software systems of Figure 2 lie below 100 tuples. These conclusions are in line with our measurements of the IFS/2 when supporting the CLIPS Production Rule System (see below).

(b) For the particular join tests of Figure 2, there is no advantage in having more than 27 search modules in the IFS/2 hardware for cardinalities under about 3000 tuples. Above that size, the IFS/2 performance curve can be kept linear by adding more search modules.

For production rule systems, we may represent facts (i.e. Working Memory Elements) and the left-hand side of rules (i.e. Condition Elements) as IFS tuples of constants and wild cards. This permits the use of the IFS/2's associative memory and its relational algebraic capability to speed up the time-consuming match phase of production systems. To test this scheme, we have re-programmed the run-time software of the CLIPS production system so as to interface it with the IFS/2 [22]. CLIPS (C Language Integrated Production System) is an OPS5-like system developed by the NASA Johnson Space Research Center. We have carried out measurements on two versions of two simple CLIPS synthetic test programs whose common static characteristics are: three Condition Elements per rule; a maximum of two shared and two unshared variables per rule; 100 rules. The right-hand ('action') side of each rule was arranged to cause a change to the working memory which subsequently matched with a left-hand-side CE containing either one or two shared variables. The two versions, V1, V2, of the two synthetic programs P1, P2, are distinguished thus:

program	no. of attributes/ rule	WM-change matches a CE with:
P1 V1	3	1 shared var.
P1 V2	3	2 shared vars.
P2 V1	6	1 shared var.
P2 V2	6	2 shared vars.

Table3

The tests were performed for given numbers of initial facts (Working Memory Elements) in the range 10 to 100,000 facts. The results, described more fully in [22], are plotted in Figure 3 on a log/log scale to show the speed-up factor of IFS-CLIPS over standard CLIPS. For this exercise, 'standard' CLIPS execution-times were measured on a Sun Sparc Workstation running at 24Mhz and having 16Mbytes of RAM.

It is seen from Figure 3 that use of the IFS/2 may be expected to speed CLIPS execution times by three orders of magnitude when there are several tens of thousand initial facts. Conversely, the IFS/2 actually slows down performance for production systems having fewer than about 100 initial facts. This is in line with the observations on Figure 2 given previously.

9. Conclusions.

In addressing the problems of slow and complex software, we have introduced a systems architectural framework that allows direct hardware support for a useful range of primitive operations which occur frequently in non-numeric (e.g. symbolic) applications. The hardware takes the form of an add-on active memory unit called the IFS/2, in which SIMD techniques are employed to exploit parallelism in a manner that requires no effort on the part of the applications programmer. The actual implementation of the active memory unit is, however, not the primary concern of this paper. The important point is that the high-level (i.e. problem-solving) requirements of applications programmers have been recognised in terms of well-used bulk data types. A representational formalism has been devised, and whole-structure operations embedded in a procedural interface to a low-level supporting unit that can be implemented in a cost-effective manner. A software simulator of the IFS/2 active memory has been distributed to several institutions. We are currently evaluating a three-node, 27 search module, IFS/2 hardware prototype at Essex. Figures for relational ***join*** and for the match phase of a production rule system have been obtained for the IFS/2, when attached as a performance accelerator to a standard Sun Workstation. The results indicate that the IFS/2 has the potential to increase the speed of whole-structure operations by a useful amount.

10. Acknowledgements.

It is a pleasure to acknowledge the contribution of other members of the IFS team at Essex. Particularly, Jenny Emby and Andy Marsh contributed to the IFS/2 hardware design; Jiwei Wang was responsible for an earlier version of the software simulator; Edward Tsang has contributed to the definition of graph operations; Shutian Lin was responsible for measurements on the CLIPS production system. The work described in this paper has been supported by SERC grants GR/F/06319, GR/F/61028 and GR/G/30867.

11. References.

1 D.A. Watt and P. Trinder, 'Towards a Theory of Bulk Types'. ESPRIT FIDE (BRA 3070) Technical Report 91/26, University of Glasgow, 1991.

2 W.D. Hillis, 'The Connection Machine'. The MIT Press, 1987.

3 S.J. Stalfo and D.P. Miranker, 'DADO: A Parallel Processor for Expert Systems'. Proc. IEEE Conf. on Parallel Processing, 1984, pages 92-100.

4 G. Marino, G. Succi, 'Data Structures for Parallel Execution of Functional Languages'. Proc. of PARLE '89, Eindhoven, LNCS 366, pp. 346-356

5 R.B.K. Dewar, E. Schonberg, J.T. Schwartz, 'High-Level Programming - An Introduction to the Programming Language SETL'. Courant Institute of Math. Sciences, New York, 1983.

6 P. Watson and P Townsend, 'The EDS Parallel Relational Database System'. Published in Parallel Database Systems, the proceedings of the 1990 PRISMA Workshop. Springer-Verlag, 1991, LNCS 503, pages 149-166.

7 B. Goldberg, 'Multiprocessor Execution of Functional Programs', International Journal of Parallel Computing, Vol. 17, No. 5, pages 425-473, 1988.

8 Arvind, R.S. Nikhil, K.K. Pingali, 'I-Structures: Data Structures for Parallel Computing'. MIT LCS (CSG Memo 269), Cambridge, Mass. Feb 1987.

9 Persistent Programming Research Group 'PS_algol Reference Manual', Fourth Edition, Persistent Programming Research Report No. 12. Department of Computing Science, University of Glasgow and Department of Computational Science, University of St. Andrews 1987.

10 S. H. Lavington, M. Standring, Y. J. Jiang, C. J. Wang and M. E. Waite, "Hardware Memory Management for Large Knowledge Bases". Proceedings of PARLE, the Conference on Parallel Architectures and Languages Europe, Eindhoven, June 1987, pages 226 - 241. (Published by Springer-Verlag as Lecture Notes in Computer Science, Nos. 258 & 259).

11 J. Page, 'High Performance Database for Client/server Systems'. UNICOM Seminar on Commercial Parallel Processing, London, June 1990, pages 21-42.

12 I. Robinson, 'A Prolog Processor Based on a Pattern Matching Memory Device'. Proceedings Third Int. Conf. on Logic Programming, London, 1986, pages 172-179. (Springer-Verlag, LNCS 225).

13 C. J. Wang and S. H. Lavington, 'SIMD Parallelism for Symbol Mapping'. Proceedings of the International Workshop on VLSI for Artificial Intelligence and Neural Networks, Oxford, September 1990, pages C38-C51.

14 J. Robinson and S. H Lavington, "A Transitive Closure and Magic Functions Machine". Proceedings of the Second International Symposium on Databases in Parallel and Distributed Systems Dublin, July 1990, pages 44-54 (IEEE Computer Society Press).

15 S. H. Lavington, "Technical Overview of the Intelligent File Store". Knowledge-Based Systems, Vol.1, No.3, June 1988, pages 166 - 172.

16 S. H. Lavington, J. M. Emby, A. J. Marsh, E. E. James and M. J. Lear, "A Modularly Extensible Scheme for Exploiting Data Parallelism". Presented at the Third International Conference on Transputer Applications, Glasgow, 1991. Published in Applications of Transputers 3, IOS Press, 1991, pages 620-625.

17 H. Walther 'Performance Measurement of the Associative Memory IFS at the Artifical Intelligence Applications Institute at Edinburgh, 13th November 1989 - 24th November 1989'. UBILAB Report 2/89, February 1989, Union Bank of Switzerland, Zurich.

18 M. Atkinson, C. Lecluse, P. Philbrow and P. Richard, "Maps as Bulk Types for Database Programming Languages". Proceedings of the Annual ESPRIT Conference, Brussels, November 1991. CEC Publication EUR 13853 EN, pages 731-758.

19 S. Reddaway, "High performance text retrieval with highly parallel hardware". Proc. UNICOM Seminar on Commercial Parallel Processing, London, February 1992, pages 61-66.

20 A. J. Marsh and S. H. Lavington, "A synthetic ***join*** benchmark for evaluating DBMS/KBS hardware and software". Department of Computer Science, University of Essex, Internal Report CSM-173, July 1992.

21 B. L. Rosser, J. M. Bedford and C. R. Dobbs, "The Ferranti Prolog Database Engine". Ferranti International Report No. ASP/AI/REP/68, September 1992.

22 S. H. Lavington, C. J. Wang, N. Kasabov and S. Lin, "Hardware support for data parallelism in production systems". Presented at the Third International Workshop on VLSI for Neural Networks and Artificial Intelligence, Oxford, September 1992, pages A1-A12.

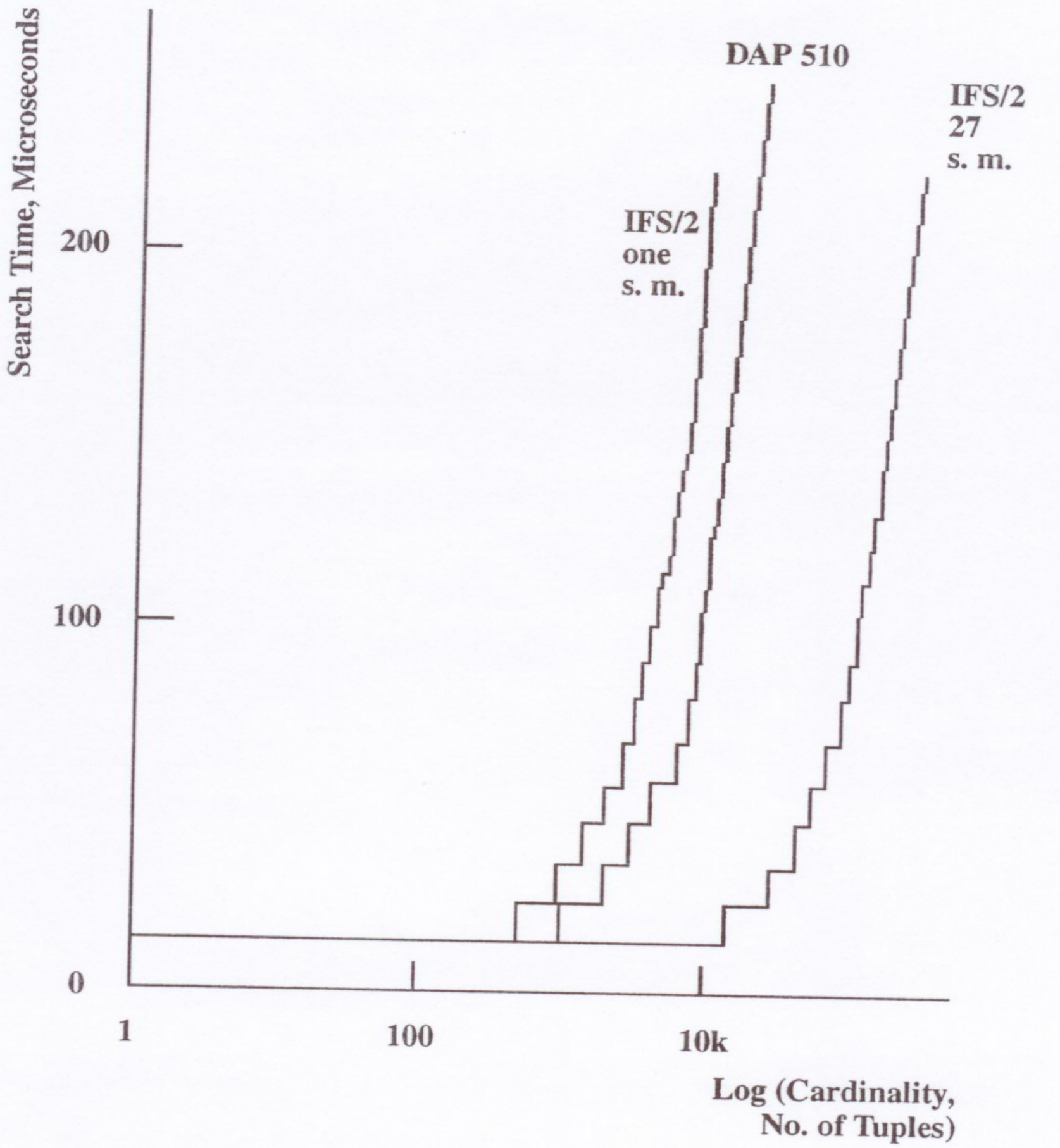

FIGURE 1: Worst-case *member* search: IFS/2 compared with a Distributed Array Processor.

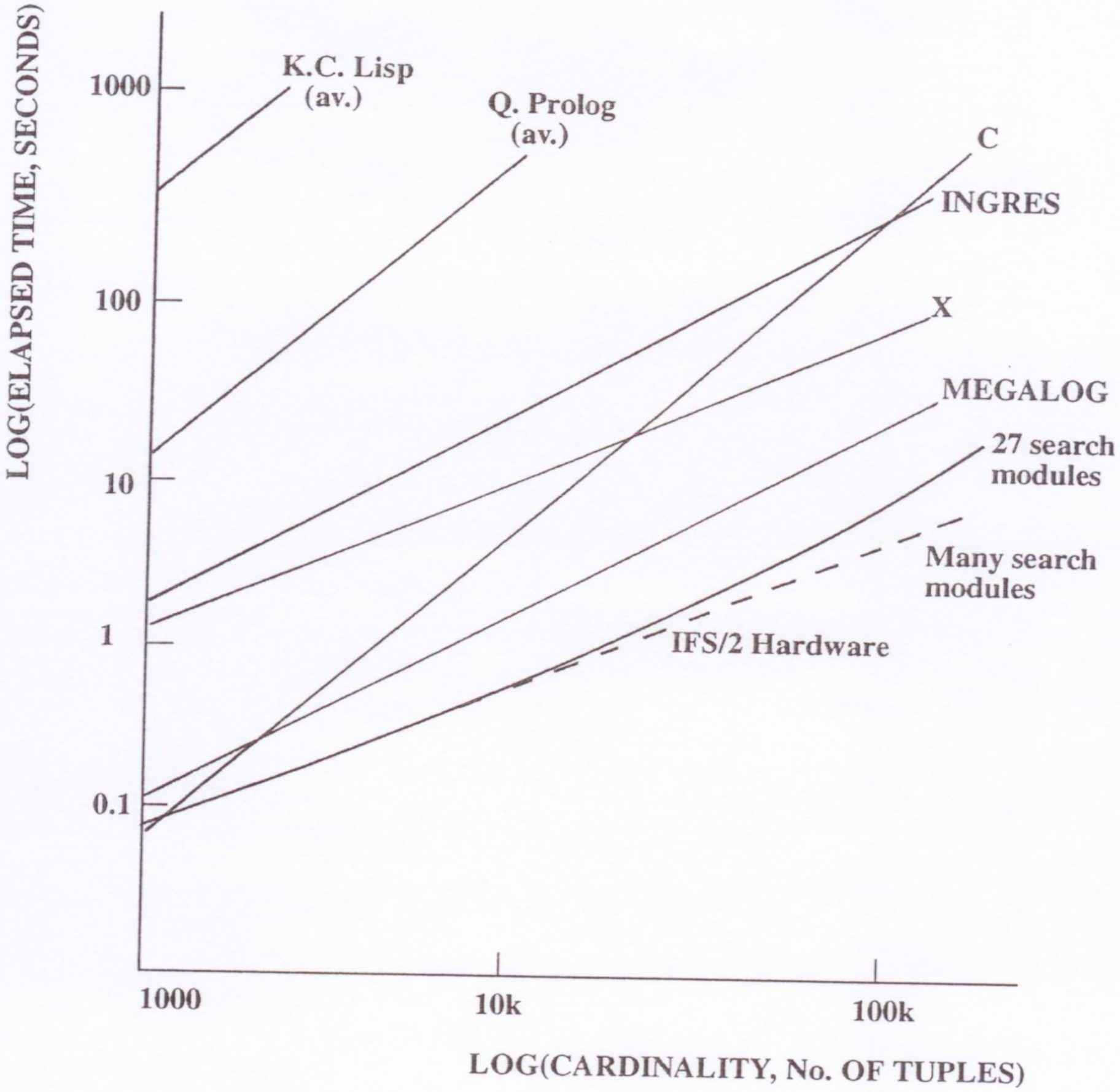

FIGURE 2: **Elapsed time for *join*, showing the performance of the IFS/2 hardware when compared with six software systems.**

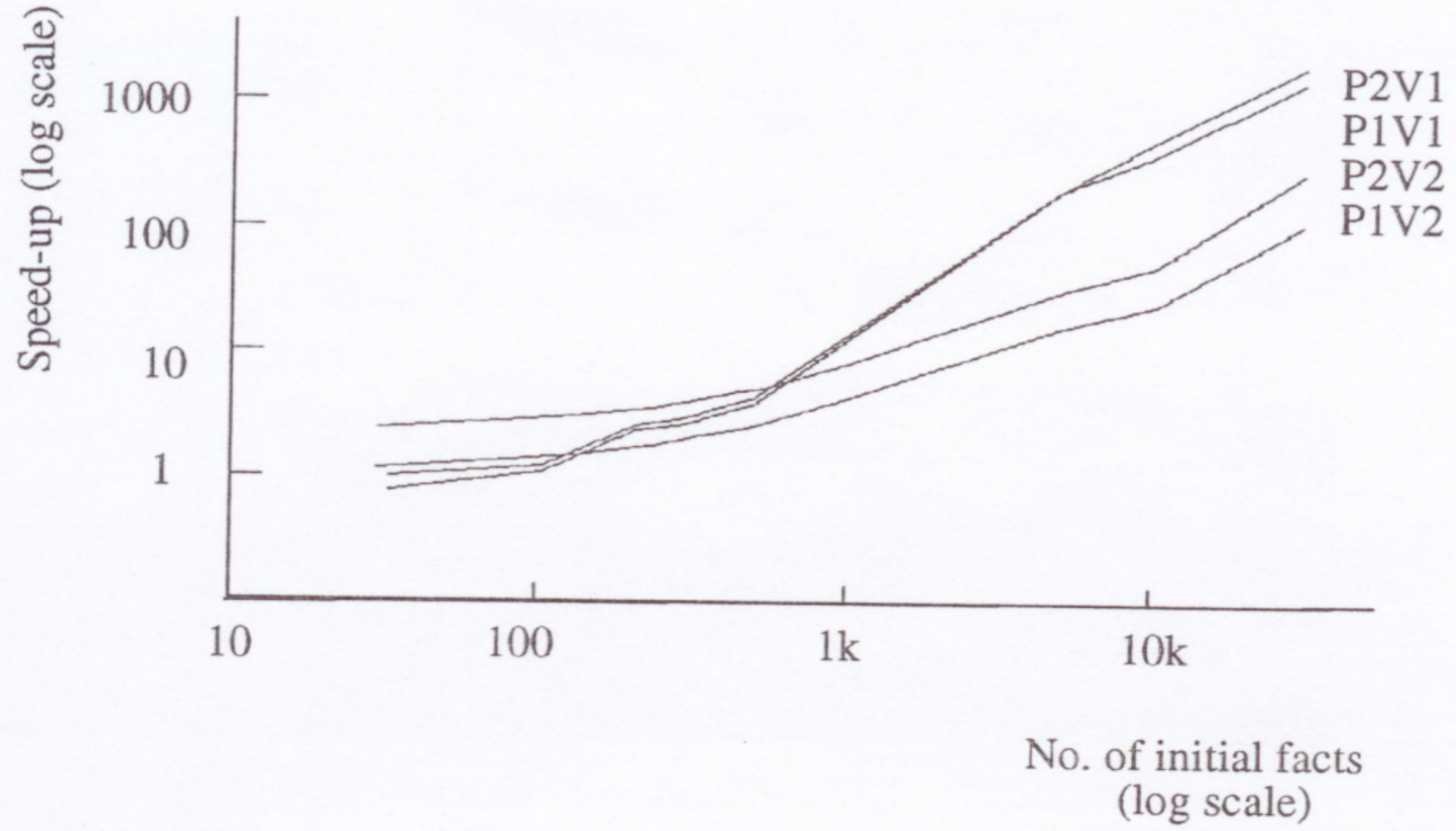

FIGURE 3: Speed-up of IFS-CLIPS versus standard CLIPS