1. TOWARDS A MASSIVE PARALLEL DATABASE COMPUTER

1.1 Problems and Required Technologies

High-volume processing of databases requires frequent references to a very large storage space, which makes it inevitable to frequently access mechanically-accessed secondary memory devices like moving head disk units.

Historically, this kind of problems has been repetitively encountered, and resolved through the combination of two technologies. A buffer memory placed between a primary memory and a secondary memory does not only increase access speed but also decreases secondary memory access frequency, while data clustering into segments increases access locality, and does not only decrease segment references but also enhances the buffer effect by increasing a chance of repetitive references to a small set of segments.

These two techniques necessarily introduce a secondary memory access unit called a segment. Segmentation divides database processing into two processing levels, i.e., segment search and segment processing. For a given transaction, segment search searches file directories to generate a set of segment processing commands with one or two segment locations as operands. Segment processing, on the other hand, executes, for each segment command, a basic database operation on one or two operand segments. It requires to fetch operand segments from disks to a work space if they are not there yet. Decomposition of a given transaction into segment processing commands must be controlled by a well-defined concurrency control scheme to maintain database integrity.
Introduction of a buffer memory between a primary memory and a secondary memory is attempted in recent computer systems to improve database system performance (Fig. 1.1). Such a buffer is sometimes referred to by a disk cache memory. It is a semiconductor memory placed between a primary memory and disk units. Segments, when they are accessed from disk units, are also written on some pages in this buffer and be kept there as long as possible. For each segment access request, if the segment is kept in the buffer, it is read out from the buffer and no disk access is performed. This will reduce disk accesses and increase the data transfer rate. If the buffer overflows, the least recently used page in the buffer is assigned for a new segment. The performance improvement effects of such a buffer are observed in relational database management systems by some main framers. Unfortunately, however, exact figures of the improvement are not publicized due to their confidentiality. It may well be assumed that the introduction of such a buffer memory to massive parallel database machines will also bring the same effect. It is desirable, however, that this buffer memory is a multiport memory shared by a bunch of processors and a set of secondary memory devices. Otherwise, access conflicts are inevitable and its connections to processors and secondary memory devices must be frequently changed and hence it requires explicit control of interconnection networks.

Introduction of parallelism into a single-processor database system as shown in Fig. 1.1 will change it to a configuration as shown in Fig. 1.2.