


Papers on Compiler Optimizations: Analysis and Transformations (1952-1994)

Federico Bruzzone 
Computer Science Department
Università degli Studi di Milano
federico.bruzzone@unimi.it
<https://federicobruzzone.github.io/>

30 September 2025

Title	Authors	Venue Type	Venue Name	Type	Year
The problem of simplifying truth functions [9]	Quine W.	T&F	AMM	J	1952
Minimization of Boolean functions [10]	McCluskey E.	Bell Labs	Bell System Tech. J.	J	1956
An algorithm for translating Boolean expressions [11]	Arden B., Galler B., Graham R.	ACM	JACM	J	1962
High speed compilation of efficient object code [12]	Gear C.	ACM	CACM	J	1965
Peephole optimization [13]	McKeeman W.	ACM	CACM	J	1965
Program optimization [14]		-	PP	C	1966
Index Register Allocation [15]	Horwitz L., Karp R., Miller R., Winograd S.	ACM	JACM	J	1966
Analysis of Programs for Parallel Processing [16]	Bernstein A.	IEEE	TC (TEC)	J	1966
Object code optimization [17]	Lowry E., Medlock C.	ACM	CACM	J	1969
Local optimizations [18]	Bagwell Jr J.	ACM	PLDI (SCO)	C	1970
Detection and parallel execution of independent instructions [19]	Tjaden G., Flynn M.	IEEE	TCO	J	1970
Global common subexpression elimination [20]	Cocke J.	ACM	PLDI (SCO)	C	1970
The Generation of Optimal Code for Arithmetic Expressions [21]	Sethi R., Ullman J.	ACM	JACM	J	1970
Control flow analysis [22]	Allen F.	ACM	PLDI (SCO)	C	1970
A Basis for Program Optimization [23]		IFIP	NH	C	1971
A catalogue of optimizing transformations [24]	Frances E. Allen J.	-	PH	C	1972
Flow graph reducibility [25]	Hecht M., Ullman J.	ACM	STOC	C	1972
Use-definition chains with applications [26]		Elsevier	COLA (COMLAN)	C	1972
A global flow analysis algorithm [27]	Kennedy K.	T&F	JCM	J	1972
Safety of code motion [28]		T&F	JCM	J	1972
On the Number of Operations Simultaneously Executable in Fortran-Like Programs and Their Resulting Speedup [29]	Kuck D., Muraoka Y.	IEEE	TCO	J	1972
Testing flow graph reducibility [30]	Tarjan R.	ACM	STOC	C	1973
A unified approach to global program optimization [31]	Kildall G.	ACM	POPL	C	1973
Fast algorithms for the elimination of common subexpressions [32]	Ullman J.	Springer	Acta Inf.	J	1973
Interprocedural Analysis and the Information derived by it [33]	Allen F.	Springer	Prog. Meth.	J	1974
Register allocation via usage counts [34]	Freiburghouse R.	ACM	CACM	J	1974
Analysis of structured programs [35]	Kosaraju S.	ACM	STOC	C	1974

Continued on next page

Title	Authors	Venue Type	Venue Name	Type	Year
Characterizations of Reducible Flow Graphs [36]	Hecht M., Ullman J.	ACM	JACM	J	1974
The parallel execution of DO loops [37]	Lamport L.	ACM	CACM	J	1974
Interprocedural Data Flow Analysis [38]		-	IFIP	J	1974
Application of lattice algebra to loop optimization [39]	Fong A., Kam J., Ullman J.	ACM	POPL	C	1975
Program optimization - theory and practice [40]	Loveman D., Faneuf R.	ACM	PLDI (SCC)	C	1975
Time and parallel processor bounds for linear recurrence systems [41]	Chen S., Kuck D.	IEEE	TCO	J	1975
A simple algorithm for global data flow analysis problems [42]	Hecht M., Ullman J.	SIAM	SICOMP	J	1975
Program improvement by source to source transformation [43]	Loveman D.	ACM	POPL	C	1976
A program data flow analysis procedure [44]	Allen F., Cocke J.	ACM	CACM	J	1976
Recursion analysis for compiler optimization [45]	Walter K.	ACM	CACM	J	1976
Optimal Code Generation for Expression Trees [46]	Aho A., Johnson S.	ACM	JACM	J	1976
A Fast and Usually Linear Algorithm for Global Flow Analysis [47]	Graham S., Wegman M.	ACM	JACM	J	1976
Code generation for expressions with common subexpressions [48]	Aho A., Johnson S., Ullman J.	ACM	POPL	C	1976
Global data flow analysis and iterative algorithms [49]	Kam J., Ullman J.	ACM	JACM	J	1976
Compiler Analysis of the Value Ranges for Variables [50]	Harrison W.	IEEE	TSE	J	1977
On Live-Dead Analysis for Global Data Flow Problems [51]	Kou L.	ACM	JACM	J	1977
Symbolic evaluation and the global value graph [52]	Reif J., Lewis H.	ACM	POPL	C	1977
High-level data flow analysis [53]	Rosen B.	ACM	CACM	J	1977
Abstract interpretation [54]	Cousot P., Cousot R.	ACM	POPL	C	1977
An algorithm for reduction of operator strength [55]	Cocke J., Kennedy K.	ACM	CACM	J	1977
A transformation system for developing recursive programs [56]	Burstall R., Darlington J.	ACM	JACM	J	1977
Arithmetic shifting considered harmful [57]	Steele G.	ACM	SIGPLAN Notices	J	1977
Monotone data flow analysis frameworks [58]	Kam J., Ullman J.	Springer	Acta Inf.	J	1977
Program Improvement by Source-to-Source Transformation [59]	Loveman D.	ACM	CACM	J	1977
An analysis of inline substitution for a structured programming language [60]	Scheffler R.	ACM	CACM	J	1977
Program Optimization Using Invariants [61]	Katz S.	IEEE	TSE	J	1978
A new method for compiler code generation [62]	Glanville R., Graham S.	ACM	POPL	C	1978
A practical interprocedural data flow analysis algorithm [63]	Barth J.	ACM	CACM	J	1978
Data Flow Analysis for Procedural Languages [64]	Rosen B.	ACM	JACM	J	1979
Constructing the Call Graph of a Program [65]	Ryder B.	IEEE	TSE	J	1979
Data flow languages [66]	Ackerman W.	IEEE	MARK	W	1979
Time and parallel processor bounds for Fortran-like loops [67]	Chen S.	IEEE	TCO	J	1979
Unrolling loops in Fortran [68]	Dongarra J., Hinds A.	Wiley	SPE	J	1979
A fast algorithm for finding dominators in a flowgraph [69]	Lengauer T., Tarjan R.	ACM	TOPLAS	J	1979
An efficient way to find the side effects of procedural calls and the aliases of variables [70]	Banning J.	ACM	POPL	C	1979
Global optimization by suppression of partial redundancies [71]	Morel E., Renvoise C.	ACM	CAMC	J	1979
Predicting the effects of optimization on a procedure body [72]	Ball J.	ACM	PLDI (SCC)	C	1979
Structural analysis: A new approach to flow analysis in optimizing compilers [73]	Sharir M.	Elsevier	COLA (COMLAN)	C	1980
The design and application of a retargetable peephole optimizer [74]	Davidson J., Fraser C.	ACM	TOPLAS	J	1980
Data flow supercomputers [75]	Dennis J.	IEEE	Computer	J	1980
Program optimization and parallelization using idioms [76]	Pinter S., Pinter R.	ACM	POPL	C	1980
High-speed multiprocessors and compilation techniques [77]	Padua K.	IEEE	TCO	J	1980
A composite algorithm for strength reduction and code movement optimization [78]	Dhamdhere D., Isaac J.	Springer	ACIS	J	1980
Interprocedural data flow analysis in the presence of pointers, procedure variables, and label variables [79]	Weihl W.	ACM	POPL	C	1980
Deciding Linear Inequalities by Computing Loop Residues [80]	Shostak R.	ACM	JACM	J	1981
A precise inter-procedural data flow algorithm [81]	Myers E.	ACM	POPL	C	1981

Continued on next page

Title	Authors	Venue Type	Venue Name	Type	Year
Register allocation via coloring [82]	Chaitin G., Auslander M., Chandra A., John H., Martin E M.	Elsevier	COLA (COMLAN)	C	1981
Reduction of operator strength [83]	Allen F., Cocke J., Kennedy K.	NJ	Program flow analysis	J	1981
Dependence graphs and compiler optimizations [84]	Kuck D., Kuhn R., Padua D., Leasure B.	ACM	POPL	C	1981
On the performance enhancement of paging systems through program analysis and transformations [85]	Kuck L.	IEEE	TCO	J	1981
Using Peephole Optimization on Intermediate Code [86]	Tanenbaum A., van Staveren H., Stevenson J.	ACM	TOPLAS	J	1982
Optimizing delayed branches [87]	Gross T., Hennessy J.	ACM	MICRO	W	1982
A composite hoisting-strength reduction transformation for global program optimization part i [88]	Joshi S., Dhamdhere D.	T&F	JCM	J	1982
Optimization of range checking [89]	Markstein V., Cocke J., Markstein P.	ACM	PLDI (SCC)	C	1982
Register allocation and spilling via graph coloring [90]	Chaitin G.	ACM	PLDI	C	1982
Experience with the SETL optimizer [91]	Freudenberger S., Schwartz J., Sharir M.	ACM	TOPLAS	J	1983
Postpass Code Optimization of Pipeline Constraints [92]	Hennessy J., Gross T.	ACM	TOPLAS	J	1983
Conversion of control dependence to data dependence [93]	Allen J., Kennedy K., Porterfield C.	ACM	POPL	C	1983
Code selection through object code optimization [94]	Davidson J., Fraser C.	ACM	TOPLAS	J	1984
Register allocation and exhaustive peephole optimization [95]	Davidson J., Fraser C.	Wiley	SPE	J	1984
Automatic generation of peephole optimizations [96]	Davidson J., Fraser C.	Springer	CC	C	1984
Analysis of interprocedural side effects in a parallel programming environment [97]	Callahan D., Kennedy K.	Springer	ICSP	C	1984
Polyvariant mixed computation for analyzer programs [98]	Bulyonkov M.	Springer	Acta Inf.	J	1984
Stream processing [99]	Goldberg A., Paige R.	ACM	LFP	C	1984
A hierarchical basis for reordering transformations [100]	Warren J.	ACM	POPL	C	1984
Register allocation by priority-based coloring [101]	Chow F., Hennessy J.	ACM	PLDI (SCC)	C	1984
Automatic loop interchange [102]	Allen J., Kennedy K.	ACM	PLDI (SCC)	C	1984
Efficient computation of flow insensitive interprocedural summary information [103]	Cooper K., Kennedy K.	ACM	PLDI (SCC)	C	1984
Program transformations in a denotational setting [104]	Nielson F.	ACM	TOPLAS	J	1985
On linearizing parallel code [105]	Ferrante J., Mace M.	ACM	POPL	C	1985
Distributed execution of functional programs using serial combinators [106]	Hudak P., Goldberg B.	IEEE	TCO	J	1985
Strictness analysis-a practical approach [107]	Clack C., Peyton Jones S.	Springer	FPCA	C	1985
A linear algorithm for finding dominators in flow graphs and related problems [108]	Harel D.	ACM	STOC	C	1985
Advanced compiler optimizations for supercomputers [109]	Padua D., Wolfe M.	ACM	CACM	J	1986
Efficient compilation of linear recursive functions into object level loops [110]	Harrison P., Khoshnevisan H.	ACM	PLDI (SCC)	C	1986
The impact of interprocedural analysis and optimization in the Rn programming environment [111]	Cooper K., Kennedy K., Torczon L.	ACM	TOPLAS (LOPES)	C	1986
Efficient instruction scheduling for a pipelined architecture [112]	Gibbons P., Muchnick S.	ACM	PLDI (SCC)	C	1986
Efficient symbolic analysis of programs [113]	John H. Reif H.	ACM	JCSS	J	1986
Graph-Based Algorithms for Boolean Function Manipulation [114]	Bryant R.	IEEE	TC	J	1986
Loops skewing: The wavefront method revisited [115]	Wolfe M.	Springer	JPP	J	1986
Highly concurrent scalar processing [116]	Hsu P., Davidson E.	ACM	CAN	J	1986
Multiplication by integer constants [117]	Bernstein R.	Wiley	SPE	J	1986

Continued on next page

Title	Authors	Venue Type	Venue Name	Type	Year
Global register allocation at link time [118]	Wall D.	ACM	PLDI	C	1986
Interprocedural constant propagation [119]	Callahan D., Cooper K., Kennedy K.	ACM	PLDI	C	1986
Interprocedural optimization: eliminating unnecessary recompilation [120]	Cooper K., Kennedy K., Torczon L.	ACM	CC	C	1986
Interprocedural dependence analysis and parallelization [121]	Burke M., Cytron R.	ACM	PLDI	C	1986
Effectiveness of a machine-level, global optimizer [122],	Johnson M., Miller T.	ACM	PLDI (SCC)	C	1986
Direct parallelization of call statements [123]	Triplet R., Irigoin F., Feautrier P.	ACM	PLDI (SCC)	C	1986
Code motion of control structures in high-level languages [124]	Cytron R., Lowry A., Zadeck F.	ACM	POPL	C	1986
Automatic inference and fast interpretation of peephole optimization rules [125]	Davidson J., Fraser C.	Wiley	SPE	J	1987
Compiler Algorithms for Synchronization [126]	Midkiff S., Padua D.	IEEE	TCO	J	1987
Automatic decomposition of scientific programs for parallel execution [127]	Allen r., Callahan D., Kennedy K.	ACM	POPL	C	1987
Guided Self-Scheduling: A Practical Scheduling Scheme for Parallel Supercomputers [128]	Polychronopoulos C., Kuck D.	IEEE	TCO	J	1987
Loop coalescing: A compiler transformation for parallel machines [129]	Polychronopoulos C.	ACM	ICPP	C	1987
Strategies for cache and local memory management by global program transformation [130]	Gannon D., Jalby W., Gallivan K.	Springer	ICS	C	1987
The program dependence graph and its use in optimization [131]	Ferrante J., Ottenstein K., Warren J.	ACM	TOPLAS	J	1987
Automatic translation of Fortran programs to vector form [132]	Allen R., Kennedy K.	ACM	TOPLAS	J	1987
Safety consideration for storage allocation optimizations [133]	Chase D.	ACM	PLDI	C	1988
Efficient and correct execution of parallel programs that share memory [134]	Shasha D., Snir M.	ACM	TOPLAS (LOPES)	C	1988
On the control dependence in the program dependence graph [135]	Natour I.	ACM	CSC	C	1988
Resource requirements of dataflow programs [136]	Culler D.	ACM	ISCA	C	1988
Incremental data flow analysis via dominator and attribute update [137]	Carroll M., Ryder B.	ACM	POPL	C	1988
An efficient approach to data flow analysis in a multiple pass global optimizer [138]	Jain S., Thompson C.	ACM	PLDI	C	1988
A solution to a problem with Morel and Renvoise's "Global optimization by suppression of partial redundancies" [139]	Drechsler K., Stadel M.	ACM	TOPLAS	J	1988
Efficient computation of flow-insensitive interprocedural summary information—a correction [140]	Cooper K., Kennedy K.	ACM	SIGPLAN Notices	J	1988
Dependence of multi-dimensional array references [141]	Wallace D.	ACM	ICS	C	1988
The importance of direct dependences for automatic parallelization [142]	Brandes T.	ACM	ICS	C	1988
Introducing symbolic problem solving techniques in the dependence testing phases of a vectorizer [143]	Lichnewsky A., Thomasset F.	ACM	ICS	C	1988
Generating sequential code from parallel code [144]	Ferrante J., Mace M., Simons B.	ACM	SC	C	1988
CRegs: a new kind of memory for referencing arrays and pointers [145]	Dietz H., Chi C.	ACM	ICS	C	1988
Advanced loop optimizations for parallel computers [146]	Polychronopoulos C.	ACM	ICS	C	1988
An introduction to a formal theory of dependence analysis [147]	Banerjee U.	Springer	JSC	J	1988
Analysis of interprocedural side effects in a parallel programming environment [148]	Callahan D., Kennedy K.	ACM	ICS	C	1988
Array expansion [149]	Feautrier P.	ACM	ICS	C	1988
Loop quantization: A generalized loop unwinding technique [150]		ACM	JPDC	J	1988
Supernode partitioning [151]	Irigoin F., Triolet R.	ACM	PLDI	C	1988
A fast algorithm for code movement optimisation [152]	Dhamdhare D.	ACM	PLDI	C	1988
Compiling programs for distributed-memory multiprocessors [153]	Callahan D., Kennedy K.	Elsevier	JSC	J	1988

Continued on next page

Title	Authors	Venue Type	Venue Name	Type	Year
Optimal loop parallelization [154]	Aiken A., Nicolau A.	ACM	PLDI	C	1988
Software pipelining: an effective scheduling technique for VLIW machines [155]	Lam M.	ACM	PLDI	C	1988
The program summary graph and flow-sensitive interprocedural data flow analysis [156]	Callahan D.	ACM	PLDI	C	1988
A framework for determining useful parallelism [157]	Allen F., Burke M., Cytron R., Ferrante,	ACM	ICS	C	1988
An overview of the PTRAN analysis system for multiprocessing [158]	Allen F., Burke M., Charles P., Cytron	Elsevier	JPDC	J	1988
Efficient interprocedural analysis for program parallelization and restructuring [159]	Li Z., Yew P.	ACM	PPoPP (PPEALS)	C	1988
Compiling issues for supercomputers [160]	Girkar M., Polychronopoulos C.	ACM/IEE	SC	C	1988
Minimizing register usage penalty at procedure calls [161]	Chow F.	ACM	PLDI	C	1988
Estimating interlock and improving balance for pipelined architectures [162]	Callahan D., Cocke J., Kennedy K.	Elsevier	JPDC	J	1988
Global value numbers and redundant computations [163]	Rosen B., Wegman M., Zadeck F.	ACM	POPL	C	1988
Perfect pipelining: A new loop parallelization technique [164]	Aiken A., Nicolau A.	Springer	ESOP	C	1988
Code scheduling and register allocation in large basic blocks [165]	Goodman J., Hsu W.	ACM	SC	C	1988
Interprocedural side-effect analysis in linear time [166]	Cooper K., Kennedy K.	ACM	PLDI	C	1988
A technique for summarizing data access and its use in parallelism enhancing transformations [167]	Balasundaram V., Kennedy K.	ACM	PLDI	C	1989
Interprocedural analysis vs. procedure integration [168]	Stephen Richardson M.	Elsevier	IPL	J	1989
Unified management of registers and cache using liveness and cache bypass [169]	Chi C., Dietz H.	ACM	PLDI	C	1989
A new algorithm for composite hoisting and strength reduction optimisation [170]	Dhamdhare D.	T&F	JCM	J	1989
Interprocedural data flow testing [171]	Harrold M., Soffa M.	ACM	SEN	C	1989
Program optimization for instruction caches [172]	McFarling S.	ACM	ASPLOS	C	1989
Dependence analysis for pointer variables [173]	Horwitz S., Pfeiffer P., Reps T.	ACM	PLDI	C	1989
The program dependence graph and vectorization [174]	Baxter W., Bauer H.	ACM	POPL	C	1989
Achieving high instruction cache performance with an optimizing compiler [175]	Hwu W., Chang P.	ACM	ISCA	C	1989
Evaluating the performance of four snooping cache coherency protocols [176]	Eggers S., Katz R.	IEEE	ICSA	C	1989
Scans as primitive parallel operations [177]	Blelloch G.	IEEE	TCO	J	1989
Code generation using tree matching and dynamic programming [178]	Aho A., Ganapathi M., Tjiang S.	ACM	TOPLAS	J	1989
Register allocation via clique separators [179]	Gupta R., Soffa M., Steele T.	ACM	PLDI	C	1989
Fast interprocedural alias analysis [180]	Cooper K., Kennedy K.	ACM	POPL	C	1989
Coloring heuristics for register allocation [181]	Briggs P., Cooper K., Kennedy K., Torczon L.	ACM	PLDI	C	1989
More iteration space tiling [182]	Wolfe M.	ACM/IEE	SC	C	1989
Data dependence analysis on multi-dimensional array references [183]	Li Z., Yew P., Zhu C.	ACM	ICS	C	1989
Spill code minimization techniques for optimizing compilers [184]	Bernstein D., Golumbic M., Mansour y., Pinter R., D. K., H. N.	ACM	PLDI	C	1989
Customization: Optimizing compiler technology for SELF, a dynamically-typed OOP language [185]	Chambers C., Ungar D.	ACM	PLDI	C	1989
An efficient method of computing static single assignment form [186]	Cytron R., Ferrante J., Rosen B., Wegman,	ACM	POPL	C	1989

Continued on next page

Title	Authors	Venue Type	Venue Name	Type	Year
The value flow graph: A program representation for optimal program transformations [187]	Steffen B., Knoop J., Rüthing O.	Springer	ESOP	C	1990
Compiler techniques for data partitioning of sequentially iterated parallel loops [188]	Hudak D., Abraham S.	ACM	SC	C	1990
Vectorization of tree traversals [189]		Elsevier	JCP	J	1990
Loop displacement: an approach for transforming and scheduling loops for parallel execution [190]	Gupta R.	ACM	SC	C	1990
Experience with interprocedural analysis of array side effects [191]	Havlak P., Kennedy K.	ACM	SC	C	1990
An approach to ordering optimizing transformations [192]	Whitfield D., Soffa M.	ACM	PPoPP	C	1990
Register allocation across procedure and module boundaries [193]	Santhanam V., Odnert D.	ACM	PLDI	C	1990
Region Scheduling: An Approach for Detecting and Redistributing Parallelism [194]	Gupta R., Soffa M.	IEEE	TSE	J	1990
Constructing the procedure call multigraph [195]	Callahan D., Carle A., Hall M., Kennedy K.	IEEE	TSE	J	1990
On the perfect accuracy of an approximate subscript analysis test [196]	Klappholz D., Psarris K., Kong X.	ACM	CAN	J	1990
Structured dataflow analysis for arrays and its use in an optimizing compiler [197]	Gross T., Steenkiste P.	Wiley	Software: Practice and Experience	J	1990
Compilation of Haskell array comprehensions for scientific computing [198]	Anderson S., Hudak P.	ACM	PLDI	C	1990
How to read floating point numbers accurately [199]	Clinger W.	ACM	PLDI	C	1990
How to print floating-point numbers accurately [200]	Steele G., White J.	ACM	PLDI	C	1990
Profile guided code positioning [201]	Pettis K., Hansen R.	ACM	PLDI	C	1990
Updating distributed variables in local computations [202]	Gerndt M.	Wiley	SPE	J	1990
An interval-based approach to exhaustive and incremental interprocedural data-flow analysis [203]	Burke M.	ACM	TOPLAS	J	1990
The priority-based coloring approach to register allocation [204]	Chow F., Hennessy J.	ACM	TOPLAS	J	1990
Improving register allocation for subscripted variables [205]	Callahan D., Carr S., Kennedy K.	ACM	PLDI	C	1990
Analysis of pointers and structures [206]	Chase D., Wegman M., Zadeck F.	ACM	PLDI	C	1990
Loop distribution with arbitrary control flow [207]	Kennedy K., McKinley K.	ACM	PLDI	C	1990
Graph coloring register allocation for processors with multi-register operands [208]	Nickerson B.	ACM	PLDI	C	1990
A data locality optimizing algorithm [209]	Wolf M., Lam M.	ACM	PLDI	C	1991
On optimal parallelization of arbitrary loops [210]	Uwe Schwiegelshohn F.	Elsevier	JPDC	J	1991
On the adequacy of dependence-based representations for programs with heaps [211]	Pfeiffer P., Selke R.	Springer	TACS	C	1991
Automatic construction of sparse data flow evaluation graphs [212]	Choi J., Cytron R., Ferrante J.	ACM	POPL	C	1991
Compiler optimizations for Fortran D on MIMD distributed-memory machines [213]	Hiranandani S., Kennedy K., Tseng C.	ACM	SC	C	1991
Register allocation via hierarchical graph coloring [214]	Callahan D., Koblenz B.	ACM	PLDI	C	1991
Circular scheduling: a new technique to perform software pipelining [215]	Jain S.	ACM	PLDI	C	1991
Efficient DAG construction and heuristic calculation for instruction scheduling [216]	Smotherman M., Krishnamurthy S., Aravind P.	ACM	MICRO	W	1991
Efficiently computing static single assignment form and the control dependence graph [217]	Cytron R., Ferrante J., Rosen B., Wegman,	ACM	TOPLAS	J	1991
Software prefetching [218]	Callahan D., Kennedy K., Porterfield A.	ACM	ASPLOS	C	1991
Compiling global name-space parallel loops for distributed execution [219]	Koelbel C., Mehrotra P.	IEEE	TPDS	J	1991

Continued on next page

Title	Authors	Venue Type	Venue Name	Type	Year
An implementation of interprocedural bounded regular section analysis [220]	Havlak P., Kennedy K.	ACM	TPDS	J	1991
Limits of instruction-level parallelism [221]	Wall D.	ACM	ASPLOS	C	1991
Uniform techniques for loop optimization [222]	Pugh W.	ACM	ICS	C	1991
A data locality optimizing [209]	Wolf M., Lam M.	ACM	PLDI	C	1991
Constant propagation with conditional branches [223]	Wegman M., Zadeck F.	ACM	TOPLAS	J	1991
Efficient and exact data dependence analysis [224]	Maydan D., Hennessy J., Lam M.	ACM	PLDI	C	1991
Efficiently computing static single assignment [225]	Cytron R., Ferrante J., Rosen B., Wegman,	ACM	TOPLAS	J	1991
Global instruction scheduling for superscalar machines [226]	Bernstein D., Rodeh M.	ACM	PLDI	C	1991
Practical adaption of the global optimization algorithm of Morel and Renvoise [227]	Dhamdhere D.	ACM	TOPLAS	J	1991
The cache performance and optimizations of blocked algorithms [228]	Lam M., Rothberg E., Wolf M.	ACM	ASPLOS	C	1991
A loop transformation theory and an algorithm to maximize parallelism [229]	Wolf M., Lam M.	IEEE	TPDS	J	1991
Dataflow analysis of array and scalar references [230]	Feautrier P.	Springer	JPP	J	1991
Optimization of array accesses by collective loop transformations [231]	Sarkar V., Gao G.	ACM	SC	C	1991
Interprocedural transformations for parallel code generation [232]	Hall M., Kennedy K., McKinley K.	ACM/IEE	SC	C	1991
Practical dependence testing [233]	Goff G., Kennedy K., Tseng C.	ACM	PLDI	C	1991
Procedure merging with instruction caches [234]	McFarling S.	ACM	PLDI	C	1991
An experiment with inline substitution [235]	Cooper K., Hall M., Torczon L.	Wiley	SPE	J	1991
Access normalization: loop restructuring for NUMA compilers [?]		ACM	ASPLOS	C	1991
Non-unimodular transformations of nested loops [236]	Ramanujam J.	ACM	SC	C	1992
A general framework for iteration-reordering loop transformations [237]	Sarkar V., Thekkath R.	ACM	PLDI	C	1992
A program integration algorithm that accommodates semantics-preserving transformations [238]	Yang W., Horwitz S., Reps T.	ACM	TOSEM	J	1992
The transitive closure of control dependence: The iterated join [239]	Weiss M.	ACM	TOPLAS (LOPLAS)	J	1992
Abstract description of pointer data structures: an approach for improving the analysis and optimization of imperative programs [240]	Hummel J., Hendren L., Nicolau A.	ACM	TOPLAS (LOPLAS)	J	1992
Abstractions for recursive pointer data structures: improving the analysis and transformation of imperative programs [241]	v. Hanxleden R., Kennedy K.	ACM	PLDI	C	1992
Generalized dominators and post-dominators [242]	Gupta R.	ACM	POPL	C	1992
A comprehensive approach to parallel data flow analysis [243]	Lee Y., Ryder B.	ACM	ICS	C	1992
Integrating scalar optimization and parallelization [244]	Tjiang S., Wolf M., Lam M., Pieper K.	Springer	LCPC	C	1992
Sharlit—a tool for building optimizers [245]	Tjiang S., Hennessy J.	ACM	PLDI	C	1992
How to analyze large programs efficiently and informatively [246]	Dhamdhere D., Rosen B., Zadeck F.	ACM	PLDI	C	1992
Compiler code transformations for superscalar-based high performance systems [247]	Mahlke S., Chen W., Gyllenhaal J., Hwu W.	IEEE	SC	C	1992
Engineering a simple, efficient code-generator generator [248]	Fraser C., Hanson D., Proebsting T.	ACM	TOPLAS (LOPLAS)	J	1992
Some efficient solutions to the affine scheduling problem. I. One-dimensional time [249]	Feautrier P.	Elsevier	JPP	J	1992
Avoiding unconditional jumps by code replication [250]	Mueller F., Whalley D.	ACM	PLDI	C	1992
Rematerialization [251]	Briggs P., Cooper K., Torczon L.	ACM	PLDI	C	1992

Continued on next page

Title	Authors	Venue Type	Venue Name	Type	Year
Coloring register pairs [252]	Briggs P., Cooper K., Torczon L.	ACM	TOPLAS (LOPLAS)	J	1992
Optimizing for parallelism and data locality [253]	Kennedy K., McKinley K.	ACM	SC	C	1992
Maximizing loop parallelism and improving data locality via loop fusion and distribution [254]	Kennedy K., McKinley K.	ACM	LCPC	W	1992
Beyond induction variables [255]	Wolfe M.	ACM	PLDI	C	1992
Efficient call graph analysis [256]	Hall M., Kennedy K.	ACM	TOPLAS (LOPLAS)	J	1992
A safe approximate algorithm for interprocedural aliasing [257]	Landi W., Ryder B.	ACM	PLDI	C	1992
Relaxing SIMD control flow constraints using loop transformations [241]	v. Hanxleden R., Kennedy K.	ACM	PLDI	C	1992
Unexpected side effects of inline substitution: a case study [258]	Cooper K., Hall M., Torczon L.	ACM	TOPLAS (LOPLAS)	J	1992
Eliminating false positives using the omega test [259]	Pugh W., Wonnacott D.	ACM	PLDI	C	1992
Lazy code motion [260]	Knoop J., Rüthing O., Steffen B.	ACM	PLDI	C	1992
Software support for speculative loads [261]	Rogers A., Li K.	ACM	ASPLOS	C	1992
A practical algorithm for exact array dependence analysis [262]	Pugh W.	ACM	CACM	J	1992
Array privatization for parallel execution of loops [263]	Li Z.	ACM	ICS	C	1992
The power test for data dependence [264]	Wolfe M., Tseng C.	IEEE	TCO	J	1992
Eliminating branches using a superoptimizer and the GNU C compiler [265]	Granlund T., Kenner R.	ACM	PLDI	C	1992
Design and evaluation of a compiler algorithm for prefetching [266]	Mowry T., Lam M., Gupta A.	ACM	ASPLOS	C	1992
Data dependence and data-flow analysis of arrays [267]	Maydan D., Amarsinghe S., Lam M.	Springer	LCPC	C	1992
Advanced compiler optimizations for sparse computations [268]	Bik A., Wijshoff H.	ACM	SC	C	1993
Access normalization: loop restructuring for NUMA computers [269]	Li W., Pingali K.	ACM	TOCS	J	1993
On the conversion of indirect to direct recursion [270]	Kaser O., Ramakrishnan C., Pawagi S.	ACM	TOPLAS (LOPLAS)	J	1993
A practical data flow framework for array reference analysis and its use in optimizations [271]	Duesterwald E., Gupta R., Soffa M.	ACM	PLDI	C	1993
Using lifetime predictors to improve memory allocation performance [272]	Barrett D., Zorn B.	ACM	PLDI	C	1993
Data flow analysis for parallel programs [273]	Ito M., Zaafrani A.	ACM	CSC	C	1993
Dependence-based program analysis [274]	Johnson R., Pingali K.	ACM	PLDI	C	1993
A variation of Knoop, Rüthing, and Steffen's lazy code motion [275]	Drechsler K., Stadel M.	ACM	PLDI	C	1993
Lazy strength reduction [276]	Knoop J.	C&H	JPL	J	1993
Collective loop fusion for array contraction [277]	Gao G., Olsen R., Sarkar V., Thekkath R.	Springer	LCPC	C	1993
Interprocedural constant propagation: an empirical study [278]	Metzger R., Stroud S.	ACM	TOPLAS (LOPLAS)	J	1993
Symbolic analysis: A basis for parallelization, optimization, and scheduling of programs [279]	Haghighat M., Polychronopoulos C.	Springer	LCPC	W	1993
Array-data flow analysis and its use in array privatization [280]	Maydan D., Amarasinghe S., Lam M.	ACM	POPL	C	1993
Automatic array alignment in data-parallel programs [281]	Chatterjee S., Gilbert J., Schreiber R.	ACM	POPL	C	1993
Global optimizations for parallelism and locality on scalable parallel machines [282]	Anderson J., Lam M.	ACM	PLDI	C	1993
Instruction-Level Parallel Processing: History, Overview, and Perspective [283]	Rau B., Fisher J.	Springer	JSC	J	1993
A practical system for intermodule code optimization at link-time [284]	Srivastava A.	C&H	JPL	J	1993
Loop-level parallelism in numeric and symbolic programs [285]	Larus J.	IEEE	TPDS	J	1993
Orchestrating interactions among parallel computations [286]	Graham S., Lucco S., Sharp O.	ACM	PLDI	C	1993

Continued on next page

Title	Authors	Venue Type	Venue Name	Type	Year
The superblock: an effective technique for VLIW and superscalar compilation [287]	Hwu W., Mahlke S., Chen W., Chang,	Springer	JSC	J	1993
Optimizing array bound checks using flow analysis [288]	Gupta R.	ACM	PLDI	C	1993
Register allocation with instruction scheduling [289]	Pinter S.	ACM	PLDI	C	1993
A methodology for procedure cloning [290]	Cooper K., Hall M., Kennedy K.	Elsevier	COLA (COMLAN)	J	1993
Efficient flow-sensitive interprocedural computation of pointer-induced aliases and side effects [291]	Choi J., Burke M., Carini P.	ACM	POPL	C	1993
Automatic array privatization [292]	Tu P., Padua D.	Springer	LCPC	W	1993
Interprocedural modification side effect analysis with pointer aliasing [293]	Landi W., Ryder B., Zhang S.	ACM	PLDI	C	1993
Branch prediction for free [294]	Ball T., Larus J.	ACM	PLDI	C	1993
Enabling unimodular transformations [295]	Sass R., Mutka M.	ACM	SC	C	1994
Compiler optimizations for improving data locality [296]	Carr S., McKinley K., Tseng C.	ACM	ASPLOS	C	1994
The revival transformation [297]	Feigen L., Klappholz D., Casazza R.	ACM	PLDI	C	1994
The range test: a dependence test for symbolic, non-linear expressions [298]	Blume W., Eigenmann R.	ACM	SC	C	1994
Link-time optimization of address calculation on a 64-bit architecture [299]	Srivastava A., Wall D.	ACM	PLDI	C	1994
Effective partial redundancy elimination [300]	Briggs P., Cooper K.	ACM	PLDI	C	1994
Partial dead code elimination [301]	Knoop J., Rüthing O., Steffen B.	ACM	PLDI	C	1994
A general data dependence test for dynamic, pointer-based data structures [240]	Hummel J., Hendren L., Nicolau A.	ACM	PLDI	C	1994
Instruction scheduling over regions: A framework for scheduling across basic blocks [302]	Mahadevan U., Ramakrishnan S.	Springer	CC	C	1994
Value dependence graphs: representation without taxation [303]	Weise D., Crew R., Ernst M.	ACM	POPL	C	1994
Zero-cost range splitting [304]	Kurlander S., Fischer C.	ACM	PLDI	C	1994
Reducing branch costs via branch alignment [305]	Calder B., Grunwald D.	ACM	OSR	J	1994
Improving the ratio of memory operations to floating-point operations in loops [306]	Carr S., Kennedy K.	ACM	TOPLAS	J	1994
Optimizing multi-method dispatch using compressed dispatch tables [307]	Amiel E., Gruber O., Simon E.	ACM	OOPSLA	C	1994
Improving the accuracy of static branch prediction using branch correlation [308]	Young C., Smith M.	ACM	ASPLOS	C	1994
A compiler framework for restructuring data declarations to enhance cache and TLB effectiveness [309]	David F. Bacon J., Dz-Ching Ju K.	IEEE	CASCON	C	1994
False sharing and spatial locality in multiprocessor caches [310]	Torrellas J., Lam H., Hennessy J.	IEEE	TCO	J	1994
Reassociation and strength reduction [311]	Markstein P., Markstein V., Zadeck F.	ACM	SCO	C	1994
The alignment-distribution graph [312]	Chatterjee S., Gilbert J., Schreiber R.	Springer	LCPC	W	1994
Improvements to graph coloring register allocation [313]	Briggs P., Cooper K., Torczon L.	ACM	TOPLAS	J	1994
Context-sensitive interprocedural points-to analysis in the presence of function pointers [314]	Emami M., Ghiya R., Hendren L.	ACM	PLDI	C	1994
Interprocedural may-alias analysis for pointers: beyond k-limiting [315]	Deutsch A.	ACM	PLDI	C	1994

Table 1: The papers from *Advanced Compiler Design and Implementation* [1] (A), *Compilers: Principles, Techniques, and Tools* [2] (D), *Engineering a Compiler* [3] (E), *Modern Compiler Implementation in C/Java/ML* [4–6] (M), *Optimizing Compilers for Modern Architectures: A Dependence-Based Approach* [7] (O), and the Bacon’s survey [8] (S). In the **Type** column, C stands for conference, J for journal, and W for workshop.

References

- [1] Steven Muchnick. *Advanced compiler design implementation*. Morgan kaufmann, 1997.
- [2] Alfred V. Aho, Monica S. Lam, Ravi Sethi, and Jeffrey D. Ullman. *Compilers: Principles, Techniques, and Tools*. Addison-Wesley, Boston, MA, USA, second edition, 2006.
- [3] Keith D. Cooper and Linda Torczon. *Engineering a Compiler*. Morgan Kaufmann, November 2022.
- [4] Andrew W Appel. Modern compiler design in c, 1998.
- [5] W Appel Andrew and Palsberg Jens. Modern compiler implementation in java. In ISBN 0–521–58388–8. Cambridge University Press, 2002.
- [6] Andrew W Appel. *Modern compiler implementation in ML*. Cambridge university press, 1998.
- [7] Ken Kennedy and John R Allen. *Optimizing compilers for modern architectures: a dependence-based approach*. Morgan Kaufmann Publishers Inc., 2001.
- [8] David F Bacon, Susan L Graham, and Oliver J Sharp. Compiler transformations for high-performance computing. *ACM Computing Surveys (CSUR)*, 26(4):345–420, 1994.
- [9] Willard V Quine. The problem of simplifying truth functions. *The American mathematical monthly*, 59(8):521–531, 1952.
- [10] Edward J McCluskey. Minimization of boolean functions. *The Bell System Technical Journal*, 35(6):1417–1444, 1956.
- [11] Bruce W Arden, Bernard A Galler, and Robert M Graham. An algorithm for translating boolean expressions. *Journal of the ACM (JACM)*, 9(2):222–239, 1962.
- [12] C. W. Gear. High speed compilation of efficient object code. *Commun. ACM*, 8(8):483–488, August 1965.
- [13] W. M. McKeeman. Peephole optimization. *Commun. ACM*, 8(7):443–444, July 1965.
- [14] Frances E. Allen. Program optimization. In Mark Halpern and Christopher Shaw, editors, *Annual Review of Automatic Programming*, volume 5, pages 239–307. Pergamon Press, Elmsford, NY, 1966.
- [15] L. P. Horwitz, R. M. Karp, R. E. Miller, and S. Winograd. Index register allocation. *J. ACM*, 13(1):43–61, January 1966.
- [16] A. J. Bernstein. Analysis of programs for parallel processing. *IEEE Transactions on Electronic Computers*, EC-15(5):757–763, 1966.
- [17] Edward S. Lowry and C. W. Medlock. Object code optimization. *Commun. ACM*, 12(1):13–22, January 1969.
- [18] John T Bagwell Jr. Local optimizations. In *Proceedings of a symposium on Compiler optimization*, pages 52–66, 1970.
- [19] Garold S Tjaden and Michael J Flynn. Detection and parallel execution of independent instructions. *IEEE Transactions on computers*, 19(10):889–895, 1970.

REFERENCES

- [20] John Cocke. Global common subexpression elimination. In *Proceedings of a Symposium on Compiler Optimization*, page 20–24, New York, NY, USA, 1970. Association for Computing Machinery.
- [21] Ravi Sethi and J. D. Ullman. The generation of optimal code for arithmetic expressions. *J. ACM*, 17(4):715–728, October 1970.
- [22] Frances E. Allen. Control flow analysis. *SIGPLAN Not.*, 5(7):1–19, July 1970.
- [23] Frances E. Allen. A basis for program optimization. In Charles V. Freiman, John E. Griffith, and Jack L. Rosenfeld, editors, *Information Processing, Proceedings of IFIP Congress 1971, Volume 1 - Foundations and Systems, Ljubljana, Yugoslavia, August 23-28, 1971*, pages 385–390. North-Holland, 1971.
- [24] Frances E. Allen and John Cocke. A catalogue of optimizing transformations. In Randall Rustin, editor, *Design and Optimization of Compilers*. Prentice-Hall, 1972.
- [25] Matthew S. Hecht and Jeffrey D. Ullman. Flow graph reducibility. In *Proceedings of the Fourth Annual ACM Symposium on Theory of Computing*, STOC ’72, page 238–250, New York, NY, USA, 1972. Association for Computing Machinery.
- [26] Ken Kennedy. Use-definition chains with applications. *Computer Languages*, 3(3):163–179, 1978.
- [27] Ken Kennedy. A global flow analysis algorithm. *International Journal of Computer Mathematics*, 3(1-4):5–15, 1972.
- [28] Ken Kennedy. Safety of code motion. *International Journal of Computer Mathematics*, 3(1-4):117–130, 1972.
- [29] D.J. Kuck, Y. Muraoka, and Shyh-Ching Chen. On the number of operations simultaneously executable in fortran-like programs and their resulting speedup. *IEEE Transactions on Computers*, C-21(12):1293–1310, 1972.
- [30] Robert Tarjan. Testing flow graph reducibility. In *Proceedings of the Fifth Annual ACM Symposium on Theory of Computing*, STOC ’73, page 96–107, New York, NY, USA, 1973. Association for Computing Machinery.
- [31] Gary A Kildall. A unified approach to global program optimization. In *Proceedings of the 1st annual ACM SIGACT-SIGPLAN symposium on Principles of programming languages*, pages 194–206, 1973.
- [32] Jeffrey D Ullman. Fast algorithms for the elimination of common subexpressions. *Acta Informatica*, 2(3):191–213, 1973.
- [33] Frances E. Allen. Interprocedural analysis and the information derived by it. In *Programming Methodology, 4th Informatik Symposium*, page 291–322, Berlin, Heidelberg, 1974. Springer-Verlag.
- [34] Robert A Freiburghouse. Register allocation via usage counts. *Communications of the ACM*, 17(11):638–642, 1974.
- [35] S. Rao Kosaraju. Analysis of structured programs. In *Proceedings of the Fifth Annual ACM Symposium on Theory of Computing*, STOC ’73, page 240–252, New York, NY, USA, 1973. Association for Computing Machinery.
- [36] M. S. Hecht and J. D. Ullman. Characterizations of reducible flow graphs. *J. ACM*, 21(3):367–375, July 1974.
- [37] Leslie Lamport. The parallel execution of do loops. *Commun. ACM*, 17(2):83–93, February 1974.
- [38] Frances E. Allen. Interprocedural data flow analysis. In Jack L. Rosenfeld, editor, *Information Processing, Proceedings of the 6th IFIP Congress 1974, Stockholm, Sweden, August 5-10, 1974*, pages 398–402. North-Holland, 1974.
- [39] Amelia Fong, John Kam, and Jeffrey Ullman. Application of lattice algebra to loop optimization. In *Proceedings of the 2nd ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages*, POPL ’75, page 1–9, New York, NY, USA, 1975. Association for Computing Machinery.
- [40] David B. Loveman and Ross A. Faneuf. Program optimization - theory and practice. *SIGPLAN Not.*,

REFERENCES

- 10(3):97–102, January 1975.
- [41] Shyh-Ching Chen and David J. Kuck. Time and parallel processor bounds for linear recurrence systems. *IEEE Transactions on Computers*, 100(7):701–717, 1975.
- [42] Matthew S Hecht and Jeffrey D Ullman. A simple algorithm for global data flow analysis problems. *SIAM Journal on Computing*, 4(4):519–532, 1975.
- [43] David B. Loveman. Program improvement by source to source transformation. In *Proceedings of the 3rd ACM SIGACT-SIGPLAN Symposium on Principles on Programming Languages*, POPL ’76, page 140–152, New York, NY, USA, 1976. Association for Computing Machinery.
- [44] F. E. Allen and J. Cocke. A program data flow analysis procedure. *Commun. ACM*, 19(3):137, March 1976.
- [45] Kenneth G Walter. Recursion analysis for compiler optimization. *Communications of the ACM*, 19(9):514–516, 1976.
- [46] A. V. Aho and S. C. Johnson. Optimal code generation for expression trees. *J. ACM*, 23(3):488–501, July 1976.
- [47] Susan L. Graham and Mark Wegman. A fast and usually linear algorithm for global flow analysis. *J. ACM*, 23(1):172–202, January 1976.
- [48] Alfred V Aho, Stephen C Johnson, and Jeffrey D Ullman. Code generation for expressions with common subexpressions. In *Proceedings of the 3rd ACM SIGACT-SIGPLAN symposium on Principles on programming languages*, pages 19–31, 1976.
- [49] John B Kam and Jeffrey D Ullman. Global data flow analysis and iterative algorithms. *Journal of the ACM (JACM)*, 23(1):158–171, 1976.
- [50] W.H. Harrison. Compiler analysis of the value ranges for variables. *IEEE Transactions on Software Engineering*, SE-3(3):243–250, 1977.
- [51] Lawrence T. Kou. On live-dead analysis for global data flow problems. *J. ACM*, 24(3):473–483, July 1977.
- [52] John H. Reif and Harry R. Lewis. Symbolic evaluation and the global value graph. In *Proceedings of the 4th ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages*, POPL ’77, page 104–118, New York, NY, USA, 1977. Association for Computing Machinery.
- [53] Barry K. Rosen. High-level data flow analysis. *Commun. ACM*, 20(10):712–724, October 1977.
- [54] Patrick Cousot and Radhia Cousot. Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Proceedings of the 4th ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages*, POPL ’77, page 238–252, New York, NY, USA, 1977. Association for Computing Machinery.
- [55] John Cocke and Ken Kennedy. An algorithm for reduction of operator strength. *Commun. ACM*, 20(11):850–856, November 1977.
- [56] Rod M Burstall and John Darlington. A transformation system for developing recursive programs. *Journal of the ACM (JACM)*, 24(1):44–67, 1977.
- [57] Guy L. Steele. Arithmetic shifting considered harmful. *SIGPLAN Not.*, 12(11):61–69, November 1977.
- [58] John B Kam and Jeffrey D Ullman. Monotone data flow analysis frameworks. *Acta informatica*, 7(3):305–317, 1977.
- [59] David B. Loveman. Program Improvement by Source-to-Source Transformation. *Journal of the ACM*, 24(1):121–145, January 1977.
- [60] Robert W. Scheifler. An analysis of inline substitution for a structured programming language. *Commun. ACM*, 20(9):647–654, September 1977.
- [61] S. Katz. Program optimization using invariants. *IEEE Transactions on Software Engineering*, SE-

REFERENCES

- 4(5):378–389, 1978.
- [62] R. Steven Glanville and Susan L. Graham. A new method for compiler code generation. In *Proceedings of the 5th ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages*, POPL '78, page 231–254, New York, NY, USA, 1978. Association for Computing Machinery.
- [63] Jeffrey M. Barth. A practical interprocedural data flow analysis algorithm. *Commun. ACM*, 21(9):724–736, September 1978.
- [64] Barry K. Rosen. Data flow analysis for procedural languages. *J. ACM*, 26(2):322–344, April 1979.
- [65] B.G. Ryder. Constructing the call graph of a program. *IEEE Transactions on Software Engineering*, SE-5(3):216–226, 1979.
- [66] William B Ackerman. Data flow languages. In *1979 International Workshop on Managing Requirements Knowledge (MARK)*, pages 1087–1095. IEEE, 1979.
- [67] Shyh-Ching Chen et al. Time and parallel processor bounds for fortran-like loops. *IEEE Transactions on Computers*, 100(9):660–670, 1979.
- [68] Jack J Dongarra and A_R Hinds. Unrolling loops in fortran. *Software: Practice and Experience*, 9(3):219–226, 1979.
- [69] Thomas Lengauer and Robert Endre Tarjan. A fast algorithm for finding dominators in a flowgraph. *ACM Trans. Program. Lang. Syst.*, 1(1):121–141, January 1979.
- [70] John P Banning. An efficient way to find the side effects of procedure calls and the aliases of variables. In *Proceedings of the 6th ACM SIGACT-SIGPLAN symposium on Principles of programming languages*, pages 29–41, 1979.
- [71] E. Morel and C. Renvoise. Global optimization by suppression of partial redundancies. *Commun. ACM*, 22(2):96–103, February 1979.
- [72] J Eugene Ball. Predicting the effects of optimization on a procedure body. *ACM SIGPLAN Notices*, 14(8):214–220, 1979.
- [73] M. Sharir. Structural analysis: A new approach to flow analysis in optimizing compilers. *Comput. Lang.*, 5(3–4):141–153, January 1980.
- [74] Jack W Davidson and Christopher W Fraser. The design and application of a retargetable peephole optimizer. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 2(2):191–202, 1980.
- [75] Jack B Dennis. Data flow supercomputers. *Computer*, 13(11):48–56, 1980.
- [76] Shlomit S. Pinter and Ron Y. Pinter. Program optimization and parallelization using idioms. In *Proceedings of the 18th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '91, page 79–92, New York, NY, USA, 1991. Association for Computing Machinery.
- [77] Padua, Kuck, and Lawrie. High-speed multiprocessors and compilation techniques. *IEEE Transactions on Computers*, 100(9):763–776, 1980.
- [78] Dhananjay M Dhamdhere and JR Isaac. A composite algorithm for strength reduction and code movement optimization. *International Journal of Computer & Information Sciences*, 9:243–273, 1980.
- [79] William E. Weihl. Interprocedural data flow analysis in the presence of pointers, procedure variables, and label variables. In *Proceedings of the 7th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '80, page 83–94, New York, NY, USA, 1980. Association for Computing Machinery.
- [80] Robert Shostak. Deciding linear inequalities by computing loop residues. *J. ACM*, 28(4):769–779, October 1981.
- [81] Eugene M. Myers. A precise inter-procedural data flow algorithm. In *Proceedings of the 8th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '81, page 219–230, New York, NY, USA, 1981. Association for Computing Machinery.

REFERENCES

- [82] Gregory J Chaitin, Marc A Auslander, Ashok K Chandra, John Cocke, Martin E Hopkins, and Peter W Markstein. Register allocation via coloring. *Computer languages*, 6(1):47–57, 1981.
- [83] Frances E Allen, John Cocke, and Ken Kennedy. Reduction of operator strength. *Program Flow Analysis*, pages 79–101, 1981.
- [84] D. J. Kuck, R. H. Kuhn, D. A. Padua, B. Leasure, and M. Wolfe. Dependence graphs and compiler optimizations. In *Proceedings of the 8th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '81, page 207–218, New York, NY, USA, 1981. Association for Computing Machinery.
- [85] Kuck and Lawrie. On the performance enhancement of paging systems through program analysis and transformations. *IEEE Transactions on Computers*, 100(5):341–356, 1981.
- [86] Andrew S. Tanenbaum, Hans van Staveren, and Johan W. Stevenson. Using peephole optimization on intermediate code. *ACM Trans. Program. Lang. Syst.*, 4(1):21–36, January 1982.
- [87] Thomas R. Gross and John L. Hennessy. Optimizing delayed branches. *SIGMICRO Newsl.*, 13(4):114–120, October 1982.
- [88] SM Joshi and Dhananjay M Dhamdhare. A composite hoisting-strength reduction transformation for global program optimization part i. *International Journal of Computer Mathematics*, 11(1):21–41, 1982.
- [89] Victoria Markstein, John Cocke, and Peter Markstein. Optimization of range checking. In *Proceedings of the 1982 SIGPLAN Symposium on Compiler Construction*, SIGPLAN '82, page 114–119, New York, NY, USA, 1982. Association for Computing Machinery.
- [90] Gregory J Chaitin. Register allocation and spilling via graph coloring. *ACM Sigplan Notices*, 17(6):98–101, 1982.
- [91] Stefan M Freudenberger, Jacob T Schwartz, and Micha Sharir. Experience with the setl optimizer. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 5(1):26–45, 1983.
- [92] John L. Hennessy and Thomas Gross. Postpass code optimization of pipeline constraints. *ACM Trans. Program. Lang. Syst.*, 5(3):422–448, July 1983.
- [93] John R Allen, Ken Kennedy, Carrie Porterfield, and Joe Warren. Conversion of control dependence to data dependence. In *Proceedings of the 10th ACM SIGACT-SIGPLAN symposium on Principles of programming languages*, pages 177–189, 1983.
- [94] Jack W. Davidson and Christopher W. Fraser. Code selection through object code optimization. *ACM Trans. Program. Lang. Syst.*, 6(4):505–526, October 1984.
- [95] Jack W Davidson and Christopher W Fraser. Register allocation and exhaustive peephole optimization. *Software: Practice and Experience*, 14(9):857–865, 1984.
- [96] Jack W Davidson and Christopher W Fraser. Automatic generation of peephole optimizations. In *Proceedings of the 1984 SIGPLAN symposium on Compiler construction*, pages 111–116, 1984.
- [97] David Callahan and Ken Kennedy. Analysis of interprocedural side effects in a parallel programming environment. In *International Conference on Supercomputing*, pages 138–171. Springer, 1987.
- [98] Mikhail A. Bulyonkov. Polyvariant mixed computation for analyzer programs. *Acta Informatica*, 21(5):473–484, 1984.
- [99] Allen Goldberg and Robert Paige. Stream processing. In *Proceedings of the 1984 ACM Symposium on LISP and Functional Programming*, LFP '84, page 53–62, New York, NY, USA, 1984. Association for Computing Machinery.
- [100] Joe Warren. A hierarchical basis for reordering transformations. In *Proceedings of the 11th ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages*, POPL '84, page 272–282, New York, NY, USA, 1984. Association for Computing Machinery.

REFERENCES

- [101] Frederick Chow and John Hennessy. Register allocation by priority-based coloring. *SIGPLAN Not.*, 19(6):222–232, June 1984.
- [102] John R Allen and Ken Kennedy. Automatic loop interchange. In *Proceedings of the 1984 SIGPLAN symposium on Compiler construction*, pages 233–246, 1984.
- [103] Keith D Cooper and Ken Kennedy. Efficient computation of flow insensitive interprocedural summary information. In *Proceedings of the 1984 SIGPLAN symposium on Compiler construction*, pages 247–258, 1984.
- [104] Flemming Nielson. Program transformations in a denotational setting. *ACM Trans. Program. Lang. Syst.*, 7(3):359–379, July 1985.
- [105] Jeanne Ferrante and Mary Mace. On linearizing parallel code. In *Proceedings of the 12th ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages*, POPL ’85, page 179–190, New York, NY, USA, 1985. Association for Computing Machinery.
- [106] Paul Hudak and Benjamin Goldberg. Distributed execution of functional programs using serial combinators. *IEEE Transactions on computers*, 100(10):881–891, 1985.
- [107] Chris Clack and Simon L Peyton Jones. Strictness analysis—a practical approach. In *Conference on Functional Programming Languages and Computer Architecture*, pages 35–49. Springer, 1985.
- [108] D Harel. A linear algorithm for finding dominators in flow graphs and related problems. In *Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing*, STOC ’85, page 185–194, New York, NY, USA, 1985. Association for Computing Machinery.
- [109] David A. Padua and Michael J. Wolfe. Advanced compiler optimizations for supercomputers. *Commun. ACM*, 29(12):1184–1201, December 1986.
- [110] Peter Harrison and Hessam Khoshnevisan. Efficient compilation of linear recursive functions into object level loops. In *Proceedings of the 1986 SIGPLAN Symposium on Compiler Construction*, SIGPLAN ’86, page 207–218, New York, NY, USA, 1986. Association for Computing Machinery.
- [111] Keith D. Cooper, Ken Kennedy, and Linda Torczon. The impact of interprocedural analysis and optimization in the rn programming environment. *ACM Trans. Program. Lang. Syst.*, 8(4):491–523, August 1986.
- [112] Philip B Gibbons and Steven S Muchnick. Efficient instruction scheduling for a pipelined architecture. In *Proceedings of the 1986 SIGPLAN symposium on Compiler construction*, pages 11–16, 1986.
- [113] John H. Reif and Harry R. Lewis. Efficient symbolic analysis of programs. *Journal of Computer and System Sciences*, 32(3):280–314, 1986.
- [114] Randal E. Bryant. Graph-based algorithms for boolean function manipulation. *IEEE Transactions on Computers*, C-35(8):677–691, 1986.
- [115] Michael Wolfe. Loops skewing: The wavefront method revisited. *International Journal of Parallel Programming*, 15(4):279–293, 1986.
- [116] P YT Hsu and Edward S Davidson. Highly concurrent scalar processing. *ACM SIGARCH Computer Architecture News*, 14(2):386–395, 1986.
- [117] Robert Bernstein. Multiplication by integer constants. *Software: practice and experience*, 16(7):641–652, 1986.
- [118] David W. Wall. Global register allocation at link time. In *Proceedings of the 1986 SIGPLAN Symposium on Compiler Construction*, SIGPLAN ’86, page 264–275, New York, NY, USA, 1986. Association for Computing Machinery.
- [119] David Callahan, Keith D Cooper, Ken Kennedy, and Linda Torczon. Interprocedural constant propagation. *ACM SIGPLAN Notices*, 21(7):152–161, 1986.
- [120] Keith D. Cooper, Ken Kennedy, and Linda Torczon. Interprocedural optimization: eliminating un-

REFERENCES

- necessary recompilation. In *Proceedings of the 1986 SIGPLAN Symposium on Compiler Construction*, SIGPLAN '86, page 58–67, New York, NY, USA, 1986. Association for Computing Machinery.
- [121] Michael Burke and Ron Cytron. Interprocedural dependence analysis and parallelization. *ACM Sigplan Notices*, 21(7):162–175, 1986.
 - [122] Mark S. Johnson and Terrence C. Miller. Effectiveness of a machine-level, global optimizer. In *Proceedings of the 1986 SIGPLAN Symposium on Compiler Construction*, SIGPLAN '86, page 99–108, New York, NY, USA, 1986. Association for Computing Machinery.
 - [123] Rémi Triolet, Francois Irigoin, and Paul Feautrier. Direct parallelization of call statements. *SIGPLAN Not.*, 21(7):176–185, July 1986.
 - [124] Ron Cytron, Andy Lowry, and F Kenneth Zadeck. Code motion of control structures in high-level languages. In *Proceedings of the 13th ACM SIGACT-SIGPLAN symposium on Principles of programming languages*, pages 70–85, 1986.
 - [125] Jack W Davidson and Christopher W Fraser. Automatic inference and fast interpretation of peephole optimization rules. *Software: Practice and Experience*, 17(11):801–812, 1987.
 - [126] Samuel P. Midkiff and David A. Padua. Compiler algorithms for synchronization. *IEEE Transactions on Computers*, C-36(12):1485–1495, 1987.
 - [127] r. Allen, D. Callahan, and K. Kennedy. Automatic decomposition of scientific programs for parallel execution. In *Proceedings of the 14th ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages*, POPL '87, page 63–76, New York, NY, USA, 1987. Association for Computing Machinery.
 - [128] Constantine D. Polychronopoulos and David J. Kuck. Guided self-scheduling: A practical scheduling scheme for parallel supercomputers. *IEEE Transactions on Computers*, C-36(12):1425–1439, 1987.
 - [129] Constantine D Polychronopoulos. Loop coalescing: A compiler transformation for parallel machines. Technical report, Illinois Univ., Urbana (USA), 1987.
 - [130] Dennis Gannon, William Jalby, and Kyle Gallivan. Strategies for cache and local memory management by global program transformation. In *International Conference on Supercomputing*, pages 229–254. Springer, 1987.
 - [131] Jeanne Ferrante, Karl J. Ottenstein, and Joe D. Warren. The program dependence graph and its use in optimization. *ACM Trans. Program. Lang. Syst.*, 9(3):319–349, July 1987.
 - [132] Randy Allen and Ken Kennedy. Automatic translation of fortran programs to vector form. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 9(4):491–542, 1987.
 - [133] D. R. Chase. Safety consideration for storage allocation optimizations. In *Proceedings of the ACM SIGPLAN 1988 Conference on Programming Language Design and Implementation*, PLDI '88, page 1–10, New York, NY, USA, 1988. Association for Computing Machinery.
 - [134] Dennis Shasha and Marc Snir. Efficient and correct execution of parallel programs that share memory. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 10(2):282–312, 1988.
 - [135] I. A. Natour. On the control dependence in the program dependence graph. In *Proceedings of the 1988 ACM Sixteenth Annual Conference on Computer Science*, CSC '88, page 510–519, New York, NY, USA, 1988. Association for Computing Machinery.
 - [136] D. E. Culler and Arvind. Resource requirements of dataflow programs. In *Proceedings of the 15th Annual International Symposium on Computer Architecture*, ISCA '88, page 141–150, Washington, DC, USA, 1988. IEEE Computer Society Press.
 - [137] M. D. Carroll and B. G. Ryder. Incremental data flow analysis via dominator and attribute update. In *Proceedings of the 15th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '88, page 274–284, New York, NY, USA, 1988. Association for Computing Machinery.
 - [138] S. Jain and C. Thompson. An efficient approach to data flow analysis in a multiple pass global

REFERENCES

- optimizer. *SIGPLAN Not.*, 23(7):154–163, June 1988.
- [139] Karl-Heinz Drechsler and Manfred P. Stadel. A solution to a problem with morel and renvoise’s “global optimization by suppression of partial redundancies”. *ACM Trans. Program. Lang. Syst.*, 10(4):635–640, October 1988.
- [140] Keith D Cooper and Ken Kennedy. Efficient computation of flow-insensitive interprocedural summary information—a correction. *ACM SIGPLAN Notices*, 23(4):35–42, 1988.
- [141] D. R. Wallace. Dependence of multi-dimensional array references. In *Proceedings of the 2nd International Conference on Supercomputing*, ICS ’88, page 418–428, New York, NY, USA, 1988. Association for Computing Machinery.
- [142] T. Brandes. The importance of direct dependences for automatic parallelization. In *Proceedings of the 2nd International Conference on Supercomputing*, ICS ’88, page 407–417, New York, NY, USA, 1988. Association for Computing Machinery.
- [143] A. Lichnewsky and F. Thomasset. Introducing symbolic problem solving techniques in the dependence testing phases of a vectorizer. In *Proceedings of the 2nd International Conference on Supercomputing*, ICS ’88, page 396–406, New York, NY, USA, 1988. Association for Computing Machinery.
- [144] J. Ferrante, M. Mace, and B. Simons. Generating sequential code from parallel code. In *Proceedings of the 2nd International Conference on Supercomputing*, ICS ’88, page 582–592, New York, NY, USA, 1988. Association for Computing Machinery.
- [145] H. Dietz and C.-H. Chi. Cregs: a new kind of memory for referencing arrays and pointers. In *Supercomputing ’88: Proceedings of the 1988 ACM/IEEE Conference on Supercomputing, Vol. I*, pages 360–367, 1988.
- [146] Constantine D. Polychronopoulos. Advanced loop optimizations for parallel computers. In E. N. Houstis, T. S. Papatheodorou, and C. D. Polychronopoulos, editors, *Supercomputing*, pages 255–277, Berlin, Heidelberg, 1988. Springer Berlin Heidelberg.
- [147] Utpal Banerjee. An introduction to a formal theory of dependence analysis. *The Journal of Supercomputing*, 2:133–149, 1988.
- [148] David Callahan and Ken Kennedy. Analysis of interprocedural side effects in a parallel programming environment. In E. N. Houstis, T. S. Papatheodorou, and C. D. Polychronopoulos, editors, *Supercomputing*, pages 138–171, Berlin, Heidelberg, 1988. Springer Berlin Heidelberg.
- [149] Paul Feautrier. Array expansion. In *ACM International Conference on Supercomputing 25th Anniversary Volume*, pages 99–111, 1988.
- [150] Alexandru Nicolau. Loop quantization: A generalized loop unwinding technique. *Journal of Parallel and Distributed Computing*, 5(5):568–586, 1988.
- [151] François Irigoin and Rémi Triolet. Supernode partitioning. In *Proceedings of the 15th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 319–329, 1988.
- [152] Dhananjay M Dhamdhere. A fast algorithm for code movement optimisation. *ACM SIGPLAN Notices*, 23(10):172–180, 1988.
- [153] David Callahan and Ken Kennedy. Compiling programs for distributed-memory multiprocessors. *The Journal of Supercomputing*, 2:151–169, 1988.
- [154] Alexander Aiken and Alexandru Nicolau. Optimal loop parallelization. *ACM SIGPLAN Notices*, 23(7):308–317, 1988.
- [155] M. Lam. Software pipelining: an effective scheduling technique for vliw machines. *SIGPLAN Not.*, 23(7):318–328, June 1988.
- [156] David Callahan. The program summary graph and flow-sensitive interprocedural data flow analysis. In *Proceedings of the ACM SIGPLAN 1988 conference on Programming Language design and Implemen-*

REFERENCES

- tation, pages 47–56, 1988.
- [157] Frances Allen, Michael Burke, Ron Cytron, Jeanne Ferrante, and Wilson Hsieh. A framework for determining useful parallelism. In *Proceedings of the 2nd international conference on Supercomputing*, pages 207–215, 1988.
 - [158] Frances Allen, Michael Burke, Philippe Charles, Ron Cytron, and Jeanne Ferrante. An overview of the ptran analysis system for multiprocessing. *Journal of parallel and distributed computing*, 5(5):617–640, 1988.
 - [159] Zhiyuan Li and Pen-Chung Yew. Efficient interprocedural analysis for program parallelization and restructuring. *SIGPLAN Not.*, 23(9):85–99, January 1988.
 - [160] M. Girkar and C. Polychronopoulos. Compiling issues for supercomputers. In *Supercomputing '88: Proceedings of the 1988 ACM/IEEE Conference on Supercomputing, Vol. I*, pages 164–173, 1988.
 - [161] Fred C Chow. Minimizing register usage penalty at procedure calls. In *Proceedings of the ACM SIGPLAN 1988 conference on Programming Language design and Implementation*, pages 85–94, 1988.
 - [162] David Callahan, John Cocke, and Ken Kennedy. Estimating interlock and improving balance for pipelined architectures. *Journal of Parallel and Distributed Computing*, 5(4):334–358, 1988.
 - [163] B. K. Rosen, M. N. Wegman, and F. K. Zadeck. Global value numbers and redundant computations. In *Proceedings of the 15th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL '88*, page 12–27, New York, NY, USA, 1988. Association for Computing Machinery.
 - [164] Alexander Aiken and Alexandru Nicolau. Perfect pipelining: A new loop parallelization technique. In *European Symposium on Programming*, pages 221–235. Springer, 1988.
 - [165] James R Goodman and W-C Hsu. Code scheduling and register allocation in large basic blocks. In *ACM International Conference on Supercomputing 25th Anniversary Volume*, pages 88–98, 1988.
 - [166] Keith D Cooper and Ken Kennedy. Interprocedural side-effect analysis in linear time. *ACM SIGPLAN Notices*, 23(7):57–66, 1988.
 - [167] V. Balasundaram and K. Kennedy. A technique for summarizing data access and its use in parallelism enhancing transformations. *SIGPLAN Not.*, 24(7):41–53, June 1989.
 - [168] Stephen Richardson and Mahadevan Ganapathi. Interprocedural analysis vs. procedure integration. *Information Processing Letters*, 32(3):137–142, 1989.
 - [169] C-H Chi and Hank Dietz. Unified management of registers and cache using liveness and cache bypass. In *Proceedings of the ACM SIGPLAN 1989 conference on Programming language design and implementation*, pages 344–353, 1989.
 - [170] Dhananjay M Dhamdhere. A new algorithm for composite hoisting and strength reduction optimization. *International Journal of Computer Mathematics*, 27(1):1–14, 1989.
 - [171] Mary Jean Harrold and Mary Lou Soffa. Interprocedural data flow testing. *SIGSOFT Softw. Eng. Notes*, 14(8):158–167, November 1989.
 - [172] S. McFarling. Program optimization for instruction caches. *SIGARCH Comput. Archit. News*, 17(2):183–191, April 1989.
 - [173] S. Horwitz, P. Pfeiffer, and T. Reps. Dependence analysis for pointer variables. *SIGPLAN Not.*, 24(7):28–40, June 1989.
 - [174] W. Baxter and H. R. Bauer. The program dependence graph and vectorization. In *Proceedings of the 16th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL '89*, page 1–11, New York, NY, USA, 1989. Association for Computing Machinery.
 - [175] W-m W Hwu and Pohua P Chang. Achieving high instruction cache performance with an optimizing compiler. In *Proceedings of the 16th Annual International Symposium on Computer Architecture*, pages 242–251, 1989.

REFERENCES

- [176] Susan J Eggers and Randy H Katz. Evaluating the performance of four snooping cache coherency protocols. In *Proceedings of the 16th annual international symposium on Computer architecture*, pages 2–15, 1989.
- [177] Guy E Blleloch. Scans as primitive parallel operations. *IEEE Transactions on computers*, 38(11):1526–1538, 1989.
- [178] Alfred V. Aho, Mahadevan Ganapathi, and Steven W. K. Tjiang. Code generation using tree matching and dynamic programming. *ACM Trans. Program. Lang. Syst.*, 11(4):491–516, October 1989.
- [179] R. Gupta, M. L. Soffa, and T. Steele. Register allocation via clique separators. In *Proceedings of the ACM SIGPLAN 1989 Conference on Programming Language Design and Implementation*, PLDI ’89, page 264–274, New York, NY, USA, 1989. Association for Computing Machinery.
- [180] K. D. Cooper and K. Kennedy. Fast interprocedural alias analysis. In *Proceedings of the 16th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL ’89, page 49–59, New York, NY, USA, 1989. Association for Computing Machinery.
- [181] P. Briggs, K. D. Cooper, K. Kennedy, and L. Torczon. Coloring heuristics for register allocation. In *Proceedings of the ACM SIGPLAN 1989 Conference on Programming Language Design and Implementation*, PLDI ’89, page 275–284, New York, NY, USA, 1989. Association for Computing Machinery.
- [182] M. Wolfe. More iteration space tiling. In *Proceedings of the 1989 ACM/IEEE Conference on Supercomputing*, Supercomputing ’89, page 655–664, New York, NY, USA, 1989. Association for Computing Machinery.
- [183] Zhiyuan Li, Pen-Chung Yew, and Chuag-Qi Zhu. Data dependence analysis on multi-dimensional array references. In *Proceedings of the 3rd International Conference on Supercomputing*, ICS ’89, page 215–224, New York, NY, USA, 1989. Association for Computing Machinery.
- [184] D. Bernstein, M. Golumbic, y. Mansour, R. Pinter, D. Goldin, H. Krawczyk, and I. Nahshon. Spill code minimization techniques for optimizing compilers. In *Proceedings of the ACM SIGPLAN 1989 Conference on Programming Language Design and Implementation*, PLDI ’89, page 258–263, New York, NY, USA, 1989. Association for Computing Machinery.
- [185] Craig Chambers and David Ungar. Customization: Optimizing compiler technology for self, a dynamically-typed object-oriented programming language. *ACM SIGPLAN Notices*, 24(7):146–160, 1989.
- [186] Ron Cytron, Jeanne Ferrante, Barry K Rosen, Mark N Wegman, and F Kenneth Zadeck. An efficient method of computing static single assignment form. In *Proceedings of the 16th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 25–35, 1989.
- [187] Bernhard Steffen, Jens Knoop, and Oliver Rüthing. The value flow graph: A program representation for optimal program transformations. In Neil Jones, editor, *ESOP ’90*, pages 389–405, Berlin, Heidelberg, 1990. Springer Berlin Heidelberg.
- [188] David E. Hudak and Santosh G. Abraham. Compiler techniques for data partitioning of sequentially iterated parallel loops. In *Proceedings of the 4th International Conference on Supercomputing*, ICS ’90, page 187–200, New York, NY, USA, 1990. Association for Computing Machinery.
- [189] Lars Hernquist. Vectorization of tree traversals. *Journal of Computational Physics*, 87(1):137–147, 1990.
- [190] R. Gupta. Loop displacement: an approach for transforming and scheduling loops for parallel execution. In *Supercomputing ’90: Proceedings of the 1990 ACM/IEEE Conference on Supercomputing*, pages 388–397, 1990.
- [191] Paul Havlak and Ken Kennedy. Experience with interprocedural analysis of array side effects. In *Proceedings of the 1990 ACM/IEEE Conference on Supercomputing*, Supercomputing ’90, page 952–961,

REFERENCES

- Washington, DC, USA, 1990. IEEE Computer Society Press.
- [192] D. 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 '90, page 137–146, New York, NY, USA, 1990. Association for Computing Machinery.
 - [193] Vatsa Santhanam and Daryl Odnert. Register allocation across procedure and module boundaries. *SIGPLAN Not.*, 25(6):28–39, June 1990.
 - [194] Rajiv Gupta and Mary Lou Soffa. Region scheduling: An approach for detecting and redistributing parallelism. *IEEE Trans. Softw. Eng.*, 16(4):421–431, April 1990.
 - [195] D. Callahan, A. Carle, M.W. Hall, and K. Kennedy. Constructing the procedure call multigraph. *IEEE Transactions on Software Engineering*, 16(4):483–487, 1990.
 - [196] David Klappholz, Kleanthis Psarris, and Xiangyun Kong. On the perfect accuracy of an approximate subscript analysis test. *ACM SIGARCH Computer Architecture News*, 18(3b):201–212, 1990.
 - [197] 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, 1990.
 - [198] Steven Anderson and Paul Hudak. Compilation of haskell array comprehensions for scientific computing. *ACM SIGPLAN Notices*, 25(6):137–149, 1990.
 - [199] William D. Clinger. How to read floating point numbers accurately. *SIGPLAN Not.*, 25(6):92–101, June 1990.
 - [200] Guy L. Steele and Jon L. White. How to print floating-point numbers accurately. *SIGPLAN Not.*, 25(6):112–126, June 1990.
 - [201] Karl Pettis and Robert C. Hansen. Profile guided code positioning. *SIGPLAN Not.*, 25(6):16–27, June 1990.
 - [202] Michael Gerndt. Updating distributed variables in local computations. *Concurrency: Practice and Experience*, 2(3):171–193, 1990.
 - [203] Michael Burke. An interval-based approach to exhaustive and incremental interprocedural data-flow analysis. *ACM Trans. Program. Lang. Syst.*, 12(3):341–395, July 1990.
 - [204] Fred C Chow and John L Hennessy. The priority-based coloring approach to register allocation. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 12(4):501–536, 1990.
 - [205] David Callahan, Steve Carr, and Ken Kennedy. Improving register allocation for subscripted variables. *ACM Sigplan Notices*, 25(6):53–65, 1990.
 - [206] David R. Chase, Mark Wegman, and F. Kenneth Zadeck. Analysis of pointers and structures. In *Proceedings of the ACM SIGPLAN 1990 Conference on Programming Language Design and Implementation*, PLDI '90, page 296–310, New York, NY, USA, 1990. Association for Computing Machinery.
 - [207] Ken Kennedy and Kathryn S McKinley. Loop distribution with arbitrary control flow. In *SC*, pages 407–416. Citeseer, 1990.
 - [208] Brian R. Nickerson. Graph coloring register allocation for processors with multi-register operands. *SIGPLAN Not.*, 25(6):40–52, June 1990.
 - [209] Michael E. Wolf and Monica S. Lam. A data locality optimizing algorithm. *SIGPLAN Not.*, 26(6):30–44, May 1991.
 - [210] Uwe Schwiegelshohn, Franco Gasperoni, and Kemal Ebcioglu. On optimal parallelization of arbitrary loops. *Journal of Parallel and Distributed Computing*, 11(2):130–134, 1991.
 - [211] Phil Pfeiffer and Rebecca Parsons Selke. On the adequacy of dependence-based representations for programs with heaps. In Takayasu Ito and Albert R. Meyer, editors, *Theoretical Aspects of Computer Software*, pages 365–386, Berlin, Heidelberg, 1991. Springer Berlin Heidelberg.
 - [212] Jong-Deok Choi, Ron Cytron, and Jeanne Ferrante. Automatic construction of sparse data flow

REFERENCES

- evaluation graphs. In *Proceedings of the 18th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '91, page 55–66, New York, NY, USA, 1991. Association for Computing Machinery.
- [213] Seema Hiranandani, Ken Kennedy, and Chau-Wen Tseng. Compiler optimizations for fortran d on mimd distributed-memory machines. In *Supercomputing '91: Proceedings of the 1991 ACM/IEEE Conference on Supercomputing*, pages 86–100, 1991.
- [214] David Callahan and Brian Koblenz. Register allocation via hierarchical graph coloring. *ACM Sigplan Notices*, 26(6):192–203, 1991.
- [215] Suneel Jain. Circular scheduling: a new technique to perform software pipelining. In *Proceedings of the ACM SIGPLAN 1991 Conference on Programming Language Design and Implementation*, PLDI '91, page 219–228, New York, NY, USA, 1991. Association for Computing Machinery.
- [216] Mark Smotherman, Sanjay Krishnamurthy, P. S. Aravind, and David Hunnicutt. Efficient dag construction and heuristic calculation for instruction scheduling. In *Proceedings of the 24th Annual International Symposium on Microarchitecture*, MICRO 24, page 93–102, New York, NY, USA, 1991. Association for Computing Machinery.
- [217] Ron Cytron, Jeanne Ferrante, Barry K. Rosen, Mark N. Wegman, and F. Kenneth Zadeck. Efficiently computing static single assignment form and the control dependence graph. *ACM Trans. Program. Lang. Syst.*, 13(4):451–490, October 1991.
- [218] David Callahan, Ken Kennedy, and Allan Porterfield. Software prefetching. In *Proceedings of the Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS IV, page 40–52, New York, NY, USA, 1991. Association for Computing Machinery.
- [219] C. Koelbel and P. Mehrotra. Compiling global name-space parallel loops for distributed execution. *IEEE Transactions on Parallel and Distributed Systems*, 2(4):440–451, 1991.
- [220] Paul Havlak and Ken Kennedy. An implementation of interprocedural bounded regular section analysis. *IEEE Transactions on Parallel & Distributed Systems*, 2(03):350–360, 1991.
- [221] David W. Wall. Limits of instruction-level parallelism. *SIGOPS Oper. Syst. Rev.*, 25(Special Issue):176–188, April 1991.
- [222] William Pugh. Uniform techniques for loop optimization. In *Proceedings of the 5th international conference on Supercomputing*, pages 341–352, 1991.
- [223] Mark N. Wegman and F. Kenneth Zadeck. Constant Propagation with Conditional Branches. *ACM Trans. Program. Lang. Syst.*, 13(2):181–210, April 1991.
- [224] Dror E. Maydan, John L. Hennessy, and Monica S. Lam. Efficient and exact data dependence analysis. *SIGPLAN Not.*, 26(6):1–14, May 1991.
- [225] Ron Cytron, Jeanne Ferrante, Barry K. Rosen, Mark N. Wegman, and F. Kenneth Zadeck. Efficiently computing static single assignment form and the control dependence graph. *ACM Trans. Program. Lang. Syst.*, 13(4):451–490, October 1991.
- [226] David Bernstein and Michael Rodeh. Global instruction scheduling for superscalar machines. *SIGPLAN Not.*, 26(6):241–255, May 1991.
- [227] Dhananjay M. Dhamdhare. Practical adaption of the global optimization algorithm of morel and renvoise. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 13(2):291–294, 1991.
- [228] Monica D. Lam, Edward E. Rothberg, and Michael E. Wolf. The cache performance and optimizations of blocked algorithms. In *Proceedings of the Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS IV, page 63–74, New York, NY, USA, 1991. Association for Computing Machinery.
- [229] Michael E Wolf and Monica S Lam. A loop transformation theory and an algorithm to maximize

REFERENCES

- parallelism. *IEEE Transactions on Parallel & Distributed Systems*, 2(04):452–471, 1991.
- [230] Paul Feautrier. Dataflow analysis of array and scalar references. *International Journal of Parallel Programming*, 20:23–53, 1991.
 - [231] Vivek Sarkar and Guang R. Gao. Optimization of array accesses by collective loop transformations. In *Proceedings of the 5th International Conference on Supercomputing*, ICS '91, page 194–205, New York, NY, USA, 1991. Association for Computing Machinery.
 - [232] Mary W Hall, Ken Kennedy, and Kathryn S McKinley. Interprocedural transformations for parallel code generation. In *Proceedings of the 1991 ACM/IEEE conference on Supercomputing*, pages 424–434, 1991.
 - [233] Gina Goff, Ken Kennedy, and Chau-Wen Tseng. Practical dependence testing. *ACM SIGPLAN Notices*, 26(6):15–29, 1991.
 - [234] Scott McFarling. Procedure merging with instruction caches. *ACM SIGPLAN Notices*, 26(6):71–79, 1991.
 - [235] Keith D Cooper, Mary W Hall, and Linda Torczon. An experiment with inline substitution. *Software: Practice and Experience*, 21(6):581–601, 1991.
 - [236] J. Ramanujam. Non-unimodular transformations of nested loops. In *Supercomputing '92: Proceedings of the 1992 ACM/IEEE Conference on Supercomputing*, pages 214–223, 1992.
 - [237] Vivek Sarkar and Radhika Thekkath. A general framework for iteration-reordering loop transformations. In *Proceedings of the ACM SIGPLAN 1992 conference on Programming language design and implementation*, pages 175–187, 1992.
 - [238] Wu Yang, Susan Horwitz, and Thomas Reps. A program integration algorithm that accommodates semantics-preserving transformations. *ACM Trans. Softw. Eng. Methodol.*, 1(3):310–354, July 1992.
 - [239] Michael Weiss. The transitive closure of control dependence: The iterated join. *ACM Letters on Programming Languages and Systems (LOPLAS)*, 1(2):178–190, 1992.
 - [240] Joseph Hummel, Laurie J. Hendren, and Alexandru Nicolau. A general data dependence test for dynamic, pointer-based data structures. In *Proceedings of the ACM SIGPLAN 1994 Conference on Programming Language Design and Implementation*, PLDI '94, page 218–229, New York, NY, USA, 1994. Association for Computing Machinery.
 - [241] Reinhard v. Hanxleden and Ken Kennedy. Relaxing simd control flow constraints using loop transformations. *SIGPLAN Not.*, 27(7):188–199, July 1992.
 - [242] Rajiv Gupta. Generalized dominators and post-dominators. In *Proceedings of the 19th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '92, page 246–257, New York, NY, USA, 1992. Association for Computing Machinery.
 - [243] Yong-fong Lee and Barbara G Ryder. A comprehensive approach to parallel data flow analysis. In *Proceedings of the 6th International Conference on Supercomputing*, pages 236–247, 1992.
 - [244] S. Tjiang, M. Wolf, M. Lam, K. Pieper, and J. Hennessy. Integrating scalar optimization and parallelization. In Utpal Banerjee, David Gelernter, Alex Nicolau, and David Padua, editors, *Languages and Compilers for Parallel Computing*, pages 137–151, Berlin, Heidelberg, 1992. Springer Berlin Heidelberg.
 - [245] Steven W. K. Tjiang and John L. Hennessy. Sharlit—a tool for building optimizers. *SIGPLAN Not.*, 27(7):82–93, July 1992.
 - [246] Dhananjay M Dhamdhere, Barry K Rosen, and F Kenneth Zadeck. How to analyze large programs efficiently and informatively. In *Proceedings of the ACM SIGPLAN 1992 conference on Programming language design and implementation*, pages 212–223, 1992.
 - [247] S. A. Mahlke, W. Y. Chen, J. C. Gyllenhaal, and W.-M. W. Hwu. Compiler code transformations for

REFERENCES

- superscalar-based high performance systems. In *Proceedings of the 1992 ACM/IEEE Conference on Supercomputing*, Supercomputing '92, page 808–817, Washington, DC, USA, 1992. IEEE Computer Society Press.
- [248] Christopher W. Fraser, David R. Hanson, and Todd A. Proebsting. Engineering a simple, efficient code-generator generator. *ACM Lett. Program. Lang. Syst.*, 1(3):213–226, September 1992.
 - [249] Paul Feautrier. Some efficient solutions to the affine scheduling problem. i. one-dimensional time. *International journal of parallel programming*, 21:313–347, 1992.
 - [250] Frank Mueller and David B. Whalley. Avoiding unconditional jumps by code replication. In *Proceedings of the ACM SIGPLAN 1992 Conference on Programming Language Design and Implementation*, PLDI '92, page 322–330, New York, NY, USA, 1992. Association for Computing Machinery.
 - [251] Preston Briggs, Keith D. Cooper, and Linda Torczon. Rematerialization. *SIGPLAN Not.*, 27(7):311–321, July 1992.
 - [252] Preston Briggs, Keith D Cooper, and Linda Torczon. Coloring register pairs. *ACM Letters on Programming Languages and Systems (LOPLAS)*, 1(1):3–13, 1992.
 - [253] Ken Kennedy and Kathryn S McKinley. Optimizing for parallelism and data locality. In *Proceedings of the 6th international conference on Supercomputing*, pages 323–334, 1992.
 - [254] Ken Kennedy and Kathryn S McKinley. Maximizing loop parallelism and improving data locality via loop fusion and distribution. In *International Workshop on Languages and Compilers for Parallel Computing*, pages 301–320. Springer, 1993.
 - [255] Michael Wolfe. Beyond induction variables. *SIGPLAN Not.*, 27(7):162–174, July 1992.
 - [256] Mary W. Hall and Ken Kennedy. Efficient call graph analysis. *ACM Lett. Program. Lang. Syst.*, 1(3):227–242, September 1992.
 - [257] William Landi and Barbara G. Ryder. A safe approximate algorithm for interprocedural aliasing. In *Proceedings of the ACM SIGPLAN 1992 Conference on Programming Language Design and Implementation*, PLDI '92, page 235–248, New York, NY, USA, 1992. Association for Computing Machinery.
 - [258] Keith D. Cooper, Mary W. Hall, and Linda Torczon. Unexpected side effects of inline substitution: a case study. *ACM Lett. Program. Lang. Syst.*, 1(1):22–32, March 1992.
 - [259] William Pugh and David Wonnacott. Eliminating false data dependences using the omega test. *SIGPLAN Not.*, 27(7):140–151, July 1992.
 - [260] Jens Knoop, Oliver Rüthing, and Bernhard Steffen. Lazy code motion. *SIGPLAN Not.*, 27(7):224–234, July 1992.
 - [261] Anne Rogers and Kai Li. Software support for speculative loads. In *Proceedings of the Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS V, page 38–50, New York, NY, USA, 1992. Association for Computing Machinery.
 - [262] William Pugh. A practical algorithm for exact array dependence analysis. *Commun. ACM*, 35(8):102–114, August 1992.
 - [263] Zhiyuan Li. Array privatization for parallel execution of loops. In *Proceedings of the 6th International Conference on Supercomputing*, ICS '92, page 313–322, New York, NY, USA, 1992. Association for Computing Machinery.
 - [264] Michael Wolfe, Chau-Wen Tseng, et al. The power test for data dependence. *IEEE Transactions on Parallel and Distributed Systems*, 3(5):591–601, 1992.
 - [265] Torbjörn Granlund and Richard Kenner. Eliminating branches using a superoptimizer and the gnu c compiler. In *Proceedings of the ACM SIGPLAN 1992 conference on Programming language design and implementation*, pages 341–352, 1992.
 - [266] Todd C. Mowry, Monica S. Lam, and Anoop Gupta. Design and evaluation of a compiler algorithm

REFERENCES

- for prefetching. *SIGPLAN Not.*, 27(9):62–73, September 1992.
- [267] D. Maydan, S. Amarsinghe, and M. Lam. Data dependence and data-flow analysis of arrays. In Utpal Banerjee, David Gelernter, Alex Nicolau, and David Padua, editors, *Languages and Compilers for Parallel Computing*, pages 434–448, Berlin, Heidelberg, 1993. Springer Berlin Heidelberg.
 - [268] A. J. C. Bik and H. A. G. Wijshoff. Advanced compiler optimizations for sparse computations. In *Proceedings of the 1993 ACM/IEEE Conference on Supercomputing*, Supercomputing '93, page 430–439, New York, NY, USA, 1993. Association for Computing Machinery.
 - [269] Wei Li and Keshav Pingali. Access normalization: loop restructuring for numa computers. *ACM Trans. Comput. Syst.*, 11(4):353–375, November 1993.
 - [270] Owen Kaser, C. R. Ramakrishnan, and Shaunak Pawagi. On the conversion of indirect to direct recursion. *ACM Lett. Program. Lang. Syst.*, 2(1–4):151–164, March 1993.
 - [271] Evelyn Duesterwald, Rajiv Gupta, and Mary Lou Soffa. A practical data flow framework for array reference analysis and its use in optimizations. In *Proceedings of the ACM SIGPLAN 1993 Conference on Programming Language Design and Implementation*, PLDI '93, page 68–77, New York, NY, USA, 1993. Association for Computing Machinery.
 - [272] David A. Barrett and Benjamin G. Zorn. Using lifetime predictors to improve memory allocation performance. *SIGPLAN Not.*, 28(6):187–196, June 1993.
 - [273] M. Robert Ito and A. Zaafrani. Data flow analysis for parallel programs. In *Proceedings of the 1993 ACM Conference on Computer Science*, CSC '93, page 318–325, New York, NY, USA, 1993. Association for Computing Machinery.
 - [274] Richard Johnson and Keshav Pingali. Dependence-based program analysis. In *Proceedings of the ACM SIGPLAN 1993 Conference on Programming Language Design and Implementation*, PLDI '93, page 78–89, New York, NY, USA, 1993. Association for Computing Machinery.
 - [275] Karl-Heinz Drechsler and Manfred P Stadel. A variation of knoop, rüthing, and steffen's lazy code motion. *ACM SIGPLAN Notices*, 28(5):29–38, 1993.
 - [276] Jens Knoop. Lazy strength reduction. *International Journal of Programming Languages*, 1(1):71–91, 1993.
 - [277] G. Gao, R. Olsen, V. Sarkar, and R. Thekkath. Collective loop fusion for array contraction. In Utpal Banerjee, David Gelernter, Alex Nicolau, and David Padua, editors, *Languages and Compilers for Parallel Computing*, pages 281–295, Berlin, Heidelberg, 1993. Springer Berlin Heidelberg.
 - [278] Robert Metzger and Sean Stroud. Interprocedural constant propagation: an empirical study. *ACM Lett. Program. Lang. Syst.*, 2(1–4):213–232, March 1993.
 - [279] Mohammad R Haghighat and Constantine D Polychronopoulos. Symbolic analysis: A basis for parallelization, optimization, and scheduling of programs. In *International Workshop on Languages and Compilers for Parallel Computing*, pages 567–585. Springer, 1993.
 - [280] Dror E. Maydan, Saman P. Amarasinghe, and Monica S. Lam. Array-data flow analysis and its use in array privatization. In *Proceedings of the 20th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '93, page 2–15, New York, NY, USA, 1993. Association for Computing Machinery.
 - [281] Siddhartha Chatterjee, John R Gilbert, Robert Schreiber, and Shang-Hua Teng. Automatic array alignment in data-parallel programs. In *Proceedings of the 20th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 16–28, 1993.
 - [282] Jennifer M Anderson and Monica S Lam. Global optimizations for parallelism and locality on scalable parallel machines. *ACM Sigplan Notices*, 28(6):112–125, 1993.
 - [283] B. Ramakrishna Rau and Joseph A. Fisher. *Instruction-Level Parallel Processing: History, Overview,*

REFERENCES

- and Perspective*, pages 9–50. Springer US, Boston, MA, 1993.
- [284] Amitabh Srivastava. A practical system for intermodule code optimization at link-time. *Journal of programming Languages*, 1(1):1–18, 1993.
 - [285] James R Larus. Loop-level parallelism in numeric and symbolic programs. *IEEE Transactions on Parallel and Distributed Systems*, 4(7):812–826, 1993.
 - [286] Susan L. Graham, Steven Lucco, and Oliver Sharp. Orchestrating interactions among parallel computations. *SIGPLAN Not.*, 28(6):100–111, June 1993.
 - [287] Wen-Mei W Hwu, Scott A Mahlke, William Y Chen, Pohua P Chang, Nancy J Warter, Roger A Bringmann, Roland G Ouellette, Richard E Hank, Tokuzo Kiyohara, Grant E Haab, et al. The superblock: an effective technique for vliw and superscalar compilation. *The Journal of Supercomputing*, 7(1-2):229–248, 1993.
 - [288] Rajiv Gupta. Optimizing array bound checks using flow analysis. *ACM Lett. Program. Lang. Syst.*, 2(1-4):135–150, March 1993.
 - [289] Shlomit S. Pinter. Register allocation with instruction scheduling. *SIGPLAN Not.*, 28(6):248–257, June 1993.
 - [290] Keith D Cooper, Mary W Hall, and Ken Kennedy. A methodology for procedure cloning. *Computer Languages*, 19(2):105–117, 1993.
 - [291] Jong-Deok Choi, Michael Burke, and Paul Carini. Efficient flow-sensitive interprocedural computation of pointer-induced aliases and side effects. In *Proceedings of the 20th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 232–245, 1993.
 - [292] Peng Tu and David Padua. Automatic array privatization. In *International Workshop on Languages and Compilers for Parallel Computing*, pages 500–521. Springer, 1993.
 - [293] William Landi, Barbara G Ryder, and Sean Zhang. Interprocedural modification side effect analysis with pointer aliasing. *ACM SIGPLAN Notices*, 28(6):56–67, 1993.
 - [294] Thomas Ball and James R Larus. Branch prediction for free. *ACM SIGPLAN Notices*, 28(6):300–313, 1993.
 - [295] R. Sass and M. Mutka. Enabling unimodular transformations. In *Supercomputing '94: Proceedings of the 1994 ACM/IEEE Conference on Supercomputing*, pages 753–762, 1994.
 - [296] Steve Carr, Kathryn S. McKinley, and Chau-Wen Tseng. Compiler optimizations for improving data locality. *SIGPLAN Not.*, 29(11):252–262, November 1994.
 - [297] Lawrence Feigen, David Klappholz, Robert Casazza, and Xing Xue. The revival transformation. In *Proceedings of the 21st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '94, page 421–434, New York, NY, USA, 1994. Association for Computing Machinery.
 - [298] William Blume and Rudolf Eigenmann. The range test: a dependence test for symbolic, non-linear expressions. In *Proceedings of the 1994 ACM/IEEE Conference on Supercomputing*, Supercomputing '94, page 528–537, Washington, DC, USA, 1994. IEEE Computer Society Press.
 - [299] Amitabh Srivastava and David W. Wall. Link-time optimization of address calculation on a 64-bit architecture. In *Proceedings of the ACM SIGPLAN 1994 Conference on Programming Language Design and Implementation*, PLDI '94, page 49–60, New York, NY, USA, 1994. Association for Computing Machinery.
 - [300] Preston Briggs and Keith D Cooper. Effective partial redundancy elimination. *ACM SIGPLAN Notices*, 29(6):159–170, 1994.
 - [301] Jens Knoop, Oliver Rüthing, and Bernhard Steffen. Partial dead code elimination. *ACM Sigplan Notices*, 29(6):147–158, 1994.
 - [302] Uma Mahadevan and Sridhar Ramakrishnan. Instruction scheduling over regions: A framework for

REFERENCES

- scheduling across basic blocks. In Peter A. Fritzson, editor, *Compiler Construction*, pages 419–434, Berlin, Heidelberg, 1994. Springer Berlin Heidelberg.
- [303] Daniel Weise, Roger F. Crew, Michael Ernst, and Bjarne Steensgaard. Value dependence graphs: representation without taxation. In *Proceedings of the 21st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '94, page 297–310, New York, NY, USA, 1994. Association for Computing Machinery.
- [304] Steven M. Kurlander and Charles N. Fischer. Zero-cost range splitting. *SIGPLAN Not.*, 29(6):257–265, June 1994.
- [305] Brad Calder and Dirk Grunwald. Reducing branch costs via branch alignment. *SIGOPS Oper. Syst. Rev.*, 28(5):242–251, November 1994.
- [306] Steve Carr and Ken Kennedy. Improving the ratio of memory operations to floating-point operations in loops. *ACM Trans. Program. Lang. Syst.*, 16(6):1768–1810, November 1994.
- [307] Eric Amiel, Olivier Gruber, and Eric Simon. Optimizing multi-method dispatch using compressed dispatch tables. *SIGPLAN Not.*, 29(10):244–258, October 1994.
- [308] Cliff Young and Michael D. Smith. Improving the accuracy of static branch prediction using branch correlation. *SIGOPS Oper. Syst. Rev.*, 28(5):232–241, November 1994.
- [309] David F. Bacon, Jyh-Herng Chow, Dz-Ching Ju, Kalyan Muthukumar, and Vivek Sarkar. A compiler framework for restructuring data declarations to enhance cache and TLB effectiveness. In John E. Botsford, Ann Gawman, W. Morven Gentleman, Evelyn Kidd, Kelly A. Lyons, Jacob Slonim, and J. Howard Johnson, editors, *Proceedings of the 1994 Conference of the Centre for Advanced Studies on Collaborative Research, October 31 - November 3, 1994, Toronto, Ontario, Canada*, page 3. IBM, 1994.
- [310] J. Torrellas, H.S. Lam, and J.L. Hennessy. False sharing and spatial locality in multiprocessor caches. *IEEE Transactions on Computers*, 43(6):651–663, 1994.
- [311] Peter W Markstein, Victoria Markstein, and F Kenneth Zadeck. Reassociation and strength reduction. *Optimization in Compilers*, 1994.
- [312] Siddhartha Chatterjee, John R Gilbert, and Robert Schreiber. The alignment-distribution graph. In *Languages and Compilers for Parallel Computing: 6th International Workshop Portland, Oregon, USA, August 12–14, 1993 Proceedings 6*, pages 234–252. Springer, 1994.
- [313] Preston Briggs, Keith D. Cooper, and Linda Torczon. Improvements to graph coloring register allocation. *ACM Trans. Program. Lang. Syst.*, 16(3):428–455, May 1994.
- [314] Maryam Emami, Rakesh Ghiya, and Laurie J. Hendren. Context-sensitive interprocedural points-to analysis in the presence of function pointers. In *Proceedings of the ACM SIGPLAN 1994 Conference on Programming Language Design and Implementation*, PLDI '94, page 242–256, New York, NY, USA, 1994. Association for Computing Machinery.
- [315] Alain Deutsch. Interprocedural may-alias analysis for pointers: beyond k-limiting. In *Proceedings of the ACM SIGPLAN 1994 Conference on Programming Language Design and Implementation*, PLDI '94, page 230–241, New York, NY, USA, 1994. Association for Computing Machinery.
- [316] Steve Carr and Ken Kennedy. Scalar replacement in the presence of conditional control flow. *Softw. Pract. Exper.*, 24(1):51–77, January 1994.