Theories of Hypergraph-Graph (HG(2)) Data Structure

Shiladitya Munshi
Meghnad Saha Institute of Technology
Kolkata, India
Distributed Computing research Lab
Golfgreen, Kolkata: 700095, India
Email: shiladitya.munshi@yahoo.com

Ayan Chakraborty
Techno India College of Technology
Kolkata, India
Distributed Computing research Lab
Golfgreen, Kolkata: 700095, India
Email: achakraborty.tict@gmail.com

Debjayoti Mukhopadhyay
Maharashtra Institute of Technology
Pune, India
Distributed Computing research Lab
Golfgreen, Kolkata: 700095, India
Email: debjayoti.mukhopadhyay@gmail.com

Abstract—Current paper introduces a Hypergraph-Graph (HG(2)) model of data storage which can be represented as a hybrid data structure based on Hypergraph and Graph. The proposed data structure is claimed to realize complex combinatorial structures. The formal definition of the data structure is presented along with the proper justification from real world scenarios. The paper reports some elementary concepts of Hypergraph and presents theoretical aspects of the proposed data structure including the concepts of Path, Cycle etc. The detailed analysis of weighted HG(2) is presented along with discussions on Cost involved with HG(2) paths. Keywords—Hypergraph, Graph, Hyperpath, Hyperedges, HG(2), Cost of HG(2) Path

I. INTRODUCTION

Hypergraphs, a generalization of Graph, is being widely and deeply investigated since last few decades as a successful tool to represent and model complex concepts and structures in various areas of Computer Science and Discrete Mathematics. Graphs on the other hand have enjoyed the spotlight of heterogeneous research as an important and implementable tool to represent and analyze real world complex scenarios.

Hence, both Hypergraphs and Graphs are the well suited candidates for representing complex structures having logical or real world relations with certain advantages over each other. While Hypergraphs have more potential than graphs in representing complex scenarios, it lacks readability, simplicity and easiness in conceptualization and physical representation.

In the current paper, a new data structure "Hypergraph - Graph" (HG(2)) has been introduced, which takes the advantages of both the data structures of Graph and Hypergraph. This data structure can model a problem space into two logical partitions with different levels of complexities. The more complex level can be represented by Hypergraph in order to take the leverages of its power to designate the relationships among a group of objects. The other level which is characterized by comparatively lesser complexity and better orderliness, could be represented by Graph in order to take the advantage of its naturalness and simplicity.

With a high objective of introducing HG(2), the present paper has been organized as follows - Section 2 proposes HG(2) and related theoretical aspects formally with definitions and illustrative examples. Section 3 reports the concepts of paths within HG(2) with some core related issues, followed by discussions on the impact of weights on HG(2) and computation of Cost of any HG(2) Path in Section 4. Lastly, Section 5 concludes the current investigation emphasizing the level of contribution, importance and future research scope related with study of HG(2).

II. HYPERGRAPH - GRAPH (HG(2)) DATA STRUCTURE

Hypergraph - Graph data structure denoted as HG(2) is conceptualized as a model to represent a complex problem space based on certain criteria. The criteria could be formalized as follows - The problem space (P S) must logically be divided into two levels with different complexities, one (P SG) with relatively lesser complexities, better orderliness and bounded by formalized set of rules, and another (P SH) which could be characterized by greater complexities and absence of ordered rule sets.

The some or all interrelationship between objects of P SH must be dictated by the objects of P SG and even the behavior of some or all objects of P SH must be defined by the objects of P SG. Here the term "object" is being used informally and must not necessarily indicate any Object Oriented paradigm. As the complex real life combinatorial structures are not rare at all, proposed HG(2) has an intrinsic objective to represent P SH with Hypergraphs and P SG with Graphs. The inter dependencies of P SH and P SG form the basis of evolution of the behavior of HG(2) as a whole.

The theories behind the Hypergraph Data Structure are presented in [5] and due to space constraint, it is omitted in the current discussion. On this background, following subsection presents the theories of Hypergraph - Graph (HG(2)) data structure directly.

A. Introducing HG(2)

A Hypergraph-graph data structure HG(2) is a triple denoted as HG(2) = (H, G, C) where H is a Hypergraph, G is a graph and C is a set of connectors.

H is a Hypergraph defined as H = (V^h, E^h), where V^h = v_1^h, v_2^h, \ldots, v_n^h, n = 2, 3, \ldots; and E^h = E_1^h, E_2^h, \ldots, E_m^h.
G is a Graph defined as \( G = (V^g, E^g) \), where \( V^g = v^g_1, v^g_2, \ldots \) and \( E^g = e^g_{p=1, 2, \ldots} \), \( q=2, 3, \ldots \);

where each \( e^g_i \) could be expressed in the form of \( e^g_{xy} \) which connects \( v^g_x \) from \( v^g_y \).

\( C \) is a set of connectors, which could be conceptualized as a set of all the dependencies between \( P \), \( SH \), and \( P \), \( SG \) (as described earlier) which are characterized by \( H \) and \( P \) respectively.

Here we define two types of connectors:

(a) \( c^v_{xy} \) which connects a node in the Graph \( v^g_x \) from a node in the Hypergraph \( v^h_x \). It is to note that the behavioral dependency of an object of \( P \) \( SH \) on an object of \( P \) \( SG \) gets realized through \( c^v_{xy} \); and

(b) \( c^e_{xy} \) which connects a node in the Graph \( v^g_x \) from a Hyperedge \( E^h_y \). Here \( c^e_{xy} \) realizes the dependencies of collective behavior (bound with a specific relation) of the objects of \( P \) \( SH \) on the objects of \( P \) \( SG \).

The set of all \( c^v_{xy} \)s is noted as \( C^v \) while \( C^e \) represents all \( c^e_{xy} \)s. Hence on the basis of ongoing discussion, it could be concluded that \( C = (C^v, C^e) \). For the rest of the paper, it is assumed for simplicity that the dependency flows from \( P \) \( SH \) to \( P \) \( SG \). That is the Hypergraph layer is dependent on the Graph layer. No dependency flows through a connector backward from the Graph layer to the Hypergraph layer. Above discussion can be illustrated with the example as shown in Fig 1. In this figure, an HG(2) is shown to have a Hypergraph \( H \) and a Graph \( G \). The \( H \) and the \( G \) are connected with connectors \( C \). The Hypergraph \( H \) is composed of \( (V^h, E^h) \) where \( V^h \) consists nodes 1, 2, 3, 4, 5, 6 & 7 and \( E^h \) consists \( E_1, E_2, E_3 \) & \( E_4 \). Further, the Graph \( G \) is composed of nodes \( a, b, c, d, e \) & \( f \). For simplicity the edges of the \( G \) are not highlighted.

It is to be noted that \( E_1 \) is composed of nodes 1, 2, 3; \( E_2 \) is composed of nodes 4, 5, 6; \( E_3 \) is composed of nodes 4, 5, 7 and finally \( E_4 \) is composed of nodes 5, 6, and 7.

Following is the identification of head and tail nodes for each Hyperedge:

\( H(E_1) = 1, 2 \) and \( T(E_1) = 3 \)
\( H(E_2) = 3, 4 \) and \( T(E_2) = 5, 6 \)
\( H(E_3) = 4, 5 \) and \( T(E_3) = 7 \)
\( H(E_4) = 5, 6 \) and \( T(E_4) = 7 \)

The Fig 1 further illustrates that there are three connectors (marked with dashed square ended bi-directional arrow) that connect Hyperedge nodes and Graph nodes. They are \( c_{1a}^h = c_1, c_{6b}^h = c_2 \) and \( c_{2a}^h = c_3 \). Moreover there are three connectors (marked with bold bidirectional arrow) that connect a Hyperedge with a Graph node. They are identified as \( c_{3a}^g = d_1, c_{3b}^g = d_2 \) and \( c_{4f}^g = d_3 \). Hence formally \( C \) which is a pair \((C^v, C^e)\) holds the following:

\( C^v = c_1, c_2 \) and \( c_3 \) and \( C^e = d_1, d_2 \) and \( d_3 \).

Though the diagram shows the individual connectors with bidirectional arrow, the earlier assumption still holds that the direction of the dependency is from Hypergraph to Graph layer. The bidirectional arrows have been given for easy diagrammatic identification of connectors out of many different edges.

However, Fig 1 shows the entire problem space to be logically divided (by a dashed horizontal divider) into \( P \) \( SG \) and \( P \) \( SH \) which are represented by the Graph and Hypergraph respectively.

On the basis of current discussion, next section reports critical issues related with concept of Path in HG(2).

III. CONCEPT OF PATH IN HG(2)

The study of Paths in HG(2) and the related algorithms demand formalization of certain essential terminologies which are presented next.

A node dependent pair or simply the node pair is defined as the pair of Hyperedges and graph nodes which share a dependency relation through \( c^v_{xy} \) and is denoted as \( v^g_y(h^h_x) \). In Fig 1, \( 1(a), 6(b) \) and \( 5(d) \) could be identified as three node pairs. It is to be noted that, in context to a Hypergraph node not involved in \( C^v \), the node pair is symbolized as \( v^g_y \). The node pair of a Hypergraph node \( n \) is designated as \( NP^h_n \).

Similarly an edge dependent pair or simple edge pair is defined as the pair of Hyperedges and graph nodes which share a dependency relation through \( c^e_{xy} \) and is denoted as \( E^h_x(v^g_y) \). Any edge pair in context to a Hyperedge not involved in \( C^e \) is symbolized as \( E^h_x \). In Fig 1, three edge pairs could be identified as \( E_1(c), E_3(e) \) and \( E_4(f) \). However, the edge pair of a Hyperedge \( n \) is designated as \( EP^h_n \).

Based on the terminologies discussed above, A path \( P_{st}^{(\alpha, \beta, \gamma)} \) in HG(2) (known as HG(2) path of length \( q \)) could be defined as a sequence of node pair and edge pair. That is \( P_{st}^{(\alpha, \beta, \gamma)} = NP^h_{s,t}, NP^h_{t+1,t}, \ldots, EP^h_{q-1, q} \); where following conditions are all true:

i) \( P_{st}^{(\alpha, \beta, \gamma)} \) is a valid Hyperpath considering only the Hyperedges in \( H \)

ii) each \( NP^h_{xy} = v^g_y(h^h_x) \) and each \( EP^h_{xy} = E^g_x(v^g_y) \), \( i \) and \( x = 2, 3, \ldots \)
iii) for any NP$^v_i$, $EP^E_i$ pair, there exists a valid path from $v^g_i$ to $v^g_j$ in $G$
iv) for any $EP^E_i$, NP$^h_i$ pair, there exists a valid path from $v^g_i$ to $v^g_j$ in $G$

A path in Graph $G$ traced by the Hyperpath in Hypergraph $H$ is defined as Graph Path or simply Graph Path. A Graph Path is formed by the graph nodes involved in constituent node pairs and edge pairs (in sequence) of a HG(2) path.

An HG(2) path $P_{st}^{HG(2)}$ is said to contain a Graph Loop or simply GLoop if there exists a cycle in GPath.

An HG(2) path $P_{st}^{HG(2)}$ is said to contain a Hypergraph Loop or simply HLoops if there exists a cycle in Hyperpath considering the Hypergraph only.

An HG(2) path $P_{st}^{HG(2)}$ containing GLoop must not necessarily contain HLoop and the vice-versa is also true. Even the Gloop and Hloop may co-exist in a single $P_{st}^{HG(2)}$.

An HG(2) path $P_{st}^{HG(2)}$ is qualified as Elementary if all the nodes in the Hyperpath are distinct, and the same is termed as Simple if all the Hyperedges are distinct.

However, the idea regarding HG(2) path, that evolved up from the ongoing study, could be illustrated with the example shown in Fig 1.

Let us consider an HG(2) path $P_{17}^{HG(2)}$. This path starts from Hypergraph node 1 ($V_1^s = 1$) and reaches another Hypergraph node 7 ($V_7^s = G$). $P_{17}^{HG(2)}$ follows two Hyperpaths The nodes and edges of the first and second Hyperpaths result sequences of node pair and edge pair as shown below:

(1) $\{a, \ (c, 3), \ E_2(\ ), \ 5(d), \ E_3(e), \ T(\ )\}$ and;
(1) $\{a, \ E_1(c), \ 3(\ ), \ E_2(\ ), \ 6(b), \ E_4(f), \ T(\ )\}$

Both these HG(2) paths ensure that the every following graph node in the sequence is connected to its previous node. For example in $\{a, c, d, e\}$, c is connected to a, d is connected to c (through b) and e is connected to d. That is $\{a, c, b, d, e\}$ and $\{a, c, d, b\}$ form two valid paths in Graph G. Hence these two paths can be termed as Graph Path.

There exists another Hyperpath $\{1, \ E_2(\ ), \ 3, \ E_3(\ ), \ 5, \ E_7(\ )\}$ that can be followed for $P_{17}^{HG(2)}$. The corresponding node pair and edge pair sequence is generated as

(1) $\{a, \ E_1(c), \ 3(\ ), \ E_2(\ ), \ 5(d), \ E_4(f), \ T(\ )\}$ and the sequence of Graph nodes corresponding the node pair and edge pair sequence is generated as $\{a, c, d, f\}$. It is to be noted that node f is not connected to node d and hence this sequence cannot form a valid path in the Graph G. Hence (1) $\{a, \ E_1(c), \ 3(\ ), \ E_2(\ ), \ 5(d), \ E_4(f), \ T(\ )\}$ is not a valid HG(2) path.

A path in a HG(2), thus, as seen in the above discussion, a valid Hyperpath with some additional constraints. These additional constraints can be attributed to the Graph Path. Hence it is to be considered that an HG(2) path can have greater potential than a Hypergraph to model sequence of complex relationships.

On this background, the next section discusses the notions of Weighted HG(2) and consequently the cost involved in HG(2) paths.

IV. COST ANALYSIS OF PATH ON A WEIGHTED HG(2)

A Weighted HG(2) is composed of a weighted Hypergraph, a weighted Graph and weighted Connectors. In this investigation, the weight simply means the weight of edges that means the cost of connecting nodes within individual Graph, Hypergraph, nodes of Graph An Hypergraph and nodes of Graph and edges of Hypergraph. The weight of graph edges that represents the cost of connecting two Graph nodes $v^g_i$ and $v^g_j$ is denoted by $C_{ij}^e$ or $C_{ij}^h$, where $e^h_i$ connects node i and node j. Similarly, the weight of hyperedge that represents the cost of connecting $v^h_i$, $v^h_j$, $v^h_n$, related by hyperedge $E^h_{ij}$ is denoted by $C_{ij}^h$. The weight of node to node connector $c^h_{ij}$ is the cost of connecting a node of Hypergraph $v^g_i$ and a node of graph $v^g_j$, and it is denoted as $C_{ij}^h$. Hence $C_{ij}^h$ will determine the weight of edge to node connector $c^h_{ij}$ which connect $E^h_i$ to $c^h_{ij}$. Based on this discussion, the cost associated with a path can be viewed as follows.

Before proceeding any further, it is to be noted that any HG(2) Path may give rise to multiple valid Hyperpath routes. For example, HG(2) Path $P_{17}^{HG(2)}$ in the HG(2) shown in Fig 3 gives rise three valid Hyperpath routes $\{1, E_1(3), E_2(5), E_3(7)\}$, $\{1, E_1(3), E_2(6), E_4(7)\}$ and $\{1, E_1(3), E_2(6), E_4(7)\}$. Consequently we get two valid HG(2) route out of these three Hyperpath routes as:

$\{1, a, E_1(c), 3(\ ), E_2(\ ), 5(d), E_3(e), T(\ )\}$ and;
$\{1, a, E_1(c), 3(\ ), E_2(\ ), 6(b), E_4(f), T(\ )\}$

Each of these valid HG(2) routes will be denoted by $R_{ij}^{HG(2)}$, and $R_{pq}^{HG(2)}$ for HG(2) path $P_{ij}^{HG(2)}$.

It is to be further noted that for each individual valid HG(2) route, there may exist multiple Graph Paths. Each of these Graph paths are the valid connectors among the subgraph of G which has been induced by corresponding $c^h_{ij}$ and $c^h_{pq}$ of the chosen HG(2) route $R_{ij}^{HG(2)}$. Hence the notation $GP_{pq}$ will denote the subgraph of G induced by the valid HG(2) route $R_{ij}^{HG(2)}$ of $\{1, a, E_1(c), 3(\ ), E_2(\ ), 5(d), E_3(e), T(\ )\}$.

$\{1, a, E_1(c), 3(\ ), E_2(\ ), 5(d), E_3(e), T(\ )\}$ An $G$ Hyperpath $P_{st}^{HG(2)}$ on $G_{pq}$ denotes a path from graph node $p$ to graph node $q$ where $p$ and $q$ are the first and last graph nodes encountered tracing a HG(2) Path $P_{st}^{HG(2)}$. The $G$ Hyperpath for graph node pair $p$ and $q$ is thus denoted by $GP_{pq}$.

The simplified concept of HG(2) Path presented in the previous section could be represented by single or multiple HG(2) routes where each such HG(2) routes can be traced with single or multiple Graph Paths. The cost of HG(2) Path hence will be dependent on cost of HG(2) routes and cost of Graph Path.

However Cost of HG(2) route is the summation of cost of Hyperedges that is $C_{ij}^{h_{pq}} = \sum v^h_i$ where $E^h_i$ to $E^h_n$ are the Hyperedges that constitute the Hyperpath of the HG(2)
route. Similarly the cost of GPath \( C_{GP_{pq}} = \sum_{i=0}^{n} C_{e_{i}} \) where \( e_{i} \) to \( e_{n} \) constitute the GPath.

This paragraph will investigate the effects of cost of connectors into the computation of cost of HG(2) Path. While considering \( GP_{pq} \), there might be some nodes which are connected with Hypernodes and Hyperedges for the Hyperpath selected, and at the same time there might exist some node pairs which are not connected with the any of the Hypernodes or Hyperedges, rather they got traced in order to establishing a valid path between two consecutive graph nodes of node pair and edge pair (discussed in previous section).

Let us consider the case as shown in Fig 3. For the HG(2) Path \( P_{st}^{HG(2)} \) and for G(2) route \( R_{i}^{HG(2)} \), node \( b \) got selected to establish the path between node \( c \) and node \( d \). Here node \( c \) and node \( d \) are directly connected to the Hyperedge \( E_{1} \) and Hypernode \( 5 \) respectively, but node \( b \) has no connection with the selected Hyperpath. While the nodes connected with the selected Hyperpath are referred to as Participating Nodes, the other types of graph nodes are referred to as Auxiliary Nodes.

The characterization of Auxiliary Nodes results that there might be three cases as follows:
(i) A particular Auxiliary Node has no connection at all. Neither it participates in any \( c_{ij} \) nor in any \( e_{ij} \)
(ii) A particular Auxiliary Node has connections out of the scope of the selected Hyperpath;
(iii) A particular Auxiliary Node has connections within the scope of the selected Hyperpath.

The example shown just above represents the case (ii).

The engagement of Case (i) clearly has no effect in computing Cost of HG(2) Path, while engagement of Case (ii) and case (iii) induces some dependency of Graph layer to Hypergraph layer. But as stated earlier in Section 3, for simplicity, there exists no dependency from Graph layer to Hypergraph layer, so even the existence of Case (ii) and case (iii) does not make any difference in computation of Costs of HG(2) Path. Had there been a scope of mutual dependency between Graph and Hypergraph layer, the existence of Case (ii) and Case (iii) had have severe effect in HG(2) Path cost, but in present research, this is beyond the scope of any further discussions.

On this background, for all the Participating Nodes, there are weighted connectors and only the connectors connected to the Participating Nodes will contribute the cost in computing HG(2) Path. Hence the cost of connectors for a set of specific \( R_{i}^{HG(2)} \) and specific \( GP_{pq} \), which is denoted as \( C_{xy} \) could be given by the relation:

\[
C_{xy} = \sum_{i=0}^{n} C_{c_{ij}} + \sum_{i=0}^{n} C_{e_{ij}} \text{ where each } c_{ij} = c_{ij}^v \text{ and each } e_{ij} = e_{ij}^e
\]

denote node to node and edge to node connector respectively which are connected to each of the participating node \( j \) for a specific selected \( GP_{pq}^{w} \) \((w = 1, 2, \ldots)\).

All the preceding discussions identify the cost expression of a HG(2) Path \( P_{st}^{HG(2)} \) for a specific HG(2) route \( R_{i}^{HG(2)} \) and specific GPath \( GP_{pq} \) as

\[
C_{P_{st}^{HG(2)}} = C_{P_{st}^{HG(2)}}^{u} + C_{GP_{pq}} + C_{uz}
\]

For a given \( P_{st}^{HG(2)} \) there will be multiple set of \((u, z)\) pair and the \( \min(C_{P_{st}^{HG(2)}}^{u}, z) \) will denote the least cost.

V. CONCLUSION

The present research has introduced a data structure which has been denoted as HG(2). This proposed data structure has a great potential in modeling and resolving complex combinatorial problems. The basic Hypergraph and Graph based architecture of this data structure has been presented. The theoretical concepts of Paths in HG(2) has been discussed. Cost Analysis of HG(2) Path has been thoroughly presented.

Within the current discussion, the future scope of HG(2) research has aptly been identified to focus on generation of Minimum Spanning Tree out of HG(2), Identification of Least Cost Path, HG(2) Path Traversal algorithms and modeling of different real life problems (like RDF integration in Semantic Web).

REFERENCES

[1] Berge, C.: Graphs and Hypergraphs. North-Holland, Amsterdam, 1973.
[2] Boullie, F.: Un modéle universel de banque de données, simultanément partageable, portable et partagé. Thése de Doctorat d’Etat es Sciences Mathmatiques (mention Informatique), Université Pierre et Marie Curie, 1977.
[3] Berge, C.: Minimax theorems for normal hypergraphs and balanced hypergraphs - a survey. Annals of Discrete Mathematics, 21 (1984), 3-19.
[4] Berge, C.: Hypergraphs: Combinatorics of Finite Sets. North-Holland, Amsterdam, 1989.
[5] Gallo, G., Longo, G., Nguyen, S., Pallottino, S.: Directed Hypergraphs and Applications. www.cis.upenn.edu/~huang3/wpe2/papers/gallo92directed.pdf