Efficient Topological OLAP on Information Networks[⋆]

Qiang Qu, Feida Zhu, Xifeng Yan, Jiawei Han, Philip S. Yu, and Hongyan Li

{quqiang,lihy}@cis.pku.edu.cn, fdzhu@smu.edu.sg,
xyan@cs.ucsb.edu, hanj@cs.uiuc.edu, psyu@cs.uic.edu

Abstract. We propose a framework for efficient OLAP on information networks with a focus on the most interesting kind, the *topological OLAP* (called "*T-OLAP*"), which incurs topological changes in the underlying networks. T-OLAP operations generate new networks from the original ones by rolling up a subset of nodes chosen by certain constraint criteria. The key challenge is to efficiently compute measures for the newly generated networks and handle user queries with varied constraints. Two effective computational techniques, *T-Distributiveness* and *T-Monotonicity* are proposed to achieve efficient query processing and cube materialization. We also provide a T-OLAP query processing framework into which these techniques are weaved. To the best of our knowledge, this is the first work to give a framework study for topological OLAP on information networks. Experimental results demonstrate both the effectiveness and efficiency of our proposed framework.

1 Introduction

Since its introduction, OLAP (On-Line Analytical Processing) [10,2,11] has been a critical and powerful component lying at the core of the data warehouse systems. With the increasing popularity of network data, a compelling question is the following: "*Can we perform efficient OLAP analysis on* information networks?" A positive answer to this question would offer us the capability of interactive, *multi-dimensional* and *multi-level* analysis over tremendous amount of data with complicated network structure.

Example 1 (Academia Social Network Interaction). From an academic publication database like DBLP, it is possible to construct a heterogeneous information network as illustrated in Figure 1. There are four kinds of nodes each representing institutions, individuals, research papers and topics. Edges between individuals and institutions denote affiliation relationship. Edges between two individuals denote their collaboration relationship. A paper is connected to its authors, and also to its research topic. ■

OLAP operations could expose two kinds of knowledge that are hard to discover in the original network.

1. *Integrating knowledge from different parts of the network.* As an example, users could be interested in questions like "who are the leading researchers in the topic of social network?". This knowledge involves integrating information lying in two

⋆ This work is supported by Natural Science Foundation of China (NSFC) under grant numbers: 60973002 and 60673113.

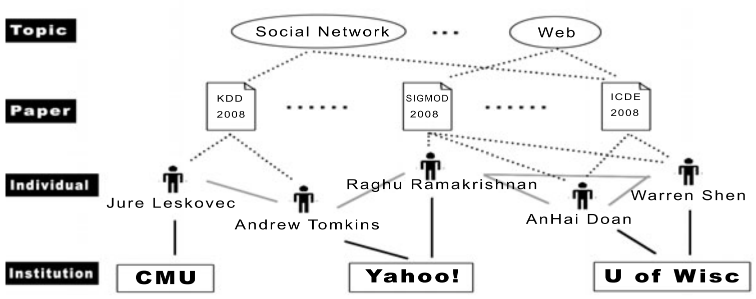

Fig. 1. A Heterogeneous Information Network

parts of the network: (1) the linkages between the individuals and papers, and (2) the linkages between the papers and the topics. As shown in the example, for the nodes representing papers, we can roll-up on them and group them by the same topics, as shown in Figure 2. As the nodes are being merged, the original edges between the papers and individuals would be aggregated accordingly, and the resulting edges would denote the authors' publication prominence in the research of every topic.

2. *Investigating knowledge embedded in different granularity levels of the network.* Besides synchronous drilling in traditional OLAP, many knowledge discovery tasks in information networks may need asynchronous drilling. For example, in Figure 3, users could be interested in the collaborative relationship between the Yahoo! Lab and related individual researchers. For instance, such analysis could show strong collaborations between AnHai Doan and researchers at both Yahoo! Lab by examining the corresponding edges. On the other hand, if the whole Wisconsin database researchers be merged into a single node, it would be hard to discover such knowledge since, collectively, there would be even stronger collaborations between Wisconsin and other universities, which may shadow Doan's link to Yahoo! Lab. Such asynchronous drilling should be guided by what can be potentially found in knowledge discovery, and thus leading to the concept of *discovery-driven OLAP*.

Based on the above motivating example, we propose a new framework for OLAP over information networks. Under this framework, we assume nodes and edges of an information network are associated with multiple hierarchical dimensions. OLAP (such as dicing and drilling) on information network takes a given network as input data and generates new networks as output. This is rather different from traditional OLAP which takes facts in the base cuboid and generates aggregate measures at high-level cuboids.

The second major difference between our OLAP model from the traditional one is the concept of asynchronous, *discovery-driven* OLAP. In the traditional data warehouse systems, drilling is performed synchronously across all the data in a cuboid. However, for OLAP in an information network, such synchronous drilling may fail to expose some subtle yet valuable information for knowledge discovery.

The information network OLAP (i.e., InfoNet OLAP) poses a major research issue: How to efficiently perform InfoNet OLAP? This paper answers this question by proposing two general properties, *T-distributiveness* and *T-monotonicity*, for efficient

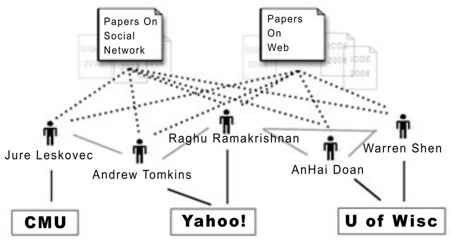

Fig. 2. Roll-up on Papers of the Same Topic

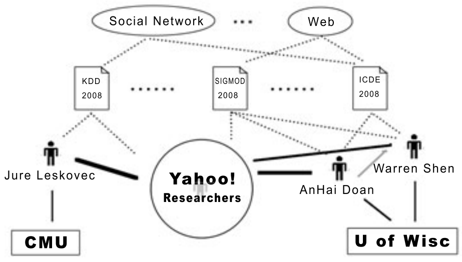

Fig. 3. Asynchronous Roll-up on Researchers to Institutions

computation of different measures in InfoNet OLAP. Our focus of this study is on efficient T-OLAP, the OLAP operations that change the topological structure (such as node merging) of the network. Moreover, we provide algorithms for computing the measures discussed in our categorization. In particular, we also examine the monotonicity property and their impact on efficient query processing. Our experiments on both real and synthetic networks demonstrate the effectiveness and efficiency of the application of our framework.

2 Problem Formulation

We study a general model of *attributed* networks, where both vertices and edges of a network G could be associated with attributes. The attributes, depending on their semantic meanings, could be either of categorical values or numeric ones. We use the DBLP co-authorship network, referred to as "DBLP network" from now on, as a running example for many illustrations in later discussions.

DBLP Network Example. In DBLP co-authorship network, each node v represents an individual author, associated with attributes: *Author Name*, *Affiliated Institution*, and *Number of Papers Published*. Each edge (u, v) between two authors u and v represents their coauthor relationship for a particular conference in a particular year, with attributes like *Conference, Year, Number of Coauthored Papers*. Evidently, there could be multiple edges between two vertices in the DBLP network if two authors have coauthored papers in different conferences. For instance, it could be found between two authors u and v edges like *(ICDE, 2007, 2)* and *(SIGMOD, 2008, 1)* and so on.

A network is *homogeneous* if every edge and vertex represents the same kind of entities and relationships, e.g. the DBLP network. Otherwise, it is *heterogeneous*.

We focus our discussion on homogeneous networks in this paper, and it should be evident that most of the results apply to heterogeneous networks as well. As a convention, the *vertex set* of a network G is denoted by $V(G)$ and the *edge set* by $E(G)$. The size of a network G is defined as the number of edges of G, written as $|E(G)|$. Let $\Sigma_V^i, 1 \leq i \leq m$ and $\Sigma_E^i, 1 \leq i \leq n$ denote the sets of valid attribute values for vertices and edges respectively.

Definition 1. [Attributed Network Model] *An attributed network is a triple* (G, L_V, L_E) *in which* $G = (V(G), E(G))$, $L_V : V(G) \mapsto \Sigma_V^1 \times \Sigma_V^2 \times \ldots \times \Sigma_V^m$ *and* $L_E \subseteq V(G) \times V(G) \times \Sigma_E^1 \times \Sigma_E^2 \times \ldots \times \Sigma_E^n$*, where* m *and* n *are numbers of attribute dimensions for vertices and edges respectively.*

In InfoNet OLAP, the underlying data for a measure is now a network, instead of isolated numerical values, thus measures could in this case take the form of a network. Given an attributed network G and a measure θ, we use $\theta(G)$ to denote the measure θ computed on the network G.

In general, given G and θ, a query in InfoNet OLAP could be represented as "SELECT G' FROM G WHERE $f_C(\theta(G'), \delta) = 1$" in which $G' \subseteq G$ and $f_C()$ is a boolean function taking as input a constraint C, a measure $\theta(G')$ computed on G' and a user-defined threshold δ, such that $f_C(\theta(G'), \delta) = 1$ if and only if $\theta(G'), \delta$ satisfies constraint C. For example, given a network G, suppose the measure θ is "diameter", then a corresponding constraint C could be "$\theta(G') <= \delta$". Then $f_C(\theta(G'), \delta) = 1, G' \subseteq G$ if and only if the diameter of G' is at most δ.

Such queries can be issued for the original data network in which every node can be considered as a data cuboid. However, for T-OLAP on InfoNet, these kind of queries could more likely be issued for some summarized network generated from the original one by merging or rolling up certain subgraphs as illustrated in Figure 2 and 3. For efficient OLAP in traditional data warehouse, data cube computation has been playing an important role with many algorithms developed. However, for InfoNet OLAP, materialization of information network "cubes" may not be realistic due to the huge number of possible flexible "cubes" that have to be precomputed, considering drilling may not even be "synchronized" (*i.e.*, rolling all the network nodes up to the same level) as one may like to perform selective drilling for effective discovery-driven OLAP. On the other hand, it is often the case that we already have some partially materialized cubes as a result of preceding queries on some summarized level. Then the central question is the following: *Can we make use of the partially materialized cubes to more efficiently answer a new coming query? If yes, how?*

3 Techniques and Framework

We propose two constraint-pushing techniques based on the unique characteristics of InfoNet OLAP, *T-Distributiveness* and *T-Monotonicity*. The framework taps the powerful techniques in traditional OLAP on data cube and extends them further into the information network setting. We use a simple motivating example to introduce the two techniques.

DBLP Query Example. Given the DBLP author network, suppose the measure θ of interest is the "total number of publications", *i.e.*, for a given node v, denoted as $\theta(v)$ its total number of publications. Depending on the level of network to which v belongs, v could represent an individual researcher, a research group, or an institution. A user could then submit queries asking to return "all researchers v such that $\theta(v) \geq \delta$". ■

The measure in the above example is in fact the "Degree Centrality". We use $C_D(v)$ to denote this measure, Degree Centrality, for a node v. To formally represent the concept of networks at different levels, we need a definition of *OLAP network hierarchy*

Definition 2. ***[OLAP Network Hierarchy]*** *Given a network $G(V, E)$ and a partition Π of $V(G)$ such that $\Pi_G = \{V_1, V_2, \ldots, V_m\}, m \leq |V(G)|$. A network G' is called a* higher-level *network of G if G' is obtained by merging each $V_i \in \Pi_G, 1 \leq i \leq m$ into a higher-level node v'_i and the edges accordingly. G is then called a* lower-level *network of G' and denoted by $G \preceq G'$. For each $v \in V(G)$, v'_i is called the* higher-level *node of v if $v \in V_i$, which is denoted as $v \preceq_V v'_i$.*

Notice that topological OLAP operations could be asynchronous. A higher-level network can be obtained by merging portions of a lower-level one, leaving the rest unchanged.

3.1 T-Distributiveness

Suppose we have three levels of networks where nodes represent individuals, research groups and institutions in each network respectively. Instead of individuals, users could query about the institutions with the total number of publications beyond a certain threshold δ. The straightforward way is to construct the network G'' at the institution level by merging the constituent author nodes for each institution from the original network G, and compute the measure by summing up over each. For large institutions, the computation could be costly. Now suppose we have already computed the measure for the network G' at the research group level, can we exploit this partial result to improve efficiency? It turns out we can do that in this case due to the distributiveness of this measure function. Basically, the measure value of an institution can be correctly obtained by summing up over the measure values already computed for the research groups. Consider any set of vertices $S = \{v_1, v_2, \ldots, v_k\}$ and a partition Π_S of S such that $\Pi_S = \{S_1, S_2, \ldots, S_m\}, m \leq k$. Each $S_i \in \Pi_S$ is merged to a new vertex v'_i and the whole set S is merged to a new vertex v'' by a T-OLAP roll-up operation. We also overload the notation to denote $\Pi_S = \{v'_1, v'_2, \ldots, v'_m\}$. It is easy to verify that

$$\begin{aligned}
C_D(v'') &= \left(\sum_{v_i \in S} C_D(v_i)\right) - 2|E_S| \\
&= \sum_{1 \leq i \leq m} \left(\sum_{v_i \in S_i} C_D(v_i) - 2|E_{S_i}|\right) - 2|E_{\Pi_S}| \\
&= \left(\sum_{v'_i \in \Pi_S} C_D(v'_i)\right) - 2|E_{\Pi_S}|
\end{aligned}$$

where E_S is the set of edges with both end vertices in S. It is clear that, since addition and subtraction are commutative, distributive and associative, the result of computing by definition from the bottom-level network is the same as the result of computing from the intermediate-level one. Figure 4 is an illustration of the computation. $C_D(v'')$ is a total of $4 + 2 + 5 + 3 = 14$ from G''. We can get this measure directly from the original network G by the given formula $\sum_{v_i \in S} C_D(v_i) - 2|E_S| = (3 + 8 + 3 + 7 + 10 + 11 + 7 + 5 + 6) - 2(2 + 3 + 3 + 1 + 2 + 4 + 1 + 2 + 4 + 1) = 14$. We can also

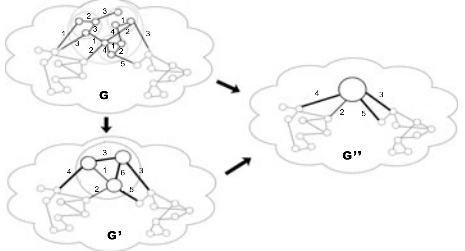

Fig. 4. T-Distributiveness for Degree Centrality

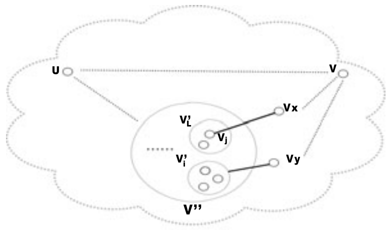

Fig. 5. T-Distributiveness for Shortest Path

use partial measure results computed for the intermediate network G' and compute by $\sum_{v'_i \in \Pi_S} C_D(v'_i) - 2|E_{\Pi_S}| = (8+12+14) - 2(3+1+6) = 14$. The computational cost is reduced to $O(m + |E_{\Pi_S}|)$. This example shows that the computation cost is greatly reduced by taking advantage of partial measure results already computed. This kind of distributiveness of a measure function is termed *T-Distributiveness* in this topological OLAP setting.

We now give the formal definition of *T-Distributiveness*.

Definition 3. ***[T-Distributiveness]*** *Given a measure θ and three attributed networks G, G' and G'' obtained by T-OLAP operations such that $G \preceq G' \preceq G''$, suppose we have available $\theta(G)$ and $\theta(G')$, then θ is* T-Distributive *if there exists a function g such that $\theta(G'') = g(\theta(G')) = g(\theta(G))$.*

Although this example of "Degree Centrality" may seem simple, it is interesting to note that other more complicated measures, even those involving topological structures, are also T-distributive. For instance, it can be shown that the measure of "Shortest Path" is also T-distributive. Shortest path computation is a key problem underlying many centrality measures, such as *Closeness Centrality* and *Betweenness Centrality*, as well as important network measures like *Diameter*.

T-Distributiveness for Shortest Paths. It is well-known that the shortest path problem has the property of optimal substructures. In fact, shortest-path algorithms typically rely on the property that a shortest path between two vertices contains other shortest paths within it. Formally, we have the following lemma, the proof of which is omitted and readers are referred to [6].

Lemma 1. *Given an attributed network G with a weight attribute on edges given by function $w : E(G) \mapsto R$, let $p = \langle v_1, v_2, \ldots, v_k \rangle$ be a shortest path from vertex v_1 to vertex v_k and, for any i and j such that $1 \leq i \leq j \leq k$, let $p_{ij} = \langle v_i, v_{i+1}, \ldots, v_j \rangle$ be the sub-path of p from vertex v_i to vertex v_j. Then, p_{ij} is a shortest path from v_i to v_j.*

Rationale. The significance of the optimal substructure property of the shortest path problem is that it means the measure is T-distributive, thus providing an efficient way to compute the measure for T-OLAP roll-up operations.

We show our algorithm in Algorithm 2. The main algorithm is Algorithm 1 in which we show that, instead of computing from scratch from the lowest network G, we are actually able to compute the measure network $\theta(G'')$ for G'' from the measure network $\theta(G')$ already computed for an intermediate network G'.

In Algorithm 1, in Line 3, we first compute all shortest paths from the single source v'' to all other vertices. From Lines 4 to 7, we update the shortest path between each pair of vertices (u, v) by picking the smaller-weight one between the existing shortest path between them and the one which passes through the new vertex v''. In Algorithm 2, in Lines 1 and 2, we first set the shortest path weight between v'' and other vertices to be a maximum weight value. From lines 3 to 6, we calculate the shortest paths between v'' and every other vertex u by picking the one with the minimum weight among all the shortest paths between vertices in S' and u. It is easy to verify that the time complexity of computational cost of $ShortestPath\_Local$ is $O(|S'| \cdot |V(G) \setminus S|)$. The time complexity of the entire algorithm is therefore $O(|V(G)|^2)$.

The correctness of the entire algorithm can be seen from the observation that for any pair of vertices u and v, if the final shortest path $p_{u,v}$ in G'' does not pass through the new vertex v'', then it should also be the shortest path between u and v in the lower-level network G'. Therefore, the final shortest path $p_{u,v}$ in G'' must be the smaller-weight one between the existing shortest path between them in G' and the new shortest path passing through v''. By the optimal substructure property in Lemma 1, the new shortest path passing through v'' must be the union of the two shortest paths, one between u and v'', and the other between v'' and v. When computing the shortest paths between v'' and other vertices, we do not use standard single source shortest path algorithms. Instead, Algorithm $ShortestPath\_Local$ harness the T-distributiveness of the shortest path measure.

Theorem 1. *Given an attributed network G with edge weights, G'' is obtained by merging a set of vertices $S = \{v_1, v_2, \ldots, v_k\}$, $S \subseteq V(G)$ in a T-OLAP roll-up operation to a new vertex v'', and G' is obtained by partitioning S by $\Pi = \{S_1, S_2, \ldots, S_k\}$ and merging the vertices in each S_i into $v'_i \in S', 1 \leq i \leq k$, then given the shortest path measure network $\theta(G')$, $ShortestPath\_Local$ computes the shortest paths between v'' and all vertices in $V(G) \setminus S$.*

Proof. The proof is omitted due to the limitation of space. ■

Algorithm 1. ShortestPath_Main

```
Input: $S'$, $G$ and $\theta(G')$
Output: $\theta(G'')$

1: $\theta(G'') \leftarrow \theta(G')$
2: Merge $S'$ into $v''$ and add $v''$ to $G''$;
3: $\theta(G'') \leftarrow ShortestPath\_Local(S', G, \theta(G''))$;
4: for each $u \in V(G''), u \neq v''$
5:     for each $v \in V(G''), v \neq v''$
6:         if $w(p_{uv}) > w(p_{uv'}) + w(p_{v'v})$
7:             $w(p_{uv}) \leftarrow w(p_{uv'}) + w(p_{v'v})$
8:return $\theta(G'')$;
```

Algorithm 2. ShortestPath_Local

```
Input: $S'$, $G$ and $\theta(G'')$
Output: $\theta(G'')$

1: for each $u \in V(G) \setminus S'$
2:     $w(p_{v''u}) \leftarrow +\infty$;
3: for each $u \in V(G) \setminus S'$
4:     for each $v \in S'$
5:         if $w(p_{vu}) < w(p_{v''u})$
6:             $w(p_{v''u}) \leftarrow w(p_{vu})$;
7:return $\theta(G'')$;
```

3.2 T-Monotonicity

Suppose the user queries for all pairs of collaborating researchers with the number of joint publications above a threshold δ. The observation is that the total number of publications of an institution is at least as large as that of any of its constituent individual. This simple monotone property could help prune unnecessary data search space substantially in the query processing: Given the threshold δ, any institution node pairs with its measure value less than δ could be safely dropped without expanding to explore any of its constituent nodes at lower level networks. The monotonicity of a constraint like this is termed *T-Monotonicity* in this topological OLAP setting. The definition of T-Monotonicity is as follows.

Definition 4. ***[T-Monotonicity]*** *Given a measure θ and a constraint C, let G and G' be two networks such that $G \preceq G'$, C is* T-Monotone *if $f_C(P_1) = 1 \rightarrow f_C(P_2) = 1$ for all $P_1 \subseteq G, P_2 \subseteq G'$ and $P_1 \preceq P_2$.* ■

It is not just simple and common measures like the example above that are T-monotone, in fact, it can be shown that many complicated and important measures which involve network structures are also T-monotone. Interestingly, "Shortest Path" is again a good case in point.

T-Monotonicity for Shortest Paths. For shortest path, it turns out the corresponding constraints have the property of T-monotonicity. The intuition is that when nodes from a lower-level network are merged to form nodes in a higher-level network, the shortest paths between any pair of nodes in the higher-level network *cannot* be elongated, which is proved as follows.

Theorem 2. *Given two networks G_1 and G_2 such that $G_1 \preceq G_2$, for any two nodes $u, v \in V(G_1)$, let $u', v' \in V(G_2)$ be the corresponding higher-level nodes such that $u \preceq_V u'$ and $v \preceq_V v'$. Then we have $Dist(u', v') \leq Dist(u, v)$.*

Proof. Denote $w(u, v)$ as the weight of edge (u, v). Let one of the shortest paths between u and v in G_1 be $p = \langle v_0, v_1, \ldots, v_k \rangle$ where $v_0 = u$ and $v_k = v$. Since $G_1 \preceq G_2$, there exists some i and d such that vertices $v_i, v_{i+1}, \ldots, v_{i+d}$, $0 \leq i, d \leq k$ of p are merged into a single vertex w in G_2. Then the weight of the shortest path between u' and v' in G_2 will have

$$\delta(u', v') = \delta(u', w) + \delta(w, v') \leq \sum_{0 \leq j < i} w(v_j, v_{j+1}) + \sum_{i+d \leq j < k} w(v_j, v_{j+1}) \leq \delta(u, v)$$

We give a summary of some common network measures in Figure 6.

3.3 T-OLAP Query Processing Framework

Both T-distributiveness and T-monotonicity would be pushed into the framework for processing T-OLAP queries. The framework of T-OLAP query processing consists of the following stages:

Pre-computation: Given a network G and the measure θ to be computed, the query algorithm first computes the base cuboids to be materialized.

Constraints	SUM	MIN, MAX	Min Degree, Max Degree	Density	Bridging Capital	Degree Centrality	Closeness Centrality	Betweenness Centrality	Diameter	Structural Cohesion	Containment
T-Monotonicity	Yes	Yes	No	No	No	No	Yes	No	Yes	No	No
T-Distributiveness	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	No	No

Fig. 6. A General Picture of Typical InfoNet OLAP Constraints

Query Processing:

1. **Abstraction Level Processing:**
   Given the OLAP abstraction level from the user query, the algorithm locates the most immediate higher-level and lower-level networks whose corresponding cubes have been partially materialized.
2. **Measure Computation:**
   Given the constraint C from the user query, the higher-level network will be used to prune search space by applying T-monotonicity whenever available. Lower-level network will be used for more efficient measure computation for the required abstraction level by applying T-distributiveness whenever available.

4 Experimental Results

4.1 Synthetic Data

All the experiments are conducted on a Pentium(R) 3GHz with 1G RAM running Windows XP professional SP2.

T-Distributiveness. We perform experiments for two measures, *Degree Centrality* and *Closeness Centrality* on synthetic data to demonstrate the power of T-distributiveness.

Since our aim is to provide studies on measures for InfoNet OLAP in general, our synthetic data networks are not confined to specific types and statistical properties. Our synthetic data networks are generated in a random fashion such that (1) the entire network is connected, (2) the vertices have an average degree of $\hat{d}$ and (3) the edges have an average weight of $\hat{w}$.

Given a network G, users can choose a subset S of vertices to roll-up into a single vertex v' and compute the measure network for the new network G'. Such an OLAP operation is called a user OLAP request. We give a model for incoming user OLAP requests as follows. For a network G, we recursively partition G into π connected non-overlapping components of equal number of vertices, until each resulting component is of a predefined minimum number of vertices, i.e., suppose $|V(G)| = 1024$ and $\pi = 4$, we first partition G into 4 connected subgraphs each with 256 vertices, and recursively partition the 4 subgraphs. The partition process identifies a sequence T of connected subgraphs of the original network G. Now we reverse the sequence T and let the resulting sequence be T'. Consequently, observe that, for any subgraph Q in sequence T', all the subgraphs of Q appear before Q. We model the sequence of incoming user OLAP requests as the subgraph sequence T', i.e., the i-th user OLAP request would take the original network G and choose to merge the i-th subgraph in T' into a single vertex and thus obtain a higher-level network G'. The task then is to compute the measure network $\theta(G')$ for G'.

Our baseline algorithm for comparison is denoted as NaiveOLAP. For each user OLAP request, the naive algorithm would first merge the corresponding subgraph into a single vertex and then compute the measure network for the new graph directly from the original network G. Our approach, called T-distributiveOLAP (or TD-OLAP for short), would take advantage of the T-distributiveness of the measure and take the measures already computed for π lower level networks as input to compute the new measure network. In other words, if put in traditional OLAP terminology, we are considering the best scenario here in which, when computing the measure for a cuboid, all the cuboids immediately below have already been materialized.

Degree Centrality. The measure of *Degree Centrality* has the nice property of T-distributiveness. TD-OLAP could therefore make use of the measures computed for the lower-level networks and gain significant efficiency boost than the NaiveOLAP. The average vertex degree is set to $\hat{d} = 5$. The partition size π is set as 4 such that each high level vertex has 4 lower-level children vertices.

Figure 7 shows the running time comparison for the two approaches as the number of vertices for the original network increases. In this case, the original network G is recursively partitioned for a recursion depth of two with a partition size of 4. The running time is the result of summing up the computation cost for all the user OLAP requests in T'. It can be observed that with T-distributiveness the measure network computation cost increases much slower than the NaiveOLAP approach.

Figure 8 shows that, when the total number of vertices of the network G is fixed to 4096 and the average vertex degree is set to 5, how the granularity of T-OLAP operations can affect the running time of both approaches. As the number of partitions increases, the size of the set of vertices to be merged in the T-OLAP roll-up get smaller, which means the user is examining the network with a finer granularity. Since the measure of degree centrality has a small computational cost, both approaches have in this case rather slow increase in the running time. However, notice that the TD-OLAP still features a flatter growth curve compared with the NaiveOLAP approach.

Closeness Centrality. The measure of *Closeness Centrality* has the nice property of T-distributiveness. As such, TD-OLAP would use the algorithms as shown in Algorithm 1 to assemble the measures computed for the lower-level networks and save tremendous computational cost than the NaiveOLAP which simply merge subsets of vertices and run costly shortest path algorithm to compute the new measure network from scratch. In this example, the average degree is set to $\hat{d} = 5$ and the average weight on edges is set as $\hat{w} = 10$. The partition size π is set as 4 such that each high level vertex has 4 lower-level children vertices.

Figure 9 shows the running time comparison for the two approaches as the number of vertices for the original network increases. In this case, the original network G is recursively partitioned for a recursion depth of two with a partition size of 4. The running time is the result of summing up the computation cost for all the 20 user OLAP requests in T'. It is clear that, by harnessing T-distributiveness, the measure networks can be computed much more efficiently, almost in time linear to the size of the original data network, than the naive OLAP approach.

Figure 10 shows how the granularity of the T-OLAP roll-up can impact the running time for both approaches. As the number of partitions increases, the original network is partitioned into components of increasingly smaller sizes. The figure shows the average cost for computing the new measure network for one OLAP request as users choose to merge smaller set of vertices in the T-OLAP operations. The network in this case contains 1024 vertices. As shown in the figure, for TD-OLAP, the granularity hardly affects the computational cost since the complexity of the function to combine the measures of lower-level networks to obtain the new one is in general very low compared with the function to compute the measure itself. As the partition size only affects the number of lower-level vertices to taken into consideration, the running time therefore remains steady. On the other hand, as fewer vertices are merged with increasing number of partitions, the NaiveOLAP has to compute the measure network with an input network of greater size. Hence the increasing running time for the NaiveOLAP.

T-Monotonicity. We perform experiments on the measure of *Shortest Distance* to demonstrate the power of T-monotonicity. The number of nodes is set to 1024. The average node degree is set to 5 and the average weight on edges is set to 5. The T-OLAP scenario is the following. The user would perform T-OLAP operations on the underlying network G in the same fashion as in the experiment settings for T-distributiveness. We obtain a higher-level network G' with π partitions, each becoming a higher-level node. Then the user would present queries in an asynchronous T-OLAP manner as follows. Two partitions (nodes) of G' will be expanded into their constituent lower-level nodes while the rest partitions remain as higher-level nodes, thus generating a new network $\hat{G}_1$ such that $G \preceq \hat{G}_1 \preceq G'$. We can then choose another two partitions of G', proceed likewise and obtain another network $\hat{G}_2$. For a number of partitions π, we can obtain $\binom{\pi}{2}$ networks $\hat{G}_1, \hat{G}_2, \ldots, \hat{G}_{\binom{\pi}{2}}$ by the sequence of asynchronous T-OLAP operations. In the process, the user would query for the shortest distance for every pair of lower-level nodes u and v in $\hat{G}_i$ for $1 \leq i \leq \binom{\pi}{2}$ such that u and v are expanded out of different higher-level nodes, under the constraint that the minimum of all these shortest distances is smaller than a threshold δ. It is easy to see that the naive way would have to compute all-pair shortest distances for each $\hat{G}$ to find the minimum. Due to the T-monotonicity of shortest distance, we can prune data search space as follows. If we pre-compute the shortest distances between every pair of higher-level nodes in G', then if the shortest distance between two nodes u' and v' of G' is greater than δ, then for any pair of nodes u and v expanded out of u' and v' respectively, the shortest distance between u and v in the corresponding network $\hat{G}$ must be greater than δ. Therefore there is no need to expand u' and v' for all-pair shortest distance computation, thus reducing computational cost. Figure 11 shows how much running time we are able to save for a successful pruning by T-monotonicity as the number of partitions π increases. The curve well illustrates the cost saving which is proportional to the size of the OLAP-generated network $\hat{G}$ upon which the naive method would need to compute all-pair shortest distances. It is not monotone since the size of $\hat{G}$ first decreases and then increases as the number of partitions π increases. Figure 12 shows the situation where π is set to be 64 and the average edge weight is 500. User queries in this case also ask to return the shortest distances between all lower-level nodes for all $\hat{G}_i$ but with the constraint that the shortest distance is smaller than a threshold δ. Figure 12 shows

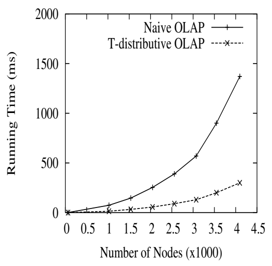

Fig. 7. Run Time Comparison

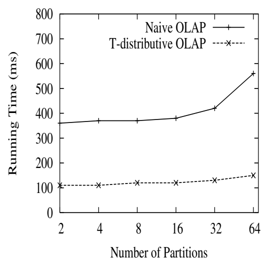

Fig. 8. T-OLAP Granularity

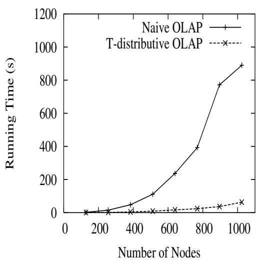

Fig. 9. Run Time Comparison

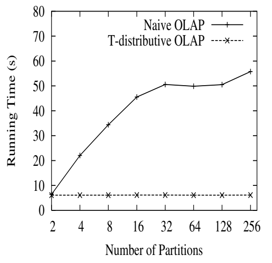

Fig. 10. T-OLAP Granularity

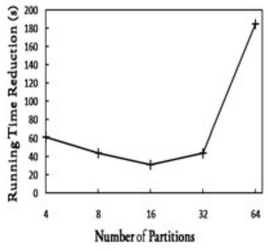

Fig. 11. Cost Reduction

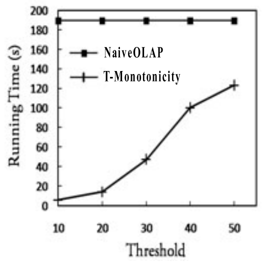

Fig. 12. Run Time Comparison

the running time for the query processing as the user-defined threshold δ increases. It clearly shows that as δ increases, the pruning power weakens since when $\delta \rightarrow \infty$, it means all shortest distances need to be returned to the user.

4.2 Real Data

Based on the DBLP data, we can semi-automatically construct a heterogeneous network as illustrated in Figure 13. Edges between different types of entities could carry different attributes. For instance, edges connecting researchers and topics could have the relevant publication on this topic by this author; edges between two researchers could carry the publications co-authored by them; edges between a researcher and an institution could carry those researchers from the institution who have collaborations with this researcher, etc.. Wider edges indicate stronger relationships in terms of greater quantities. By performing discovery-driven, asynchronous T-OLAP operations, users would be able to examine, analyze and discover knowledge in a multi-dimensional and multi-level fashion, uncovering hidden information which is previously hard to be identified in traditional data warehouse scenario. For example, Figure 13 shows a snapshot of the network after a sequence of discovery-driven T-OLAP operations. One can easily observe that while Michael Stonebraker, Jennifer Widom and Rajeev Motwani all work on the topic of "stream data", they also have their own separate heavily-involved research topics of "C-Store", "Uncertainty" and "Web" respectively.

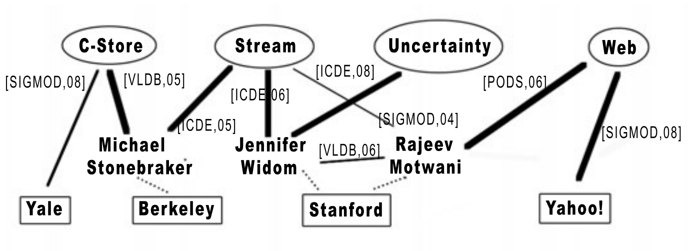

Fig. 13. A Snapshot Of A Portion Of A Real Heterogeneous Network

5 Related Work

Social network analysis, including Web community mining, has attracted much attention in recent years. Abundant literature has been dedicated to the area of social network analysis, ranging from the network property, such as power law distribution [18] and small world phenomenon [15], to the more complex network structural analysis such as [8], evolution analysis [16], and statistical learning and prediction [13]. The static behavior of large graphs and networks has been studied extensively with the derivation of power laws of in- and out- degrees, communities, and small world phenomena. This work is not to study network distribution or modeling but to examine a general analytical process, with which users can easily manipulate and explore massive information networks to uncover interesting patterns, measures, and subnetworks.

OLAP (On-Line Analytical Processing) was studied extensively by researchers in database and data mining communities [10]. Major research themes on OLAP and data cube include efficient computation of data cubes [2], iceberg cubing [7], partial materialization and constraint "pushing" [20].

Although OLAP for the traditional form of spreadsheet data has been extensively studied, there are few studies on OLAP on information networks although information networks have been emerging in many real-world applications. One interesting study that puts graphs in a multi-dimensional and multi-level OLAP framework is in [5]. However, it focuses on informational OLAP in which the rolling/drilling operations only merge multiple edges between the same pair of nodes. As there is no merging of nodes, there is no change in the underlying network structure. As such, [5] only covers a rather limited subset of all the possible OLAP operations on information networks, whereas topological OLAP (T-OLAP), the more powerful ones for knowledge discovery, has not yet been systematically explored.

InfoNet OLAP provides users with the ability to analyze the network data from any particular perspective and granularity. The T-OLAP operation of rolling-up delivers a summarized view of the underlying network. Therefore, from the perspective of generating summarized views of graph data, different aspects of the problem has been examined in one form or another such as compression, summarization, and simplification. [21,3] study the problem of compressing large graphs, especially Web graphs. Yet they only focus on how the Web link information can be efficiently stored and easily manipulated to facilitate computations like PageRank and authority vectors. [4] develops statistical summaries that analyze simple graph characteristics like degree distributions and hop-plots. While these papers studied effective summarization of graph data, they did not aim to give a comprehensive study of multi-dimensional and multi-granularity network analysis with OLAP operations.

Similar aspects have also been explored by the graphics community under the topic of graph simplification. [26,1,17], aim to condense a large network by preserving its skeleton in terms of topological features. Works on graph clustering (to partition similar nodes together), dense subgraph detection (for community discovery, link spam identification, *etc.*) and graph visualization include [19], [9,22], and [12], respectively. The visualization and summarization of cohesive subgraphs has been studied in [24]. These studies provide some kind of summaries, but the objective and results achieved are substantially different from those of this paper.

Summarizing attributed networks with OLAP-style functionalities is studied in [23]. It introduces an operation called SNAP, which merges nodes with identical labels, combines corresponding edges, and aggregates a summary graph that displays relationships for such "generalized" node groups. There have been recent works examining certain particular network measures in great detail such as shortest paths [25] and reachability [14]. However, all these work are not aimed to study measure computation in T-OLAP setting in general and offer common constraint properties for a general query processing framework.

6 Conclusion

In this paper we have performed a framework study for topological InfoNet OLAP. In particular, we propose two techniques in a constraint-pushing framework, *T-Distributiveness* and *T-Monotonicity*, to achieve efficient query processing and cube materialization. We put forward a query processing framework incorporating these two techniques. Our experiments on both real and synthetic data networks have shown the effectiveness and efficiency of the application of our techniques and framework to the measures.

References

1. Archambault, D., Munzner, T., Auber, D.: TopoLayout: Multilevel graph layout by topological features. IEEE Trans. Vis. Comput. Graph. 13(2), 305–317 (2007)
2. Beyer, K.S., Ramakrishnan, R.: Bottom-up computation of sparse and iceberg cubes. In: SIGMOD Conference, pp. 359–370 (1999)
3. Boldi, P., Vigna, S.: The WebGraph framework I: Compression techniques. In: WWW, pp. 595–602 (2004)
4. Chakrabarti, D., Faloutsos, C.: Graph mining: Laws, generators, and algorithms. ACM Comput. Surv. 38(1) (2006)
5. Chen, C., Yan, X., Zhu, F., Han, J., Yu, P.S.: Graph OLAP: Towards online analytical processing on graphs. In: Proc. 2008 Int. Conf. Data Mining (ICDM) (2008)
6. Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C. (eds.): Introduction to Algorithms. MIT Press, Cambridge (2001)
7. Fang, M., Shivakumar, N., Garcia-Molina, H., Motwani, R., Ullman, J.D.: Computing iceberg queries efficiently. In: VLDB, pp. 299–310 (1998)

8. Flake, G., Lawrence, S., Giles, C.L., Coetzee, F.: Self-organization and identification of web communities. IEEE Computer 35, 66–71 (2002)
9. Gibson, D., Kumar, R., Tomkins, A.: Discovering large dense subgraphs in massive graphs. In: VLDB, pp. 721–732 (2005)
10. Gray, J., Chaudhuri, S., Bosworth, A., Layman, A., Reichart, D., Venkatrao, M., Pellow, F., Pirahesh, H.: Data cube: A relational aggregation operator generalizing group-by, cross-tab, and sub totals. Data Min. Knowl. Disc. 1(1), 29–53 (1997)
11. Gupta, A., Mumick, I.S. (eds.): Materialized Views: Techniques, Implementations, and Applications. MIT Press, Cambridge (1999)
12. Herman, I., Melançon, G., Marshall, M.S.: Graph visualization and navigation in information visualization: A survey. IEEE Trans. Vis. Comput. Graph. 6(1), 24–43 (2000)
13. Jensen, D., Neville, J.: Data mining in networks. In: Papers of the Symp. Dynamic Social Network Modeling and Analysis. National Academy Press, Washington DC (2002)
14. Jin, R., Xiang, Y., Ruan, N., Wang, H.: Efficiently answering reachability queries on very large directed graphs. In: SIGMOD 2008: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, pp. 595–608. ACM, New York (2008)
15. Kleinberg, J.M., Kumar, R., Raghavan, P., Rajagopalan, S., Tomkins, A.: The web as a graph: Measurements, models, and methods. In: Asano, T., Imai, H., Lee, D.T., Nakano, S.-i., Tokuyama, T. (eds.) COCOON 1999. LNCS, vol. 1627, pp. 1–17. Springer, Heidelberg (1999)
16. Leskovec, J., Kleinberg, J., Faloutsos, C.: Graphs over time: Densification laws, shrinking diameters and possible explanations. In: Proc. 2005 ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD 2005), Chicago, IL, pp. 177–187 (August 2005)
17. Navlakha, S., Rastogi, R., Shrivastava, N.: Graph summarization with bounded error. In: SIGMOD Conference, pp. 419–432 (2008)
18. Newman, M.E.J.: The structure and function of complex networks. SIAM Review 45, 167–256 (2003)
19. Ng, A.Y., Jordan, M.I., Weiss, Y.: On spectral clustering: Analysis and an algorithm. In: NIPS, pp. 849–856 (2001)
20. Ng, R.T., Lakshmanan, L.V.S., Han, J., Pang, A.: Exploratory mining and pruning optimizations of constrained association rules. In: SIGMOD Conference, pp. 13–24 (1998)
21. Raghavan, S., Garcia-Molina, H.: Representing web graphs. In: ICDE, pp. 405–416 (2003)
22. Sun, J., Xie, Y., Zhang, H., Faloutsos, C.: Less is more: Sparse graph mining with compact matrix decomposition. Stat. Anal. Data Min. 1(1), 6–22 (2008)
23. Tian, Y., Hankins, R.A., Patel, J.M.: Efficient aggregation for graph summarization. In: SIGMOD Conference, pp. 567–580 (2008)
24. Wang, N., Parthasarathy, S., Tan, K.-L., Tung, A.K.H.: CSV: visualizing and mining cohesive subgraphs. In: SIGMOD Conference, pp. 445–458 (2008)
25. Wei, F.: Tedi: efficient shortest path query answering on graphs. In: SIGMOD 2010: Proceedings of the 2010 International Conference on Management of Data, pp. 99–110. ACM, New York (2010)
26. Wu, A.Y., Garland, M., Han, J.: Mining scale-free networks using geodesic clustering. In: KDD, pp. 719–724 (2004)