Characterization of Routers Using Combinatorial Designs

ALEXANDER KUZNETSOV, SIMON BERKOVICH
Department of Computer Science
The George Washington University
801 22nd Street NW, Suite 704
Washington, DC 20052
USA

Abstract: - The paper presents a new type of parallel router architecture based on a combinatorial arrangement of routing processing elements. The suggested approach reveals some properties of combinatorial designs, useful for economical implementation of parallel routing. In particular, such an architecture has explicit advantages for packet switching and multicast routing implementation.

Key-Words: - packet switching, network routing, multicast, combinatorial designs, computer interconnection

1 Introduction

From one of the general prospectives parallel computing is a problem of resource allocation. While in the extreme, the solutions of having a single processing unit or having an infinite number of processing units are rather trivial, anything in between assumes complex trade-offs. At the same time most of the computational problems can be characterized as a process of the interaction of two entities. Such a limitation suggests an application of combinatorial math, which in turn inspires some useful ideas.

In this work we describe an application of combinatorial arrangements to the problem of network routing. Routing can be considered a problem of pairwise interaction of input and output channels and thus appear to be a good candidate for combinatorial arrangements.

The main target of the research was to obtain an economical solution for the problem of parallel routing. Further development of this field has revealed additional benefits, such as simplified implementation of multicast traffic routing or pre-processing scheduling.

2 Combinatorial Designs

Combinatorial design – a popular mathematical construction, which presents a collection of subsets of a given set, selected in accordance with some prescribed condition. Among different types of combinatorial designs, one presents a particular point of interest in computer applications – the Balanced Incomplete Block Designs (BIBD). Every subset in a non-trivial BIBD contains less elements than the whole set.

A BIBD is characterized by five parameters:

- \( b \) - the number of blocks (subsets),
- \( v \) - the number of elements in the whole set,
- \( r \) - the number of blocks containing each element,
- \( k \) - the number of elements in each block, and
- \( \lambda \) - the number of blocks in which every pair of elements appears

So a BIBD is also called a \((b, v, r, k, \lambda)\) - configuration.

These parameters satisfy relations [1]:

\[
\begin{align*}
b \cdot k &= v \cdot r \\
\lambda \cdot (v - 1) &= r \cdot (k - 1) \\
b &\geq v
\end{align*}
\]

BIBD exists only if all conditions are satisfied.

A BIBD, for which \( v = b \) and \( r = k \) is called a symmetric BIBD. For \( \lambda = 1 \) symmetric BIBD should satisfy the following criteria:
\[ v = b = s^2 + s + 1 \]
\[ r = k = s + 1 \]
\[ \lambda = 1 \]
\[ s = p \cdot m \]

where \( p \) is a prime and \( m \) is a positive number.

So far no general necessary conditions for existence of symmetric BIBDs have been proven for higher \( \lambda \) [1].

For the sake of illustration, let us consider a set of 7 elements \( X = \{ x_1, x_2, x_3, x_4, x_5, x_6, x_7 \} \). A BIBD constitutes a collection of subsets \( b_i = 1, 2, \ldots, 7 \) as shown by the columns in Table 1.

\[
\begin{array}{cccccccc}
\hline
 & b_1 & b_2 & b_3 & b_4 & b_5 & b_6 & b_7 \\
\hline
x_1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \\
x_2 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\
x_3 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \\
x_4 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\
x_5 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\
x_6 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\
x_7 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\
\hline
\end{array}
\]

*Table 1: A symmetric BIBD*

The pairwise balanced distribution of the elements with \( \lambda = 1 \) assumes that any pair of elements of \( X \), say \( x_i \) and \( x_j \), can be found in exactly one block (which in our case is \( b_j \)).

In a symmetric design the number of elements satisfies
\[
4n - 1 \leq v = n^2 + n + 1,\quad \text{were } n = k - \lambda \text{ is the order of the design [2].} \]
The upper bound corresponds to finite projective plane of order \( n \), in which symmetric BIBDs will have \( (n^2 + n + 1, n + 1, n + 1, 1) \) - configurations. Some well-known projective plane configurations are shown in Table 2.

While we need the whole set of BIBD blocks to cover any possible pair, all elements of a given set can be found in a smaller subset of blocks. Such a subset for a symmetric BIBD contains \( r \) blocks.

For example, consider element \( x_1 \). This element is contained in blocks \( b_1, b_5, \) and \( b_7 \). The full covering implies that \( b_1 \cup b_5 \cup b_7 = X \).

Thus, \( b_1 = \{ x_1, x_2, x_4 \} \), \( b_5 = \{ x_1, x_5, x_6 \} \), \( b_7 = \{ x_1, x_3, x_7 \} \), so the union of these sets apparently presents the whole set \( X = \{ x_1, x_2, x_3, x_4, x_5, x_6, x_7 \} \). The validity of this statement follows directly from the property of the pairwise balance.

Indeed, the block with a given element contains its pairings with all other elements. Therefore, the union of these blocks must contain all the elements of the basic set \( X \).

### 3 Processor Using BIBD

Combinatorial designs can provide an economic solution to the problem of computer interconnections.

Consider a computer network where processor nodes are associated with the elements of a primary set and subsets of these nodes are linked by buses according to a BIBD. Thanks to the pairwise balanced arrangement of nodes, any given processor can directly communicate with any other processor through a particular bus. Also, in this scheme each interconnecting bus can be attached to a separate memory unit, so any pair of processor nodes can access a common section of memory. These combinatorial interconnections provide the facilities of full connectivity in a cost-effective fashion. At the same time having a dedicated bus toward any direction...
reduces contention and increases available bandwidth.

A multiprocessor network, also called REBUS architecture suggested in [3], presents another principle for combinatorial interconnections. The role of the components of the BIBD are reversed: the elements of the primary set represent links, like buses, while the subsets of these elements represent the processor nodes.

In the given example, every register is stored in three copies over the processor nodes. Suppose an exchange of information (a two-address command) has to occur between registers #3 and #5. Due to the pairwise balanced distribution of objects, there is a processor node where these registers coexist, in this case it is node 6. Thus, this pair of registers can interact right after creating new versions of the register values without moving data across the communications links. Then information updates are sent through two different buses delivering new versions of registers #3 and #5 to corresponding processor nodes: 3, 4, and 1, 5. This organization of communications allows the information exchange in the whole system of objects to run concurrently.

4 Network Router/Switch Using BIBD

A similar approach could be applied to the routing or switching problem, especially in packet-switching networks, where routing requires intense real-time computations. In a network router, a packet arrived through an input channel should be transmitted to one of the output channels according to the routing table. Thus the routing in general can be reduced to the problem of pairwise coupling of input and output channels.

Communicating objects are associated with the connection links and, correspondingly, these objects are replicated in each of the processor nodes. For any pair of objects there will always exist a processor node where they reside together. As a result, the information exchange between objects can be performed through immediate interaction of their replicated copies. The transmission of information by object replication is completely different from conventional communication by message passing or shared memory.

The object replication entails redundancy, which is mainly determined by the number of subsets in the underlying BIBD. By conditions, outlined in the previous section, the number of these subsets cannot be less than the number of elements in the primary set: $b \leq v$.

To lessen object replications, it is beneficial to have symmetric BIBD, where $b = v$ (and hence $r = k$). The simplest example of this arrangement is given by the $(7,7,3,3,1)$ - configuration as shown in Fig. 1.

![Figure 1: REBUS microprocessor architecture](image1)

![Figure 2: Combinatorial router architecture](image2)
The routing organization using the suggested scheme is shown in Fig. 2. In the case of \((7, 7, 3, 3, 1)\) router, each PE has three input channels, a memory module, a CPU and three output channels. Initially, all the PEs should be loaded with the routing information. Each of PEs contains a part of the routing table, corresponding channels directly connected to the PE. In the given example the output contention resolution is made with output buffers O1-O7, while it can be done by other means.

Assume a packet came from the input interface I2 and should be routed to the interface O5. The interface I2 is connected to the processing units PE2, PE3 and PE5 with the line 2. All of the units listed start to search the route to the destination in their routing tables. Processing units PE2 and PE3 fail as they have no route to O5. PE5 has the route listed in its routing table and it forwards the packet received to the O5.

There is an ambiguity if a packet must be routed to a destination channel with the same sequence number as the source channel. This may happen, for example, if there is a loop in the routing table, a mis-configured downstream router or an address spoofing attack occurred. In this case the packet should be dropped, with possible notification of the sender.

Due to the distributed nature of processing, the suggested design has better resistance to overloads while Denial of Service (DoS) attacks or global routing instability has happened. While overload of any of the channels is fatal for a traditional single-processor, shared-bus and shared-memory router, combinatorial router remains mostly operational even some of the channels and PEs get clogged. This makes the combinatorial router a good candidate for Internet routers.

### 4.1 Pre-processing Scheduling

One of the problems with parallel processing in general and routing in particular is scheduling. Given a problem (a packet for a router), some mechanism should allocate processing resources and properly distribute the data among them. Due to its complexity, the problem requires substantial resources by itself. Thus the scheduling operation could easily be a bottleneck for the whole system.

Unlike existing solutions, a router with BIBD has no scheduling overhead. There is a predetermined subset of processing nodes, which could perform the routing for a packet, arrived through an input channel. It is worth to be noted, such scheduling is done by bare wires with no computational resources involved. In the previous example, the packet (arrived through I2) will be processed by a fully covering subset of processing nodes, which consists of \{ PE2, PE3, PE5 \}. Such a subset is sufficient for routing to any combination of output interfaces.

### 4.2 Routing Table Reduction

As you can see in Fig. 2, every processing element has a limited number of available output channels. Thus the element needs only a part of the routing table, which corresponds to the connected outputs only. Some overhead is introduced by the longest prefix match algorithm. The routing program in a PE needs to consider, if there could be other PEs, which can route a packet. In the worst case scenario, when prefixes of every length have different destinations, every processing element should hold the entire routing table.

### 4.3 Multicasting

One of the problems during a multicast transmission is output blocking avoidance. The bigger is the multicast group (i.e. the more channels is involved into it), the more probable is the contention for output channels.

It is obvious, that such a construction can easily route multicast traffic. As it was noted before, the minimum necessary number of processing elements, which can route a packet to any number of output channels is equal to the number of blocks in a fully covering subset, which is \(r\).

During a multicast transmission, the number of feasible output channels for a single processing elements is equal to \(k = n + \lambda\). This simplifies the problem to be solved during the output scheduling stage with an algorithm like ESLIP [4].

At the same time, an implementation of the router with output buffering and subsequent independent transmission eliminates the necessity of such synchronization.
4.4 Route Caching

In general, Internet traffic can be characterized as “highly non-uniform”. Over time, users' interests are shifting from one host to another, generating funnel flow towards the attraction source. These bursts could create an excessive load on the routing elements, but fortunately the number of source-destination pairs are limited and so any caching technique could be very beneficial. Once a destination route has been determined, the router can put the destination address along with destination port index into cache. Any subsequent packet should first be verified against what has been cached to find if a similar packet has been routed.

To make use of caching, a parallel router should possess a property of persistence. If a processing element had performed routing for a packet, the successor packets should be processed by the same processing element. As the presented architecture has a predetermined route for every input/output combination, the property of persistence successfully holds.

5 Conclusion

Using combinatorial designs for router construction has some advantages due to virtue of such designs.

1. Effectual pre-processing scheduling with bare wires.
2. Simplified multicast implementation.
3. Simplifying more complex architecture development given extra requirements like performance or reliability.
4. The property of easy scaling without changing the processing element itself.
5. Effectual switching capabilities.

References: