文章编号:1007－6735(2018)06－0519－06 DOI: 10.13255/j.cnki.jusst.2018.06.002

# 最大团问题的竞争决策算法

黄 飞， 宁爱兵， 刘志民， 何永梅， 张惠珍
（上海理工大学 管理学院，上海 200093）

**摘要：** 分析了最大团问题的数学性质，根据推导出来的性质设计求解最大团问题的竞争决策算法，且算法的时间复杂度分析结果为 $O(n^3)$。并用提出的算法求解最大团问题中的标准测试示例，测试结果表明，算法具有良好的求解效果。

**关键词：** 竞争决策算法；最大团；竞争力函数；决策函数；资源交换规则
**中图分类号：** TP 301.6 **文献标志码：** A

# Competitive Decision Algorithm for Maximum Clique Problems

HUANG Fei, NING Aibing, LIU Zhimin, HE Yongmei, ZHANG Huizhen
(*Business School, University of Shanghai for Science and Technology, Shanghai 200093, China*)

**Abstract:** The mathematical properties of maximum clique problems were analyzed, which can be used to put forward the design of a competitive decision algorithm. The time complexity of the algorithm was analyzed, and the result is $O(n^3)$. To assess the efficiency of the algorithm, it was applied to a set of benchmark problems in maximum clique problems. It turns that our algorithm clearly outperforms other heuristics for solving maximum clique problems, while obtaining better or comparable solutions.

**Keywords:** *competitive decision algorithm*; *maximum clique problem*; *competitive force function*; *decision function*; *resources exchange rules*

最大团问题（MCP）[1-3]是组合优化中的 NP 难题（non-deterministic polynomial problems），在实践中有着广泛的应用，如社会网络分析[4]、生物信息学[5]、分子生物学、编码理论[6]及经济学[7]等，是图论、运筹学及离散数学等领域中一个重要的研究目标。

本文介绍、提出并论证了最大团的部分数学性质，在此基础上设计了一个求解该问题的竞争决策算法（CDA）。为了阐述算法的基本原理，本文给出了一个简单的案例，运用该算法进行手动求解。最后运用该算法对最大团问题的标准测试图进行求解，计算实验得到了较好的结果。测试中有些情况下还可以得到目前最优解，如后面的实例 p_hat300-2，MANN_a27 和 MANN_a45。

**收稿日期：** 2017−11−22
**基金项目：** 国家自然科学基金资助项目（71401106）；上海高校一流学科建设计划（S1201YLXK）；高等学校博士学科点专项科研基金联合资助课题（20123120120005）
**第一作者：** 黄 飞（1990−），女，硕士研究生．研究方向：算法设计、系统工程．E-mail：15800913627@163.com
**通信作者：** 宁爱兵（1972−），男，副教授．研究方向：算法设计、系统工程．E-mail：nabnab@163.com

# 1 竞争决策算法

自从CDA在文献[8]中提出之后，该算法被应用在了很多NP问题的求解上，具体可参阅文献[9-13]，且都得到了较好的结果。本文仅给出算法的原理与概念，算法的详细情况可参阅文献[9]。

## 1.1 原理

竞争决策现象普遍存在于大自然中，这些竞争决策都在一定的法则下进行，且同时受到一种或多种因素的共同影响。竞争决策就是参与竞争的各方根据竞争决策机制分别占据一定的资源的过程。竞争决策之后，会达到一个新的均衡状态。如果新形成的均衡状态优于初始状态，则能实现优化的目的。CDA就是模拟这个机制来实现优化。它首先根据求解的问题构造出一个或多个竞争参与者，参与对资源的竞争，然后利用相应的竞争机制，通过决策判定竞争者是否占有资源以及占有哪些资源，直至达到均衡状态。

## 1.2 基本概念

**a.** 竞争者：参与竞争资源的各方，在本文中，如果算法中只存在一个竞争者，则假设存在另一个虚构的竞争者$N$(可以假定代表自然)，同时参与争夺资源。$N$没有竞争力函数。

**b.** 资源：竞争者所争夺的目标，整个算法的核心。

**c.** 竞争决策状态：某一瞬间，所有参与竞争的各方各自所占有的资源的一种状态。初始状态下，参与竞争的各方均不占有资源，虚拟竞争者除外，且占有所有资源。

**d.** 竞争力函数：代表竞争者争夺资源的能力。

**e.** 决策函数：决策函数起到判定的作用，用于判定资源的分配，分3种情况。

(a)若除了虚拟竞争者之外只有一个竞争者，决策函数用于判定该竞争者能否从虚拟竞争者那里争夺资源；

(b)若除了虚拟竞争者之外有多个竞争者，决策函数不仅要判定哪些竞争者可以优先占有资源，还要判定这些竞争者具体可以占有哪些资源；

(c)决定是否要从某个竞争者手中强制性地剥夺出资源，并将资源分配给另外一个竞争者。

**f.** 竞争决策均衡状态：完成资源的初步分配后所达到的一种状态。均衡状态下，非虚拟竞争者不能根据竞争决策机制争夺更多资源，即除非引入资源交换规则，否则将不再进行竞争决策。

**g.** 资源交换规则：达到均衡状态后，资源交换规则将强制全部或部分竞争者(包括虚拟竞争者)之间相互交换资源，使竞争进入非稳定的均衡状态。非稳定的均衡状态下，非虚拟的竞争者能根据竞争决策机制再次争夺资源，直至达到一个新的均衡状态。进行资源交换后，必须使竞争进入非稳定的状态，这样才能使竞争者对资源再次进行竞争，达到新的竞争决策均衡状态，实现优化的目的。

# 2 最大团问题的竞争决策算法

## 2.1 问题介绍

最大团问题描述如下：给定1个简单无向图$G=(V, E)$，$S$是结点集合$V$的1个子集，若$S$中任意2个结点之间都相邻，即由$S$导出的子图$G[S]$是完全子图，且$G[S]$不包含在图$G$的更大的完全子图中，则称$G[S]$为团。最大团问题就是求出图中结点个数最多的团。

## 2.2 数学符号

$G=(V, E)$：$G$代表简单的无向图，$V$代表图的结点集合，$E$代表图的边集，且$E$由$V$中的结点对表示。

$n$：图中结点的个数。

$N(v)$：结点$v$的开邻集，所有与结点$v$相邻的点的集合。

$N[v]$：结点$v$的闭邻集，即$N(v)\cup\{v\}$。

$d(v)$：结点$v$的度，即集合$N(v)$中所包含的结点的个数。

$G[V_1]$：$G[V_1]$是图$G=(V, E)$的点导出子图，$G[V_1]=(V_1, E_1)$，其中，$V_1\subset V$，$E_1\subset E$，且对于图$G$中的每一条2个端点都在$V_1$中的边$e_i$都有$e_i\in E_1$。

$G_1$：初始状态下，$G_1$等于原图$G$，即$G_1=(V, E)=(V_1, E_1)$。

$d(v_i)$：结点$v_i$的度，即图$G$中与$v_i$相邻的点的个数。

$d_1(v_i)$：结点$v_i$的度，即图$G_1$中与$v_i$相邻的点的个数。

$p(i)$：结点$v_i$在$G_1$图中的竞争力函数。

$G_b$：算法在均衡时所求得的最大完全子图。

$V_{ad}$：与完全子图$G_b$中各个结点相邻结点的集合，即

$$V_{ad}=\bigcap_{v\in V_b}N(v) \quad (1)$$

$G_{max}$：目前为止算法中所发现的最大团。

$|G|$：$G$ 图中所包含的结点的个数。

$|V|$：结点集合所包含的结点个数。

### 2.3　数学性质

现仅给出竞争决策算法所涉及到的最大团的数学性质，性质 1～3 的证明参阅文献[9−10]。推论 1 及性质 4 为本文提出的。算法中，推论 1 用于删除冗余结点，性质 4 用于资源交换。

**性质 1**　若结点集合 $V$ 中某个结点 $v$ 的度是 0，那么，$v$ 必然不属于所求的最大团的结点集合 $S$[14]。

**性质 2**　若结点集合 $V$ 中某结点 $v$ 的度为 $n-1$，那么，$v$ 必然属于所求的最大团的结点集合 $S$。

**性质 3**　$S$ 为求得的最大团的结点集合，若 $v\in S$ 且 $d(v)=k$；那么，$|S|\leqslant k+1$[15]。

**推论 1**　设已求出的最大完全子图为 $G_{max}$，其结点个数为 $|G_{max}|$，$G_1$ 中某一结点 $v$ 的度 $d(v)=k$，如果 $k<|G_{max}|$，则可以将 $v$ 从 $G_1$ 中删除，从而使问题得到简化。

**证明**　根据性质 3 可得，度为 $k$ 的结点 $v$ 所在的完全子图最多有 $k+1$ 个顶点，若 $k\leqslant|G_{max}|-1$，则包含 $v$ 的完全子图的顶点个数最多为 $|G_{max}|$ 个，不大于已求出的最大完全子图 $G_{max}$ 的结点个数，所以，将 $v$ 作为冗余结点从 $G_1$ 中删除，从而使问题得到简化而不影响求解结果。

**性质 4**　$G_2=(V_2, E_2)$ 是 $G_1=(V_1, E_1)$ 的 1 个子图，且是完全图，若 $V_3\subset V_1-V_2$，$G_3=(V_3, E_3)=G[V_3]$，$V_4\subset V_2$，$V_3$ 与 $V_2-V_4$ 中的所有结点都相邻，若 $|V_3|>|V_4|$，那么，将 $V_4$ 从 $G_2$ 中删除，将 $V_3$ 加入 $G_2$，此时，$G_2$ 为更大的完全子图。

**证明**　$G_3$ 本身是 1 个完全子图，且 $V_3$ 中的所有结点与 $V_2-V_4$ 中的所有结点都相邻，因而新生成的图 $G_2$ 是 1 个结点个数更多的完全子图。

### 2.4　算法思想

在 CDA 中，结点是竞争者争夺的资源，所需求解的最大团是竞争者 $A$，同时参与竞争的是虚拟竞争者 $N$。初始状态时，虚拟竞争者 $N$ 占有全部结点，竞争者 $A$ 不占有任何结点。达到均衡状态时，竞争者 $A$ 具有的结点即为所求的最大团。

### 2.5　初始状态、竞争力函数、决策函数、资源交换规则

**a.** 初始状态。

对于含有 $n$ 个结点的图，一共有 $n$ 个初始状态，将结点按各个度的大小降序排列之后，第 $i$ 个初始状态为竞争者 $A$ 仅占有资源 $v_i$，其余结点资源都被虚拟竞争者 $N$ 所占有。在每个初始状态开始时需要检查 $d(v_i)$ 是否小于已知最大完全子图中结点的个数，若是，则该初始状态不会出现更好的解且放弃掉该初始状态。

**b.** 竞争力函数。

采用一个竞争力函数

$$p(i)=\begin{cases} d_1(v_i), & v_i\in V_1 \\ 0, & \text{其他} \end{cases} \quad (2)$$

**c.** 决策函数。

本文只有一个决策函数，即在竞争中将竞争力函数值 $p(i)$ 最大的结点优先加入到图 $G_b$ 中，当多个结点的竞争力函数值相等时，则将编号小的结点加入到图中。

**d.** 资源交换规则。

资源交换规则见下面的算法流程。

### 2.6　算法的过程

算法的过程如下：

**a.** 初始图 $G_b$ 和 $G_{max}$ 是 1 个没有结点、没有边的空图，首先将 $G_1$ 中竞争力函数 $p(i)$ 最大的点 $v_i$ 加入 $G_b$ 中，然后将 $V_{ad}$ 中竞争力函数值 $p(i)$ 最大的结点加入到 $G_b$ 中。根据式(1)重新计算 $V_{ad}$，持续该步骤，直到 $V_{ad}$ 为空集。

**b.** 检查是否存在冗余结点，如果存在冗余结点，则删除冗余结点并更新竞争力函数。具体的做法参见下面的算法流程。

**c.** 执行资源交换，交换后再次检查是否存在冗余结点，如果存在，则剔除冗余结点后更新竞争力函数。详细的内容见下面的算法流程。

### 2.7　算法流程

**步骤 0**　$k=1$；将结点按各个结点的度从大到小排序，$V_{max}=\{\}$，$E_{max}=\{\}$，$G_{max}=(V_{max}, E_{max})$。

**步骤 1**　初始化。

设置初始状态。竞争者 $A$ 只占有资源 $v_k$，其余结点都被虚拟竞争者 $N$ 占有。由推论 1 可知，若 $|d(v_i)|<|V_{max}|$，则该初始状态不会出现更好的解，因此，此时跳到步骤 3 而放弃该初始状态；否则，继续执行下面的操作：

$G_1=(V_1, E_1)=(V, E)=G$，从 $G_1$ 中删除结点 $v_k$ 以及不与 $v_k$ 相邻的所有结点，$V_b=\{v_k\}$，$E_b=\{\}$，$G_b=(V_b, E_b)$，$V_i=\{v_k\}$，$V^*=\{\}$，$E^*=\{\}$，$G^*=(V^*, E^*)$，$V_{ad}=\bigcap_{v\in V_b}N(v)$。

**步骤 2** 竞争决策。

根据式(2)，计算初始状态下竞争者 $A$ 对结点资源的竞争力函数值 $p(i)$，并降序排列。

**a.** 本轮竞争阶段 1：资源分配阶段。

(a)根据性质 1，删除竞争力函数值为 0 的结点，若删除后 $G_1$ 为空集，算法结束，问题没有意义。

根据性质 2 将度为 $n-1$ 的结点全部放入 $G_b$ 中。

(b)根据式(1)计算 $V_{ad}$，并将 $V_{ad}$ 中竞争力函数值 $p(i)$最大的结点加入到 $G_b$ 中，之后重新计算 $V_{ad}$，持续该过程，直到 $V_{ad}$ 为空集。此时得到的图是 1 个完全子图，且结点个数为 $|G_b|$。若 $|G_b|>|G_{max}|$，则 $|G_{max}|=|G_b|$；否则，执行步骤 c。

**b.** 本轮竞争阶段 2：删除冗余结点。

利用推论 1 删除冗余结点，即将图 $G_1$ 中度小于 $|G_{max}|$的顶点删除，并重新计算 $G_1$ 中结点的竞争力函数，降序排列，输出图 $G_1$ 此时的结点个数 $n$。

**c.** 本轮竞争阶段 3：资源交换阶段。

为了提高算法的速度，本算法只找出性质 4 中 $|V_3|=2$ 且 $|V_4|=1$ 的结点，用 $V_3$ 中的 2 个结点交换 $V_4$ 中的 1 个结点，这样能得到一个更大的完全子图 $G_b$。持续查找和替换这种操作，直至图 $G_1$ 中不存在这种满足资源交换条件的结点。此时，若 $|G_b|>|G_{max}|$，则 $|G_{max}|=|G_b|$，利用推论 1 删除冗余结点；否则，舍弃 $G_b$，执行步骤 3。

**步骤 3** $k=k+1$，若 $k\leqslant n$，则跳到步骤 1；否则，跳到步骤 4。

**步骤 4** 算法结束。

输出竞争决策的结果。

## 3 算法的时间复杂度分析

步骤 2 的 a 最坏的情况下时间复杂度为 $O(n^2)$；步骤 2 的 b 剔除图中冗余结点在最坏的情况下时间复杂度为 $O(n)$；步骤 2 的 c 最多执行的次数为 $n$ 次，1 个初始状态查找并执行 1 次交换的时间复杂度为 $O(n^2)$，因此，步骤 2 的 c 在最坏的情况下时间复杂度为 $O(n^3)$。综上所述，该算法的时间复杂度为 $O(n^3)$。

## 4 示例分析

现给出 1 个案例来阐述算法的原理以及过程，同时也只给出第 1 个初始状态下的竞争决策过程，如图 1 所示。

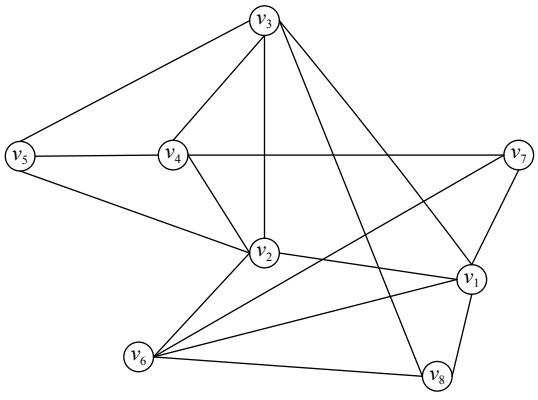


**图 1 示例图 $G_1$**

**Fig.1 Sample graph $G_1$**

竞争决策计算如下：

**a.** 初始化阶段。

给出第 1 个初始状态下的竞争决策过程，因此，最开始时可以设置竞争者 $A$ 没有占有资源结点 $v_1$，其他结点资源都被虚拟竞争者 $N$ 占有。从图 $G_1$ 中删除 $v_1$ 以及不与 $v_1$ 相邻的所有结点 $v_4$ 和 $v_5$，$V=V_1=\{v_2, v_3, v_6, v_7, v_8\}$，$G_1=(V, E)=(V_1, E_1)$，$G_b=(V_b, E_b)$，$V_b=\{v_1\}$，$E_b=\{\}$，$G_{max}=(V_{max}, E_{max})$，$V_{max}=\{v_1\}$，$E_{max}=\{\}$。

**b.** 资源分配阶段。

根据式(1)可得集合 $V_{ad}=\{v_2, v_3, v_6, v_7, v_8\}$，计算竞争力函数，$p(1)=0$，$p(2)=5$，$p(3)=5$，$p(4)=0$，$p(5)=0$，$p(6)=4$，$p(7)=3$，$p(8)=3$。

将 $V_{ad}$ 中竞争力函数值最大的 1 个结点加入 $V_b$ 中，如果存在若干个竞争力函数值最大的结点，任选 1 个加入到 $V_b$ 中。因此，先将 $v_2$ 加入到 $V_b$，再重复计算 1 次后将 $v_3$ 也加入到 $V_b$，此时得到 1 个团 $V_b=\{v_1, v_3, v_2\}$，且 $V_{ad}$ 为空集。

因初始时 $V_{max}=\{v_1\}$，所以，此时满足 $|G_b|>|G_{max}|$，将 $G_b$ 中的结点放入 $G_{max}$，$|G_{max}|=3$，$V_{max}=\{v_1, v_3, v_2\}$。

**c.** 根据 b 删除图中冗余结点。

删除 $V_1$ 中度小于 $|V_{max}|=3$ 的所有结点及与其相连的边，因图中所有结点的度都大于等于 3，所以，此过程没有冗余结点被删除。

**d.** 资源交换阶段。

找出性质 4 中 $|V_3|=2$ 且 $|V_4|=1$ 的结点，此时找到 $V_3=\{v_4, v_5\}$，$V_4=\{v_1\}$，用 $V_3$ 中的 2 个结点替换 $V_4$ 中的 1 个结点，从而得到 1 个更大的完全子

图 $G_b$=( $V_b$, $E_b$)，其中，$V_b$=$V_{max}$={$v_2$, $v_3$, $v_4$, $v_5$}，该完全子图如图 2 所示。在图 1 中利用推论 1 删除冗余结点，即将图 1 中度小于|$V_{max}$|=4 的顶点删除，据此依次删除 $v_7$, $v_8$, $v_6$, $v_1$, $v_2$, $v_3$, $v_4$, $v_5$, 图 1 变为空图，因此，$S$={$v_2$, $v_3$, $v_4$, $v_5$}为最优解。虽然这个实例得到了最优解，但是，这个算法并不能保证求解所有的实例时都能得到最优解。

**e.** 输出竞争决策结果。

$V_{max}$={$v_2$, $v_3$, $v_4$, $v_5$}，最优，其代表的图如图 2 所示。

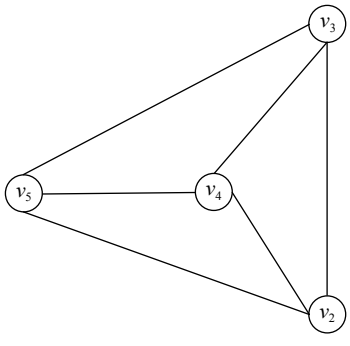


**图 2　最大团的图**

**Fig.2　Graph of maximum clique**

## 5　数值计算及分析

为了对算法的有效性进行验证，用 Java 语言在 PC 机上进行计算实验。运用本算法对 DIMACS (The Center for Discrete Mathematics and Theoretical Computer Scinence) 的基准图进行求解，并与目前最好的结果进行比较，如表 1 所示，相关的测试实例可以登录 http://iridia.ulb.ac.be/~fmascia/maximum_clique/DIMACS-benchmark#detDSJC1000_5 下载。

$$e=\frac{m-l}{m}\times 100\%$$

式中：$e$ 为误差率；$m$ 为目前最好解；$l$ 为本文解。

最大团问题是 NP 难解问题，除非能够证明所有 NP 问题都可以转化为具有多项式算法的判定问题，否则不存在多项式时间的精确算法。精确算法能够求出最优解，但是，求解时间随着问题规模的增大而呈指数增长，时间复杂度较高。启发式算法求解速度快，但是，一般情况下不能找到最优解，只能找到近似解或者局部最优解。本文的竞争决策算法是启发式算法，利用本文算法对基准图进行计算求解，在 Eclipse 软件上运行，从表 1 中可以看出，利用 CDA 求得的解，和最优解之间的误差率基本保持在 10% 以内。有些示例可以取得与目前最好的结果相同的解，如示例 hamming8−4 和示例 p_hat300−2，有些示例可以取得与目前最好结果极其相近的解，如示例 C125.9，MANN_a27，MANN_a45。作为比较，Belachew 等[11]用秩为 1 的对称非负矩阵逼近算法求解 DIMACS 中的示例，求解示例 C500.9，MANN_a27，p_hat1500−2，p_hat1500−3 得到的最好解分别为 50，123，61，92，而本文的求解结果分别为 52，125，59，85。文献[16]还将求解结果与其他 2 个启发式算法的求解结果进行比较，结果显示，启发式算法在求解的 36 个案例中，只有 3 个案例得到了与该算法相近的解。

**表 1　CDA 算法结果与目前最好结果的比较**

**Tab.1　Comparison between the results of CAD algorithum and the best results at present**

| 示例名称 | 结点个数 | 边密度 | 目前最好解 | 本文解 | 误差率/% |
|---|---|---|---|---|---|
| C125.9 | 125 | 0.90 | 34 | 33 | 2.94 |
| C500.9 | 500 | 0.90 | 57 | 52 | 8.77 |
| DSJC500_5 | 500 | 0.50 | 13 | 12 | 7.69 |
| MANN_a27 | 378 | 0.99 | 126 | 125 | 0.79 |
| MANN_a45 | 1 035 | 0.99 | 345 | 342 | 0.87 |
| gen200_p0.9_44 | 200 | 0.90 | 44 | 40 | 9.09 |
| hamming8-4 | 256 | 0.64 | 16 | 16 | 0 |
| keller4 | 171 | 0.65 | 11 | 9 | 9.09 |
| p_hat300-2 | 300 | 0.49 | 25 | 25 | 0 |
| p_hat300-3 | 300 | 0.74 | 36 | 32 | 11.11 |
| p_hat700-2 | 700 | 0.50 | 44 | 41 | 6.82 |
| p_hat700-3 | 700 | 0.75 | 62* | 59 | 4.84 |
| p_hat1500-2 | 1 500 | 0.51 | 65* | 59 | 9.23 |
| p_hat1500-3 | 1 500 | 0.75 | 94* | 85 | 9.57 |

## 6　结束语

竞争决策算法是解决组合优化问题的有效算法，该算法根据所求解的问题的数学性质设计出竞争规则、竞争力函数、决策函数以及设定初始格局就能很好地求解组合优化问题。求解最大团问题时，该算法在资源分配阶段即可形成 1 个初始解，根据性质 4 判断是否存在满足资源交换条件的结点，若存在则进入资源交换阶段，若不存在则舍弃该解，进入新一轮的竞争决策，同时根据推论 1 删除冗余结点。初始解越接近最优解，

即初始解越好，求解速度就越快，可以通过设计合理的竞争规则和竞争力函数以及决策函数来得到较好的初始解。特别是在求解稀疏图的最大团问题时，竞争决策算法可以得到较好的初始解，同时根据推论 1 快速删除较多冗余结点，实现快速求解。例如，1 个度小于等于 2 的图，根据性质 3 可知，该图的最大团最多只有 3 个结点，竞争决策算法可以在资源分配阶段就能得到最优解，同时根据推论 1 删除所有结点，问题得以快速求解。竞争决策算法原理简单，目前已经应用于 TSP 问题 (旅行商问题)、多目标 TSP 问题、0−1 背包问题、最小顶点覆盖问题等，具有普遍的适用性。

**参考文献：**


[1] FOMIN F V, GRANDONI F, KRATSCH D. A measure & conquer approach for the analysis of exact algorithms[J]. Journal of the ACM, 2009, 56(5): 25.

[2] 王丽爱, 周旭东, 陈崚. 最大团问题研究进展及算法测试标准[J]. 计算机应用研究, 2007, 24(7): 69–70.

[3] 任文涛. 关于最大团问题的分支搜索算法的优化[D]. 成都: 电子科技大学, 2012.

[4] SETHURAMAN S, BUTENKO S. The maximum ratio clique problem[J]. Computational Management Science, 2015, 12(1): 197–218.

[5] MALOD-DOGNIN N, ANDONOV R, YANEV N. Maximum cliques in protein structure comparison[C]// Proceedings of the 9th International Conference on Experimental Algorithms. Naples, Italy: Springer, 2009: 106–117.

[6] DU D Z, PARDALOS P M. Handbook of combinatorial optimization[M]. Dordrecht, Boston: Kluwer Academic Publishers, 1999: 394.

[7] BOGINSKI V, BUTENKO S, PARDALOS P M. Mining market data: a network approach[J]. Computers & Operations Research, 2006, 33(11): 3171–3184.

[8] 宁爱兵, 王波, 熊小华, 等. 竞争决策算法原理及其应用[J]. 上海理工大学学报, 2008, 30(4): 369–373.

[9] 支志兵, 宁爱兵, 陈吉珍, 等. 最大团问题的加权分治算法[J]. 计算机工程与应用, 2016, 52(2): 50–53.

[10] 宁爱兵, 刘艳芳, 王英磊. 最大团问题降阶算法[J]. 小型微型计算机系统, 2013, 34(5): 1137–1140.

[11] BELACHEW M T, GILLIS N. Solving the maximum clique problem with symmetric rank-one non-negative matrix approximation[J]. Journal of Optimization Theory and Applications, 2017, 173(1): 279–296.

[12] 宁爱兵, 马良. 大规模旅行商问题的竞争决策算法[J]. 计算机工程, 2005, 31(9): 23–26.

[13] 宁爱兵, 马良. 竞争决策算法及其在车辆路径问题中的应用[J]. 管理科学学报, 2005, 8(6): 10–18.

[14] 宁爱兵, 马良. 0/1 背包问题竞争决策算法[J]. 计算机工程与应用, 2008, 44(3): 14–16.

[15] 宁爱兵, 熊小华, 马良. 多目标旅行商问题竞争决策算法[J]. 计算机工程与应用, 2009, 45(34): 13–16.

[16] 金婷婷, 王波, 宁爱兵. 最小顶点覆盖问题的竞争决策算法[J]. 计算机工程与应用, 2011, 47(1): 32–34.


（编辑：石　瑛）