基于 FSG 的最大频繁子图挖掘算法 *

郭景峰, 柴 然, 张 伟
(燕山大学 信息与工程学院, 河北 秦皇岛 066004)

摘 要: 图挖掘已成为数据挖掘领域研究的热点,然而挖掘全部频繁子图很困难且得到的频繁子图过多,影响结果的理解和应用。可通过挖掘最大频繁子图来解决挖掘结果数量巨大的问题,最大频繁子图挖掘得到的结果数量很少且不丢失信息,节省了空间和以后的分析工作。基于算法 FSG 提出了最大频繁子图挖掘算法 FSG-MaxGraph;结合节点的度、标记及邻接列表来计算规范编码,提出两个定理来减少子图同构判断的次数,并应用改进后的决策树来计算支持度。实验证明,新算法解决了挖掘结果太多理解困难的问题,且提高了挖掘效率。

关键词: 数据挖掘; 规范编码; 最大频繁子图; 决策树; 子图同构
中图分类号: TP311 **文献标志码**: A **文章编号**: 1001-3695(2010)09-3303-04
doi:10.3969/j.issn.1001-3695.2010.09.027

FSG-based algorithm for mining maximal frequent subgraph

GUO Jing-feng, CHAI Ran, ZHANG Wei
(*College of Information & Engineering, Yanshan University, Qinhuangdao Hebei* 066004, *China*)

Abstract: Graph mining has become a hot topic in the field of data mining, however, mining all frequent subgraph is very difficult and will get excessive frequent subgraph this impact on the understanding and application of the outcome. Through mining maximal frequent subgraph to solve the problems of the number of the result is huge. Maximal frequent subgraph mining obtained a small number of the results and this without loss information, mining maximal frequent subgraph saved space and the work of analysis. This paper based on algorithm FSG proposed an algorithm FSG-MaxGraph for mining maximal frequent subgraphs. Combined the degree, nodes and adjacency list to calculating normal matrix coding and proposed two theorems that could reduce the times of subgraph isomorphism this improve the efficiency of the algorithm. Last, used the improved decision tree to computing support. The experiment can prove the new algorithm can solve the problem of the mining results difficult to understand and this new algorithm can improve the efficiency of mining.

Key words: data mining; canonical code; maximal frequent subgraph; decision tree; subgraph isomorphism

数据挖掘是从存放在数据库或其他信息库中的大量数据中挖掘有趣知识的过程。最初数据挖掘的研究对象是结构化的数据,而如何从半结构化(XML 文本、有序树等)和非结构化(科学数据、空间数据等)数据中发现知识,是所面临的技术难题。图结构能够模拟几乎所有事物之间的联系,它能应用到半结构化和非结构化的数据挖掘中。但是图这种数据结构非常复杂,增加了挖掘令人感兴趣的子图的难度。其中子图同构及图同构的判断起着至关重要的作用,这一过程往往是 NP 完全问题。

在大的图数据库中进行高效数据挖掘的关键问题是,如何处理挖掘得到的数量巨大的频繁模式。挖掘全部频繁子图的代价高且得到的结果数量巨大,严重地影响结果的理解和应用。最大频繁子图隐含了所有频繁子图,即其所有的子图都是频繁的,而且某些数据挖掘应用只需发现最大频繁子图。最大频繁子图的挖掘大大减少了挖掘得到的子图的数量,可大大节省空间和以后的分析工作,且挖掘最大频繁子图并没有丢失信息,所以可将频繁子图挖掘问题转换为挖掘最大频繁子图。

1 图的数据挖掘原理

最大频繁子图挖掘必须解决子图同构问题,该问题已经被证明是 NP 完全问题,所以要尽可能地减少子图同构的判断,降低算法的时间复杂度。

1.1 关于图的一些概念

定义 1 标记图。用一五元组 $G=(V,E,\Sigma V,\Sigma E,l)$ 表示。其中:V 是非空的节点集合;E 是边的集合;ΣV、ΣE 分别为节点标记和边标记的集合;l 定义为 $V\to\Sigma V, E\to\Sigma E$ 的映射。

定义 2 子图。给出一对标记图 $G_1=(V_1,E_1,\Sigma V_1,\Sigma E_1,l_1)$ 和 $G_2=(V_2,E_2,\Sigma V_2,\Sigma E_2,l_2)$,$G_1$ 为 G_2 的子图当且仅当 V_1、E_1 分别包含于 V_2、E_2,且任取 $u\in V_1$,$l_1(u)=l_2(u)$,任取 $(u,v)\in E_1$,$l_1(u,v)=l_2(u,v)$。

定义 3 同构。一标记图 $G_1=(V_1,E_1,\Sigma V_1,\Sigma E_1,l_1)$ 同构于另一图 $G_2=(V_2,E_2,\Sigma V_2,\Sigma E_2,l_2)$ 当且仅当存在一个映射 f:$V_1\to V_2$ 且任取 $u\in V_1$,$(l_1(u)=l_2(f(u)))$,任取 $u,v\in V_1$,$((u,v)\in E_1$ 则 $(f(u),f(v))\in E_2)$ 且任取 $(u,v)\in E_1$,$(l_1(u,v)=l_2(f(u),f(v)))$。

收稿日期: 2010-03-09; **修回日期**: 2010-04-30 **基金项目**: 国家自然科学基金资助项目(60673136);河北省教育厅 2009 年自然科学研究基金资助项目(2009101)

作者简介:郭景峰(1962-),男,河北秦皇岛人,教授,博士,CCF 会员,主要研究方向为数据库理论及应用、数据挖掘技术等;柴然(1985-),女,河北沧州人,硕士,主要研究方向为图挖掘(chairan1234@163.com);张伟(1985-),女,河北保定人,硕士,主要研究方向为图挖掘.

定义 4 子图同构。如果说图 G_1 子图同构于图 G_2，当且仅当在图 G_2 中存在子图 G_2'，使得 G_2' 同构于 G_1。

如图 1 中(b)是(a)的子图且子图同构于(a)。

定义 5 支持度。对一个图数据库 $D=\{G_1,G_2,\cdots,G_n\}$ 及用户给定的最小支持度 $\text{minsup}\in(0,1]$，令

$$\zeta(g,G)=\begin{cases}1 & \text{如果 } g \text{ 与 } G \text{ 子图同构}\\ 0 & \text{如果 } g \text{ 与 } G \text{ 的任意子图不同构}\end{cases}$$

设 $\sigma(g,D)=\sum\zeta(g,G_i)$，$G_i\in D$，则 $\sigma(g,D)$ 是图 g 在 D 中出现的频率，则 g 在 D 中的支持度 $\sup(g,D)=\sigma(g,D)/n$，n 为图数据库 D 中图的数量。频繁子图挖掘就是从 D 中找出所有满足如下条件的子图 g：$\sup(g,D)\geqslant\text{minsup}$。

如果说图 g 是最大频繁子图，则不能在图集 D 中找到一个图 G 使得 g 子图同构于它，且 $\sup(G)\geqslant\text{minsup}$。

1.2 数据结构

本文对图的数据结构定义采用邻接矩阵表示法，只考虑无向图，根据对称性，只保留邻接矩阵的下三角阵。无向图 G 的邻接矩阵元素 X_{ij} 定义为

$$X_{ij}=\begin{cases}0 & i<j\\ \text{顶点标记} & i=j\\ \text{边标记或 }0 & i>j\end{cases}$$

当 $i>j$ 时 X_{ij} 的值取决于顶点 V_i 与 V_j 之间是否有边，当有边时 X_{ij} 的值为边标记；否则为 0。这样图 1(a)的邻接矩阵表示如图 2(a)与(b)所示。

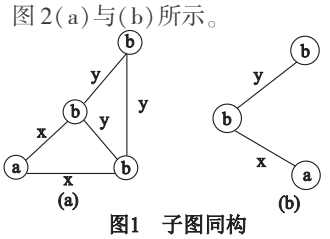

图1 子图同构

a	0	0	0
x	b	0	0
x	y	b	0
0	y	y	b

(a)

b	0	0	0
y	b	0	0
y	y	b	0
0	x	x	a

(b)

图2 邻接矩阵

因为节点可以按不同的顺序排列，所以一个图可以得到多个不同的邻接矩阵。为此本文引入了规范矩阵的概念，使得图与矩阵形成一一对应的关系。对符合以上定义的邻接矩阵进行如下方式的编码：$\text{code}(X)=X_{11}X_{21}X_{22}\cdots X_{n1}X_{n2}\cdots X_{n,n-1}X_{nn}$，即为将包含对角线元素的下三角矩阵元素按从第一行到第 n 行且每行按从左到右的顺序串联起来。

规范矩阵的定义为：当图 G 只有一个邻接矩阵时，该邻接矩阵就是图 G 的规范矩阵；当图 G 存在多个邻接矩阵时，取 code 值最大的邻接矩阵作为图 G 的规范矩阵，记为 $cl(G)$。图 2 中两个邻接矩阵的编码分别为 code(2a)=axbxyb0yyb，code(2b)=bybyyb0xxa。计算图的同构等价于计算图的规范编码，因为若两图同构则它们的规范编码相同。计算图的规范编码的一般方法是列出图的所有邻接矩阵，计算每个邻接矩阵的编码，然后进行一一比对从中找出最大的一个，复杂度为 $m!$（m 为顶点数量）。

人们又提出了按照节点的度和标记对节点进行排序，可以提高计算效率。本文提出了一种新的节点排序策略，即按照节点的度、标记和邻接列表对节点进行排序。

用元组 $(l(e),d(v),l(v))$ 来表示一个节点，其中：$l(e)$ 表示的是与节点 v 相关联边的标记；$d(v)$ 为节点的度即与节点相关联的边数；$l(v)$ 为节点的标记。节点 v 的列表记为 $nl(v)$ 其中包括每个与 v 邻接的节点的元组。利用邻接列表将节点分为不相交的集合，如果节点 u、v 在同一个分区当且仅当 $nl(u)=nl(v)$。下面用一个实例进行说明。图 3 给出了一个含有七条边的图，图 4(a)为只结合节点的度和标记得到的分区，(b)为结合邻接列表后的分区。

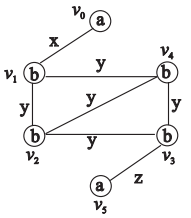

图3 G_1

b	y	0	y	x	0
y	b	y	y	0	0
0	y	b	y	0	z
y	y	y	b	0	0
x	0	0	0	a	0
0	0	z	0	0	a

(a)

b	0	0	0	0	0
y	b	0	0	0	0
0	y	b	0	0	0
y	y	y	b	0	0
x	0	0	0	a	0
0	0	0	z	0	a

(b)

图4 规范编码示例

如图 4(a)因为节点 v_1、v_2、v_3、v_4 的度均为 3，所以分在同一个分区内，节点 v_0、v_5 的度为 1 分在同一个分区内，且将度大的节点排在前面。因为节点的度体现了与该节点相连接的边的情况，度越高相连接的边越多，使得非 0 元素尽可能地出现在编码的前面。在本例中结合节点的度进行排序后，要得到规范编码需进行 $4!\times2!$ 次比较。

如图 4(b)因为在按节点的度和标记进行分区后，在同一分区中的节点 v_2 与 v_4 的邻接列表仍相同均为 $(y,3,b)$，$(y,3,b)$，$(y,3,b)$，其余的节点的邻接列表互不相同可对其进行更细致的划分，得到规范编码要进行 $2!$ 次比较。

2 算法的改进

算法 FSG 是基于 Apriori 的，采用逐级扩展的策略来发现图数据库中的频繁子图，FSG 的关键特性是：用稀疏图来表示使得存储和计算都最小化；通过每次添加一条边的方法来扩展图；用规范编码来惟一表示一个图；可以处理非常大的数据库，并可以在有效的时间内发现所有的频繁子图。本文对其进行改进应用到挖掘最大频繁子图中，该算法与 FSG 不同的是判断是否拥有相同 $k-1$ 子图的方法，及产生候选子图后是否添加到候选最大频繁子图的集合中的条件，以及本文得到的结果集是最大频繁子图的集合。

2.1 候选子图的产生

算法 FSG 在产生 $k+1$ 候选子图时，通过连接两个拥有相同 $k-1$ 子图的 k 子图来实现。在判断两个 k 子图是否有相同的 $k-1$ 子图时采用每次删除一条边的方法，然而子图同构判断是一个 NP 完全问题，先删除哪条边才会减少子图同构判断的次数。比如，两个 k 子图 G_i^k、G_j^k 不拥有相同的 $k-1$ 子图，需要 k 次子图同构判断才能得出结论，这是很耗时的。本文针对这个问题提出了两个定理，在删除边之前先进行判断，在一定程度上减少了计算量。在给出定理之前先作以下陈述：

a)对图按如下规则进行扫描，这并不增加时间复杂度，因为在计算规范编码时同样要扫描图，在这里扫描图不仅得到节点的度同时还得到边的相关信息。(a)将每条与顶点相关联的边都记录下来，边用四元组 $(l(v_i),l(v_j),l(v_i,v_j),a)$ 来表示，$l(v_i)$、$l(v_j)$ 分别为连接边的两个顶点的标记，$l(v_i,v_j)$ 为边标记，a 是边 $(l(v_i),l(v_j),l(v_i,v_j))$ 在图中出现的次数。(b)表示边的四元组中第一个数相同的边组成一个集合 E_i。(c)当边在图 G_i^k 中第一次出现时计数为 1，如再次出现计数值加 1 而不必对相同的边重复存储。(d)最后得到一个集合 E，E 以每个小的集合 E_i 为元素。提出这些规则是为下一步的判断提供方便节省时间。扫描两个图 G_i^k 和 G_j^k 得到两个集合 E_1 和 E_2。

b)判断图 G_i^k 中某边是否在图 G_j^k 中不存在相同边，这样边

的数量。依次取图 G_i^k 中的边与图 G_j^k 中的边进行比较,可通过比较 E_1 和 E_2 的元素来实现。依次取 E_1 中的元素表示的边 e,将其与 E_2 中的元素比较,若 e 与 E_2 某一表示边的四元组的第一个数不同则小集合中的其他四元组不用进行比较,因为同一个小集合中的所有四元组的第一个数相同。然后与下一小集合中元素进行比较,其他小集合同理,如出现相同的边则停止比较;对 E_2 中四元组第一个数和表示边 e 的四元组中第一个数相同的元素都比较完后,如无相同的边,则其他的小集合不用再判断。标记将 e 的两个顶点标记交换顺序且其他均相同的边,因为 e 与其实际上表示的为同一条边,只是访问顶点的顺序不同。

给出一个计数值 n 初始为 0,用 n 来表示图 G_i^k 中的不能在 G_j^k 中找到相同边的边数量。如果在 E_2 中发现某边与边 e 相同只是边的计数不同,且在 E_2 中边的计数小于在 E_1 中该边的计数,或 e 与 E_2 中所有元素比较完后没有发现相同的边则增加 n 值,将引起 n 值增加的边存储起来。

设边$(l(v_i),l(v_j),l(v_i,v_j))$在图 G_i^k 与 G_j^k 中的计数(若图 G_j^k 中不存在该边则计数为 0)差值为 c,c 大于 0 则增加 n 值:(a)$l(v_i)=l(v_j)$则 n 增加 $c/2$。(b)若 $l(v_i)\neq l(v_j)$则增加 c。

定理 1　如果 $n\geq 2$ 图 G_i^k 与 G_j^k 无相同的 $k-1$ 子图,不用进行子图同构判断。

证明　$n\geq 2$ 说明图 G_i^k 与 G_j^k 至少存在两条不相同的边。子图的定义中有一个条件是:若图 G_1 为图 G_2 的子图,则图 G_1 的边集是图 G_2 边集的子集。无论删除图 G_i^k 中哪一条边,得到的 $k-1$ 子图的边的集合都会至少包含一条 G_j^k 中不存在的边,则 $k-1$ 子图边的集合不是图 G_j^k 边集合的子集,不能满足子图定义的条件,所以 $n\geq 2$ 时图 G_i^k 与图 G_j^k 无相同的 $k-1$ 子图。

定理 2　如果对 E_1 中的所有边进行判断后 $n=1$,此时图 G_i^k 与 G_j^k 可能拥有相同的 $k-1$ 子图,在删除边进行判断时可只删除引起 n 值增加的边。

证明　$n=1$ 说明图 G_i^k 至少存在一条边(设边为 f)不能在 G_j^k 中找到相同的边。在判断 G_i^k 与 G_j^k 是否拥有相同的 $k-1$ 子图时删除 G_i^k 中任意一条不为 f 的边得到 $k-1$ 子图,该 $k-1$ 子图的边的集合均会包含这条边 f,所以不能满足子图定义中 $k-1$ 子图的边集应为 G_j^k 边集子集的条件,得到的 $k-1$ 子图不是图 G_j^k 的子图,只需删除引起 n 值增加的边 f,就可判断两图是否有相同的 $k-1$ 子图。

这两个定理在以下情况下能很好地减少子图同构判断的次数:当图中无重复节点标记或重复节点标记很少时;节点的度和数量很大边数较少;没有相同的 $k-1$ 子图时。因为当图中无重复节点标记时,一个边的三元组$(l(v_i),l(v_j),l(v_i,v_j))$体现了该边在图中的位置及边的标记等各种信息,判断出 n 值后可根据 n 值直接判断有无相同的 $k-1$ 子图。如果节点的度很大,根据扫描图的几个原则,可以减少扫描完图后的判断环节的复杂度,因为如果 E 中的某条边 e 与 E_2 中某小集合的第一个四元组的第一个数不同则该小集合的其他元素不用进行比较,可节省时间。判断环节的复杂度为 $O(k^2)$,当节点数很多时即使根据定理不能判断出是否拥有相同的 $k-1$ 子图,其复杂度与计算子图同构的相比可以忽略,尤其是两图无相同的 $k-1$ 子图时(因为此时需要进行 k 次子图同构判断)。提出了两个定理之后虽然不能避免子图同构的判断,但是可减少了子图同构判断的次数,因为子图同构判断是一个 NP 完全问题,虽然在本文中提出了好的计算规范编码的方法,但是计算规范编码同样很复杂,减少子图同构判断的次数可减少了计算量。所以提出这两个定理是有必要的。

当对 E 中的所有边进行判断后 $n=0$,不能通过定理来判断,则按 FSG 中提到的方法来确定两个图是否拥有相同的 $k-1$ 子图。如拥有相同的 $k-1$ 子图则可进行连接,通过连接得到 $k+1$ 子图后,只有当这个 $k+1$ 子图满足两个条件时才作为候选最大频繁子图加入到候选集中,即在候选最大频繁子图集合中不存在它的超集,且它的所有 k 子图均是频繁的(这是根据 Apriori 算法的性质提出来的)。MFG 为最大频繁子图集合,初始为图集中的最大频繁事务(即不存在其他的事务为它的频繁超集)。算法 1 给出了候选最大频繁子图产生的伪代码。

算法 1　fsgmg-gen(FG^k) (Candiate Generation)

```
1 C^{k+1} ← Φ
2 for 从 FG^k 中任取 G_i^k, G_j^k, i <= j
3     if cl(G_i^k) <= cl(G_j^k)
4         for each edge e ∈ G_i^k
5             G_i^{k-1} ← G^k - e_i
              //通过删除一条边来创建 G_i^k 的一个(k-1)子图
6             if G_i^{k-1} 包含于 G_j^k 中
              then{G_i^k 与 G_j^k 拥有相同的 k-1 子图}
7             T^{k+1} = fsg-join(G_i^k, G_j^k)
8             for each G_j^{k+1} 属于 T^{k+1} do
9             if 最大频繁子图集合 MFG 及
              C^{k+1} 中不存在 G_j^{k+1} 的超集 then
10                任取一条边 el ∈ G_j^{k+1} do
11                    H_l^k = G_j^{k+1} - el
12                    if H_l^k 不是频繁的 then
13                        flag = false
14                        break
15            if G_j^{k+1} 满足向下封闭的性质
              //即它的所有 k 子图均是频繁的
16            flag = true then
17        C^{k+1} = C^{k+1} ∪ {G_j^{k+1}}
18 return C^{k+1}
```

在 Apriori-MaxGraph 算法中还提到,会利用性质剪掉候选子图集合中当前最大频繁子图的子集,这会造成本应产生的候选子图,因为不满足连接条件,而无法产生,从而提出了恢复策略,但需要进行多次的子图同构判断,这是很耗时的。本文在下节提出了决策树,所有的频繁子图在树中对应一个节点,可以避免上述情况的出现,同时可以利用决策树来判断 H_l^k 是否是频繁的。

2.2　支持度的计算

本文中利用决策树来计算支持度,在算法 FSM 中采用每次增加一个顶点的方法来构造决策树,而算法 FSG 通过每次增加一条边的方式来扩展图,所以要对 FSM 中的决策树的构造方法进行改进,不是每次增加一个顶点而是每次增加一条边,每一个频繁子图对应着树中的一个节点。如果表示该图的节点为 leaf node 则该图的支持度为 1,如果为非 leaf node,则它的支持度为以该节点为 root node 的决策树中 leaf node 所表示的不重复的图数量。

决策树示例如图 5 所示。假设图集含有两个图即图 G_1 和 G_2,则图集的决策树如图 6(a)和(b)所示。由于空间原因只给出由顶点 A 扩展而得到的子图的决策树,(b)中第一层的节点与(a)中的(A)为同一节点(子图用边集来表示)。

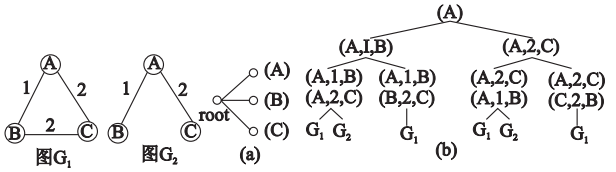

图4　决策树

决策树是基于机器学习的数据挖掘技术,它形式简单,分类速度快,无须先验知识,对样本分布无要求。而且由决策树表达的规则直观清晰,生成决策树的时间复杂度为 $O(NM_i)$。其中:N 为图集中边数最多的图的边数,i 是图集中图的数量,M 为决策树中每层子图的置换矩阵的平均数量。如使用原始的方法来计算每个生成的子图的支持度都要进行 i 次子图同构判断,计算量是巨大的。本文引入决策树后很好地减少了计算支持度时进行子图同构判断的次数,使时间复杂度降低。

2.3　最大频繁子图的产生

算法 2 给出了最大频繁子图产生的伪代码。

算法 2　MFG-gen(C^{k+1})(Maximal Frequent Subgraph Generation)

1 MFG←图集中的最大频繁事务//当前发现的最大频繁子图的集合

2 for each candidate $G_j^{k+1} \in C^{k+1}$ do

3 found G_j^{k+1} 在决策树 T 中

4　if G_j^{k+1} 是决策树的 leaf node

5　　$sup(G_j^{k+1})$←1

6　else $sup(G_j^{k+1})$←以该图所对应的节点为根节点的决策树中不重复的 leaf node 的数量

7　if $sup(G_j^{k+1}) >= minsup$

8　　if 在 MFG 中不存在 G_j^{k+1} 的超集

9　　MFG←MFG∪G_j^{k+1}

10　　else 如果 G_j^{k+1} 减去某条边 e 得到的子图 H_1^k 是频繁的//可以利用决策树来计算它的支持度

11　　且 H_1^k 不属于 MFG

12　　MFG←MFG∪(H_1^k)

13return MFG

3　实验结果

通过 FSG-MaxGraph 算法,采用的数据集总共包含 233 644 个化合物,它是从 developmental therapeutics program(DTP)获得的,从数据集的 223 644 个化合物中随机选取 10 000 个组成一个图集 D。实验结果如表 1 所示。支持度由 10% 变化到 2%,ǔ F ù 为发现的所有频繁子图的数量,ǔ M ù 为发现的最大频繁子图的数量,最后一列为运行时间单位为 s。

表 1　算法 FSG-MaxGraph 运行结果

σ	ǔ F ù	ù M ù	t/s
10.0	267	9	66
8.0	405	10	80
7.0	606	11	91
6.0	713	12	107
5.0	939	12	118
4.0	1 431	12	220
3.0	2 489	14	1 300
2.0	3 807	16	1 500

本实验的主要目的是评价各种优化对候选产生的影响,来说明 FSG-MaxGraph 发现最大频繁子图的效力。

4　结束语

本文提出了一种挖掘最大频繁子图的新算法 FSG-MaxGraph。首先给出了一种根据节点的度,标记及邻接列表计算邻接矩阵的规范编码的方法,大大降低了求规范编码的复杂度;其次提出两个定理,可以减少判断两图是否拥有相同的 $k-1$ 子图时子图同构判断的次数;最后将决策树应用到算法中提高计算支持度的效率。实验结果表明算法 FSG-MaxGraph 有很高的挖掘效率。

参考文献:

[1] 胡作霆,董兰芳,王洵. 图的数据挖掘算法研究[J]. 计算机工程, 2006, 32(3):76-78.

[2] GOLUMBIC M, HARTMAN I. Graph theory, combinatorics and algorithms[M]. New York: Springer Press, 2005: 101-128.

[3] REN Wei, ZHOU Yang. FSM: A frequent subgraph mining algorithm based on subgraph and structure isomorphism[J]. 西南大学学报, 2008, 30(6):158-163.

[4] 王映龙, 杨炳儒, 宋泽峰, 等. 一种挖掘最大频繁子图的新算法[J]. 系统仿真学报, 2008, 20(18):4872-4877.

[5] 严蔚敏,吴伟民. 数据结构(C 语言版)[M]. 北京:清华大学出版社, 2007:20-38.

(上接第 3296 页)尽管算法引入了一定误差,但却大幅缩短了方位估计时间,且 10^{-1} 量级的测向误差对于水下目标定向等工程应用而言,具备了一定的实用价值。此外,在计算复杂度相差不大的情况下,本文提出的改进量子进化算法与传统 QEA 算法相比,方位估计的精确性和稳定性均更优。但是,QEA 类优化算法的估计精度仍有待进一步提高,这将是今后研究的重点。

参考文献:

[1] HEY T. Quantum computing: an introduction[J]. Computing & Control Engineering Journal, 1996, 10(3): 105-112.

[2] 王凌. 量子进化算法研究进展[J]. 控制与决策, 2008, 23(12): 1321-1326.

[3] HAN K H, KIM J H. Quantum-inspired evolutionary algorithm for a class of combinatorial optimization[J]. IEEE Trans on Evolutionary Computation, 2002, 6(6): 580-593.

[4] TALBI H, DRAA A, BATOUCHE M. A new quantum-inspired genetic algorithm for solving the travelling salesman problem[C]//Proc of IEEE International Conference on Industrial Technology. Washington DC: IEEE Computer Society, 2004: 1192-1197.

[5] ZHANG G X, GU Y J, HU L Z, *et al*. A novel genetic algorithm and its application to digital filter design[C]//Proc of IEEE International Conference on Intelligent Transportation Systems. 2003: 1600-1605.

[6] 李映,张艳宁,赵荣椿,等. 免疫量子进化算法[J]. 西北工业大学学报, 2005, 23(4): 543-547.

[7] 王惠刚,李志舜,孙进才. 基于相位匹配原理的稳健方位估计[J]. 电子与信息学报, 2005, 27(2): 189-191.

[8] SCHMIDT R O. Multiple emitter location and signal parameter estimation[J]. IEEE Trans on AP, 1986, 34(3): 276-280.

[9] KRIM H, VIBERG M. Two decades of array signal processing research[J]. IEEE Signal Processing Magazine, 1996, 13(4): 67-94.

[10] RAO C R. Handbook of statistics 9: computational statistics[M]. Amsterdam: North-Holland, 1993: 467-508.