一种基于混合策略的彩色编码算法

王建新 杨志彪 刘云龙 陈建二

(中南大学信息科学与工程学院 长沙 410083)

摘 要 彩色编码是求解实际工程中难解问题的一种新兴而重要的技术.在应用该技术时,算法复杂度取决于彩色编码着色方案的规模,因此规模的大小将成为衡量彩色编码算法优劣的标准.彩色编码的研究在最近几年得到了许多有重要意义的结果.基于完全散列函数的 PH 算法产生的着色方案规模为 $O^*(6.1^k n)$,是目前世界上最好的确定彩色编码结果;彩色编码算法 PBCC 是一种利用组合思想针对 $n \leq 2k$ 的有效着色算法.文中以分治算法为基础,结合核心化技术,并利用 PBCC 算法求解子问题,提出了一种基于混合策略的彩色编码算法 HABCC,并且证明了由 HABCC 算法产生的着色方案确实可以覆盖到所有子集,着色方案规模为 $|S(n,k)| \leq 2^k \cdot \lceil \log k \rceil^{k-1} \cdot n$.通过与 PH 算法的比较,说明了 HABCC 算法具有更小的着色方案规模,对彩色编码技术的实际应用具有重要的意义.

关键词 彩色编码;分治;核心化

中图法分类号 TP301 **DOI 号**: 10.3724/SP.J.1016.2010.01024

An Improved Color Coding Algorithm Based on Hybrid Architecture

WANG Jian-Xin YANG Zhi-Biao LIU Yun-Long CHEN Jian-Er

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

Abstract Color coding is a new and important technique to solve engineering hard problems. The complexity of algorithm using color coding depends on the scale of the coloring schemes, so the scale size becomes a standard to measure the color coding algorithm. Recently, the researches of color coding received many important improvements. The PH (Perfect Hashing) algorithm based on perfect hash functions constructs a scheme of size $O^*(6.1^k n)$, which has been the best deterministic result for color coding so far. The PBCC (Partition-Based Color-Coding) algorithm is an effective color coding algorithm using combination thought while $n \leq 2k$. Basing on divide-and-conquer algorithm, combining with kernelization technique and using PBCC algorithm to solve sub-problem, this paper proposes a hybrid architecture based coloring algorithm HABCC (Hybrid Architecture Based Color-Coding), and proves that the coloring scheme generated by HABCC can cover all the subsets, moreover, the scheme scale satisfying $|S(n,k)| \leq 2^k \cdot \lceil \log k \rceil^{k-1} \cdot n$. Via comparing with PH algorithm, HABCC algorithm constructs a smaller coloring scheme, which is significant to practical application of color coding technology.

Keywords color coding; divide-and-conquer; kernelization

收稿日期:2008-07-17;最终修改稿收到日期:2009-12-26.本课题得到国家自然科学基金(60773111)、国家"九七三"重点基础研究发展规划项目前期研究专项(2008CB317107)、高等学校博士学科点专项科研基金项目(20090162110056)和国家教育部创新团队资助计划(IRT0661)资助.**王建新**,男,1969 年生,博士,教授,主要研究领域为计算机算法、网络优化理论、生物信息学. E-mail: jxwang@mail.csu.edu.cn. **杨志彪**,男,1983 年生,硕士,主要研究方向为参数计算、计算机理论. **刘云龙**,男,1983 年生,硕士,主要研究方向为参数计算、计算机理论. **陈建二**,男,1954 年生,博士,教授,主要研究领域为生物信息学、计算机理论、计算复杂性及优化.

1 引　言

随着信息产业和现代工业的快速发展，各领域中的信息海量增长，许多组合问题利用目前人类具有的计算资源无法在有效的时间内解决. 在计算机复杂性理论中，这些计算难解问题大多属于 NP 难解问题. 然而，大量的这些 NP 难解问题又是工程应用中不可避免而且必须解决的问题.

子集选择(subset selection)是组合学研究领域中的一类重要的问题，通常被描述成下面的形式：给定一个包含 n 个元素的全集 U，找到满足某一特定条件 R，包含 k 个元素的子集 $W\subseteq U$. 所有此类问题都可以用穷举的方法在指数时间内得到可行解，即枚举所有的组合情况构成的集合，该穷举算法的时间复杂度为 $O(\binom{n}{k})$.

k-Path 问题是一个典型的子集选择问题，给定一个图 G 和一个整数 k，要求在图 G 中找到一个包含 k 个节点的简单路径. k-Path 问题的难点在于对路径简单性的要求(即路径节点不重复)，最早提出的算法时间复杂度为 $O(2^k k!n^{O(1)})$[2,8]. Alon 等人通过对 k-Path 问题的深入研究，阐明了解决该问题的本质是在图 G 所有的 n 个节点中选择 k 个节点构成的子集. 基于这一观察，于 1995 年首次提出了彩色编码技术，得到了一个时间复杂度为 $O^*(5.44^k n^{O(1)})$ 的随机算法[1]，极大地降低了求解问题的时间复杂度.

彩色编码的基本思想可以描述为：给定元素集合 U 和颜色集合 C，其中 $|U|=n$，$|C|=k$，将 U 中的每个元素都使用 C 中的一种颜色进行着色. 假定原问题的目标解包含的 k 个元素正好被着为 k 种不同的颜色，原问题将获得额外的着色约束条件而简化. 为了便于对彩色编码的理解，沿用文献[4]中的定义如下.

定义 1((n,k)着色). 给定元素全集 $U=\{e_1,e_2,\cdots,e_n\}$ 和颜色全集 $C=\{c_1,c_2,\cdots,c_k\}$，将集合 U 中的所有元素使用 C 中的任意颜色进行着色 $h_i=f(e_i)$(其中 $f(e)$ 表示对元素 e 赋予一种颜色)，得到一个 n 元组 $H=\langle h_1,h_2,\cdots,h_n\rangle$，且 $\forall i,h_i\in C$，$\bigcup_{i=1}^{n}\{h_i\}=C$，即 U 中每个元素均可对应一种颜色，且颜色全集中每种颜色至少被使用过一次，则称该 n 元组为一个(n,k)着色.

定义 2(覆盖). 给定一个(n,k)着色 $H=\langle h_1,h_2,\cdots,h_n\rangle$ 和一个子集 W，其中 W 是从 $U=\{e_1,e_2,\cdots,e_n\}$ 中任意选取 k 个元素所组成的子集，对任意 i 和 j，若 $e_i,e_j\in W$ 且 $i\neq j$，则 $h_i\neq h_j$，则称着色 H 覆盖子集 W.

定义 3((n,k)着色方案). 若(n,k)着色的集合满足：任取 U 的一个 k 元素子集 W，至少存在一个可以覆盖 W 的(n,k)着色，则称该集合为(n,k)着色方案，记作 $S(n,k)$. 方案包含着色的数目称为着色方案的规模，记作$|S(n,k)|$.

值得注意的是，彩色编码技术本身并不直接解决目标问题，而是通过额外的限制条件有效地降低了目标问题的搜索空间. 在求解某些问题时，需要枚举所有的元素组合情况，而利用彩色编码技术后，一种着色可以覆盖大量不同的组合情况，从而有效地减少了搜索空间. 彩色编码已成功应用于求解 k-Path[1,3]、k-Cycle[1]、Matching[13]、Packing[13] 等 NP 难问题，并在生物蛋白质网络中的路径查找[9]和 Motif 查找[10-11,14]等问题上有着广泛的应用.

基于彩色编码的算法时间复杂度依赖于着色方案的规模，根据算法所用的着色方式不同，彩色编码可以分为随机式着色和确定式着色.

Alon 等人提出彩色编码思想时，详细介绍了随机式着色算法. 使用 k 种不同的颜色对图中的所有节点进行着色，在着色图中利用动态规划寻找正确着色(k 个节点颜色各不相同)的问题实例. 通过分析问题实例正确着色的概率，重复着色-求解过程足够次数，将有较大概率求得正确解，算法的时间复杂度为 $O^*(5.44^k n^{O(1)})$. Hüffner 等人[13]利用 $1.3k$ 种颜色着色，将随机算法复杂度改进到 $O^*(4.32^k)$. 最近，Chen 等人利用分治思想，组合小规模的解来构成较大规模的解，提出了复杂度为 $O^*(4^k)$ 的随机算法[3]. 同样基于分治思想，Kneis 等人也提出了划分着色的随机算法[5]，复杂度为 $O^*(4^k)$. 虽然随机着色算法可以得到一定精度的结果，但在对结果精度有很高要求的情况下，这样的概率解是不能令人满意的.

为了构造确定式着色，Alon 等人在文献[1]中指出可以利用完全散列函数族将随机算法确定化. 根据文献[6-7]中的理论，(n,k)完全散列函数族 F 可以在 $O(2^{O(k)}n\log n)$ 的时间复杂度内构造出来，且函数族 F 的大小为 $O(2^{O(k)}\log n)$. 通过枚举 F 中的所有函数，得到大小为 $O(2^{O(k)}\log n)$ 的确定式着色方案. 经过分析，表示任一散列函数需要多于 12kbits，枚举所有的散列函数至少需要 $2^{12k}>4000^k$，因此，

这种着色方案的构造算法即使在 k 较小时也不太实用.

最近,Chen 等人大幅改进了确定式着色方案的研究结果,提出了复杂度为 $O^*(6.1^k n)$ 的算法.在具体的构造过程中,采用了 4 层散列处理来避免冲突,即 $Z_n \rightarrow Z_k^2 \rightarrow Z_{k/4} \rightarrow Z_{c_j(c_j-1)}$. 该算法(记为 PH(Perfect Hashing))首先使用核心化方法将 (n,k) 着色问题在多项式时间内规约到 (k^2,k) 着色问题,然后将 k^2 元素散列到 $k/4$ 个位置上,再利用另一组散列函数分别将每个位置的元素重新散列到 $c_j(c_j-1)$ 个位置上来解决冲突,最后使用已有的 $(c_j(c_j-1),c_j)$ 着色方案构造总体着色方案.进一步,通过对 c_j 大小的分类讨论,确定算法的复杂度为 $O^*(6.1^k n)$. 虽然算法得到了很好的理论复杂度,但在实际应用中的效果不尽理想.

直接利用组合思想也可以得到确定式着色方案.最近,从组合算法角度对着色方案构造算法的研究得到了一个新的结果.文献[4]对彩色编码在 $n \leq 2k$(非小参数)情况下的应用进行了深入研究,提出了一种基于划分的着色算法 PBCC(Partition-Based Color-Coding).在实际情况下的应用表明,PBCC 比 PH 算法有更高的实用价值.但是 $n \leq 2k$ 约束条件的限制,PBCC 算法在很多情况下无法应用.

本文从简单的分治策略出发,结合核心化技术,并充分利用现有的 PBCC 算法,提出了一种基于混合策略的着色算法(Hybrid-Architecture-Based Color-Coding,HABCC).通过理论分析以及实际着色方案规模的比较,HABCC 算法比现有算法具有更小规模的着色方案,并且易于构造,从而具有更好的实用性能,对彩色编码技术的应用具有重要的意义.

2 基于混合策略的着色算法

基于混合策略的着色算法将从简单的分治策略出发,结合核心化技术,并利用现有的 PBCC 算法处理子问题,下面几节将分别介绍以上算法技术.

2.1 分治着色算法

为了便于接下来的讨论,定义着色以及着色方案的变换操作如下.

定义 4(着色的连接). 给定 (n_1,k_1) 着色 $H_1=\langle h_1,h_2,\cdots,h_{n1}\rangle$ 及 (n_2,k_2) 着色 $H_2=\langle h'_1,h'_2,\cdots,h'_{n_2}\rangle$,$H_1$ 和 H_2 的连接定义为 $\langle h_1,h_2,\cdots,h_{n_1},h'_1,h'_2,\cdots,h'_{n_2}\rangle$,记为 $H_1 \oplus H_2$.

定义 5(着色方案的和连接). 给定着色方案 $S(n_1,k_1)$ 及 $S(n_2,k_2)$,不妨设 $|S(n_1,k_1)| \geq |S(n_2,k_2)|$,着色方案的和连接定义为 $S(n_1,k_1) \oplus S(n_2,k_2)=\{H_i \oplus f(S(n_2,k_2)) \mid \forall H_i \in S(n_1,k_1)\}$,其中函数 $f(S(n_2,k_2))$ 返回着色方案 $S(n_2,k_2)$ 中的任意一个着色.因此,着色方案规模 $|S(n_1,k_1) \oplus S(n_2,k_2)|=\max(|S(n_1,k_1)|,|S(n_2,k_2)|)$.

定义 6(着色方案的积连接). 给定着色方案 $S(n_1,k_1)$ 及 $S(n_2,k_2)$,着色方案的积连接定义为 $S(n_1,k_1) \otimes S(n_2,k_2)=\{H_i \oplus H_j \mid \forall H_i \in S(n_1,k_1), \forall H_j \in S(n_2,k_2)\}$. 即 $S(n_1,k_1)$ 中的每个着色都将和 $S(n_2,k_2)$ 中的每个着色进行连接.因此,着色方案规模 $|S(n_1,k_1) \otimes S(n_2,k_2)|=|S(n_1,k_1)| \times |S(n_2,k_2)|$.

分治着色算法的基本思想是将元素分成两部分,分别进行着色.实际着色时,首先将元素全集 U 分成两个部分,记为 U_1,U_2,其中 $U_1 \cap U_2=\varnothing$,$U_1 \cup U_2=U$,且 $|U_1|=n_1$,$|U_2|=n_2$. 为使构造出的着色方案确定式地覆盖所有选择情况,必须枚举颜色数在 U_1 和 U_2 中的所有分布情况.U 中任一 k 元素的子集 W,满足 $|U_1 \cap W|=k_1$,$|U_2 \cap W|=k_2$,显然 $k_1+k_2=k$. 如果存在集合 $U_1 \cap W$ 的 k_1 着色方案 $S(n_1,k_1)$ 和 $U_2 \cap W$ 的 k_2 着色方案 $S(n_2,k_2)$,那么 $S(n_1,k_1) \otimes S(n_2,k_2)$ 将覆盖 U 的所有 k 元素子集.此外,当 $k_1 \times k_2=0$ 时,有两种情况:$k_1=0$,此时 $S(n_1,k)$ 可覆盖子集 W;或 $k_2=0$,此时 $S(n_2,k)$ 可覆盖子集 W. 故 $S(n_1,k) \oplus S(n_2,k)$ 即可覆盖 $W \subseteq U_1$ 或 $W \subseteq U_2$ 的所有情况.因此,取 $n_1=\lceil n/2 \rceil$,$n_2=\lfloor n/2 \rfloor$,分治着色算法构造着色方案的公式如下:

$$S(n,k)=S(\lceil n/2 \rceil,k) \oplus S(\lfloor n/2 \rfloor,k) \cup \bigcup_{k_1+k_2=k}^{k_1,k_2 \geq 1} S(\lceil n/2 \rceil,k_1) \otimes S(\lfloor n/2 \rfloor,k_2) \quad (1)$$

显然,式(1)是一个递推式,在假设可以获得较小规模问题解的前提下,组合小规模问题的解可以获得大规模问题的解.公式中的第一部分是目标解全部存在于同一个部分中的特殊情况,此时,只需要将目标解分别在两部分中的着色方案进行和连接即可,连接后的着色方案规模等于较大的子着色方案规模.公式中的第二部分则将每种分布的两个部分着色方案进行积连接,加入总的着色方案.

递归需要终止条件,根据式(1),给出 3 个简单情况下的处理,其中 n 表示元素个数,k 表示颜色数目.

(1) $n<k$ 时,(n,k) 着色不存在,直接返回空集;

(2) 当 $n=k$ 时，每个元素对应一种颜色，返回着色方案$\{\langle 1,2,\cdots,n\rangle\}$即可；

(3) 当 $k=1$ 时，所有元素着同一种颜色，返回着色方案$\{\langle 1,1,\cdots,1\rangle\}$.

每次调用递归算法，元素个数 n 都将减半，在经过$\lceil\log n\rceil$层递归调用后，此时 n 为 1，直接返回着色方案. 这样，可以在不产生着色方案的情况下，分析分治着色算法产生的着色方案规模.

便于后面的分析，首先给出以下引理.

引理 1.　对任意的正整数 $n\geq 2$，$\lceil\log\lceil n/2\rceil\rceil=\lceil\log n\rceil-1$.

证明.

(1) 当 $n\geq 2$ 且 n 为偶数时，$\lceil\log\lceil n/2\rceil\rceil=\lceil\log n-1\rceil=\lceil\log n\rceil-1$，显然成立.

(2) 当 $n\geq 2$ 且 n 为奇数时，$\lceil\log\lceil n/2\rceil\rceil=\lceil\log(n+1)-1\rceil=\lceil\log(n+1)\rceil-1$. 假设$\lceil\log(n+1)\rceil-1=\lceil\log n\rceil-1$不成立，即$\lceil\log(n+1)\rceil\neq\lceil\log n\rceil$，则此时必然有 $n=2^k$，其中 $k=1,2,3,\cdots$，这与 n 为奇数矛盾. 因此，n 为奇数时，$\lceil\log(n+1)\rceil-1=\lceil\log n\rceil-1$ 也成立.

因此，对任意的正整数 $n\geq 2$，$\lceil\log\lceil n/2\rceil\rceil=\lceil\log n\rceil-1$ 成立.　　证毕.

定理 1.　对任意的正整数 n 和 k 且 $n\geq k$，分治着色算法产生的着色方案满足 $|S(n,k)|\leq\lceil\log n\rceil^{k-1}$.

证明.

(1) 当 $k=1$ 时，只需将 n 个元素着同一种颜色，返回该$(n,1)$着色构成的集合. 此时，$|S(n,1)|\leq\lceil\log n\rceil^0=1$，显然成立.

(2) 当 $k=2$ 时，由式(1)有

$$\begin{aligned}|S(n,2)|&=\max(|S(\lfloor n/2\rfloor,2)|,|S(\lceil n/2\rceil,2)|+|S(\lceil n/2\rceil,1)|\times|S(\lfloor n/2\rfloor,1)|)\\&\leq\max(|S(\lfloor n/2\rfloor,2)|,|S(\lceil n/2\rceil,2)|)+1.\end{aligned}$$

假设 $|S(n,2)|\leq\lceil\log n\rceil$ 成立，只需证明 $\max(|S(\lfloor n/2\rfloor,2)|,|S(\lceil n/2\rceil,2)|)+1\leq\lceil\log n\rceil$ 也成立. 根据引理 1 有

$$\max(\lceil\log(\lfloor n/2\rfloor)\rceil,\lceil\log(\lceil n/2\rceil)\rceil)+1=\lceil\log(\lceil n/2\rceil)\rceil+1=(\lceil\log n\rceil-1)+1=\lceil\log n\rceil.$$

由上可知，$|S(n,2)|\leq\lceil\log n\rceil$ 成立.

(3) 假设 $k\leq i-1$ 时，$|S(n,k)|\leq\lceil\log n\rceil^{k-1}$ 均成立，那么，当 $k=i$ 时，

$$|S(n,i)|=\max(|S(\lfloor n/2\rfloor,i)|,|S(\lceil n/2\rceil,i)|)+\sum_{\substack{k_1+k_2=i\\k_1,k_2\geq 1}}|S(\lceil n/2\rceil,k_1)|\times|S(\lfloor n/2\rfloor,k_2)|$$

$$\begin{aligned}&\leq\max(|S(\lfloor n/2\rfloor,i)|,|S(\lceil n/2\rceil,i)|)+(i-1)\lceil\log(\lceil n/2\rceil)\rceil^{i-2}\\&\leq\lceil\log(\lceil n/2\rceil)\rceil^{i-1}+(i-1)\lceil\log(\lceil n/2\rceil)\rceil^{i-2}\\&=(\lceil\log n\rceil-1)^{i-1}+(i-1)(\log n-1)^{i-2}\\&\leq((\lceil\log n\rceil-1)+1)^{i-1}\\&=\lceil\log n\rceil^{i-1}.\end{aligned}$$

因此，$|S(n,i)|\leq\lceil\log n\rceil^{i-1}$ 成立.

(4) 综上所述，对任意的正整数 n 和 k 且 $n\geq k$，分治着色算法产生的着色方案满足 $|S(n,k)|\leq\lceil\log n\rceil^{k-1}$.　　证毕.

与 Chen 等人提出的复杂度为 $O^*(6.1^k n)$ 的着色方案构造算法相比，不需要 n 远大于 k 的限制，即 k 可以取不大于 n 的任意正整数. 而且上述算法构造简单，易于实现，同时复杂度中没有忽略与 n 及 k 相关的多项式系数. 当然随着问题规模 n 的增长，算法复杂度增长很快. 下面一节介绍核心化技术，以削弱问题规模 n 对算法复杂度的影响.

2.2　核心化

简单分治算法对问题规模 n 的增长过于敏感，从而局限了算法的应用范围. 尽管目前基于完全散列函数的着色算法都不尽实用，它们的一个理论基础：小参数理论在降低参数 n 对复杂度的影响上仍然有很大的作用. 应用完全散列函数可以得到的着色方案将满足 $|S(n,k)|\leq 2n|S(k^2,k)|$. 这个结果表明，应用一次完全散列函数就可以从很大程度上降低元素全集规模 n 增长所带来的不利影响，算法可以专注于解决(k^2,k)的着色问题.

引理 2[3].　如果(k^2,k)着色问题可以得到一个规模为 r 的着色方案，那么(n,k)着色问题可以得到一个规模不大于 $2nr$ 的着色方案.

文献[3]通过应用 Fredman 等人对完全散列函数的研究结果[6]证明了引理 2 的正确性，同时给出了着色方案的构造方法.

设 p 是满足 $n\leq p<2n$ 的素数，在集合 $Z_p=\{a\mid 0\leq a<p,a\in N\}$ 上定义函数族

$$\varphi_a(x)=(ax \bmod p)\bmod k^2.$$

假设 $S(k^2,k)=\{H_1,H_2,\cdots,H_r\}$，那么

$$S(n,k)=\{H_i\cdot\varphi_a\mid 1\leq i\leq r,\ a\in Z_p\},$$

其中 $H_i\cdot\varphi_a$ 是定义为 $H_i\cdot\varphi_a(x)=H_i(\varphi_a(x))$ 的函数.

由引理 2 可知，对原问题应用一次核心化预处理，可以大幅降低问题规模 n 对算法复杂度的影响. 应用核心化技术后，算法首先比较 n 与 k^2 的大小. 若 $n>k^2$，则由分治着色算法首先构造(k^2,k)着色

方案，再利用散列函数，映射为(n,k)着色方案；若$n\leq k^2$，则直接利用分治着色算法构造(n,k)着色方案. 结合引理 2 与定理 1，得到以下结论.

定理 2. 对任意的正整数 n 和 k 且 $n\geq k$，核心化分治着色算法产生的着色方案满足 $|S(n,k)|\leq 2^k\cdot\lceil\log k\rceil^{k-1}\cdot n$.

证明. 对任意的正整数 n 和 k，$|S(n,k)|$表示核心化分治着色算法产生的着色方案规模.

(1) 当 $k\leq n\leq k^2$ 时，由定理 1 有，$|S(n,k)|\leq\lceil\log n\rceil^{k-1}\leq 2^{k-1}\cdot\lceil\log k\rceil^{k-1}$.

显然，$|S(n,k)|\leq 2^k\cdot\lceil\log k\rceil^{k-1}\cdot n$ 成立.

(2) 当$n>k^2$时，结合定理 1 及引理 2，$|S(n,k)|\leq|S(k^2,k)|\cdot 2n\leq 2^{k-1}\cdot\lceil\log k\rceil^{k-1}\cdot 2n$.

因此，对任意的正整数 n 和 k 且$n\geq k$，$|S(n,k)|\leq 2^k\cdot\lceil\log k\rceil^{k-1}\cdot n$ 均成立. 证毕.

与 PH 算法相比，核心化分治着色算法产生的着色方案没有 $O^*(c^k)$形式的规模上界，但是 PH 算法产生的实际着色方案的规模为 $O^*(6.1^k 4^{\log k-1}k^{4\log k-1}n)$，$O^*(6.1^k n)$并不能很好地反映 k 较小时的实际规模. 而上界 $2^k\cdot\lceil\log k\rceil^{k-1}\cdot n$ 不仅没有忽略多项式的系数，甚至没有忽略任何的常系数，因而是相当准确的.

2.3 HABCC 算法

对所有的分治算法而言，加快分治的终止速度是改善分治算法性能的关键. 简单的分治策略仅仅包含 $k=1$，$n=k$ 等几个简单的终止条件，严重影响了分治算法的性能. 然而，基于划分的着色算法 PBCC[4] 的出现从很大程度上改善了这一情况. PBCC 算法通过枚举着色块的分布情况，得到了 $n\leq 2k$ 情况下的优化着色方案，具体算法参见文献[4]. 通过对实际着色规模的分析，文献[4]还指出，在目前计算能力可及并满足 $n\leq 2k$ 的情况下，PBCC 算法可以得到比 PH 算法更小规模的着色方案.

下面，我们在分治策略的基础上，结合核心化预处理，并利用 PBCC 算法处理 $n\leq 2k$ 情况的子问题，提出一种基于混合策略的着色算法(HABCC). 由于分治着色算法元素 n 递归下降的性质，任何(n,k)的着色问题都可以被递归到以下 4 种情况之一：

(1) $n<k$：返回空集；

(2) $k=1$：直接着色；

(3) $k\leq n\leq 2k$：PBCC 算法处理；

(4) $n>2k$：n 减半，递归调用.

算法 HABCC 中的递归子程序 IDCC 描述如图 1 所示.

```
算法 IDCC(n,k)
Input: An integer n=|U|, an integer k=|C|
       where n≥1 且 k≥1
Output: A coloring scheme S(n,k)
1. if n<k then return ∅;
2. if k=1 then return {⟨1,1,…1⟩};
3. if n≤2k then return PBCC(n,k);
4. S(n,k)=∅;
5. Partition U into 2 parts, |U1|=n1=⌈n/2⌉, |U2|=
   n2=⌊n/2⌋;
6. S(n,k)←IDCC(n1,k)⊕IDCC(n2,k);
7. for i←1 to k−1 do
     S(n,k)←S(n,k)∪(IDCC(n1,i)⊗IDCC(n2,k−i));
   end
8. return S(n,k);
```

图 1 HABCC 中递归子程序 IDCC

正如 2.2 节中的介绍，在 $n>k^2$ 情况下，核心化预处理可以大幅降低问题规模 n 对算法复杂度的影响. 在算法 HABCC 中，若 $n>k^2$，则由分治着色算法首先构造(k^2,k)着色方案，再利用散列函数，映射为(n,k)着色方案；若 $n\leq k^2$，则直接利用分治着色算法构造(n,k)着色方案. 算法 HABCC 的描述如图 2 所示.

```
算法 HABCC(n,k)
Input: An integer n=|U|, an integer k=|C|
       where n≥1 且 k≥1
Output: A coloring scheme S(n,k)
1. S(n,k)=∅;
2. if n>k² then
     S(k²,k)←IDCC(k²,k)={H1,H2,…,Hr};
     S(n,k)←{Hi·ψa | 1≤i≤r, a∈Zp};
3. else
     S(n,k)←IDCC(n,k);
   end
4. return S(n,k);
```

图 2 基于混合策略的着色算法 HABCC

3 HABCC 算法分析

3.1 HABCC 算法的正确性分析

定理 3. HABCC 算法产生的着色方案能够覆盖全集 U 的所有 k 元素子集.

证明. 当 $n>k^2$ 时，根据引理 2 及其相应的着色方案构造算法，只需证明 IDCC 算法能产生(k^2,k)着色方案；当 $n\leq k^2$ 时，需要证明 IDCC 算法能产生(n,k)着色方案. 综上所述，只需证明 $n\leq k^2$ 时，IDCC 算法能产生(n,k)着色方案. 这样，HABCC 算法能构造(n,k)着色方案，从而覆盖全集 U 的所有 k 元素子集.

在 $n\leq k^2$ 情况下，IDCC 算法分情况处理：(1) $n<k$：返回空集；(2) $k=1$：所有元素着相同颜色

并返回；(3) $k\leq n\leq 2k$：PBCC 算法处理，返回(n,k)着色方案；(4) $2k<n\leq k^2$：n 减半，枚举 k 分布情况，递归调用. 在情况(1)(2)下，显然 IDCC 算法可以返回(n,k)着色方案；情况(3)时，根据文献[4]中的理论，同样返回(n,k)着色方案；情况(4)时，依据简单的分治枚举策略，将问题递归到更小的规模. 由于递归下降的性质，任何(n,k)的着色问题都可以递归到以上四种情况之一. 在不超过 $2\lceil\log k\rceil$ 层递归调用后，此时 $n=1,k\geq 1$，返回着色方案，递归结束. 由分治枚举的性质知，在 $n\leq k^2$ 的情况下，IDCC 算法都返回(n,k)着色方案.

因此，HABCC 算法能构造(n,k)着色方案，从而覆盖全集 U 的所有 k 元素子集.　　证毕.

3.2　HABCC 算法产生的着色方案规模分析

为了便于与 PH 算法比较，首先回顾一下文献[3]中的相关内容，分析 PH 算法的着色方案规模. PH 算法产生的着色方案规模依赖于 C1～C5 的参数组合，分析所需枚举的各个参数的数目，得到以下数据：

C1. 枚举整数对(a,b)，需要 $p^2=O(n^2)=O(k^4)$；

C2. 枚举整数序列 C，需要 $\binom{5k/4-1}{k/4-1}=O(1.8692^k)$；

C3. 枚举 r 个整数对的序列 L，其中 $r\leq\log k-2$，需要 $p^{2r}=O(4^{\log k-2}\,k^{4\log k-8})$；

C4. 枚举 C 与 L 间的映射关系，需要 $2^{k/2}=O(1.4143^k)$；

C5. 枚举满足条件的子着色方案，需要 $O(2.4142^k)$.

从以上数据可以看出，PH 算法的理论复杂度[①] $O^*(6.1^k n)$中的 6.1^k来自于所有的指数因子部分，而其余的部分都作为低阶因子被忽略. 从算法理论的渐近上界表示角度考虑，这种做法是允许的. 然而，从实际的角度分析，很多因子仍然是必须考虑的. 就 PH 算法而言，综合所有的情况，着色方案规模应为 $O^*(6.1^k 4^{\log k-1}k^{4\log k-1}n)$. 以 $n=10000,k=20$ 为例，$n=1\mathrm{E}4,4^{\log k-1}=1\mathrm{E}2,k^{4\log k-1}=1.55\mathrm{E}21$，与 $6.1^k=5.09\mathrm{E}15$ 相比，显然 $k^{4\log k-1}$是不能忽略的. 图 3 展示了随着 k 值的增长各因子的增长趋势.

尽管随着 k 值的增长，因子 6.1^k终将取得决定性作用，但从图 3 可以看出，在整个问题规模增大到无法现实求解之前，因子 $k^{4\log k-1}$都仍然有着重大的作用，因而是无法忽略的. 只有在同时充分考虑两者的影响时，$k^{4\log k-1}6.1^k n$ 才大致反映了 PH 算法着色方案规模的增长趋势.

值得注意的是：在图 3 的比较中，我们没有考虑相同的核心化预处理过程所产生的影响. 从图 3 可以看出，相同 k 值情况下，$2^{k-1}\cdot\lceil\log k\rceil^{k-1}$比 $6.1^k 4^{\log k-1}k^{4\log k-1}$ 小数个数量级，大约仅为 $6.1^k 4^{\log k-1}k^{4\log k-1}$ 的平方根. 经过数值计算，只有当 $k>180$ 时，$2^{k-1}\cdot\lceil\log k\rceil^{k-1}$才大于 $6.1^k 4^{\log k-1}k^{4\log k-1}$，而此时规模已达到 10^{210}. 因此，在现实可计算的范围内，HABCC 算法产生的着色方案规模上界远小于 PH 算法.

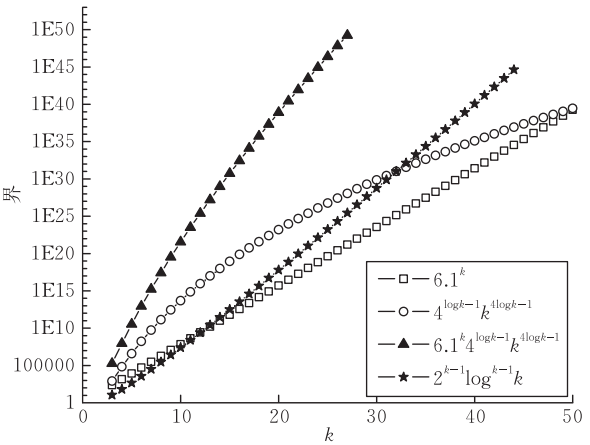

图 3　k 值增长时各因子增长趋势比较

有了以上分析后，接下来比较分析 HABCC 与 PH 算法产生着色方案的实际规模. 首先，根据式(1)以及文献[4]中的式(1)，HABCC 算法产生的着色方案规模可以在不产生实际着色方案的情况下计算得出. 此外，根据文献[3]中的理论，PH 算法产生的着色方案也可以通过枚举以下参数组合情况计算得出. 具体步骤如下：

1. 依据 C2，用一个长度为 $5k/4-1$ 且包括正好 $k/4-1$ 个 0 的二进制串 B，其中第 j 段正好包含了 c_j 个 1，来表示 C. 按照规则$\sum_{j=0}^{k/4-1}c_j(c_j-1)$过滤这些候选串.

2. 依据 C5，对符合上面条件的二进制串，计算 $T=\prod_j|F_{c_j}|$，其中$|F_{c_j}|$已知[②].

3. 将 c_j和(a_i,b_i)之间的映射关系表示为长度为$\sum_j i\cdot q/2^i$的二进制串$A>1$，其中包括正好 j 个 0 并且 q 表示$c_j>1$的数目. 因此，第 j 个 01^{i-1}的子串就可以代表 c_j和(a_i,b_i)之间的关系.

---

① 实际为 $O^*(6.4^k n)$，更复杂的分析才得到 $O^*(6.1^k n)$的结果.

② $|F_{c_j}|$数据来自文献[3].

4. 依据 C3，对剩余的 $A>1$，计算 $R=\sum_{A>1}T\cdot(p\cdot(p-1))^{i'}$，其中 p 是不小于 k^2 的最小素数，而 i' 为 $A>1$ 中最长子串 01^{i-1} 的参数 i.

5. 对步 1 中的所有 B 计算 R，并求和 $Q=\sum_{B}R$. 若 k 能被 4 整除，则 $H=Q\cdot p$，即为着色方案的规模.

6. 若 k 不能被 4 整除，计算 $k'=k+4-k\%4$，根据步 1～5 计算 Q'，p' 是不小于 k'^2 的最小素数，则着色方案规模 $H=\binom{k'}{k}\cdot Q'\cdot p'$.

这样，PH 算法产生的着色方案实际规模也可以在不生成实际方案的情况下计算出来. 图 4 展示了 $n=10000$ 的情况①下，HABCC 算法、PH 算法的着色规模以及组合数(COMBIN)C_n^k 的比较.

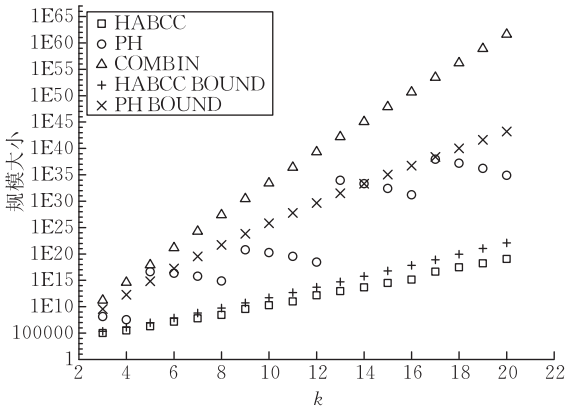

图 4 着色方案规模比较($n=10000$)

从图 4 可以看出，$n=10000$ 的情况下，HABCC 与 PH 算法产生的着色方案规模都小于组合数，且随着 k 值的增长，在可计算的范围内，规模增长趋势也较缓. 且由于核心化处理削弱了问题规模对复杂度的影响，因此，n 值越大，HABCC 与 PH 算法相对于组合算法的优势更明显. HABCC 算法和 PH 算法采用了相同的核心化预处理，使得着色方案规模只与问题规模成线性关系，因此，HABCC 和 PH 算法的比较与 n 值无关. 从图 4 还可以看出，在相同 k 值情况下，HABCC 算法产生的实际着色方案规模的数量级大约仅为 PH 算法的一半.

另外，PH BOUND 和 HABCC BOUND 分别表示 PH 以及 HABCC 算法产生的着色方案规模上界，从图 4 中可以看出，两种算法得到的规模上界与实际值都很接近，是很紧的界. 一方面说明了定理 1、2 中关于着色方案规模上界的分析很成功，另一方面也显示了图 3 中关于两者规模上界的比较是很有意义的. 同时发现，当 k 取 5 或 13 时，PH 算法产生着色方案的实际规模超过了上界，这是由于理论分析时忽略了向上取整的影响，而实际构造时必须考虑.

综上所述，HABCC 算法提供了优化的着色方案，其规模不仅远小于组合数目，而且在现实可计算的范围内，比目前最优化的构造算法小数个数量级.

4 结 论

彩色编码是解决 NP 难问题的一种新兴而重要的技术，该技术，特别是其确定化技术，在最近几年的研究中得到了许多有重要意义的改进. 目前最好的结果是 Chen 等人在文献[3]中提出的一种 PH 算法，其时间复杂度被证明为 $O^*(6.1^k n)$. 而另一方面，为了使彩色编码技术更加实用化，文献[4]提出了一种基于划分的着色算法，将彩色编码应用在 $n\leq 2k$ 的情况下，弥补了彩色编码在非小参数化中应用的空白.

本文以分治算法为基础，结合核心化技术，提出了混合策略构造着色方案的算法 HABCC，该算法有机地融合了 PH 及 PBCC 算法的优点，同时避免了它们不足. 经过理论及实际数据分析，该算法可以得到比现有最好算法更小的着色方案，且易于构造，从而大幅改进了彩色编码的实用性.

参 考 文 献

[1] Alon N, Yuster R, Zwick U. Color-coding. Journal of the ACM, 1995, 42(4): 844-856

[2] Monien B. How to find long paths efficiently. Annals of Discrete Mathematics, 1985, 25: 239-254

[3] Chen Jianer, Lu Songjian, Sze Sing-Hoi, Zhang Fenghui. Improved algorithms for path, matching, and packing problems//Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms. New Orleans, Louisiana, USA, 2007: 298-307

[4] Wang Jian-Xin, Liu Yun-Long, Chen Jian-Er. An effective coloring algorithm for close relationship between the scales of element set and color set and its application. Chinese Journal of Computers, 2008, 31(1): 32-42(in Chinese)
(王建新，刘云龙，陈建二. 一种在元素与颜色规模相近时的有效着色算法及其应用. 计算机学报，2008，31(1)：32-42)

[5] Kneis J, Molle D, Richter S, Rossmanith P. Divide-and-color//Proceedings of the 32nd International Workshop on Graph-Theoretic Concepts in Computer Science. Bergen, Norway, 2006: 58-67

① PH 算法需要 n 远大于 k 的限制，应在其适用范围内比较.

[6] Fredman Michael L, Komlós János, Szemerédi Endre. Storing a sparse table with $O(1)$ worst case access time. Journal of the ACM, 1984, 31(3): 538-544

[7] Schmidt Jeanette P, Siegel Alan. The spatial complexity of oblivious k-probe hash functions. SIAM Journal on Computing, 1990, 19(5): 775-786

[8] Bodlaender H. On linear time minor tests with depth-first search. Jounal of Algorithms, 1993, 14(1): 1-23

[9] Scott Jacob, Ideker Trey, Karp Richard M, Sharan Roded. Efficient algorithms for detecting signaling pathways in protein interaction networks//Proceedings of the 9th Annual International Conference on Research in Computational Molecular Biloogy. Cambridge, MA, USA, 2005:1-13

[10] Styczynski M, Jensen K, Rigoutsos I, Stephanopoulos G. An extension and novel solution to the (l, d)-motif challenge problem. Genome Informatics, 2004, 15(2): 63-71

[11] Wang Jian-Xin, Huang Yuan-Nan, Chen Jian-Er. A motif finding algorithm based on color coding technology. Journal of Software, 2007, 18(6): 1298-1307(in Chinese)
(王建新，黄元南，陈建二. 一种基于彩色编码的基序发现算法. 软件学报，2007，18(6)：1298-1307)

[12] Liu Y, Lu S, Chen J, Sze S-H. Greedy localization and color-coding: Improved matching and packing algorithms//Proceedings of the 2nd International Workshop on Parameterized and Exact Computation. Zurich, Switzerland, 2006: 84-95

[13] Hüffner Falk, Wernicke Sebastian, Zichner Thomas. Algorithm engineering for color-coding to facilitate signaling pathway detection//Proceedings of the 5th Asia-Pacific Bioinformatics Conference. Hong Kong, China, 2007: 277-286

[14] Betzler Nadja, Fellows Michael R, Komusiewicz Christian, Niedermeier Rolf. Parameterized algorithms and hardness results for some graph motif problems//Proceedings of the 19th Annual Symposium on Combinatorial Pattern Matching. Pisa, Italy, 2008: 31-43

WANG Jian-Xin, born in 1969, Ph. D., professor. His research interests include computer algorithm, network optimization theory, bioinformatics.

YANG Zhi-Biao, born in 1983, master. His main research interests include parameterized computation, computer theory.

LIU Yun-Long, born in 1983, master. His main research interests include parameterized computation, computer theory.

CHEN Jian-Er, born in 1954, Ph. D., professor. His main research interests include bioinformatics, computer theory, computation complexity and optimization.

Background

This work is supported by the National Basic Research Program of China (973 Program) (2008CB317107) which focuses on the design of parameterized algorithm in various information processing fields, the National Natural Science Foundation of China under Grants (60773111) which focuses on the development of parameterized algorithm technique and the application of parameterized algorithm in network, the Doctoral Discipline Foundation of Higher Education Institution (20090162110056) and the Program for Changjiang Scholars and Innovative Research Team in University of China under Grant (IRT0661). In the field of parameter computation, Color-Coding is an important technique which can be used to design efficient algorithm for various NP-hard problems, such as k-path, 3-set packing etc. The advantage of Color-Coding technique is that it can reduce the search space of given instance by making all elements in the objective solution have different colors. In this paper, from practical point of view, based on the PBCC algorithm handling the case $n \leq 2k$, the authors give a combined coloring algorithm using divide and conquer method. The size of color-coding scheme constructed by our algorithm is much smaller than the one constructed by the current best algorithm, which enhances efficiency of Color-Coding technique. Moreover, the authors also give a theoretical lower bound about the size of the color-coding scheme constructed by our algorithm.