

# Bibliography

- [Bal88] BAL, H. E., AND TANENBAUM, A. S. 1988. Distributed programming with shared data. In *Proceedings of the IEEE International Conference on Computer Languages* (Miami), pp. 82-91.
- [Bal89] BAL, H. E., STEINER, J. G., AND TANENBAUM, A. S. 1989. Programming languages for distributed computing systems. In *ACM Computing Surveys* 21, 3, pp. 261-322.
- [Benker89] BENKER, H. ET AL. 1989. The knowledge crunching machine at ECRC: A joint R&D project of a high speed prolog system. In *ICL Technical Journal*, Nov. 1989, pp. 737-753.
- [Bernstein81] BERNSTEIN, P. A., AND GOODMAN, N. 1981. Concurrency control in distributed database systems. In *Computing Surveys* 13, 2, pp 185-221.
- [Bowen81] BOWEN, D. L., BYRD, L., PEREIRA, L. M., PEREIRA, F. C. N., AND WARREN, D. H. D. 1981. PROLOG on the DECSYSTEM-10 user's manual. Technical Report, Dept. of Artificial Intelligence, University of Edinburgh, Scotland.
- [Clark84] CLARK, K. L., AND GREGORY, S. 1984. PARLOG: Parallel programming in logic. Research Report DOC 84/4, Dept. of Computing, Imperial College of Science and Technology, London.
- [Clocksin84] CLOCKSIN, W. F., AND MELLISH, C. S. 1984. *Programming in Prolog*. Springer-Verlag, Berlin.
- [Coffman71] COFFMAN, E. G., ELPHICK, M. J., AND SHOSHANI, A. 1971. System deadlocks. In *Computing Surveys* 3, 2, pp. 67-78.
- [Conery87] CONERY, J. S. 1987. *Parallel Execution of Logic Programs*. Kluwer Academic Publishers, Boston.
- [Dijkstra75] DIJKSTRA, E. W. 1975. Guarded commands, nondeterminacy and formal derivation of programs. In *Communications of the ACM* 18, 8, pp. 453-457.

- [Dijkstra83] DIJKSTRA, E. W., FEIJEN, W. H. J., AND VAN GASTEREN, A. J. M. 1983. Derivation of a termination detection algorithm for distributed computations., In *Information Processing Letters* 16, 5, pp. 217-219.
- [vanEmden76] VAN EMDEN, M. H., AND KOWALSKI, R. A. 1976. The semantics of predicate logic as a programming language. In *Journal of the ACM* 23, 4, pp. 733-742.
- [vanEmden82] VAN EMDEN, M. H., AND DE LUCENA FILHO, G. J. 1982. Predicate logic as a language for parallel programming. In *Logic Programming*, K. L. Clark, and S.-A. Tärnlund, Eds. Academic Press, London, pp. 189-198.
- [Flynn66] FLYNN, M. J. 1966. Very high-speed computing systems. In *Proceedings of the IEEE* 54, 12, pp. 1901-1909.
- [Foster87] FOSTER, I., AND TAYLOR, S. 1987. Flat Parlog: A basis for comparison. In *International Journal of Parallel Programming* 16, 2, pp. 87-125.
- [Foster88] FOSTER, I. 1988. Parallel implementation of Parlog. In *Proceedings of the International Conference on Parallel Processing* (St. Charles), Ill., Vol. II, pp. 9-16.
- [Foster89] FOSTER, I. 1989. A multicomputer garbage collector for a single assignment language. In *International Journal of Parallel Programming* 18, 3, pp. 181-203.
- [Foster90] FOSTER, I., AND TAYLOR, S. 1990. *Strand, New Concepts in Parallel Programming*. Prentice-Hall, Englewood Cliffs, New Jersey.
- [Fuchi86] FUCHI, K., AND FURUKAWA, K. 1987. The role of logic programming in the Fifth Generation Computer Project. In *New Generation Computing* 5, 1, pp. 3-28.
- [Funke92] FUNKE, R. ET AL. 1992. An optimized reconfigurable architecture for Transputer networks. In *Proceedings of the 25th Hawaii International Conference on Computer Sciences* (Hawaii), pp. 237-245.
- [Gray92] GRAY, R. W. ET AL. 1992. Eli: A complete compiler construction system. In *Communications of the ACM* 35, 2, pp. 121-131.
- [Gregory87] GREGORY, S. 1987. *Parallel Logic Programming in PARLOG: The Language and Its Implementation*. Addison-Wesley Publishing Company, Wokingham, England.

- [Glaesser90a] GLÄSSER, U., KÄRCHER, M., LEHRENFELD, G., AND VIETH, N. 1990A. Flat Concurrent Prolog on Transputers. In *Proceedings of the IFIP Working Conference on Decentralized Systems* (Lyon), C. Girault and M. Cosnard, Eds. North-Holland, Amsterdam, pp. 183-194.
- [Glaesser90b] GLÄSSER, U., KÄRCHER, M., LEHRENFELD, G., AND VIETH, N. 1990B. Flat Concurrent Prolog on Transputers. In *Journal of Microcomputer Applications: Special Issue on Transputer Applications 13, 1*, pp. 3-18 (extended version of [Glaesser90a]).
- [Glaesser90c] GLÄSSER, U., AND LEHRENFELD, G. 1990C. A distributed implementation of Flat Concurrent Prolog on Transputer architectures. In *Proceedings of the UNESCO Conference on Parallel Computing in Engineering and Engineering Education* (Paris), pp. 181-185.
- [Glaesser91a] GLÄSSER, U., KÄRCHER, M., AND LEHRENFELD, G. 1991. Dynamische Partitionierung asynchroner Prozeßnetzwerke am Beispiel Paralleler Logischer Programmierung. To appear in *Proceedings of the TAT '91* (Aachen, FRG, Sep. 17-18, 1991), Informatik-Fachberichte, Springer-Verlag, Berlin.
- [Glaesser91b] GLÄSSER, U., HANNESEN, G., KÄRCHER, M., AND LEHRENFELD, G. 1991. A distributed implementation of Flat Concurrent Prolog on multi-Transputer environments. To appear in *Proceedings of the First International Conference of the Austrian Center for Parallel Computation* (Salzburg, Sep. 29 - Oct. 02, 1991), Lecture Notes in Computer Science, Springer-Verlag, Berlin.
- [Glaesser92] GLÄSSER, U. 1992. A distributed implementation of Flat Concurrent Prolog on multi-Transputer environments. In *Distributed Prolog*, P. Kacsuk and M. Wise, Eds. John Wiley & Sons Ltd., Chichester, pp. 287-309.
- [van de Goor89] VAN DE GOOR, A. J. 1989. *Computer Architecture and Design*. Addison-Wesely, New York.
- [Grunzig92] GRUNZIG, P. 1992. Konzepte zur Hardware-Realisierung der Reduktionseinheit einer parallelen FCP-Maschine (Diplomarbeit). Paderborn University, Dept. of Mathematics & Computer Science, Paderborn, FRG, Feb. 1992.
- [Hannesen91] HANNESEN, G. 1991. Implementierung und Optimierung von sequentiellen und parallelen FCP-Maschinen (Diplomarbeit). Paderborn University, Dept. of Mathematics & Computer Science, Paderborn, FRG, June 1991.
- [Harel85] HAREL, D., AND PNUELI, A. 1985. On the development of reactive systems. In *Logics and Models of Concurrent Systems*. K. R. Apt, Ed. Lecture Notes in Computer Science, Springer-Verlag, Berlin.

- [Hoare78] HOARE, C. A. R. 1978. Communicating sequential processes. In *Communications of the ACM* 21, 8, pp. 666-677.
- [Houris87] HOURI, A., AND SHAPIRO, E. 1987. A sequential abstract machine for Flat Concurrent Prolog. In *Concurrent Prolog: Collected Papers*. Vol. 2, E. Shapiro, Ed. MIT Press, Cambridge, Mass., pp. 513-574.
- [Ichiyoshi87] ICHIYOSHI, N., MIYAZAKI, T., AND TAKI, K. 1987. A distributed implementation of Flat GHC on the Multi-PSI. In *Logic Programming - Proceedings of the Fourth International Conference on Logic Programming* (Melbourne), J.-L. Lassez, Ed. MIT Press Cambridge, pp. 257-275.
- [INMOS91] INMOS LTD. 1991. *The T9000 Transputer Products Overview Manual*. INMOS Databook series, Marlow, UK.
- [Kaercher92] KÄRCHER, M. 1992. Automatische Parallelisierung am Beispiel eines verteilten FCP-Interpreters (Diplomarbeit). Paderborn University, Dept. of Mathematics & Computer Science, Paderborn, FRG (available before July, 1992).
- [Kleine Buening86] KLEINE BÜNING, H., AND SCHMITGEN, S. 1986. *PROLOG*. B. G. Teubner Stuttgart.
- [Kliger88] KLIGER, S., YARDENI, E., KAHN, K., AND SHAPIRO, E. 1988. The language FCP(:,?). In *Proceedings of the International Conference on Fifth Generation Computer Systems* (Tokyo), Ohmsha Ltd. Tokyo, Springer-Verlag, Berlin, pp. 763-773.
- [Kowalski79a] KOWALSKI, R. 1979A. Algorithm = logic + control. In *Communications of the ACM* 22, 7, pp. 424-436.
- [Kowalski79b] KOWALSKI, R. 1979B. *Logic for Problem Solving*. North-Holland, Amsterdam.
- [Kurfess91] KURFESS, F. 1991. *Parallelism in Logic: Its Potential for Performance and Program Development*. Verlag Vieweg, Braunschweig, FRG.
- [Lehrenfeld90] LEHRENFELD, G. 1990. Konzeption und Implementierung einer parallelen FCP-Maschine (Diplomarbeit). Paderborn University, Dept. of Mathematics & Computer Science, Paderborn, FRG, Dec. 1990.
- [Leighton92] LEIGHTON, F. T. 1992. *Introduction to Parallel Algorithms and Architectures: Arrays o Trees o Hypercubes*. Morgan Kaufmann Publishers, San Mateo, Calif.
- [Lueling91] LÜLING, R., MONIEN, B., AND RAMME, F. 1991. Load balancing in large networks: A comparative study. In *Proceedings of the 3rd IEEE Symposium on Parallel and Distributed Processing* (Dallas), pp. 686-689.

- [Lueling92] LÜLING, R., AND MONIEN, B. 1992. Load balancing for distributed branch & bound algorithms. In *Proceedings of the 6th International Parallel Processing Symposium* (Beverly Hills), pp. 543-549.
- [May85] MAY, D., AND SHEPHERD, R. 1985. Occam and the Transputer. In *Concurrent Languages in distributed Systems*. G. L. Reijens, and E. L. Dagless, Eds. North-Holland, Amsterdam, pp. 19-33.
- [May90] MAY, D. 1990 Future directions in Transputer technology. In *Proceedings of UNESCO Conference on Parallel Computing in Engineering and Engineering Education* (Paris), pp. 193-203.
- [Mierowsky85] MIEROWSKY, C., TAYLOR, S., SHAPIRO, E., LEVY, J., AND SAFRA, S. 1985. The design and implementation of Flat Concurrent Prolog. Technical Report CS85-9, Dept. of Computer Science, The Weizmann Institute of Science, Rehovot, Israel.
- [Nakajima92] NAKAJIMA, K. 1992. Distributed implementation of KL1 on the Multi-PSI. In *Distributed Prolog*, P. Kacsuk and M. Wise, Eds. John Wiley & Sons Ltd., Chichester (To appear in June 1992).
- [Okumura87] OKUMURA, A., AND MATSUMOTO, Y. 1987. Parallel programming with layered streams. In *Proceedings of the IEEE Symposium on Logic Programming* (San Francisco). IEEE New York, pp. 224-231.
- [Parsec89] PARSEC 1989. *Par.C System: User's Manual and Library Reference Version 1.22*. Parsec Developments, Leiden, The Netherlands.
- [Pnueli86] PNUELI, A. 1986. Applications of temporal logic to the specification and verification of reactive systems: A survey of current trends. In *Current Trends in Concurrency, Overviews and Tutorials*, Lecture Notes in Computer Science, Vol. 224, J. W. de Bakker, W.-P. de Roever, and G. Rozenberg, Eds. Springer-Verlag, New York, pp. 510-584.
- [Ramme90] RAMME, F. 1990. Lastausgleichsverfahren in verteilten Systemen (Diplomarbeit). Paderborn University, Dept. of Mathematics & Computer Science, Paderborn, FRG, March 1990.
- [Rokusawa88] ROKUSAWA, K., ICHIYOSHI, N., CHIKAYAMA, T., AND NAKASHIMA, H. 1988. An efficient termination detection and abortion algorithm for distributed processing systems. In *Proceedings of the International Conference on Parallel Processing*, Vol. I, pp. 18-22.
- [Shapiro83] SHAPIRO, E. 1983. A subset of concurrent prolog and its interpreter. ICOT Technical Report TR-003, Institute for New Generation Computer Technology, Tokyo.

- [Shapiro86] SHAPIRO, E. 1986. Concurrent Prolog: A progress report. *IEEE Computer* 19, 8, pp. 44-58.
- [Shapiro89] SHAPIRO, E. 1989. The family of concurrent logic programming languages. In *ACM Computing Surveys* 21, 3, pp. 413-510.
- [Silverman87] SILVERMAN, W., HIRSCH, M., HOURI, A., AND SHAPIRO, E. 1987. The Logix system user manual version 1.21. In *Concurrent Prolog : Collected Papers*. Vol. 2, E. Shapiro, Ed. MIT Press, Cambridge, Mass., pp. 46-77.
- [Takeda90] TAKEDA, Y., NAKASHIMA, H., MASUDA, K., CHIKAYAMA, T., AND TAKI, K. 1990. A load balancing mechanism for large scale multiprocessor systems and its implementation. In *New Generation Computing*, 7, pp. 179-195.
- [Takeuchi87] TAKEUCHI, A., AND FURUKAWA, K. 1987. Parallel logic programming Languages. In *Proceedings of the 3rd International Conference on Logic Programming* (London), Lecture Notes in Computer Science, Vol. 225, Springer-Verlag, New York, pp. 242-254
- [Tanenbaum87] TANENBAUM, A. S. 1987 *Operating Systems: Design and Implementation*. Prentice-Hall, Englewood Cliffs, New Jersey.
- [Taylor87] TAYLOR, S., SAFRA, S., AND SHAPIRO, E. 1987. A parallel implementation of Flat Concurrent Prolog. In *International Journal of Parallel Programming* 15, 3, pp. 245-275.
- [Taylor89] TAYLOR, S. 1989. *Parallel Logic Programming Techniques*. Prentice-Hall, Englewood Cliffs, New Jersey.
- [Treleaven82] TRELEAVEN, P. C., BROWNBIDGE, D. R., AND RICHARD, P. H. 1982. Data-driven and demand-driven computer architecture. In *ACM Computing Surveys* 14, 1, pp. 93-143.
- [Ueda86] UEDA, K. 1986. Guarded Horn clauses. In *Logic Programming*. Lecture Notes in Computer Science, Vol. 221, Springer-Verlag, Berlin, pp. 168-179.
- [Ueda89] UEDA, K. 1989. Parallelism in logic programming. In *Proceedings of the IFIP Congress*, North-Holland, Amsterdam, pp. 957-964.
- [Van Roy92] VAN ROY, P., AND DESPAIN, A. M. 1992. High-performance logic programming with the Aquarius Prolog Compiler. In *IEEE Computer* 25, 1, pp. 54-68.
- [Warren83] WARREN, D. H. D. 1983. An abstract Prolog instruction set. Technical Note 309, Artificial Intelligence Center, SRI.

- [Weinbaum87] WEINBAUM, D., AND SHAPIRO, E. 1987. Hardware description and simulation using Concurrent Prolog. In *Proceedings of the CHDL '87*, Elsevier Science Publishing, pp. 9-27.
- [Xu90] XU, J., AND HWANG, K. 1990. Dynamic load balancing for parallel program execution on a message-passing multicomputer. In *Proceedings of the Second IEEE Symposium on Parallel and Distributed Processing* (Dallas), pp. 402-406.

# Index

- abstract interpreter, 27
- abstract system architecture, 45
- all-pair shortest-path algorithm, 48
- AND-parallel execution model, 47
- answer substitution, 30
- atom, 12
- atomic action, 56, 61
  
- backtracking, 39, 53
  - shallow, 36
- benchmark programs, 98
- busy waiting, 37
  
- clause, 18
  - different types, 18
  - evaluation, 36
  - selection, 36, 37
  - try function, 23, 25
- coarse grain processing, 47
- communication, 16, 50
  - asynchronous model, 16, 50
  - at the logical layer, 87
  - at the physical layer, 87
  - between Reducer and Distributor, 89
  - point-to-point, 52
  - synchronous model, 50
  - through message-passing, 50
  - through shared variables, 50
- communication channel, 16
- communication control unit (CCU), 89
- communication network (CN), 45
  - topology, 48
- communication primitive, 16
  
- communication tasks, 84
- computation, 30
  - locality of, 44
  - state of, 29
- computation goal, 13, 17
- computation deadlock, 71
- computational model, 16
- concurrent processes, 16
- concurrent logic programming languages, 19, 20
  - implementations, 81
- Concurrent Prolog, 19
- cross environment reference (XER), 46, 52
  
- data, 40
  - immutable, 51, 54
  - replication of, 50
  - representation of, 40
  - distributed representation of, 53, 54
- data coherence problem, 51
- data consistency problem, 51
- data object, 53
- data redundance, 54
- data structures, 55
- data-flow synchronization, 20, 39
- de Bruijn-type network, 99
- deadlock, 73
  - prevention of, 74
- demand-driven structure copying, 54
- directed acyclic graph (DAG), 56
- distributed computation, 46, 61
- distributed control, 83
- distributed reduction algorithm, 45, 61



- complexity issues, 70
- distributed variable representation scheme, 55, 82
- Distributor, 86, 89
- dynamic work load balancing, 45, 47, 77, 78, 83
- Eli compiler construction system, 82, 98
- fine grain processing, 47
- Flat Concurrent Prolog (FCP), 20
  - implementations of, 82
- flat languages, 19, 81
- Flat Parlog, 20
- garbage collection, 81
- global address space, 53
- guard evaluation, 37
- guard test predicates, 19, 23, 24
- guarded command, 18
- Guarded Horn Clauses (GHC), 19
- head unification, 37
- host unit, 45, 85, 96
- housekeeping tasks, 84
- input matching, 21
- I/O facilities, 86
- interprocess communication (IPC), 16, 50
- KL1, 18, 23
- list structures, 12
- livelock, 73
  - detection of, 76
  - prevention of, 73
- load balancing models, 80
- load balancing policy, 78
- local address space, 53
- logic program, 13
- logical variable, 12, 14
- single-assignment feature, 51
  - globally shared, 55
- Matrix-Multiplication, 98
- memory hierarchy, 53
- message buffer, 95
- message routing, 94
- message-passing communication, 46
- message-passing mechanism, 52
- MIMD scheme, 106
- multi-programming, 73
- multiple instruction stream-multiple data stream (MIMD), 47
- multiple-path multiple-data system (MPMD), 55
- multiprocessor systems, 50
- multi-Transputer system, 105
- mutual exclusive write access, 56
- network messages, 90
- node attributes, 56
- nondeterminism, 19
- nondeterministic process selection, 27
- observable behaviour, 30, 34, 35
- Occam, 86
- operation tasks, 84
- operational semantics, 23
- OR-parallel clause selection, 27
- OR-parallelism, 18, 27
- overall parallelization scheme, 106
- owner/member relationship, 64
- parallel FCP machine, 48
  - architecture, 45, 47, 52
  - dynamic configuration scheme, 85
  - implementation, 85, 86, 98, 106
    - prototype implementation, 98
  - micro-architecture, 84
  - modularization, 84
  - network topology, 105
  - parallelization, 46, 106

- scalability, 48, 105, 106
- PARLOG, 19
- Par.C, 86
- process, 16, 17
  - data state, 17
  - distributed representation, 58
  - hierarchical clustering, 79
- process migration, 47
- process network, 16, 46, 47
- process reduction, 17, 36
  - reduction costs, 42
- process scheduling, 66
  - scheduling costs, 43
- process selection policy, 78
- process structures, 61
  - distribution of, 61
- process synchronization, 20
- processing element, 45, 85
- program clause, 13, 17
- program procedure, 13, 17
- read-only mgu, 26
- read-only operator, 22
- read-only test unification, 22
- read-only variable, 22
- reconfigurable system architecture, 98
- reduction failure, 20
- reduction operation, 65
- reduction unit, 45, 85, 86
- resolvent, 46
- routing vector, 48
- Quicksort, 31
- Reducer, 86
- Remote Instance Stack, 88
- Router, 86, 94
- Routing Control Unit (RCU), 94
- sequential FCP machine, 39
  - data representation, 40, 41
  - operational behaviour, 42
- short-circuit technique, 73
- single-path single-data (SPSD) organization, 55
- speedup, 98, 101, 102, 103
- Strand, 20, 82
- subcomputation, 46
- substitution, 14, 25
- suspension, 20
  - on read-only variables, 66
  - on XERs, 68
  - suspension mechanism, 20, 62, 65
- suspension note list (SNL), 66
- suspension notes, 66
- synchronization, 22, 50, 62
  - Reducer and Distributor, 88
- tail variable, 12
- term, 12
- termination detection algorithm, 71
- Towers of Hanoi, 98
- trail-stack, 40, 65
- transformation of communication, 87
- transition rule, 29
- transition system, 28
- unification, 14
  - atomicity of, 21
  - compilation of, 39
  - non-atomic, 21
- unifier, 14, 15
- unit identifier, 48
- unit of parallelism, 46, 77
- variable locking mechanism, 75
- variable member, 58
- variable migration operation, 62, 66, 88
- variable owner, 58
- variable request, 93
- virtual global memory, 53
- virtual parallel machine, 105
- Warren Abstract Machine (WAM), 38, 45