基于概念分层的图汇总算法

孙 翀 卢炎生
(华中科技大学计算机学院 武汉 430074)

摘 要 将原始图中节点分配到多个分组并根据原始边来确立分组间关系,这样得到的图称作汇总图。汇总图的规模可以由用户设定,用户可以通过浏览小规模的汇总图来获得原始图的相关信息。K-SGS方法是一种新的基于节点概念分层的图汇总算法,它解决了传统K-SNAP算法的汇总图规模参数受限问题。为了解决该问题,算法引入了节点的属性值概念分层,从而增强了图汇总过程中节点分组的灵活性:不仅可以合并同值的节点,还可合并具有相似值的节点。除了关注汇总过程中边的信息损失外,K-SGS方法还关注节点的信息损失,它将图汇总问题建模成多目标规划问题,并通过分层序列法和基于分级的统一评价函数两种不同策略来解决该问题。算法上,提出了快速的层次聚类方法,使得每轮可以合并多个聚类,从而提高效率。经数据集上的实验表明,新算法能生产各种规模参数的汇总图,并具有较好的汇总质量。

关键词 图汇总,概念分层,多目标规划,层次凝聚
中图法分类号 TP311 **文献标识码** A

Clustering-based Algorithms to Semantic Summarizing Graph with Multi-attributes' Hierarchical Structures

SUN Chong LU Yan-sheng
(School of Computer Science, Huazhong University of Science and Technology, Wuhan 430074, China)

Abstract This paper allocated the raw nodes of graph into some different groups and established relationships among the groups by considering the raw edges of graph. We called this procedure graph summarization and the newly built graph was called graph summary. Users can set the scale value of graph summary and obtain the information of raw graph with the help of it. However, by using the classic method, we can only build the graph summaries whose sizes are above a certain scale lower bound, which cannot be acceptable in many applications. K-SGS is a novel graph summarization method which solves the scale limits. By using the concept hierarchy of the nodes' attributes, K-SGS can group the nodes in a flexible way. It groups the nodes not only with same values but also with similar values. Besides the edges' information loss, it also considers the nodes' information loss during the summarization and models the summarization as multi-objective planning. We proposed two hierarchical agglomerative algorithms. One is based on forbearing stratified sequencing method and the other is based on unified objective function method. The experiment on real life dataset shows that our methods can solve the problem and get the graph summaries with good quality.

Keywords Graph summarization, Concept hierarchy, Multi-objective planning, Hierarchical agglomerative

1 引言

现实世界的大量数据集被建模成图模型,图在各领域中被广泛应用,如社交网络、生物网络及动态交通网络等。大多数应用中,图中的节点用来代表现实世界中的对象且具有各种属性与之关联;对象间的关系用节点间的边来表示,且边可以具有很多种类。然而,实际应用中的图模型往往是巨大的(具有成千上万甚至上百万的节点和边),去理解被编码在这些(巨大的)图中的信息是十分困难的。因此,有效提取图中潜在信息的汇总方法被广泛研究。

传统的图汇总方法使用简单的基于统计理论的方法来描述图的特性[1-3]。例如:使用绘制点的分布来调查图的无标度特性;利用跳数图来研究小世界效果;利用群聚系数来度量大图的丛聚情况。虽然这些方法是有用的,但是汇总图包含的信息有限且难以理解和处理。挖掘图的频繁项的方法[4,5]也被用于理解大图的特征,但这些挖掘算法经常会产生大量的结果,使得用户更加迷茫。图划分算法[6,7]经常被用于寻找大的复杂网络的团结构(密集子图),团探测完全基于节点的连通性,而节点的属性却被大量忽略。图可视化技术虽然能帮助用户更好地浏览图[8-10],但快速地浏览大图十分困难。

K-SNAP方法[11]是一种交互式的、直观的且可操作的图汇总方法。根据用户提出的规模参数 K,K-SNAP对图模型中节点和边的信息汇总从而产生含有 K 个组节点及若干组关系的较小的图模型,称之为汇总图(graph summary)。该方

到稿日期:2012-11-01 返修日期:2013-01-11 本文受国家"十一五"部委预研基金(513150402)资助。
孙 翀(1981—),男,博士生,主要研究方向为嵌入式数据管理、数据挖掘,E-mail:Nicksun217@163.com;卢炎生(1953—),男,教授,博士生导师,主要研究方向为现代数据库系统、软件测试、数据挖掘。

法允许用户自由选择感兴趣的属性和关系类型，利用这些兴趣点来产生小的富含信息的汇总图。用户能控制结果汇总图的规模(即 K 的大小)，能对图进行如传统数据库系统中的OLAP风格的聚集操作。

K-SNAP汇总的基本思想是：将原始图中在关注属性上具有相同值的节点合并到同一分组，并且尽力使得同一分组中的节点在原始图中具有相似的邻接关系。下面以例子说明该方法。

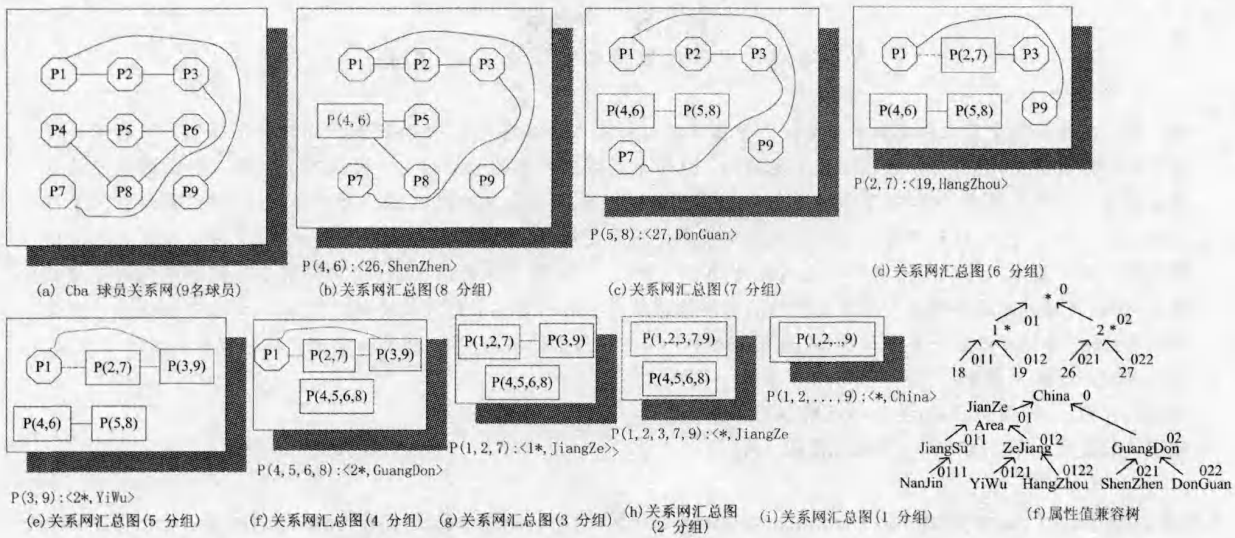

图 1 CBA 球员关系网汇总过程

图 1(b)－(d)给出了图 1(a)所示的 cba 球员关系网在不同规模下的 K-SNAP 汇总图，球员节点的个人信息保存在表 1 中。本例中，我们关注节点属性 Age 和 Location，球员节点间的边表示球员间的朋友关系，原始图包含 9 个球员节点，其节点编号见表 1 的附加列 Pid。图 1(b)－(d)显示了采取 K-SNAP 方法使得汇总图规模依次递减的结果。

表 1 CBA 球员信息表

Name	Age	Location	DeweyCode	Pid
CAI Li-long	18	NanJin	011,0111	P1
JIANG Kai-yi	19	HangZhou	012,0122	P2
ZHANG Da-yu	18	YiWu	011,0121	P3
ZHU Fang-yu	26	ShenZhen	021,021	P4
ZHANG Kai	27	DongGuan	022,022	P5
W. Shi-peng	26	ShenZhen	021,021	P6
QiuBiao	19	HangZhou	012,0122	P7
RenJunFei	27	DongGuan	022,022	P8
HuXueFeng	26	YiWu	021,0121	P9

图中汇总方法为：选择属性值相同的分组对进行合并，同等情况下优先合并造成边兼容损失最少的两个分组。边兼容准则保证了同分组内的节点具有相同或相似的邻居组节点集。实线边表示合并后边信息无损，虚线边表示有损。边无损意味着这次汇总没有影响原始节点的边关系。如：在图 1(c)中合并属性值均为(19，HangZhou)的节点 P2 和 P7 后，由于原始图中 P2 和 P7 均与 P3 存在边，因此图 1(d)中边(P(2，7)，P3)为无损；另一方面，P2 和 P7 中仅存在边(P2，P1)，则图 1(d)中边(P(2，7)，P1)为有损，用虚线表示。

文献[12]对 K-SNAP 进行了两点改进：将节点属性值的类型从分类型扩展到数值型；为用户提供了自动化设定分辨率的方法。然而，其扩展算法仍沿用传统的基于边兼容损失的度量标准进行汇总，忽略了汇总过程中对节点属性值的兼容损失的考量；更为重要的是，K-SNAP 的汇总图规模存在着下界(后简称 K 下限)，这导致用户对汇总图的规模 K 无法进行全定义域设定而仅能设定在某个下限值之上，用户对汇总图规模的控制受到限制。显然，该缺陷在许多应用中是常见却无法被接受的。

针对上述问题，本文提出一种全规模的图汇总方法，简称 K-SGS。K-SGS 在汇总过程中不仅关注边的兼容损失，还关注节点的兼容损失。为了统计汇总过程产生的节点兼容损失，K-SGS 为原始数据图构造了节点属性值的多维兼容尺，并提出汇总图的 Beta 度量。相对于 K-SNAP，K-SGS 具有更好的汇总效果。

2 形式化定义与问题描述

本文采用通用图模型，图的节点关联着任意数量的属性，边具有多种类型。形式上，原始图表示为 $G=(V,E)$，V 是节点集，E 是边集。节点的关联属性集用 $A=\{a_1,a_2,\cdots,a_m\}$ 表示，任意节点 $v\in V$ 具有 A 上的一个值；边的类型集用 $T=\{t_1,t_2,\cdots,t_n\}$ 表示，T 的非空真子集 $T(u,v)$ 显示了边 (u,v) 所具有的边类型。

使用上述约定符号，汇总图的形式化定义为：给定原始图 G，V 上的一个划分 $\Phi=\{g_1,g_2,\cdots,g_k\}$($\bigcup_{i=1}^{k} g_i=V$ 且 $\forall i\neq j$，$g_i\cap g_j=\phi$)，基于 Φ 的汇总图 $Gs=(V_s,E_s)$，其中 $V_s=\Phi$，$E_s=\{(g_i,g_j)\mid \exists u\in g_i,v\in g_j,(u,v)\in E\}$。任意边 $(g_i,g_j)\in E_s$ 的类型集 $T(g_i,g_j)=\bigcup_{(u,v)\in E,u\in g_i,u\in g_j} T(u,v)$。

Gs 的节点称为组节点，对应 Φ 中的一个分组，而 Gs 的边称为组关系，反映组节点间在原始图中的连通性。给定组节点 g_i 和 g_j，当且仅当 g_i 与 g_j 对应的两组原始节点集间至少存在一条原始边，g_i 和 g_j 存在组关系，所有的组关系保存在 E_s 中。

图的汇总问题可描述为：给定规模参数 K，寻找 G 上 V 的某个划分 Φ，并基于 Φ 得到汇总图 Gs，使得：1)$|\Phi|=K$；2)$Measure(Gs)$获得最优值。其中，$Measure$ 是度量函数，用于确定汇总图的质量。传统的 K-SNAP 的度量函数为 Delta 度量，而 K-SGS 的度量方法不仅考虑了 Delta 度量，还提出对节点属性值兼容的考虑(即 Beta 度量)。

3 K-SNAP 方法概述

3.1 组节点的特性

K-SNAP 对组节点定义了两种特性：属性兼容性和边兼容性。这两种特性保证了组节点对应的原始节点集中的任意节点具有同值和同构的特性。若汇总图的所有组节点均满足上述特性，则称该汇总图为属性和边兼容汇总图。下面给出这两种特性的定义。

属性兼容性：组节点对应的所有原始节点在(用户选择的)属性上具有相同的属性值。即给定组节点 g_i，若 $\forall v_1, v_2 \in g_i$，有 $v_1=v_2$，则 g_i 具有属性兼容性。

边兼容性：针对用户所选边类型，同一组节点中不同的原始节点在汇总图中均具有相同的邻居组节点集。即给定 T 和 g_i，若 $\forall v_1, v_2 \in g_i$，有 $Neighb(v_1)=Neighb(v_2)$，则 g_i 具有边兼容性。其中，$Neighb(v)$ 为节点 v 的邻居组节点集，即：

$$Neighb(v)=\{g' \mid \exists u \in g', g' \in V_s \text{ s. t. } T(u,v)\neq\phi\}$$

多数情况下，构建同时满足属性和边兼容性的汇总图是困难的，某些情况下更是不存在这样的汇总图。因此，K-SNAP 放宽对边兼容性的限制：同个组节点内的原始节点间具有相似的邻居组节点集。放宽限制后，可能产生大量的具有规模参数 K 的汇总图。K-SNAP 定义了 Delta 度量来衡量这些汇总图的质量。

3.2 汇总图的 Delta 度量

Delta 度量通过检查汇总图与假定的理论上完美的汇总图之间的结构差异情况来判断汇总图质量。Delta 度量的实质是统计了汇总后汇总图的边信息损失。下面给出 Delta 的定义，为了便于描述，假设原始图中的边类型唯一，所有的边均保存在边集合 E 中。

组关系参与度 $P_{i,j}$：$P_{i,j}$ 描述了组关系 (g_i, g_j) 中组节点 g_i 和 g_j 参与成边的原始点所占比率。其形式化定义为：

$$P_{i,j}=\frac{|p_{g_j}(g_i)|+|p_{g_i}(g_j)|}{|g_i|+|g_i|}$$

其中，$p_{g_j}(g_i)$ 代表 g_i 在组关系 (g_i, g_j) 中的所参与成边的原始节点集合，$p_{g_j}(g_i)=\{u \mid u \in g_i \text{ and } \exists v \in g_j \text{ s. t. } (u,v) \in E\}$。

组关系兼容度 $\delta_{i,j}$：$\delta_{i,j}$ 描述了组关系 (g_i, g_j) 与假定完美的组关系的兼容差异。其形式化定义为：$\delta_{i,j}=\delta_{g_j}(g_i)+\delta_{g_i}(g_j)$。

理想汇总图的组关系参与度要么是 100%(即两个组节点内的所有原始点全部参与成边)，要么是 0%(即所有点全不参与成边)。因此，参与度大于 50% 的组关系被称作强组关系，应该与“100%参与度”的理想情况比较兼容差异；反之，组关系被称作弱组关系，与“0%参与度”的理想情况比较。对于强组关系，$|p_{g_j}(g_i)|$ 表示 g_i 中有多少个节点与“理想情况”相符合；对于弱组关系，$|p_{g_j}(g_i)|$ 表示 g_i 中有多少个节点与“理想情况”不相符合。由此定义 g_i 在组关系 (g_i, g_j) 上的兼容度：

$$\delta_{g_j}(g_i)=\begin{cases}|p_{g_j}(g_i)|, & p_{i,j}\leqslant 0.5\\ |g_i|-|p_{g_j}(g_i)|, & \text{否则}\end{cases}$$

汇总图 Delta 度量 $\Delta(G_s)$：$\Delta(G_s)$ 统计了汇总图 G_s 中任意组关系的不兼容性。Delta 准则的形式化定义为：

$$\Delta(G_s)=\sum_{1\leqslant i<j\leqslant k}\delta_{i,j}$$

综上所述，K-SNAP 汇总图方法可描述为：以 Delta 获得最小值为目标，分组原始图中的节点而产生含有 K 个组节点的具有属性兼容性的汇总图。

寻找 Delta 获得最优解的汇总图已被证明是 NP 完全问题，K-SNAP 给出两种近似最优算法：自底向上和自顶向下。自底向上法的基本思想是：从原始图开始，每次选择使 Delta 准则值变化最小(即边兼容度最小)的两个等值节点进行合并，直到组节点集的个数减小至 K。自顶向下的基本思想是：首先分组所有等值节点，然后依次选择使 Delta 准则变化最小的组节点进行分裂，直到节点集的个数为 K。

3.3 汇总图规模受限

如前所述，K-SNAP 要求组节点对应的原始点等值，假设所有等值的原始点均被汇总到相同的组节点中，此时形成的汇总图被称作最大属性兼容汇总图。显然，K-SNAP 无法产生小于该规模的汇总图。因此，K-SNAP 产生汇总图的能力受限于最大属性兼容汇总图的规模。

另一方面，最大属性兼容汇总图的规模受原始图节点集在多维属性值空间的分布影响：分布越离散则规模越大；反之越小。当规模过大时，K-SNAP 无法产生“可读”规模的汇总图。节点属性的维度和属性值域较大时常常会导致这种情况发生，这是用户无法接受的。

本文例子中，图 1(d)为图 1(a)的最大属性兼容汇总图，其规模数为 6。该图中所有分组节点的值均不相同，无法继续汇总，因此，用户无法通过 K-SNAP 获得规模数小于 6 的汇总图。

K-SNAP 回避了该缺陷。在实验中，它选择值域较小的(如分类型)属性或对值域较大的(如数值型)属性作近似处理(对值域分段简化为分类属性)，使得最大属性兼容汇总图的规模非常小来确保 K-SNAP 能产生满足用户需求的汇总图。这样的处理方式牺牲了结果汇总图组节点的多样性，近似处理的程度不可控，且忽略了汇总过程中近似处理所造成的节点信息损失。

4 K-SGS 方法

为了克服上述缺陷，K-SGS 放宽了属性兼容约束：允许在语义上相似的原始节点被分配到同个组节点中。例如：图 1(e)中，我们可以将语义值相似的 P3 节点和 P9 节点合并到同一分组，因为这两名球员的球队归属地相同且年纪相似。K-SGS 产生的规模数小于 6 的汇总图如图 1(e)－(i)所示。

属性兼容约束的放宽虽然解决了 K-SNAP 的规模受限，但是会引入新的问题：1）如何度量组节点的语义相似性（后称作节点兼容性）和统计汇总图的节点兼容性损失；2）如何兼顾节点和边的损失。

4.1 汇总图的 Beta 度量

我们为节点建立兼容尺 $H=\{h_1,h_2,\cdots,h_m\}$ 来度量组节点的语义损失。H 包含一系列属性值兼容树 h_i，m 为关联属性个数。兼容树是描述属性值域上概念兼容关系的一种树形层次结构，它反映属性的本体语义信息以及属性值间的兼容关系。兼容树中节点的位置决定了其语义的“详细”程度和兼容能力：位置越接近根的属性值，其语义信息越模糊，但兼容的范围会越大，任意属性值能兼容以其为根的子树内的所有属性值。兼容树可以由领域专家进行人工构造。

图 1(f)分别给出了分类型属性 Location 和数值型属性 Age 的兼容树：对于分类型的属性 Location，属性值为“省”级的概念可以兼容属性值为“城市”，例如 DongGuan 市隶属于 GuangDong 省；对于数值型属性 Age，可通过人工方式建立区间来构建分类关系，例如使用区间[10,19]（并用 1 * 表示）来兼容所有 10 至 19 岁的属性值，隶属于该年龄段的球员均可以通过 1 * 表示。

我们使用具有有向边的节点标签树模型来描述属性值兼容树。树 $h_i=(V_i,E_i)$ 表示属性 a_i 的兼容树，V_i 为 h_i 的树节点集，V_i 中节点对应于属性 a_i 的取值范围，函数 $label(x)$ 与 $level(x)$ 分别表示属性值 x 对应节点的标识（编码）及其在 h_i 中的层数。E_i 为 h_i 的有向边集，E_i 中任意有向边为一有序二元组，有向边 $\langle v_1,v_2\rangle$ 表示 v_1，v_2 间存在兼容关系；E_i^* 为 E_i 的传递闭包，它包含了 h_i 中的所有路径，其描述的是 a_i 中任意属性值间的兼容关系。

我们首先定义单属性的兼容度量，再扩展到含有多个属性的节点，最后给出汇总图的 Beta 度量。

属性值兼容：给定兼容树 h_i，节点属性 a_i 的属性值 x，若兼容属性值 x' 在 E_i^* 存在 $\langle label(x),label(x')\rangle$，则称 x' 可以兼容 x，记为 $x\prec x'$。

属性值集兼容及其兼容度：给定属性值集合 Sa，兼容树 ha，兼容属性 x'，若任意 $x\in Sa$ 有 $x\prec x'$，则称 Sa 在 x' 上兼容，记为 $Sa\prec_{attset}x'$。

$Sa\prec_{attset}x'$ 意味着在 h_a 中，$label(x')$ 是 Sa 中所有属性值对应树节点集的公共祖先节点。Sa 的所有公共祖先节中，最近的公共祖先节点所对应的兼容属性值，称作 Sa 的最优兼容属性值，记作 $Sa\prec^{opt}_{attset}x'$。

例如：本例中，属性值集 NYH＝{Nanjin, Yiwu, Hangzhou}中任意城市均属于 JiangZhe 和 China，则有 NYH$\prec_{attset}$ JiangZhe 和 NYH$\prec_{attset}$China；又因为 JiangZhe 是 NYH 的最近公共祖先，则有 NYH$\prec^{opt}_{attset}$JiangZhe。

最优兼容属性值是衡量该属性值集中各属性值间语义相似性的重要标准。显然，最优兼容属性值在 h_a 中的位置越接近叶节点，其对应兼容属性值集的语义兼容性就越好。Sa 的兼容度 $\theta(Sa)$ 定义如下：

$$\theta(Sa)=\sum_{x\in Sa} level(x)-level(x'),Sa\prec^{opt}_{attset}x'$$

θ 是 Sa 中所有属性值（在兼容树中）到最优兼容属性值的距离和，其反映了使用最优属性值的属性来替换原始属性值所产生的语义损失。值得注意的是，兼容树中任意边可附带权重值，本文默认边权重为 1。

节点集兼容及其兼容度：给定节点集 Sp，点 $p=\langle x_1,x_2,\cdots,x_m\rangle$，设 Sp^i 为 Sp 在第 i 个属性上投影所得属性值集。对于 $\forall i\in[1,m]$，若有 $Sp^i\prec_{attset}x_i$，则称 Sp 在 p 上兼容，记为 $Sp\prec_{nodset}p$；若 $Sp^i\prec^{opt}_{attset}x_i$，则称 Sp 在 p 上最优兼容，记为 $Sp\prec^{opt}_{nodset}p$。

节点集的兼容度是各属性上兼容度之和，其反映了节点间的语义相似性，其定义为 $\omega(Sp)=\sum_{i=1}^{m}\theta(Sp^i)$。

节点集的兼容度是将单属性值集的定义扩展到多属性的情况，是对节点投影产生的多个属性值集的兼容度的累加。

汇总图的 Beta 度量：Beta 度量对汇总图中组节点所对应的原始节点集的兼容度进行累加，其形式化定义为：

$$\beta(\Phi)=\sum_{g\in\Phi}\omega(g)$$

4.2 K-SGS 方法概述

K-SGS 汇总图问题可描述为：给定规模参数 K，寻找 G 上 V 的某个划分 Φ，并基于 Φ 得到汇总图 G_s，使得：1）$|\Phi|=K$；2）$Beta(G_s)$ 与 $Delta(G_s)$ 获得最小值。

实际应用中，首先确定原始图节点集的规模下限，若用户设定的规模系数 k 大于该下限，则调用传统的 K-SNAP 方法，否则调用 K-SGS 算法。值得注意的是，K-SGS 汇总图的规模小于最大属性兼容汇总图，因此 K-SGS 无法设计成从最大属性兼容汇总图开始自顶向下的分裂算法，而只能设计为自底向上的合并算法。

初始时，将原始图的每个点视作组节点；每轮循环中，贪心地选择 NodeDiff 和 EdgeDiff 较小的一对组节点进行合并，直到点集的规模为 K。其中，NodeDiff 和 EdgeDiff 分别为 Beta 度量和 Delta 度量的启发函数，其定义如下：

给定组节点 g_i 和 g_j，

$$NodeDiff(g_i,g_j)=\frac{\omega(g_i\cup g_j)-\omega(g_i)-\omega(g_j)}{|g_i|+|g_j|}$$

$$EdgeDiff(g_i,g_j)=\sum_{t\neq i,j}|p_{i,t}-p_{j,t}|,t,i,j\in[1,\cdots|\Phi|]$$

NodeDiff 对合并前后节点的属性值兼容度变化均值进行了统计。EdgeDiff 统计了 g_i 和 g_j 与当前汇总图中其他组节点间参与度差异的累加，该差异越小，说明两个组节点在汇总图上越同构。

另一方面，K-SGS 具有 Beta 和 Delta 两个目标。我们采取分层序列和基于分级的统一函数两种策略来解决该多目标优化问题[13]。

分层序列的基本思想是：首先，根据 NodeDiff 在当前组节点对中选择具有较优值的部分组节点对作为候选解；其次，在候选解中选择 EdgeDiff 具有最优值的一对组节点作为最终选择。

值得注意的是，K-SNAP 方法首先保证了组节点的属性值等值要求，在此基础上进一步考虑 Delta 准则。这实际暗含着组节点的属性兼容性准则的优先级高于边的兼容性准则。因此，我们在使用多层序列法解决多目标优化问题时，将

Beta 作为第一序列目标函数，而 Delta 作为第二序列目标函数。

统一函数的基本思想是：根据 NodeDiff 和 EdgeDiff 分别对所有组节点对排序，每对组节点将获得两个序号，选择序号之和最小的组节点对为最终选择。

4.3 K-SGS 算法实现

K-SGS 算法分为预处理和核心算法两个阶段。预处理阶段对原始图节点进行编码，编码后的节点可以快速判断属性值间的兼容关系，提高算法效率；核心处理阶段实现了 4.2 节所述算法。

I. 预处理：基于 Dewey 编码的快速计算

在预处理阶段中，利用 Dewey 码来编码兼容树从而建立原始节点的数据字典。根据数据字典，将原始图的节点进行转码，转码后的节点作为核心处理算法的输入。

Dewey 码是一种快速树编码，常用于树的索引及检索。其编码规则描述为：根节点编码为“0”；若节点 i 的编码为“0…xx”，则节点 i 的第 t 个子节点的编码为“0…xxt”。Dewey 编码将兼容关系编码到兼容树节点，通过节点的 Dewey 码能快速地判断节点间的兼容关系。如图 1(f)所示，Age 和 Location 兼容树中每个节点的右上角数字为其 Dewey 编码。

在核心算法阶段，最优兼容组节点的构建和组节点间的 NodeDiff 计算被频繁调用。原始图节点经过转码后，我们能快速对上述操作进行处理，从而提高算法效率。表 1 中附加列 DeweyCode 记录了各节点根据 Age 和 Location 数据字典转码后的值。

给定节点集的 Dewey 码集，该节点集上最优兼容组节点的 Dewey 码由 Dewey 码集在各维属性上投影所产生的字符串集的最长公共前缀子串组成。例如：本例中，节点集{p_2，p_3}编码后为{(012,0122),(011,0121)}，Age 属性上的最长公共前缀子串为 01，Location 属性上最长公共前缀子串为 012，则{p_2，p_3}的最优兼容组节点 Dewey 码为(01,012)，即(1 *，Zhejiang)。

NodeDiff 的计算需要确定属性值在兼容树中所处的层数，而 Dewey 码的字符串长度即为层数。例如：本例中，Age 属性值 18 的 Dewey 码为 011，则 $level(18)=|011|=3$。

II. 核心处理：基于贪心策略的多目标优化

1)主要数据结构

分组指示器 $A[n]$：$A[n]$用于跟踪原始节点集的划分变化。数组元素 $A[0]$记录当前划分中的分组数，数组元素 $A[i](1\leqslant i\leqslant n)$记录节点标识(PID)为 i 的原始节点所隶属的分组标识(GID)。若数组元素的节点标识等于其分组标识(即 PID==GID)，称该节点为组头节点。算法中的主要计算根据组头节点进行。初始时，每个节点都是组头节点，即 $A[i]:=i$。合并多个分组为同一个分组时，选择这些数组元素中最小的 GID 作为合并后分组的 GID，并将这些数组元素更新为新分组的 GID。图 2 描述了合并 GID 为 l 和 m 分组前后分组指示器的变换，组头节点的 GID 用下划线显示。如图所示，合并前 $A[n]$中，GID=1 的分组包含节点 PID=1、2 和 $l+1$，GID=l 的分组包含节点 PID=l 和 $l+2$，GID=m 的分组包含节点 PID=m 和 $m+k$；合并后原来 GID=m 的节点均更新为 l。

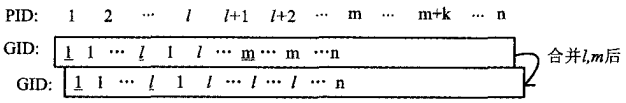

图 2 分组指示器变化过程

节点邻居位码表 $Neighbour[n]$：$Neighbour[n]$记录了原始节点与当前各分组间是否存在边，其数组元素为一个位码，第 i 个数组元素 $Neighbour[i]$的第 k 位保存了编号为 i 的节点与当前分组 $GID=k$ 的分组是否存在邻居关系，若存在则为 1，若不存在则为 0。若合并 $GID=l,m(l<m)$的分组后，$Neighbour[n]$的更新方法为：对每个数组元素的第 l 和第 m 位做异或运算，并将结果保存在第 l 位，图 3 描述了该过程。

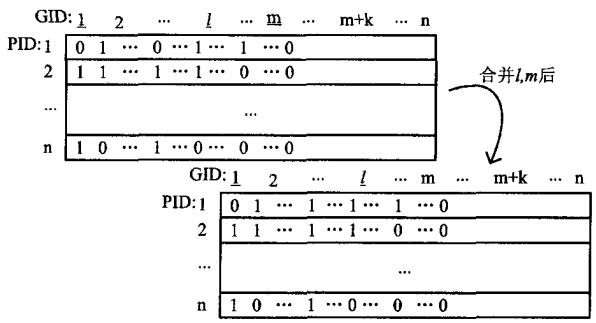

图 3 节点邻居位码表变化过程

NodeDiff 候选值集 BetaBTree：B 树索引 BetaBTree 记录当前分组情况下任意 $GID=i$ 和 j 的分组对的 $NodeDiff(i,j)$值。以该值作为 B 树的关键字，每个值伴有对应的两个分组的 GID。

EdgeDiff 变化值 DeltaBTree：B 树索引 DeltaBTree 记录当前分组情况下任意 $GID=i$ 和 j 的分组对的 $EdgeDiff(i,j)$值。以该值作为 B 树的关键字，每个值伴有对应的两个分组的 GID。

2)算法代码描述

算法 1 实现了分层序列法且每轮合并两个分组，代码如下所示。

Algorithm 1 KSGS_Pair(G,K,L)

```
Inuput:G,set 原始图包括节点集和边集;
       K,int 汇总图的规模因子;
       L,float 候选因子
Output:A[n],array G 上的一个划分
Begin
1. A[n] :=Init_Indicator();
2. Neighbour[n] :=Init_Neighbour(G);
3. BetaBTree :=Init_BetaBtTree(G);
4. DeltaBTree :=Init_DeltaBtTree(G);
5. While(A[0]>K)
6.    Candidate :=TopLSelect(BetaBTree,L);
7.    Choice :=MaxSelect(Candidate,DeltaBTree);
8.    IndicatorUpdate(A[n],Choice);
9.    NeighbourUpdate(Neighbour [n],Choice);
10.   BetaBTreeUpdate(BetaBTree,Choice);
```

```
11.  DeltaBTreeUpdate(DeltaBTree,Choice);
12. End While
13. Return A[n];
END
```

算法1行1－4为各数据结构初始化。Init_BetaBtTree函数成对计算两分组(初始时为原始图节点)合并为一个分组节点所产生的Beta变化值,并以该值作为关键字建立B树索引BetaBTree;Init_DeltaBtTree函数与之类似,对应为Delta准则。TopLSelect函数在BetaBTree中选取L个候选对,存放在Candidate候选对集。MaxSelect使用DeltaBTree在Candidate候选集中选择Delta准则变化最小的候选对作为本轮的最终选择Choice。合并Choice对应的两个分组,新产生的分组编号为Choice中较小的GID号。随后,更新各数据结构:IndicatorUpdate更新$A[n]$,将Choice中具有较大GID的分组中所有成员的GID更新为Choice中具有较小GID分组的GID号;NeighbourUpdate更新方法如前所述;BetaBTree-Update更新合并后的BetaBTree结构,其操作包括:删除节点对中包含Choice中较大GID的分组的Beta变化值,更新节点对中包含Choice中较小GID的分组的Beta变化值;DeltaBTreeUpdate与BetaBTreeUpdate相类似,不再累述。

输入规模为n、汇总图组节点规模为k时,BetaBTree和DeltaBTree中的B树关键字数为$O(n^2)$,查找和删除B树中某个关键字的代价为$O(\log_t n)$,t为每个分页中包含的关键字数,每轮循环中约需要删除和更新$O(n)$个关键字,故每轮的维护代价为$O(n\log_t n)$,算法共循环$(n-k)$次,故算法的执行代价约为$O(n(n-k)\log_t n)$。

算法1的贪心策略是基于每次合并两个分组的情况下进行的。为了提高算法的收敛性,我们期望在每轮迭代中能合并更多的分组。新的算法允许每轮合并两个及两个以上的分组,当一次合并m个分组时,可以认为连续经过了$m-1$轮的两两合并操作,而合并前后目标函数变化值除以$(m-1)$称作目标函数变化的单轮均值。其所采取的贪心策略为:选择在Beta变化值和Delta变化值在单轮均值上较小的候选对。例如,假设由$GID=3$和5产生的候选分组能兼容$GID=7$的分组,则其变化值的单轮均值为合并分组3,5,7前后Beta变化值和Delta变化值分别除以合并的分组数减一。单轮合并3,5,7的操作可以视作先合并分组3和分组5再合并分组(3,5)和分组7的连续两轮操作。另一方面,改进的算法对候选集中的项分别依据Beta和Delta的单轮均值进行分级的方法,每次选择Beta和Delta分级之和最小的项目进行合并。

算法2实现了基于分级的统一函数法,并且每轮可以合并多个分组,代码描述如下。

Algorithm 2　KSGS_RANK (G,K,L)

```
Inuput:G,set 原始图包括节点集和边集;
       K,int 汇总图的规模因子;
       L,float 候选因子
Output:A[n],array G 上的一个划分
Begin
1. A[n] :=Init_Indicator();
2. Neighbour[n] :=Init_Neighbour(G);
3. BetaBTree :=Init_BetaBtTree(G);
4. DeltaBTree :=Init_DeltaBtTree(G);
5. While(A[0]>K)
6.    Candidate :=TopLSelect(BetaBTree,L);
7.    CoveringTest(Candidate);
8.    BetaRanking(Candidate);
9.    DeltaRanking(Candidate,DeltaBTree);
10.   Choice :=SelectByRank()
11.   IndicatorUpdate(A[n],Choice);
12.   NeighbourUpdate(Neighbour[n],Choice);
13.   BetaBTreeUpdate(BetaBTree,Choice);
14.   DeltaBTreeUpdate(DeltaBTree,Choice);
15. End While
16. Return A[n];
END
```

函数CoveringTest遍历Candidate中每个元素,检查由这些分组对所产生的新分组(最优即合并后分组的最优兼容组节点)能否兼容其它分组,记录每个对所能兼容的分组号。BetaRanking根据"单轮NodeDiff变化均值"对Candidate中的候选对进行分级,每个候选对保存其分级号BetaRank,其计算方法为:统计当前候选对的最优兼容组节点所能兼容的分组,计算合并前后Beta值变化并除以兼容的分组数,从而得到NodeDiff变化均值。合并DeltaRanking与BetaRanking类似,根据"边差异均值"对Candidate中的候选对进行分级,每个候选对保存其DeltaRank。SelectByRank函数根据Candidate中的两种分级策略,选择分级号之和最小的候选对作为最终结果,并合并其所能兼容的所有分组。由于每轮可以合并多个分组,算法2收敛速度快于算法1,算法2具有更好的执行效率。

5　实验

5.1　实验环境

实验采用主频为1.6GHz的CPU(Intel CoreDuo E2140)以及1GRAM(DDR);算法使用CSharp实现,编译环境为VS.NET2005。

实际应用中的许多图模型都满足幂律度分布和小世界特性。因此,我们使用GTgraph图模型生成套件中的R-MAT模型来生成满足幂律度分布和小世界特性的原始图。对于原始图中的节点,使用真实数据集Adult Dataset的属性作为原始图节点的关联属性。其属性的特性描述如表2所列。

表2　实验基础数据集特性

属性名	属性值个数	兼容树层数
Age	74	4
Work Class	7	3
Education	16	4
Native Country	41	3

经数据清理后,Adult数据集中元组数约为45k,对于原始图中每个节点,我们用满足均匀分布的随机函数在Adult数据集中选择一个元组作为其关联属性值。按照上述处理方法,依据原始图的不同规模产生如下两组实验数据集,数据集特性如表3所列。

表 3　实验原始图特性

数据集	节点数	边数	不同值数	关联属性	节点平均度
D1	4267	16375	984	Age, Education	1.92
D2	29916	87852	16853	Age, Work Class, Native Country	5.8

5.2　实验结果及分析

如前所述，K-SGS 算法的主要目的是解决 K-SNAP 算法的规模缺陷问题。因此，当 K-SGS 算法在构建不存在规模缺陷的汇总图时会蜕变为 K-SNAP 算法，其算法效率、汇总图质量与 K-SNAP 算法无差别。本实验主要集中讨论构建出现规模缺陷的汇总图情况。

我们在上述数据集上比较 KSGS_Pair 算法和 KSGS_Rank 算法的效率并且检验所产生的汇总图的质量。汇总图的质量根据节点和边的兼容情况两个指标来检验。对于规模为 k 的汇总图 G_k，我们定义其节点兼容损失率为 $Beta(G_k)/Beta(G)*100\%$ 来显示汇总图的节点兼容质量；其边兼容质量用边兼容损失均值：$Delta(G_k)/K$ 来表示。

图 4 至图 9 显示了数据集 D1 和 D2 上算法 1 和算法 2 在不同规模数 k 上的实验结果。

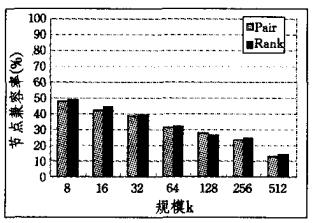

图 4　D1 的汇总图节点兼容质量

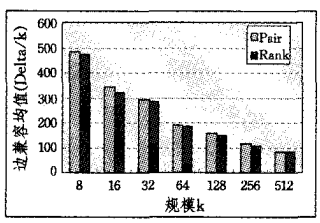

图 5　D1 的汇总图边兼容质量

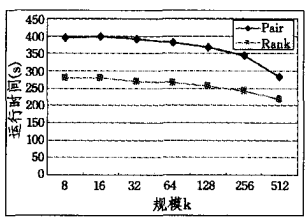

图 6　算法 1 和算法 2 的执行效率

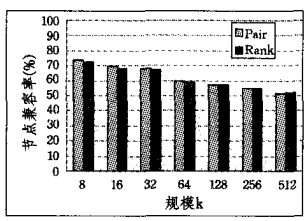

图 7　D2 的汇总图节点兼容质量

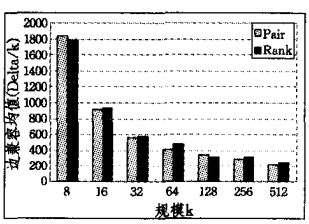

图 8　D2 的汇总图边兼容质量

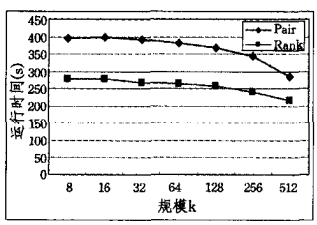

图 9　算法 1 和算法 2 的执行效率

图 4 和图 7 中，汇总图的节点兼容损失率随着规模数的增大而降低，这是因为更多的分组能使得分组内的节点集具有更好的兼容性。然而，任意合并若干分组并不一定会带来兼容性的绝对或者大幅降低，这是因为合并某两个分组为一个分组后，新分组的节点集最优兼容元组可能不变或者发生很小的变换。这一点体现为：图 4 中，规模为 64，128，256 的兼容质量相近而与 512 相远；图 7 中，规模从 64 到 512 的兼容质量相近。两种算法产生的汇总图的属性兼容质量基本相同，算法 1 略优于算法 2。这是由于在解决多目标规划问题时，算法 1 的分层序列策略相对于算法 2 的基于分级的统一函数策略更偏向于属性的兼容质量。

汇总图的边兼容质量随着规模的增大而提高，这是因为更多的分组能保留更多的原始边信息。两种算法产生的汇总图的边兼容质量基本相同，算法 2 的边兼容质量略优于算法 1，这同样是由解决多目标规划采取的策略所导致的。图 5 和图 8 反映了上述情况。

图 6 和图 9 显示了两种算法的执行效率，算法 2 相对于算法 1 具有更好的执行效率，这是因为算法 2 具有更快的收敛速度和更少的循环次数，特别是在算法执行初期每轮循环中可以快速合并多个分组；然而到算法执行后期，随着汇总图规模数的减小，每轮合并多个分组的机会也变小，两种算法收敛速度开始近似。另一方面，对于大数据集 D2，由于算法需要更多的 I/O 操作，相对于小数据集算法执行时间会大大增长。

结束语　K-SGS 方法是一种基于节点概念分层的图汇总算法，它解决了传统 K-SNAP 算法的汇总图规模参数受限问题，并且在汇总过程中兼顾了节点的信息损失。

今后的研究中，我们将设计针对更大规模图模型的轻量级汇总算法，例如通过调整图模型在不同分辨率下的视图来加快汇总算法。

参 考 文 献

[1] Chakrabarti D, Faloutsos C. Graph mining: Laws, generators, and algorithms[J]. ACM. Comput. Surv., 2006, 38(1)

[2] Chakrabarti D, Faloutsos C, Zhan Y. Visualization of large networks with min-cut plots, A-plots and R-MAT [J]. Int. J. Hum. -Comput. Stud., 2007, 65(5): 434-445

[3] Newman M E J. The structure and function of complex networks[J]. SIAM Review, 2003, 45: 167-256

[4] Huan J, Wang W, Prins J, et al. SPIN: Mining maximal frequent subgraphs from graph databases[C]// Proceedings of KDD'04. 2004: 581-586

[5] Wang W, Wang C, Zhu Y, et al. GraphMiner: A structural pattern-mining system for large disk-based graph databases and its applications[C]// Proceedings of SIGMOD'05. 2005: 879-881

[6] Sun J, Xie Y, Zhang H, et al. Less is more: Sparse graph mining with compact matrix decomposition[J]. Stat. Anal. Data Min., 2008, 1(1): 6-22

[7] Xu X, Yuruk N, Feng Z, et al. SCAN: A structural clustering algorithm for networks[C]// Proceedings of KDD'07. 2007: 824-833

[8] Battista G, Eades P, Tamassia R, et al. Graph Drawing: Algorithms for the Visualization of Graphs[M]. PrenticeHall, 1999

[9] Herman I, Melancon G, Marshall M S. Graph visualization and navigation in information visualization: A survey [J]. IEEETrans. Vis. Comput. Graph., 2000, 6(1): 24-43

[10] Navlakha S, Rastogi R, Shrivastava N. Graph summarization with bounded error[C]// Proceedings of SIGMOD'08. 2008: 567-580

[11] Tian Y, Hankins R, Patel J M. Efficient aggregation for graph summarization[C]// Proceedings of SIGMOD'08. 2008: 419-432

[12] Zhang N, Tian Y, Patel J M. Discovery-Driven Graph Summarization[M]. Data Engineering (ICDE), 2010

[13] 林锉云. 多目标优化的方法与理论[M]. 长春：吉林教育出版社，1992