We can see from Figure 6 that the GRS is considerably better than the SRS and the IDCS. The advantage of the GRS over the SRS is primarily due to the fact that in the GRS the instructions cannot be blocked by data requests, whereas in the SRS, instructions can enter the blockage buffer and thus hold up the decoding process. (Instructions are assumed to be decoded in sequence in all structures.) In fact, for \( M > 10 \) the IDCS is better than the SRS even though it does not utilize a blockage buffer. The GRS is better than the IDCS since data requests are spread over two memory cycles and a blockage buffer is used to hold data requests. This greater data accessing capability allows the GRS to service more instructions in every two memory cycles; that is, more instructions can be handled without overloading the Data Request Queue. For example for \( M = 10 \), the IDCS can request a maximum of 10 instructions every two memory cycles, whereas the GRS can request 14 instructions every two memory cycles (\( M_1 \) was optimized at 7 for \( M = 10 \) and \( L = 4 \)).

VI. Conclusions

From the above discussions, we conclude that the GRS is to be preferred over the other structures. That is, its grouping of instructions and use of the blockage buffer for data requests are superior design decisions. In addition, we note that the IDCS provides a good average memory bandwidth yet is simple to implement since it does not require the use of a blockage buffer. Thus, for a system structure simpler than the GRS, this structure is preferred. For special applications where extremes of \( \alpha \) or \( \lambda \) values occur, the above conclusions could be varied somewhat; however, the techniques have been developed to analyze such special applications.

References


Corrigenda

**Computer Systems**

In "A First Order Approximation to the Optimal Checkpoint Interval" by John W. Young, *Comm. ACM* 17, 9 (Sept. 1974), 530–531, the author has pointed out the omission of the third lambda in the numerator of the equation on lines 10 and 11 of the right column on page 531. The equation should read:

\[
1 - \exp(-\lambda (T_e + T_o))
\]

\[
dT_I = \frac{-T_e(-\exp(-\lambda (T_e + T_o))\lambda}{[1 - \exp(-\lambda (T_e + T_o))]^2} = 0.
\]

And in the same column on page 531, the line with the equation in the next to the last paragraph should read:

\[
to be T_e = \left[ \frac{2}{60} \times 14.72 \times (60) \right] = 441.6 \text{ min}.
\]

**Operating Systems**

In "Monitors: An Operating System Structuring Concept" by C.A.R. Hoare, *Comm. ACM* 17, 10 (Oct. 1974), 549–557, the author reports two errors pointed out to him by Peter Denning: (1) omission of "\( \text{count} := \text{count} + 1 \)" from the first "\( \text{procedure remove} \)" on page 553; and (2) use of "\(-\)" instead of "\(+\)" in the "\( \text{procedure release} \)" on page 554.