Parallel LSI Router

ABSTRACT

As LSI become more complex every year, faster routing programs are required. In ICOT, a new method for LSI routing based on concurrent objects modeling has been developed. We have implemented this on a parallel inference machine PIM, and have evaluated its performance.

KEY FEATURES

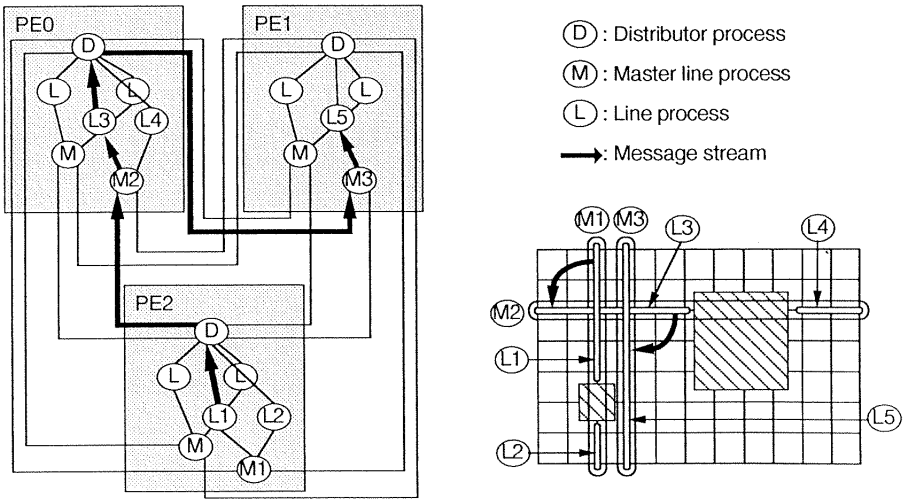
Programming based on concurrent objects model

All line segments, including free lines and occupied lines on routing grid lines, are implemented as *line processes*. Routing is executed by communication between these *line processes*.

Master line processes control the communication of messages between *line processes*.

Applicability to large-scale data

Distributor processes are assigned to each processor to reduce the amount of communication between processors. A one hundred percent wiring for large-scale data (2746×3643 grid and 556 nets) is attained.



Process structure and processor mapping

BACKGROUND

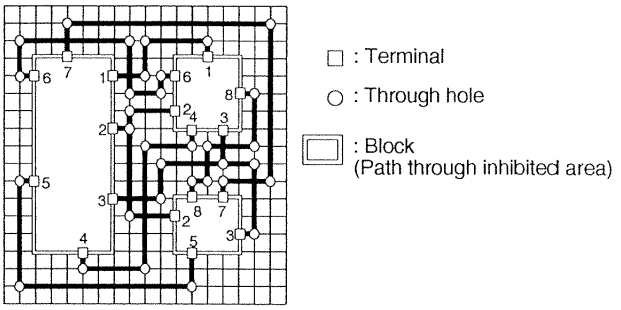
LSI routing is used to determine the connections between circuit terminals on LSI chips, after the placement of terminals is given. As the degree of LSI is large, routing design takes much time to develop. Thus, a fast and flexible router to deal with a routing constraints is desired.

A new routing program based on a concurrent objects model has been developed, and has been evaluated from the viewpoint that the parallel inference machine PIM is a high performance MIMD machine with distributed memory.

ROUTER SPECIFICATION

The program is applicable for two layer routing. That is, using two layers, one for vertical and one for horizontal paths, each connection is routed on a virtual grid on the chip surface.

The block and through hole inhibition conditions are also dealt with.



An example of two layer routing

PROGRAMMING BASED ON CONCURRENT OBJECTS MODEL

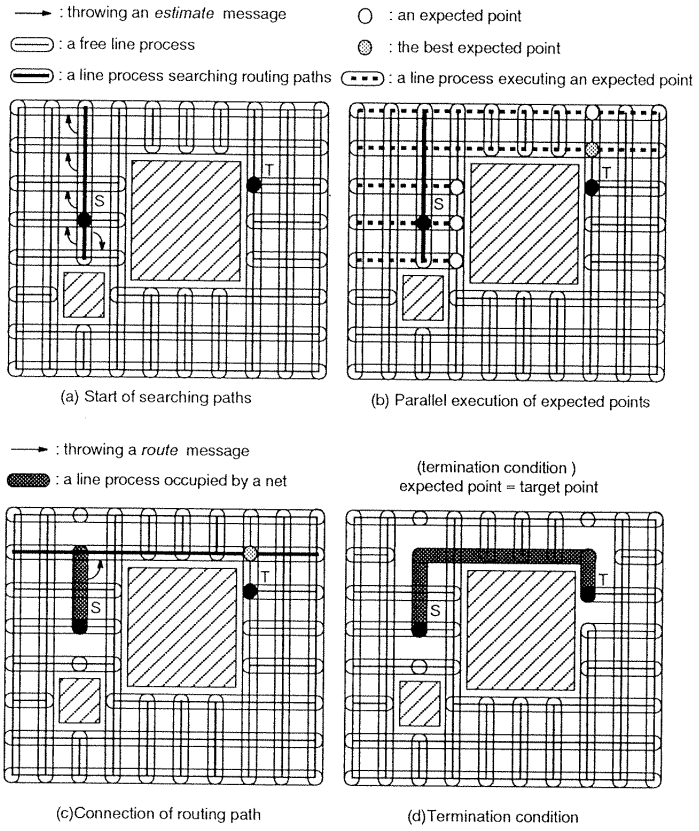
Formalizing a problem based on the *concurrent objects model* is one of the most promissing ways of embedding parallelism in a given problem. In our routing program, each line segment on a routing grid line corresponds to a process (*line process*). *Line processes* determine their routing path in communication with other *line processes.*

Then, the processes, which exchange messages frequently, are grouped. The groups are assigned to processors and are executed in parall by the various processors.

CONCURRENT ROUTING ALGORITHM

In our program, each line segment corresponds to a process. So, we used the *lookahead* line search method as a basic algorithm. This algorithm guarantees connection between a start point and a target point when paths exist between them.

Two types of parallelism are embedded. One is in the *lookahead* operation, and the other is concurrent routing of different nets.



Parallel execution of expected points

Parallel execution of *lookahead* operation for one net is as follows. A start point **S** and a target point **T** are given. When the operation starts searching virtically from **S**, request messages (*estimate* messages) for calculation of expected points are distributed from the line process including **S** to the line

process that crosses it. Here the expected points is defined as the closest point to target point **T** of all points on each line process.

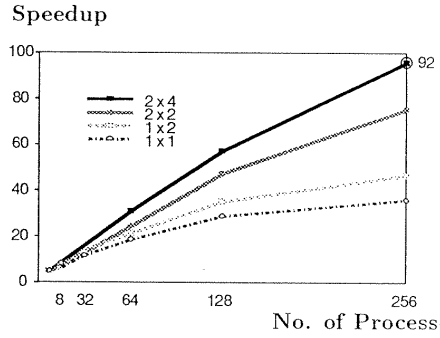
When all answers have arrived at the requesting line process, the line process found to be closest to the target point is selected. Then, **S** and the intersection point of the two line processes are connected.

Next, a request message (*route* message) for searching paths is thrown to the line process, which returned the closest point to the target point. Finally, the routing process is completed when the expected point is equal to the target point.

PROCESS STRUCTURE FOR LARGE-SCALE DATA

The designing of routing programs based on a concurrent objects model requires many communication paths between processors. So, a lot of memory is consumed in controlling them. To solve this problem, we assign a *distributor process* to each processor to reduce the number of communication paths between processors.

PARALLEL SPEEDUP



Each data is constructed by copying the small-scale data described later

Speedup

92-fold speedup is attained using 256 processors. Also a good wiring rate of 99 % to 100 % was attained in this case.

OUTLINE OF DEMONSTRATION

For small-scale data (262 × 106 grid and 136 nets) and large-scale data (1048 × 636 grid and 3264 nets), our routing program executes a routing path on PIM/m with 256 processors.