Notes de cours : DEA d'informatique
LIFO – Année 1998-1999

Analyse de flots de données : bibliographie

François Thomasset
INRIA

23 février 2001

Quelques points de repère

- Introduction à la compilation : [ASU86, WM94, Wol96, Muc97, NNH99]
- Flot de contrôle : [JC97, Wol91, Ram99, Hav97]
- État de l'art sur l'analyse de flots de données : [MR90, KKS98, Kno98, Kno99, CF99]
- Treillis : [DP90]
- Connections de Galois : [Aar92]
- Méthodes itératives : [Kil73, HU73, KU76, HDT87, Dwy95]
- Méthodes d'élimination : [HU72, Ull73, HU74, Ken76, RP86, TH92, Sha80, Ros77b, Sre95]
- Forme à Assignation Unique (*"SSA : Static Single Assignment"*) : [AWZ88, RWZ88, CFR+89, CG93, Hav93, Hav94, MJ92, JP93, Joh94, BM94, Ste95, Sre95, SG95, SGL96, Cli95b, Cli95a, TP95, BCHS98, CCK+97, KS98, KCL+99, Won99]
- Forme à Assignation Unique gardée (*"Gated SSA"*) : [BMO90, Hav93, Hav94, Sto95]
- Analyse interprocédurale : [Hav94]
- Analyse interprocédurale, Analyse des pointeurs : "Points-To" [Ban79, Wei80, Mye81, MJ81a, Cou86, CG93, LRZ93, ZRL96, CBC93, LR91, LR92, Lan92, WL95, JH94, HRS95b, KS94, Sto95, Wol96, KRS96, BCCH97, LH96, CG93, GH95, GH96, Ghi98, Wil97, Zha98]
- Analyse des pointeurs : "Shape Analysis" : [LH88, CWZ90, Deu92, Deu94, HHN92, HHN94, Gau97, SRW96, SRW98, FGL96, HH98, BRS99]
- Analyse des pointeurs : "context insensitive" : [HH98, And94, Ste96b, Ste96a]
- *Program Dependence Graph* : [OO84, HPR88, CF89, FOW77, CMN91]
- Interprétation abstraite : [Flo67, Weg75, CC76b, CC76a, CC79, Cou81, CC92a, CC92b, Cou98, QG97, MN83, AH87, Wad88, RK92, HV92, AJ90, HY91, Bur91, Bou92, Bou93, FGL96, Cou96, Mar93, JN94, ACG99]
- Recherche des relations linéaires entre variables entières d'un programme : [Kar76, CH78, Gra89, Gra90]
- Interprétation abstraite et programmation logique : [Mar93, CC92a]

– Analyse des dépendances pour la parallélisation : analyse des références aux tableaux : [Ban88a, Ban88b, Fea88, Fea91, GS90, May92, Pug92, PW93, PW94, PW98, CK88, Wol96, HP92, HP96]

– Application à la parallélisation : [HN90, CI96, JH94, Wol96]

– Typage des langages orientés objet : [Lar94, Cas97]

– Logique modale et analyse statique : [Ste91]

– "*Slicing*" : [Wei84, OO84, Bin93, JR94, RHSR94, RR95, Tip95, Gal96]

– Privatisation de tableaux : [CI96, GLL97]

Abréviations

CACM Communications of the ACM

ICALP International Colloquium on Automata, Languages, and Programming
(Les actes sont reproduits par SIGPLAN Notices)

IEEE-TSE IEEE, Transactions on Software Engineering

JACM Journal of the ACM

LNCS Lecture Notes in Computer Sciences

PLDI Conference on Programming Language Design and Implementation

POPL Annual ACM Symposium on Principles of Programming Languages

TOPLAS ACM Transactions on Programming Languages and Systems

Références

[Aar92] C.J. Aarts. Galois Connections Presented Calculationally. Graduating Dissertation, Eindhoven University of Technology, July 1992.
`ftp://ftp.win.tue.nl/pub/math.prog.construction/galois.dvi.Z`.

[ABS94] T.M. Austin, S.E. Breach, and G.S. Sohi. Efficient Detection of All Pointer and Array Access Errors. In *PLDI*, pages 290–301. ACM, 1994.
`ftp://ftp.cs.wisc.edu/sohi/papers/1994/pldi.safe-c.ps.gz|`.

[AC76] F.E. Allen and J. Cocke. A Program Data Flow Analysis Procedure. *CACM*, 19(3):137–147, 1976.

[ACG99] Isabelle Attali, Denis Caromel, and Romain Guider. Static analysis of java for distributed and parallel programming. Rapport de recherche de l'INRIA RR-3634, INRIA, Mars 1999.
`ftp://ftp.inria.fr/INRIA/publication/publi-ps-gz/RR/RR-3634.ps.gz`.

[AH87] Samson Abramsky and Chris Hankin, editors. *Abstract Interpretation of Declarative Languages*. Ellis Horwood series in computers and their applications. J. Wiley & sons, 1987.

[AHLT99] S. Alstrup, D. Harel, P.W. Lauridsen, and M. Thorup. Dominators in linear time. *SIAM Journal on Computing*, 28(6):2117–2132, 1999.

[AJ90] Samson Abramsky and Thomas P. Jensen. A Relational Approach to Strictness Analysis for Higher-Order Polynomial Functions. In *18th POPL*, pages 49–54. ACM, 1990.

[AL95] R.Z. Altucher and William Landi. An Extended Form of Must Alias Analysis for Dynamic Allocation. In *POPL*, pages 74–84. ACM, 1995.

[And94] Lars Ole Andersen. *Program Analysis and Specialization for the C Programming Language*. Ph.d. thesis, DIKU, University of Copenhagen, May 1994. DIKU report 94/19, `ftp://ftp.diku.dk/diku/semantics/papers/D-203.dvi.Z`.

[Ass96] Uwe Assmann. How To Uniformly Specify Program Analysis and Transformation. In P. Fritzson, editor, *Compiler Construction (CC)*. Springer, 1996. `http://i44s11.info.uni-karlsruhe.de:80/~assmann/`.

[ASU86] A.V. Aho, R. Sethi, and J.D. Ullman. *Compilers : principles, techniques and tools*. Addison Wesley, 1986. Traduction française : "*Compilateurs : principes, techniques et outils*", Interéditions, 1989.

[AU76] A.V. Aho and J.D. Ullman. Node Listings for Reducible Flow Graphs. *Journal of Computer and System Sciences*, 13(3):286–299, December 1976.

[AWZ88] Bowen Alpern, Mark N. Wegman, and F.K. Zadeck. Detecting Equality of Variables in Programs. In *15th POPL*, pages 1–11. ACM, 1988.

[BA98] Rastislav Bodik and Sadun Anik. Path-sensitive value-flow analysis. In *25th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, San Diego, California*, pages 237–251. ACM, January 1998. `http://www.cs.wisc.edu/~bodik/research/popl98.ps`.

[Ban79] John P. Banning. An Efficient Way to find the Side Effects of Procedure Calls and the Aliases of Variables. In *6th POPL*, pages 29–41. ACM, 1979.

[Ban88a] Utpal Banerjee. *Dependence analysis for supercomputing*. Kluwer Academic, 1988.

[Ban88b] Utpal Banerjee. An introduction to a formal theory of dependence analysis. *The Journal of Supercomputing*, 2(2):133–149, October 1988.

[BCC00] D. Barthou, A. Cohen, and J.-F. Collard. Maximal static expansion. *Int. Journal of Parallel Programming*, 28(3):213–243, June 2000.
Ancienne version : Proceedings of POPL'98, pp. 98-106.
`http://www.prism.uvsq.fr/~bad/ps/mse.ps.gz`.

[BCCH97] Michael Burke, Paul Carini, Jong-Deok Choi, and Michael Hind. Interprocedural pointer alias analysis. IBM Research Report 21055, State University of New York, New Paltz, New York, December 1997.
`http://www.mcs.newpaltz.edu/~hind/papers/RC21055.ps`.

[BCHS98] P. Briggs, K. D. Cooper, T. J. Harvey, and L. T. Simpson. Practical Improvements to the Construction and Destruction of Static Single Assignment Form. *Software–Practice and Experience*, 28(8):859–881, July 1998.
`http://www.cs.rice.edu/~harv/ssa.ps`.

[BG95] Rastislav Bodik and Rajiv Gupta. Array Data-Flow Analysis for Load-Store Optimizations in Superscalar Architectures. In *8th Annual Workshop on Languages and Compilers for Parallel Computing*, pages 1–15. Springer Verlag, LNCS 1033, 1995.
`http://www.cs.arizona.edu/people/gupta/research/Publications/Comp/lcpc95.ps`.

[BGS93] D. Berson, R. Gupta, and M.L. Soffa. URSA: A Unified ReSource Allocator for Registers and Functional Units in VLIW Architectures. In *Conference on Architectures and Compilation Techniques for Fine and Medium Grain Parallelism, IFIP Transactions A-23*, pages 243–254, Orlando, Florida, January 1993.
`http://www.cs.arizona.edu/people/gupta/research/Publications/Comp/pact93.ps`.

[BGS94] D. F. Bacon, S. L. Graham, and O. J. Sharp. Compiler transformations for high-performance computing. *ACM Computing Surveys*, 26(4):345–420, December 1994.
`ftp://ftp.cs.berkeley.edu/ucb/parallel/survey.ps`.

[BGS95] D. Berson, R. Gupta, and M.L. Soffa. GURRR: A Global Unified Resource Requirements Representation. In *ACM SIGPLAN Workshop on Intermediate Representations*, San Francisco, California, January 1995.
`http://www.cs.arizona.edu/people/gupta/research/Publications/Comp/gurr.ps`.

[BGS98] R. Bodik, R. Gupta, and M.L. Soffa. Complete removal of redundant expressions. *ACM SIGPLAN Notices*, 33(5):1–14, May 1998. Proceedings of ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI'98).
`http://www.cs.arizona.edu/people/gupta/research/Publications/Comp/pldi98.ps`.

[BGS99] R. Bodik, R. Gupta, and M.L. Soffa. Load-reuse analysis: Design and evaluation. *ACM SIGPLAN Notices*, 34(5):64–76, May 1999. Proceedings of ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI'99).
`http://www.cs.wisc.edu/~bodik/research/pldi99.ps`.

[Bin93] David Binkley. Precise executable interprocedural slices. *ACM Letters on Programming Languages and Systems*, 2(1-4):31–45, March-Dec 1993.

[BKRW98] A.L. Buchsbaum, H. Kaplan, A. Rogers, and J.R. Westbrook. A new, simpler linear-time dominators algorithm. *ACM Transactions on Programming Languages and Systems*, 20(6):1265–1296, November 1998.

[BM94] Marc M. Brandis and Hanspeter Mössenböck. Single-Pass Generation of Static Single-Assignment Form for Structured Languages. *ACM Transactions on Programming Languages and Systems*, 16(6):1684–1698, November 1994.
Technical Report, Johannes Kepler University Linz, Austria, Number CS-SSW-P94-11, November 1, 1994.
`ftp://ftp.ssw.uni-linz.ac.at/pub/Reports/CS-SSW-P94-11.ps.Z`.

[BMO90] Robert A. Ballance, Arthur B. Maccabe, and Karl J. Ottenstein. The program dependence web : a representation supporting control-, data-, and demand-driven interpretation of imperative languages. In *Programming Language Design and Implementation (PLDI'90)*, pages 257–271. ACM Press, June 1990.

[Bou92] François Bourdoncle. *Sémantiques des Langages Impératifs d'Ordre Supérieur et Interprétation Abstraite*. Thèse de doctorat, École Polytechnique, 1992.
`http://www.ensmp.fr/~bourdonc/these.html`.

[Bou93] François Bourdoncle. Abstract Debugging of Higher-Order Imperative Languages. *SIGPLAN Notices*, 28(6):46–55, June 1993. Proceedings of PLDI'93.
`http://www.ensmp.fr/~bourdonc/pldi93.html`.

[BP96] Gianfranco Bilardi and Keshav Pingali. A Framework for Generalized Control Dependence. In *Proceedings of the ACM SIGPLAN '96 conference on Programming language design and implementation (PLDI'96)*, pages 291–300, May 1996.

[BP00] Gianfranco Bilardi and Keshav Pingali. The Static Single Assignment Form and its Computation. Research report, Cornell University, 2000.
`http://www.cs.cornell.edu/Info/Projects/Bernoulli/papers/ssa.ps`.

[BR90] M.G. Burke and Barbara G. Ryder. A Critical Analysis of Incremental Iterative Data Flow Analysis Algorithms. *IEEE-TSE*, 7(16):723–728, July 1990.

[BRS99] M. Benedikt, T. Reps, and M. Sagiv. A decidable logic for describing linked data structures. In *European Symposium On Programming (ESOP)*, pages 2–19. Springer, Lecture Notes in Computer Science # 1576, March 1999.
`http://www.math.tau.ac.il/~sagiv/esop99.ps`.

[BT93] Preston Briggs and Linda Torczon. An Efficient Representation for Sparse Sets. *ACM Letters on Programming Languages and Systems*, 2(1-4):59–69, March 1993. Sets, DFA.

[Bur90] Geoffrey L. Burn. A Relationship between Abstract Interpretation and Projection Analysis. In *17th POPL*, pages 151–156. ACM, 1990.

[Bur91] Geoffrey L. Burn. *Lazy Functional Languages: Abstract Interpretation and Compilation.* Research Monographs in Parallel and Distributed Computing. Pitman, 1991.

[Cas97] Giuseppe Castagna. *Object-oriented programming: a unified foundation.* Progress in theoretical computer science. Birkhäuser, 1997.

[CBC93] J.D. Choi, M. Burke, and P. Carini. Efficient Flow-Sensitive Interprocedural Computation of Pointer-Induced Aliases and Side Effects. In *20th POPL*, pages 232–245, 1993.

[CC76a] Patrick Cousot and Radhia Cousot. Abstract Interpretation: a Unified Lattice Model for Static Analysis of Programs by construction or Approximation of FixPoints. In *4th POPL, Los Angeles*, pages 238–252, 1976.
`http://www.dmi.ens.fr/~cousot/COUSOTpapers/POPL77.shtml`.

[CC76b] Patrick Cousot and Radhia Cousot. Static Determination of Dynamic Properties of programs. In Bernard Robinet, editor, *Symposium on Programming, Paris*, pages 106–130. Dunod informatique, 1976.
`http://www.dmi.ens.fr/~cousot/COUSOTpapers/ISOP76.shtml`.

[CC79] Patrick Cousot and Radhia Cousot. Systematic Design of Program Analysis Frameworks. In *6th POPL, San Antonio, Texas*, pages 269–282, 1979.
`http://www.dmi.ens.fr/~cousot/COUSOTpapers/POPL79.shtml`.

[CC92a] Patrick Cousot and Radhia Cousot. Abstract Interpretation and Applications to Logic Programs. *Journal of Logic Programming*, 13(2-3):103–180, July 1992.
`http://www.dmi.ens.fr/~cousot/COUSOTpapers/JLP92.shtml`.

[CC92b] Patrick Cousot and Radhia Cousot. Comparing the Galois Connection and Widening/Narrowing Approaches to Abstract Interpretation. In *PLILP'92*, pages 269–295. Springer-Verlag, 1992. LNCS#631
`http://www.dmi.ens.fr/~cousot/COUSOTpapers/PLILP92.shtml`.

[CCF91] J.D. Choi, R. Cytron, and J. Ferrante. Automatic Construction of Sparse Data Flow Evaluation Graphs. In *8th POPL*, pages 55–66. ACM, 1991.

[CCK+97] Fred Chow, Sun Chan, Robert Kennedy, Shin-Ming Liu, Raymond Lo, and Peng Tu. A new algorithm for partial redundancy elimination based on SSA form. *ACM SIGPLAN Notices*, 32(5):273–286, June 1997. Proceedings of the ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI-97).

[CDG96] Craig Chambers, Jeffrey Dean, and David Grove. Whole-program optimization of object-oriented languages. Research Report UW-CSE-96-06-02, Department of Computer Science and Engineering, University of Washington, 1996.
`ftp://ftp.cs.washington.edu/homes/chambers/whole-program.ps.gz`.

[CF89] Robert Cartwright and Matthias Felleisen. The Semantics of Program Dependence. In *Proceedings of the SIGPLAN '89 Conference on Programming Language Design and Implementation (PLDI'89)*, pages 13–27, June 1989. ACM SIGPLAN Notices, 24(7).
`http://www.cs.rice.edu/CS/PLT/Publications/pldi89-cf.ps.gz`.

[CF99] A. Cortesi and G. Filé. *Static Analysis.* Number LNCS 1694 in Lecture Notes in Computer Science. Springer, 1999. Proceedings of 6th International Symposium, SAS'99, Venice, Italy, `http://link.springer-ny.com/link/service/series/0558/tocs/t1694.htm`.

[CFR+89] Ron Cytron, Jeanne Ferrante, Barry K. Rosen, Mark N. Wegman, and F.K. Zadeck. An Efficient Method of Computing Static Single Assignment Form. In *16th POPL*, pages 25–35. ACM, 1989.

[CG93] Ron Cytron and Reid Gershbein. Efficient Accomodation of May-Alias Information in SSA Form. *SIGPLAN Notices*, 28(6):36–45, June 1993. PLDI'93.

[CH78] Patrick Cousot and Nicolas Halbwachs. Automatic Discovery of Linear Restraints among Variables of a Program. In *5th POPL*, 1978.
`http://www.di.ens.fr/~cousot/COUSOTpapers/POPL78.shtml`.

[CI96] Béatrice Creusillet and François Irigoin. Exact versus Approximate Array Region Analyses. In *Ninth International Workshop on Languages and Compilers for Parallel Computing.* LNCS, 1996.
`http://www.cri.ensmp.fr:80/~creusil/`.

[CK88] David Callahan and Ken Kennedy. Analysis of Interprocedural Side Effects in a Parallel Programming Environment. *Journal of Parallel and Distributed Computing*, 5:517–550, 1988.

[CK94] Steven Carr and Ken Kennedy. Improving the ratio of memory operations to floating-point operations in loops. *ACM Transactions on Programming Languages and Systems*, 16(6):1768–1810, November 1994.
`ftp://cs.mtu.edu/pub/carr/unroll.ps.gz`.

[Cli95a] Clifford Click. *Combining Analyses, Combining Optimizations.* PhD thesis, Rice University, February 1995.
`http://www.cs.rice.edu/~cliffc/`.

[Cli95b] Clifford Click. Global Code Motion, Global Value Numbering. *SIGPLAN Notices*, 30(6):246–257, June 1995. (PLDI'95).

[CMN91] Jong-Deok Choi, Barton P. Miller, and Robert H. B. Netzer. Techniques for debugging parallel programs with flowback analysis. *ACM Transactions on Programming Languages and Systems*, 13(4):491–530, April 1991.

[Col97] Christian S. Collberg. Reverse interpretation + mutation analysis = automatic retargeting. In *Proceedings of the ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI-97)*, pages 57–70. ACM, June 1997. ACM SIGPLAN Notices, Vol. 32, 5,
`http://www.cs.auckland.ac.nz/~collberg/Research/Publications/`.

[Col99] Jean-François Collard. Array SSA for Explicitly Parallel Programs. In P. Amestoy, P. Berger, M. Daydé, and I. S. Duff, editors, *Proceedings of Euro-Par'99*, pages 383–400. Springer-Verlag, LNCS 1685, September 1999.
`http://www.prism.uvsq.fr/~jfc/diff_europar99.ps.gz`.

[Cou81] Patrick Cousot. Semantic Foundations of Program Analysis. In Steven S. Muchnick and Neil D. Jones, editors, *Program Flow Analysis: Theory and Applications*, chapter 10, pages 303–342. Prentice-Hall, 1981.
`http://www.dmi.ens.fr/~cousot/COUSOTpapers/PFA81.shtml`.

[Cou86] Deborah S. Coutant. Retargetable high level analysis. In *Proceedings of 13th POPL Symposium*, pages 110–118. ACM, 1986.

[Cou96] Patrick Cousot. Abstract Interpretation. *ACM Computing Surveys*, 28(2):324–328, June 1996.
`http://www.dmi.ens.fr/~cousot/COUSOTpapers/CS96.shtml`.

[Cou98] Patrick Cousot. The calculational design of a generic abstract interpreter. In M. Broy and R. Steinbrüggen, editors, *Calculational System Design. NATO ASI Series F.* IOS Press, Amsterdam, 1998.
`http://www.dmi.ens.fr/~cousot/COUSOTpapers/Marktoberdorf98.shtml`.

[CP89] Jiazhen Cai and Robert Paige. Program Derivation by Fixed Point Computation. *Science of Computer Programming*, 11:197–261, 1989.

[CWZ90] David R. Chase, Mark Wegman, and F. K. Zadeck. Analysis of Pointers and Structures. In *17th POPL*, pages 296–310. ACM, 1990.

[Deu90] Alain Deutsch. On Determining Lifetime and Aliasing of Dynamically Allocated Data in Higher-Order Functional Specifications. In *7th POPL*, pages 157–168. ACM Press, 1990.

[Deu92] Alain Deutsch. A Storeless Model of Aliasing and its Abstractions using Finite Representations of Right-Regular Equivalence Relations. In *IEEE, International Conference on Computer Languages*, pages 2–13, 1992.

[Deu94] Alain Deutsch. Interprocedural May-Alias Analysis for Pointers: beyond k-limiting. *SIGPLAN Notices*, 6:230–241, 1994.

[DGS93] Evelyn Duesterwald, Rajiv Gupta, and Mary Lou Soffa. A Practical Data Flow Framework for Array Reference Analysis and its Application in Optimizations. *SIGPLAN Notices*, 28(6):68–77, June 1993. PLDI'93,
`http://www.cs.arizona.edu/people/gupta/research/Publications/Comp/pldi93.ps`.

[DGS94] Evelyn Duesterwald, Rajiv Gupta, and Mary Lou Soffa. Reducing the Cost of Data Flow Analysis by Congruence Partitioning. In Peter A. Fritzson, editor, *Compiler Construction*, pages 357–373. Springer-Verlag, LNCS # 786, 1994.
`http://www.cs.arizona.edu/people/gupta/research/Publications/Comp/cc94.ps`.

[DGS95] Evelyn Duesterwald, Rajiv Gupta, and Mary Lou Soffa. Demand-Driven Computation of Interprocedural Data Flow. In *22nd POPL*, pages 37–48. ACM, 1995.
`http://www.cs.arizona.edu/people/gupta/research/Publications/Comp/popl95.ps`.

[DGS96] Evelyn Duesterwald, Rajiv Gupta, and Mary Lou Soffa. A Demand-Driven Analyzer for Data Flow Testing at the Integration Level. In *IEEE International Conference on Software Engineering*, pages 575–586, March 1996. Berlin, Germany.
`http://www.cs.arizona.edu/people/gupta/research/Publications/SE/icse96.ps`.

[DGS97] Evelyn Duesterwald, Rajiv Gupta, and Mary Lou Soffa. A practical framework for demand-driven interprocedural data flow analysis. *ACM Transactions on Programming Languages and Systems*, 19(6):992–1030, November 1997.
`http://www.cs.arizona.edu/people/gupta/research/Publications/Comp/toplas98.ps`.

[Dha88] D.M. Dhamdhere. A Fast Algorithm for Code Movement Optimization. *SIGPLAN Notices*, 23(10):172–180, 1988. PLDI'88.

[DP90] B.A. Davey and H.A. Priestley. *Introduction to Lattices and Order*. Cambridge University Press, 1990.

[Dwy95] Matthew B. Dwyer. *Data Flow Analysis for Verifying Correctness Properties of Concurrent Programs.* Phd dissertation, University of Massachusetts, Amherst, MA, September 1995.
`http://www.cis.ksu.edu/~dwyer/papers/thesis.ps`.

[EGH94] Maryam Emami, Rakesh Ghiya, and Laurie J. Hendren. Context-Sensitive Interprocedural Points-To Analysis in the Presence of Function Pointers. *SIGPLAN Notices*, 6:242–256, 1994.
`http://www-acaps.cs.mcgill.ca/~hendren/`.

[EH94] Ana M. Erosa and Laurie J. Hendren. Taming Control Flow: A Structured Approach to Eliminating GOTO Statements. In *International Conference on Computer Languages*, pages 229–240, 1994.
`ftp://ftp-acaps.cs.mcgill.ca/pub/doc/papers/ICCL94.ps.gz`.

[Fea88] Paul Feautrier. Parametric Integer Programming. *RAIRO, Recherche Opérationnelle*, 22(3):243–268, September 1988.
`http://www.prism.uvsq.fr/public/paf/parmprog.ps`.

[Fea91] Paul Feautrier. Dataflow Analysis of Array and Scalar References. *International Journal of Parallel Processing*, 20(1):23–53, 1991.
`http://www.prism.uvsq.fr/public/paf/dataflow.ps`.

[FGL96] Pascal Fradet, Ronan Gaugne, and Daniel LeMetayer. An Inference Algorithm for the Static Verification of Pointer Manipulation. Rapport de Recherche RR-2895, IRISA, Rennes, Campus Universitaire de Beaulieu, Mai 1996.
`http://zenon.inria.fr:8003/Equipes/LANDE-eng.html`.

[Flo67] Robert W. Floyd. Assigning Meaning to Programs. In Jacob T. Schwartz, editor, *Mathematical Aspects of Computer Science*, pages 19–32, New York, 1967. American mathematical society.

[Fon77] Amelia C. Fong. Generalized Common Subexpressions in Very High Languages. In *4th POPL*, pages 48–57, Los Angeles, Calif., 1977.

[FOW77] J. Ferrante, K.J. Ottenstein, and J.D. Warren. The Program Dependence Graph and its use in Optimization. *ACM Transactions on Programming Languages and Systems*, 9(3):319–349, July 1977.

[Gal96] Keith Gallagher. A survey of program slicing. Research report, Loyola College, Computer Science Department, 1996.
`http://www.cs.loyola.edu/~kbg/survey.ps.gz`.

[Gau97] Ronan Gaugne. *Techniques d'analyse statique pour l'aide à la mise au point de programmes avec manipulation explicite de pointeurs*. Thèse de doctorat, Université de Rennes 1, Octobre 1997.
`ftp://ftp.irisa.fr/pub/techreports/theses/1997/gaugne.ps.gz`.

[Ger96] Michael P. Gerlek. *Beyond Induction Variables: Detecting and Classifying Sequences Using a Demand-driven SSA Form*. Ph.D. Thesis, Oregon Graduate Institute of Science and Technology, Computer Science Dept., January 1996.
`ftp://cse.ogi.edu/pub/tech-reports/1995/95-TH-004.ps.gz`.

[GH95] Rakesh Ghiya and Laurie J. Hendren. Connection Analysis: a Practical Interprocedural Heap Analysis for C. In C.H. Huang, P. Sadayappan, U. Banerjee, D. Gelernter, A. Nicolau, and D. Padua, editors, *8th Workshop on Languages and Compilers for Parallel Computing*. Springer-Verlag, LNCS # 1033, 1995.
`ftp://ftp-acaps.cs.mcgill.ca/pub/doc/papers/lcpc95.ps.gz`.

[GH96] Rakesh Ghiya and Laurie J. Hendren. Is it a Tree, a DAG, or a Cyclic Graph? a Shape Analysis for Heap-Directed Pointers in C. In *23rd POPL*, pages 1–15. ACM, 1996.
`ftp://ftp-acaps.cs.mcgill.ca/pub/doc/papers/POPL96.ps.gz`.

[Ghi98] Rakesh Ghiya. *Putting Pointer Analysis to Work*. Ph.d. thesis, School of Computer Science, McGill University, January 1998.
`http://www-acaps.cs.mcgill.ca/~ghiya/ftp/thesis.ps.gz`.

[GKT91] Gina Goff, Ken Kennedy, and Chau-Wen Tseng. Practical dependence testing. In *SIGPLAN '91 Conference on Programming Language Design and Implementation*, pages 15–29. ACM, June 1991. SIGPLAN Notices, 26 (6), June 1991

CRPC Repport CRPC-TR90103-S
`ftp://softlib.rice.edu/pub/CRPC-TRs/reports/CRPC-TR90103-S.ps.gz`.

[GLL97] Junjie Gu, Zhiyuan Li, and Gyungho Lee. Experience with efficient array data flow analysis for array privatization. *SIGPLAN Notices*, 32(7):157–167, 1997. Proc. of PPoPP.

[Gou97] Valérie Gouranton. *Dérivation d'analyseurs dynamiques et statiques à partir de spécifications opérationnelles.* Thèse de doctorat, Université de Rennes I, No d'ordre 1845, Septembre 1997.
`ftp://ftp.irisa.fr/pub/techreports/theses/1997/gouranton.ps.gz`.

[GPPS99] R. Gupta, S. Pande, K. Psarris, and V. Sarkar. Compiling for parallel systems. *Parallel Computing journal*, 25:1741–1783, 1999.
`http://www.cs.arizona.edu/people/gupta/research/Publications/Comp/parco.ps`.

[Gra89] Philippe Granger. Static Analysis of Arithmetical Congruences. *Int. J. Computer. Math.*, 30:165–190, 1989.

[Gra90] Philippe Granger. Static Analysis of Linear Congruence Equalities among Variables of a Program. Rapport de Recherche 90/10, Ecole Polytechnique, Laboratoire d'Informatique, Novembre 1990.

[GS90] Thomas Gross and Peter Steenkiste. Structured dataflow analysis for arrays and its use in an optimizing compiler. *Software — Practice and Experience*, 20(2):133–155, February 1990.

[GSW95] Michael P. Gerlek, Eric Stoltz, and Michael Wolfe. Beyond Induction Variables: Detecting and Classifying Sequences Using a Demand-driven SSA Form. *ACM Transactions on Programming Languages and Systems*, 17(1):85–122, 1995.
`http://www.cse.ogi.edu/~stoltz/www.dir/papers/iv.ps`.

[Gup92] Rajiv Gupta. Generalized Dominators and Post-Dominators. In *19th POPL*, pages 246–257. ACM, January 1992. also: Information Processing Letters, Vol. 53, pages 193-200, 1995,
`http://www.cs.arizona.edu/people/gupta/research/Publications/Comp/ipl.ps`.

[Gup94] Rajiv Gupta. Optimizing Array Bound Checks using Flow Analysis. *ACM Letters on Programming Languages and Systems*, 2(1-4):135–150, March 1994.
`http://www.cs.pitt.edu/ gupta/research/Comp/loplas.ps`.

[GV95] Elana Granston and Alexander Veidenbaum. Combining flow and dependence analyses to expose redundant array accesses. Research Report CRPC-TR95545, Center for Research on Parallel Computation, Rice University, May 1995.
`ftp://softlib.rice.edu/pub/CRPC-TRs/reports/CRPC-TR95545.ps.gz`.

[GW76] Susan L. Graham and Mark N. Wegman. A Fast and Usually Linear Algorithm for Global Data Flow Analysis. *Journal of the ACM*, 23(1):172–202, January 1976.

[Har77] William H. Harrison. Compiler Analysis of the Value Ranges for Variables. *IEEE Transactions on Software Engineering*, SE-3(3):243–250, May 1977.

[Hav93] Paul Havlak. Construction of thinned gated single-assignment form. In *6th International Workshop on Languages and Compilers for Parallel Computing*, pages 477–499. Lecture Notes in Computer Science 768, Springer-Verlag, August 1993.

[Hav94] Paul Havlak. *Interprocedural Symbolic Analysis.* Ph.d. thesis, CRPC, Rice University, Houston, Texas, May 1994.
`ftp://softlib.rice.edu/pub/CRPC-TRs/reports/CRPC-TR94451-S.ps.gz`.

[Hav97] Paul Havlak. Nesting of reducible and irreducible loops. *ACM Transactions on Programming Languages and Systems*, 19(4):557–567, July 1997.

[HDT87] Susan Horwitz, Alan Demers, and Tim Teitelbaum. An Efficient General Iterative Algorithm for Dataflow Analysis. *Acta Informatica*, 24(6):679–694, 1987.

[Hen81] John Hennessy. Program Optimization and Exception Handling. In *8th POPL*, pages 200–206. ACM, 1981.

[HH98] R. Hasti and S. Horwitz. Using static single assignment form to improve flow-insensitive pointer analysis. In *PLDI'98 (Montréal, Canada)*, June 1998.
`http://www.cs.wisc.edu/wpis/papers/pldi98.ps`.

[HHN92] Laurie J. Hendren, Joseph Hummel, and Alexandru Nicolau. Abstractions for Recursive Pointer Data Structures: Improving the Analysis and Transformation of Imperative Programs. In *PLDI*, pages 249–260. ACM, 1992.
`ftp://ftp-acaps.cs.mcgill.ca/pub/doc/papers/PLDI92.ps.gz`.

[HHN94] Joseph Hummel, Laurie J. Hendren, and Alexandru Nicolau. A General Data Dependence Test, Pointer-Based Data Structures. In *PLDI*, pages 218–229. ACM, 1994.
`ftp://ftp-acaps.cs.mcgill.ca/pub/doc/papers/PLDI94b.ps.gz`.

[HN90] Laurie J. Hendren and Alexandru Nicolau. Parallelizing Programs with Recursive Data Structures. *IEEE, Transactions on Parallel and Distributed Systems*, 1(1):35–47, January 1990.

[HP92] M. Haghighat and C. Polychronopoulos. Symbolic program analysis and optimization for parallel compilers. Technical Report 1237, CSRD, Univ. of Illinois, August 1992.
Presented at the 5th Annual Workshop on Languages and Compilers for Parallel Computing, New Haven, CT, August 3-5, 1992.

[HP96] M. Haghighat and C. Polychronopoulos. Symbolic analysis for parallelizing compilers. *ACM Transactions on Programming Languages and Systems*, 18(4):477–518, July 1996.

[HPR88] Susan Horwitz, Jan Prins, and Thomas Reps. On the Adequacy of Program Dependence Graphs for Representing Programs. In *15th POPL*, pages 146–157. ACM, 1988.
`http://www.cs.wisc.edu/wpis/papers/popl88b.ps`.

[HR81] L. Howard Holley and Barry K. Rosen. Qualified Data Flow Problems. *IEEE Transactions on Software Engineering*, SE-7(3):60–78, January 1981.

[HRS95a] Susan Horwitz, Thomas Reps, and Mooly Sagiv. Demand Interprocedural Dataflow Analysis. In *SIGSOFT Symposium on Foundations of Software Engineering*. ACM, 1995.
`http://www.cs.wisc.edu/wpis/papers/tr1283r.ps`.

[HRS95b] Susan Horwitz, Thomas Reps, and Mooly Sagiv. Precise Interprocedural Dataflow Analysis via Graph Reachability. In *22nd POPL*, pages 49–61. ACM, 1995.
`http://www.cs.wisc.edu/wpis/papers/popl95.ps`.

[HU72] M.S. Hecht and J.D. Ullman. Flow Graph Reducibility. *SIAM J. Comput.*, 1(2):188–202, June 1972.

[HU73] M.S. Hecht and J.D. Ullman. Analysis of a Simple Algorithm for Global Data Flow Problems. In *POPL*, pages 207–217. ACM, 1973.

[HU74] M.S. Hecht and J.D. Ullman. Characterizations of Reducible Flow Graphs. *JACM*, 21(3):367–375, July 1974.

[HV92] R. Nigel Horspool and Jan Vitek. Static Analysis of Postcript Code. In *IEEE, International Conference on Computer Languages*, pages 14–23, 1992.

[HY91] Paul Hudak and Jonathan Young. Collecting interpretations of expressions. *ACM Transactions on Programming Languages and Systems*, 13(2):269–290, April 1991.

[JC97] Johan Janssen and Henk Corporaal. Making graphs reducible with controlled node splitting. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 19(6):1031–1052, November 1997.

[JH94] J. Justiani and Laurie J. Hendren. Supporting Array Dependence Testing for an Optimizing/Parallelizing C Compiler. In Peter A. Fritzson, editor, *5th International Conférence on Compiler Construction (CC'94)*, pages 309–323. Springer-Verlag, LNCS # 786, 1994. ACAPS Technical Memo 74, October 1993, McGill University
ftp://ftp-acaps.cs.mcgill.ca/pub/doc/memos/memo74.ps.gz.

[JM86] N. Jones and A. Mycroft. Data Flow Analysis of Applicative Programs using Minimal Functions Graphs. In *13th POPL*, pages 296–306. ACM, 1986.

[JN94] Neil D. Jones and Fleming Nielson. Abstract Interpretation: a Semantics-Based Tool for Program Analysis,. Research report, DIKU, June 1994. in Handbook of Logic in Computer Science, Oxford University Press, 1994, 527-629.
ftp://ftp.diku.dk/diku/semantics/papers/D-58.ps.

[Joh94] Richard Craig Johnson. *Efficient Program Analysis using Dependence Flow Graphs*. Ph.d. thesis, Cornell University, August 1994.
http://simon.cs.cornell.edu/Info/Projects/Bernoulli/.

[JP93] Richard Johnson and Keshav Pingali. Dependence-Based Program Analysis. *SIGPLAN Notices*, 28(6):78–89, June 1993. (PLDI'93)
http://simon.cs.cornell.edu/Info/Projects/Bernoulli/papers/pldi93.ps.

[JR94] Daniel Jackson and Eugene J. Rollins. A New Model of Program Dependences for Reverse Engineering. In *SIGSOFT'94*, pages 2–9. ACM, 1994.

[JT88] Suneel Jain and Carol Thompson. An Efficient Approach to Data Flow Analysis in a Multiple Pass global Optimizer. *SIGPLAN Notices*, 23(7):154–163, July 1988. PLDI'88.

[Kar76] M. Karr. Affine Relationships among Variables of a Program. *Acta Informatica*, 6:133–151, 1976.

[KCL+99] Robert Kennedy, Sun Chan, Shin-Ming Liu, Raymond Lo, Peng Tu, and Fred Chow. Partial redundancy elimination in SSA form. *ACM Transactions on Programming Languages and Systems*, 21(3):627–676, May 1999.

[Ken76] K. Kennedy. A Comparison of two Algorithms for Data flow analysis. *SIAM J. Comput.*, 5(1):158–180, March 1976.

[Kil73] G.A. Kildall. A Unified Approach to Global Program Optimization. In *POPL*, pages 194–206. ACM, 1973.

[KKS98] J. Knoop, D. Koschützki, and B. Steffen. Basic blocks: Living dinosaurs? In K. Koskimies, editor, *Compiler Construction*, pages 65–79. Springer, 1998.

[Kno98] Jens Knoop. *Optimal interprocedural program optimization: a new framework and its application*. Lecture notes in computer science, 0302-9743. Springer, 1998.

[Kno99] Jens Knoop. From dfa-frameworks to dfa-generators: A unifying multiparadigm approach. In *Proceedings of the 5th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'99) (Amsterdam, The Netherlands)*, pages 360–374. Springer-Verlag, Heidelberg, 1999. LNCS 1579
http://guiness.cs.uni-dortmund.de/~knoop/publications.html.

[KRS92] J. Knoop, O. Rüthing, and B. Steffen. Lazy Code Motion. *SIGPLAN Notices*, 27(7):224–234, 1992. (PLDI'92).
ftp://ftp.uni-passau.de/pub/local/parallel/papers/lcm-9208-a4.ps.Z.

[KRS93a] J. Knoop, O. Rüthing, and B. Steffen. Optimal Code Motion: Theory and Practice. MIP-Bericht 9310, Fakultät für Mathematik und Informatik, Universität Passau, December 1993. `ftp://ftp.uni-passau.de/pub/local/parallel/papers/ocm-9310-a4.ps.Z`.

[KRS93b] Jens Knoop, O. Rüthing, and B. Steffen. Lazy Strength Reduction. *Journal of Programming Languages*, 1:71–91, 1993. `ftp://ftp.uni-passau.de/pub/local/parallel/papers/lsr-jpl-a4.ps.Z`.

[KRS94] J. Knoop, O. Rüthing, and B. Steffen. Partial Dead Code Elimination. In *PLDI*, pages 147–158. ACM, 1994. `ftp://ftp.uni-passau.de/pub/local/parallel/papers/pde-pldi94-a4.ps.Z`.

[KRS96] Jens Knoop, O. Rüthing, and B. Steffen. Towards a Tool Kit for the Automatic Generation of Interprocedural Data Flow Analyses. *Journal of Programming Languages*, 4(4):211–246, 1996. `http://www.thomson.com:8866/jp/pdfpapers/jp040402.pdf`, `ftp://ftp.uni-passau.de/pub/local/parallel/papers/tk-9413-a4.ps.Z`.

[KS94] Jens Knoop and Bernhard Steffen. The Interprocedural Coincidence Theorem. In U. Kastens and P. Pfahler, editors, *Proceedings of the 4th International Conference on Compiler Construction (CC'92), Paderborn, Germany*, pages 124–13. Springer-Verlag, 1994. LNCS 641 `ftp://ftp.uni-passau.de/pub/local/parallel/papers/ict-cc92-a4.ps.Z`.

[KS98] Kathleen Knobe and Vivek Sarkar. Array SSA form and its use in parallelization. In *25th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 107–120, January 1998.

[KSV94] Jens Knoop, Bernhard Steffen, and Jürgen Vollmer. Parallelism for free: Efficient and Optimal Bitvector Analyses for Parallel Programs. Research Report MIP-9409, Universität Passau, August 1994.

[KU76] John B. Kam and Jeffrey D. Ullman. Global Data Flow Analysis and Iterative Algorithms. *Journal of the ACM*, 23(1):158–171, 1976.

[KU77] John B. Kam and Jeffrey D. Ullman. Monotone Data Flow Frameworks. *Acta Informatica*, 7:305–317, 1977.

[Lan92] William Landi. *Interprocedural Aliasing in the Presence of Pointers*. PhD thesis, Rutgers University, January 1992. `ftp://athos.rutgers.edu/pub/technical-reports/lcsr-tr-174.ps.Z`.

[Lar94] Jean-Marie Larchevêque. Interprocedural Type Propagation for Object-Oriented Languages. *Science of Computer Programming*, 22:257–282, 1994.

[LH88] J.R. Larus and P.N. Hilfinger. Detecting Conflicts between Structure Accesses. *SIGPLAN Notices*, 23(7):21–34, June 1988. (PLDI'88).

[LH96] C. Lapkowski and Laurie J. Hendren. Extended SSA numbering: Introducing SSA Properties to Languages with Multi-Level Pointers. In *CASCON'96, "Meeting of Minds"*, pages 64–75, 1996. `http://www-acaps.cs.mcgill.ca/~lapkow/`.

[LR91] William Landi and Barbara G. Ryder. Pointer-Induced Aliasing : a Clarification. In *18th POPL*, pages 93–103, 1991.

[LR92] William Landi and Barbara G. Ryder. A Safe Approximate Algorithm for Interprocedural Pointer Aliasing. *SIGPLAN Notices*, 27(7):235–248, July 1992. Proceedings of PLDI'92. `ftp://athos.rutgers.edu/pub/technical-reports/lcsr-tr-168.ps.Z`.

[LRZ93] William Landi, Barbara G. Ryder, and Sean Zhang. Interprocedural Modification Side Effect with Pointer Aliasing. *SIGPLAN Notices*, 28(6):56–67, June 1993. Proceedings of PLDI'93. `ftp://athos.rutgers.edu/pub/technical-reports/lcsr-tr-168.ps.Z`.

[LT79] Thomas Lengauer and Robert Endre Tarjan. A Fast Algorithm for Finding Dominators in a Flowgraph. *ACM Transactions on Programming Languages and Systems*, 1(1):121–141, July 1979.

[Mar93] Kim Marriott. Frameworks for abstract interpretation. *Acta Informatica*, 30(2):103–129, 1993. IBM Research Report RC18428 (March 1990).

[May92] D. E. Maydan. *Accurate Analysis of Array References.* Ph.d. thesis, Stanford University, Computer Systems Laboratory, September 1992.
`http://suif.stanford.edu/papers/maydan92c.ps`.

[MCL+93] T.J. Marlowe, Jong-Deok Choi, W.G. Landi, M.G. Burke, Barbara G. Ryder, and P. Carini. Pointer-Induced Aliasing: a Clarification. *ACM SIGPLAN Notices*, 28(9):67–70, September 1993.

[MHT96] A. Matsumoto, D.S. Han, and T. Tsuda. Alias Analysis of Pointers in Pascal and Fortran-90: Dependence analysis between pointer references. *Acta Informatica*, 33:99–130, 1996.

[MJ81a] S.S. Muchnick and Neil D. Jones. Flow Analysis and Optimisation of LISP-like Structures. In S.S. Muchnick and Neil D. Jones, editors, *Program Flow Analysis: Theory and Applications*, chapter 4, pages 102–131. Prentice-Hall, 1981.

[MJ81b] S.S. Muchnick and Neil D. Jones, editors. *Program Flow Analysis: Theory and Applications.* Prentice-Hall, 1981.

[MJ92] Carl McConnell and Ralph E. Johnson. Using static single assignment form in a code optimizer. *ACM Letters on Programming Languages and Systems*, 1(2):152–160, June 1992.

[MMRsZ] Stephen P. Masticola, Thomas J. Marlowe, and Barbara G. Ryder. Lattice Frameworks for Multisource and Bidirectional Data Flow Problems. *ACM Transactions on Programming Languages and Systems*, 17(5):777–803, September 1995
`ftp://athos.rutgers.edu/pub/technical-reports/lcsr-tr-241.ps.Z`.

[MN83] A. Mycroft and F. Nielson. Strong Abstract Interpretation using Power Domains. In J. Diaz, editor, *10th ICALP*, pages 536–547. LNCS # 154, 1983.

[Mor97] John David Morgenthaler. *Static Analysis for a Software Transformation Tool.* Ph.d. thesis, Department of Computer Science and Engineering, School of Engineering, University of California, San Diego, August 1997.
`http://www-cse.ucsd.edu/users/jdm/Papers/thesis.ps.gz`.

[MR79] E. Morel and C. Renvoise. Global Optimization by Suppression Partial Redundancies. *CACM*, 22(2):96–103, February 1979.

[MR90] T.J. Marlowe and Barbara G. Ryder. Properties of Data Flow Frameworks. *Acta Informatica*, 28:121–163, 1990.

[MS76] Zohar Manna and Adi Shamir. The Theoretical Aspects of the Optimal Fixed Point. *SIAM J.Comput.*, 5(3):414–426, September 1976.

[Muc97] S.S. Muchnick. *Compiler Design Implementation.* Morgan and Kaufmann, 1997.

[Mye81] E. W. Myers. A Precise Interprocedural Data Flow Algorithm. In *8th POPL*, pages 219–230. ACM, 1981.

[NNH99] Flemming Nielson, Hanne Riis Nielson, and Chris Hankin. *Principles of Program Analysis.* Springer-Verlag, 1999.
`http://www.daimi.au.dk/~hrn/PPA/ppa.html`.

[OO84] Karl J. Ottenstein and Linda M. Ottenstein. The Program Dependence Graph and its use in a Software Development Environment. *SIGPLAN Notices*, 19(5):177–184, May 1984.

[PBJ+90] Keshav Pingali, Micah Beck, Richard Johnson, Mayan Moudgill, and Paul Stodghill. Dependence Flow Graphs: an Algebraic Approach to Program Dependencies. In *18th POPL*, pages 67–78. ACM, 1990.

[PC90] Andy Podgurski and Lori A. Clarke. A formal model of program dependences and its implications for software testing, debugging, and maintenance. *IEEE Transactions on Software Engineering*, 16(9):965–979, September 1990.

[PC94] John Plevyak and A. Chien. Iterative Flow Analysis. In *Conf. Object-Oriented Programming Systems*, 1994.

[PS89] L.L. Pollock and Mary Lou Soffa. An Incremental Version of Iterative Data Flow Analysis. *IEEE-TSE*, 15(12):1537–1549, December 1989.

[Pug92] William Pugh. The Omega test: a fast and practical integer programming algorithm for dependence analysis. *Communications of the ACM*, 8:102–114, August 1992.

[PW93] William Pugh and David Wonnacott. An Exact Method for Analysis of Value-based Array Data Dependence. In *6th Intl. Workshop on Languages and Compilers for Parallel Computing*, pages 546–566, August 1993.

[PW94] William Pugh and David Wonnacott. Counting Solutions to Presburger Formulas: How and Why. Technical Report CS-TR-3234, University of Maryland Institute for Advanced Computer Studies, March 1994.
`ftp://ftp.cs.umd.edu/pub/papers/papers/3234/3234.ps.Z`.

[PW98] William Pugh and David Wonnacott. Constraint-based array dependence analysis. *ACM Transactions on Programming Languages and Systems*, 20(3):635–678, May 1998.
`http://triluminary.cs.haverford.edu/people/davew/array-analysis/WhatHowWhy.ps.gz`.

[QG97] Xiaolei Qian and Allen Goldberg. Bounds analysis by abstract interpretation. In *First Workshop on Automated Analysis of Programs*, 1997.
`ftp://ftp.kestrel.edu/pub/papers/goldberg/goldberg-AAS97.ps`.

[Ram97] G. Ramalingam. On sparse evaluation representations. In Pascal Van Hentenryck, editor, *Proceedings from the 4th International Static Analysis Symposium*, number 1302 in Lecture Notes in Computer Science, pages 1–15. Springer-Verlag, 1997.
`http://www.research.ibm.com/people/r/rama/Papers/ibmtr21245.ps`.

[Ram99] G. Ramalingam. Identifying loops in almost linear time. *ACM Transactions on Programming Languages and Systems*, 21(3):175–188, March 1999.

[RC87] Barbara G. Ryder and M.D. Carroll. Incremental Data Flow Analysis via Attributes. Technical Report LCSR-TR-93, Rutgers University, Dept. of Computer Science, New Jersey, June 1987.

[Rei79] John H. Reif. Data Flow Analysis of Communicating Processes. In *6th POPL*, pages 257–268. ACM, 1979.

[RHSR94] Thomas Reps, Susan Horwitz, Mooly Sagiv, and Geneviève Rosay. Speeding up Slicing. In *SIGSOFT'94*, pages 11–220. ACM, 1994.
`http://www.cs.wisc.edu/wpis/papers/fse94.ps`.

[RK92] Uday S. Reddy and Samuel N. Kamin. On the power of abstract interpretation. In *IEEE, International Conference on Computer Languages*, pages 24–33, 1992.

[RL77] John H. Reif and Harry R. Lewis. Symbolic Evaluation and the Global Value Graph. In *4th POPL*, pages 104–118. ACM, 1977.

[RL86] John H. Reif and Harry R. Lewis. Efficient Symbolic Analysis of Programs. *Journal of Computer and System Sciences*, 32(3):280–314, June 1986.

[RLP90] Barbara G. Ryder, William Landi, and Marvin C. Paull. Profiling an Incremental Data Flow Analysis Algorithm. *IEEE-TSE*, 16(6):129–140, February 1990.

[Ros77a] Barry K. Rosen. Applications of High-Level Control Flow. In *4th POPL*, pages 38–47, Los Angeles, Calif., 1977.

[Ros77b] Barry K. Rosen. High-Level Data Flow Frameworks. *CACM*, 20(10):712–724, October 1977.

[Ros78] Barry K. Rosen. Monoids for Rapid Data Flow analysis. In *5th POPL*, pages 47–59, Tucson, 1978.

[Ros82] Barry K. Rosen. A lubricant for Data Flow Analysis. *SIAM J. Comput.*, 11(3), August 1982.

[RP86] Barbara G. Ryder and Marvin C. Paull. Elimination Algorithms for Data Flow Analysis. *ACM Computing Surveys*, 18(3):277–316, September 1986.

[RP88] Barbara G. Ryder and Marvin C. Paull. Incremental Data Flow Analysis Algorithms. *ACM Transactions on Programming Languages and Systems*, 10(1):1–50, January 1988.

[RR95] Thomas Reps and Geneviève Rosay. Precise Interprocedural Slicing. *Software Engineering Notes*, 20(4):41–52, October 1995.

[RT81] John H. Reif and Robert Endre Tarjan. Symbolic Program Analysis in Almost Linear Time. *SIAM J. Comput.*, 11(1):81–93, February 1981.

[RWZ88] Barry K. Rosen, Mark N. Wegman, and F.K. Zadeck. Global Value Numbering and Redundant Computations. In *15th POPL*, pages 12–27. ACM, 1988.

[Ryd79] Barbara G. Ryder. A survey of Data Flow Analysis Techniques. Technical Report DCS-TR-85, Rutgers University, Dept. of Computer Science, New Jersey, December 1979.

[SAS98] Shamik Sharma, Anurag Acharya, and Joel Saltz. Deferred data-flow analysis. Technical Report TRCS98-38, University of California, Computer Science, Santa Barbara, December 1998. `ftp://ftp.cs.ucsb.edu/pub/techreports/TRCS98-38.ps`.

[SEFR89] S. Sagiv, O. Edelstein, N. Francez, and M. Rodeh. Resolving Circularity in Attribute Grammars with Application to Data Flow Analysis. In *6th POPL*, pages 36–48, 1989.

[SF94] A. Sabry and M. Felleisen. Is Continuation-Passing Style Useful for Data Flow Analysis? In *PLDI*, pages 1–12. ACM, 1994. `http://www.cs.uoregon.edu/~sabry/home.html`.

[SG95] Vugranam C. Sreedhar and Guang R. Gao. A linear time algorithm for placing ϕ-nodes. In *Proceedings of the 22nd Symposium on Principles of Programming Languages (POPL'95)*, pages 62–73. ACM, January 1995. `ftp://ftp-acaps.cs.mcgill.ca/pub/doc/memos/memo75.ps.gz`.

[SGL93] Vugranam C. Sreedhar, Guang R. Gao, and Yong-Fong Lee. Efficient data flow ananlysis using DJ graphs: Elimination methods revisited. ACAPS Technical Memo 93, School of Computer Science, McGill University, Montréal, June 1993. `ftp://ftp-acaps.cs.mcgill.ca/pub/doc/memos/memo93.ps.gz`.

[SGL94] Vugranam C. Sreedhar, Guang R. Gao, and Yong-Fong Lee. An efficient incremental algorithm for maintaining dominator trees and its application to ϕ-nodes updates. ACAPS Technical Memo 77, School of Computer Science, McGill University, Montréal, July 1994. `ftp://ftp-acaps.cs.mcgill.ca/pub/doc/memos/memo77.ps.gz`.

[SGL96] Vugranam C. Sreedhar, Guang R. Gao, and Yong-Fong Lee. Identifying Loops Using DJ Graphs. *TOPLAS*, 18(6):649–658, November 1996. ACAPS Technical Memo 98 `ftp://ftp-acaps.cs.mcgill.ca/pub/doc/memos/memo98.ps.gz`.

[SH97] Marc Shapiro and Susan Horwitz. Fast and accurate flow-insensitive points-to analysis. In *24th POPL*. ACM, 1997.
`http://www.cs.wisc.edu/wpis/papers/popl97.ps`.

[Sha80] Micha Sharir. Structural Analysis: a New Approach to Flow Analysis in Optimizing Compilers. *Computer Languages*, 5(3-4):141–153, 1980.

[SMR90] R.C. Sekar, P. Misra, and I.V. Ramakrishnan. On the Power and Limitation of Strictness based on Abstract Interpretation. In *18th POPL*, pages 37–48. ACM, 1990.

[Sre95] Vugranam C. Sreedhar. *Efficient Program Analysis using DJ Graphs*. Phd, School of Computer Science, McGill University, Montréal, September 1995.
`http://www-acaps.cs.mcgill.ca/~sreedhar/`.

[SRW96] Mooly Sagiv, Thomas Reps, and Reinhard Wilhelm. Solving Shape-Analysis Problems in Languages with Destructive Updating. In *23rd POPL*. ACM, 1996.
`http://www.cs.uchicago.edu/~sagiv/courses/shape.dvi`.

[SRW98] Mooly Sagiv, Thomas Reps, and Reinhard Wilhelm. Solving Shape-Analysis Problems in Languages with Destructive Updating. *ACM Transactions on Programming Languages and Systems*, 20(2):1–50, March 1998.
`http://www.cs.wisc.edu/wpis/papers/toplas98a.ps`.

[SS96] Jonas Skeppstedt and Per Stenström. Using Data Flow Analysis to Reduce Ownership Overhead in Cache Coherence Protocols. *TOPLAS*, 18(6):659–682, November 1996.

[Ste91] Bernhard Steffen. Generating data flow analysis algorithms from modal specifications. Technical Report 91-31, RWTH Aachen, Fachgruppe Informatik, 1991.
`ftp://ftp.uni-passau.de/pub/local/parallel/papers/dfams-scp.ps.Z`.

[Ste95] Bjarne Steensgaard. Sparse functional stores for imperative programs. Technical Report MSR-TR-95-07, Microsoft Research, March 1995.
`ftp://ftp.research.microsoft.com/pub/Tech-Reports/Winter94-95/TR-95-07.PS`.

[Ste96a] B. Steensgaard. Points-to analysis by type inference of programs with structures and unions. In *Proceedings of the 1996 International Conference on Compiler Construction*, pages 136–150. Springer-Verlag, April 1996.
`ftp://ftp.research.microsoft.com/users/rusa/cc96.ps`.

[Ste96b] B. Steensgaard. Points-to analysis in almost linear time. In *Proceedings of the Twentythird Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 32–41. ACM, January 1996.
`ftp://ftp.research.microsoft.com/users/rusa/popl96.ps`.

[Sto95] Eric James Stoltz. *Intermediate Compiler Analysis via Reference Chaining*. Ph.d. thesis, Oregon Graduate Institute of Science and Technology, Computer Science Dept., January 1995. CS/E 95-TH-001
`ftp://cse.ogi.edu/pub/tech-reports/1995/95-TH-001.ps.gz`.

[Str90] Jan Stransky. A lattice for abstract interpretation of dynamic lisp-like structures. Rapport de Recherche 90-03, Ecole Polytechnique, Laboratoire d'Informatique, June 1990.

[Tar74] Robert Endre Tarjan. Finding Dominators in Directed Graphs. *SIAM J. Comput.*, 3(1):62–89, March 1974.

[Tar81a] Robert Endre Tarjan. Fast Algorithms for Solving Path Problems. *JACM*, 28(3):594–614, July 1981.

[Tar81b] Robert Endre Tarjan. A Unified Approach of Path Problems. *JACM*, 28(3):577–593, July 1981.

[TH92] S. Tjiang and J. Hennessy. Sharlit: A Tool for Building Optimizers. *SIGPLAN Notices*, 27(7):82–93, July 1992. (PLDI'92)
`http://suif.stanford.edu/papers/tjiang92.ps`.

[Tip95] Frank Tip. A Survey of Program Slicing Techniques. *Journal of Programming Languages*, 3(3), 1995.
`ftp://ftp.cwi.nl/pub/gipe/reports/Tip94.ps.Z`.

[TP95] Peng Tu and David Padua. Efficient Building and Placing of Gating Functions. *SIGPLAN Notices*, 30(6):47–66, June 1995. (PLDI'95)
`http://polaris.cs.uiuc.edu/reports/1389.ps.gz`.

[Ull73] J. D. Ullman. Fast algorithms for the elimination of common subexpressions. *Acta Informatica*, 2(191-213), 1973.

[Vol94] Jürgen Vollmer. Dataflow Equations for Parallel Programs that Share Memory. In *POPL*, 1994.

[Wad88] P. Wadler. Strictness Analysis Aids Time Analysis. In *15th POPL*, pages 119–132. ACM, 1988.

[WCES94] D. Weise, R.F. Crew, M. Ernst, and B. Steensgaard. Value dependence graphs: Representations without taxation. In *21st POPL*. ACM, 1994.
`ftp://ftp.research.microsoft.com/pub/analysts/vdg.ps`.

[Weg75] B. Wegbreit. Property Extraction in Well-Founded Property Sets. *IEEE, Transactions on Software Engineering*, SE-1(3):270–285, 1975.

[Weg83] Mark N. Wegman. Summarizing Graphs by Regular Expressions. In *10th POPL*, pages 203–216, Austin, Texas, January 1983.

[Weg85] Mark N. Wegman. Constant Propagation with Conditional Branches. In *12th POPL*, pages 203–216, Austin, Texas, January 1985.

[Wei80] W.E. Weihl. Interprocedural Data Flow Analysis in the presence of Pointers, procedure variables, and label variables. In *17th POPL*, pages 83–94. ACM, 1980.

[Wei84] Mark Weiser. Program Slicing. *IEEE Transactions on Software Engineering*, SE-10(4):352–356, July 1984.

[Wil97] Robert P. Wilson. *Efficient Context-Sensitive Pointer Analysis for C Programs*. Ph.d. thesis, Stanford University, Computer Systems Laboratory, December 1997.
`http://suif.stanford.edu/papers/wilson97.ps`.

[Wis94] R. Wismüller. Debugging of Globally Optimized Programs using Data Flow Analysis. In *PLDI*, pages 278–289. ACM, 1994.

[WL95] Robert P. Wilson and Monica S. Lam. Efficient Context-Sensitive Pointer Analysis for C Programs. In *Programming Language Design and Implementation (PLDI)*, pages 1–12, 1995. Paru dans SIGPLAN notices, vol 30, n0 6.
`http://suif.stanford.edu/~bwilson/papers/pldi95/paper.html`
`http://suif.stanford.edu/papers/wilson95.ps`.

[WM94] Reinhard Wilhelm and Dieter Maurer. *Les compilateurs : théorie, construction, génération*. Masson, 1994.

[Wol91] Michael Wolfe. Flow graph anomalies: What's in a loop? Research Report CS/E 92-012, OGI Department of Computer Science and Engineering, Oregon Graduate Institute of Science and Technology, February 1991.
`ftp://cse.ogi.edu/pub/tech-reports/1992/92-012.ps.gz`.

[Wol96] Michael Wolfe. *High Performance Compilers for Parallel Computing*. Addison-Wesley, 1996.

[Won99] David Wonnacott. Constant propagation through array variables. Technical Report 386, Rutgers University, June 1999.
`ftp://www.cs.rutgers.edu/pub/technical-reports/dcs-tr-386.ps.Z`.

[WT92] Michael Wolfe and Chau-Wen Tseng. The power test for data dependence. *IEEE Transactions on Parallel and Distributed Systems*, 3(5):591–601, September 1992.

[WZ91] Mark N. Wegman and F.K. Zadeck. Constant Propagation with Conditional Branches. *ACM Transactions on Programming Languages and Systems*, 13(2):181–210, April 1991.

[Zad84] F.K. Zadeck. Incremental Data Flow Analysis in a Structured Program Editor. *SIGPLAN Notices*, 16(3):132–143, June 1984. Proc. of ACM-SIGPLAN'84 Symposium on Compiler Construction.

[Zha98] Xiang-Xiang Sean Zhang. *Practical Pointer Aliasing Analyses for C.* Ph.d. thesis, Rutgers University, August 1998.
`ftp://athos.rutgers.edu/pub/technical-reports/dcs-tr-367.ps.Z`.

[ZRL96] Sean Zhang, Barbara G. Ryder, and William Landi. Program Decomposition for Pointer Analysis: a Step toward Practical Analyses. *Software Engineering Notes*, 21(6):81–92, November 1996. SIGSOFT'96: 4th ACM-SIGSOFT Symposium on the Foundations of Software Programming,
`ftp://athos.rutgers.edu/pub/technical-reports/lcsr-tr-259.ps.Z`
`http://www.prolangs.rutgers.edu/refs/docs/fse96.ps`.