一个基于着色和动态规划的 3-维匹配问题算法*)

宁 丹 王建新

(中南大学信息科学与工程学院 长沙 410083)

摘 要 3-维匹配问题是六个经典的 NP 完全问题之一,在调度、分配、交通和网络流等问题方面有很强的应用。参数计算理论是近年来发展起来的研究和解决 NP-难问题的新方法。针对 3-维匹配问题,目前确定式参数算法的最好结果是 $O^*(16^{3k})$。本文结合着色和动态规划技术,提出了一个算法运行时间为 $O^*(3.42^{3k})$的确定式参数算法,大大提高了算法的运行效率。

关键词 3-维匹配,着色,动态规划

An Algorithm for 3-Dimensional Matching Problem Based on Color Coding and Dynamic Programming

NING Dan WANG Jian-Xin

(School of Information Science and Engineering, Central South University, Changsha 410083)

Abstract 3-Dimensional Matching is one of the six classic NP-complete problems, which has extensive application in scheduling, assignment, transportation and network flow problem, etc. Parameterized computation theory is a recently developed new method to study and solve NP- hard problems. At the present, for the 3-Dimensional Matching problem, the best result of deterministic parameterized algorithm is $O^*(16^{3k})$. This paper combines color coding and dynamic programming, and presents a deterministic parameterized algorithm of running time $O^*(3.42^{3k})$, which significantly improves the previous best deterministic algorithm.

Keywords 3-Dimensional matching, Color coding, Dynamic programming

1 引言

图的匹配问题(Matching Problem)在调度、分配、交通和网络流等方面有很强的应用。下面首先给出问题的定义[1]:

定义 1(图的匹配问题) 给定一个图 $G=(V(G),E(G))$,$V(G)$表示图 G 中点的集合,$E(G)$表示图 G 中边的集合。图 G 的一个匹配 $P(G)$是 $E(G)$的一个子集,且 $P(G)$中任何两条边都不存在公共点。如果匹配 $P(G)$中的边可以覆盖图 G 的每一个点,则该匹配是完美匹配。

3-维匹配问题(3-Dimensional Matching problem,简称 3-D Matching problem)是图的匹配问题中的一个很典型的问题,其定义可以描述为[2]:

定义 2(3-维匹配问题) 给定一个集合 $M\subseteq X\times Y\times Z$,其中 X,Y,Z 是 3 个互不相交的大小为 n 的符号集合,我们要从中找出一个匹配 M',M'是 M 的一个子集,即 $M'\subseteq M$,匹配 M'的大小为 n 并且匹配中没有任意两个三元组的符号在同一坐标是相同的。

3-维匹配问题是一个经典的 NP 完全问题[3]。人们首先从近似算法角度研究该问题,文[4]给出了该问题的一个多项式时间的近似比为 $3/2+\varepsilon(\varepsilon>0)$的近似算法,文[5]证明了该问题是 APX 完全问题。近年来由于参数复杂性理论的发展,人们开始从参数复杂性理论角度来研究这个问题[6]。针对实际工程需要,将问题与一个很小的参数联系起来,使得许多理论上难解的计算问题在实际中有可能得到有效解决。下面给出了 3-维匹配参数问题的定义[6]。

定义 3(3-维匹配参数问题) 给定一个问题实例(S,k)。X,Y,Z 是互不相交的具有相同符号个数的集合,S 是 X,Y,Z 三个集合笛卡儿乘积后的一个子集,即 $S\subseteq X\times Y\times Z$。$S$ 中有 n 个三元组,k 是一个整数。我们要解决的问题是要么能找到 S 的一个 k 大小的匹配 M,匹配 M 中包含 k 个三元组并且这 k 个三元组中没有两个符号在任意一个坐标上是相同的,要么报告说没有一个这样的匹配存在。

文[6]中 Garey 和 Johnson 证明了 3-维匹配参数问题是一个固定参数可解问题,他们在基于散列函数的着色上给出了一个运行时间为 $O^*((3k)!(3k)^{9k+1})$(对于一个参数 k 的函数 $f(k)$,我们用 $O^*(f(k))$来表示界 $O(f(k)n^{o(1)})$)的算法。文[7]中 Chen 等用贪婪局部法将 3-维匹配问题的时间到 $O^*((5.7k)^k)$。文[8]中 Fellows 等首先表示了 3-维匹配问题有一个 $O(k^3)$核,同时提出了运行时间为 $O^*(12.67^{3k}T(k))$的算法,$T(k)$是动态规划算法的运行时间,也就是用 $13k$ 种颜色来为一个三元组集合着色,要找出一个有 k 个三元组的匹配,匹配中所有符号都要被着上不同的颜色(用目前已知的技术知 $T(k)$至少是 $O^*(10.4^{3k})$。文[9]中 Kneis 等基于分治法提出了 3-维匹配问题的一个运行时间是 $O^*(16^{3k})$的确定式算法。

本文主要研究了 3-维匹配参数问题的算法。我们通过对问题结构的进一步研究,将着色和动态规划结合起来并将其应用在 3-维匹配问题上,从而得到了一个更好的算法,算法运行时间是 $O^*(3.42^{3k})$。

本文后面部分的组织如下:第 2 部分提出了两个关于 3-维匹配参数问题的着色及动态规划结合的算法 CCDP 和 CDM。第 3 部分是算法时间复杂度分析。最后是结论。

2 3-维匹配参数问题的算法

由定义 3 可知 S 是 X、Y 和 Z 三个互不相交的具有相同

*)基金项目:国家自然科学基金重点项目:生物信息学中的相关组合理论和算法研究(60433020)。宁 丹 硕士研究生,主要研究领域为参数计算,计算机理论;王建新 博士,教授,博士生导师,主要研究领域为计算机算法、网络优化理论、生物信息学。

符号个数的集合的笛卡儿乘积后的一个子集，即 $S\subseteq X\times Y\times Z$。$S$ 中的每个元组 t 用 $t=(x, y, z)$ 表示，其中 $x\in X$，$y\in Y$，$z\in Z$。为了方便描述算法，本文将对应于 X、Y 和 Z 中的符号定义为来自第一列、第二列和第三列的符号。用 $Val(t)$ 表示整个集合 $\{x, y, z\}$ 的符号，$Val^1(t)$ 表示符号 x，即 $Val^1(t)=\{x\}$；$Val^2(t)$ 表示符号 y，即 $Val^2(t)=\{y\}$；$Val^3(t)$ 表示符号 z，即 $Val^3(t)=\{z\}$。用 p、q 表示下标集合 $\{1,2,3\}$ 中的任意两个下标。

在 3-维匹配参数问题算法的研究中，着色算法[10]和动态规划是解决该问题的主要技术。着色算法可以用来解决从元素全集中选子集的问题。着色算法是给定一个有 n 个元素的全集 U 和有 k 种颜色的全集 C，对全集 U 中的每个元素进行着色，每个元素对应一种颜色，颜色全集中每种颜色至少被使用过一次。在利用了文[11]提出的着色算法的基础上，本文深入分析了 3-维匹配问题本身的特点，提出了一种新的动态规划算法，获得了一个算法运行时间为 $O^*(3.42^{3k})$ 的确定式参数算法。

下面给出一个我们关于 3-维匹配参数问题算法的观察：

如果 S 存在一个包含 $i+1$ 个三元组的匹配 M_{i+1}，那么 S 中一定有一个包含 i 个三元组的匹配 M_i，使得 M_i 的每个三元组中至少包含 M_{i+1} 中的 2 个符号，也就是说 M_i 中至少有 $2i$ 个符号在 M_{i+1} 中。

基于以上观察，本文首先研究了如何从一个已知的包含 i 个三元组的匹配 M_i 求一个包含 $i+1$ 个三元组的匹配 M_{i+1}，使得 M_i 中至少有 $2i$ 个符号在 M_{i+1} 中。为了解决这个问题，本文提出了一个基于着色和动态规划的 CCDP 算法。CCDP 算法先用穷举法构造了一个有 3^i 个元素的 W 集合，W 中的每个元素 U_w 是 M_i 中 i 个不在 M_{i+1} 中的符号，既然知道 M_i 中 i 个不在 M_{i+1} 中的符号，也就确定 M_i 中哪 $2i$ 个在 M_{i+1} 中的符号。因为三元组中的符号在 3 列之间是不重复的，如果将相同的符号排在一块，这些有相同的符号的三元组的匹配中只能出现一个。所以 CCDP 算法将其中包含符号个数最少的一列按其符号进行排序，排序后相同的符号放在一块。CCDP 算法找出 M_i 在 M_{i+1} 的 $2i$ 个符号中在哪两列符号个数至少有 $\lceil 4i/3\rceil$ 个，这样保证 $2i$ 个符号在另一列出现的符号个数是最小的，也就使得在着色时 CCDP 算法会利用较多的颜色来找出不相同的符号，所以 CCDP 算法的动态规划时只要考虑这两列中不相同的符号对。由分析可知总存在一个下标对 (p,q) 使得 M_i 的 $2i$ 个符号在 p、q 两列的符号个数至少为 $\lceil 4i/3\rceil$，而这 $\lceil 4i/3\rceil$ 个符号也在 M_{i+1} 的 p、q 两列中。由于 M_{i+1} 在 p、q 两列上的符号个数为 $2(i+1)=2i+2$，因此 M_{i+1} 中的 p、q 两列最多有 $\lceil 2i/3+2\rceil$ 个符号是不在 M_i 的 p、q 两列上，即有 $\lceil 2i/3+2\rceil$ 个符号是在 M_{i+1} 的 p、q 两列而不在 M_i 的 p、q 两列。集合 W 中的每个元素 U_w 都能确定两个下标 p、q，对于每个 (p,q) CCDP 算法用一个 $\lceil 2i/3+2\rceil$ 种颜色的着色方案 f（f 包含了颜色种数是 $\lceil 2i/3+2\rceil$ 的多种着色方法。）对 $(Val^p(S)\cup Val^q(S))-(Val^p(M_i)\cup Val^q(M_i))$ 的所有符号着色，从而找出在 M_{i+1} 中而不在 M_i 中的 M_{i+1} 中的 $\lceil 2i/3+2\rceil$ 个符号，也就是在一个有 $(2n-2i)$ 个元素（每个元素是一个符号）的全集 U 中，用一个有 $\lceil 2i/3+2\rceil$ 种颜色的全集 g 为 U 中的每个元素进行着色，每个元素使用的颜色都不相同，从而找出 $\lceil 2i/3+2\rceil$ 个符号并且这些符号恰好用上了 g 的所有颜色。另一方面由于 CCDP 算法能在 p、q 两列上确定 M_i 的 $\lceil 4i/3\rceil$ 个符号，假设这 $\lceil 4i/3\rceil$ 个符号也对应 $\lceil 4i/3\rceil$ 种颜色，当然这些颜色是不同于全集 g 的 $\lceil 2i/3+2\rceil$ 种颜色，所以把所有的颜色加起来，总共需要 $\lceil 4i/3\rceil+\lceil 2i/3+2\rceil\leqslant 2i+4$ 种颜色。

在着色方案 f 的每一种着色方法下，CCDP 算法都要在至多 $2i+4$ 个颜色中选出所有可能的不同颜色的符号对应的符号对构成的集合，使得最终找出一个能构造出一个 $k+1$ 大小的匹配 M_{i+1} 的 $(i+1)$ 个符号对。对于每一个 (p,q)，只要 CCDP 算法能找到一个 $k+1$ 大小的匹配 M_{i+1}，就返回 M_{i+1} 算法结束。我们令集合 Q_{old}，Q_{new} 的每个元素 C 对应为三元组在 p、q 两列上不同颜色的符号构成的符号对的一个集合，当三元组在 r 列上的符号改变时，集合 Q_{old}，Q_{new} 的元素包含的符号对的个数会相应增加。CCDP 算法如图 1 所示。

算法 CCDP(S, M_i, f)
输入：一个三元组的集合 S，一个整数 i，一个 $\lceil 2i/3+2\rceil$ 种颜色的着色方案 f。
输出：如果找到 S 的一个匹配 M_{i+1}，则输出 M_{i+1}。否则输出“S 不存在一个 i+1 大小的匹配”。
1. 用穷举法找出一个有 3^i 个元素的集合 W，满足 W 的每个元素是一个有 i 个符号的集合 U_w。对于每个 U_w，M_i 的每个三元组中都只有一个符号在集合 U_w 中；
2. for W 中的每个 U_w do {
遍历 M_i 的所有三元组找出不属于集合 U_w 的符号的个数至少为 $\lceil 4i/3\rceil$ 的两列，将这两列的下标值赋给(p、q)。
for 着色方案 f 对 $(Val^p(S)\cup Val^q(S))-(Val^p(M_i)\cup Val^r(M_i))$ 的所有符号的每种着色 do {
令 $r=\{1, 2, 3\}-\{p、q\}$　　//r 列是三列中除去 p、q 两列所剩下的那一列；
设在 $Val^r(S)$ 中的所有符号为 x_1，x_2，…x_m
初始化为 $Q_{old}=\{\phi\}$，$Q_{new}=\{\phi\}$，$C=\{\phi\}$；
for j=1 to m do {
for Q_{old} 中每个元素 C do
for S 中的每个三元组 t，并且 $Val^r(t)=x_j$ do
if 在 C 中符号的颜色与 $Val^p(t)\cup Val^q(t)$ 中任一个符号的颜色不相同则将 C 并到集合 $\{(Val^p(t), Val^q(t))\}$ 中，令 $C'=C\cup\{(Val^p(t), Val^q(t))\}$。如果 C' 中没有超过 i 个符号对并且 Q_{new} 元素中的符号使用的颜色与 C' 中的符号使用的颜色不相同则将 C' 加到 Q_{new} 集合中；
$Q_{old}=Q_{new}$；
}
如果 Q_{old} 存在一个包含 i+1 个符号对的元素，用 C_{i+1} 表示这 i+1 个符号对，利用 C_{i+1} 构造一个二分图 B_{i+1}，求该 B_{i+1} 的一个 i+1 条边的图的匹配。如果能找到该匹配，也就存在 S 的一个匹配 M_{i+1}，则返回 M_{i+1} 算法结束。
}
}
3. 如果没找到一个 M_{i+1}，则输出“S 不存在一个 i+1 大小的匹配”，算法结束。

图 1　CCDP 算法

用 C_{i+1} 表示为 Q_{old} 中的 $i+1$ 个符号对的集合。最后构造一个二分图 $B_{i+1}=(V_L\cup V_R, E)$，V_L 包含 $i+1$ 个点，对应 C_{i+1} 中的 $i+1$ 个符号对，V_R 中的点是 $Val^r(S)$ 中的所有符号。如果在 S 中有一个三元组 (x,y,z) 则 V_L 中的一个点 $(y,$

z)和 V_R 中的一个点 x 之间存在一条边。求该二分图的一个 $i+1$ 条边的图的匹配,从而找出 S 的一个匹配 M_{i+1}。

CCDP算法能由 S 的一个已知的匹配 M_i 来找出一个 M_{i+1},在所有 S 中的三元组中每次选择其中一个三元组 t,把 t 看成是一个只有一个元组的匹配 M_1,然后在 M_1 上递归调用CCDP算法,由 M_1 不断扩充直到找到 S 的一个($k+1$)大小的匹配 M_k,算法结束。否则报告说没有找到一个这样的匹配,算法结束。具体算法 CDM 如图 2 所示。

```
算法 CDM(S, k)
输入:一个大小为 n 的实例 S,一个整数 k。
输出:如果 S 存在一个 k 大小的匹配 M,则将输出 Mk。否则报告说没有找到一个这样的匹配。
1. 令 S 中的所有三元组表示为{t1, t2...tn};
2. for j=1 to n do
     M1= tj;
     for i=1 to k-1 do
       Mi+1=CCDP(S, Mi,f);
       if CCDP(S, Mi,f)返回了一个 Mi+1, 算法继续;
       else if CCDP(S, Mi,f)返回"S 不存在一个 i+1 大小的匹配", 算法转到 2;
     如果存在一个 Mk, 则输出 Mk, 算法结束;
3. 如果没有找到一个 Mk,则输出"S 不存在一个 k 大小的匹配",算法结束。
```

图 2 CDM 算法

3 算法时间复杂度分析

假设算法 CCDP 的时间复杂度是 $Tp(k)$,则 CDM 算法的时间复杂度为 $n \cdot k \cdot Tp(k)$,$n \cdot k$ 是算法时间复杂度的多项式时间部分,与 $Tp(k)$这部分指数时间比较时可以忽略,所以下面我们重点分析算法 CCDP 的时间复杂度。

集合 W 的大小为 3^i,所以 CCDP 算法最外层的循环为 3^i 次。对于每个 p、q,M_i 中 p、q 两列的符号个数至少有$\lceil 4i/3 \rceil$个,而这$\lceil 4i/3 \rceil$个符号也在 M_{i+1} 的 p、q 两列中。由于 M_{i+1} 在 p、q 两列上的符号个数为 $2(i+1)=2i+2$,因此 M_{i+1} 中的 p、q 两列最多有$\lceil 2i/3+2 \rceil$个符号是不在 M_i 的 p、q 两列上。令 f 是一个$\lceil 2i/3+2 \rceil$种颜色的着色方案,CCDP 算法用 f 对$(Val^p(S) \cup Val^q(S)) - (Val^p(M_i) \cup Val^q(M_i))$的符号进行着色,利用文[11]的着色结果 CCDP 算法的第二个 for 循环语句循环的次数为该方案着色的数目,也就是着色方案的规模为 $O^*(6.1^{2i/3+3})=O^*(6.1^{2i/3})$,并在时间 $O^*(6.1^{2i/3})$内完成。算法第 2 步中的 3 个 for 循环是要找出一个($i+1$)个符号对的集合,由这($i+1$)个符号对可以构造一个 M_{i+1}。由于已经确定 M_i 的$\lceil 4i/3 \rceil$个符号,假设这$\lceil 4i/3 \rceil$个符号也对应着$\lceil 4i/3 \rceil$种颜色,当然这些颜色是不同于全集 g 的$\lceil 2i/3+2 \rceil$种颜色,因此把所有的颜色加起来,总共需要$\lceil 4i/3 \rceil + \lceil 2i/3+2 \rceil \leqslant 2i+4$ 种颜色。在着色方案 f 的每一次$\lceil 2i/3+2 \rceil$种颜色的着色方法下,CCDP 算法都要在至多 $2i+4$ 个颜色中选出所有可能的不同颜色的符号对应的符号对构成的集合,使得最终找出一个能构造出一个 M_{i+1} 的($i+1$)个符号对。Q_{dd} 集合的每个元素 C 对应为三元组在 p、q 两列上不同颜色的符号构成的符号对的一个集合,因为算法最后返回的是一个 M_{i+1},所以 CCDP 算法需要的是一个有($i+1$)个符号对的元素,当三元组在 r 列上的符号改变时,Q_{dd} 集合的元素 C 包含的符号对的个数会相应增加,所以 Q_{dd} 最多包含 $\sum_{j=0}^{i+1}\binom{2i+4}{2j}$ 个元素。如果 Q_{dd} 中恰好有一个包含($i+1$)个符号对的元素,用这($i+1$)个符号对 CCDP 算法可以找到 S 的一个匹配 M_{i+1}。因此每次执行算法第 2 步中的 3 个 for 执行循环需要时间 $O^*\left(\sum_{j=0}^{i+1}\binom{2i+4}{2j}\right)=O^*(2^{2i})$。CCDP 算法用这($i+1$)个符号对来构造一个二分图,并利用二分图的匹配从而找出 S 的一个 $i+1$ 大小的匹配,二分图的匹配可以在多项式时间内完成。算法 CCDP(S, M_i, f)的时间复杂度是 $Tp(k)=3^i \cdot O^*(6.1^{2i/3}) \cdot O^*(2^{2i})=O^*(3.42^{3i})$,这是由一个已知的 M_i 生成一个 M_{i+1}所需的时间。算法 CDM 实现的是对于一个实例(S, k)找到 S 的一个 k 大小的匹配 M_k。由算法 CDM 递归调用算法 CCDP,第一次调用算法 CCDP 由 M_1 找到一个 M_2,再调用一次算法 CCDP 由 M_2 找到一个 M_3,以次类推最后由 M_{k-1} 找到一个 M_k,故整个过程需要的时间是 $O^*(3.42^3)+O^*(3.42^6)+\cdots+O^*(3.42^{3(k-1)})=O^*(3.42^{3k})$。由以上分析可知本文提出的 3-维匹配参数问题算法 CDM 的时间复杂度是 $O^*(3.42^{3k})$。

结论 本文在利用文[11]提出的着色算法的基础上,深入分析了 3-维匹配问题本身的特点,提出了一种新的动态规划算法,获得了一个算法运行时间为 $O^*(3.42^{3k})$的确定式参数算法。很明显我们的结果比 Kneis 等的结果快很多,即使在 $k=4$ 的时候,我们的算法也将比 Kneis 等的算法提高了上亿倍以上。由此看出,算法的时间主要集中在着色和动态规划两部分的时间,如果所需颜色种数越少,则着色部分的时间也越少。同样,如果不同颜色集合个数越少,则动态规划部分的时间也越少。所以我们将未来的研究工作重点放在着色和动态规划上的改进,同时也要努力发现新的技术来解决 3-维匹配参数问题。

参 考 文 献

1 Hell P, Kirkpatrick D. On the complexity of a generalized matching problem. In: Proc. Tenth ACM Symp. on Theory of Computing, 1978. 309～318
2 Karp R M. On the Complexity of Combinatorial Problems. Networks, 1975,75: 45～68
3 Garey M R, Johnson D S. Computers and Intractability: A Guide to the Theory of NP-Completeness, Freeman, San Francisco, CA, 1979
4 Hurkens C A J, Schrijver A. On the size of systems of sets every t of which have an SDR, with an application to the worst-case ratio of heuristics for packing problems. SIAM J. Disc. Math, 1989(2):68～72
5 Kann V. Maximum bounded 3-dimensional matching is MAX SNP-complete, Inform. Process. Lett,1991,37: 27～35
6 Downey R G, Fellows M R. Parameterized Complexity. Springer, New York, 1999
7 Chen J, Friesen D K, Jia W, et al. Using nondeterminism to design deterministic algorithms. Algorithmica, 2004,40: 83～97
8 Fellows M R, Knauer C, Nishimura N, et al. Faster fxed-parameter tractable algorithms for matching and packing problems. Lecture Notes in Computer Science 3221, (ESA 2004), 2004. 311～322
9 Kneis J, Molle D, Richter S, et al. Divide-and-color. Manuscript, 2006
10 Alon N, Yuster R, Zwick U. Color-coding. Electro nic Colloquium on Computational Complexity (ECCC), 1(009), J. ACM, 1995, 42(4):844～856
11 Chen J, Lu S, Sze S-H, et al. Improved algorithms for path, matching, and packing problems, Manuscript, 2006
12 Prieto E, Sloper C. Either/Or: Using Vertex Cover Structure in designing FPT-algorithms - the case of k-Internal Spanning Tree. In: Proceedings of WADS. Workshop on Algorithms and Data Structures, Ottawa, Canada, LNCS 2748, 2003. 474～483
13 Fellows M, Heggernes P, Rosamond F, et al. Exact algorithms for finding k disjoint triangles in an arbitrary graph. In: Proc. 30th Workshop on Graph Theoretic Concepts in Computer Science, Lecture Notes in Computer Science, Vol. 3353, Springer, Berlin, 2004. 235～244