


SB:3630-85-001

7 July 1985

TO: Distribution 
FROM: Steve Bellenot
SUBJECT: COMMO* Object to Node Assignments

Here are some tentative COMMO* object to hypercube node assignments for all dimensions up to five. Although these are just (intelligent?) guesses, the assignments are all designed from the same viewpoint. So they should serve as good "first pass" benchmarks. All the assignments are good in the sense that if object A sends messages to object B, then either both objects are on the same node or their nodes are nearest neighbors.

SB:gl

Distribution:

B. Balogh	510-110
Steven Bellenot	510-202
Brian Beckman	264-748
Dave Curkendall	264-748
Phil Hontalas	264-748
Steven Hughes	126-200
Dave Jefferson	510-202
Eric Levy	510-113
Todd Litwin	264-748
Garry Paine	510-202
Richard Masline	510-110
John Spagnuolo	510-264
Tuan Truong	510-271
Jack Tupman	510-202
Van Warren	510-264
Fred Wieland	510-264
Barbara Zimmerman	264-748

COMMO* Object to Node Assignments

REVIEW of COMMO* objects:

Figure 1 is Fred Wieland's graph of COMMO* object types. In the current COMMO* program, only the nets are different from what is pictured in Figure 1. There are no Division Commo nets. Battalion Commo nets handle all communication between Battalion and Brigade levels. While Brigade Commo nets handle all communication between Brigade and Division levels.

Figure 2 gives the logical grouping of the COMMO* objects used for the assignments. Each net group is connected to several groups. A Brigade Commo net is connected to one Division group and three Brigade groups. A Battalion Commo net is connected to one Brigade group and either two or four Battalion groups (see Figures 5, 6 and 7). (Note that the template given in Figure 4 is used for most the figures that follow it.)

Figure 3 isolates the COMMO* initialization objects. These objects are not assigned nodes and they don't appear except in Figures 1 and 3. (See Assumption 1 below.)

The non-initializing objects divide into two groups, those under Division G2 and those under Division G3, which do not sent messages to each other. The logical grouping of either G2 or G3 is abstracted in Figure 5. Figures 6 and 7 list the Division G3 and G2 object groups in the current COMMO* program. Each circle in these figures is one of the ovals in Figure 2. Figure 8 is a "more load balanced version" of Figure 7. Figure 8 can always be used for the Division G2 assignments as given in Figures 9, 11, 13 and 15.

ASSUMPTIONS:

1. The six driver objects and the prime mover object (see Figures 1 and 3) are unimportant in the long term. (They are "active" only at time zero.) These objects are henceforth ignored.
2. All other object types are roughly equal for purposes of load balancing. Although this is a questionable assumption, there is no better information at this time. Also the assignments below give most of the nodes roughly the same number of each "meta object type" (i.e. delay, mess. gen. and commo objects as in Figure 2). (See Tables 1 and 2.) This assumption implies that load balancing amounts to equalizing the number of objects per node.
3. Hence the problem is to minimize the communications costs subject to the limitation in 2 above.

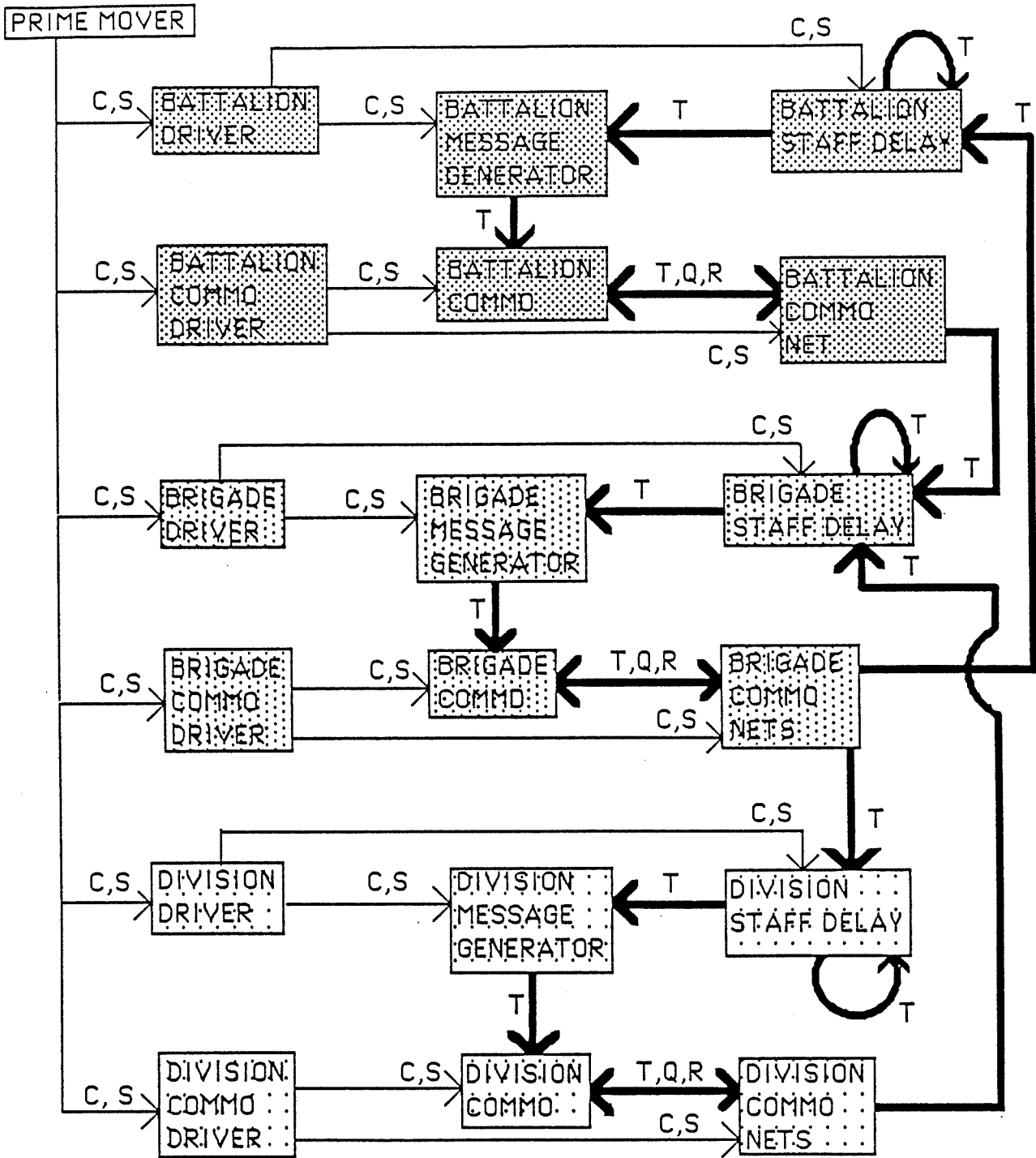


FIGURE 1. COMMO* INTERACTION DIAGRAM

KEY:

C CREATE MESSAGE Q QUERY
 T TACTICAL MESSAGE R QUERY REPLY
 S STATE DEFINING MESSAGES

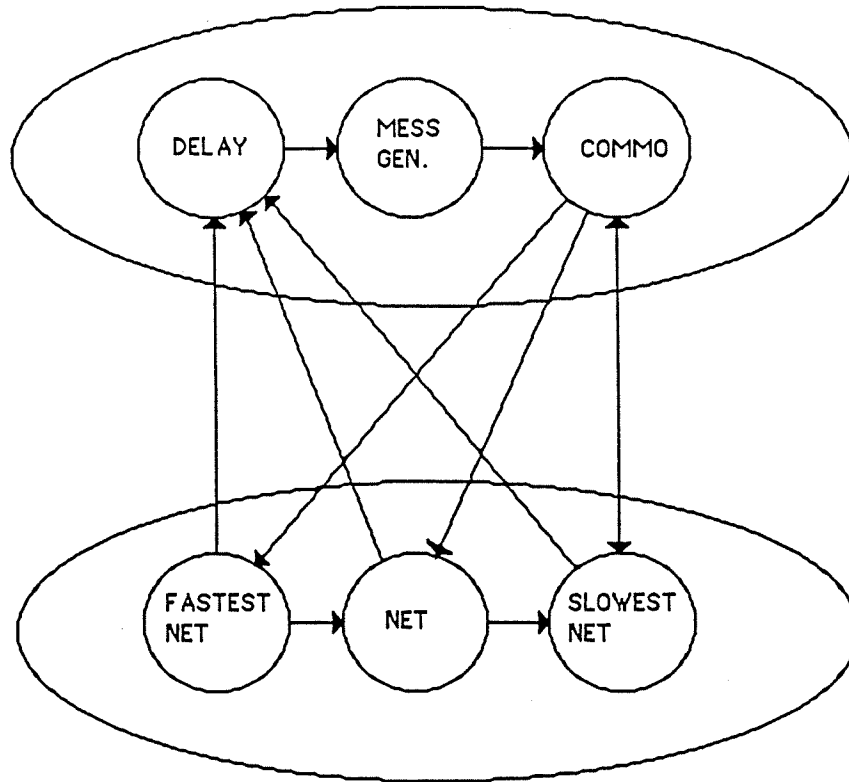
 BATTALION LEVEL

 BRIGADE LEVEL

 DIVISION LEVEL

HEAVY DARK LINES DENOTE THE PATH OF TACTICAL MILITARY MESSAGES.

DIV, BDE, & BN
OBJECT
GROUPS



NET OBJECT
GROUPS
(1-3 NETS)

FIGURE 2: COMMO* OBJECTS AND MESSAGE PATHS

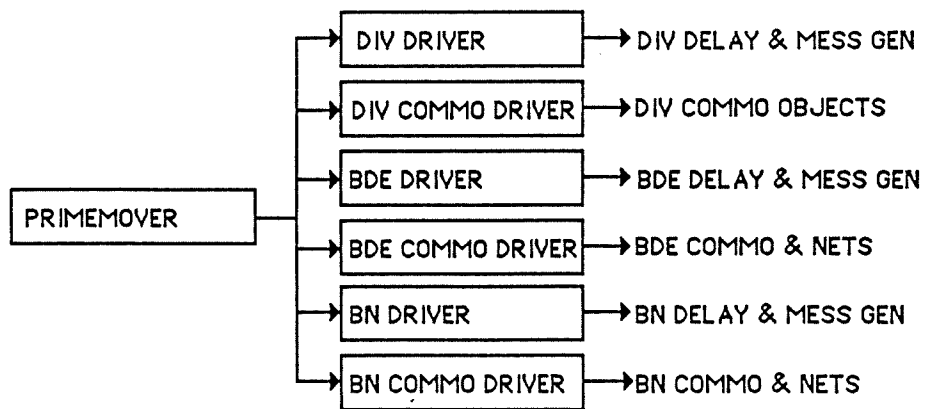


FIGURE 3: COMMO* INITIALIZING OBJECTS

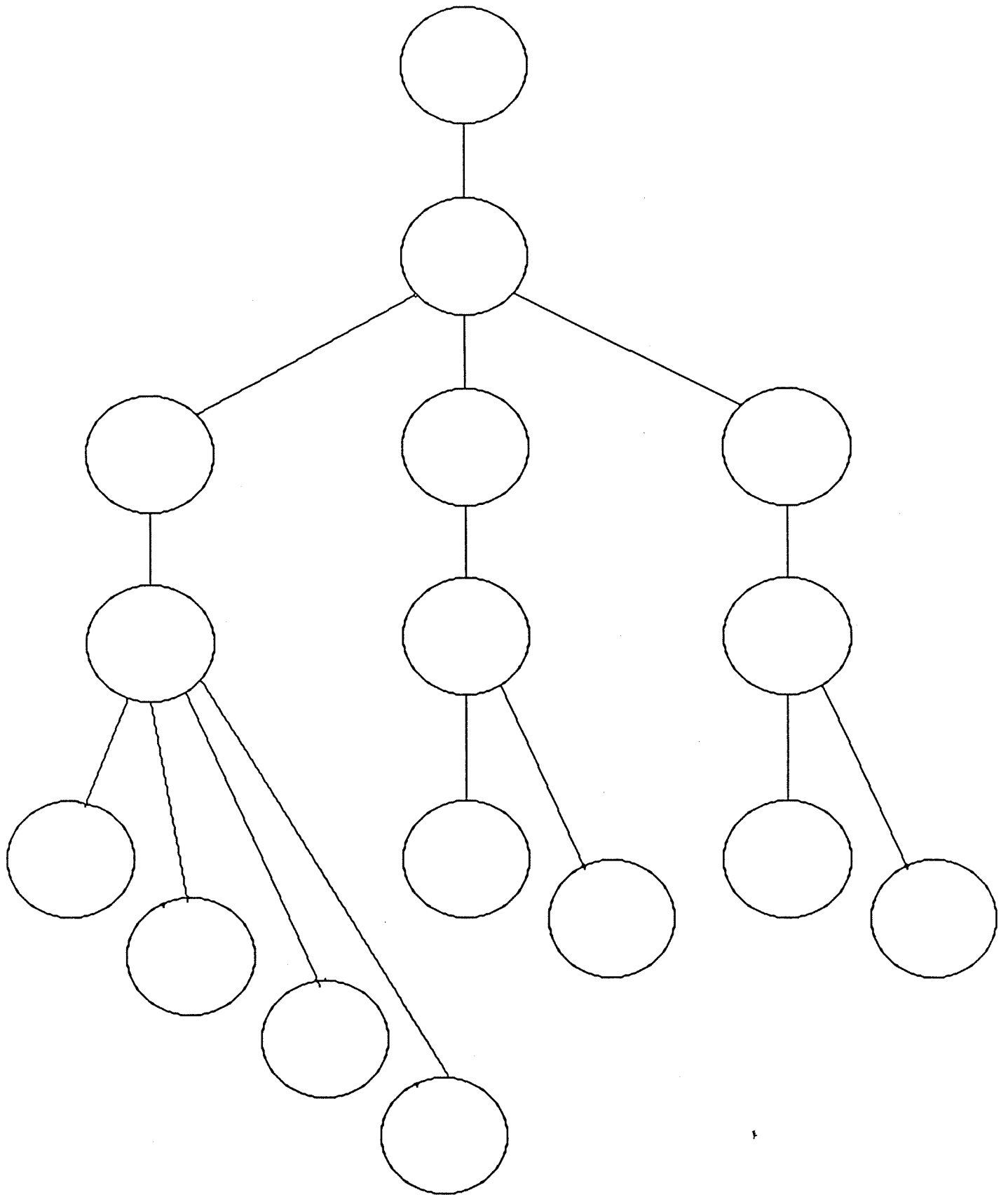


FIGURE 4: TEMPLATE FOR FIGURES 4,5,6,7,8,9,11,13 & 15.

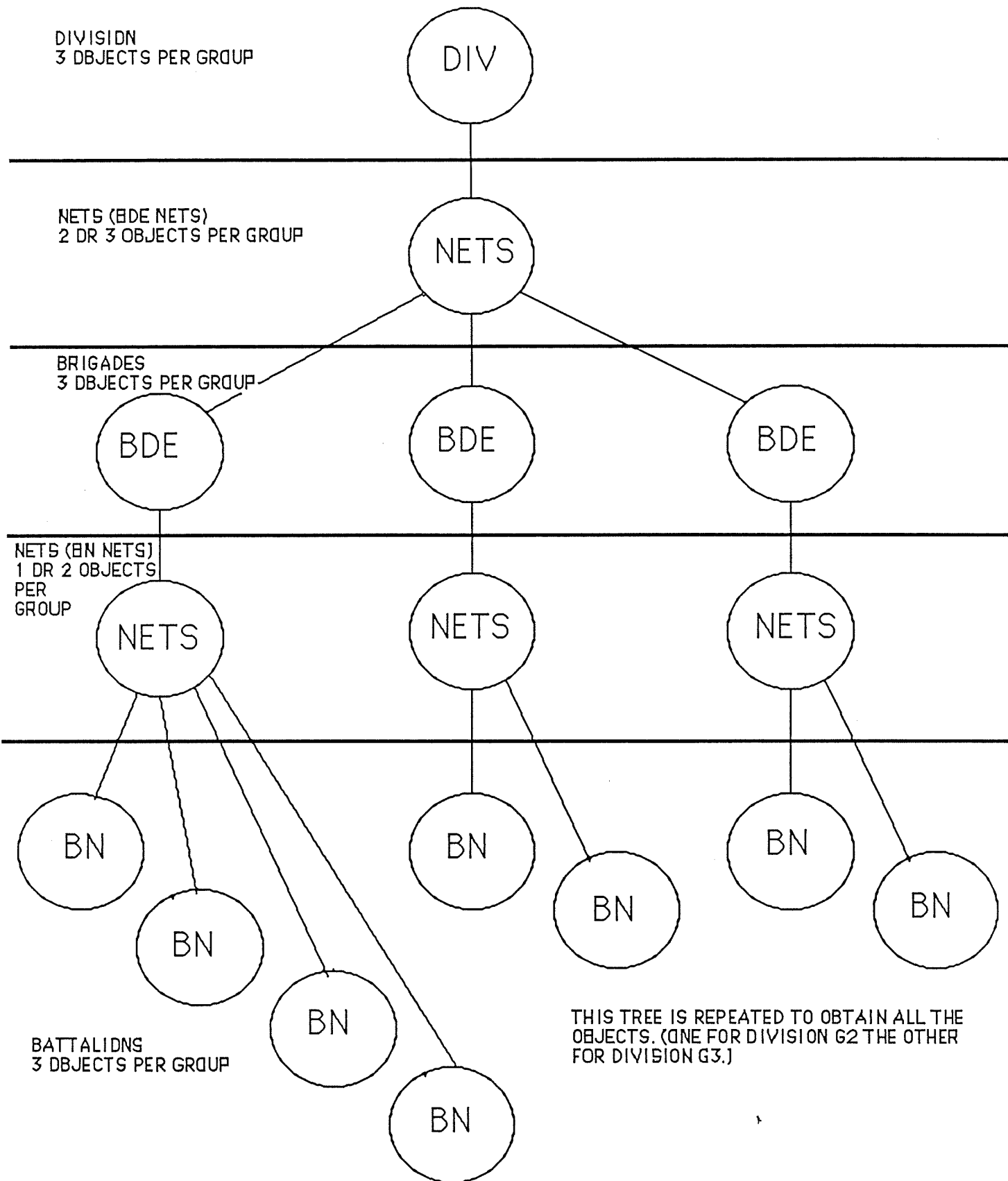


FIGURE 5: COMMO* OBJECT GROUPS

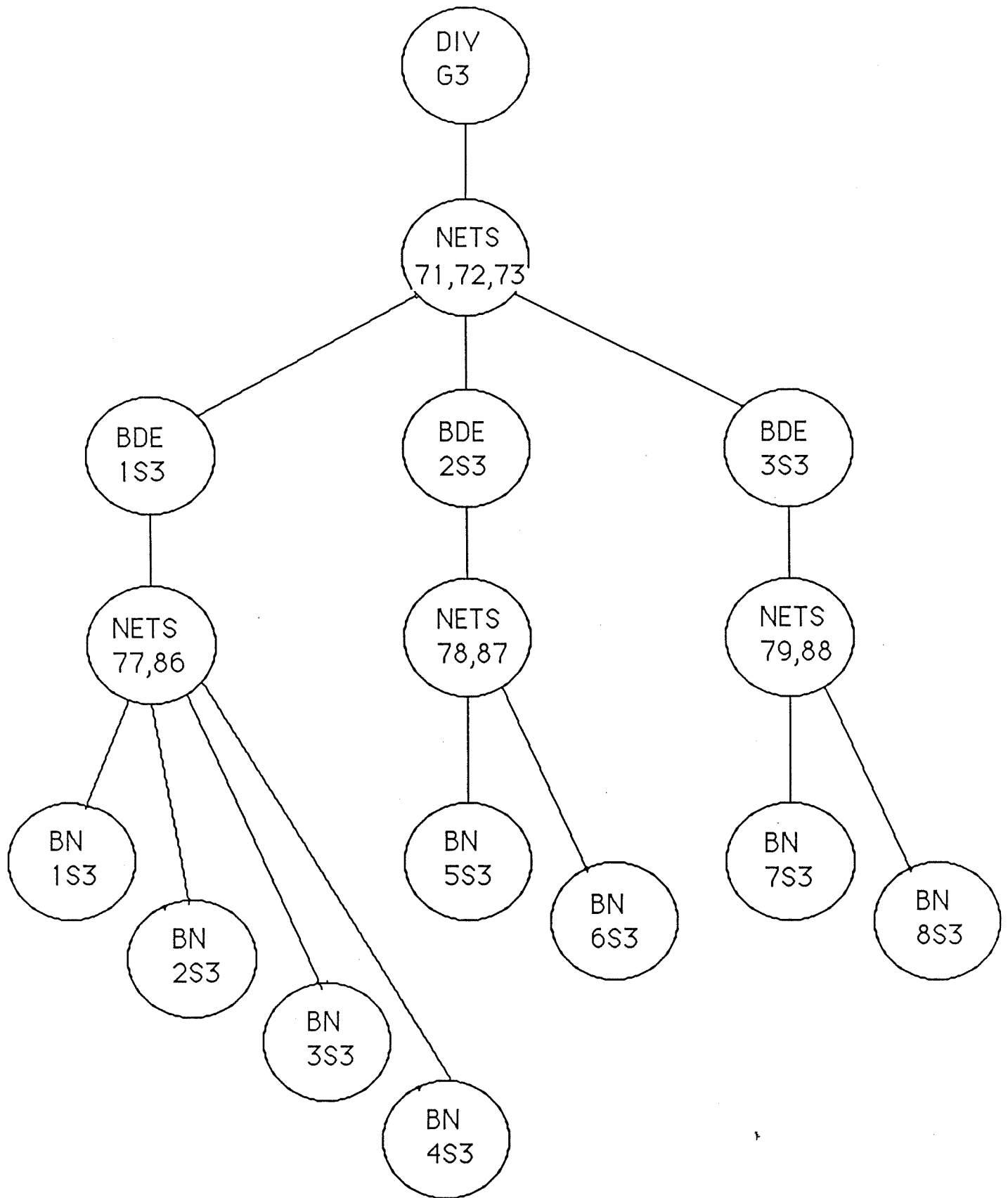


FIGURE 6: DIVISION G3 COMMO* OBJECT GROUPS

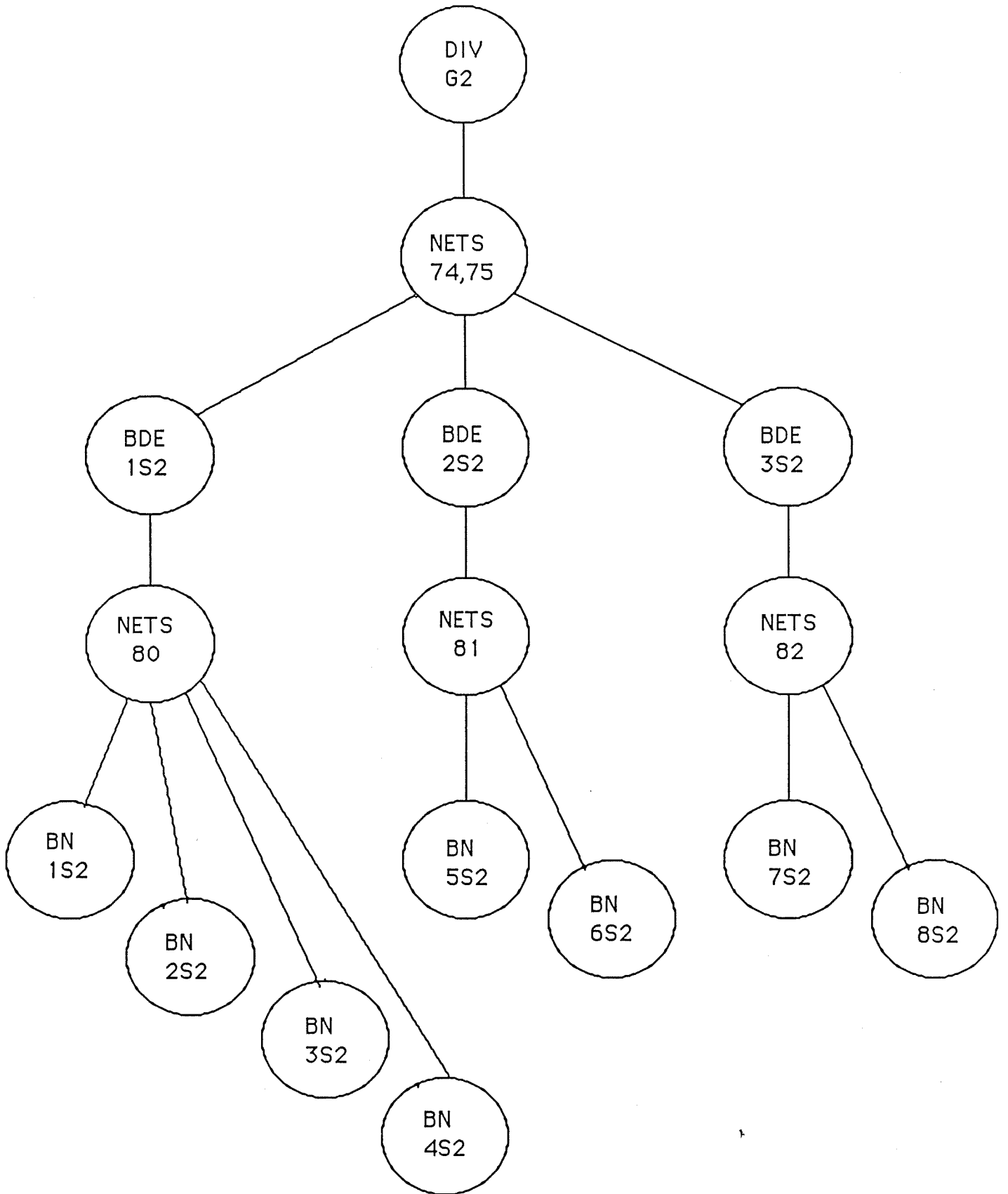
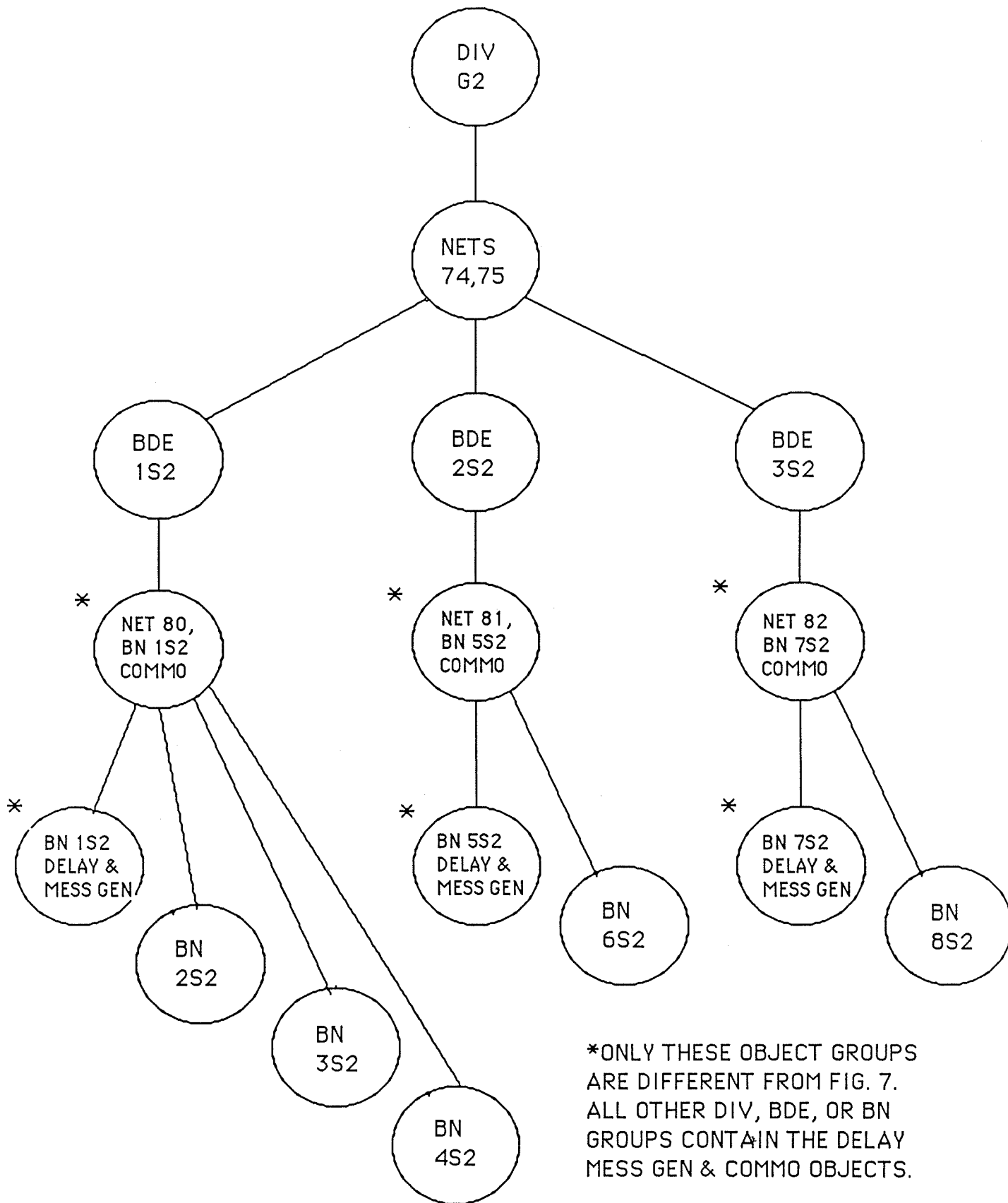


FIGURE 7: DIVISION G2 COMMO* OBJECT GROUPS



*ONLY THESE OBJECT GROUPS ARE DIFFERENT FROM FIG. 7. ALL OTHER DIV, BDE, OR BN GROUPS CONTAIN THE DELAY MESS GEN & COMMO OBJECTS.

FIGURE 8: MODIFIED DIV G2 COMMO* OBJECT GROUPS

ASSIGNMENTS:

Figures 9 (five dimensional), 11 (four dimensional), 13 (three dimensional) and 15 (two dimensional) graphically give the node assignments for each group. Note that the node grouping are labeled A, B, etc for Figures 11 and 13. Each group has two node numbers, one for Division G3 (lower number) and the other for Division G2 (upper number).

Figures 10, 12 and 14 try to "picture" these logical grouping in the correct dimensional hypercube. The pictures for 3-D (Figure 14) and 4-D (Figure 12) also show the hypercube connections unused by COMMO*. The 5-D (Figure 10) case is less successfully pictured.

RESULTS:

1. Communication costs are low. Interobject messages are either within one node or between nearest neighbor nodes.

2. Load balancing is less clear. Of the 86 objects in Figure 5, there are 41 objects in Division G2 and 45 objects in Division G3. (The 7 other objects in Figure 3 are still ignored.) Since 86 is not a power of two, it is not possible to give each node the same number of objects. Table 1 gives the number of nodes with a given number of objects for each dimension's assignment. Table 2 gives a more detailed decomposition of "meta object types" per node. Define

$$\text{unbalance} = (M - m) / M * 100\%,$$

where: M = the maximum number of objects per node, and
m = the minimum number of objects per node.

Dimension	Objects per node	Unbalance
0	86	0.0%
1	45 on 1 node, 41 on the other	8.9%
2	20, 21, 22 and 23	13.0%
3	11 on 6 nodes, 10 on 5 nodes	9.1%
4	6 on 6 nodes, 5 on 10 nodes	16.7%
5	3 on 22 nodes, 2 on 10 nodes	33.3%

TABLE 1 Objects per node.

	Number of nodes	Delay objects	Mess. gen. objects	Commo objects	Net objects
5-D	21	1	1	1	0
	4	0	0	0	2
	3	0	0	1	1
	3	1	1	0	0
	1	0	0	0	3
4-D	4	2	2	2	0
	4	2	2	1	0
	4	1	1	1	2
	3	1	1	2	1
	1	1	1	1	3
3-D	2	3	3	3	2
	2	3	3	3	1
	1	4	4	4	0
	1	4	4	3	0
	1	2	2	2	5
	1	2	2	3	3
2-D	1	6	6	6	5
	1	6	6	6	4
	1	6	6	6	3
	1	6	6	6	2
1-D	1	12	12	12	9
	1	12	12	12	5
0-D	1	24	24	24	14

TABLE 2: Load balance by meta types.

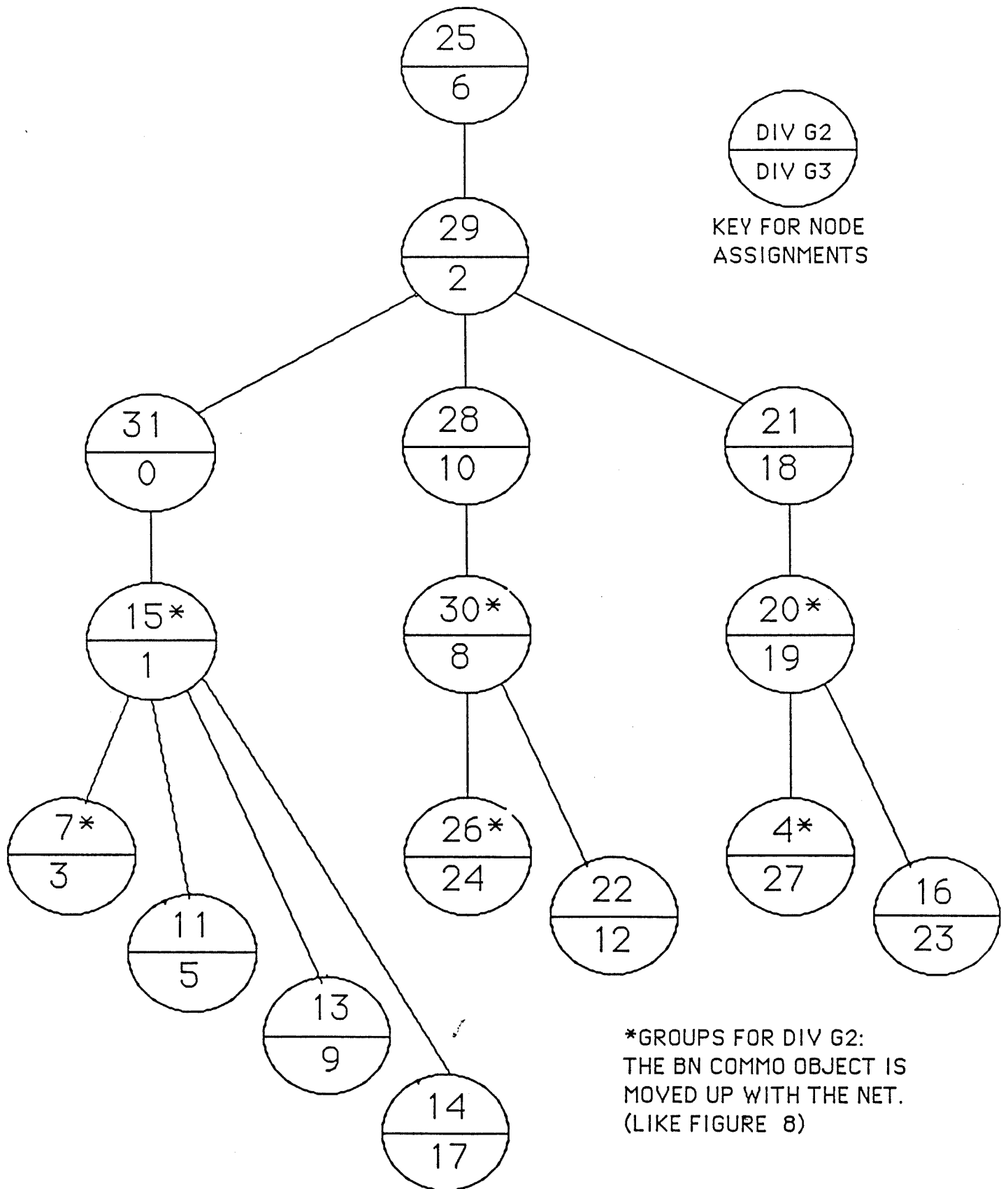
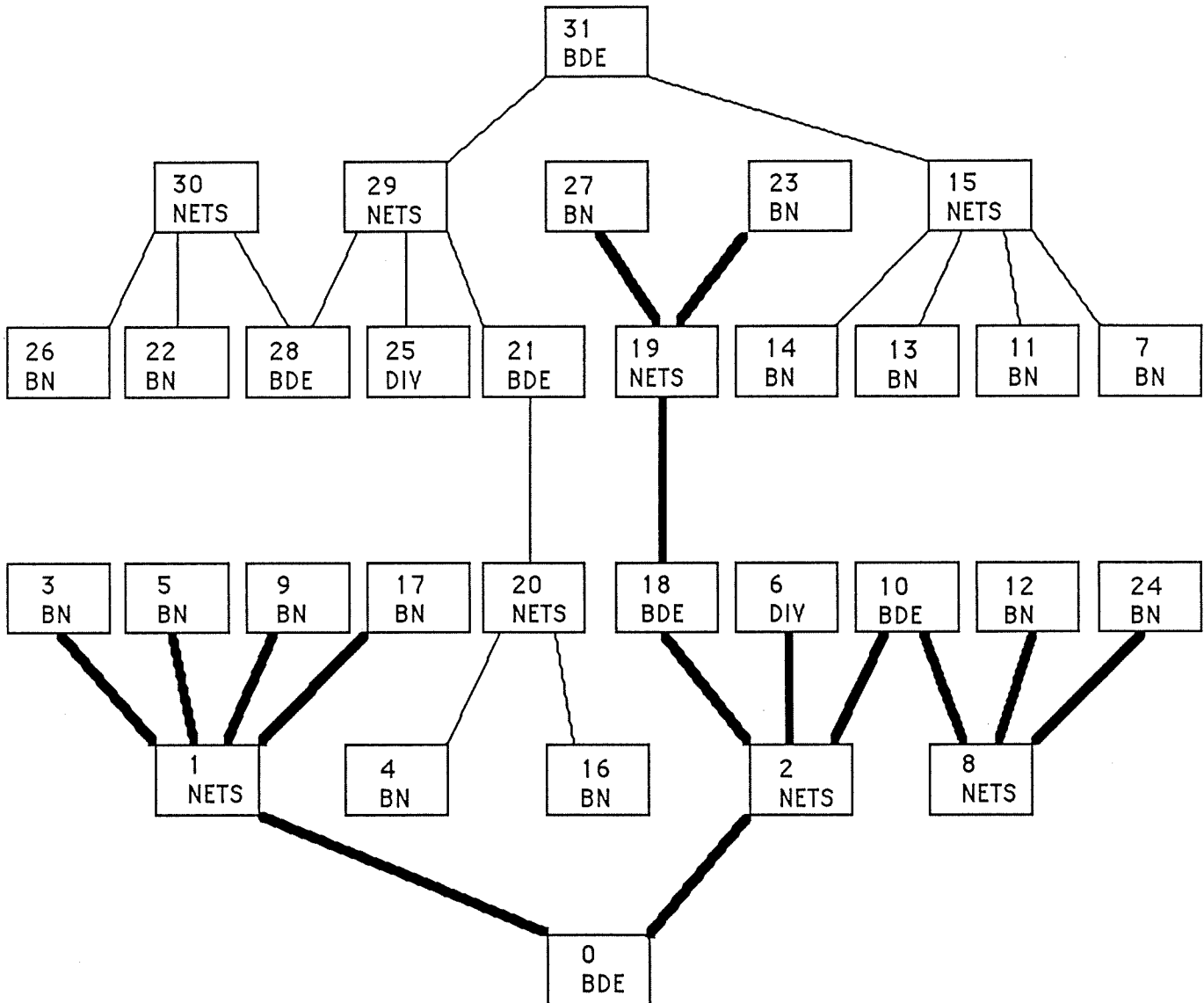


FIGURE 9: COMMO* OBJECT TO NODE (DIMENSION = 5)

Any two nodes in the same row of this picture have the same number of ones in their binary representation, hence they are the same distance from node zero.

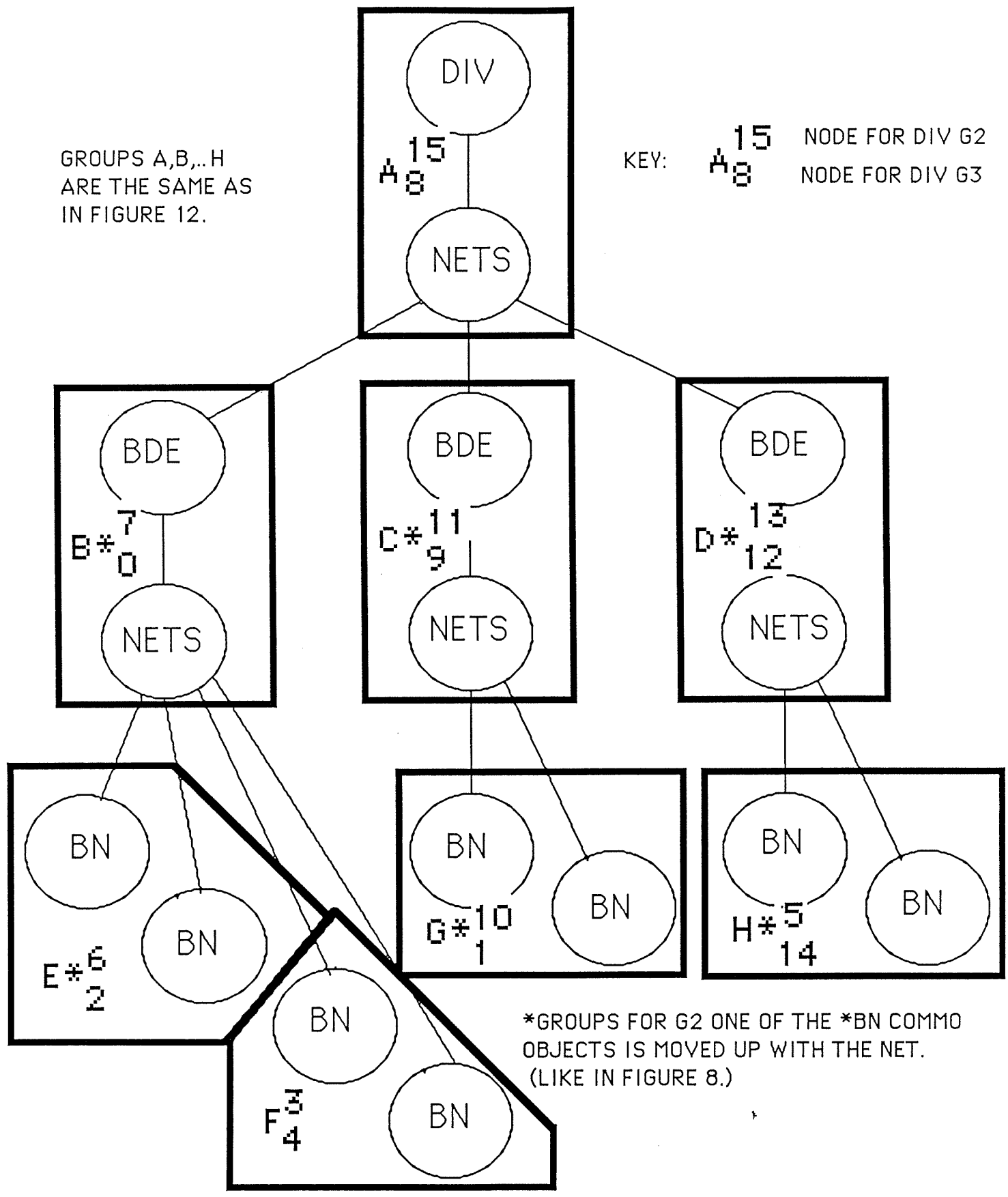


— DIVISION G3 CONNECTIONS
 — DIVISION G2 CONNECTIONS
 UNUSED HYPERCUBE CONNECTIONS ARE NOT SHOWN

FIGURE 10: COMMO* IN A 5-D HYPERCUBE

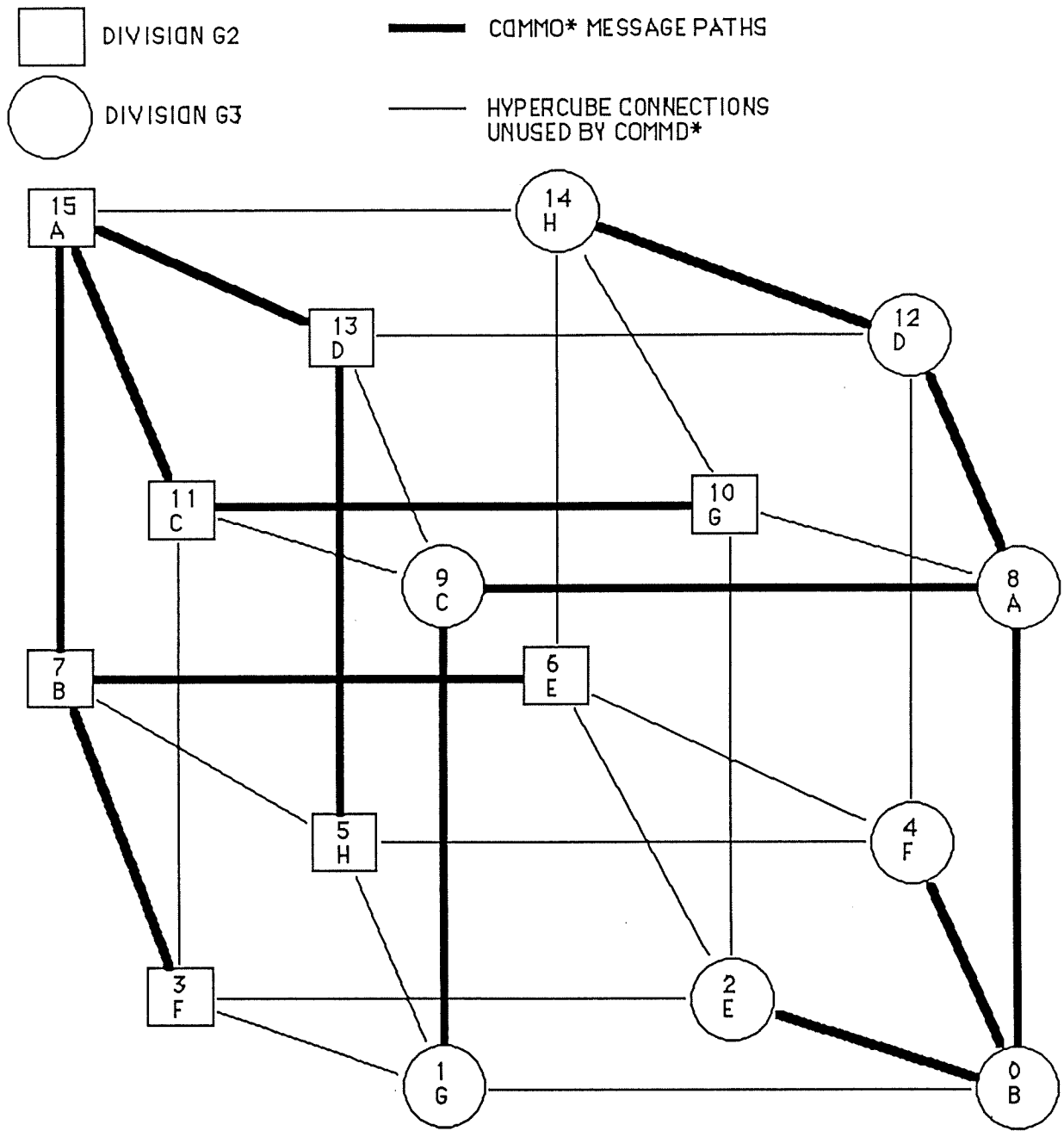
GROUPS A,B,..H
ARE THE SAME AS
IN FIGURE 12.

KEY: $\begin{matrix} 15 \\ A8 \end{matrix}$ NODE FOR DIV G2
 $\begin{matrix} 13 \\ D* \\ 12 \end{matrix}$ NODE FOR DIV G3



*GROUPS FOR G2 ONE OF THE *BN COMMO
OBJECTS IS MOVED UP WITH THE NET.
(LIKE IN FIGURE 8.)

FIGURE 11 COMMO* OBJECT TO NODE (DIMENSION = 4)



GROUPS A, B,...H ARE THE SAME AS IN FIGURE 11.

FIGURE 12: COMMO* IN A 4-D HYPERCUBE

GROUPS A,B,...D ARE THE SAME AS IN FIGURE 14.

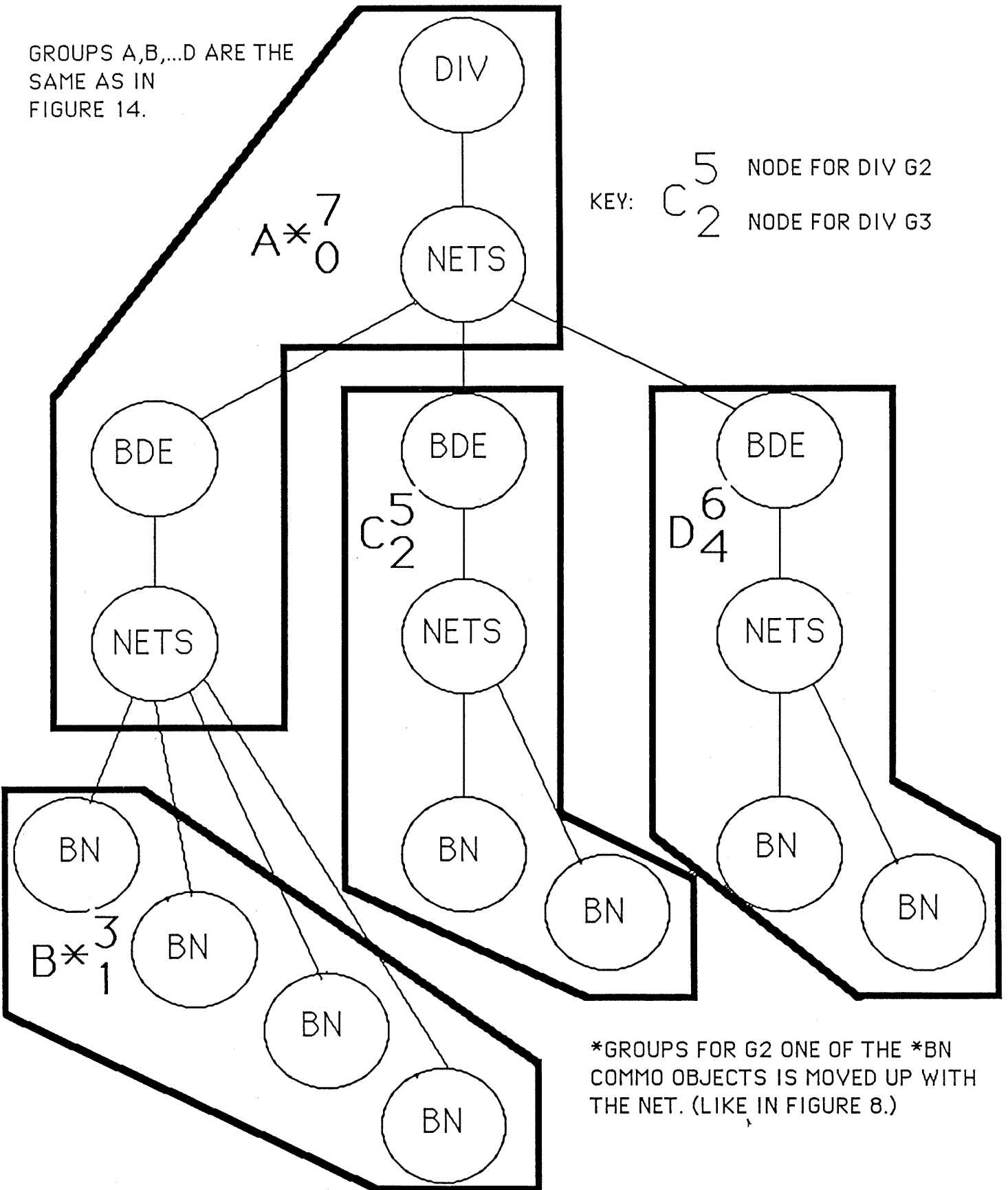
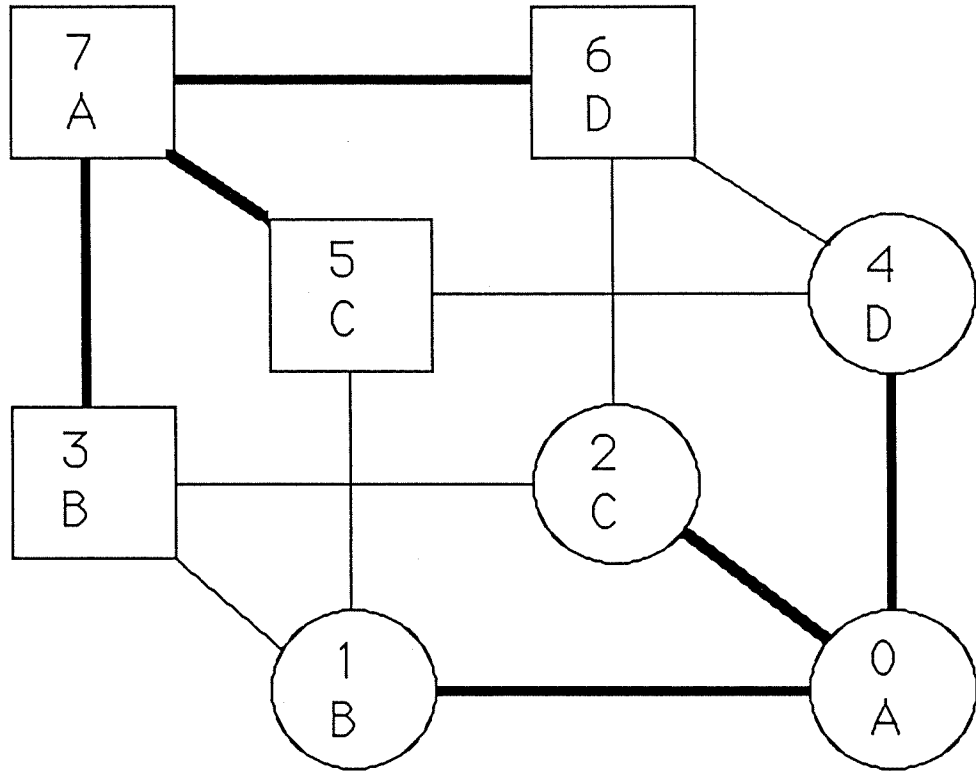


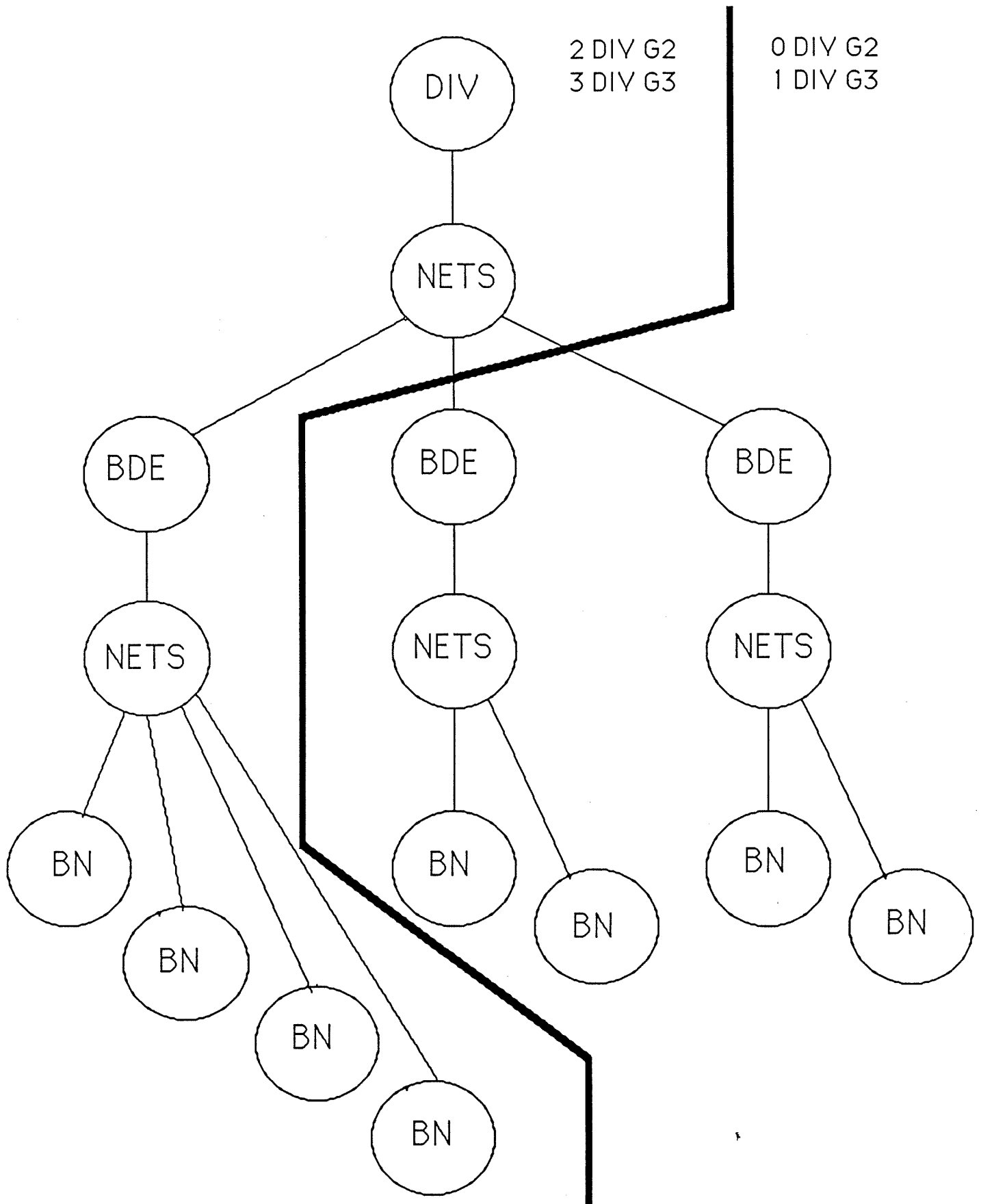
FIGURE 13: COMMO* OBJECT TO NODE (DIMENSION = 3)

GROUPS A, B, ...D
ARE THE SAME AS
IN FIGURE 13.



- KEY:
- CONNECTION USED BY COMMO*
 - UNUSED BY COMMO*
 - DIVISION G2 GROUPS
 - DIVISION G3 GROUPS

FIGURE 14: COMMO* GROUPS IN 3-D (HYPER?) CUBE



2 DIV G2
3 DIV G3

0 DIV G2
1 DIV G3

FIGURE 15: COMMO* OBJECT TO NODE (DIMENSION =2)