

References

- [ACK81] F. E. Allen, J. Cocke, and K. W. Kennedy. Reduction of operator strength. In Muchnick and Jones [MJ81], chapter 3, pages 79 – 101.
- [AH82] M. Auslander and M. Hopkins. An overview of the PL.8 compiler. In *Proc. ACM SIGPLAN Symposium on Compiler Construction'82*, volume 17,6 of *ACM SIGPLAN Notices*, pages 22 – 31, June 1982.
- [AHU83] A. V. Aho, J. E. Hopcroft, and J. D. Ullman. *Data Structures and Algorithms*. Addison-Wesley, 1983.
- [AJU77] A. V. Aho, S. C. Johnson, and J. D. Ullman. Code generation for expressions with common subexpressions. *Journal of the ACM*, 24(1):146 – 160, 1977.
- [ASU85] A. V. Aho, R. Sethi, and J. D. Ullman. *Compilers: Principles, Techniques and Tools*. Addison-Wesley, 1985.
- [AU75] A. V. Aho and J. D. Ullman. Node listings for reducible flow graphs. In *Proc. 7th ACM Symposium on the Theory of Computing*, pages 177 – 185, Albuquerque, NM, 1975.
- [BC94] P. Briggs and K. D. Cooper. Effective partial redundancy elimination. In *Proc. ACM SIGPLAN Conference on Programming Language Design and Implementation'94*, 1994.
- [Ber57] C. Berge. Two theorems in graph theory. *Proc. Nat. Acad. Sci. USA*, pages 842–844, 1957.
- [Bes87] E. Best, editor. *Theory of Petri Nets: the Free Choice Hiatus*, volume 254 of *Lecture Notes in Computer Science*. Springer-Verlag, 1987.
- [BG97] R. Bodík and R. Gupta. Partial dead code elimination using slicing transformations. In *Proc. ACM SIGPLAN Conference on Programming Language Design and Implementation'97*, volume 32, 5 of *ACM SIGPLAN Notices*, pages 159–170, June 1997.
- [BGS98] R. Bodík, R. Gupta, and M. L. Soffa. Complete removal of redundant expressions. In *Proc. ACM SIGPLAN Conference on Programming Language Design and Implementation'98*, pages 1–14, Montreal, Quebec, Canada, June 1998.
- [BR91] D. Bernstein and M. Rodeh. Global instruction scheduling for superscalar machines. In *Proc. ACM SIGPLAN Conference on Programming Language Design and Implementation'91*, volume 26,6 of *ACM SIGPLAN Notices*, pages 241–255, Toronto, Ontario, June 1991.
- [Bri92a] P. Briggs. Private communication with Preston Briggs. 1992.
- [Bri92b] P. Briggs. *Register Allocation via Graph Coloring*. PhD thesis, Rice University, Houston, Texas, 1992.
- [CAC⁺81] G. J. Chaitin, M. A. Auslander, A. K. Chandra, J. Cocke, M. E. Hopkins, and P. W. Markstein. Register allocation via coloring. *Journal of Computer Languages*, 6:47 – 57, 1981.

- [CC77] P. Cousot and R. Cousot. Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Conf. Record of the 4th ACM Symposium on the Principles of Programming Languages*, pages 238 – 252, Los Angeles, CA, 1977.
- [CC95] C. Click and K. D. Cooper. Combining analyses, combining optimizations. *ACM Transactions on Programming Languages and Systems*, 17(2):181 – 196, 1995.
- [CH90] F. C. Chow and J. L. Hennessy. The priority-based coloring approach to register allocation. *ACM Transactions on Programming Languages and Systems*, 12(4):501–536, 1990.
- [Cha82] G. J. Chaitin. Register allocation and spilling via graph coloring. *ACM SIGPLAN Notices*, 17(6):98–105, 1982.
- [Cho83] F. Chow. *A portable machine independent optimizer – Design and measurements*. PhD thesis, Stanford University, Dept. of Electrical Engineering, Stanford, CA, 1983. Published as Tech. Rep. 83-254, Computer Systems Lab., Stanford University.
- [CK77] J. Cocke and K. W. Kennedy. An algorithm for reduction of operator strength. *Communications of the ACM*, 20(11):850 – 856, 1977.
- [Cli95] C. Click. *Combining Analyses, Combining Optimizations*. PhD thesis, Rice University, Houston, Texas, 1995. 149 pages.
- [CLZ86] R. Cytron, A. Lowry, and F. K. Zadeck. Code motion of control structures in high-level languages. In *Conf. Record of the 13th ACM Symposium on the Principles of Programming Languages*, January 1986.
- [CP91] J. Cai and R. Paige. Look ma, no hashing, and no arrays neither. In *Conf. Record of the 18th ACM Symposium on the Principles of Programming Languages*, pages 143 – 154, Orlando, FL, 1991.
- [DGG⁺96] A. Dold, Th. Gaul, W. Goerigk, G. Goos, F. W. Heberle, F. W. von Henke, U. Hoffmann, H. Langmaack, H. Pfeifer, H. Rueß, and W. Zimmermann. Compiler correctness and implementation verification: The *verifix* approach. In *Proc. 6th Conference on Compiler Construction (CC)*, Lecture Notes in Computer Science 1060, pages 106 – 120, Linköping, Sweden, 1996. Springer-Verlag.
- [Dha83] D. M. Dhamdhere. Characterization of program loops in code optimization. *Journal of Computer Languages*, 8(2):69 – 76, 1983.
- [Dha88] D. M. Dhamdhere. A fast algorithm for code movement optimization. *ACM SIGPLAN Notices*, 23(10):172 – 180, 1988.
- [Dha89a] D. M. Dhamdhere. Corrigendum: A new algorithm for composite hoisting and strength reduction optimisation. *International Journal of Computer Mathematics*, 27:31 – 32, 1989.
- [Dha89b] D. M. Dhamdhere. A new algorithm for composite hoisting and strength reduction optimisation (+ Corrigendum). *International Journal of Computer Mathematics*, 27:1 – 14 (+ 31 – 32), 1989.
- [Dha91] D. M. Dhamdhere. Practical adaptation of the global optimization algorithm of Morel and Renvoise. *ACM Transactions on Programming Languages and Systems*, 13(2):291 – 294, 1991. Technical Correspondence.
- [DK93] D. M. Dhamdhere and U. P. Khedker. Complexity of bidirectional data flow analysis. In *Conf. Record of the 20th ACM Symposium on the Principles of Programming Languages*, pages 397–409, Charleston, SC, January 1993.
- [DP93] D. M. Dhamdhere and H. Patil. An elimination algorithm for bidirectional data flow problems using edge placement. *ACM Transactions on Programming Languages and Systems*, 15(2):312 – 336, April 1993.
- [DRZ92] D. M. Dhamdhere, B. K. Rosen, and F. K. Zadeck. How to analyze large programs efficiently and informatively. In *Proc. ACM SIGPLAN Conference*

- on *Programming Language Design and Implementation'92*, volume 27,7 of *ACM SIGPLAN Notices*, pages 212 – 223, San Francisco, CA, June 1992.
- [DS88] K.-H. Drechsler and M. P. Stadel. A solution to a problem with Morel and Renvoise's "Global optimization by suppression of partial redundancies". *ACM Transactions on Programming Languages and Systems*, 10(4):635 – 640, 1988. Technical Correspondence.
- [DS93] K.-H. Drechsler and M. P. Stadel. A variation of Knoop, Rüthing and Steffen's lazy code motion. *ACM SIGPLAN Notices*, 28(5):29 – 38, 1993.
- [FKCX94] L. Feigen, D. Klappholz, R. Casazza, and X. Xue. The revival transformation. In *Conf. Record of the 21nd ACM Symposium on the Principles of Programming Languages*, Portland, Oregon, 1994.
- [FOW87] 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*, pages 319–349, July 1987.
- [GJ79] M. R. Garey and D. S. Johnson. *Computers and Intractability – A Guide to the Theory of NP-Completeness*. W.H. Freeman & Co, San Francisco, 1979.
- [GKL⁺96] A. Geser, J. Knoop, G. Lüttgen, O. Rüthing, and B. Steffen. Non-monotone fixpoint iterations to resolve second order effects. In *Proc. 6th Conference on Compiler Construction (CC)*, Lecture Notes in Computer Science 1060, pages 106 – 120, Linköping, Sweden, 1996. Springer-Verlag.
- [GM86] P. B. Gibbons and S. S. Muchnik. Efficient instruction scheduling for a pipeline architecture. In *Proc. ACM SIGPLAN Symposium on Compiler Construction'86*, volume 21, 7 of *ACM SIGPLAN Notices*, pages 11–16, June 1986.
- [GMW81] R. Giegerich, U. Möncke, and R. Wilhelm. Invariance of approximative semantics with respect to program transformations. In *Proc. of the third Conference of the European Co-operation in Informatics*, Informatik-Fachberichte 50, pages 1–10. Springer, 1981.
- [Gro95] V. Grover. Private communication with Vinod Grover, sun inc. 1995.
- [Hal35] M. Jr. Hall. On representatives of subsets. *Journal of the London Mathematical Society*, 10:26–30, 1935.
- [Har77] A. W. Harrison. Compiler analysis of the value ranges for variables. *IEEE Transactions on Software Engineering*, SE-3(3), may 1977.
- [HDT87] S. Horwitz, A. Demers, and T. Teitelbaum. An efficient general iterative algorithm for data flow analysis. *Acta Informatica*, 24:679 – 694, 1987.
- [Hec77] M. S. Hecht. *Flow Analysis of Computer Programs*. Elsevier, North-Holland, 1977.
- [HK73] J. E. Hopcroft and R. M. Karp. An $n^{\frac{5}{2}}$ algorithm for maximum matchings in bipartite graphs. *SIAM Journal on Computing*, 2(4):225–231, 1973.
- [JD82a] S. M. Joshi and D. M. Dhamdhere. A composite hoisting-strength reduction transformation for global program optimization – part I. *International Journal of Computer Mathematics*, 11:21 – 41, 1982.
- [JD82b] S. M. Joshi and D. M. Dhamdhere. A composite hoisting-strength reduction transformation for global program optimization – part II. *International Journal of Computer Mathematics*, 11:111 – 126, 1982.
- [JP93] R. Johnson and K. Pingali. Dependency based program analysis. In *Proc. ACM SIGPLAN Conference on Programming Language Design and Implementation'94*, pages 78 – 89, Albuquerque, NM, 1993.
- [KD94] U. P. Khedker and D. M. Dhamdhere. A generalized theory of bit vector data flow analysis. *ACM Transactions on Programming Languages and Systems*, 16(5):1472 – 1511, September 1994.
- [Ken75] K. W. Kennedy. Node listings applied to data flow analysis. In *Conf. Record of the 2nd ACM Symposium on the Principles of Programming Languages*, pages 10 – 21, Palo Alto, CA, 1975.

- [Ken81] K. W. Kennedy. A survey of data flow analysis techniques. In Muchnick and Jones [MJ81], chapter 1, pages 5 – 54.
- [Kno93] J. Knoop. *Optimal Interprocedural Program Optimization: A new framework and its application*. PhD thesis, Institut für Informatik und Praktische Mathematik, Christian-Albrechts-Universität Kiel, Germany, 1993. To appear as monograph in the series *Lecture Notes in Computer Science*, Springer-Verlag, Heidelberg, Germany, 1997.
- [Kou77] L. T. Kou. On live-dead analysis for global data flow problems. *Journal of the ACM*, 24(3):473 – 483, July 1977.
- [KRS92] J. Knoop, O. Rüthing, and B. Steffen. Lazy code motion. In *Proc. ACM SIGPLAN Conference on Programming Language Design and Implementation'92*, volume 27,7 of *ACM SIGPLAN Notices*, pages 224 – 234, San Francisco, CA, June 1992.
- [KRS93] J. Knoop, O. Rüthing, and B. Steffen. Lazy strength reduction. *Journal of Programming Languages*, 1(1):71–91, 1993.
- [KRS94a] J. Knoop, O. Rüthing, and B. Steffen. Optimal code motion: Theory and practice. *ACM Transactions on Programming Languages and Systems*, 16(4):1117–1155, 1994.
- [KRS94b] J. Knoop, O. Rüthing, and B. Steffen. Partial dead code elimination. In *Proc. ACM SIGPLAN Conference on Programming Language Design and Implementation'94*, volume 29,6 of *ACM SIGPLAN Notices*, pages 147–158, Orlando, FL, June 1994.
- [KRS95] J. Knoop, O. Rüthing, and B. Steffen. The power of assignment motion. In *Proc. ACM SIGPLAN Conference on Programming Language Design and Implementation'95*, volume 30,6 of *ACM SIGPLAN Notices*, pages 233–245, La Jolla, CA, June 1995.
- [KRS96a] J. Knoop, O. Rüthing, and B. Steffen. Syntactic versus semantic code motion: Analogies and essential differences. Technical Report 9616, Institut für Informatik und Praktische Mathematik, Christian-Albrechts-Universität Kiel, Germany, 1996. 18 pages.
- [KRS96b] J. Knoop, O. Rüthing, and B. Steffen. Towards a tool kit for the automatic generation of interprocedural data flow analyses. *Journal of Programming Languages*, 4:211–246, 1996.
- [KRS98] J. Knoop, O. Rüthing, and B. Steffen. Code motion and code placement: Just synonyms? In *Proc. 6th European Symposium on Programming (ESOP)*, Lecture Notes in Computer Science 1381, pages 154 – 196, Lisbon, Portugal, 1998. Springer-Verlag.
- [KS92] J. Knoop and B. Steffen. Optimal interprocedural partial redundancy elimination. Extended abstract. In *Addenda to Proc. 4th Conference on Compiler Construction (CC)*, pages 36 – 39, Paderborn, Germany, 1992. Published as Tech. Rep. No. 103, Department of Computer Science, University of Paderborn.
- [KU76] J. B. Kam and J. D. Ullman. Global data flow analysis and iterative algorithms. *Journal of the ACM*, 23(1):158 – 171, 1976.
- [KU77] J. B. Kam and J. D. Ullman. Monotone data flow analysis frameworks. *Acta Informatica*, 7:309 – 317, 1977.
- [LP86] L. Lovász and M. D. Plummer. *Matching Theory*, volume 29 of *Annals of Discrete Mathematics*. North Holland, 1986.
- [MJ81] S. S. Muchnick and N. D. Jones, editors. *Program Flow Analysis: Theory and Applications*. Prentice Hall, Englewood Cliffs, NJ, 1981.
- [MMR95] P. M. Masticola, T. J. Marlowe, and B. G. Ryder. Lattice frameworks for multisource and bidirectional data flow problems. *ACM Transactions on Programming Languages and Systems*, 17(5):777 – 802, 1995.

- [Mor84] E. Morel. Data flow analysis and global optimization. In B. Lorho, editor, *Methods and tools for compiler construction*. Cambridge University Press, 1984.
- [MR79] E. Morel and C. Renvoise. Global optimization by suppression of partial redundancies. *Communications of the ACM*, 22(2):96 – 103, 1979.
- [MR81] E. Morel and C. Renvoise. Interprocedural elimination of partial redundancies. In Muchnick and Jones [MJ81], chapter 6, pages 160 – 188.
- [Muc97] S. S. Muchnick, editor. *Advanced Compiler Design & Implementation*. Morgan Kaufmann, San Francisco, CA, 1997.
- [New42] M. H. A. Newman. On theories with a combinatorial definition of equivalence. *Annals of Math.*, 43,2:223–243, 1942.
- [Nie86] F. Nielson. A bibliography on abstract interpretation. *ACM SIGPLAN Notices*, 21:31 – 38, 1986.
- [Rei85] W. Reisig. *Petri Nets: An Introduction*. EATCS Monographs on Theoretical Computer Science. Springer-Verlag, 1985.
- [RL77] J. H. Reif and R. Lewis. Symbolic evaluation and the global value graph. In *Conf. Record of the 4th ACM Symposium on the Principles of Programming Languages*, pages 104 – 118, Los Angeles, CA, 1977.
- [RP88] B.G. Ryder and M.C. Paull. Incremental data-flow analysis algorithms. *ACM Transactions on Programming Languages and Systems*, 10(1):1–50, January 1988.
- [Rüt98a] O. Rüthing. Bidirectional data flow analysis in code motion: Myth and reality. In *Proc. Int. Static Analysis Symposium (SAS'98)*, Lecture Notes in Computer Science, Pisa, Italy, September 1998. Springer-Verlag. To appear.
- [Rüt98b] O. Rüthing. Optimal code motion in the presence of large expressions. In *Proc. International Conference on Computer Languages*, pages 216–225, Chicago, IL, May 1998. IEEE.
- [RWZ88] B. K. Rosen, M. N. Wegman, and F. K. Zadeck. Global value numbers and redundant computations. In *Conf. Record of the 15th ACM Symposium on the Principles of Programming Languages*, pages 12 – 27, San Diego, CA, 1988.
- [SKR90] B. Steffen, J. Knoop, and O. Rüthing. The value flow graph: A program representation for optimal program transformations. In *Proc. 3rd European Symposium on Programming (ESOP)*, Lecture Notes in Computer Science 432, pages 389 – 405, Copenhagen, Denmark, 1990. Springer-Verlag.
- [SKR91] B. Steffen, J. Knoop, and O. Rüthing. Efficient code motion and an adaptation to strength reduction. In *Proc. 4th International Joint Conference on the Theory and Practice of Software Development (TAPSOFT)*, Lecture Notes in Computer Science 494, pages 394 – 415, Brighton, UK, 1991. Springer-Verlag.
- [Sor89] A. Sorkin. Some comments on a solution to a problem with Morel and Renvoise's "Global optimization by suppression of partial redundancies". *ACM Transactions on Programming Languages and Systems*, 11(4):666 – 668, 1989. Technical Correspondence.
- [Ste91] B. Steffen. Data flow analysis as model checking. In *Proc. TACS*, Lecture Notes in Computer Science 526, pages 346 – 364, Sendai, Japan, 1991. Springer-Verlag.
- [Ste93] B. Steffen. Generating data flow analysis algorithms from modal specifications. *International Journal on Science of Computer Programming*, 21:115–139, 1993.
- [Ste96] B. Steffen. Property oriented expansion. In *SAS/ALP/PLILP'96*, volume 1145, pages 22–41, Aachen (D), September 1996. Springer Verlag. Proc. Int. Static Analysis Symposium (SAS'96),.
- [Tar79] R. E. Tarjan. Applications of path compression on balanced trees. *Journal of the ACM*, 26(4):690 – 715, 1979.

- [Tar81a] R. E. Tarjan. Fast algorithms for solving path problems. *Journal of the ACM*, 28(3):594 – 614, 1981.
- [Tar81b] R. E. Tarjan. A unified approach to path problems. *Journal of the ACM*, 28(3):577 – 593, 1981.
- [W.78] Kennedy. K. W. Use-definition chains with applications. *Journal of Computer Languages*, 3, 1978.
- [WS90] D. L. Whitfield and M. L. Soffa. An approach to ordering optimizing transformations. In *Proceedings of the Second ACM SIGPLAN Symposium on Principles & Practice of Parallel Programming (PPOPP)*, volume 25,3 of *ACM SIGPLAN Notices*, pages 137 – 147, Seattle, Washington, March 1990.
- [WS97] D. L. Whitfield and M. L. Soffa. An approach for exploring code improving transformations. *ACM Transactions on Programming Languages and Systems*, 19(6):1053–1084, November 1997.
- [WZ85] M. Wegman and K. Zadeck. Constant propagation with conditional branches. In *Conf. Record of the 12th ACM Symposium on the Principles of Programming Languages*, pages 291 – 299, January 1985.
- [WZ91] M. N. Wegman and F. K. Zadeck. Constant propagation with conditional branches. *ACM Transactions on Programming Languages and Systems*, 13(2), April 1991.

Index

- assignment motion
 - enhanced, 204
 - maximal, 175
- basic block, 12
- bit-vector analysis
 - elimination technique, 112
 - iterative, 112
 - node listing, 112
 - path compression technique, 112
- chain of blockades, 177
- code pattern, 13
- complexity, 174
 - of BEM_φ , 35
 - of BEM_ϕ , 110
 - of $LFEM_\phi$, 110
 - of $FFEM_\phi$, 111
 - of LEM_φ , 35
 - of LEM_ϕ , 110
- computationally better, 24
- confluence, 159
 - local, 161, 168
 - modular, 161
- correctness
 - of replacements, 23
 - partial, 156
 - total, 156
- DAG, 10, 198
 - expression, 88, **91**
 - labelled expression, 91
- deficiency, 51
- deficiency set, 51
- definition–use chain, 188
- edge splitting, 14
- EE-effect, 180
- elimination
 - maximal, 175
 - of dead assignment, 143
 - of faint assignment, 148
 - of redundant assignments, 149
- elimination-elimination effect
 - weak, 184
- equivalence of programs, 168
- expression, 10
 - level of, 39
- expression motion, 22
 - admissible, 22, **23**
 - busy, 24, 25
 - computationally optimal, **24**, **41**
 - flat, 38
 - fully flexible, 96
 - lazy, 26, 28
 - levelwise structured, 60
 - of a single expression, 22
 - profitable, 116
 - structured, 40
 - busy, 43
 - lazy, 46
- flow graph, 12
 - depth of, 112
 - width of, 131
- graph, 9
 - bipartite, **9**, 62
 - induced, 93
 - matching, 51
 - directed, 10
 - acyclic,
 - see DAG10
 - undirected, 9
- initialisation candidate, 88
 - relevant, 92
- initialisation candidate, 198
- iteration
 - slot-wise, 174
- level, 39

- lifetime range, 26, 27
 - structured, 47
- marriage problem, 52
- matching, **51**
 - complete, 51
 - maximal, 51
- motion-elimination couple, 155
 - \mathcal{M} -adequate, 202
 - strongly \mathcal{M} -adequate, 202
 - uniform, 158
 - consistent, 159
 - weakly consistent, 161
- neighbours
 - of a vertex, 9
- nodes
 - of a flow graph, 12
- occurrence
 - origin, 158
 - sink, 157
 - source, 157
- optimal
 - lifetime, 49
 - structured, 49
- path
 - alternating, 52
 - augmenting, 53
 - finite, 10
 - insertion-replacement, 26
 - insertion-use, 47
- penalty costs, 174
- Petri-net, 210
 - free choice, 210
- predecessor
 - of a vertex, 10
 - zig-zag, 123
- predicate
 - local, 22
- preorder, 15, 170
 - computationally better, 24
 - lifetime better, 27
- program point, 12
- register
 - expression, 88
 - relevant, 92
 - pressure, 26
 - symbolic, 22
- register allocation, 51
- release candidate, 88, 198
- safety
 - down-, 23
 - homogeneous, 118
 - of insertions, 23
 - up-, 23
- second order effect, **144**
- splitting transformation, 190
- statement, 11
 - assignment, 11
 - immobile, 11
 - irrelevant, 12
 - output, 11
- strength reduction, 60
- subexpression
 - immediate, 10
 - mediate, 11
- subset
 - irreducible, 58, 75
- successor
 - of a vertex, 10
 - zig-zag, 123
- superexpression, 11
- temporary variable, 22
- tight set, 51, 53
- tightness
 - defect, 58
- trade-off candidates
 - lower, 50, **61**
 - upper, 50, **61**
- trade-off pair, 92
 - gain of, 92
 - optimal, 92
- universe
 - of an MEC, 170

universe of expressions

flat, 38

structured, 39

variable

temporary, 22

width, 131

Index of Symbols

$\mathbf{U}_{\mathcal{M}}(\cdot)$, 170

$B_{\dot{n}}$, 93

$D_{\dot{n}}$, 91

$\mathcal{AP}(\cdot)$, 13

$\mathcal{EP}(\cdot)$, 13

Φ_{f1} , 38

Φ , 39

$\Phi^{<i}$, 39

$\Phi^{<i}$, 39

Φ^i , 39

\mathcal{E} , 10

$MaxSubExpr(\psi)$, 39

$Occ_{\alpha,p}(\cdot)$, 13

$Occ_{\alpha}(\cdot)$, 13

\dot{N} , 12

$\Theta_{\dot{n}}^{dn}$, 93

$\Theta_{\dot{n}}^{up}$, 93

$\Theta_{\dot{n}}^{dn(i)}$, 61

$SubExpr^*(\cdot)$, 11

$SubExpr(\cdot)$, 11

$\Theta_{\dot{n}}^{up(i)}$, 61

Δ^{block} , 178

Δ_n^{block} , 178

$\bar{\Delta}^{block}$, 183

$\bar{\Delta}_n^{block}$, 183

$defic(\cdot)$, 51

d , 112

$\langle \cdot \rangle$, 10

$l.$, 91

$Lev_{\Phi}(\cdot)$, 39

$LtRg(\cdot)$, 27

$neigh(\cdot)$, 9

$Orig(\cdot)$, 158

IRP, 26

\mathbf{P}_G , 10

$|\cdot|$, 10

\dot{s} , 13

\dot{e} , 13

$\mathcal{R}_n^{ic}(\cdot)$, 92

$\mathcal{R}_n^{re}(\cdot)$, 92

$SLtRg(\cdot)$, 47

$Dst_{AM_{\alpha}}(\cdot)$, 157

$Src_{AM_{\alpha}}(\cdot)$, 157

s , 12

e , 12

w , 131

$zpred(\cdot)$, 123

$zsucc(\cdot)$, 123

$pred(\cdot)$, 10

$succ(\cdot)$, 10

Predicates

assignment motion

OrigJust, 163

eliminable(\cdot , \cdot), 158

AssMod, 140

AssOcc, 140

Blocked, 140

Dead, 143

Faint, 148

\bullet -*Insert*, 140

\bullet -*Just*, 141

LhsMod, 140

LhsUsed, 140

Redundant, 149

\bullet -*Remove*, 140

RhsMod, 140

\bullet -*Subst*, 141

expression motion

Comp, 22

Transp, 22

Hom-Delayed, 120

Hom-DnSafe, 118

Hom-Earliest, 118

LChange, 62

Hom-Latest, 120

Hom-Safe, 118

\bullet -*Insert*, 38, 60

\bullet -*Replace*, 38, 60

A1-Delayed, 61

A1-DnSafe, 61

A1-Latest, 61

A2-Delayed, 62

A2-Latest, 62
UsedLater[•], 61
A-Delayed, 96
A-Latest, 96
Change, 96
 •-*Correct*, 23
Delayed, 27
DnSafe, 23
Earliest, 25
 •-*Insert*, 22, 40, 97
Latest, 28
 •-*Replace*, 22, 40
 •-*Replace*, 96
Safe, 23
UpSafe, 23
UsedLater, 88

Orders and Preorders

\models , 168
 λ_{ass} , 171
 λ_{Φ}^{Φ} , 38, 41
 λ_{exp} , 171
 λ_{exp}^{Φ} , 38, 48
 λ_{lt}^{Φ} , 24, 27
 \sqsubseteq , 10

Transformations

assignment motion
DAE $_{\alpha}$, 144
EAAH $_{\alpha}$, 204
EASAS $_{\alpha}$, 204
FAE $_{\alpha}$, 148, 149
 \xrightarrow{ah} $_{\alpha}$, 140
 \xrightarrow{as} $_{\alpha}$, 139
 \xrightarrow{am} $_{\alpha}$, 157
 \xrightarrow{dae} $_{\alpha}$, 144
 \xrightarrow{eah} $_{\alpha}$, 204
 \xrightarrow{eam} $_{\alpha}$, 204
 \xrightarrow{eas} $_{\alpha}$, 204
 \xrightarrow{el} $_{\alpha}$, 157
 $\circ \rightarrow$, 169
EAM $_{\parallel}^{\mu}(G)$, 207
 \xrightarrow{fae} $_{\alpha}$, 148
AM $_{\alpha}^{\mu}(G)$, 175
EL $_{\alpha}^{\mu}(G)$, 175

SP $_{\varphi}^{\mu}(G)$, 190
 \xrightarrow{rae} $_{\alpha}$, 149
 \mapsto , 157
 $\text{TR}_{\alpha_1} \parallel \dots \parallel \text{TR}_{\alpha_k}$, 176
PDCE, 143
PFCE, 148
PRAE, 148
UPRE, 150
 CUMEC, 159
 EMEC, 204
 EUMEC, 204
 MEC, 155
 UMEC, 158
 expression motion
AE \mathcal{M}_{φ} , 24
BEM $_{\Phi_{f1}}$, 38
BEM $_{\Phi}$, 43
BEM $_{\varphi}$, 25
CBEM $_{\varphi}$, 118
CLEM $_{\varphi}$, 120
COEM $_{\varphi}$, 25
LFEM $_{\Phi}$, 60
FFEM $_{\Phi}$, 96
LEM $_{\Phi_{f1}}$, 38
LEM $_{\Phi}$, 46
LEM $_{\varphi}$, 28
 \xrightarrow{em} $_{\varphi}$, 190
AE \mathcal{M}_{Φ} , 41
COEM $_{\Phi}$, 41