the target system is of MC68000 computers. That is, the hardware architecture of the target system is a bus-based architecture in which all execution information passes through the internal buses between the CPU and the main memory. In this case, even the dynamic location of a variable can be derived from the runtime information collected from the buses, notwithstanding that "postprocessing" the collected information is complicated. If the target system is different (for example, in modern systems that use caching and prefetching), the monitoring system architecture of our approach will require modifications, as stated in our article.

Jeffrey Tsai, K.Y. Fang,
and H.Y. Chen
University of Illinois at Chicago

References

1. D.C. Marinescu et al., "A Model for Monitoring and Debugging Parallel and Distributed Software," *Proc. Compsac 89*, Computer Society Press, Los Alamitos, Calif., Order No. 1964, pp. 81-88.

2. H. Tokuda, M. Kotera, and C.W. Mercer, "A Real-Time Monitor for a Distributed Real-Time Operating System," *ACM SIGPlan Notices*, Vol. 24, No. 1, Jan. 1989, pp. 68-77.

3. J.J.P. Tsai et al., "A Noninterference Monitoring and Replay Mechanism for Real-Time Software Testing and Debugging," to appear in *IEEE Trans. Software Engineering*, Vol. SE-16, No. 8, Aug. 1990.

Computer welcomes your letters.
Send technical correspondence to Editor-in-Chief Bruce D. Shriver, Vice President for Research, University of Southwestern Louisiana, PO Drawer 42730, Lafayette, LA 70504-2730. Send other comments to Letters Editor, *Computer*, 10662 Los Vaqueros Circle, PO Box 3014, Los Alamitos, CA 90720-1264.
All submissions are subject to editing for style, length, and clarity.

Correction

The following corrects the figure published on p. 56 of the June 1990 issue of *Computer*. Because two shades of black appeared to be continuous when printed, the speech bar chart in the figure did not properly illustrate the difference between the "Read only processing by block" and the "Unoptimized trace" portions of the bars. *–Ed.*

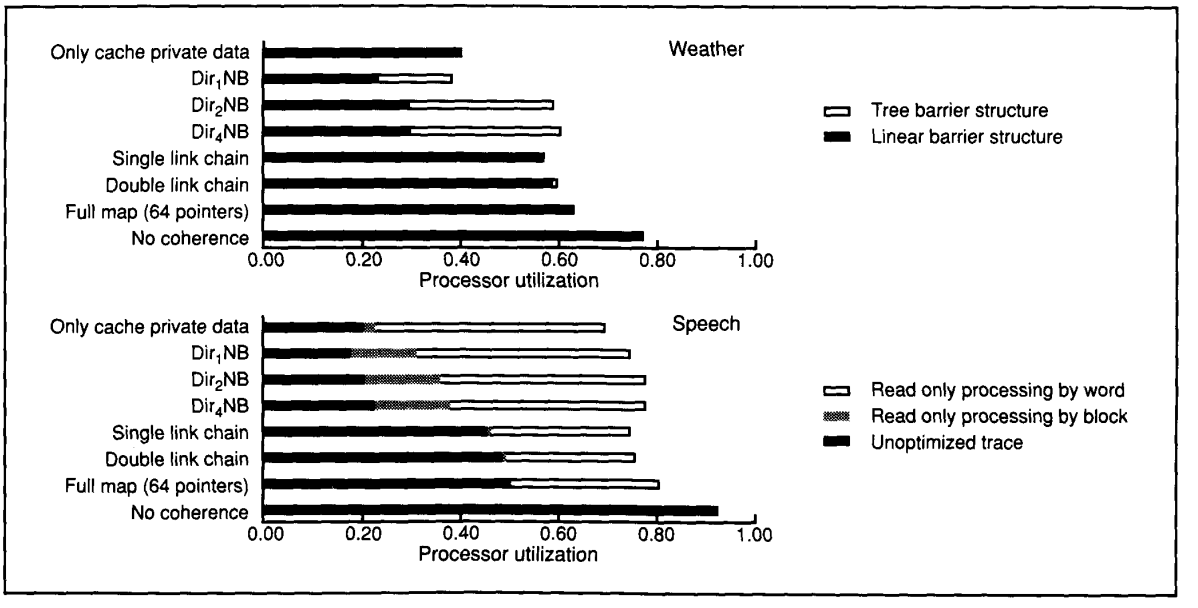

Figure 4. System-level optimizations.