Compatibility Relation and its Application to Network Segmentation and Decentralization

1 D. Singh, 2 T.O William-West (Research Scholar)
1 Former Professor, IIT Bombay
2 Department of Mathematics, Ahmadu Bello University Zaria, Nigeria

Keywords: Compatibility relation, compatibility classes, maximal compatibility classes, minimal covering, network segmentation, network decentralization

ABSTRACT

In this paper, some new properties of compatibility classes of a finite set endowed with a suitable compatibility relation are described. An algorithm to compute maximal compatibility classes is constructed and an application of maximal compatibles to network segmentation and decentralization is demonstrated. Moreover, vector representation of compatibility classes is presented.

1. INTRODUCTION

A relation on a set which is reflexive and symmetric is called a compatibility relation (CR). It is difficult to trace since when the profound mathematics of compatibility relation came into existence. To our knowledge, Kurepa [12] seems to be the earliest full-blown mathematical exposition on the study of reflexive symmetric relations and graphs. Since then, a number of works ([16], [23], [15], [21], [5], [6], [7], [8], [11], [17], [4], [20], [14] and [19], etc.), dealing with fundamentals of compatibility relation as well as its applications, have appeared.

However, by now, it can be said that the actual import of compatibility relation has gone far beyond its ordinary linguistic connotation and even that of its mathematical characterization. It has found several applications in different fields of knowledge. Essentially, from application points of view, compatibility relation is useful in solving a class of minimization problems, particularly when the problems are incompletely specified:

(a) In switching theory, particularly for incompletely specified problems [16, 19];
(b) In incompletely specified sequential machines (ISSM) for reduction of the number of internal states [6];
(c) In designing of a class of digital control Units [2, 7];
(d) In graph theory [8];
(e) In solving some combinatorial problems, such as scheduling of traffic control [6];
(f) In phonology [17]; just to mention a few.

In this paper, we propose an alternative algorithm for computing MCCs and its application to network segmentation and decentralization.

2. COMPATIBILITY AND ITS CHARACTERISTIC PROPERTIES

Let \( S \) be a set with \( n \) elements, usually annotated \( \{x_1, x_2, \ldots, x_n\} \). A family \( \mathcal{A} = \{A_1, A_2, \ldots, A_k\} \) of non-empty subsets of \( S \) is called a covering of \( S \) if \( S = \bigcup_{i=1}^{k} A_i \). A cover of a finite set \( S \) is called minimal if none of its proper subclasses covers \( S \). Note that \( x_i \) 's are not necessarily disjoint and hence, it may not define a partition.

Notationally, a compatibility relation is sometimes denoted \( \preceq \). Also, if \( R \) be a compatibility relation on a set \( S \), then \( x \preceq y \) are called R-compatible or simply compatible to each other if \( x \preceq y \), i.e., \( x \) is an R-relative of \( y \). Note that compatibility relation, not being necessarily transitive, may not define a partition. However, it does define a covering ([18], for details).

Essentially, a compatibility relation defined on a finite set decomposes the set into its possibly pair wise non-disjoint subclasses, henceforth called compatibility classes (CCs). It follows that the elements of a CC are pair wise compatible (PC). Note also that every subclass of \( S \) may not be a CC, that is, elements of such subclasses are non-pair wise compatible (NPC).

Let \( S \) be an \( n \)-set \( \{x_1, x_2, \ldots, x_n\} \) and \( R \) a compatibility relation on \( S \). The \( x_i \)'s may be representing nodes in a network system or states in a finite machine or vertices in a graph. A subclass \( \mathcal{M} \subseteq S \) is called a maximal compatibility class (MCC) if any element of \( \mathcal{M} \) is compatible to its every other element and no other element of \( S - \mathcal{M} \) is compatible to all the elements of \( \mathcal{M} \). Equivalently, a compatibility class of \( S \) is maximal if it is not a proper subclass of any other compatibility class of \( S \). Graphically, MCCs for a given compatibility relation \( R \) can also be viewed as the largest complete polygons in the graph of \( R \). A polygon in which every node is connected to its every other node is called a complete polygon. A triangle is always a complete polygon and, for a quadrilateral to be a complete polygon, we need both the diagonals. Note that a complete polygon is a CC, which need not be an MCC, unless it satisfies the criteria of being so. Also, any element of the set that relates only to itself is an MCC, and any two elements of \( S \) which are compatible to one another but to no other elements of \( S \) form an MCC.
The following results hold:

(i) There must exist a family \( \{C_1, C_2, \ldots, C_n\} \) of non-empty CCs of \( S \) such that \( S = \bigcup_{i \leq n} C_i \). Moreover, the least number of CCs that cover \( S \) constitute a minimal cover of \( S \). It is easy to see that the number of minimal covers may be more than one.

(ii) Every element of \( S \) must be an element of at least one of the MCCs of \( S \). Also, whenever \( x \in R \), then \( \{x\} \) must be a subclass of at least one MCC of \( S \). Note, however that, in general, for a pair of compatible nodes, the inclusion of one of the nodes in an MCC does not necessarily imply the inclusion of the other.

(iii) Only CCs can be MCCs. More explicitly, for any compatibility class \( C \) of \( S \), either \( C \) itself is an MCC or \( C \) is a subclass of some other MCC of \( S \). It follows that a CC of \( S \) with a maximum cardinality is an MCC. Note that there may be more than one CC of the same cardinality and none or some or all of them are MCCs.

(iv) Let \( [M_1, M_2, \ldots, M_k] \) be a family of all MCCs of \( S \), then \( M_i \)s are pair wise incomparable with respect to \( \subseteq \) (subsethood).

(v) There must exist a family \( \{M_1, M_2, \ldots, M_k\} \) of non-empty CCs of \( S \) such that \( S = \bigcup_{i \leq k} M_i \). Moreover, the least number of CCs that cover \( S \) constitute a minimal cover of \( S \). Also, whenever \( x \in R \), then \( \{x\} \) must be a subclass of at least one MCC of \( S \). Note, however that, in general, for a pair of compatible nodes, the inclusion of one of the nodes in an MCC does not necessarily imply the inclusion of the other.

(vi) There must exist a family \( \{M_1, M_2, \ldots, M_k\} \) of non-empty CCs of \( S \) such that \( S = \bigcup_{i \leq k} M_i \). Moreover, the least number of CCs that cover \( S \) constitute a minimal cover of \( S \). Also, whenever \( x \in R \), then \( \{x\} \) must be a subclass of at least one MCC of \( S \). Note, however that, in general, for a pair of compatible nodes, the inclusion of one of the nodes in an MCC does not necessarily imply the inclusion of the other.

(vii) There must exist a family \( \{M_1, M_2, \ldots, M_k\} \) of non-empty CCs of \( S \) such that \( S = \bigcup_{i \leq k} M_i \). Moreover, the least number of CCs that cover \( S \) constitute a minimal cover of \( S \). Also, whenever \( x \in R \), then \( \{x\} \) must be a subclass of at least one MCC of \( S \). Note, however that, in general, for a pair of compatible nodes, the inclusion of one of the nodes in an MCC does not necessarily imply the inclusion of the other.

(viii) There must exist a family \( \{M_1, M_2, \ldots, M_k\} \) of non-empty CCs of \( S \) such that \( S = \bigcup_{i \leq k} M_i \). Moreover, the least number of CCs that cover \( S \) constitute a minimal cover of \( S \). Also, whenever \( x \in R \), then \( \{x\} \) must be a subclass of at least one MCC of \( S \). Note, however that, in general, for a pair of compatible nodes, the inclusion of one of the nodes in an MCC does not necessarily imply the inclusion of the other.

Theorem:

Corresponding to every minimal cover, there exists a unique compatibility relation.

Proof. Let \( R \) be a compatibility relation on an \( n \)-set \( S \), and \( \Gamma = \{M_1, M_2, \ldots, M_k\} \) be a minimal cover of \( S \). Now, since each element of \( M_i \in \Gamma \) is \( R \)-related to every other element of \( M_i \), and not \( R \)-related to all the elements of \( M_j, i \neq j \), it follows that all the elements of \( M_i \) \& \( M_j \) are elements of \( R \), \( i = 1, 2, \ldots, k \). Then

\[
R = \bigcup_{i=1}^{k} M_i \times M_i
\]

It is immediate to see that \( R \) is a compatibility relation. Let us consider the example of section II. Now, \( \Gamma = \{M_1, M_2\} \) where \( M_1 = \{x_1, x_2, x_3, x_4\} \), \( M_2 = \{x_2, x_3, x_4, x_5\} \). Then

\[
R = (M_1 \times M_1) \cup (M_2 \times M_2)
\]

Let \( R \) be a compatibility relation on \( S \) whose simplified graph of is represented in figure 1 below.
Remark 1

For an alternative proof of the aforesaid theorem, let us consider the example discussed above. It may be recalled that $M_1 = \{x_1, x_2, x_3\}$ and $M_2 = \{x_3, x_4, x_5\}$ are the only maximal compatibility classes which minimally covers the set $\{x_1, x_2, x_3, x_4, x_5\}$. We proceed as follows:

Compatibility table (CT) for maximal compatibility classes.

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_3$</th>
<th>$x_4$</th>
<th>$x_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

From CT, we list the compatibility classes viz:

$\{x_1, x_2, x_3\}$ and $\{x_3, x_4, x_5\}$

These are all the compatibility subclasses of level 1.

Now we generate compatibility subclasses of level 2, i.e., all 3-member subclasses from the class of 2-member subclasses viz;

$\{x_1, x_2, x_3\}$, $\{x_2, x_3, x_4\}$, $\{x_3, x_4, x_5\}$ and $\{x_4, x_5, x_1\}$

Next we generate the compatibility classes of level 3, i.e., all 4-member subclasses viz;

$\{x_1, x_2, x_3, x_4\}$ The guiding factor for the aforesaid construction is the compatibility of elements in $M_1$ and $M_2$.

Finally, the complete family of compatibility classes is exactly

$\{x_1, x_2, x_3, x_4, x_5\}$

Note that only complete polygons are elements of the family which give the compatibility relation.

Remark 2

Technique to represent MCCs for a given compatibility relation on a set $\{x_1, x_2, \ldots, x_n\}$

Akin to representing equivalence classes with the help of two vectors called FIRST and MEMBER (20), with a little extra effort, the MCCs can be represented. The steps are as follows:

The $i$th component of the FIRST for $x_i$ contains the number which is the first in the MCC to which $x_i$ belongs. The $i$th component of MEMBER contains the number which follows $i$ in the MCC, unless $i$ is the last element, in which case MEMBER $[i]$ is equal to zero.

In order to illustrate the aforesaid description, let us consider the example of section II, figure 1.

That is, the set is $\{x_1, x_2, x_3, x_4, x_5\}$ and the MCCs are $\{1, 2, 3, 4, 5\}$.

The vectors FIRST and MEMBER representing these MCCs are shown below:

<table>
<thead>
<tr>
<th>FIRST</th>
<th>Set</th>
<th>MEMBER</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>1, 2</td>
<td>2</td>
<td>3, 4</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>1, 2</td>
<td>4</td>
<td>0, 5</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>0</td>
</tr>
</tbody>
</table>

Again, as for equivalence relation, a number of applications of the aforesaid vector representation can be seen for compatibility relation.

For example, 3 and 5 are in the same MCC, since FIRST [3] = FIRST [5], and 1 and 3 are not in the same MCC, since the FIRST vectors are not the same. However, as we are dealing with compatibility, unlike equivalence classes, the MCCs may not be disjoint. Thus we need to modify the criteria followed in the case of equivalence classes. For example, in order to determine whether two elements of the set belong to the same MCC, we need to check if their FIRST vectors are equal or one is a part of the other.

Also, the number of MCCs will be equal to the number of MEMBER vectors containing zero (0) or containing elements of which 0 is one. For example, in our illustration discussed in section II, the number of MCCs is two.

Similar to the procedure for constructing equivalence classes, the aforesaid representation can be applied to construct an algorithm to compute compatibility classes of a given compatibility relation on a finite set.

3. THE COMPATIBILITY MATRIX (CM) OF A COMPATIBILITY RELATION R AND CONSTRUCTION OF AN ALGORITHM TO COMPUTE CCs/ MCCs

Abstracting from [16], [5] and [1], we present a simplified algorithm to compute CCs/ MCCs as follows. In order to construct the CM of $R$, for each pair of nodes $(x_i, x_j)$, a one (1) is assigned to it if $x_i R x_j$, and a zero (0) if it is not the case that $x_i R x_j$. Consequently, the table for the CM has an all – 1 leading diagonal (due to reflexively of $R$). Moreover, since the CM of $R$ is symmetric, it is sufficient to tabulate only the elements of its lower
triangular parts. However, for a better comprehension, we will consider the complete matrix for discussion in this paper.

**Step1:** Compute all CMCCs using matrix table (as in [1], but without columns for their generators and weights).

**Step2:** Begin with the CMCC of row 1. If it is of cardinality 1 or 2, list it in the column of CCs. The same holds for all other rows of the compatibility table.

**Step3:** If the CMCC in consideration is of cardinality greater than 2, check whether or not there exists a zero (0) at any intersection of rows and columns of the sub matrix generated by the constituent states of that CMCC i.e., the sub matrix is an all-1 sub matrix or not. If the sub matrix is all-1, list it in the column for CCs.

**Step4:** If the matrix in step 3 is not all-1, ignore the row(s) and column(s) containing the largest number of zeros and list the states which produce the truncated matrix, which is an all-1, in the column for CCs. If not, repeat step 4 until an all-1 matrix is obtained. In such cases, there may be two different CCs.

**Step5:** Similar to all other existing algorithms, proceed as above for all the rows exhaustively.

**Step6:** Once the column of CCs are completed, delete all CCs which are proper subsets of some other CC. Also, if repeated CCs occur, delete all such CCs except one of them. The remaining CCs are distinct MCCs.

This approach takes care of deleting superfluous states of the system, and, in turn, makes the construction of (minimal) cover simpler.

The significance of the proposed approach is many-fold. For example, a system designer would preferably work for a compatibility relation on a given set which gives rise to a family of minimum number of MCCs that covers S.

For an illustration of the above algorithm, let \( R \) be a compatibility relation on a set \( S = \{X_1, X_2, \ldots, X_n\} \) defined by the compatibility matrix (CM) as follows (note that the elements of a CMCC may be PC or NPC):

<table>
<thead>
<tr>
<th>x_1</th>
<th>x_2</th>
<th>x_3</th>
<th>x_4</th>
<th>x_5</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

**Remark 3**

The compatibility table to identify CMCCs is adopted from [1]. Although, the paper does not use the notion of generator of a CMCC, the following observations are noted below

From definition 2 ([1], p.214) of the generating state of a CMCC in a given ISSM viz,
Thus, in order that a CMCC has a unique generator (and, that is what perhaps the author intends to), the aforesaid definition needs to be modified as follows: “The state of a CMCC, which is compatible with each of its states and with no other state of its complement in the given ISSM, is called its generating state”. Moreover, the authors finding (p.218) that a “22-state machine will have 30 MCCs” is valid only if the repeated MCCs as well are counted (see (ix), section II).

4. APPLICATION OF MAXIMAL COMPATIBLES TO NETWORK SEGMENTATION

4.1 Preliminaries

Packet queues are buckets that hold data for a network device when it cannot be immediately transmitted. Most packet queues use FIFO (first in, first out), unless they have been specially configured to do otherwise. In other words, when the packet queue for a device is completely full, the packet most recently placed in the queue will be sent over the device only after all the other packets in the queue at that time have been sent.

A packet is one unit of binary data capable of being routed through a computer network. In order to improve upon communication performance and reliability, each message sent between two devices is often subdivided into packets by the underlying hardware and software.

A network switch is a small hardware device that joins multiple computers together within one local area network (LAN). Technically, network switches operate at layer two (Data link layer) of the OSI model.

Switches look at each packet for conformity and correctness and process it accordingly rather than merely repeating the signals to all ports. Switches assign Ethernet addresses of nodes residing on each part of the network and then allow only necessary traffic to pass through. Switches hinder misaligned packets from spreading by not forwarding them.

When a packet is received by a switch, it inspects the destination and source hardware addresses and compares them to a table of network segments and addresses. If the segments are the same, the packet is dropped; if the segments are different, then the packet is forwarded to the proper segment. In [3], it was observed that dropping of packets and regeneration of forwarded packets enable switching technology to split a network into separate collision domains. In a switched network, each segment is an independent collision domain.

In order to properly describe the topology of a network, it needs to be emphasized that introducing switches in a congested network serves the purpose different from that of increasing the number of switched ports already installed. A switch installed in a location where it forwards most of the traffic it receives, will be less helpful than the one that filters them.

In particular, the goal of introducing switches in a network is to minimize its congestion, if any. A network which is not congested needs no introduction of switches [3].

![Switched network.](image)

In a network system, issues pertaining to network congestion are encountered due to heavy traffic on the network or increase in the number of users sharing it. This can reasonably slow down the performance of the network [10].

More often than not, more data are required to be added to a shared network. As a result, performance deteriorates due to competition for Ethernet bus by users (competitors) of the shared network [3].

In the event of problems occurring with too many nodes in the same collision domain, suitable network segmentation is introduced to reduce collision, retransmission and contention for bandwidth [3]. Usually, in a segmented network, most of the communication activities are performed by the subnets. At times, the sub-network gets flooded with messages reporting status and throughput statistics to the Network Monitoring Centre (NMC) programs, and as users’ traffic increases, larger network capacity is consumed. Not very frequently, lack of provision for coordinating and analyzing performance measures at terminals in the sub-network as well as at the host tell upon the efficiency of the network segmentation method (NSM).

Network segmentation is an act of splitting a computer network into subnets comprehended as network segments or network layers. A segmentation of a congested network consists in its decomposition into smaller segments which give rise to a sort of partition. Essentially, a physically separate path (not intersecting with another path) for each pair of communicating ends needs to be built, which tells upon efficiency as well as costing.

We propose that the notion of covering instead of partitioning a congested network may be applied to obtain a more compact and cost effective network. Thus, in order to achieve a competing segmentation of a congested network, a suitable compatibility relation, instead of an equivalence relation, needs to be defined. This, in turn, gives rise to a decomposition of the network into (maximal) compatibility classes (segments) whose union
is its minimal covering. It is observed that this procedure does not disconnect any segment or sub-network from the entire network, since the pair wise intersection of these (maximal) subnets is non-empty.

Essentially, a compatibility relation based segmentation procedure segments a congested network into subnets (maximal compatibility blocks). This, in turn, structures the network in such a way that no node of any given subnet communicates to every node of another subnet. As a result, the communicating network is reasonably minimized without adversely affecting its goal. In fact, when the network is segmented, there is a tradeoff between time and quality data. That is, if a particular node $x_i$ in a subnet $A$ does not relate to a node $x_j$ in subnet $B$, but needs information from node $x_k$, then time is traded off to obtain that information from $x_k$. Whenever $x_i$ relates to node $x_j$ in $B$. This is because the rate of packet delivery from $x_i$ (source) to $x_j$ (destination) is equal to the rate of packet delivery from $x_k$ to $x_j$ plus some time $t > 0$. However, there is a compensation for this delay. The compensation being that the maximal compatibility class (maximal subnet) through this node $x_k$ (source) lowers its throughput to meet that of a (possible) congested path. This type of congestion control is similar to rate-based technique.

Further, it is observed that $x_k$ above is a link from subnet $A$ to subnet $B$. Therefore, such $x_k$ in any subnet supports high bandwidth and low latency (the amount of time taken for a packet to travel from source to destination) between subnets $A$ and $B$, since $x_k$ can be reached.

A schematic representation of how a CR-based network congestion control procedure works:

Let $N$ be a network system on which a compatibility relation is defined with compatibility classes $C_k; k = 1, 2, \ldots$

Label $x_k$ as $L_k$ (links) if for some compatibility classes $C_i$ and $C_j$, we have $C_i \cap C_j \neq \emptyset$ for some $i, j, k$. These $L_k$’s in $N$ form its backbone.

Now, the packets are routed through the network. It is important to observe that the performance measure of a refined network will be a function of the cardinality of the set of $L_k$’s.

Further, there exists at least one node in $C_k$ which maintains a separate queue for any other subnet $C_i$ and a separate queue for nodes in itself.

Denote the first queue as $q_i$ and the second queue as $q_j$. Then any node within a compatibility class $C_i$ maintains a separate $q_i$ for each $L_k$ in compatibility class $C_j$ and maintains $q_i$ for nodes in any other compatibility class $C_j$. Also, it follows that each $L_k \subseteq N$ maintains $q_i$ for some $C_j$ (i.e.,) in $N$. Therefore, communication flow can be re-channelized and assigned to the appropriate $L_k$’s with the information about the network traffic and topology. In fact, accounting for the instantaneous queue length between $L_k$’s, controller (which implements the electronic circuitry required to communicate using a specific physical layer and data link layer standard, such as Ethernet, Wi-Fi, etc.) allocates packets to the least congested link.

The following diagram illustrates the concept described above.

Fig 3: Diagram of links and queues.
The diagram (fig. 2) represents the links and queues.

5. APPLICATION OF MAXIMAL COMPATIBLES TO NETWORK DECENTRALIZATION

Usually, a computer network connects nodes (computing devices) and supports communication between these nodes. The network uses different schemes to create these connections. The scheme used is a function of the size of the network, services offered, etc. A decentralized network is such a scheme commonly used to create and support a network.

In a decentralized network system, issues pertaining to distribution of control versus optimum communication flow have been drawing attention since long: manageability becomes constrained as the system grows larger ([9], [23]).

A decentralized network is a scheme that allocates resources (both hardware and software) to each individual work station. It aims at connecting users and resources in a transparent, open, and scalable manner. This is achieved by creating multiple locations that support different operating systems in order to avoid occurrences of a complete neutralization caused by any
form of attack ([22] provides a diagrammatic description of a decentralized network).

In order to properly describe the topology of a decentralized network, it needs to be emphasized that the main goal of a decentralized network is to connect users and resources in a transparent, open and scalable fashion. Essentially, a network which demands significantly high bandwidth, latency or communication requirements need not be decentralized. However, the choice of a decentralized network depends on the needs of a specified application.

Schematically, the graph of a compatibility relation $R$ of fig. I may be viewed as a decentralized network if, each node is a client and a server as well. It is observed that the symmetric property of a compatibility relation guarantees each node in the graph of $R$ to assume client–server functions. Since the topology of $R$, viewed as a decentralized network, is identical to a mesh topology, there is reasonably enhanced reliability and speed in a decentralized network. Also, it is observed that this advantage grows significantly when the decentralized network is configured with redundant servers (nodes) - multiple nodes providing the same service.

Thus, by applying the notion of MCCs, the distribution of control to nodes would be reasonably minimized without adversely affecting the goal of network decentralization.

As discussed above, introduction of compatibility relation reasonably facilitates the distribution of control to nodes without adversely affecting the goal of decentralizing a network. We analyze some characteristics for the topology of a compatibility-based decentralized network. The intent is to develop a broad framework for constructing network design that can be applied to some specific domains.

In what follows, considering some typical criteria used for performance, we put forth analysis of decentralized network, pertaining to compatibility-based decentralized network:

i. Manageability: It may be easier to control various functions (updating, repairing, logging etc.) of a system.

ii. Fault tolerance: The ability of the system to handle failure is enhanced: failure of some particular node(s) does not drastically hinder the functioning of the rest of the network.

iii. Extendibility: Adding new resource to enlarge the system may be transparent: as any node can be introduced into the network to share files with other nodes.

iv. Scalability: How large could the system grow? In reality, this characteristics is quite complex. In practice, any algorithm required to keep a decentralized network coherent is bound to carry a lot of overheads. Consequent to the cumulative growth of such overheads, the network may not scale.

6. CONCLUDING REMARKS

Concluding, it can be emphasized that in a foreseeable future, besides in hard sciences like Engineering, Computer sciences, etc., in order to model problems in soft sciences like biology, economics, sociology, etc., which are invariably supposed to deal with a large number of incompletely specified interactions, applications of compatibility relation would play an important role.
REFERENCES


