A Categorized Bibliography on Incremental Computation

G. Ramalingam and Thomas Reps
University of Wisconsin

1. Introduction

In many kinds of computational contexts, modifications of the input data are to be processed at once so as to have immediate effect on the output. Because small changes in the input to a computation often cause only small changes in the output, the challenge is to compute the new output *incrementally* by updating parts of the old output, rather than by recomputing the entire output from scratch (as a "batch computation"). Put another way, the goal is to make use of the solution to one problem instance to find the solution to a "nearby" problem instance.

The abstract problem of incremental computation can be phrased as follows: The goal is to compute a function f on the user's "input" data x—where x is often some data structure, such as a tree, graph, or matrix—and to keep the output $f(x)$ updated as the input undergoes changes. An *incremental algorithm* for computing f takes as input the "batch input" x, the "batch output" $f(x)$, possibly some auxiliary information, and the change in the "batch input" Δx. The algorithm computes the new "batch output" $f(x + \Delta x)$, where $x + \Delta x$ denotes the modified input, and updates the auxiliary information as necessary.

From the standpoint of the programming-languages community, interest in incremental computation stems from the following four research topics:

- The creation of languages with facilities that support incremental computation (for the purpose of providing language support for interactive systems, in which input undergoes modifications).
- Incremental computation as a paradigm for program optimization, particularly loop optimization in very-high-level languages.
- The development of incremental language-processing algorithms (for use in interactive programming tools).
- The creation of compilers and programming tools that implement the above ideas.

In addition to the work that has gone on in these areas of direct interest, there is a considerable body of related work on incremental computation that, although less well known in the programming-languages community, may have much to offer:

- Several different criteria have been developed for comparing the performance of different algorithms for an incremental-computation problem.
- There have been a few advances made towards establishing general principles of incremental computing (*e.g.*, the work on *dynamization* of static problems).
- There are results on a large number of individual incremental-computation problems; these may provide an opportunity for extracting new general principles or for generating new ideas through which language support for incremental computation might be provided.

This work was supported in part by a David and Lucile Packard Fellowship for Science and Engineering, by the National Science Foundation under grant CCR-9100424, and by the Defense Advanced Research Projects Agency under ARPA Order No. 8856 (monitored by the Office of Naval Research under contract N00014-92-J-1937).

Authors' address: Computer Sciences Department, University of Wisconsin–Madison, 1210 W. Dayton St., Madison, WI 53706.

E-mail: {ramali, reps}@cs.wisc.edu

This document provides a guide to some of the literature that has appeared on incremental computing; however, it is by no means a complete list of papers in the area, even for the topics with which we are the most familiar.

The seven points listed above could have served as the organizational principle for this bibliography; however, we chose to follow a different approach to classifying the existing work on incremental computation, attempting to survey the field of incremental computation *per se*, rather than just the narrower topic of incremental-computation research in the programming-languages community. In so doing, the goal was to expose ties between the ideas on incremental computation that developed out of research on programming languages and programming tools, and ideas that have been developed by researchers in other areas.

2. Assessment of Incremental Algorithms

2.1. Computational Complexity of Incremental Computation

One of the first problems that one must come to grips with when dealing with algorithms for incremental-computation problems is that the criteria commonly used to assess the performance of algorithms for batch-computation problems can be unsatisfactory. In particular, a common way to evaluate the time complexity of a batch algorithm is to use asymptotic analysis and to express the cost of the computation as a function of the size of the input; however, for incremental-computation problems, this kind of analysis can have several drawbacks:

- It may fail to distinguish between two different incremental algorithms for a problem, one of which is clearly superior to the other. (In many cases, it even fails to distinguish between an incremental algorithm and the batch start-over algorithm.)
- For some incremental-computation problems, it can lead to the (erroneous) conclusion that the batch start-over algorithm is optimal.

Some other analysis criteria that can be of utility for incremental-computation problems are given below.

Direct comparison with the batch start-over algorithm

. Yellin, D. and Strom, R., "INC: A language for incremental computations," *ACM Trans. Program. Lang. Syst.* **13**(2) pp. 211-236 (April 1991).

Amortized-cost analysis

With *amortized-cost analysis*, the performance of an algorithm is averaged over a worst-case sequence of operations. This sometimes leads to an overall time bound that is much smaller than the worst-case time per operation multiplied by the number of operations.

. Sleator, D.D. and Tarjan, R.E., "A data structure for dynamic trees," *Journal of Computer and System Sciences* **26** pp. 362-391 (1983).

. Tarjan, R.E., "Amortized computational complexity," *SIAM J. Algebraic Discrete Methods* **6**(2) pp. 306-318 (April 1985).

Competitiveness

Another approach that measures the performance of algorithms over a sequence of operations is the analysis of algorithms' *competitiveness*. Given an on-line problem (*i.e.*, there is a sequence of requests and actions that must be performed on-line in response to requests, where each action has an associated cost), assume that there is an adversary with "maximally destructive intent" generating the requests. The notion of competitiveness assesses the amount of damage that the adversary can inflict, in the sense of comparing the performance of an on-line algorithm to the performance of an optimal off-line algorithm. The *competitive ratio* is the maximum value—over any sequence of requests—of the ratio between the cost of the on-line algorithm and the cost of an optimal off-line algorithm. Thus, an algorithm designer seeks a competitive ratio as small as possible (where the smallest possible ratio is 1).

What has sparked particular interest in this model for analyzing on-line algorithms is that it is possible to use randomness to "inhibit" the power of the adversary. That is, for some problems it is possible to find random on-line algorithms that have a smaller competitive ratio than the best deterministic on-line algorithm.

- Sleator, D.D. and Tarjan, R.E., "Amortized efficiency of list update and paging rules," *Commun. of the ACM* **28**(2) pp. 202-208 (February 1985).
- McGeoch, L.A. and Sleator, D.D. (eds.), *On-Line Algorithms*, American Mathematical Society, Providence, RI (1992).
- Karp, R.M., "On-line algorithms versus off-line algorithms: How much is it worth to know the future?," pp. 416-429 in *Information Processing 92: Proceedings of the IFIP Twelfth World Computer Congress*, ed. J. van Leeuwen,North-Holland, Amsterdam (September 1992).

Probabilistic analysis

- Louchard, G., Randrianarimanana, B., and Schott, R., "Dynamic algorithms in D.E. Knuth's model: A probabilistic analysis," *Theoretical Computer Science* **93** pp. 201-225 (1992).

Incremental relative lower bounds

- Berman, A.M., Paull, M.C., and Ryder, B.G., "Proving relative lower bounds for incremental algorithms," *Acta Informatica* **27** pp. 665-683 (1990).

Reductions between problems

- Reif, J.H., "A topological approach to dynamic graph connectivity," *Information Processing Letters* **25**(1) pp. 65-70 (1987).

Boundedness

Because an incremental algorithm makes use of the solution to one problem instance to find the solution to a "nearby" problem instance, another alternative to expressing the cost as a function of the size of the input is to measure the time complexity of an incremental algorithm in terms of the sum of the sizes of the *changes* in the input and output. An incremental algorithm is said to be *bounded* if, for all input data-sets and for all changes that can be applied to an input data-set, the time it takes to update the output solution depends only on the size of the *change* in the input and output, and not on the size of the *entire* current input. Otherwise, an incremental algorithm is said to be *unbounded*. A problem is said to be bounded (unbounded) if it has (does not have) a bounded incremental algorithm.

- Reps, T., Teitelbaum, T., and Demers, A., "Incremental context-dependent analysis for language-based editors," *ACM Trans. Program. Lang. Syst.* **5**(3) pp. 449-477 (July 1983).
- Alpern, B., Hoover, R., Rosen, B.K., Sweeney, P.F., and Zadeck, F.K., "Incremental evaluation of computational circuits," pp. 32-42 in *Proceedings of the First Annual ACM-SIAM Symposium on Discrete Algorithms*, (San Francisco, CA, Jan. 22-24, 1990), Society for Industrial and Applied Mathematics, Philadelphia, PA (1990).
- Ramalingam, G. and Reps, T., "On the computational complexity of incremental algorithms," TR-1033, Computer Sciences Department, University of Wisconsin, Madison, WI (August 1991).
- Ramalingam, G. and Reps, T., "An incremental algorithm for a generalization of the shortest-path problem," TR-1087, Computer Sciences Department, University of Wisconsin, Madison, WI (May 1992).
- Ramalingam, G. and Reps, T., "On the complexity of incremental computation," Unpublished report, Computer Sciences Department, University of Wisconsin, Madison, WI (October 1992).

2.2. Measurements of Actual Performance

There have been relatively few papers in which the performance of an incremental algorithm has been evaluated from an experimental standpoint. The little work that does exist actually suggests that from a practical standpoint incremental algorithms that do not have "good" theoretical performance (according to the criteria listed above) can give satisfactory performance in real systems.

- Dionne, R., "Etude et extension d'un algorithme de Murchland," *INFOR* **16**(2) pp. 132-146 (June 1978).
- Taylor, G.S. and Ousterhout, J.K., "Magic's incremental design-rule checker," pp. 160-165 in *Proceedings of the Twenty-First Design Automation Conference*, IEEE Computer Society, Washington, DC (1984).
- Scott, W.S. and Ousterhout, J.K., "Plowing: Interactive stretching and compaction in Magic," pp. 166-172 in *Proceedings of the Twenty-First Design Automation Conference*, IEEE Computer Society, Washington, DC (1984).
- Hoover, R., "Incremental graph evaluation," Ph.D. dissertation and Tech. Rep. 87-836, Dept. of Computer Science, Cornell University, Ithaca, NY (May 1987).
- Ryder, B.G., Landi, W., and Pande, H.D., "Profiling an incremental data flow analysis algorithm," *IEEE Transactions on Software Engineering* **SE-16**(2)(February 1990).
- Zaring, A., "Parallel evaluation in attribute grammar based systems," Ph.D. dissertation and Tech. Rep. 90-1149, Dept. of Computer Science, Cornell University, Ithaca, NY (August 1990).
- Harrison, M.A. and Munson, E.V., "Numbering document components," *Electronic Publishing* **4**(1)(January 1991).

3. Data-Structure Update Problems

3.1. Incremental Updating of Annotations on Graphs and Trees

One class of incremental-computation problems involves computing a function $f(x)$, where x is some data structure ("the substrate"), such as a tree, graph, or matrix, and $f(x)$ represents some annotation of the x data structure—a mapping from more primitive elements that make up x to some space of values (for example, a mapping from spreadsheet cells to values). Each annotation value is a function of other annotation values, often those of neighboring elements in the substrate data structure. The incremental-computation problem is to keep the annotation values updated as the substrate data structure undergoes some changes.

3.1.1. Selective Recomputation

In selective recomputation, annotation values that are independent of changed data are never recomputed. Annotation values that are dependent on changed data are recomputed, but after each new annotation value is obtained, the old and new annotation values may be compared in order to help determine what further recomputations must be performed.

Incremental updating of attributed derivation trees

- Demers, A., Reps, T., and Teitelbaum, T., "Incremental evaluation for attribute grammars with application to syntax-directed editors," pp. 105-116 in *Conference Record of the Eighth ACM Symposium on Principles of Programming Languages,* (Williamsburg, VA, Jan. 26-28, 1981), ACM, New York, NY (1981).

- Reps, T., "Optimal-time incremental semantic analysis for syntax-directed editors," pp. 169-176 in *Conference Record of the Ninth ACM Symposium on Principles of Programming Languages,* (Albuquerque, NM, January 25-27, 1982), ACM, New York, NY (1982).

- Reps, T., Teitelbaum, T., and Demers, A., "Incremental context-dependent analysis for language-based editors," *ACM Trans. Program. Lang. Syst.* **5**(3) pp. 449-477 (July 1983).

- Reps, T., *Generating Language-Based Environments,* The M.I.T. Press, Cambridge, MA (1984).

- Yeh, D., "On incremental evaluation of ordered attributed grammars," *BIT* **23** pp. 308-320 (1983).

- Jones, L. and Simon, J., "Hierarchical VLSI design systems based on attribute grammars," pp. 58-69 in *Conference Record of the Thirteenth ACM Symposium on Principles of Programming Languages,* (St. Petersburg, FL, Jan. 13-15, 1986), ACM, New York, NY (1986).

- Reps, T., Marceau, C., and Teitelbaum, T., "Remote attribute updating for language-based editors," pp. 1-13 in *Conference Record of the Thirteenth ACM Symposium on Principles of Programming Languages,* (St. Petersburg, FL, Jan. 13-15, 1986), ACM, New York, NY (1986).

- Kaplan, S. and Kaiser, G., "Incremental attribute evaluation in distributed language-based editors," pp. 121-130 in *Proceedings of the Fifth ACM Symposium on Principles of Distributed Computing,* (1986).

- Hoover, R. and Teitelbaum, T., "Efficient incremental evaluation of aggregate values in attribute grammars," *Proceedings of the SIGPLAN 86 Symposium on Compiler Construction,* (Palo Alto, CA, June 25-27, 1986), *ACM SIGPLAN Notices* **21**(7) pp. 39-50 (July 1986).

- Hoover, R., "Incremental graph evaluation," Ph.D. dissertation and Tech. Rep. 87-836, Dept. of Computer Science, Cornell University, Ithaca, NY (May 1987).

- Parigot, D., "Transformations, évaluation incrémentale et optimizations des grammaires attribuées: Le système FNC-2," These de Doctorat, L'Université Paris XI, Centre D'Orsay, Orsay, France (1988).

- Kaiser, G.E., "Incremental dynamic semantics for language-based programming environments," *ACM Trans. Program. Lang. Syst.* **11**(2) pp. 169-193 (April 1989).

- Teitelbaum, T. and Chapman, R., "Higher-order attribute grammars and editing environments," *Proceedings of the ACM SIGPLAN 90 Conference on Programming Language Design and Implementation,* (White Plains, NY, June 20-22, 1990), *ACM SIGPLAN Notices* **25**(6) pp. 197-208 (June 1990).

- Zaring, A., "Parallel evaluation in attribute grammar based systems," Ph.D. dissertation and Tech. Rep. 90-1149, Dept. of Computer Science, Cornell University, Ithaca, NY (August 1990).

The incremental circuit-annotation problem

- Pardo, R.K. and Landau, R., "Process and apparatus for converting a source program into an object program," U.S. Patent No. 4,398,249, United States Patent Office, Washington, DC (August 9, 1983).

- Alpern, B., Hoover, R., Rosen, B.K., Sweeney, P.F., and Zadeck, F.K., "Incremental evaluation of computational circuits," pp. 32-42 in *Proceedings of the First Annual ACM-SIAM Symposium on Discrete Algorithms,* (San Francisco, CA, Jan. 22-24, 1990), Society for Industrial and Applied Mathematics, Philadelphia, PA (1990).

- Ramalingam, G. and Reps, T., "On the computational complexity of incremental algorithms," TR-1033, Computer Sciences Department, University of Wisconsin, Madison, WI (August 1991).

- Ramalingam, G. and Reps, T., "On the complexity of incremental computation," Unpublished report, Computer Sciences Department, University of Wisconsin, Madison, WI (October 1992).

Incremental data-flow analysis

- Rosen, B.K., "Linear cost is sometimes quadratic," pp. 117-124 in *Conference Record of the Eighth ACM Symposium on Principles of Programming Languages,* (Williamsburg, VA, January 26-28, 1981), ACM, New York, NY (1981).

- Ryder, B., "Incremental data flow analysis based on a unified model of elimination algorithms," Ph.D. dissertation and Tech. Rep. DCS-TR-117, Rutgers University, New Brunswick, NJ (September 1982).

- Ghodssi, V., "Incremental analysis of programs," Ph.D. dissertation, Dept. of Computer Science, University of Central Florida, Orlando, FL (1983).

- Zadeck, F.K., "Incremental data flow analysis in a structured program editor," Ph.D. dissertation, Mathematical Sciences Dept., Rice University, Houston, TX (October 1983).

- Zadeck, F.K., "Incremental data flow analysis in a structured program editor," *Proceedings of the SIGPLAN 84 Symposium on Compiler Construction,* (Montreal, Can., June 20-22, 1984), *ACM SIGPLAN Notices* **19**(6) pp. 132-143 (June 1984).

- Cooper, K.D. and Kennedy, K., "The impact of interprocedural analysis and optimization in the R^n programming environment," *ACM Trans. Program. Lang. Syst.* **8**(4) pp. 491-523 (October 1986).

- Burke, M., "An interval-based approach to exhaustive and incremental interprocedural data flow analysis," Res. Rep. RC 12702, IBM T.J. Watson Research Center, Yorktown Heights, NY (April 1987).

- Burke, M. and Ryder, B., "Incremental iterative data flow analysis algorithms," Res. Rep. RC 13170, IBM T.J. Watson Research Center, Yorktown Heights, NY (October 1987).

- Carroll, M. and Ryder, B., "Incremental data flow update via attribute and dominator updates," pp. 274-284 in *Conference Record of the Fifteenth ACM Symposium on Principles of Programming Languages,* (San Diego, CA, January 13-15, 1988), ACM, New York, NY (1988).

- Ryder, B.G. and Paull, M.C., "Incremental data flow analysis algorithms," *ACM Trans. Program. Lang. Syst.* **10**(1) pp. 1-50 (January 1988).

- Marlowe, T.J., "Data flow analysis and incremental iteration," Ph.D. dissertation and Tech. Rep. DCS-TR-255, Rutgers University, New Brunswick, NJ (October 1989).

- Marlowe, T.J. and Ryder, B.G., "An efficient hybrid algorithm for incremental data flow analysis," pp. 184-196 in *Conference Record of the Seventeenth ACM Symposium on Principles of Programming Languages,* (San Francisco, CA, Jan. 17-19, 1990), ACM, New York, NY (1990).

- Rosene, C.M., "Incremental dependence analysis," Ph.D. dissertation and Tech Rep. CRPC-TR90044, Center for Research on Parallel Computation, Rice University, Houston, TX (March 1990).

- Ramalingam, G. and Reps, T., "On the computational complexity of incremental algorithms," TR-1033, Computer Sciences Department, University of Wisconsin, Madison, WI (August 1991).

Maintaining shortest distances and other path problems in graphs

- Murchland, J.D., "The effect of increasing or decreasing the length of a single arc on all shortest distances in a graph," Tech. Rep. LBS-TNT-26, London Business School, Transport Network Theory Unit, London, UK (1967).

- Loubal, P., "A network evaluation procedure," *Highway Research Record* **205** pp. 96-109 (1967).

- Rodionov, V., "The parametric problem of shortest distances," *U.S.S.R. Computational math. and math. Phys.* **8**(5) pp. 336-343 (1968).

- Halder, A.K., "The method of competing links," *Transportation Science* **4** pp. 36-51 (1970).

- Dionne, R., "Etude et extension d'un algorithme de Murchland," *INFOR* **16**(2) pp. 132-146 (June 1978).

- Cheston, G.A., "Incremental algorithms in graph theory," Ph.D. dissertation and Tech. Rep. 91, Dept. of Computer Science, University of Toronto, Toronto, Canada (March 1976).

- Goto, S. and Sangiovanni-Vincentelli, A., "A new shortest path updating algorithm," *Networks* **8**(4) pp. 341-372 (1978).

- Rohnert, H., "A dynamization of the all pairs least cost path problem," pp. 279-286 in *Proceedings of STACS 85: Second Annual Symposium on Theoretical Aspects of Computer Science,* (Saarbruecken, W. Ger., Jan. 3-5, 1985), *Lecture Notes in Computer Science,* Vol. 182, ed. K. Mehlhorn,Springer-Verlag, New York, NY (1985).

- Even, S. and Gazit, H., "Updating distances in dynamic graphs," *Methods of Operations Research* **49** pp. 371-387 (1985).

- Lin, C.-C. and Chang, R.-C., "On the dynamic shortest path problem," *Journal of Information Processing* **13**(4)(1990).

- Ausiello, G., Italiano, G.F., Spaccamela, A.M., and Nanni, U., "Incremental algorithms for minimal length paths," *Journal of Algorithms* **12** pp. 615-638 (1991).

- Ramalingam, G. and Reps, T., "On the computational complexity of incremental algorithms," TR-1033, Computer Sciences Department, University of Wisconsin, Madison, WI (August 1991).

- Ramalingam, G. and Reps, T., "An incremental algorithm for a generalization of the shortest-path problem," TR-1087, Computer Sciences Department, University of Wisconsin, Madison, WI (May 1992).

3.1.2. Differential Updating

In differential updating, rather than recomputing an annotation value $z' = g(y')$ in terms of its new argument y', the old annotation value $z = g(y)$ is updated by some difference Δz computed as a function of y, y', g, and z.

- Koenig, S. and Paige, R., "A transformational framework for the automatic control of derived data," pp. 306-318 in *Proceedings of the Seventh International Conference on Very Large Data Bases,* (Cannes, France, September 1981), (1981).

- Shmueli, O. and Itai, A., "Maintenance of views," pp. 240-255 in *Proceedings of the ACM SIGMOD 84 Conference,* (Boston, MA, 1984), ACM, New York, NY (1984).

- Horwitz, S., "Generating language-based editors: A relationally-attributed approach," Ph.D. dissertation, Dept. of Computer Science, Cornell University, Ithaca, NY (August 1985).

- Horwitz, S. and Teitelbaum, T., "Generating editing environments based on relations and attributes," *ACM Trans. Program. Lang. Syst.* **8**(4) pp. 577-608 (October 1986).

- Yellin, D. and Strom, R., "INC: A language for incremental computations," *ACM Trans. Program. Lang. Syst.* **13**(2) pp. 211-236 (April 1991).

3.1.3. Other Incremental Expression-Evaluation Algorithms

Several of the algorithms for incremental expression-evaluation problems do not fit in either of the above two categories.

- Pugh, W.W., "Incremental computation and the incremental evaluation of functional programs," Ph.D. dissertation and Tech. Rep. 88-936, Dept. of Computer Science, Cornell University, Ithaca, NY (August 1988).

- Pugh, W. and Teitelbaum, T., "Incremental computation via function caching," pp. 315-328 in *Conference Record of the Sixteenth ACM Symposium on Principles of Programming Languages*, (Austin, TX, Jan. 11-13, 1989), ACM, New York, NY (1989).

- Field, J. and Teitelbaum, T., "Incremental reduction in the lambda calculus," in *Conference Record of the 1990 ACM Symposium on Lisp and Functional Programming*, (Nice, France, June 1990), ACM, New York, NY (1990).

- Field, J., "Incremental reduction in the lambda calculus and related reduction systems," Ph.D. dissertation, Dept. of Computer Science, Cornell University, Ithaca, NY (May 1991).

- Cohen, R.F. and Tamassia, R., "Dynamic expression trees and their applications," pp. 52-61 in *Proceedings of the Second Annual ACM-SIAM Symposium on Discrete Algorithms*, (San Francisco, CA, Jan. 28-30, 1991), Society for Industrial and Applied Mathematics, Philadelphia, PA (1991).

3.2. Other Dynamic Graph Problems

- Cheston, G.A., "Incremental algorithms in graph theory," Ph.D. dissertation and Tech. Rep. 91, Dept. of Computer Science, University of Toronto, Toronto, Canada (March 1976).

- Even, S. and Shiloach, Y., "An on-line edge-deletion problem," *J. ACM* **28**(1) pp. 1-4 (January 1981).

- Frederickson, G., "Data structures for on-line updating of minimum spanning trees," *SIAM J. Computing* **14** pp. 781-798 (1985).

- Italiano, G.F., "Amortized efficiency of a path retrieval data structure," *Theoretical Computer Science* **48** pp. 273-281 (1986).

- Reif, J.H., "A topological approach to dynamic graph connectivity," *Information Processing Letters* **25**(1) pp. 65-70 (1987).

- La Poutré, J.A. and van Leeuwen, J., "Maintenance of transitive closures and transitive reductions of graphs," pp. 106-120 in *Graph-Theoretic Concepts in Computer Science: Proceedings of the 14th International Workshop* (1988), *Lecture Notes in Computer Science*, (1988).

- Italiano, G.F., "Finding paths and deleting edges in directed acyclic graphs," *Information Processing Letters* **28** pp. 5-11 (1988).

- Yellin, D.M., "A dynamic transitive closure algorithm," Research Report, IBM T.J. Watson Research Center, Yorktown Heights, NY (1988).

- Di Battista, G. and Tamassia, R., "Incremental planarity testing," pp. 436-441 in *Proceedings of the Thirtieth IEEE Symposium on Foundations of Computer Science*, IEEE Computer Society, Washington, DC (1989).

- Buchsbaum, A.L., Kanellakis, P.C., and Vitter, J.S., "A data structure for arc insertion and regular path finding," pp. 22-31 in *Proceedings of the First Annual ACM-SIAM Symposium on Discrete Algorithms*, (San Francisco, CA, Jan. 22-24, 1990), Society for Industrial and Applied Mathematics, Philadelphia, PA (1990).

- Yannakakis, M., "Graph-theoretic methods in database theory," pp. 230-242 in *Proceedings of the Symposium on Principles of Database Systems*, (1990).

- Frederickson, G., "Ambivalent data structures for dynamic 2-edge-connectivity and k smallest spanning trees," pp. 632-641 in *Proceedings of the Thirty-Second IEEE Symposium on Foundations of Computer Science*, IEEE Computer Society, Washington, DC (1991).

- Galil, Z. and Italiano, G.F., "Fully dynamic algorithms for edge-connectivity problems," pp. 317-327 in *Proceedings of the Twenty-Third Annual ACM Symposium on Theory of Computing*, ACM, New York, NY (1991).

- Kanevsky, A., Tamassia, R., Di Battista, G., and Chen, J., "On-line maintenance of the four-connected components of a graph," pp. 793-801 in *Proceedings of the Thirty-Second IEEE Symposium on Foundations of Computer Science*, IEEE Computer Society, Washington, DC (1991).

- La Poutré, J.A., "Maintenance of triconnected components of graphs," pp. 354-365 in *Proceedings of the Nineteenth International Collquium on Automata, Languages, and Programming*, (1992).

- Eppstein, D., Galil, Z., Italiano, G.F., and Nissenzweig, A., "Sparsification – A technique for speeding up dynamic graph algorithms," in *Proceedings of the Thirty-third IEEE Symposium on Foundations of Computer Science*, (Pittsburgh, PA, Oct. 25-27, 1992), IEEE Computer Society, Washington, DC (1992).

3.3. Dynamization of Static Data-Structure Problems

Dynamization of multi-dimensional searching problems

- Bentley, J.L., "Decomposable searching problems," *Information Processing Letters* **8** pp. 244-251 (1979).

. Bentley, J.L. and Saxe, J.B., "Decomposable searching problems I: Static-to-dynamic transformations," *Journal of Algorithms* **1** pp. 301-358 (1980).

. Overmars, M.H., *The Design of Dynamic Data Structures, Lecture Notes in Computer Science,* Vol. 156, Springer-Verlag, New York, NY (1983).

. Mehlhorn, K., *Data Structures and Algorithms 3: Multi-Dimensional Searching and Computational Geometry,* Springer-Verlag, Berlin (1984).

Dynamization of graph problems

. Eppstein, D., Galil, Z., Italiano, G.F., and Nissenzweig, A., "Sparsification – A technique for speeding up dynamic graph algorithms," in *Proceedings of the Thirty-third IEEE Symposium on Foundations of Computer Science,* (Pittsburgh, PA, Oct. 25-27, 1992), IEEE Computer Society, Washington, DC (1992).

3.4. Finite Differencing

. Earley, J., "High-level operations in automatic programming," *Proceedings of the ACM SIGPLAN Symposium on Very High Level Languages,* (March 1974), *ACM SIGPLAN Notices* **9**(4)(April 1974).

. Earley, J., "High-level iterators and a method for automatically designing data structure representation," *J. Computer Languages* **1**(4) pp. 321-342 (1976).

. Fong, A. and Ullman, J., "Induction variables in very high level languages," pp. 104-112 in *Conference Record of the Third ACM Symposium on Principles of Programming Languages,* (Atlanta, GA, Jan. 19-21, 1976), ACM, New York, NY (1976).

. Fong, A., "Elimination of common subexpressions in very high level languages," pp. 48-57 in *Conference Record of the Fourth ACM Symposium on Principles of Programming Languages,* (Los Angeles, CA, January 17-19, 1977), ACM, New York, NY (1977).

. Fong, A., "Inductively computable constructs in very high level languages," pp. 21-28 in *Conference Record of the Sixth ACM Symposium on Principles of Programming Languages,* (San Antonio, TX, Jan. 29-31, 1979), ACM, New York, NY (1979).

. Paige, R. and Koenig, S., "Finite differencing of computable expressions," *ACM Trans. Program. Lang. Syst.* **4**(3) pp. 402-454 (July 1982).

. Goldberg, A. and Paige, R., "Stream processing," in *Conference Record of the 1984 ACM Symposium on Lisp and Functional Programming,* (Austin, TX, August 6-8, 1984), ACM, New York, NY (1984).

. Paige, R., "Programming with invariants," *IEEE Software* **3**(1) pp. 56-69 (January 1986).

4. Incremental Formal Systems

4.1. Incremental Reduction

Incremental functional programming

. Lombardi, L.A. and Raphael, B., "Lisp as the language for an incremental computer," pp. 204-219 in *The Programming Language Lisp: Its Operation and Applications,* ed. E.C. Berkeley and D.G. Bobrow,The M.I.T. Press, Cambridge, MA (1964).

. Lombardi, L.A., "Incremental computation," pp. 247-333 in *Advances in Computers,* Vol. 8, ed. F.L. Alt and M. Rubinoff,Academic Press (1967).

. Sundaresh, R.S. and Hudak, P., "Incremental computation via partial evaluation," in *Conference Record of the Eighteenth ACM Symposium on Principles of Programming Languages,* (Orlando, FL, January 1991), ACM, New York, NY (1991).

. Sundaresh, R.S., "Building incremental programs using partial evaluation," *Proceedings of the SIGPLAN Symposium on Partial Evaluation and Semantics-Based Program Manipulation (PEPM 91),* (New Haven, CT, June 17-19, 1991), *ACM SIGPLAN Notices* **26**(9) pp. 83-93 (September 1991).

. Sundaresh, R.S., "Incremental computation via parial evaluation," Ph.D. dissertation and Res. Rep. YALEU/DCS/RR-889, Dept. of Computer Science, Yale University, New Haven, CT (December 1991).

. Field, J. and Teitelbaum, T., "Incremental reduction in the lambda calculus," in *Conference Record of the 1990 ACM Symposium on Lisp and Functional Programming,* (Nice, France, June 1990), ACM, New York, NY (1990).

. Field, J., "Incremental reduction in the lambda calculus and related reduction systems," Ph.D. dissertation, Dept. of Computer Science, Cornell University, Ithaca, NY (May 1991).

Incremental parsing

. Ghezzi, C. and Mandrioli, D., "Incremental parsing," *ACM Trans. Program. Lang. Syst.* **1**(1) pp. 58-70 (July 1979).

. Ghezzi, C. and Mandrioli, D., "Augmenting parsers to support incrementality," *Journal of the ACM* **27**(3) pp. 564-579 (October 1980).

. Wegman, M., "Parsing for structural editors," pp. 320-327 in *Proceedings of the Twenty-First IEEE Symposium on Foundations of Computer Science* (Syracuse, NY, October 1980), IEEE Computer Society, Washington, DC (1980).

. Jalili, F. and Gallier, J., "Building friendly parsers," pp. 196-206 in *Conference Record of the Ninth ACM Symposium on Principles of Programming Languages,* (Albuquerque, NM, Jan. 25-27, 1982), ACM, New York, NY (1982).

. Kaiser, G.E. and Kant, E., "Incremental parsing without a parser," *Journal of Systems and Software* **5**(2) pp. 121-144 (May 1985).

Algebraic laws of functional algebra

. MacLennan, B.J., "Preliminary investigation of a calculus of functional differences: Fixed differences," Report NPS52-86-010, Computer Science Department, Naval Postgraduate School, Monterey, CA (February 1986).

. MacLennan, B.J., "An algebraic approach to a calculus of functional differences: Fixed differences and integrals," Report NPS52-87-041, Computer Science Department, Naval Postgraduate School, Monterey, CA (September 1987).

. MacLennan, B.J., "A calculus of functional differences and integrals," Unpublished Draft, Department of Computer Science, University of Tennessee, Knoxville, TN ().

4.2. Truth Maintenance

. Doyle, J., "A truth maintenance system," *Artificial Intelligence* **12** pp. 231-272 (1979).

. Doyle, J., "A glimpse of truth maintenance," in *Artificial Intelligence: An M.I.T. Perspective*, ed. P.H. Winston and R.H. Brown,The M.I.T. Press, Cambridge, MA (1979).

. Perlis, D., "Bibliography of literature on non-monotonic reasoning," (Source unknown), (1984).

. de Kleer, J., "An assumption-based TMS," *Artificial Intelligence* **28** pp. 127-162 (1986).

. de Kleer, J., "Extending the ATMS," *Artificial Intelligence* **28** pp. 163-196 (1986).

. de Kleer, J., "Problem solving with the ATMS," *Artificial Intelligence* **28** pp. 197-224 (1986).

. McAllester, D., "Truth maintenance," pp. 92-104 in *Proceedings of the Eighth National Conference on Artificial Intelligence,* (Boston, MA, July 29 – August 3, 1990), AAAI Press/The M.I.T. Press, Cambridge, MA (1990).

4.3. Incremental Deduction

. Shmueli, O., Tsur, S., and Zfira, H., "Rule support in Prolog," (Source unknown), (1984).

. Mannila, H. and Ukkonen, E., "Time parameter and arbitrary deunions in the set union problem," pp. 34-42 in *Proceedings of the First Scandinavian Workshop on Algorithm Theory (SWAT 88), Lecture Notes in Computer Science,* Vol. 318, Springer-Verlag, New York, NY (1988).

Incremental deduction for static-semantic analysis

. Snelting, G. and Henhapl, W., "Unification in many-sorted algebras as a device for incremental semantic analysis," pp. 229-235 in *Conference Record of the Thirteenth ACM Symposium on Principles of Programming Languages,* (St. Petersburg, FL, Jan. 13-15, 1986), ACM, New York, NY (1986).

. Attali, I., "Compiling TYPOL with attribute grammars," pp. 252-272 in *Proceedings of the International Workshop on Programming Language Implementation and Logic Programming '88, Lecture Notes in Computer Science,* Vol. 348, ed. P. Deransart, B. Lorho, and J. Maluszynski,Springer-Verlag, New York, NY (1988).

. Attali, I. and Franchi-Zannettacci, P., "Unification-free execution of TYPOL programs by semantic attribute evaluation," pp. 160-177 in *Proceedings of the Fifth International Conference and Symposium on Logic Programming*, ed. R. Kowalski and K. Bowen,The M.I.T. Press, Cambridge, MA (1988).

. van der Meulen, E.A., "Deriving incremental implementations from algebraic specifications," Report CS-R9072, Computer Science/Department of Software Technology, Center for Mathematics and Computer Science (CWI), Amsterdam, The Netherlands (December 1990).

. Ballance, R.A., "Syntactic and semantic checking in language-based editing systems," Ph.D. dissertation and Tech. Rep. UCB/CSD 89/548, Dept. of Electrical Engineering and Computer Science, University of California–Berkeley, Berkeley, CA (December 1989).

. Ballance, R.A. and Graham, S.L., "Incremental consistency maintenance for interactive applications," in *Proceedings of the Eighth International Conference on Logic Programming*, (1991).

4.4. Incremental Constraint Solving

. Vander Zanden, B. T., "Incremental constraint satisfaction and its application to graphical interfaces," Ph.D. dissertation and Tech. Rep. TR 88-941, Dept. of Computer Science, Cornell University, Ithaca, NY (October 1988).

. Freeman-Benson, B.N, Maloney, J., and Borning, A., "An incremental constraint solver," *Commun. of the ACM* **33**(1) pp. 54-63 (January 1990).

5. Other Special-Purpose Algorithms

Incremental compilation and linking

. Fritzson, P., "Preliminary experience from the DICE system, a distributed incremental compiling environment," *Proceedings of the ACM SIGSOFT/SIGPLAN Software Engineering Symposium on Practical Software Development Environments,* (Pittsburgh, PA, Apr. 23-25, 1984), *ACM SIGPLAN Notices* **19**(5) pp. 113-123 (May 1984).

. Fritzson, P., "Towards a distributed programming environment based on incremental compilation," Linkφping Studies in Science and Technology Dissertation No. 109, Dept. of Comp. and Inf. Sci., Linkφping University, Linkφping, Sweden (1984).

. Schwartz, M., Delisle, N., and Begwani, V., "Incremental compilation in Magpie," *Proceedings of the SIGPLAN 84 Symposium on Compiler Construction,* (Montreal, Can., June 20-22, 1984), *ACM SIGPLAN Notices* **19**(6) pp. 122-131 (June 1984).

. Tichy, W.F., "Smart recompilation," *ACM Trans. Program. Lang. Syst.* **8**(3) pp. 637-654 (July 1986).

. Schwanke, R.W. and Kaiser, G.E., "Technical Correspondence: Smarter recompilation," *ACM Trans. Program. Lang. Syst.* **10**(4) pp. 627-632 (October 1988).

. Tichy, W.F., "Technical Correspondence: Tichy's response to R.W. Schwanke and G.E. Kaiser's "Smarter recompilation"," *ACM Trans. Program. Lang. Syst.* **10**(4) pp. 633-634 (October 1988).

. Cooper, K.D. and Kennedy, K., "The impact of interprocedural analysis and optimization in the **R**^{n} programming environment," *ACM Trans. Program. Lang. Syst.* **8**(4) pp. 491-523 (October 1986).

. Burke, M. and Torczon, L., "Interprocedural optimization: Eliminating unnecessary recompilation," *ACM Trans. Program. Lang. Syst.*, (). (To appear.)

Document preparation

. Chamberlin, D.D., King, J.C., Slutz, D.R., Todd, S.J.P., and Wade, B.W., "JANUS: An interactive system for document composition," *Proceedings of the ACM SIGPLAN/SIGOA Symposium on Text Manipulation,* (Portland, OR, June 8-10, 1981), *ACM SIGPLAN Notices* **16**(6) pp. 82-91 (June 1981).

. Chamberlin, D.D., "Document convergence in an interactive formatting system," *IBM Systems Journal* **31**(1) pp. 58-72 (January 1987).

. Chen, P. and Harrison, M.A., "Multiple representation document development," *IEEE Computer* **21**(1) pp. 15-31 (January 1988).

. Chen, P., "A multiple-representation paradigm for document development," Ph.D. dissertation and Tech. Rep. UCB/CSD 88/436, Dept. of Electrical Engineering and Computer Science, University of California–Berkeley, Berkeley, CA (1988).

. Harrison, M.A. and Munson, E.V., "On integrated bibliography processing," *Electronic Publishing* **2**(4) pp. 193-210 (December 1989).

. Harrison, M.A. and Munson, E.V., "Numbering document components," *Electronic Publishing* **4**(1)(January 1991).

. Brooks, K.P., "A two-view document editor with user-definable document structure," Technical Report 33, DEC Systems Research Center, Palo Alto, CA (November 1988).

VLSI design tools

. Ousterhout, J.K., Hamachi, G.T., Mayo, R.N., Scott, W.S., and Taylor, G.S., "Magic: A VLSI layout system," pp. 152-159 in *Proceedings of the Twenty-First Design Automation Conference*, IEEE Computer Society, Washington, DC (1984).

. Taylor, G.S. and Ousterhout, J.K., "Magic's incremental design-rule checker," pp. 160-165 in *Proceedings of the Twenty-First Design Automation Conference*, IEEE Computer Society, Washington, DC (1984).

. Scott, W.S. and Ousterhout, J.K., "Plowing: Interactive stretching and compaction in Magic," pp. 166-172 in *Proceedings of the Twenty-First Design Automation Conference*, IEEE Computer Society, Washington, DC (1984).

. Ousterhout, J.K., "Corner stitching: A data-structuring technique for VLSI layout tools," *IEEE Transactions on Computer-Aided Design* **CAD-3**(1) pp. 87-100 (January 1984).

6. Implementation Frameworks

Graph annotation as a specification paradigm

. Bricklin, D. and Frankston, B., *VisiCalc Computer Software Program for the Apple II and II Plus,* Personal Software, Inc., Sunnyvale, CA (1979).

. Alpern, B., Carle, A., Rosen, B., Sweeney, P., and Zadeck, K., "Graph attribution as a specification paradigm," *Proceedings of the ACM SIGSOFT/SIGPLAN Software Engineering Symposium on Practical Software Development Environments,* (Boston, MA, November 28-30, 1988), *ACM SIGPLAN Notices* **24**(2) pp. 121-129 (February 1989).

Incremental constraint solvers

. Borning, A.H., "ThingLab—A constraint-oriented simulation laboratory," Ph.D. dissertation, Comp. Sci. Dept., Stanford University, and Tech. Rep. SSL-79-3, Xerox Palo Alto Research Center, Palo Alto, CA (July 1979).

. Konopasek, M. and Jayaraman, S., *The TK!Solver Book,* Osborne/McGraw-Hill, Berkeley, CA (1984).

Systems for generating language-sensitive editors

. Reps, T. and Teitelbaum, T., *The Synthesizer Generator: A System for Constructing Language-Based Editors,* Springer-Verlag, New York, NY (1988).

. Reps, T. and Teitelbaum, T., *The Synthesizer Generator Reference Manual: Third Edition,* Springer-Verlag, New York, NY (1988).

. Borras, P., Clément, D., Despeyroux, T., Incerpi, J., Kahn, G., Lang, B., and Pascual, V., "CENTAUR: The system," *Proceedings of the ACM SIGSOFT/SIGPLAN Software Engineering Symposium on Practical Software Development Environments,* (Boston, MA, November 28-30, 1988), *ACM SIGPLAN Notices* **24**(2) pp. 14-24 (February 1989).

. Bahlke, R. and Snelting, G., "The PSG system: From formal language definitions to interactive programming environments," *ACM Trans. Program. Lang. Syst.* **8**(4) pp. 547-576 (October 1986).

. Ballance, R.A., Graham, S.L., and Van De Vanter, M.L., "The Pan language-based editing system," *ACM Trans. Software Engineering and Methodology* **1**(1) pp. 95-127 (January 1992).

. Klint, P. (ed.), "The ASF+SDF Meta-environment user's guide," Draft, Computer Science/Department of Software Technology, Center for Mathematics and Computer Science (CWI), Amsterdam, The Netherlands (1992).

Compilation techniques for very high-level imperative languages

. Paige, R. and Koenig, S., "Finite differencing of computable expressions," *ACM Trans. Program. Lang. Syst.* **4**(3) pp. 402-454 (July 1982).

. Paige, R., "Transformational programming—applications to algorithms and systems," pp. 73-87 in *Conference Record of the Tenth ACM Symposium on Principles of Programming Languages,* (Austin, TX, Jan. 24-26, 1983), ACM, New York, NY (1983).

. Goldberg, A. and Paige, R., "Stream processing," in *Conference Record of the 1984 ACM Symposium on Lisp and Functional Programming,* (Austin, TX, August 6-8, 1984), ACM, New York, NY (1984).

. Paige, R., "Programming with invariants," *IEEE Software* **3**(1) pp. 56-69 (January 1986).

. Cai, J. and Paige, R., "Binding performance at language design time," pp. 85-97 in *Conference Record of the Fourteenth ACM Symposium on Principles of Programming Languages,* (Munich, W. Germany, January 1987), ACM, New York, NY (1987).

. Cai, J. and Paige, R., "Program derivation by fixed point computation," *Science of Computer Programming* **11** pp. 197-261 (1988/89).

. Cai, J. and Paige, R., "Languages polynomial in the input plus output," in *Proceedings of the Second International Conference on Algebraic Methodology and Software Technology (AMAST),* (Iowa City, Iowa, May 22-25, 1991), (1991).

. Hoover, R., "Alphonse: Incremental computation as a programming abstraction," *Proceedings of the ACM SIGPLAN 92 Conference on Programming Language Design and Implementation,* (San Francisco, CA, June 17-19, 1992), *ACM SIGPLAN Notices* **27**(7) pp. 261-272 (July 1992).

7. Other Problems Related to Incremental Computation

Sensitivity analysis

. Bertsekas, D.P., *Linear Network Optimization: Algorithms and Codes,* The M.I.T. Press, Cambridge, MA (1991).

Parametric problems

. Gallo, G., Grigoriadis, M.D., and Tarjan, R.E., "A fast parametric maximum flow algorithm and applications," *SIAM J. Computing* **18**(1) pp. 30-55 (February 1989).

Continuous execution

. Henderson, P. and Weiser, M., "Continuous execution: The Visiprog environment," in *Proceedings of the Eighth IEEE International Conference on Software Engineering,* IEEE Computer Society, Washington, DC (1985).