Department of Communications and Networking

Graph Algorithms for Constructing and Enumerating Cycles and Related Structures

Ville Pettersson

Aalto University

DOCTORAL DISSERTATIONS

Aalto University publication series
DOCTORAL DISSERTATIONS 127/2015

Graph Algorithms for Constructing and Enumerating Cycles and Related Structures

Ville Pettersson

A doctoral dissertation completed for the degree of Doctor of Science (Technology) to be defended, with the permission of the Aalto University School of Electrical Engineering, at a public examination held at the lecture hall S3 of the school on 2 October 2015 at 12.

Aalto University
School of Electrical Engineering
Department of Communications and Networking
Information Theory

Supervising professor
Patric Östergård

Thesis advisor
Patric Östergård

Preliminary examiners
William Kocay, University of Manitoba, Canada
Wendy Myrvold, University of Victoria, Canada

Opponent
Gary McGuire, University College Dublin, Ireland

Aalto University publication series
DOCTORAL DISSERTATIONS 127/2015

ISBN 978-952-60-6364-5 (printed)
ISBN 978-952-60-6365-2 (pdf)
ISSN-L 1799-4934
ISSN 1799-4934 (printed)
ISSN 1799-4942 (pdf)
http://urn.fi/URN:ISBN:978-952-60-6365-2

Unigrafia Oy
Helsinki 2015

Finland

441 697
Printed matter

Aalto University

Abstract

Aalto University, P.O. Box 11000, FI-00076 Aalto www.aalto.fi

Author
Ville Pettersson

Name of the doctoral dissertation
Graph Algorithms for Constructing and Enumerating Cycles and Related Structures

Publisher School of Electrical Engineering

Unit Department of Communications and Networking

Series Aalto University publication series DOCTORAL DISSERTATIONS 127/2015

Field of research Information Theory

Manuscript submitted 30 April 2015 **Date of the defence** 2 October 2015

Permission to publish granted (date) 31 July 2015 **Language** English

☐ **Monograph** ☒ **Article dissertation (summary + original articles)**

Abstract

A graph is a mathematical object that consists of a set of vertices and a set of edges between some pairs of vertices. Despite the simple definition graphs are very versatile; in addition to being of theoretical interest, they have a large number of applications in several areas of science and technology, including physics, biology, and telecommunications.

This thesis focuses on algorithms for constructing and enumerating cycles and some related structures in graphs. Cycles are central objects in graph theory, and they appear in a multitude of applications. For many applications it is essential to construct or enumerate cycles with certain properties in a graph. The algorithms created here are based on canonical augmentation, dynamic programming, and recursive search, and they make heavy use of the symmetries of the graphs.

The contribution of this thesis is as follows. We present a new method for enumerating Hamiltonian cycles in general graphs, planar graphs and grid graphs. We enumerate perfect matchings in the n-cube for $n \leq 7$. We discover new chordless cycles and chordless paths in the 8-cube and establish that they are of maximal length. We discover new small planar hypohamiltonian graphs of order 40; the smallest previously known examples were of order 42. Furthermore, enumeration of certain cycles in the grid graph is used for studying Toeplitz' conjecture.

Keywords Graphs, Algorithms, Construction, Enumeration, Cycles

ISBN (printed) 978-952-60-6364-5	**ISBN (pdf)** 978-952-60-6365-2	
ISSN-L 1799-4934	**ISSN (printed)** 1799-4934	**ISSN (pdf)** 1799-4942
Location of publisher Helsinki	**Location of printing** Helsinki	**Year** 2015
Pages 143	**urn** http://urn.fi/URN:ISBN:978-952-60-6365-2	

A! Aalto-yliopisto

Tiivistelmä

Aalto-yliopisto, PL 11000, 00076 Aalto www.aalto.fi

Tekijä
Ville Pettersson

Väitöskirjan nimi
Graafialgoritmeja syklien ja niihin liittyvien rakenteiden konstruointiin ja enumerointiin

Julkaisija Sähkötekniikan korkeakoulu

Yksikkö Tietoliikenne- ja tietoverkkotekniikan laitos

Sarja Aalto University publication series DOCTORAL DISSERTATIONS 127/2015

Tutkimusala Informaatioteoria

Käsikirjoituksen pvm 30.04.2015 **Väitöspäivä** 02.10.2015

Julkaisuluvan myöntämispäivä 31.07.2015 **Kieli** Englanti

☐ **Monografia** ☒ **Yhdistelmäväitöskirja (yhteenveto-osa + erillisartikkelit)**

Tiivistelmä

Graafi on matemaattinen objekti joka koostuu solmuista ja solmujen välillä olevista kaarista. Yksinkertaisesta määritelmästä huolimatta graafit ovat hyvin monipuolisia; sen lisäksi että graafit ovat kiinnostavia teoreettisesta näkökulmasta, niillä on suuri määrä sovelluksia monella tieteen ja teknologian alalla, kuten fysiikassa, biologiassa ja tietoliikennetekniikassa.

Tässä väitöskirjassa tutkitaan algoritmeja, joita voidaan käyttää syklien ja eräiden sykleihin liittyvien rakenteiden konstruoimiseen ja lukumäärän laskemiseen graafeissa. Syklit ovat keskeisiä objekteja graafiteoriassa, ja niitä esiintyy lukuisissa sovelluksissa. Monien sovellusten kannalta on tärkeää konstruoida tai laskea sellaisia graafeissa olevia syklejä joilla on joitain tiettyjä ominaisuuksia. Väitöskirjatyössä luodut algoritmit perustuvat kanoniseen augmentointiin, dynaamiseen ohjelmointiin ja rekursiiviseen hakuun, ja ne käyttävät hyväkseen tutkittavien graafien symmetriaa.

Väitöskirjan kontribuutio on seuraavanlainen. Esitämme uuden menetelmän Hamiltonin syklien lukumäärän laskemiseen yleisissä graafeissa, tasograafeissa ja hilagraafeissa. Laskemme täydellisten paritusten lukumärän n-kuutiossa arvoon $n = 7$ asti. Löydämme uusia indusoituja syklejä ja polkuja 8-kuutiossa ja todistamme että näiden pituus on suurin mahdollinen. Löydämme uusia pieniä hypohamiltonisia tasograafeja, joissa on vain 40 solmua; pienin aiemmin tunnettu esimerkki sisälsi 42 solmua. Lopuksi tutkimme Toeplitzin konjektuuria laskemalla tietynlaisten syklien lukumäärän hilagraafissa.

Avainsanat Graafit, algoritmit, syklit, konstruointi, enumerointi

ISBN (painettu) 978-952-60-6364-5 **ISBN (pdf)** 978-952-60-6365-2

ISSN-L 1799-4934 **ISSN (painettu)** 1799-4934 **ISSN (pdf)** 1799-4942

Julkaisupaikka Helsinki **Painopaikka** Helsinki **Vuosi** 2015

Sivumäärä 143 **urn** http://urn.fi/URN:ISBN:978-952-60-6365-2

Preface

The research work for this thesis was conducted between September 2011 and April 2015 in the premises of Aalto University, in the School of Electrical engineering, Department of Communications and Networking.

The research was funded by the Academy of Finland under Grant No. 132122, the GETA Graduate School, the Nokia Foundation, and the Finnish Foundation for Technology Promotion.

I would like to thank my supervisor Patric R. J. Östergård for his guidance in the research work and writing related to this thesis. I would also like to thank the pre-examiners William Kocay and Wendy Myrvold, whose comments and suggestions helped to improve this thesis. Furthermore, I would like to thank my co-authors Mohammadreza Jooyandeh, Brendan McKay, Helge Tverberg, and Carol Zamfirescu for their contributions, and also colleagues, family and friends.

Espoo, September 1, 2015,

Ville Pettersson

Contents

List of Publications

This thesis consists of an overview and of the following publications which are referred to in the text by their Roman numerals.

I M. Jooyandeh, B. D. McKay, P. R. J. Östergård, V. H. Pettersson, C. T. Zamfirescu. Planar Hypohamiltonian Graphs on 40 Vertices. Submitted to *Journal of Graph Theory*, 2014.

II P. R. J. Östergård, V. H. Pettersson. Enumerating Perfect Matchings in n-Cubes. *Order*, **30**, 821–835, 2013.

III P. R. J. Östergård, V. H. Pettersson. Exhaustive Search for Snake-in-the-Box Codes. Accepted for publication in *Graphs and Combinatorics*, 2014.

IV P. R. J. Östergård, V. H. Pettersson. On the Maximum Length of Coil-in-the-Box Codes in Dimension 8. *Discrete Applied Mathematics*, **179**, 193–200, 2014.

V V. H. Pettersson. Enumerating Hamiltonian Cycles. *The Electronic Journal of Combinatorics*, **21**(4), #P4.7, 2014.

VI V. H. Pettersson, H. A. Tverberg, P. R. J. Östergård. A note on Toeplitz' conjecture. *Discrete and Computational Geometry*, **51**, 722–728, 2014.

Author's Contribution

The author is the main author of [III], [IV], and [V], and took part in writing [I], [II], and [VI]. The author designed and implemented the methods in [III],[IV], and [V], and designed and implemented the computational part of [VI]. In [II] the author designed the general approach together with the co-author, designed some specific details of the method, and implemented the method. In [I] the author designed and implemented the testing for hypohamiltonicity and proved Theorems 2.3 and 4.6.

List of Symbols

$\mathrm{Aut}(G)$	The automorphism group of a graph G
$c(n)$	The length of a longest chordless cycle in the n-cube
$d_H(v_0, v_1)$	The Hamming distance between codewords v_0 and v_1
E	A set of edges
$E(G)$	The set of edges in a graph G
$f(n)$	The number of perfect matchings in the n-cube
F	A field.
$F(n)$	The number of equivalence classes of perfect matchings in the n-cube
F_n	The nth Fibonacci number
F^n	The n-dimensional vector space over a field F
$\mathbb{F}_q$	The finite field of order q
g	A group element
G	A graph
$H_{m,n}$	The number of Hamiltonian cycles in the $(m+1) \times (n+1)$ grid graph
K_n	The complete graph of order n
$M(G)$	The adjacency matrix of graph G
$\mathbb{N}$	The set of natural numbers
$\mathcal{O}$	Big O notation
$p(G)$	The parent of a graph G in canonical augmentation
$\mathrm{per}(M)$	The permanent of a matrix M
P_n	The path of length n
Q_n	The n-dimensional hypercube, or n-cube
$\mathbb{R}$	The set of real numbers
$\mathbb{R}^2$	The Euclidean plane
$s(n)$	The length of a longest chordless path in the n-cube

V	A set of vertices
$V(G)$	The set of vertices in a graph G
$w(G)$	The canonical parent of a graph G in canonical augmentation
$X \cong Y$	The object X is isomorphic to Y
$\mathbb{Z}$	The set of integers
Γ_n	The automorphism group of the n-cube
ϵ	A small positive real valued number

1. Introduction

A graph is a mathematical object consisting of a set of vertices and a set of edges between some pairs of vertices. Despite the simple definition graphs are very versatile; in addition to being of theoretical interest, they have a large number of applications in several areas of science and technology, including physics [37], biology [85], and telecommunications [39]. An introduction to graphs and their many properties can be found in [78]. Figure 1.1 shows an example of a graph with 8 vertices and 14 edges.

A cycle in a graph is a sequence of vertices that starts and ends at the same vertex, with an edge between each two consecutive vertices, and has no repeated vertices other than the starting vertex. Cycles are central objects in graph theory and they appear in a multitude of applications. For many applications it is essential to determine the number of cycles with certain properties in a graph. For example, the number of Hamiltonian cycles in a graph can help understand the structural properties of proteins [37]. In addition to determining the number of cycles with certain properties, many applications benefit from explicitly constructing the cycles. For example, a chordless cycle in a hypercube graph can be used as an error-detecting code [39].

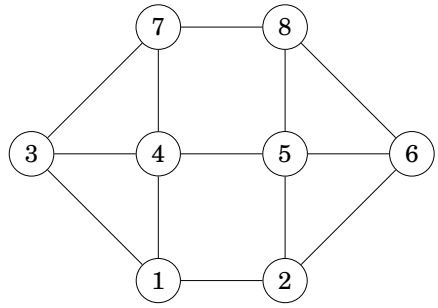

Figure 1.1. A graph

In this thesis we will discuss enumeration and construction of cycles and some related structures. The main focus is on chordless cycles and Hamiltonian cycles. We will also discuss enumeration and construction of three closely related structures, namely Hamiltonian paths, perfect matchings and hypohamiltonian graphs.

Enumeration and construction of cycles and other graph structures fall under the field of combinatorics [30], and they can be accomplished by using analytical or computational methods. Examples of analytical results on graph enumeration include counts of labeled trees on n vertices [9] and caterpillars on n vertices [62], the numbers being n^{n-2} and $2^{n-4} + 2^{\lfloor n/2-2 \rfloor}$ respectively. Analytical results are desirable, but in many cases they are difficult to obtain. For the difficult cases, computational methods can be used. Sometimes the results given by computational methods can lead to discoveries that allow one to find an analytical solution to the enumeration problem or other insights on the graph structure being studied.

Several algorithms exist for constructing and enumerating cycles and related structures. Some of these algorithms explicitly construct each object [75], while others count the number of objects [58] without explicitly constructing them. For many graphs the number of objects is too large for explicit construction, in which case only enumeration is feasible.

For chordless cycles [75], chordless paths [75], and perfect matchings in bipartite graphs [74], methods have been published that are able to construct each object within a given graph in linear time with respect to the size of the output. For these graph structures the remaining interesting research questions include counting the objects when the number is too large for explicit construction, or constructing a small subset of objects with some interesting properties. Examples of such properties include extremality and symmetry.

For Hamiltonian cycles and planar hypohamiltonian graphs, determining the hamiltonicity or hypohamiltonicity of a given graph is NP-complete [27, pp. 199–200]. No polynomial method is known (nor likely to exist) for listing all Hamiltonian cycles in a given graph or listing all planar hypohamiltonian graphs. Thus, for these graph structures even small improvements in results concerning enumeration and construction can be difficult to obtain and may require novel approaches.

1.1 Contribution

In this thesis we study graph algorithms for enumerating and constructing cycles and some related graph structures. The full list of structures studied consists of Hamiltonian cycles, chordless cycles, hypohamiltonian planar graphs, perfect matchings, and chordless paths.

As for enumeration, we study enumeration of perfect matchings in bipartite graphs with the main focus being on the case of the n-cube, and enumeration of Hamiltonian cycles, with the main focus being on grid graphs and some other graph families with a grid-like structure. Enumerating perfect matchings in the n-cube is considered to be one of the main open problems in enumeration of matchings [58, Problem 14].

As for construction, we study exhaustive construction of certain extremal (long) chordless cycles and chordless paths in the n-cube, exhaustive construction of certain extremal chordless cycles in the grid graph in an attempt to validate the Toeplitz' conjecture, and nonexhaustive construction of small planar hypohamiltonian graphs.

The contribution of this thesis is a collection of new algorithms for enumerating and constructing the aforementioned structures. These algorithms are presented in articles [I] to [VI]. The algorithms we present in [I] to [VI] are based on the principles of dynamic programming [14, Chap. 15], canonical augmentation [48], and backtracking [38, Sect. 4.1.2], and they make heavy use of symmetry.

We present a dynamic programming method for enumerating perfect matchings in bipartite graphs, and use it to enumerate labelled and unlabelled perfect matchings in the n-cube for $n \leq 7$ in [II]. Previously the number of labelled and unlabelled perfect matchings was known for $n \leq 6$ and $n \leq 5$, respectively. We create a discrete version of the Toeplitz' conjecture in the $n \times n$ grid graph, and verify it computationally for $n \leq 13$ with a backtracking method by means of generating certain extremal chordless cycles in the grid graph in [VI]. We present a method based on dynamic programming for enumerating Hamiltonian cycles in arbitrary graphs, and apply it to $n \times n$ grid graphs for $n \leq 26$, triangular grids with n vertices on one side for $n \leq 20$, $n \times n$ king's graphs for $n \leq 16$, three-dimensional $n \times m \times k$ grid graphs for $n, m, k \leq 5$ in [V]. We establish that the maximum length of an n-snake for $n = 8$ is 98 edges with a method based on canonical augmentation in [III]. We establish that the maximum length of an n-coil for $n = 8$ is 96 edges with a method based on canonical

augmentation in [IV]. We find planar hypohamiltonian graphs of order 40 by generating a family of planar gaphs with certain properties in [I].

1.2 Structure of this Thesis

The structure of this thesis is as follows. In Chapter 2 we give some preliminaries regarding graphs and symmetries, and in Chapter 3 we give some background on the algorithms and frameworks that we use. In Chapters 4 and 5 we discuss earlier work on each of the research problems studied in this thesis and how it connects to the research done in [I] to [VI]. Chapter 4 focuses on construction problems [I,III,IV,VI] and Chapter 5 focuses on enumeration problems [II,V]. The thesis is concluded in Chapter 6.

2. Basic Concepts

In this chapter we present some necessary background information on graphs and graph symmetries. For a more detailed introduction to graphs and their properties, see [38, 78].

2.1 Graphs

Here we give a formal definition of a graph and some related structures.

Definition 1. A *graph* G is an ordered pair $G = (V, E)$ where V is a finite set of vertices and $E \subseteq \{\{a, b\} : a, b \in V, a \neq b\}$ is the set of edges.

Figure 1.1 in Chapter 1 shows an example of a graph. In the following we denote the set of vertices in G by $V(G)$, and the set of edges by $E(G)$.

Definition 2. A graph G' is a *subgraph* of G if $V(G') \subseteq V(G)$, $E(G') \subseteq E(G)$, and $a, b \in V(G')$ for any $\{a, b\} \in E(G')$.

Definition 3. The *adjacency matrix* $M(G) = (m_{ij})$ of a graph G with n vertices is an $n \times n$ matrix having values

$$m_{i,j} = \begin{cases} 1 & \text{if there is an edge between vertices } v_i, v_j, \\ 0 & \text{otherwise,} \end{cases}$$

where the vertices of G are $\{v_1, v_2, \ldots, v_n\}$.

Definition 4. A *path* in a graph $G = (V, E)$ is a sequence of vertices $v_1 v_2 \cdots v_n$ such that no vertex appears more than once in the sequence, and $\{v_i, v_{i+1}\} \in E$ for all $1 \leq i < n$.

Definition 5. A *cycle* in a graph G is a sequence of vertices $v_1 v_2 \cdots v_n v_1$ such that no vertex other than v_1 appears more than once, $\{v_i, v_{i+1}\} \in E$ for all $1 \leq i < n$, and $\{v_n, v_1\} \in E$.

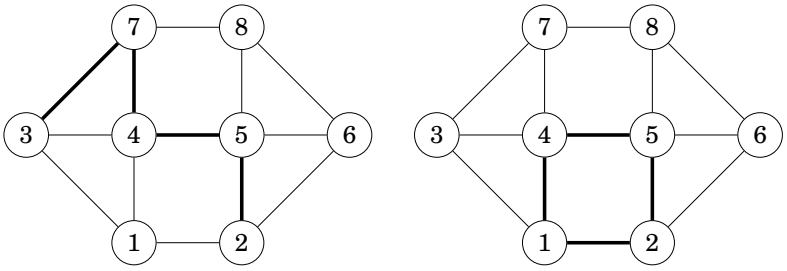

Figure 2.1. A path and a cycle

Figure 2.1 shows a path and a cycle in a graph. In Definitions 4 and 5 we defined paths and cycles as sequences of vertices; it would also be possible to define them as subgraphs of G.

Definition 6. A *Hamiltonian cycle* in a graph G is a cycle that visits each vertex exactly once.

A Hamiltonian path is defined analogously.

Definition 7. A *Hamiltonian path* in a graph G is a path that visits each vertex exactly once.

Definition 8. For a graph G, an edge $e = \{a, b\} \in E(G)$, and a vertex $v \in V(G)$, e and v are *incident* if $a = v$ or $b = v$.

Definition 9. A *perfect matching* in a graph is a subset of the edges such that each vertex is incident to exactly one edge in the subset.

Definition 10. A *k-factor* of a graph G is a k-regular subgraph of G that contains every vertex of G.

Note that a perfect matching always corresponds to a 1-factor. Figure 2.2 shows a perfect matching and a Hamiltonian cycle in a graph. Removing every other edge from a Hamiltonian cycle of even length always results in a perfect matching. The converse does not hold; a perfect matching cannot always be extended to a Hamiltonian cycle. However, in some graphs, such as the the n-cube, a perfect matching can always be extended to a Hamiltonian cycle [25].

Definition 11. A graph G is *planar* if it can be drawn in the plane without crossing edges.

Given a graph G, we denote by $G - v$ the graph obtained by removing the vertex v and all edges incident to it.

Definition 12. A graph G is *hypohamiltonian* if it is not Hamiltonian but each subgraph $G - v$ where $v \in V(G)$ is Hamiltonian.

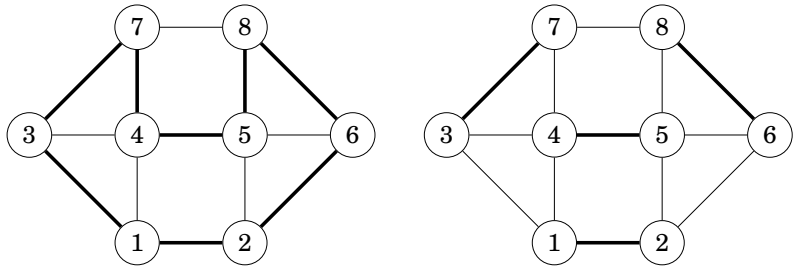

Figure 2.2. A Hamiltonian cycle and a perfect matching

Publication [I] contains several examples of planar hypohamiltonian graphs.

Definition 13. The *distance* between two vertices a and b in a graph G is the length of a shortest path from a to b in G.

Definition 14. A path in G is *chordless* if every two distinct vertices in the path that are not successive have a distance of at least 2 in G.

Definition 15. A cycle in G is *chordless* if every two distinct vertices in the cycle that are not successive have a distance of at least 2 in G.

Chordless paths and chordless cycles are also called induced paths and induced cycles.

Definition 16. In a graph G, two vertices $v, w \in V(G)$ are *adjacent* if $\{v, w\} \in E(G)$.

Definition 17. A graph G is *bipartite* if its vertices can be partitioned into two disjoint sets U and V such that no two vertices within the same set are adjacent. A bipartite graph can be denoted by $G = (U, V, E)$.

Definition 18. The *biadjacency matrix* $B(G) = (b_{ij})$ of a bipartite graph $G = (U, V, E)$ is an $r \times s$ matrix having values

$$b_{i,j} = \begin{cases} 1 & \text{if there is an edge between vertices } u_i, v_j, \\ 0 & \text{otherwise,} \end{cases}$$

where $U = \{u_1, \ldots, u_r\}$ and $V = \{v_1, \ldots, v_s\}$.

2.2 Symmetry

The concept of symmetry and isomorphism [38, Chap. 3] is an important part of most algorithms for constructing and enumerating graphs. With

the help of symmetries it is possible to avoid redundant computations, often resulting in a significant speedup.

Here we give a brief outline of symmetry.

Definition 19. Graphs G and G' are *isomorphic*, if there exists a bijection $h : V(G) \to V(G')$ such that $\{a, b\} \in E(G) \Leftrightarrow \{h(a), h(b)\} \in E(G')$. Such a bijection is called an *isomorphism* from G to G'.

Intuitively, two graphs are isomorphic if one can be obtained from the other simply by renaming vertices. Isomorphic graphs have the same structural properties; after determining some structural property of G through a computation, it is no longer necessary to run the same computation for graphs isomorphic to G, since we know that they have the same properties.

Definition 20. An *automorphism* of G is an isomorphism from G to itself.

Definition 21. The *automorphism group* of G, $\mathrm{Aut}(G)$, is the set of all automorphisms of G and forms a group with composition as the group operation.

The automorphism group of a graph describes the symmetry in the graph. It is a useful concept for designing graph algorithms that enumerate or classify subgraphs, such as perfect matchings or chordless cycles.

2.3 Some Graph Families

Here we present the graph families that are most relevant to this thesis and discuss some of their properties. These are the graph families that are studied in the construction and enumeration problems discussed in Chapters 4 and 5.

The choice of these graphs is motivated by applications, and also by the fact that they can be easily parameterized. Parameterization allows one to conveniently track progress in algorithms that study the properties of these graphs; the more efficient an algorithm is, the larger the parameter values for which the graph can be studied with a limited amount of computational resources.

We begin with the the n-dimensional hypercube graph, also known as the *n-cube*. We denote by F^n the n-dimensional vector space over a field F.

Definition 22. For vectors $x, y \in F^n$, the *Hamming distance* between x and y, $d_H(x, y)$, is the number of indices i for which $x_i \neq y_i$.

The n-cube is denoted by Q_n and defined as $Q_n = (V = \mathbb{F}_2^n, E = \{\{x, y\} \subset \mathbb{F}_2^n : d_H(x, y) = 1\})$, where $\mathbb{F}_2$ is the finite field of order 2. Figure 2.3 shows the 3-cube. Chordless paths and chordless cycles in the n-cube are called n-snakes and n-coils respectively [42].

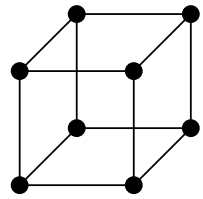

Figure 2.3. The 3-cube

The n-cube has an automorphism group Γ_n of size $2^n n!$. An automorphism of the n-cube consists of permutation of coordinates followed by addition of a constant vector.

We now proceed to define several graph families that have a grid-like structure. The $n \times m$ *grid graph* is the cartesian product of two paths of length n and m. The *three-dimensional* $n \times m \times k$ *grid graph* is the cartesian product of three paths. The $n \times n$ *king's graph* is the strong product of two paths of length n. For precise definitions of the cartesian product and the strong product, see [60]. The *triangular grid* of side length n has the vertex set $\{(i, j) : 1 \leq j \leq i \leq n\}$ and edges

$$\{\{(i, j), (i', j')\} : (i - i', j - j') \in \{(0, 1), (1, 0), (1, 1)\}\}.$$

Figure 2.4 shows the 4×4 grid graph, 4×4 king's graph, and the triangular grid with side length 4.

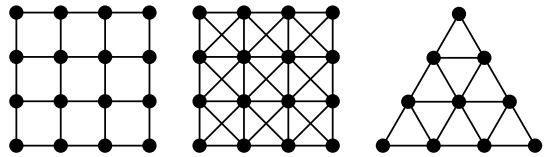

Figure 2.4. A grid graph, a king's graph, and a triangular grid

These graphs have applications in several fields. For example, the n-cube is relevant to coding theory in many ways [39], and Hamiltonian cycles in the grid graph have applications in physics [37].

3. Algorithms

In this section we give some background on three computational methods that can be used for enumerating or constructing graph structures, namely backtrack search, dynamic programming, and canonical augmentation. Backtrack search and canonical augmentation can be used for both enumerating and constructing graph structures. Dynamic programming is better suited for enumeration than construction. Compared to backtrack search, canonical augmentation is typically more laborous to implement, but it can be more efficient when the graphs that are studied have a large automorphism group.

Applying any of these methods to a specific problem requires customizing the method for the problem, and the performance depends strongly on how well this customization is done.

3.1 Backtrack Search

Backtrack search [38, Sect. 4.1.2] is a computational method that can be used for constructing all combinatorial objects that satisfy given constraints. In backtrack search we build objects by extending partial objects recursively in all possible ways. The partial objects are presented as a tuple $(a_1, a_2, \ldots, a_n)$. For each partial object we need to compute $\mathcal{A}$, the set of all possible elements a_{n+1} with which the partial object can be extended such that the result $(a_1, a_2, \ldots, a_n, a_{n+1})$ is still a valid partial object. Pseudocode for the algorithm is shown in Algorithm 1.

Initially, the backtrack search is called with $n = 0$ and an empty tuple as its parameter. The efficiency of the method depends on how well we are able to prune the partial solutions, that is, detect partial solutions that cannot possibly lead to a valid final solution. This pruning is implemented in the choice of $\mathcal{A}$.

Algorithm 1 Backtrack search

function BACKTRACK(n,$(a_1, a_2, \ldots, a_n)$)
  if $(a_1, a_2, \ldots, a_n)$ is a valid solution **then**
    OUTPUT($a_1, a_2, \ldots, a_n$)
  else
    $\mathcal{A} \leftarrow \{\alpha : (a_1, a_2, \ldots, a_n, \alpha)$ is a valid partial solution$\}$
    for all $\alpha \in \mathcal{A}$ **do**
      BACKTRACK($n + 1$,$(a_1, a_2, \ldots, a_n, \alpha)$)
    end for
  end if
end function

Note that even though the partial solutions are represented as a tuple, the method can be used for constructing any type of combinatorial object, including graphs and subgraphs. For example, paths in a graph can be represented such that the elements a_i in the tuple correspond to the vertices of the path.

3.2 Canonical Augmentation

Canonical augmentation is a method by McKay [48] for exhaustive isomorph-free generation of combinatorial objects. In canonical augmentation objects are constructed recursively by repeatedly augmenting smaller objects. To achieve isomorph free generation, it is required that each object is constructed in a canonical way, i.e., that the sequence of smaller objects in the search tree from which each object is constructed is canonical.

Canonical augmentation is a very general method and can applied to a wide class of problems. Examples of classification problems canonical augmentation has been successfully applied to include the classification of planar graphs [7], permutation arrays [5], perfect binary one-error-correcting codes [54] and Hadamard matrices [43].

Canonical augmentation has several variants. In the following we will present a variant known as weak canonical augmentation, which is sufficient for the classification problems studied in this thesis. For other variants, see [38, Sect. 4.2.3] or [48].

The weak canonical augmentation algorithm is outlined in Algorithm 2, where $C(X)$ are the children of node X. The algorithm works as fol-

Algorithm 2 Weak canonical augmentation

```
function WEAK-CA(X : node)
    if X is a valid solution then
        OUTPUT(X)
    end if
    𝒵 ← ∅
    for all Z ∈ C(X) do
        if p(Z) ≅ w(Z) then
            𝒵 ← 𝒵 ∪ {Z}
        end if
    end for
    Remove isomorphs from 𝒵
    for all Z ∈ C(X) do
        WEAK-CA(Z)
    end for
end function
```

lows. Objects are built with a recursive search, and we denote the parent of an object X in the search tree by $p(X)$. Each object has a sequence of parents $p(X), p(p(X)), \ldots$ from which it is constructed. The object X might be located in several places in the search tree, since there can be several different sequences that lead to an object isomorphic to X. To ensure that only one copy of X is actually constructed, we define a function w that takes objects in the search tree as values and satisfies $X \cong Y \implies w(X) \cong w(Y)$. In the context of canonical augmentation, $w(X)$ is called the (weak) canonical parent of X. In the recursive search an object is rejected if it is not generated from an object isomorphic to its canonical parent, i.e., if $p(X) \ncong w(X)$. Furthermore, an object is rejected if it is isomorphic to some other object that is generated previously from the same parent.

It can be shown that this approach guarantees that exactly one object is generated from each equivalence class of objects, for a proof see for example [38, Sect. 4.2.3].

Even though we defined canonical augmentation as a recursive search, we note that in many applications only one level of recursion is performed. In such cases the parents are often called *seeds*.

3.3 Dynamic Programming

Dynamic programming [14, Chap. 15] refers to an algorithm design technique where large computational problems are solved by breaking them down to several subproblems and then solving the original problem by combining the solutions to these subproblems. The solution to a subproblem is computed only once and then stored in memory, after which the solution is typically used several times during the rest of the computation.

Dynamic programming was introduced by Bellman [4] in the 1950s to solve optimization problems. Since then it has been applied to several problems; recent examples of enumeration problems include enumerating Hamiltonian circuits, walks, and chains [37], enumerating latin squares [35], and enumerating one-factorizations of the complete graph [17].

We will show a simple example of dynamic programming by using the technique to compute the nth Fibonacci number F_n. Fibonacci numbers are defined by $F_0, F_1 = 1$ and the recursive formula $F_n = F_{n-1} + F_{n-2}$. This recursive formula allows us to split the original problem into subproblems, the solutions of which are combined to obtain an answer to the original problem. As is typical with dynamic programming, we store the solutions to subproblems in memory. In this case F_i contains the solution to the subproblem of determining the ith Fibonacci number. We initialize F with $F_0, F_1 = 1$, and the values for $1 < i \leq n$ are then solved in increasing order by combining the solutions to smaller cases with the recursive formula $F_i = F_{i-1} + F_{i-2}$. The answer to the original problem will be stored in F_n.

A large body of literature exists on the topic of dynamic programming algorithms for graphs. For a survey of methods see [3]. One possible way to apply dynamic programming for enumerating cycles or other structures in a graph G is as follows. The subproblems which are stored in memory correspond to the number of partial structures in some subgraph G' of G. The number of partial structures is computed for increasingly large subgraphs G', and finally the total number of structures for G is obtained.

4. Construction

In this chapter we discuss research on construction problems for chordless paths and cycles, and hypohamiltonian planar graphs. In all of the cases studied here the problems consist of constructing a small subset of some larger set of graph structures, such that the subset has some previously defined interesting features.

The three research problems discussed in Section 4.1 deal with exhaustive construction of a well-defined subset of chordless paths and cycles. The research problem discussed in Section 4.2 deals with nonexhaustive heuristic construction of a subset of planar graphs in an attempt to find at least one object that is small and hypohamiltonian.

4.1 Chordless Cycles and Paths

Many algorithms have been proposed for constructing chordless cycles and paths. There are linear-time algorithms for listing all chordless cycles (resp. paths) in a given graph [75]. The running time for these algorithms is linear in the number of chordless cycles (paths) found in the graph. If the number of chordless cycles or paths in a graph is immense, say 10^{30}, these methods are not practical. Determining whether a graph contains a chordless cycle (path) of length at least k is known to be NP-complete [11, 27]. Thus, the problem of determining the maximum length is also computationally difficult.

4.1.1 Snakes and Coils

The study of snakes and coils started in 1958 in a paper by Kautz [39], where they were studied from the point of view of coding theory; Kautz considered coils as Gray codes with some error detection capabilities. In addition to coding theory, applications exist in genetics [85] and flash

memory [82]. For many applications, long snakes perform better than shorter ones. This has motivated research on determining the maximum length of n-snakes and n-coils. Figure 4.1 shows a 3-snake and a 3-coil.

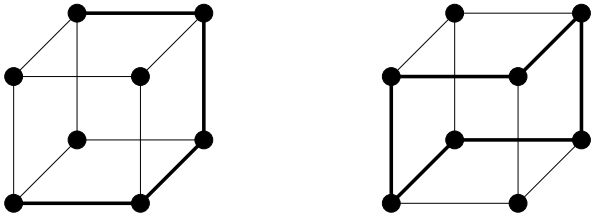

Figure 4.1. A 3-snake and a 3-coil

We denote the length of the longest n-snake by $s(n)$, and the length of the longest n-coil by $c(n)$. The exact values of $s(n)$ and $c(n)$ have previously been determined for $n \leq 7$; The work done is this thesis extends this to $n \leq 8$ [III, IV]. The exact values for $s(n)$ and $c(n)$ are in sequences A099155 and A000937 in OEIS [63], and are summarized for $n \leq 8$ in Table 4.1. In Table 4.1, each length is followed by the source of the lower bound and the upper bound respectively.

Table 4.1. Maximum lengths of n-snakes and n-coils

n	Snakes		Coils	
1	1		-	
2	2	[15],[15]	4	[39],[39]
3	4	[15],[15]	6	[39],[39]
4	7	[15],[15]	8	[39],[39]
5	13	[15],[15]	14	[39],[39]
6	26	[15],[15]	26	[24],[15]
7	50	[57],[57]	48	[24],[42]
8	98	[8], [III]	96	[56], [IV]

For $n \geq 9$, only lower and upper bounds are known. These bounds include asymptotic bounds and bounds for specific values of n. Finding a long snake or coil in the n-cube trivially gives a lower bound for $s(n)$ or $c(n)$ respectively. The best known asymptotic upper bound for the length (number of edges) of a coil is [84]

$$2^{n-1}(1 - 1/(89\sqrt{n}) + O(1/n)).$$

For other asymptotic bounds, see [1, 18, 16, 23, 64, 65, 80, 84].

We will now proceed to give an outline of the methods that have been used to obtain the record length snakes and coils that are summarized in Table 4.1.

Kautz [39] determined the optimal length for snakes in dimensions 2 to 5. These lengths were determined by trial and error. He also gave an erroneous maximum length of 24 for dimension 6. The first computational method [15] was published in 1965 by Davies. In [15] snakes and coils are constructed using exhaustive backtrack search with symmetry-based pruning. The pruning reduces the search space by a factor of $2^n n!$; in the search, it is required that the snake starts in a fixed vertex, and that the snake traverses new dimensions in a lexicographic order. This method is used in [15] to determine optimal lengths for n-snakes and n-coils for $n \leq 6$.

In [57], Potter et al. use a genetic algorithm for creating long snakes and coils, finding new record-breaking snakes in dimensions 7 and 8. Genetic algorithms are optimization algorithms that mimic biological evolution. In [57] the genetic algorithm works by creating a collection of candidate snakes, and then creating new candidate snakes by making several (random) changes to the snakes in the collection. Snakes that are of low quality ("fitness") based on some fitness function are occasionally removed from the collection. This way the population eventually "evolves" to hopefully contain high quality (long) snakes. In [57] the authors also report verifying the maximum length for a 7-snake with an exhaustive search, although details of the exhaustive method are not presented in the article.

In 1996, Kochut [42] presented a method for constructing coils using exhaustive backtrack search and some pruning rules. The method is very similar to the one in [15]. It is also mentioned in [42] that the method is very similar to the exhaustive method used in [57], but this is difficult to verify since [57] does not give any details on how its exhaustive search method works. In [42] the authors establish the maximum length of a 7-coil and find all maximum length 7-coils.

In [56] Paterson and Tuliani discover a record-breaking coil in dimension 8. It has later been established in [IV] that this coil is also of optimal length. The method in [56] is a construction based on rearranging *necklaces*, which are defined as equivalence classes of words under cyclic shifting.

In [8] Carlson uses a method based on genetic algorithms to find a record-breaking snake of length 98 in dimension 8. It has later been es-

tablished in [III] that this snake is also of optimal length. The genetic algorithm is different from [57] in the sense that it does not optimize snakes directly; instead it is used to optimize heuristics that can be used for building snakes with recursive search.

The work done in this thesis (in [III,IV]) is the first to apply a method based on canonical augmentation on the problem of finding long snakes and coils. The method turns out to be fast enough for establishing the maximum length of snakes and coils in dimension 8, and constructing some new coils of maximum length.

The maximum length of n-snakes and n-coils is not currently known for $n \geq 9$. For $n \geq 9$, only nonexhaustive search methods have been applied. We will describe two such methods, because they bear some similarities to the method in [IV]. In 2012, Wynn presented two new constructions for coils [81]. The first one consists of building a coil by attaching multiple permuted copies of one relatively short snake, and the other one consists of combining two long snakes in lower dimensions to form one long coil. Using these constructions Wynn finds some record-breaking coils in dimensions 9, 10, and 11 of length 188, 348, and 640, respectively. Combining two long snakes can be seen as a small special case of the method in [III]. Attaching multiple permuted copies of a short snake corresponds to constructing coils with certain prescribed automorphism groups.

4.1.2 Toeplitz' Conjecture

The Toeplitz' conjecture, also known as the square peg problem, is an old conjecture in topology. It was introduced in 1911 by Otto Toeplitz [72] and can be formulated as follows.

Conjecture 23 (Toeplitz' conjecture)**.** Every Jordan curve in $\mathbb{R}^2$ contains four points forming the corners of a square.

A Jordan curve is a continuous closed loop in the plane that does not intersect itself. Figure 4.2 shows a Jordan curve with an inscribed square.

So far the conjecture remains unproven. Only partial results have been obtained, which consist of proving the conjecture for Jordan curves that fulfill certain conditions, usually smoothness-related conditions. These results are based on advanced topological methods.

Already in [72] where Toeplitz' introduced his conjecture he claimed to have a proof that the conjecture holds for convex curves. However, this proof was never published. The first published results came from Emch in

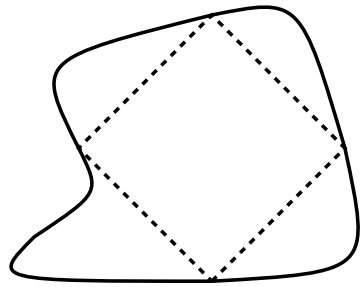

Figure 4.2. A Jordan curve with an inscribed square

the next few years. In 1913 and 1915 Emch showed [20, 21] that Toeplitz' conjecture holds for convex curves with certain smoothness constraints, and in 1916 he proved the conjecture for piecewise analytical curves with some additional constraints [22]. This last class of curves contains for example all polygons.

After the first results by Emch, numerous results have been published which prove the theorem for different classes of curves, such as locally monotone curves [68] and curves symmetric across a line [51]. For a more detailed listing of results see [46].

Toeplitz' conjecture is known to hold for polygons. However, Jordan curves can have a very complex structure that cannot be approximated adequately by polygons. Examples of features that are possible for a Jordan curve but not for a polygon include having positive area and spirals that coil an infinite number of times [53].

It is known that if one can prove a certain type of lower bound on the size of the inscribed square on a polygon, it would be possible to prove that the conjecture holds for all Jordan curves [46]. For example a bound of the form ϵr is sufficient, where $\epsilon > 0$ and r is the radius of the largest ball that can fit inside the polygon.

The study done in [VI] takes the approach described in the previous paragraph slightly further. In [VI] a conjecture is presented that gives a lower bound on the size of an inscribed square for a certain family of polygons, and it is shown that this conjecture implies the Toeplitz' conjecture. An interesting feature of the conjecture in [VI] is that it can be validated computationally.

We denote by $\mathcal{J}'$ the family of finite Jordan curves that consist of segments with endpoints (x, y) and $(x + 1, y)$ or (x, y) and $(x, y + 1)$, such that x and y are integers. Furthermore, we denote by $i(J)$ the side length of the largest axis-aligned square inside J. The conjecture presented in [VI]

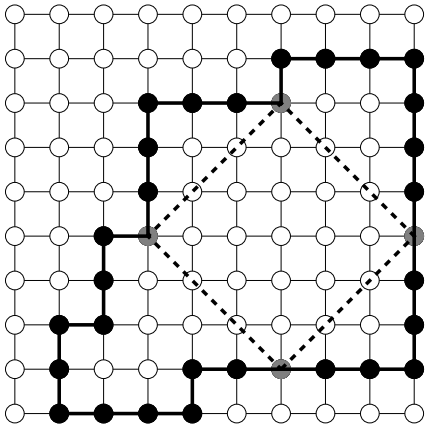

Figure 4.3. A Jordan curve with an inscribed square (on a grid)

is as follows.

Conjecture 24. A Jordan curve $J \in \mathcal{J}'$ contains four points of the integer lattice that form the corners of a square of side length at least $i(J)/\sqrt{2}$.

Conjecture 24 deals with Jordan curves that consist of axis-aligned lines of length one. Figure 4.3 shows an example of such a curve and an inscribed square. The conjecture can be validated by verifying the nonexistence of certain chordless cycles in the $n \times n$ grid graph, which is done up to $n \leq 13$ in [VI] using backtrack search with several pruning rules.

Earlier results on the Toeplitz' conjecture have been based on proving topological theorems. The approach in [VI] is different in the sense that it allows one to study the problem with computational and combinatorial methods.

4.2 Planar Hypohamiltonian Graphs

The study of hypohamiltonian graphs was started in 1963 with a research problem published by Sousselier [66]. In response to Sousselier, Gaudin et al. [28] obtained that the Petersen graph, which has 10 vertices, is the smallest hypohamiltonian graph.

The result by Gaudin et al. was the first step in determining the possible orders of hypohamiltonian graphs. In 1967, Herz et al. [34] used a systematic computer search to show that there are no hypohamiltonian graphs on 11 or 12 vertices. They also discovered hypohamiltonian graphs of orders 13 and 15. Later it was shown that hypohamiltonian graphs exist on every order larger than 12 except possibly 14 and 17 [6, 12, 19, 44, 69, 70].

The nonexistence of hypohamiltonian graphs of order 14 was confirmed in 1978 by Collier at al. using a computer search [13]. The last open case was solved by Aldred et al. in 1997 [2] by using a computer to show the nonexistence of hypohamiltonian graphs of order 17. In short, there is exactly one hypohamiltonian graph for order 10, 13 and 15, four such graphs for order 16, and no graphs for order 17. Furthermore, hypohamiltonian graphs exist for all orders greater than 17.

The existence of *planar* hypohamiltonian graphs for specific orders remains open. In 1973 Chvátal [12] asked whether planar hypohamiltonian graphs exist, and this was answered in the positive by Thomassen in 1976 [71] with the graph shown in Figure 4.4. Thomassen's graph has 105 vertices, and its discovery started a search for a planar hypohamiltonian graph with the smallest possible order.

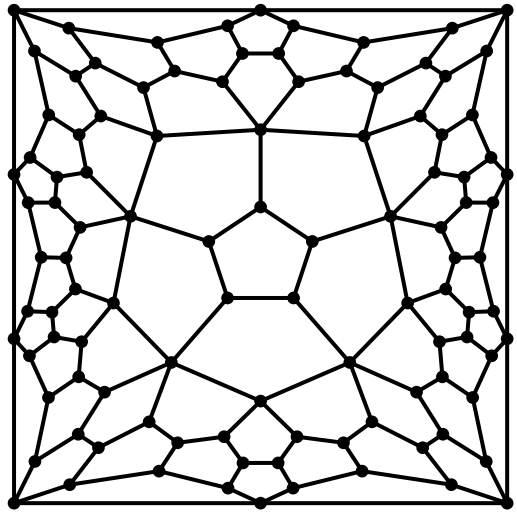

Figure 4.4. A planar hypohamiltonian graph of order 105

In 1968 Grinberg proved the following necessary condition for a planar graph to be Hamiltonian [78].

Theorem 25 (Grinberg's Theorem)**.** *Given a loopless plane graph with a Hamiltonian cycle C and f_i (f'_i) i-gons inside (outside) of C, we have*

$$\sum_i (i-2)(f_i - f'_i) = 0.$$

Grinberg's condition plays an important role in finding small planar hypohamiltonian graphs. Most of the publications on the topic make use of Grinberg's theorem in one way or another.

Even though systematic computer searches were successfully used in establishing the possible orders of small hypohamiltonian graphs, none of the earlier publications [33, 71, 79, 83] on small planar hypohamiltonian graphs discuss computer searches. The research in [I] is the first published result to apply systematic computer searches on the problem of finding small planar hypohamiltonian graphs. The *girth* of a graph is the length of a shortest cycle in the graph. In [I], a certain family of planar graphs with girth 4 and a fixed number of 4-faces is generated by exhaustively generating planar graphs of girth 5 and then adding 4-faces to these graphs. This resulted in the discovery of record-breaking planar hypohamiltonian graphs on 40 vertices. In addition to finding record-breaking graphs, it is shown in [I] through exhaustive verification that direct application of Grinberg's condition cannot result in graphs with fewer than 42 vertices. Here direct application of Grinberg's condition refers to studying graphs that can be shown to be nonhamiltonian with Grinberg's condition based on only their sequence of face sizes.

Table 4.2 shows the timeline of small planar hypohamiltonian graphs. It is shown in [I] that planar hypohamiltonian graphs exist for order 40 and every order larger than 41, but the case 41 remains open.

Table 4.2. Timeline of small planar hypohamiltonian graphs

Order	Year	Author
105	1976	Thomassen [71]
57	1979	Hatzel [33]
48	2007	Zamfirescu and Zamfirescu [83]
42	2010	Wiener and Araya [79]
40	2014	Jooyandeh et al. [I]

5. Enumeration

In this chapter we discuss research on enumeration problems for Hamiltonian cycles and perfect matchings. Most of the methods discussed in this chapter enumerate objects without explicitly constructing each object.

5.1 Perfect Matchings

Some of the earliest studies on enumerating perfect matchings appeared in the fields of chemistry and physics in the early 1900s [26, 29]. Perfect matchings played a role in the study of arrangements of some molecules, and also in modeling liquids.

There are methods that can list all perfect matchings in a bipartite graph in linear time with respect to the number of matchings [74]. Thus, the problem of enumerating perfect matchings requires new computational methods only for those graphs where the number is too large to explicitly list each matching. In general, computing the number of perfect matchings is known to be #P-complete [76].

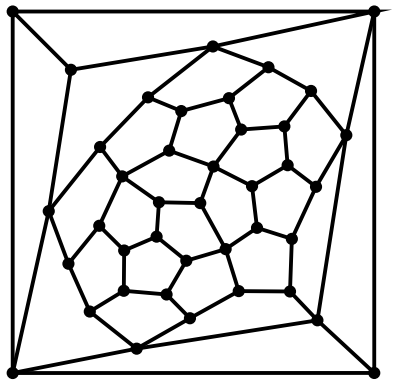

Figure 5.1. A planar hypohamiltonian graph (from [I])

Enumeration of perfect matchings in the n-cube is considered to be one of the main open problems regarding enumerations of matchings, see [58, Problem 14]. Figure 5.2 shows a perfect matching in the 3-cube.

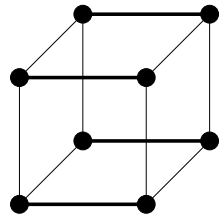

Figure 5.2. A perfect matching in the 3-cube

Some of the earlier work done on enumerating perfect matchings in the n-cube is based on permanents of matrices [31, 45]. The *permanent* of a matrix M is defined as

$$\text{per}(M) = \sum_{\pi} \prod_{i=1}^{n} m_{i,\pi(i)}. \tag{5.1}$$

It can be shown that the number of perfect matchings in a bipartite graph G is equal to $\text{per}(B(G))$, where $B(G)$ is the biadjacency matrix of the graph. Computing the permanent is computationally difficult, and is known to be #P-hard for general graphs [45]. Straightforward application of (5.1) would result in an algorithm that requires $n \cdot n!$ arithmetic operations to compute the result. However, faster methods exist. Ryser [59] proved the following formula which reduces the number of necessary operations to the order of $n2^n$:

$$\text{per}(M) = (-1)^n \sum_{S \subseteq [n]} (-1)^{|S|} \prod_{i=1}^{n} \sum_{j \in S} m_{i,j}. \tag{5.2}$$

We denote the number of perfect matchings in the n-cube by $f(n)$. Graham and Harary obtained in 1988 the values for $n \leq 5$ with a method based on permanents [31]. In 1996, Lundow obtained $f(6)$, again with a method based on permanents [45].

The work in [II] is not based on permanents. Instead, [II] presents a method based on dynamic programming. The method turns out to be faster than the earlier methods shown in [31, 45], and is able to determine $f(7)$, as well as the numbers of equivalence classes of perfect matchings, denoted here by $F(n)$, for $n \leq 7$. The method in [II] makes heavy use of the symmetry in the 7-cube. The method can be applied for any bipartite graph, but the main emphasis is on the 7-cube. Table 5.1 summarizes the results obtained in [II].

Table 5.1. Perfect matchings in the n-cube

n	$f(n)$	$F(n)$
1	1	1
2	2	1
3	9	2
4	272	8
5	589 185	336
6	16 332 454 526 976	356 788 059
7	391 689 748 492 473 664 721 077 609 089	607 158 046 495 120 886 820 621

5.2 Hamiltonian Cycles

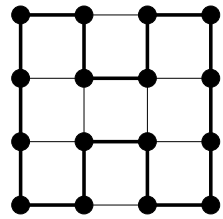

Figure 5.3. A Hamiltonian cycle in the 4×4 grid graph

Some of the research done on enumerating Hamiltonian cycles is motivated by applications in physics. Examples of such applications include the study of protein folding and polymer melting [37].

Determining whether an arbitrary graph contains Hamiltonian cycles is known to be an NP-complete problem [27, p. 199]. Some examples of algorithms for determining the hamiltonicity of a graph can be found in [77]. Counting Hamiltonian cycles in arbitrary graphs is #P-complete [76]. For some graph families there are known explicit formulas for the number of Hamiltonian cycles. Examples include the complete graph K_n, where the number is $(n-1)!/2$, and the $3 \times n$ grid graph, where the number is $2^{n/2}$ for even n and 0 for odd n [67]. We denote the number of Hamiltonian cycles in the $(m+1) \times (n+1)$ grid graph by $H_{m,n}$. For fixed m, $H_{m,n}$ is given by a linear recurrence relation [67]. The first 3 relations are

$$H_{1,n} = H_{1,n-1}$$
$$H_{2,n} = 2H_{2,n-2}$$
$$H_{3,n} = 2H_{3,n-1} + 2H_{3,n-2} - 2H_{3,n-3} + H_{3,n-4},$$

where $H_{m,n} = 1$ for $n = 1$ and $H_{m,n} = 0$ for $n < 1$. For relations for larger m, see [67]. Figure 5.3 shows a Hamiltonian cycle in a 4×4 grid graph.

Previous algorithmic approaches to enumerating Hamiltonian cycles include zero-suppressed decision diagrams [41, pp. 254–255], transfer matrix methods [61], dynamic programming [32, 37], and backtrack search [10, 55]. Dynamic programming and backtrack search were discussed in Chapter 3. We will now briefly discuss the other two approaches.

Zero-supressed decision diagrams, or ZDDs for short, are a data structure that can be used for storing boolean functions. Knuth introduced an algorithm [41, pp. 254–255] for constructing a ZDD that can represent all cycles (or various other structures) in a graph. The representation can be used for enumerating cycles in a graph without explicitly constructing each cycle. Other examples of using ZDDs include enumerating certain types of paths in a grid graph [36].

In the transfer matrix method, incomplete Hamiltonian cycles are presented as belonging to several different states, and a matrix is constructed that presents the possible transitions between the states as incomplete cycles are extended. The total number of Hamiltonian cycles can then be obtained with simple matrix multiplication. This type of approach can be found in [61, 67]. We note that the work in [67] considers Hamiltonian cycles as a regular language which can be recognized by a finite state automaton.

The method presented in [II] is based on dynamic programming and contains many improvements over earlier methods in terms of running time and memory requirements. Another advantage of the method in [II] is that it is very general and can be applied to any graph. This makes it possible to enumerate Hamiltonian cycles in larger cases than previously published for several important families of graphs, namely grid graphs, three-dimensional grid graphs, triangular grids, and king's graphs. For example, the number of Hamiltonian cycles in the $n \times n$ king's graph was previously known for $n \leq 8$, which was obtained with a method based on ZDDs [41, pp. 254–255]. In [II] the values are obtained for $n \leq 16$.

Tables 5.2 to 5.5 summarize the results obtained in [II]. New results are denoted with a star.

Table 5.2. Hamiltonian cycles in $n \times n$ grid graphs

n	Number of Hamiltonian cycles
2	1
4	6
6	1 072
8	4 638 576
10	467 260 456 608
12	1 076 226 888 605 605 706
14	56 126 499 620 491 437 281 263 608
16	65 882 516 522 625 836 326 159 786 165 530 572
18	1 733 926 377 888 966 183 927 790 794 055 670 829 347 983 946
20	1 020 460 427 390 768 793 543 026 965 678 152 831 571 073 052 662 428 097 106
22	13 404 209 505 893 761 748 339 786 653 564 937 498 745 897 123 531 041 248 680 272 448 954 956
24*	3 924 231 694 060 647 894 532 092 926 553 286 517 550 515 989 932 148 978 428 996 623 179 638 255 782 936 503 022
26*	25 578 285 385 897 276 060 130 031 526 614 700 187 075 412 685 764 186 583 833 403 069 393 167 252 132 218 312 152 073 569 856 334 502

Table 5.3. Hamiltonian cycles in $n \times n$ king's graphs

n	Number of Hamiltonian cycles
1	1
2	3
3	16
4	2 830
5	2 462 064
6	22 853 860 116
7	1 622 043 117 414 624
8	961 742 089 476 282 321 684
9∗	4 601 667 243 759 511 495 116 347 104
10∗	179 517 749 570 891 592 016 479 828 267 003 018
11∗	56 735 527 086 758 553 613 684 823 040 730 404 215 973 136
12∗	145 328 824 470 156 271 670 635 015 466 987 199 469 360 063 082 789 418
13∗	3 013 072 757 042 748 407 212 267 203 778 429 049 866 618 090 427 057 156 382 635 712
14∗	505 396 541 863 296 313 964 793 910 305 382 425 060 965 154 779 449 831 170 884 147 484 924 489 066
15∗	685 457 393 589 353 762 730 302 985 699 040 971 223 260 321 251 614 789 007 892 889 891 954 959 485 049 448 085 648
16∗	7 514 427 561 614 895 453 501 809 269 193 245 238 210 545 618 759 874 171 463 008 264 116 894 117 263 163 499 356 026 497 440 057 866

Table 5.4. Hamiltonian cycles in triangular grids with n vertices on one side

n	Number of Hamiltonian cycles
1	1
2	1
3	1
4	3
5	26
6	474
7	17 214
8	1 371 454
9∗	231 924 780
10∗	82 367 152 914
11∗	61 718 801 166 402
12∗	97 482 824 713 311 442
13∗	323 896 536 556 067 453 466
14∗	2 262 929 852 279 448 821 099 932
15∗	33 231 590 982 432 936 619 392 054 662
16∗	1 025 257 090 790 362 187 626 154 669 771 934
17∗	66 429 726 878 393 651 076 826 663 971 376 589 034
18∗	9 036 923 456 137 297 343 631 952 691 847 844 984 938 922
19∗	2 580 487 118 457 573 944 114 883 930 049 209 668 222 079 445 692
20∗	1 546 380 273 056 984 325 468 288 805 366 239 469 621 495 241 688 789 648

Table 5.5. Hamiltonian cycles in three-dimensional $n \times m \times k$ grid graphs

$n \times m \times k$	Number of Hamiltonian cycles
$2 \times 2 \times 2$	6
$2 \times 2 \times 3$	22
$2 \times 2 \times 4$	82
$2 \times 2 \times 5$	306
$2 \times 3 \times 3$	324
$2 \times 3 \times 4$*	4 580
$2 \times 3 \times 5$*	64 558
$2 \times 4 \times 4$*	232 908
$2 \times 4 \times 5$*	11 636 834
$2 \times 5 \times 5$*	2 040 327 632
$3 \times 3 \times 3$	0
$3 \times 3 \times 4$	3 918 744
$3 \times 3 \times 5$	0
$3 \times 4 \times 4$	3 777 388 236
$3 \times 4 \times 5*$	3 180 229 736 508
$3 \times 5 \times 5$	0
$4 \times 4 \times 4*$	57 958 048 716 672
$4 \times 4 \times 5*$	857 931 886 492 226 164
$4 \times 5 \times 5*$	211 005 720 188 979 351 400 104
$5 \times 5 \times 5$	0

6. Conclusions

In this thesis we studied algorithms for enumerating and constructing cycles and related structures. The problems that we studied were enumeration of perfect matchings in bipartite graphs, particularly in the n-cube, enumeration of Hamiltonian cycles in general graphs, exhaustive construction of certain extremal (long) chordless cycles and chordless paths in the n-cube, exhaustive construction of certain extremal chordless cycles in the grid graph in an attempt to validate the Toeplitz' conjecture, and nonexhaustive construction of small planar hypohamiltonian graphs.

We designed new algorithms for each of the aforementioned problems, and applied these algorithms to obtain results for many previously unsolved cases. The algorithms are presented in [I] to [VI]. In the following we present some ideas for future research.

Some of the research problems in this thesis can be studied further in a relatively straightforward way. The methods in [III] and [IV] can be used directly to obtain a complete classification of maximum length 8-coils and 8-snakes, given enough computing power. A rough estimate of the amount of CPU-time required for these two classification tasks is in the order of 100 core-years. As for the computational method used in [VI], it may be possible to improve the method simply by adding more pruning rules. This could lead to a proof of Conjecture C in [VI, Sect. 2] for larger grid graphs and possibly new theoretical insights on the conjecture.

Ordering vertices in the method of [V] in a certain way could allow parallelization of the method and greatly reduce memory requirements. The order should have the property that for some small k, the first k vertices all have neighbours among the last k vertices.

Prescribing automorphism groups of structures is a general technique for restricting the search space when searching for combinatorial structures. This technique might be well suited for finding long chordless cycles

in the n-cube. For chordless cycles in the n-cube, an automorphism must be at the same time an automorphism of a cycle and an automorphism of the n-cube, which reduces the number of possible automorphisms to consider. Prescribing automorphism groups might also be useful for finding small planar hypohamiltonian graphs. However, prescribing automorphism groups for chordless paths is probably not very useful, since a path can have an automorphism group size of at most 2. Many of the graph structures found during the research work for [I] and [IV] have nontrivial automorphism groups. Careful study of these structures may give insight into what types of automorphisms to consider.

In addition to prescribing automorphism groups, further research can be done on the topic of [I] by approaching the problem from a slightly different direction; A lower bound on the size of planar hypohamiltonian graphs (see [I]) could be obtained by exhaustively constructing all planar hypohamiltonian graphs with n vertices for small n. Such a search has been conducted for general graphs [2].

Some of the methods used in this thesis can be modified to solve other, closely related problems. The method for enumerating perfect matchings in [II] could be modified to enumerate all matchings. The method for enumerating Hamiltonian cycles in [V] could be modified to enumerate Hamiltonian paths or other similar structures.

References

[1] H. L. Abbott, M. Katchalski: On the snake in the box problem. J. Combin. Theory, Series B, 44, 12–24 (1988)

[2] R. E. L. Aldred, B. D. McKay, N. C. Wormald: Small hypohamiltonian graphs. J. Combin. Math. Combin. Comput., 23, 143–152 (1997)

[3] S. Arnborg: Efficient algorithms for combinatorial problems on graphs with bounded decomposability–A survey. BIT, 25, 1–23 (1985)

[4] R. Bellman: Dynamic Programming. Princeton University Press, Princeton, NJ (1957)

[5] M. Bogaerts: Isometries and construction of permutation arrays. IEEE Trans. Inform. Theory, 56, 3177–3179 (2010)

[6] J. A. Bondy: Variations on the Hamiltonian theme. Can. Math. Bull., 15, 57–62 (1972)

[7] G. Brinkmann, B. D. McKay: Fast generation of planar graphs. MATCH Commun. Math. Comput. Chem, 58, 323–357 (2007)

[8] B. P. Carlson, D. F. Hougen: Phenotype feedback genetic algorithm operators for heuristic encoding of snakes within hypercubes. Proc. 12th Conf. Genetic and Evolutionary Computation, 791–798 (2010)

[9] A. Cayley: A theorem on trees. Quart. J. Math, 23, 376–378 (1889)

[10] A. Chalaturnyk: A Fast Algorithm for Finding Hamilton Cycles. M.Sc. thesis, University of Manitoba (2008)

[11] Y. Chen, J. Flum: On parameterized path and chordless path problems. Proc. 22nd IEEE Conf. Computational Complexity, 250–263 (2007)

[12] V. Chvátal: Flip-flops in hypohamiltonian graphs. Can. Math. Bull., 16, 33–41 (1973)

[13] J. B. Collier, E. F. Schmeichel: Systematic searches for hypohamiltonian graphs. Networks, 8, 193–200 (1978)

[14] T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein: Introduction to Algorithms, 3rd ed.. MIT Press, Cambridge, MA (2009)

[15] D. W. Davies: Longest “separated” paths and loops in an N Cube. IEEE Trans. Electronic Computers, 2, 261–261 (1965)

[16] K. Deimer: A new upper bound for the length of snakes. Combinatorica, 5, 109–120 (1985)

[17] J. H. Dinitz, D. K. Garnick, B. D. McKay: There are 526,915,620 nonisomorphic one-factorizations of K_{12}. J. Combin. Des., 2, 273–285 (1994)

[18] R. J. Douglas: Upper bounds on the length of circuits of even spread in the d-cube. J. Combin. Theory, 7, 206–214 (1969)

[19] J. Doyen, V. van Diest: New families of hypohamiltonian graphs. Discrete Math., 13, 225–236 (1975)

[20] A. Emch: Some properties of closed convex curves in a plane. Amer. J. Math., 35, 407–412 (1913)

[21] A. Emch: On the medians of a closed convex polygon. Amer. J. Math, 37, 19–28 (1915)

[22] A. Emch. On some properties of the medians of closed continuous curves formed by analytic arcs. Amer. J. Math., 38, 6–18 (1916)

[23] A. A. Evdokimov: Maximal length of a chain in a unit n-dimensional cube. Matematicheskie Zametki, 6, 309–319 (1969)

[24] S. Even: Snake in the box codes. IEEE Trans. Electronic Computers, 1, p. 18 (1963)

[25] J. Fink: Perfect matchings extend to Hamilton cycles in hypercubes. J. Combin. Theory, Series B, 97, 1074–1076 (2007)

[26] R. H. Fowler, G. S. Rushbrooke: An attempt to expand the statistical theory of perfect solutions. Trans. Faraday Soc., 33, 1272–1294 (1937)

[27] M. R. Garey, D. S. Johnson: Computers and Intractability: A Guide to the Theory of NP-Completeness. Freeman, New York (1979)

[28] T. Gaudin, J. C. Herz, P. Rossi: Solution du problème no. 29. Rev. Franc. Rech. Operat., 8, 214–218 (1964)

[29] M. Gordon, W. H. T. Davison: Theory of resonance topology of fully aromatic hydrocarbons. I; J. of Chem. Phys., 20, 428–435 (1952)

[30] R. L. Graham, M. Grötschel, L. Lovász, (Eds.): Handbook of Combinatorics (Vol. 1). Elsevier (1995)

[31] N. Graham, F. Harary: The number of perfect matchings in a hypercube. Appl. Math. Lett., 1, 45–48 (1988)

[32] H. Haanpää, P. R. J. Östergård: Counting Hamiltonian cycles in bipartite graphs. Math. Comp., 83, 979–995 (2014)

[33] W. Hatzel: Ein planarer hypohamiltonscher Graph mit 57 Knoten. Math. Ann., 243, 213–236 (1979)

[34] J. C. Herz, J. J. Duby, F. Vigué: Recherche systématique des graphes hypohamiltoniens. Proc. 1966 Internl. Symp. in Rome P. Rosenthiel, ed., Dunod, Paris 153–160 (1967)

[35] A. Hulpke, P. Kaski, P. R. J Östergård: The number of Latin squares of order 11. Math. Comp., 80, 1197–1219 (2011)

[36] H. Iwashita, J. Kawahara, S. Minato: ZDD-based computation of the number of paths in a graph. Hokkaido University, Division of Computer Science, TCS Technical Reports, vol. TCS-TR-A-10-60 (2012)

[37] J. L. Jacobsen: Exact enumeration of Hamiltonian circuits, walks and chains in two and three dimensions. J. Phys. A: Math. Theor., 40, 14667–14678 (2007)

[38] P. Kaski, P. R. J. Östergård: Classification Algorithms for Codes and Designs. Berlin, Germany: Springer (2006)

[39] W. H. Kautz: Unit-distance error-checking codes. IRE Trans. Electronic Computers, 179–180 (1958)

[40] D. Kinny: Monte-Carlo search for snakes and coils. Proc. Multidisciplinary Trends in Artificial Intelligence, 271–283 (2012)

[41] D. E. Knuth: The Art of Computer Programming, Volume 4A, Combinatorial Algorithms, Part 1. Addison-Wesley, Upper Saddle River (2011)

[42] K. J. Kochut: Snake-in-the-box codes for dimension 7. J. Combin. Math. Combin. Comput., 20, 175–185 (1996)

[43] P. H. Lampio, F. Szöllősi, P. R. J. Östergård, : The quaternary complex Hadamard matrices of orders 10, 12, and 14. Discrete Math., 313, 189–206 (2013)

[44] W. F. Lindgren: An infinite class of hypohamiltonian graphs. Amer. Math. Monthly, 74, 1087–1089 (1967)

[45] P. H. Lundow: Computation of matching polynomials and the number of 1-factors in polygraphs. Research Reports No. 12. Department of Mathematics, Umeå University (1996)

[46] B. Matschke: A survey on the Square Peg Problem. Notices Amer. Math. Soc., 16, 346–352 (2014)

[47] B. Matschke: Equivariant topology methods in discrete geometry. Ph.D. thesis, Freie Universität Berlin (2011)

[48] B. D. McKay: Isomorph-free exhaustive generation. J. Algorithms, 26, 306–324 (1998)

[49] B. D. McKay: nauty user's guide (version 1.5). Computer Science Department, Australian National University, Canberra, Tech. Rep. TR-CS-90-02 (1990)

[50] B. D. McKay, I. M. Wanless: On the number of Latin squares. Ann. Comb., 9, 335–344 (2005)

[51] M. J. Nielsen, S. E. Wright: Rectangles inscribed in symmetric continua. Geom. Dedicata, 56, 285–297 (1995)

[52] R. Oberdorf, A. Ferguson, J. L. Jacobsen, J. Kondev: Secondary structures in long compact polymers. Physical Review E, 74, 051801 (2006)

[53] W. F. Osgood: A Jordan curve of positive area. Trans. Amer. Math. Soc., 4, 107–112. Chicago (1903)

[54] P. R. J. Östergård, O. Pottonen: The Perfect Binary One-Error-Correcting Codes of Length 15: Part I—Classification. IEEE Trans. Inform. Theory, 55, 4657–4660 (2009)

[55] V. S. Pande, C. Joerg, A. Y. Grosberg, T. Tanaka: Enumerations of the Hamiltonian-walks on a cubic sublattice. Journal of physics. A, mathematical and general, 27, 6231–6236 (1994)

[56] K. G. Paterson, J. Tuliani: Some new circuit codes, IEEE Trans. Inform. Theory, 44, 1305–1309 (1998)

[57] W. D. Potter, R. W. Robinson, J. A. Miller, K. J. Kochut, D. Z. Redys: Using the genetic algorithm to find snake-in-the-box codes. Proc. 7th Intl. Conf. on Industrial and Eng. Applic. of Artificial Intelligence and Expert Systems, 421–426 (1994)

[58] J. Propp: Enumeration of matchings: Problems and progress. In: Billera, L. J., Björner, A., Greene, C., Simion, R. E., Stanley, R. P. (Eds.) New Perspectives in Algebraic Combinatorics, 255–291. Cambridge University Press, Cambridge (1999)

[59] H. J. Ryser: Combinatorial mathematics. The Carus mathematical monographs (1963)

[60] G. Sabidussi: Graph multiplication. Mathematische Zeitschrift, 72, 446–457 (1959)

[61] T. G. Schmalz, G. E. Hite, D. J. Klein: Compact self-avoiding circuits on two-dimensional lattices. J. Phys. A: Math. Gen., 17, 445–453 (1984)

[62] A. J. Schwenk: The number of caterpillars. Discrete Math., 6, 359–365 (1973)

[63] N. J. A. Sloane: The On-Line Encyclopedia of Integer Sequences [Online]. Available: http://oeis.org (2012)

[64] H. S. Snevily: The snake-in-the-box problem: a new upper bound. Discrete Math., 133, 307–314 (1994)

[65] F. I. Solov'eva: An upper bound on the length of a cycle in an n-dimensional unit cube (in Russian). Metody Diskret. Analiz., no. 45, 71–76 and 96–97 (1987)

[66] R. Sousselier: Problème no. 29: Le cercle des irascibles. Rev. Franc. Rech. Operat., 7, 405–406 (1963)

[67] R. Stoyan, V. Strehl: Enumeration of Hamiltonian circuits in rectangular grids. J. Combin. Math. Combin. Comput., 21, 109–128 (1996)

[68] W. R. Stromquist: Inscribed squares and square-like quadrilaterals in closed curves. Mathematika, 36, 187–197 (1989)

[69] C. Thomassen: Hypohamiltonian and hypotraceable graphs. Discrete Math., 9, 91–96 (1974)

[70] C. Thomassen: On hypohamiltonian graphs. Discrete Math., 10, 383–390 (1974)

[71] C. Thomassen: Planar and infinite hypohamiltonian and hypotraceable graphs. Discrete Math., 14, 377–389 (1976)

[72] O. Toeplitz: Ueber einige Aufgaben der Analysis Situs. Verhandlungen der Schweizerischen Naturforschenden Gesellschaft in Solothurn, 4, p. 197 (1911)

[73] D. R. Tuohy, W. D. Potter, D. A. Casella: Searching for snake-in-the-box codes with evolved pruning models. Proc. 2007 Int. Conf. Genetic and Evolutionary Methods, 3–9 (2007)

[74] T. Uno: A fast algorithm for enumerating bipartite perfect matchings. In: Algorithms and Computation, Lecture Notes in Computer Science, 2223, 367–379, Springer Berlin Heidelberg. Chicago (2001)

[75] T. Uno, H. Satoh: An Efficient Algorithm for Enumerating Chordless Cycles and Chordless Paths. Discovery Science, Lecture Notes in Computer Science, 8777, 313–324 (2014)

[76] L. G. Valiant: The complexity of enumeration and reliability problems. SIAM J. Comput., 8, 410–421 (1979)

[77] B. Vandegriend: Finding Hamiltonian Cycles: Algorithms, Graphs and Performance. M.Sc. thesis, University of Alberta (1998)

[78] D. B. West: Introduction to Graph Theory, 2nd edition. Prentice Hall, Upper Saddle River (2001)

[79] G. Wiener, M. Araya: On planar hypohamiltonian graphs. J. Graph Theory, 67, 55–68 (2011)

[80] J. Wojciechowski: A new lower bound for snake-in-the-box codes. Combinatorica, 9, 91–99 (1989)

[81] E. Wynn: Constructing circuit codes by permuting initial sequences. Available in arXiv, arXiv:1201.1647 (2012)

[82] Y. Yehezkeally, M. Schwartz: Snake-in-the-box codes for rank modulation. IEEE Trans. Inform. Theory, 58, 5471–5483 (2012)

[83] C. T. Zamfirescu, T. I. Zamfirescu: A planar hypohamiltonian graph with 48 vertices. J. Graph Theory, 55, 338–342 (2007)

[84] G. Zémor: An upper bound on the size of the snake-in-the-box. Combinatorica, 17, 287–298 (1997)

[85] I. Zinovik, Y. Chebiryak, D. Kroening: Periodic orbits and equilibria in glass models for gene regulatory networks. IEEE Trans. Inform. Theory, 56, 805–820 (2010)

[86] I. Zinovik, D. Kroening, Y. Chebiryak: Computing binary combinatorial Gray codes via exhaustive search with SAT solvers. IEEE Trans. Inform. Theory, 54, 1819–1823 (2008)

Aalto-DD 127/2015

ISBN 978-952-60-6364-5 (printed)
ISBN 978-952-60-6365-2 (pdf)
ISSN-L 1799-4934
ISSN 1799-4934 (printed)
ISSN 1799-4942 (pdf)

Aalto University
School of Electrical Engineering
Department of Communications and Networking
www.aalto.fi

BUSINESS + ECONOMY

ART + DESIGN + ARCHITECTURE

SCIENCE + TECHNOLOGY

CROSSOVER

DOCTORAL DISSERTATIONS