# 一种新的连珠棋局面表示法及其在六子棋中的应用

徐长明，马宗民，徐心和
(东北大学 信息科学与工程学院，辽宁 沈阳 110004)

**摘　要**：为了提高连珠棋局面的表示效率，给出了一种基于棋形来描述棋子间联系的表示方法，并在六子棋程序 NEUConn6 中成功运用.该方法不但紧凑、高效地描述了局面状态，还方便了局面的增量更新；此外，它把在线计算转化为离线计算，并且它很自然地把棋类知识和数据结构结合在一起.该方法不限于六子棋，可广泛用于别的连珠棋博弈程序.

**关 键 词**：机器博弈；连珠棋类；数据结构；棋形；增量更新
**中图分类号**：TP 391　　**文献标识码**：A　　**文章编号**：1005-3026(2009)04-0514-04

# A New Board Representation Method for *K*-in-a-row Games with Its Application to Connect6

*XU Chang-ming*, *MA Zong-min*, *XU Xin-he*
(School of Information Science & Engineering, Northeastern University, Shenyang 110004, China. Correspondent: XU Chang-ming, E-mail: changmingxu@gmail.com)

**Abstract**: A new method is proposed to improve the efficiency of the representation of the game situation, based on the pattern to describe the relationships among stones on a *k*-in-a-row game position. It has been used successfully in our Connect6 program (NEUConn6) to not only describe the state of a position efficiently but also cater for the incremental updating. This method can transform some online calculations into offline. Furthermore, it is a natural way to integrate the game knowledge with the data structure. Not limite to Connect6, the method can be adopted in other *k*-in-a-row games.

**Key words**: computer games; *k*-in-a-row; data structure; pattern; incremental updating

机器博弈[1]是人工智能领域的重要分支，它的研究对象多以复杂的棋牌类智力游戏为主.已经得到解决的棋类游戏[2-3]，几乎全部都应归功于机器博弈近半个世纪的发展.计算机解决问题的优势在于能把不易解析的问题，借助于现代计算机的运算速度优势枚举出所有的合理情形而得解；然而，博弈问题的复杂程度决定了它不能过度依赖机器的计算能力.许多待解决的或已经解决的棋类，其状态空间复杂度或博弈树复杂度，都不低于宇宙全部粒子总数的数量级.值得注意的是，经过领域知识的约束、增强知识有助于博弈问题的求解.而应用领域知识的前提依赖于对问题的良好理解和建立恰当的问题描述模型.根据连珠棋类的特点，如棋盘较大，以及行、列、对角线上的规律明显的特征，本文给出了一种普遍适于连珠棋的高效的局面表示方法.

## 1 六子棋简介

$C(m,n,k,p,q)$表示一族 $k$ 子棋游戏[4]，$C(15,15,5,1,1)$表示广为流传的五子棋.其中，更为有趣的是六子棋[4-5]，它可以形式化地表示成 $C(m,n,6,2,1)$.和五子棋相比，六子棋规则更简单，也更公平[6].在六子棋中，对弈的双方分别执黑子和白子.在 $m\times n$ 的棋盘上，除了黑方

**收稿日期**：2008-04-27
**基金项目**：国家自然科学基金资助项目(60774097)；辽宁省博士启动基金资助项目(20061017).
**作者简介**：徐长明(1978—)，男，黑龙江齐齐哈尔人，东北大学博士研究生；马宗民(1965—)，男，山东金乡人，东北大学教授，博士生导师；徐心和(1940—)，男，黑龙江哈尔滨人，东北大学教授，博士生导师.

第一步落 $q=1$ 颗子外，双方轮流下 $p=2$ 颗子。在水平、垂直、对角线方向，率先成连续不间断的 $k=6$ 子序列者胜。六子棋不存在吃子；有子的交叉点不能落子；棋盘大小无限制，通常在 $19\times19$ 棋盘上下六子棋。六子棋的复杂度高于迄今所有已解决的棋类，状态空间复杂度和博弈树复杂度分别为 $10^{172}$ 和 $10^{140}$，是现阶段的机器博弈较为理想的研究对象之一。本文主要讨论 $C(n,n,6,2,1)$，但方法和结论都可被其他的连珠棋所借鉴。

## 2　一般的局面表示方法

### 2.1　数组表示法

棋盘中的各交叉点有三种状态。不妨令 0 表示空(未放置棋子)，1 表示有黑子，2 表示有白子。数组表示法的基本思想是：以交叉点对应的数组索引值来表达物理位置，以交叉点对应的元素值表达状态(空、黑子、白子)。令 $V=\{0,1,2\}$，棋盘的第 $i$ 个交叉点的状态 $s_i\in V$。任何 $C(n,n,6,2,1)$ 的棋局都可以表示成一个 $n\times n$ 的向量：

$$\boldsymbol{S}=(s_0,s_1,s_2,\cdots,s_i,\cdots,s_{n\times n-1})\text{。}$$

采用这种表示方法时，想知道任意两个元素 $s_i$ 和 $s_j$ 是否共线，要通过 $i$ 和 $j$ 之间的数值规律来判断，数组表示法是一种原始、低效的表示方法。

### 2.2　比特棋盘表示法

最早采用比特棋盘结构[7]的是西洋跳棋程序。1960 年，前苏联的 KAISSA 国际象棋小组率先在国际象棋程序中采用比特棋盘[8]。近年来，比特棋盘已成为众多高水平博弈程序的重要性能优化手段。

在比特棋盘中，用 1 b 代表棋局中的 1 个位置。国际象棋有 12 种棋子，分别为每种棋子建立一张 64 b 的比特棋盘。此外，还需要一些辅助的比特棋盘，如：所有黑子的比特棋盘，所有棋子的比特棋盘等。该方法至少包括以下优点：充分利用计算机位级别上的并行处理能力，主流的个人计算机都能同时处理 32 或 64 b 数据；诸如“车的所有可行着法”、“马的所有吃子着法”、“是否将军”这类问题只需要两个或两个以上的比特棋盘之间的逻辑与、异或、非运算。

六子棋不宜采用比特棋盘，原因在于：西洋跳棋、国际象棋、奥塞罗的棋盘比较小，都是 $8\times8$ 大小的，而六子棋的棋盘大得多；六子棋的着法不像西洋跳棋、象棋等棋类那么复杂。

## 3　六子棋棋子之间的基本关系

数组表示法的缺陷在于割裂了棋子之间的天然联系。注意到只有共线的六子棋棋子之间才有直接联系，提出一行(列、交叉点)内的各点之间的直接联系由“棋形”这个概念来表达。

### 3.1　棋形的基本定义

**定义 1**　棋形。由共线的空点和同色棋子所构成的最长的连续序列，称之为棋形。

**定义 2**　棋形的长度。棋形 $p$ 中交叉点的个数称为棋形的长度，记为 $|p|$。根据规则，$|p|<k$ 时，$p$ 是没用的。以后限定：称序列 $p$ 是一个棋形，必有 $k\leqslant|p|$。

**定义 3**　棋形的颜色。包含黑子(白子)的棋形，称为黑色(白色)棋形。没有任何棋子的棋形，称为空棋形。空棋形既可能是黑色棋形，也可能是白色棋形，甚至两者都是，要依据具体棋局才能判断。

如图 1 所示，将 19 个交叉点从 0 到 18 编号。图中存在且仅存在 3 个棋形：0～10 的黑色棋形；12～18 的黑色棋形；8～18 的白色棋形。0～7，4～10 等所表示的连续序列都不是黑色棋形，因为它们都不符合“最长的连续序列”的约束；基于同样原因，也不能认为 12～18 是白色棋形。

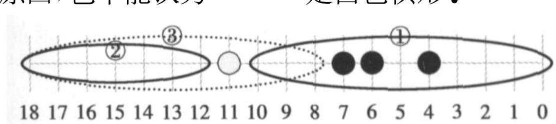


**图 1　棋形举例**

Fig. 1　Patterns

本文给出的棋形概念至少有三个优点：使得划分双方影响范围有了无歧义的依据，即，哪些空点增强黑方削弱白方，哪些空点增强白方削弱黑方；以棋形为基本单位进行表示、存储和分析，避免了冗余、低效的棋局信息存储模式；易于采用增量的方法，从而降低状态更新的代价。

### 3.2　抽象的棋形表示

比特棋盘法中，比特级别的并行计算，紧凑的数据表示思想非常值得借鉴。这里，让棋盘上每个交叉点对应一个二进制位，有子用 1 表示，无子用 0 表示。三个棋形的二进制形式如图 2 所示，等价的 10 进制形式分别为：208，0，8。

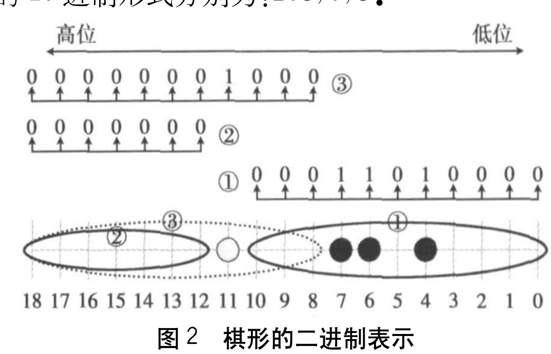


**图 2　棋形的二进制表示**

Fig. 2　Binary representation of patterns

**定理 1**　$n\in \mathbf{N}^+$ 且 $6\leqslant n$．任意的黑色棋形 $p$，$|p|=n$，集合 $Q_n=\{0,1,2,\cdots,2^n-1\}$，则

$$\phi: p\rightarrow q=\phi(p)$$

是一个从 $P$ 到 $Q_n$ 的一一映射．

显然，图 2 的映射方法就是定理 1 所描述的一个一一映射关系．图 2 的棋形有三个基本要素：颜色 color、长度 length、对应的非负整数值 id．两个棋形等价的充要条件是：二者的 color，length 和 id 都完全相等．

**定义 4**　在 $C(n,n,k,p,q)$ 中，棋形可以由三元组 $S_1=(\text{color},\text{length},\text{id})$ 表示．其中，$\text{color}\in\{\text{B},\text{W}\}$，$\text{length}\in\{k,k+1,\cdots,n\}$，$\text{id}=\phi(p)$．

根据定义 4 和定理 1，图 2 的 3 个棋形可依次表示为(B，11，208)，(B，7，0)，(W，11，8)．

**定理 2**　在 $C(n,n,k,p,q)$ 中，$n\in\mathbf{N}^+$，且 $k\leqslant n$．任意黑色棋形 $p$，$|p|\leqslant n$，$Q=\{0,1,2,\cdots,2^{n+1}-2^k-1\}$，则

$$\Phi: p\rightarrow 2^{|p|}-2^k+\phi(p)=\Phi(p)$$

是一个从 $P$ 到 $Q$ 的一一映射．其中，$\phi(p)$ 是棋形 $p$ 的二进制数值映射所对应的十进制非负整数．

定理 2 指出，任意一个定长的黑色棋形都能由一个非负整数惟一表示．

**定义 5**　棋形可以由二元组 $S_2=(\text{color},\text{id})$ 表示．其中，$\text{id}=\Phi(p)$．

根据定义 5 和定理 2，图 2 的三个棋形又可以依次表示为(B，2192)，(B，1984)，(W，1992)．

事实上，还可以将 color 信息与 id 整合到一起，但这种做法没有明显的优点．把 color 信息编码到“形状”信息，会变得不易复用．

### 3.3　具体的棋形表示

定义 4 或定义 5 所讨论的棋形是游离于具体棋局之外的棋形．上述定义侧重于通过棋形的 length 和 id 表达形状上的异同，因而，是一种抽象的表示．在具体棋局中，同时出现的两个棋形，即使形状，甚至颜色完全相同，也必须视为不同的棋形．定义 6 和定义 7 重新给出棋形的定义．图 3 中交叉点上的数字是交叉点编码．$n\times n$ 棋盘中的所有交叉点按从右向左，自底向上递增的顺序从 0 到 $n^2-1$ 进行编码．

**定义 6**　$C(n,n,k,p,q)$ 中，棋形 $p$ 由四元组 $S_3=(\text{color},\text{length},\text{id},\text{from})$ 表示．其中，棋形的编号 $\text{id}=\phi(p)$；棋形的起始交叉点 $\text{from}\in\{0,1,\cdots,n^2-1\}$ 总取编号最小的．

**定义 7**　$C(n,n,k,p,q)$ 中，棋形 $p$ 由三元组 $S_4=(\text{color},\text{id},\text{from})$ 表示．其中，棋形的编号 $\text{id}=\Phi(p)$；起点 from 同定义 6．

定义 6 和定义 7 的意义在于：不但使得局面中的具体棋形有了惟一的表示，还使得 id 相同，但颜色或起点不同的棋形仍然能够复用相同的知识．

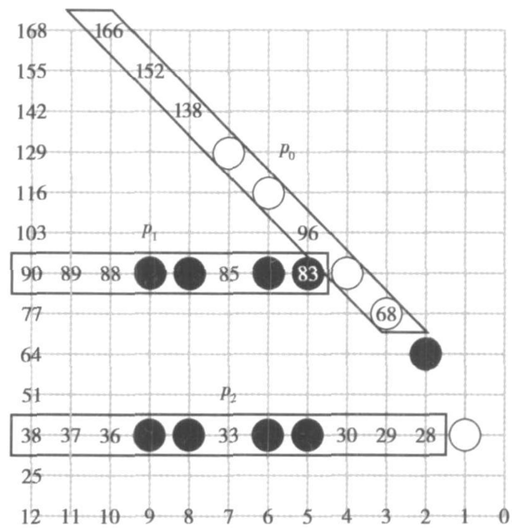


**图 3　13×13 六子棋棋局中的棋形**

**Fig.3　Patterns on the 13×13 connect 6 board**

图 3 中的三个棋形，按定义 6，可以分别表示为 $p_0=(\text{W},8,27,68)$，$p_1=(\text{B},8,27,83)$，$p_2=(\text{B},11,216,28)$．按定义 7，分别表示为 $p_0=(\text{W},215,68)$，$p_1=(\text{B},215,83)$，$p_2=(\text{B},2220,28)$．

### 3.4　棋形知识库

全部棋形的知识若能事先保存，且能方便地访问，那么，必将极大提高程序的效率和水平．

由定理 2 和定义 5 知，在 $C(n,n,k,p,q)$ 中，可能的全部黑色棋形共 $2^{n+1}-2^k-1$ 个．因为 id(按定义 7)相同的黑色棋形和白色棋形复用相同的“形状”知识，只要分配相同的知识库入口即可．假设存储每个棋形的知识需要相同的空间，设为 1 个单位，表 1 给出了六子棋和五子棋所需知识库的大小．

**表 1　几种连珠棋知识库的大小**

**Table 1　Size of knowledge database in *k*-in-a-row games**　　kb

| 19×19 六子棋 | 15×15 六子棋 | 13×13 六子棋 | 15×15 五子棋 |
|---|---|---|---|
| 999.94 | 63.93 | 15.94 | 63.97 |

从表 1 不难看出，连珠棋知识库所需的内存空间是可接受的．至于在知识库中保存何种知识，则由程序设计者的领域知识水平，以及设计者对时间和空间代价的可接受程度决定．建立棋形知识库至少有两个优点：将在线计算转化成离线计算；有利于知识的复用．

# 4　基于棋形的棋盘表示法

## 4.1　基于棋形的棋盘表示

对于 $C(n,n,k,p,q)$，容易得到下述结论。

1) 存在 $6n-4k+2$ 个可容纳棋形的线，即，$n$ 个行，$n$ 个列，$2n-2k+1$ 个 45°对角线，$2n-2k+1$ 个 135°对角线。

2) 任一条线上，最多可容纳 $\lfloor (n+1)/(k+1) \rfloor$ 个黑色棋形和 $\lfloor (n+1)/(k+1) \rfloor$ 个白色棋形。

令 $L$ 表示一行中的所有棋形：

$$L=(b_0, b_1, \cdots, b_i, \cdots, b_{\lfloor (n+1)/(k+1) \rfloor -1}, \\ w_0, w_1, \cdots, w_i, \cdots, w_{\lfloor (n+1)/(k+1) \rfloor -1}).$$

式中：$b_i$ 和 $w_i$ 分别代表一条线上的第 $i$ 个黑色棋形和第 $i$ 个白色棋形。编号 $i$ 意味着，它是从行的起始点(交叉点编号最小的点)开始的第 $i$ 个该颜色棋形。当实际的黑色棋形个数 $n < \lfloor (n+1)/(k+1) \rfloor$ 时，令 $b_n = \mathrm{NULL}$，$b_{(n+1)} = \mathrm{NULL}$，…，$b_{\lfloor (n+1)/(k+1) \rfloor -1} = \mathrm{NULL}$，用以表明这几个棋形不存在；同理，对白色棋形也有类似解释。

**定义 8**　$C(n,n,k,p,q)$ 中的一个局面 $Q$，可以表示为

$$Q=(L_0, L_1, \cdots, L_i, \cdots, L_{6n-4k+1}).$$

式中：$L_i$ 代表第 $i$ 条线。

## 4.2　建立棋形与交叉点之间的关联

尽管棋形体现了棋子间的内在联系，但是，棋盘状态的演化却是受交替著子驱动的。也就是说，选出最佳着法才是棋局分析和博弈树搜索的最终目的；因此，必须保持棋形和点之间的关联关系，并且，相关的信息能自由地相互转换。

完成上述目标，至少下述操作是必要的：接受任意一个交叉点编号，查询出经过它的(最多四条)全部的线的编号；接受任意一个以定义 6 的形式给出的棋形，求出棋形内各交叉点的编号。

为了加速转换，还可以增加一些辅助的数据表，或保存中间结果的数据结构。

# 5　增量更新

在博弈树搜索过程中，局面状态的更新、恢复是最为频繁的操作之一。棋局状态的变化，是由新增的或拿走的棋子引起的。在局面中只有该棋子所在的 4 个方向的线上的棋形会发生变化，其他的棋形都不受任何影响。一般地，当博弈树中因状态变化而执行局面状态更新操作时，只要更新定义 8 所描述的数据结构即可，但由于一些相关辅助数据结构的存在，还必须把它们也同步更新到一致的状态上。

由于采用了适宜增量更新的数据结构，程序的其他组成部分，如静态估值和搜索相关的模块也可以采用增量更新的方法，从而全面提高效率。

# 6　结　　论

本文提出了一种新的六子棋数据表示方法，它能很好地应用在六子棋等连珠棋类中。该方法的关键是：数据结构依赖于棋子间内在的联系——棋形，而不是依赖于分散的单个棋子。它的优点在于：在程序执行过程中始终维护棋子之间的全部基本关系，棋形建立了局面的数据表示和领域知识的紧密联系；棋形知识库将在线计算转化成离线计算，还能够复用棋形知识，从而进一步提高效率；增量的方法降低了棋局状态更新的代价。本文的方法和结论可扩展到 $C(n,n,k,p,q)$ 所表达的一族连珠棋类。

**参考文献：**


[1]　徐心和，王骄。中国象棋计算机博弈关键技术分析[J]。小型微型计算机系统，2006，27(6)：961－969。
(Xu Xin-he，Wang Jiao. Key technologies analysis of Chinese chess computer game [J]. *Journal of Chinese Computer Systems*，2006，27(6)：961－969.)

[2]　van den Herik J，Uiterwijk J，van Rijswijck J. Games solved：now and in the future[J]. *Artificial Intelligence*，2002，134(1/2)：277－311.

[3]　Allis L V. Searching for solutions in games and artificial intelligence[D]. Maastricht：University of Limburg，1994.

[4]　Wu I C，Huang D Y. A new family of k-in-a-row games[C]//The 11th Advances in Computer Games Conference. Taipei，2005：180－194.

[5]　Wu I C，Huang D Y，Chang H C. Connect6[J]. *ICGA Journal*，2005，28(4)：234－242.

[6]　Yu H M，Chun T S. On the fairness and complexity of generalized k-in-a-row games [J]. *Theoretical Computer Science*，2007，385：88－100.

[7]　Heinz E A. How dark thought plays chess [J]. *ICCA Journal*，1997，20(3)：167－176.

[8]　Pablo S S，Ramón G，Fernando M，*et al*. Efficient search using bitboard models[C]//International Conference on Tools with Artificial Intelligence. Washington D C：IEEE，2006：132－138.