistent authenticated dictionaries* is introduced, where the user can issue historical queries of the type "was element e in set S at time t".

In [24], we show how to use the RSA one-way accumulator to realize a dynamic authenticated dictionary for a set with n elements with $O(1)$ query authentication information size and verification time. This scheme allows a trade-off between the query and update times. For example, one can balance the two times and achieve $O(\sqrt{n})$ query and update time and $O(\sqrt{n})$ update authentication information.

Finally, we explore new efficient ways of incorporating and distributing the updates from many data authors in an authenticated data repository system [22], and we describe protocols for certified e-mail in a distributed setting [3].

3.2. Practical Contributions

We also made several practical contributions, including system implementations, experimental analyses, and vertical case studies.

In [21], we describe the architecture and implementation of a distributed system realizing an authenticated dictionary, called the *Lightweight Authenticated Information Repository* (*LAIR*). We also provide an empirical analysis of the performance of LAIR in various deployment scenarios.

LAIR has significant performance and security advantages over alternate systems, which are based on the following approaches:

- *Exposed source*: the source is online and answers queries from clients signing each response;
- *Repository forwarding*: the source distributes to responders updates to the repository and a signed hash of the repository; a responders forwards to the client the entire repository plus the hash signed by the source

as a proof of the answer (the Certificate Revocation List approach)

- *Trusted responder*: the source distributes to the responders updates to the repository; a responder answers a client query with a signed response (the OCSP approach).

LAIR involves the interaction of four primary parties: a *publisher* of data, a trusted *source*, untrusted *responders*, and *clients*. The *source* maintains a set S of key-value pairs that evolves over time. A *publisher* is a trusted entity who is given the authority to insert and delete elements in the set S. A *responder* maintains a copy of set S and answers queries from *clients* of the form "is element e in set S?" and "what is the value associated with element e?".

We show how our system can integrate smoothly with a Web services model, such as Microsoft's .NET framework. Indeed, several Web-based applications of LAIR have been implemented, including the end-to-end integrity of Web content [41], certificate revocation, and SSH host authentication [20].

Often, people and corporations alike assume that data is authentic merely by implicitly trusting the entity that delivers it. This is evident in the numerous stock tickers, sports scores, weather forecasts, and news articles widely available on the Internet. But as digital information becomes more pervasive, the threat to the integrity of the data increases and it would be imprudent to make such assumptions. For example, in August 2000, the stock market capitalization of Emulex suddenly lost $2B because of a spoofed press release, causing major financial damage to investors whose sell-stop orders were triggered by the drop. Thus, there exists a demand for an efficient system that will guarantee the end-to-end integrity of content from a trusted source to a client, especially when it is distributed and aggregated through third parties.

We have begun an investigation of authenticating distributed data using Web services and XML signatures [38]. We also introduce *prooflets*, a scalable architecture based on authenticated maps for authenticating Web content [41]. From the Web publisher's viewpoint, a *prooflet tag* is a specific HTML span tag delimiting a portion of an HTML or XML document that can be authenticated or securely retrieved. Prooflet-aware browsers recognize such tags and query the responders of an authenticated data structure to determine the authenticity of the data enclosed by prooflet tags. Prooflets achieve architectural, computational and economic scalability for the verification of the integrity (or secure retrieval) of Web content by leveraging the authenticated data structures technology.

The work on prooflets includes a vertical case study on the authentication of Web content that involves the following main tasks:

- Design of a prooflet tag library that extends standard HTML tags and can be readily embedded in any HTML document.
- Development of a Web service providing a distributed authenticated map functionality, using the standard SOAP protocol.
- Development of a web browser enhancement (e.g., a toolbar for Internet Explorer) serving as a trusted application for the verification of prooflets.

We believe prooflets overcome the major limitations of the traditional approach to content authentication, where a time-stamped cryptographic hash of the content is signed. Prooflets are especially effective when the content rapidly changes over time (e.g., stock quotes). Prooflets leverage standard Web technologies so that they can be readily adopted in today's Web publishing applications. Finally, prooflets allow portal sites to reduce the liability and costs associated with aggregating and publishing securely Web content originating from third part providers. We believe that prooflets are especially suited for the development of health-care and financial applications that provided value-added services by aggregating data from various trusted sources.

4. Future Work

For future work, we believe it would be interesting to consider the development of general techniques for building efficient authenticated data structures. Specifically, continuing our previous work on authenticated dictionaries [1, 23, 24, 25], we are interested in determining whether it is possible to construct an authenticated version of universal hash functions [10], which would simultaneously achieve constant expected query time, update time and verification time. Alternatively, we would like to be able to prove lower bounds and performance tradeoffs for authenticated dictionaries.

We are also interested in authenticated data structures supporting a variety of fundamental query operations on graphs. For example, given an undirected graph G, we would like to authenticate data structures for *connectivity* queries (i.e., whether two nodes of G are in the same connected component) and the related *biconnectivity* and *triconnectivity* queries. If G is weighted, we would like to authenticate *shortest path queries* (i.e., return a shortest path that connects two nodes of G or the length of such a path). For directed graphs, we are also interested in authenticating reachability queries and topological orderings. While the problem is already difficult in the static setting, since the space of possible queries is quadratic in the number of nodes, we would also like to efficiently G's authentication

information while G is updated by operations that involve inserting vertices and edges. We plan to develop an authenticated version of the combine-and-conquer paradigm [13], which yields a general data structuring mechanism.

In addition, we believe there is much to be gained from the study of several geometric search problems, aiming at methods that can authenticate the full, general version of the powerful *fractional cascading* technique [11]. Namely, we want to authenticate any query efficiently answered in a fractional-cascading structure via *iterative search*, where we have a collection of k dictionaries of total size n stored at the nodes of a connected graph and we want to search for an element in each dictionary in a subgraph of this graph. A number of fundamental two-dimensional geometric searching problems arising in the implementation of geographic information systems can be solved with data structures based on fractional cascading [12]. These problems include: *line intersection*, *ray shooting*, *point location*, *orthogonal range search*, *orthogonal point enclosure* and *orthogonal intersection* queries (see [39]). We are unaware of previously known authenticated data structures for the above problems, with the exception of orthogonal range search, for which an authenticated data structure was given in [32].

The security of our schemes will be based on standard cryptographic primitives, such as collision-resistant hashing and digital signatures; hence, it is practical and does not need any new cryptographic assumptions. Or preliminary results on the development of general techniques for authenticated data structure design have been promising [26].

We also plan to develop a prototype library of fundamental authenticated data structures in Java and conduct an experimental study of their performance. We will build on our previous experience in developing the JDSL data structures library (see, e.g., [44, 45]), which has more than 4,000 users.

5. Acknowledgements

The research project overviewed in this paper was supported in part by the Dynamic Coalitions Program of the Defense Advanced Research Projects Agency (DARPA) under DARPA/AFRL agreement F30602–00–2–0509.

The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of DARPA, the Air Force Research Laboratory or the U.S. Government.

We would like to thank many people who made significant contributions to this project, including Aris Anagnostopoulos, Andrea Carmignani, Robert Cohen, David Emory, Benety Goh, Jasminka Hasic, James Lentini, Jeremy Mullendore, Joel Sandin, Michael Shin, Dan Polivy, Andrew Schwerin, Christian Straub and Nikos Triandopoulos.

This work benefited from research collaborations with other DARPA-supported PIs, including Yair Amir and William Winsborough.

References

[1] A. Anagnostopoulos, M. T. Goodrich, and R. Tamassia. Persistent authenticated dictionaries and their applications. In *Proc. Information Security Conference (ISC 2001)*, volume 2200 of *LNCS*, pages 379–393. Springer-Verlag, 2001.

[2] S. Ar, M. Blum, B. Codenotti, and P. Gemmell. Checking approximate computations over the reals. In *Proc. ACM Symp. on the Theory of Computing*, pages 786–795, 1993.

[3] G. Ateniese, B. de Medeiros, and M. T. Goodrich. TRICERT: A distributed certified e-mail scheme. In *Network and Distributed Systems Security (NDSS) Symposium*, pages 47–56, 2001.

[4] M. Blum and S. Kannan. Designing programs that check their work. *J. ACM*, 42(1):269–291, Jan. 1995.

[5] J. D. Bright and G. Sullivan. Checking mergeable priority queues. In *Digest of the 24th Symposium on Fault-Tolerant Computing*, pages 144–153. IEEE Computer Society Press, 1994.

[6] J. D. Bright and G. Sullivan. On-line error monitoring for several data structures. In *Digest of the 25th Symposium on Fault-Tolerant Computing*, pages 392–401. IEEE Computer Society Press, 1995.

[7] J. D. Bright, G. Sullivan, and G. M. Masson. Checking the integrity of trees. In *Digest of the 25th Symposium on Fault-Tolerant Computing*, pages 402–411. IEEE Computer Society Press, 1995.

[8] A. Buldas, P. Laud, and H. Lipmaa. Accountable certificate management using undeniable attestations. In *ACM Conference on Computer and Communications Security*, pages 9–18. ACM Press, 2000.

[9] J. Camenisch and A. Lysyanskaya. Dynamic accumulators and application to efficient revocation of anonymous credentials. In *Proc. CRYPTO*, 2002.

[10] I. L. Carter and M. N. Wegman. Universal classes of hash functions. In *Proc. ACM Symp. on Theory of Computing*, pages 106–112, 1977.

[11] B. Chazelle and L. J. Guibas. Fractional cascading: I. A data structuring technique. *Algorithmica*, 1(3):133–162, 1986.

[12] B. Chazelle and L. J. Guibas. Fractional cascading: II. Applications. *Algorithmica*, 1:163–191, 1986.

[13] R. F. Cohen and R. Tamassia. Combine and conquer. *Algorithmica*, 18:342–362, 1997.

[14] P. Devanbu, M. Gertz, A. Kwong, C. Martel, G. Nuckolls, and S. Stubblebine. Flexible authentication of XML documents. In *Proc. ACM Conference on Computer and Communications Security*, 2001.

[15] P. Devanbu, M. Gertz, C. Martel, and S. Stubblebine. Authentic third-party data publication. In *Fourteenth IFIP 11.3 Conference on Database Security*, 2000.

[16] O. Devillers, G. Liotta, F. P. Preparata, and R. Tamassia. Checking the convexity of polytopes and the planarity of subdivisions. *Comput. Geom. Theory Appl.*, 11:187–208, 1998.

[17] G. Di Battista and G. Liotta. Upward planarity checking: "Faces are more than polygons". In S. H. Whitesides, editor, *Graph Drawing (Proc. GD '98)*, volume 1547 of *Lecture Notes Comput. Sci.*, pages 72–86. Springer-Verlag, 1998.

[18] U. Finkler and K. Mehlhorn. Checking priority queues. In *Proc. 10th ACM-SIAM Symp. on Discrete Algorithms*, pages S901–S902, 1999.

[19] I. Gassko, P. S. Gemmell, and P. MacKenzie. Efficient and fresh certification. In *Int. Workshop on Practice and Theory in Public Key Cryptography (PKC '2000)*, volume 1751 of *LNCS*, pages 342–353. Springer-Verlag, 2000.

[20] M. T. Goodrich, J. Lentini, and R. Tamassia. JSSH: A secure shell client. Technical report, Center for Geometric Computing, Brown University, 2002.

[21] M. T. Goodrich, M. Shin, R. Tamassia, and R. Cohen. LAIR: A lightweight authenticated information repository system. Technical report, Center for Geometric Computing, Brown University, 2002. http://www.cs.brown.edu/cgc/stms/papers/lair.pdf.

[22] M. T. Goodrich, M. Shin, R. Tamassia, and W. H. Winsborough. Authenticated dictionaries for fresh attribute credentials. Technical report, Center for Geometric Computing, Brown University, 2002. http://www.cs.brown.edu/cgc/stms/papers/fresh.pdf.

[23] M. T. Goodrich and R. Tamassia. Efficient authenticated dictionaries with skip lists and commutative hashing. Technical report, Johns Hopkins Information Security Institute, 2000. http://www.cs.brown.edu/cgc/stms/papers/hashskip.pdf.

[24] M. T. Goodrich, R. Tamassia, and J. Hasic. An efficient dynamic and distributed cryptographic accumulator. In *Proc. Int. Security Conference (ISC 2002)*, volume 2433 of *LNCS*, pages 372–388. Springer-Verlag, 2002.

[25] M. T. Goodrich, R. Tamassia, and A. Schwerin. Implementation of an authenticated dictionary with skip lists and commutative hashing. In *Proc. 2001 DARPA Information Survivability Conference and Exposition*, volume 2, pages 68–82, 2001.

[26] M. T. Goodrich, R. Tamassia, N. Triandopoulos, and R. Cohen. Authenticated data structures for graph and geometric searching. In *Proc. RSA-CT*, 2003. To appear.

[27] V. King. A simpler minimum spanning tree verification algorithm. In *Workshop on Algorithms and Data Structures*, pages 440–448, 1995.

[28] P. C. Kocher. On certificate revocation and validation. In *Proc. Int. Conf. on Financial Cryptography*, volume 1465 of *LNCS*. Springer-Verlag, 1998.

[29] D. Kratsh, R. McConnell, K. Mehlhorn, and J. Spinrad). Certifying algorithms for recognizing interval and permutation graphs. In *Proc. ACM-SIAM Symp. on Discrete Algorithms*, 2003.

[30] P. Maniatis and M. Baker. Enabling the Archival Storage of Signed Documents. In *Proceedings of the USENIX Conference on File and Storage Technologies (FAST 2002)*, Monterey, CA, USA, 2002.

[31] P. Maniatis and M. Baker. Secure History Preservation Through Timeline Entanglement. In *Proceedings of the 11th USENIX Security Symposium*, San Francisco, CA, USA, 2002.

[32] C. Martel, G. Nuckolls, P. Devanbu, M. Gertz, A. Kwong, and S. Stubblebine. A general model for authentic data publication, 2001. http://www.cs.ucdavis.edu/~devanbu/files/model-paper.pdf.

[33] K. Mehlhorn and S. Näher. *LEDA: A Platform for Combinatorial and Geometric Computing*. Cambridge University Press, Cambridge, UK, 2000.

[34] K. Mehlhorn, S. Näher, M. Seel, R. Seidel, T. Schilz, S. Schirra, and C. Uhrig. Checking geometric programs or verification of geometric structures. *Comput. Geom. Theory Appl.*, 12(1–2):85–103, 1999.

[35] R. C. Merkle. Protocols for public key cryptosystems. In *Proc. Symp. on Security and Privacy*, pages 122–134. IEEE Computer Society Press, 1980.

[36] R. C. Merkle. A certified digital signature. In G. Brassard, editor, *Proc. CRYPTO '89*, volume 435 of *LNCS*, pages 218–238. Springer-Verlag, 1990.

[37] M. Naor and K. Nissim. Certificate revocation and certificate update. In *Proc. 7th USENIX Security Symposium*, pages 217–228, Berkeley, 1998.

[38] D. J. Polivy and R. Tamassia. Authenticating distributed data using Web services and XML signatures. In *Proc. ACM Workshop on XML Security*, 2002.

[39] F. P. Preparata and M. I. Shamos. *Computational Geometry: An Introduction*. Springer-Verlag, 3rd edition, Oct. 1990.

[40] W. Pugh. Skip lists: a probabilistic alternative to balanced trees. *Commun. ACM*, 33(6):668–676, 1990.

[41] M. Shin, C. Straub, R. Tamassia, and D. J. Polivy. Authenticating Web content with prooflets. Technical report, Center for Geometric Computing, Brown University, 2002.

[42] G. F. Sullivan and G. M. Masson. Certification trails for data structures. In *Digest of the 21st Symposium on Fault-Tolerant Computing*, pages 240–247. IEEE Computer Society Press, 1991.

[43] G. F. Sullivan, D. S. Wilson, and G. M. Masson. Certification of computational results. *IEEE Trans. Comput.*, 44(7):833–847, 1995.

[44] R. Tamassia, M. T. Goodrich, L. Vismara, M. Handy, G. Shubina, R. Cohen, B. Hudson, R. S. Baker, N. Gelfand, and U. Brandes. JDSL: The data structures library in Java. *Dr. Dobb's Journal*, 323, April 2001.

[45] R. Tamassia and L. Vismara. A case study in algorithm engineering for geometric computing. *Internat. J. Comput. Geom. Appl.*, 11(1):15–70, 2001.