Rule 3

A multiple or a combined multiple policy cannot coexist in the optimal strategy with none of the simple policies they contain. If it is assumed that all f_i's in Table 2 are equal to zero (i.e. there is no maintenance to IPS) then Rule 2 shows that p_9 and p_{10} have to be in the optimal solution. But from Rule 3 it follows that p_9 and p_{10} are the optimal strategy for the usage considered. If it is assumed now that all $q_j = 0$ (i.e. there is no querying to the IPS) then it is clear that no indexing path is profitable.

Rule 4

If only queries and no maintenance are performed then all the candidate indexing paths are included in the optimal strategy whereas if maintenance only is done, no indexing path appears in the IPS.

5. Conclusions

A file designer who cannot determine the effects of each alternative decision is bound to make subjective or intuitive design judgements instead of objective ones. The properties and rules stated (*a*) provide the means to improve the performance of IPS by expanding the current spectrum of alternative indexing paths examined prior to making any implementation decision, and (*b*) provide for increased confidence in the decision made. In Kollias (1979) (LP) is extended to cover the case where the user transactions follow periodic variations known in advance.

References

CARDENAS, A. F. (1973). Evaluation and selection of file organization—A model and system, *CACM*, Vol. 16 No. 9, pp. 540-548.

CARDENAS, A. F. (1975). Analysis and performance of inverted data base structures, *CACM*, Vol 18 No. 5, pp. 253-263.

CASEY, R. G. (1972). Allocation of copies of a file in an information network, *Proc. AFIPS*, SJCC, Vol. 40, pp. 617-625.

DODD, G. G. (1969). Elements of data management systems, *Computing Surveys*, Vol. 1, pp. 117-133.

EFFROYMSON, M. A. and RAY, T. L. (1966). A branch and bound algorithm for the plant location, *Oper. Research*, Vol. 13 No. 3, pp. 361-368.

GRAPA, E. and BELFORD, G. G. (1977). Some theorems to aid in solving the file allocation problem, *CACM*, Vol. 20 No. 11, pp. 878-882.

HSIAO, D. and HARARY, W. F. (1970). A formal system for information retrieval from files, *CACM*, Vol. 13 No. 2, pp. 67-73.

KNUTH, D. E. (1973). *The art of computer programming*, Vol. 3, Sorting and Searching, Addison-Wesley.

KOLLIAS, J. G. (1976). The selection of secondary file organizations, *Management Datamatics*, Vol. 5 No. 6, pp. 241-250.

KOLLIAS, J. G. (1979). File organizations and their reorganization, *Information Systems*, Vol. 4 No. 1, pp. 49-54.

LEFKOVITZ, D. (1969). *File structures for on-line systems*, Spartan Books, New York.

LUM, V. Y. (1970). Multiattribute retrieval with combined indexes, *CACM*, Vol. 13 No. 11, pp. 660-665.

LUM, V. Y. and LING, H. (1971). An optimization problem of the selection of secondary keys, *Proc. ACM Nat. Conf.*, Vol. 26, pp. 349-356.

SCKHOLNICK, M. (1975). The optimal selection of secondary indices for files, *Information Systems*, Vol. 1, pp. 141-146.

SENKO, M. E., LUM, V. Y. and OWENS, P. (1968). FOREM—A File Organization Evaluation Model, *Proc. IFIP*, North Holland, Amsterdam, pp. C19.C23.

SENKO, M. E. (1972). Details of a scientific approach to information systems, *Courant Symp. in Data Base Systems*, pp. 144-174.

SPIELBERG, K. (1969). Algorithms for the simple plant location problem with some side conditions, *Oper. Research*, Vol. 17, pp. 85-111.

STONEBRAKER, M. (1974). The choice of partial inversions and combined indices, *Int. J. Comp. Inform. Sci.*, Vol. 3 No. 2, pp. 167-188.

WANG, C. P. and LUM, V. Y. (1971). Quantitative evaluation and design trade-offs in file systems, *Proc. of the Symp. of Inf. Stor. and Retr.*, pp. 155-162.

Book review

A Programming Metholodology in Compiler Construction Part I: Concepts by J. Lewi, K. De Vlaminck, J. Huens and M. Huybrechts, 1979; 308 pages. (*North-Holland*, $41.50)

In the late 1950s the task of compiler construction was considered a major undertaking. The first FORTRAN compiler, for example, took 18 man-years to implement (Backus *et al*, 1957). Now, in the late 1970s, such a task is considered a reasonable computer science student project. The factors that have led to this over the last twenty years are (*a*) the comprehension of the organisation and modular design of the compilation process, (*b*) the development of systematic techniques for handling the majority of the important tasks that occur during compilation and (*c*) the construction of software tools that assist in the implementation of compilers and compiler components. Implicit in all these three developments is the closing of the gap between theory and practice. This book is the first part of a two-part description of an environment utilising a completely closed gap. Part I introduces the basic theoretical models whilst part 2 will consider the more practical aspects of the engineering of the environment (namely the language implementation laboratory [LILA]).

The book progresses in a formal manner from the theory for iterative language constructs (regular [translation] syntaxes) through nested language constructs ([extended] context free [translation] syntaxes) to attributed language constructs (attributed [translation] syntaxes). Within each language construct it develops, from an abstract theory model (acceptors [transducers]), an acceptor [transducer] program and then develops a generator program to produce systematically acceptor [transducer] programs from the associated syntax. As such, each section is the logical progression of the previous and the methodology used in each section is a reflection of the methodology of the previous section. Hence the book is structurally pleasing and easy to read.

In conclusion, the book is ideally suited to the software engineer who is actively involved in the application of language theory to compiler construction (or the construction of any systems software (Richards, 1979; Sassa, 1979)) and who seeks a well laid out methodology for doing so. It would also be useful to the theorist who is looking for a specific area of application. As an introduction to theory or compiler construction the reader would be better advised to do some introductory reading in other compiler construction literature (Gries, 1971; or Aho and Ullman, 1978) and the book gives a very satisfactory reference listing for this purpose.

S. K. ROBINSON (Uxbridge)

References

AHO, A. V. and ULLMAN, J. D. (1978). *Principles of Compiler Design*, Addison-Wesley.

GRIES, D. (1971). *Compiler Construction for Digital Computers*, Wiley, New York.

RICHARDS, M. (1979). A Compact Function for Regular Expression Pattern Matching, *Software—Practice and Experience*, Vol. 9, 527-534.

SASSA, M. (1979). A Pattern Matching Macro Processor, *Software-Practice and Experience*, Vol. 9, 439-456.