构造、分类算法
与封闭骑士路径的枚举

*Stoyan Kapralov*加布罗沃技术大学，加布罗沃，保加利亚
s.kapralov@tugab.bg0000-0002-5713-1488

Valentin Bakoev“圣西里尔和圣美多迪乌斯”大特尔诺沃大学，大特尔诺沃，保加利亚
v.bakoev@ts.uni-vt.bg0000-0003-2503-5325

*Kaloyan Kapralov*索非亚，保加利亚
kaloyan.kapralov@gmail.com0000-0003-4971-9422

摘要。本文介绍了两种算法，用于构造所有长度不超过 16 的闭合骑士路径。本文还探讨了一种对所有此类路径进行分类（按等价关系）的方法。通过应用构造算法和分类方法，我们枚举了不受限制的骑士路径和不相交骑士路径，并展示了所得结果。

关键词：骑士图、闭合骑士路径、不相交路径、等价、枚举

1. 引言。

在图论中，骑士图是表示**国际象棋骑士**在棋盘上的所有合法走法的图。$m \times n$ 骑士图是指对应于$m \times n$ 棋盘的骑士图。开放骑士路径会恰好访问棋盘上的每个格子一次，因此它是相应骑士图中的哈密顿路径。当起点与终点重合时，这条骑士路径称为闭合骑士路径。它是该骑士图中的哈密顿回路。骑士图的类型及其性质，例如，$n \times n$ 骑士图包含 $4(n-2)(n-1)$ 条边，将在[9]中讨论。

寻找哈密顿路径和回路以及解决许多相关问题的研究由来已久——参见 [7]，以及 [8] 中发表于互联网的精彩综述。17世纪末至18世纪初，De Montmort 和 De Moivre 等数学家为经典的 8×8 棋盘上的封闭骑士路径问题提供了解法。但首位提出该问题的解法并发表相关论文的科学家，是著名的 Leonard Euler [1]。此后，Legendre、Vandermonde 等知名数学家也为这个问题提供了解法。1823年，Warnsdorff 提出的巧妙骑士路径解法尤为值得关注。为了找到解法，他建议遵循一条非常简单的规则：“从任意一个方格出发，总是将骑士移到一个未访问过、且后续可走步数尽可能少的方格。”这条规则是一个线性时间运行的贪心算法的基础 $O(n^2)$。[6] 中讨论了基于“分治”策略且具有相同时间复杂度的算法。此类方法和算法也从欧拉的构想出发进行了探讨。尽管一般情况下寻找哈密顿回路的问题是 NP完全的，但骑士图具有特殊结构

因此可以找到高效的解法。然而，找出所有彼此不等价（相对于旋转和对称变换）的解法是一个非常困难的问题。因此，直到20世纪末，这个问题才得以成功解决 [5]。

本文研究一个略有不同的问题：寻找、分类并枚举长度确定的所有哈密顿回路（即无向闭合骑士路径），包括不受限制的回路和非自交回路（将其视为多边形时）。如果一个回路可以通过对棋盘进行一次或多次平移、旋转和对称操作而得到另一个回路，则称这两个回路等价；否则称为不等价。本文对长度为 12、14 和 16 的所有不等价回路进行分类和枚举。对于更短的回路（长度为 6、8 和 10），我们已在先前的论文 [2, 3] 中解决了这一问题。

第2节概述了两种构造闭合骑士路径的算法，并通过实验比较了它们的运行时间。第3节通过一个具体示例说明等价关系——两个路径等价的含义，以及给定解满足最小性的含义。第4节展示了关于非自交回路数量的新结果。最后一节讨论了所研究问题的一些方法论方面，以及如何将其用于“算法与数据结构”“竞赛编程”等课程的教学。

2. 封闭骑士路径的构造算法

每条封闭骑士路径的长度都是偶数，记为 $k = 2l$，因为骑士每一步都会交替走到不同颜色的棋盘格上，例如黑、白、黑、白、……、黑，或者反过来：白、黑、白、黑、……、白（路径中有奇数个顶点意味着有偶数条边，即长度为偶数）。因此，这样的路径可以嵌入大小为 $(k+1) \times (k+1)$ 的棋盘中。于是我们取一个含有 $m = n = k+1$ 的 $m \times n$ 棋盘，并按行将棋盘格编号为从 1 到 $(k+1)^2$ 的整数。我们采用下文概述的两种相似方法。

在算法 1 中，对于起始单元格集合 S 中的每个 s，我们运行深度优先搜索的一种改进版本。通过穷举回溯，我们构造出所有长度为 k 的骑士路径（包含 k 个顶点和 $k-1$ 条边）。最后，我们检查最后一个棋盘格是否与起始棋盘格相邻（即能通过国际象棋骑士的一步到达）；若相邻，则找到一个新解。如果该解是字典序最小的，就将其保存——下一节将介绍最小性的定义及其检查方法。以下是算法 1 的 C++ 代码。

```
void dfs() { int u = p[len-1]; // p – 路径单元格列表 if(len < k)
{ for (int v:adj[u]) if(!vis[v]) { p[len++] = v; vis[v] =
true; dfs(); len--; vis[v] = false; }
```

```
    } else if(adjacent(u,s)) {
path pt(p,p+k); if(is_min(pt))
sol.push_back(pt); } } int
main() { ... for(s:S) { p[0] = s;
vis[s] = true; len = 1; dfs(); } ...
}
```

在算法 2 中，对于每一对起始单元格 s 和结束单元格 t，我们构造所有长度为 $k/2$ 的连接路径。然后，我们尝试将两条半程路径拼接成一条长度为 k 的完整路径。当然，如果两条路径不相交，也就是说，除两个端点外没有任何公共点，我们就可以将它们拼接起来。只需考虑无法向上或向左平移的回路，也就是说，它们至少有一个点位于棋盘的顶行，且至少有一个点位于最左列。因此，我们定义两个回路在平移（即在棋盘上滑动）意义下的等价关系。由此，出于对称性的考虑，我们也可以假设只需考虑集合 $S = \{1,2,\ldots,k/2+1\}$，将其作为可能起始点的集合。表1展示了算法 1（较慢）与算法 2（较快）的运行速度比较。比较是在一台配备第八代英特尔酷睿 i5 处理器的普通笔记本电脑上进行的。程序使用 C++ 编写。

k	数量 不等价环	算法 1	算法 2
6	25	< 1 个	< 1 个
8	480	< 1 秒	< 1 秒
10	12000	2 s	1 s
12	350256	44 s	18 s
14	10780549	2097 秒	692 秒
16	344680960	???	≈ 24 小时

表1。 两种算法所得到的不等价的闭合骑士路径数量及经过时间（秒）

3. 等价解

我们来看看表 2 中的图形。它们都是从左上角的主图形出发，通过每次旋转 90°（逆时针）并沿垂直轴进行对称变换得到的。我们称这 8 个图形彼此等价。我们的任务是枚举并分类给定长度的闭合骑士路径；从本质上说，就是在等价意义下枚举某一组图形，也就是说，从每个等价类中选出一个代表。

为剪除不必要的解，我们采用以下方法，表 2 以长度为 8 的路径为例进行了说明。棋盘上的格子按行编号为 1、2、……、$(k+1)^2$。在本例中，棋盘格按行从 1 到 25 编号。然后，每个图形都用一个由 8 个棋盘格编号组成的序列表示。每个几何图形都对应 16 个不同的序列（8 种起点选择方式，以及两种图形行进方向：顺时针或逆时针）。显然，对于表 2 中的 8 个图形，每个图形都选取这 16 个序列中字典序最小的一个作为其表示。左上角的图形是最小的，因为表示它的序列在字典序上小于或等于其他 7 个图形的表示序列。其余 7 个图形都不是最小的，因为经过有限次旋转和对称变换，每个图形都可以变换为一个表示序列字典序更小的图形。

使用算法 1 或算法 2 生成图形时，如果得到的图形不是极小的，就不予考虑，因为它与对应的极小图形等价。

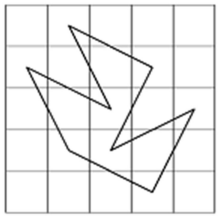
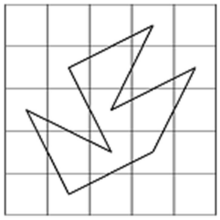
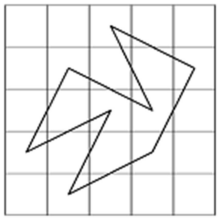
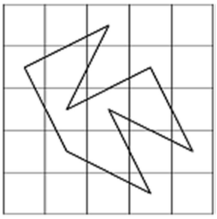
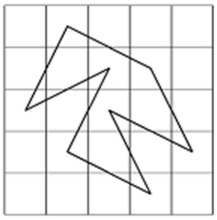
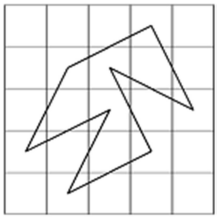
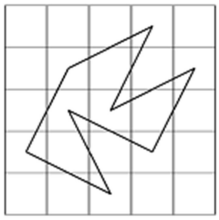
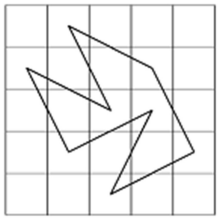
 **2,9,18,15,24,17,6,13**	 **4,7,18,11,22,19,10,13**	 **3,10,19,22,13,16,7,14**	 **3,6,17,24,13,20,9**
 **2,9,20,13,24,17,8,11**	 **4,7,16,13,22,19,8,15**	 **4,7,16,23,12,19,10,13**	 **2,9,20,23,14,17,6,13**

表 2。 8 个等价解

这样一来，就不必检查新构造的解是否与已经得到的数千个解中的任何一个等价，只需检查它是否为极小解即可。

在[3] 中，我们开始区分访问过相同棋盘格集合、但对应多边形不全等的闭合骑士路径。我们将这类路径称为几何上不同的闭合骑士路径。第一个此类情况出现在长度为$k = 8$ 时，如表3所示。

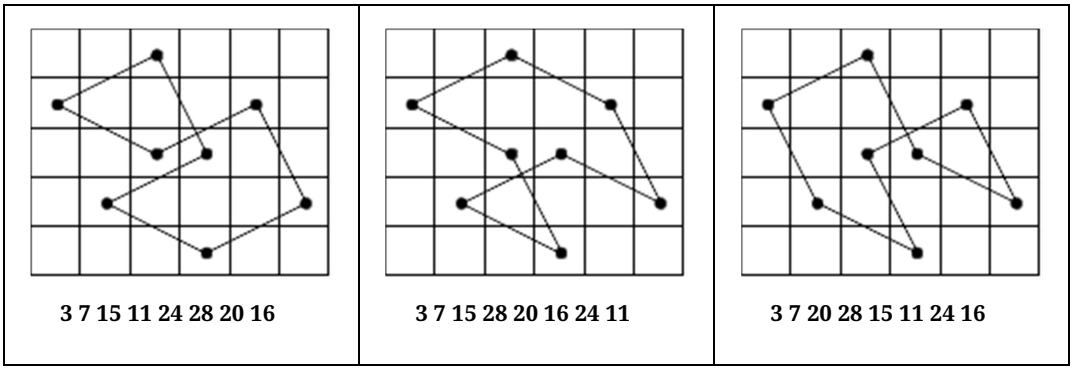


表3。长度为8的三条几何上不同的闭合骑士路径

4. 新结果

与在 $m \times n$棋盘上寻找所有马步路径（开路径或闭路径）相关的一个重要问题，是寻找最大长度为 [4, 8] 的无交叉（即非自交）马步路径。T. R. Dowson 于1930年提出了这个问题，并给出了在 8×8 棋盘上的两条长度为35的解；后来他声称（但未提供证明）这两条路径均为最大长度 [4]。有关该问题各种变体的解法及当前结果，请参见网站 [8]。

通过对算法进行适当修改，我们枚举出了彼此不等价且几何上不同的闭合骑士路径，以及其中所有不自相交的路径。表 4 列出了目前已知的所有长度为 k 的闭合骑士路径数量，以及不自相交闭合路径的数量。新结果以“*”标记。此前已知的结果来自 [2] 和 [3]。

长度 k	数量 不等价环	数量 非自交环
4	3	3
6	25 [2]	13*
8	480 [3]	178*
10	12000 [3]	3034*
12	350256*	64877*
14	10780549*	1503790*
16	344680960*	36930111*

表 4。新结果

5. 结论

迄今为止，所讨论的问题一直作为研究问题提出。它的灵感源于第一作者在 2015年全国春季信息学竞赛 [10] 中提出的一项作业。众所周知，骑士漫步问题和八皇后问题是研究递归与回溯的经典例子。因此，它们理应在相关书籍或教材中占有一席之地。在 $m \times n$ 棋盘上寻找所有开放或闭合骑士路径的一般问题及相关问题具有教学意义。这个一般问题为“程序设计”“算法与数据结构”“图算法”“竞赛编程”等课程的教学提供了许多可能性和相关示例。它还可以用于讲授递归、穷举搜索与回溯、深度优先搜索、贪心算法等主题。对于这些课程中的每一门，都可以将骑士路径或回路作为讲座、实验、作业和项目的主题。它们的难度可以分为不同层次：

- 初级——只搜索一条骑士路径或一个回路，并在不同尺寸的方形和长方形棋盘上进行实验；
- 中级——搜索并枚举所有骑士路径/回路，检验某些尺寸的棋盘上是否不存在回路，搜索半幻方骑士路径/回路（详情见 [8]）等；
- 高级——学习并将一些等价关系编写成计算机程序加以应用，同时排除等价解；这些内容可加入某些中级问题；
- 挑战——检验此处呈现的一些结果，和/或得出新结果。

最后，我们指出，骑士路径问题仍在不断发展，这主要得益于最新处理器的强大计算能力。G. Jelliss 的网站展示了骑士路径问题各种类型的许多近期解答，这些解答所用的$m \times n$ 棋盘尺寸还在不断增大[8]。

致谢

本研究部分获得加布罗沃技术大学第2209E/2022号资助，以及大特尔诺沃大学项目 FSD-31-243-23/21.03.2023的资助。

参考文献

[1] 欧拉 L.，《一个看似不属于任何分析范畴的有趣问题的解答》，皇家科学院与美文学院回忆录，1759年，第15卷，第310–337页，柏林，1766年。http://euler.free.fr/knight/index2.html，访问日期：30/03/2023。

[2] Kapralov S.、Bakoev V.、Kapralov K.，《若干封闭骑士路径的枚举》，UNITECH-2017国际科学会议论文集，2017年11月17–18日，ISSN 1313-230X。加布罗沃技术大学，保加利亚加布罗沃，II416–II418； arXiv:1711.06792。

[3] Kapralov S., Bakoev V., Kapralov K.，《长度为10的封闭骑士路径枚举》，第15届信息学与信息技术国际会议，CIIT 2018，2018年4月20日至22日，马夫罗沃，马其顿，76–78。

[4] Knuth D.，《回溯程序效率估算》，计算数学，第29卷，第129期，1975年1月，121–136。[5] McKay B. D.，《棋盘的 8 × 8 骑士巡游》。技术报告 TR-CS-97-03，澳大利亚国立大学计算机科学系，澳大利亚，1997年2月。[6] Parberry I.，《骑士巡游问题的高效算法》，离散应用数学，73，1997年，251–260。[7] Rouse Ball W. W.，《数学娱乐与随笔》，古腾堡计划，2008 [电子书 #26839]（第1版，1892年2月；1892年5月再版；第2版，1896年；1905年再版）。[8] Jelliss G.，《骑士巡游笔记》，网址：http://www.mayhematics.com/t/t.htm，访问日期：2023年3月30日。[9] 骑士图，http://mathworld.wolfram.com/KnightGraph.html。访问日期：2023年3月30日。[10] 全国中学生信息学竞赛（保加利亚语），http://www.math.bas.bg/infos/files/2015-06-14-B3.pdf。访问日期：2023年3月30日。