Connections between graph theory and cryptography

Natalia Tokareva

G2C2: Graphs and Groups, Cycles and Coverings
September, 24–26, 2014. Novosibirsk, Russia

Introduction to cryptography

Terminology

Cryptography is the **scientific** and **practical** activity associated with developing of cryptographic security facilities of information and also with argumentation of their cryptographic resistance.

Plaintext is a secret message. Often it is a sequence of binary bits.

Ciphertext is an encrypted message.

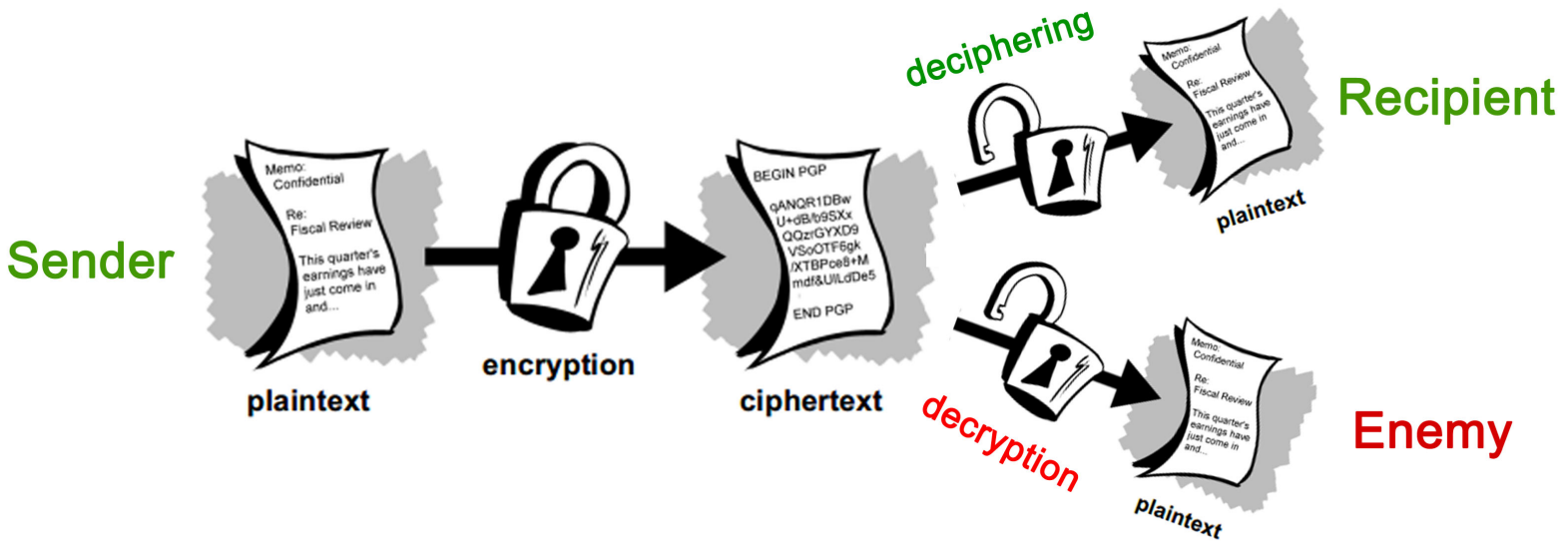
Encryption is the process of disguising a message in such a way as to hide its substance (the process of transformation plaintext into ciphertext by virtue of **cipher**).

Cipher is a family of invertible mappings from the set of plaintext sequences to the set of ciphertext sequences. Each mapping depends on special parameter — a key. Key is **removable part** of the cipher.

Terminology

Deciphering is the process of turning a ciphertext back into the plaintext that realized **with known key**.

Decryption is the process of receiving the plaintext from ciphertext **without knowing the key**.



Terminology

Cryptography is the **scientific** and **practical** activity associated with developing of cryptographic security facilities of information and also with argumentation of their cryptographic resistance.

Cryptanalysis is the **scientific** and **practical** activity of analysis of cryptographic algorithms with the goal to obtain estimations of their cryptographic resistance.

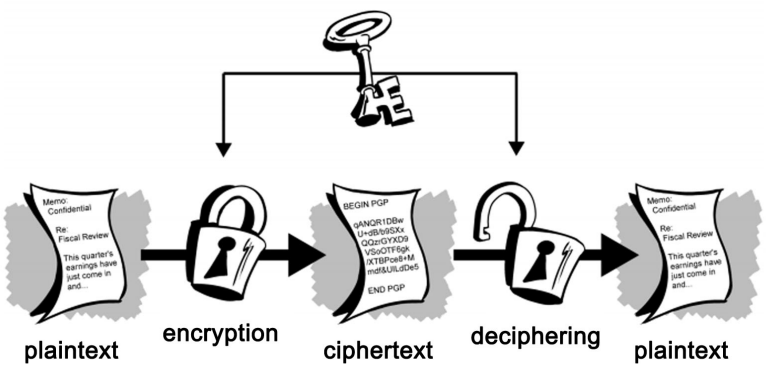
Cryptology is the concept combining both cryptography and cryptanalysis.

Cryptographic goals

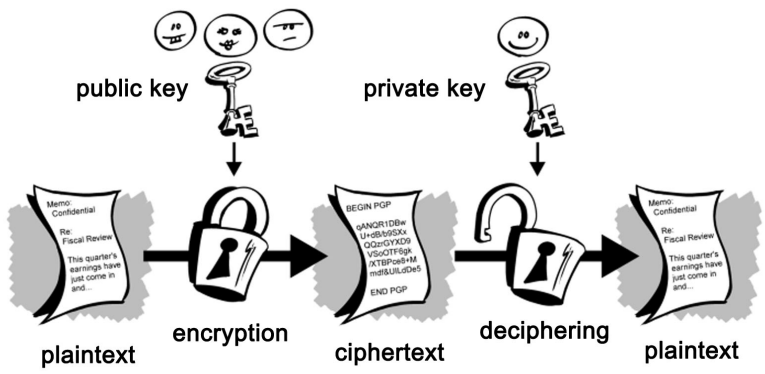
1) Confidentiality is a service used to keep the content of information from all but those authorized to have it.
2) Data integrity is a service which addresses the unauthorized alteration of data.
3) Authentication is a service related to identification. This function applies to both entities and information itself. Two parties entering into a communication should identify each other.
4) Non-repudiation is a service which prevents an entity from denying previous commitments or actions.

Types of cryptographic algorithms

- Symmetric algorithms (conventional algorithms) are algorithms where the encryption key can be calculated from the decryption key and vice versa.
- Public-key algorithms (asymmetric algorithms) are designed so that the key used for encryption (**public key**) is different from the key used for decryption (**private key**).



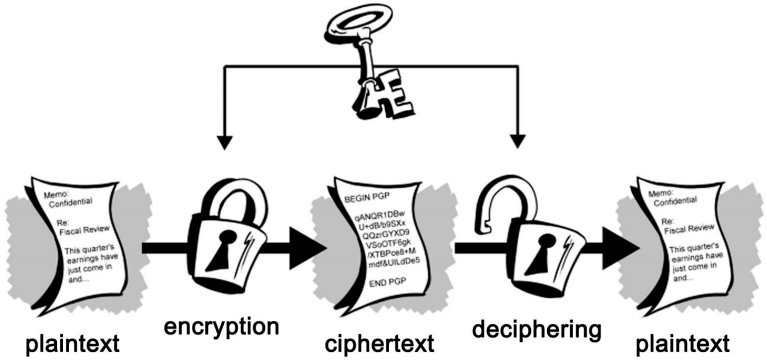
Symmetric cryptography



Public-key cryptography

Main principles

Symmetric algorithms are algorithms in which the encryption key can be calculated from the decryption key and vice versa.



Symmetric cryptography

- Usually **the encryption key = the decryption key**.
- The sender and receiver should agree on a key **before secure communication**.
- Security of a symmetric algorithm **is guaranteed by the key**; divulging the key means that anyone could encrypt and decrypt messages. As long as the communication needs to remain secret, **the key should remain secret**.

Block and stream ciphers

Symmetric algorithms can be divided into two categories:

1) First type operate on the plaintext of a **single bit** (or sometimes byte) at a time; these algorithms are called stream ciphers.

   Examples. A5/1, A2/2, Grain, Trivium, Achterbahn-128/80 and others.

2) Second type operate on the plaintext given by a **group of bits**. The groups of bits are called blocks, and the algorithms are called block ciphers.

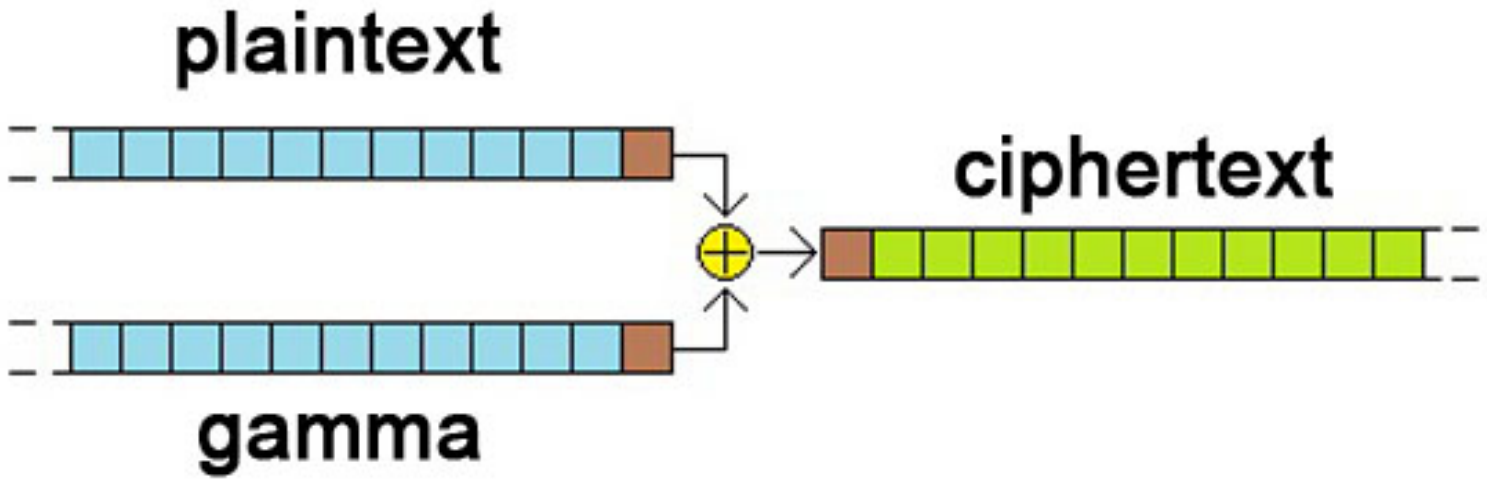
   Examples. DES, GOST 28147-89, AES, CAST-128, SMS4 and others.

Stream ciphers

A stream cipher generates a **pseudorandom sequence of bits** that should be XORed with a binary bit sequence of a plaintext. Such a sequence is called a gamma.

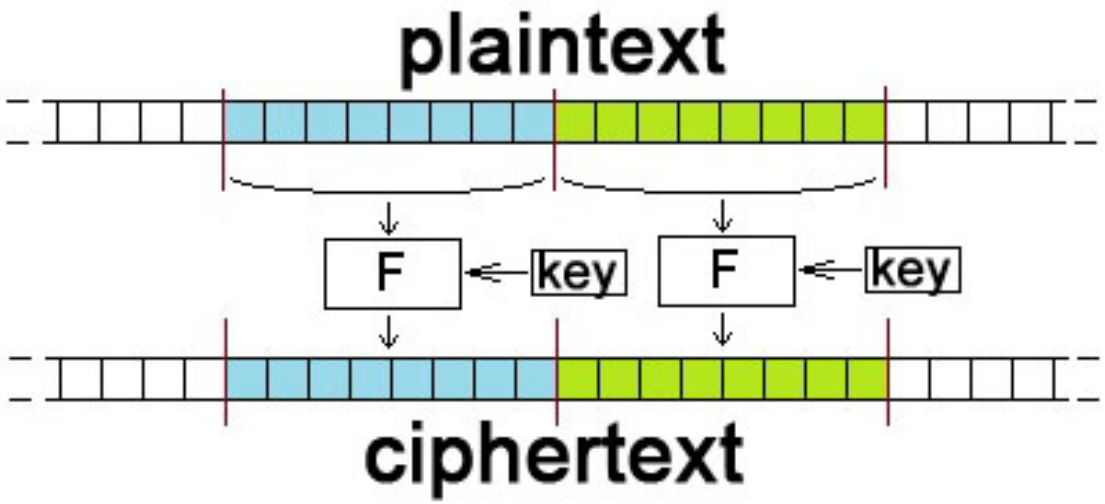
$$\text{plaintext} \oplus \text{gamma} = \text{ciphertext}$$

The **system's security** depends only on the **gamma**.



Block ciphers

The plaintext is **divided into block of bits** (typical block sizes are 64, 128, 256). Then all blocks are separately encrypted by a cipher that is a some mapping F depending on a secret key.



Connections to graph theory

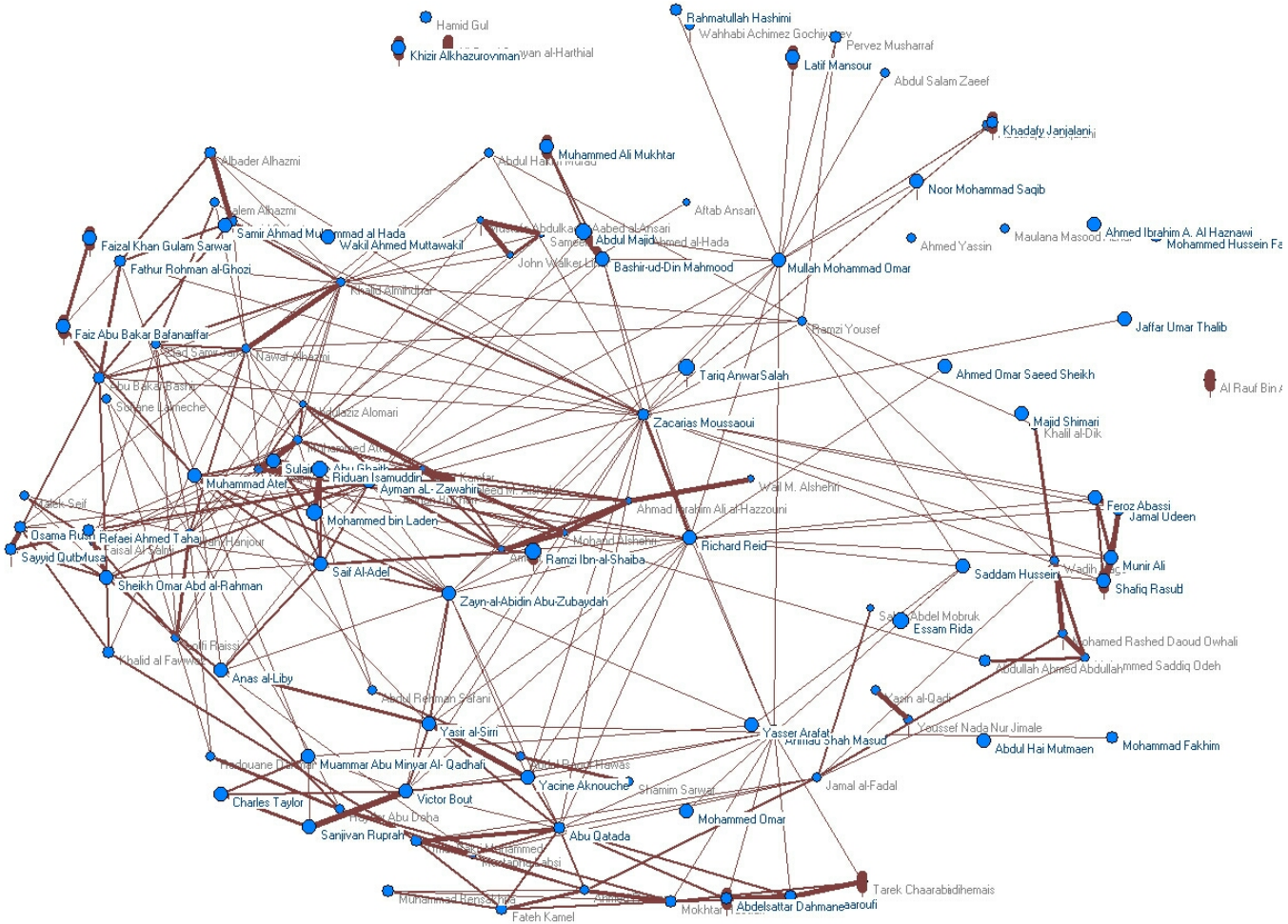
Modern cryptography is highly connected with discrete mathematics. Many cryptographic algorithms such as RSA, ElGamal, elliptic curve methods, symmetric ciphers AES, CAST, Grain, several stream ciphers, hash functions, statistical methods of cryptanalysis, cryptographic protocols, etc. are directly based on mathematical results.

In this talk we discuss connections between cryptographic methods and graph theory. We consider such topics as:

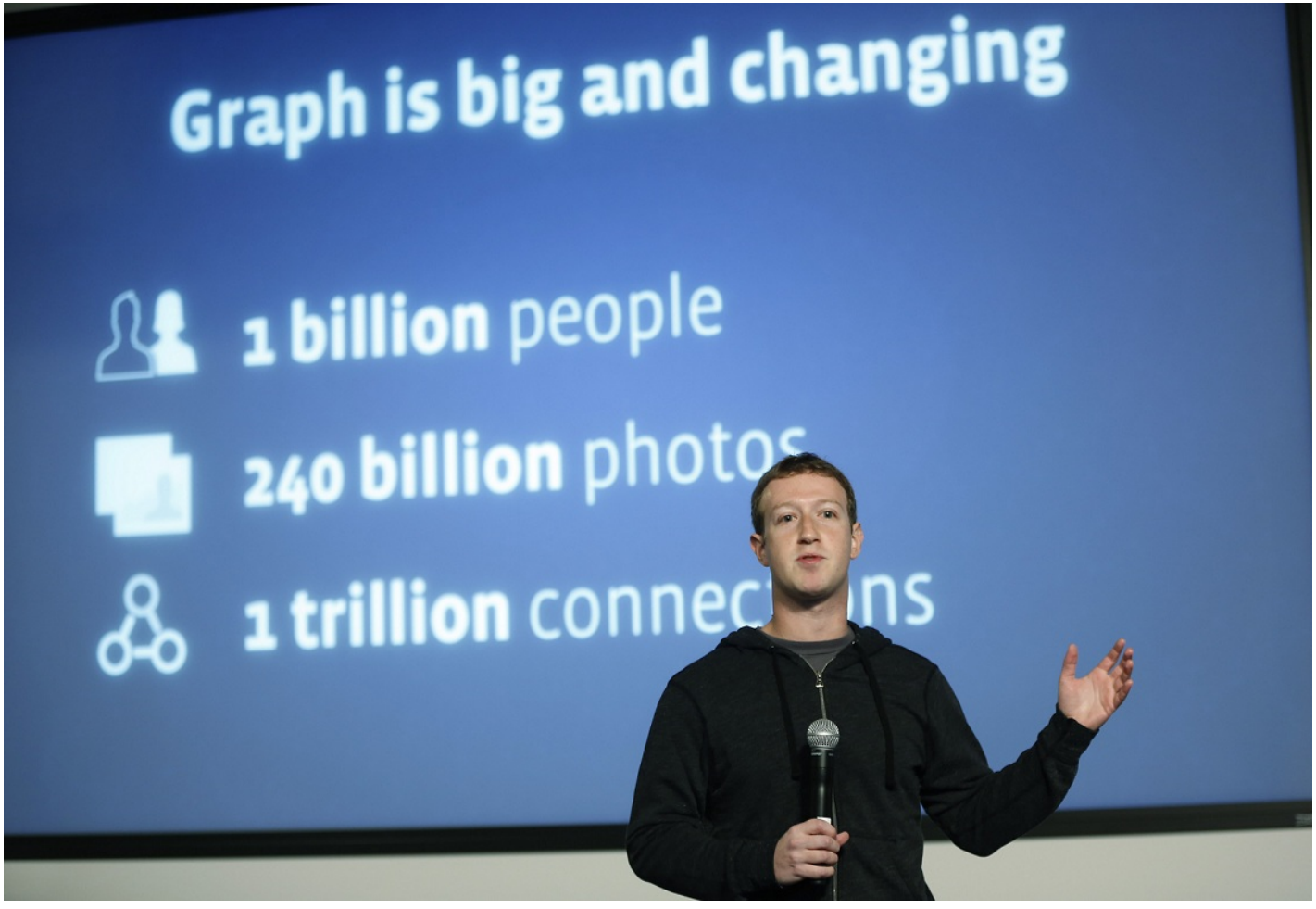
- sparse graphs, social networks and mobile security systems;
- hash functions, expander and random graphs;
- cycles of large period and linear recurrent sequences;
- cryptographic Boolean functions and graphs;
- metrical properties of Cayley graphs and bent functions.

Sparse graphs, social networks and mobile security systems

Sparse graphs, social networks and mobile security systems



Sparse graphs, social networks and mobile security systems



Sparse graphs, social networks and mobile security systems

A sparse graph is a graph in which the number of edges is much less than the possible number of edges.

A.Lee, I.Streinu (2008), L.Theran (2009) defined a sparse graph like this. A graph is (k,ℓ)-sparse if every nonempty subgraph with n vertices has at most $kn - \ell$ edges. A graph is (k,ℓ)-tight if it's (k,ℓ)-sparse and has exactly $kn - \ell$ edges.

Some details:

- trees are exactly the (1,1)-tight graphs;
- forests are exactly the (1,1)-sparse graphs;
- the facts that any planar graph with n vertices has at most $3n - 6$ edges, and that any subgraph of a planar graph is planar, together imply that the planar graphs are (3,6)-sparse. However, not every (3,6)-sparse graph is planar;
- I.Streinu and L.Theran showed that testing (k,ℓ)-sparsity may be performed in polynomial time when k and ℓ are integers and $0 \leqslant \ell < 2k$.

A problem for a mathematician. Studying sparse graphs L.Lovász mentions in his paper "Graph homomorphisms: Open problems" (2008) the following problem:

"**Problem 42.** Suppose that instead of exploring the neighborhood of a single random node, we could select two random nodes and test simple quantities associated with them, like distance, maximum flow, electrical resistance. What information can be gained by such tests? Is there a "complete" set of tests that would give enough information to determine the global structure of the graph to a reasonable accuracy?"

A problem for a mathematician. In the paper "Extremal results in sparse pseudorandom graphs" D.Conlon, J.Fox, Y.Zhao discuss several problems for sparse graphs.

For instance, analog of regularity lemma for sparse graphs was proven by Y.Kohayakawa (1997) and by V.Rödl; analog of counting lemma is still not proven.

Regularity lemma. "Roughly speaking, it says that the vertex set of every graph can be partitioned into a bounded number of parts so that the induced bipartite graph between almost all pairs of parts is pseudorandom."

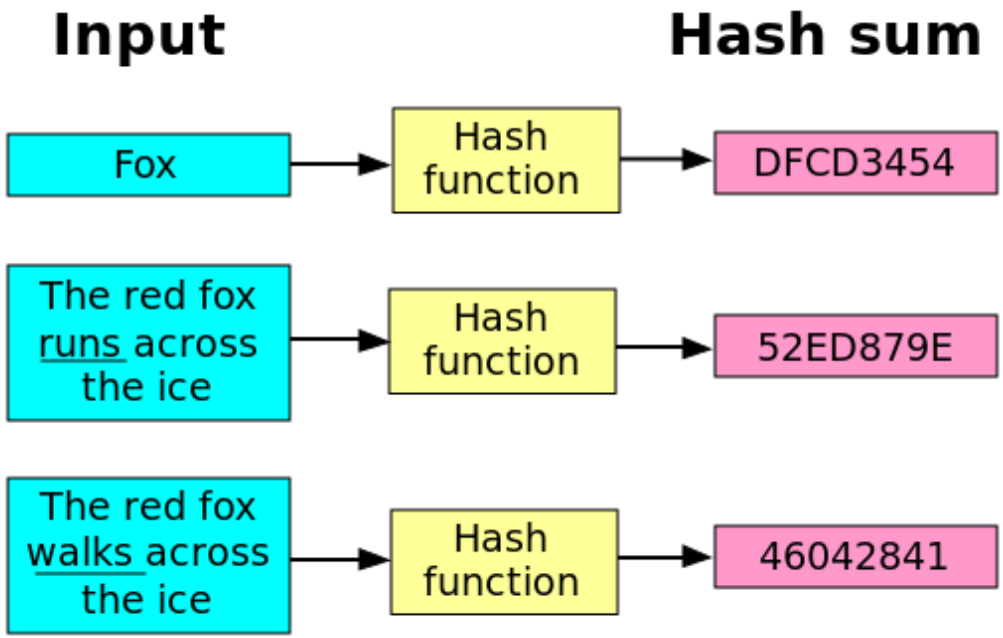
Counting lemma. "Roughly speaking, it says that the number of embeddings of a fixed graph H into a pseudorandom graph G can be estimated by pretending that G were a genuine random graph."

Hash functions, expander and random graphs

Hash functions and sparse graphs

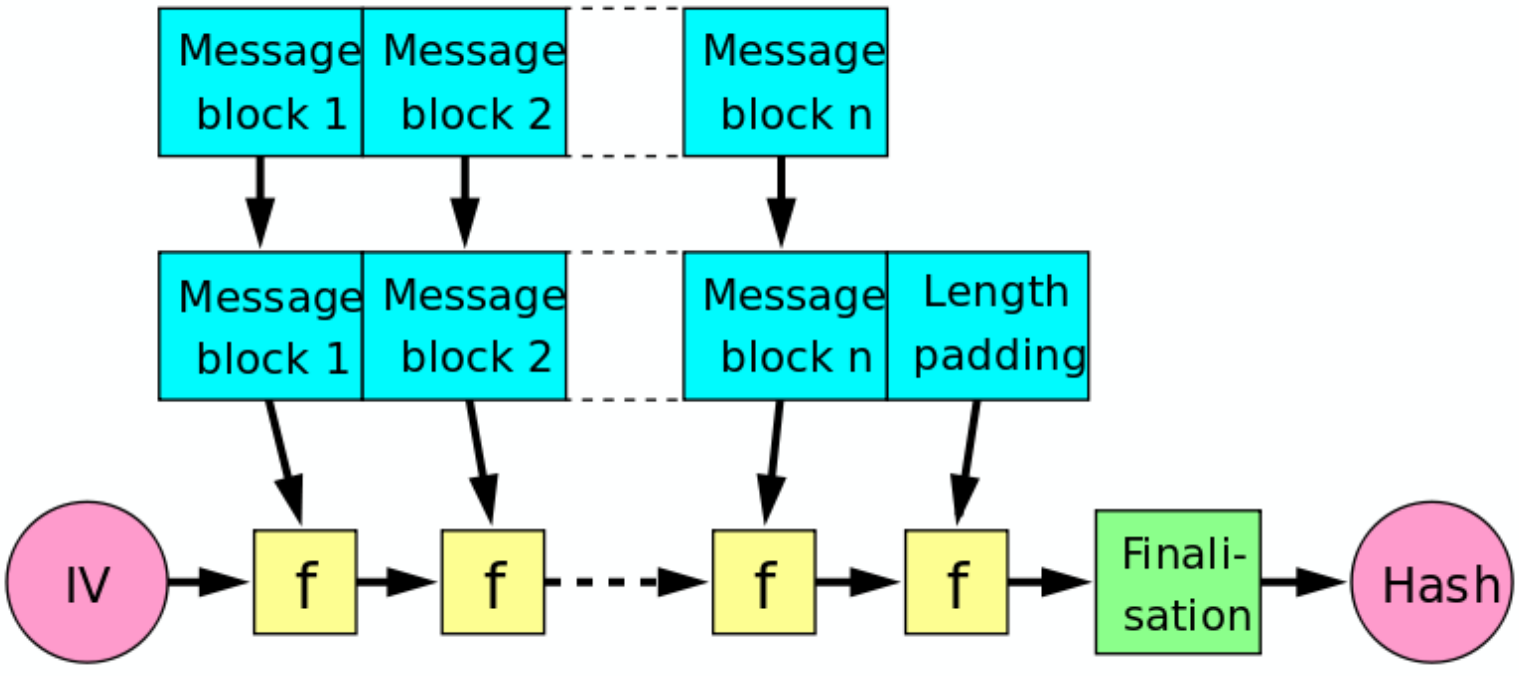
A hash function is an arbitrary function that can be used to map digital data of any size to digital data of a fixed size.

Slight differences in input data have to produce big differences in output data. Difficulties in finding of an preimage and a collision.



Hash functions and sparse graphs

Usually hash functions are constructed by steps.



Expander graphs and hash functions

Sparse graphs are used for constructing cryptographically resistant hash functions.
An expander graph is a sparse graph that has strong connectivity properties, quantified using vertex, edge or spectral expansion.

Informally, expander graphs are graphs in which the neighbor set of any "not too large" subset of vertices contains many new vertices.

Application for constructing hash functions: the input to the hash function is used as directions for walking around a graph, and the ending vertex is the output of the hash function.

An idea...

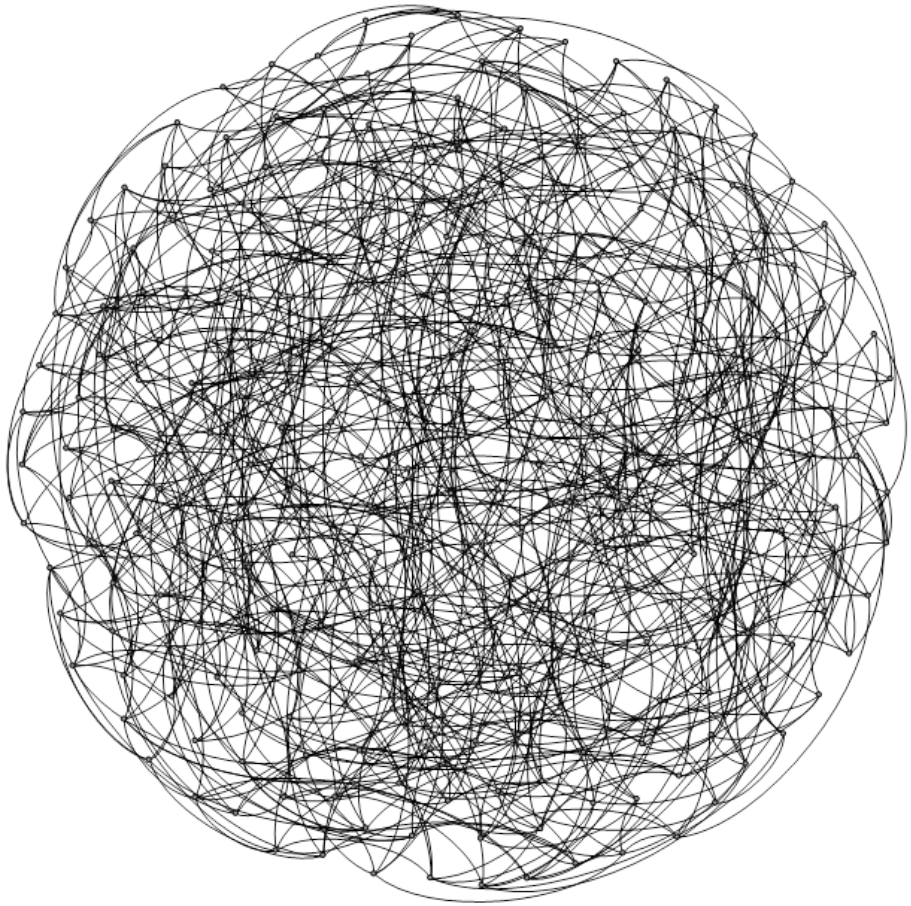
“Nel mezzo del cammin di nostra vita,
mi ritrovai per una selva oscura,
che’ la diritta via era smarrita.”

“When I had journeyed half of our life’s way,
I found myself within a shadowed forest,
for I had lost the path that does not stray.”

Dante Alighieri “The Divine Comedy”

After even several steps in an expander graph you don’t know where are you... It’s good for crypto since provides a very good **mixing**.

An example from the series of Ramanujan graphs



Examples of hash functions based on expander graphs

D.Charles, E.Goren, K.Lauter "Cryptographic hash functions from expander graphs" (2007).

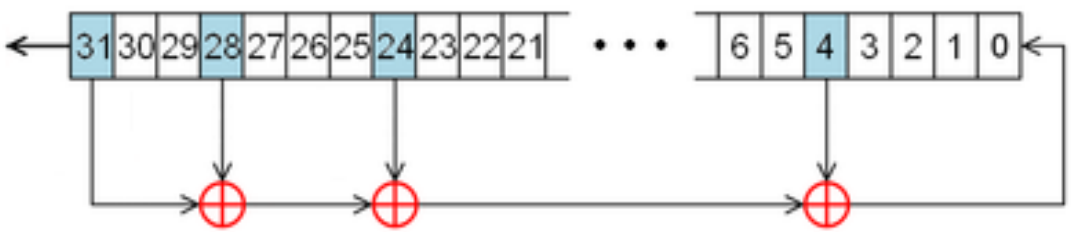
They proposed constructing provable collision resistant hash functions from expander graphs in which finding cycles is hard. As examples, they have investigated two specific families of optimal expander graphs for provable collision resistant hash function constructions: the families of Ramanujan graphs constructed by Lubotzky-Phillips-Sarnak and Pizer respectively.

Hash functions and random graphs

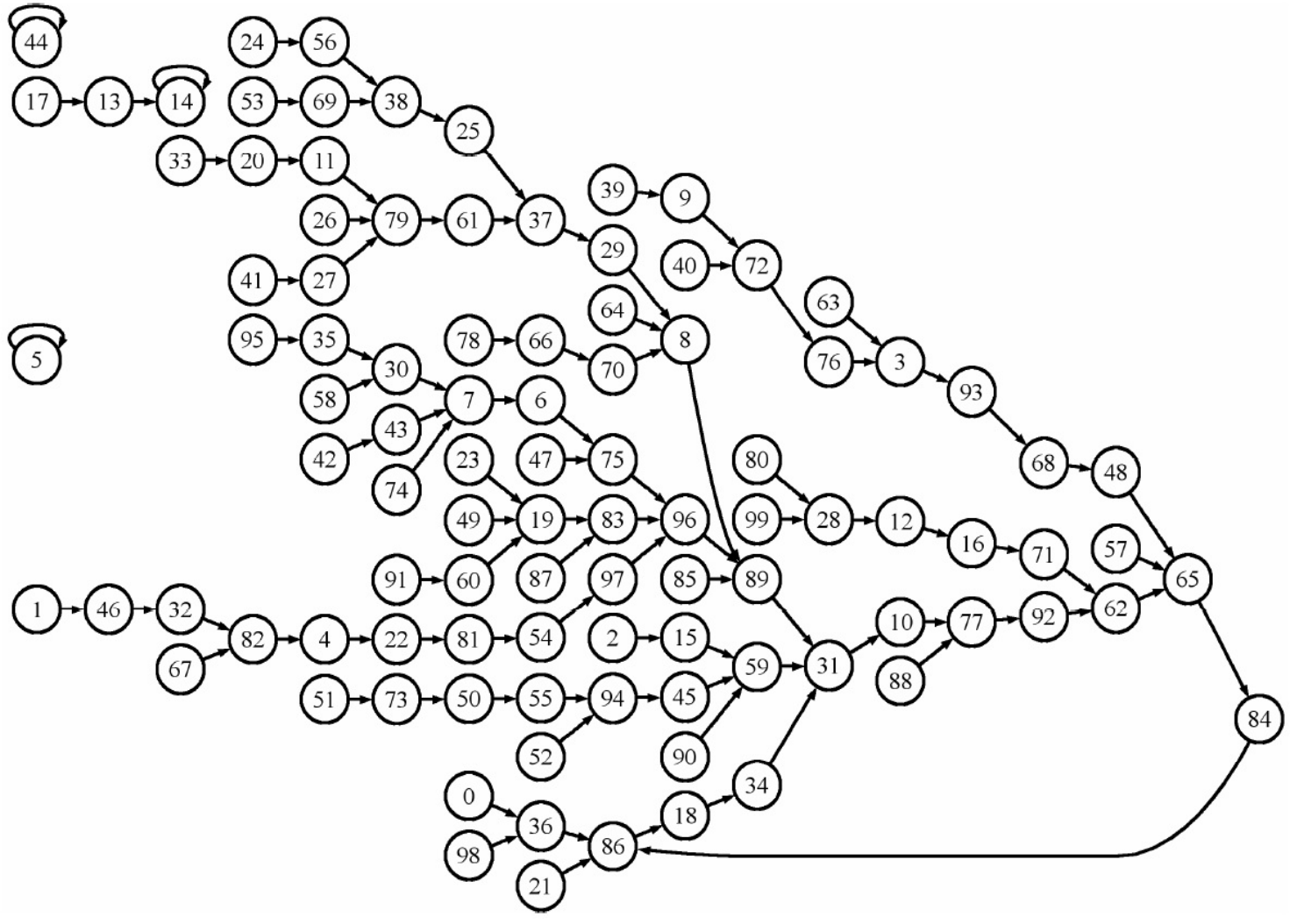
Let f be a random function defined on $M = \{0, 1, 2, \ldots, N-1\}$, namely $f : M \to M$.
A random graph is an oriented graph associated to f:

Vertices: $0, 1, 2, \ldots N-1$;
Edges: $x \to f(x)$.

In cryptography we can meet a random graph as the state graph of a some stream generator or as the graph of a hash function.



Here is the Linear Feedback Shift Register (LFSR).


44
24
56
17
13
14
53
69
38
25
33
20
11
39
9
26
79
61
37
29
40
72
41
27
64
63
95
35
78
66
8
76
3
5
30
70
93
58
7
6
68
48
43
80
42
23
47
75
99
28
12
16
71
57
74
49
19
83
96
91
60
87
97
85
89
65
1
46
32
10
77
92
62
82
4
22
81
54
2
15
59
31
67
88
51
73
50
55
94
45
84
52
90
0
34
36
18
98
86
21

A problem for a cryptographer: How to find a preimage?
I. e. for a given y to find some x, such that $f(x) = y$?
A problem for a cryptographer: How to find a collision?
I. e. distinct numbers x, x' such that $f(x) = f(x')$?

There is an algorithm of Floyd: using two pointers (one runs at normal speed, another at double speed, until they collide).

When the path has n vertices and the tail is short, algorithm of Floyd requires about $3n$ steps. When the cycle is short, the fast pointer can traverse it many times without noticing.
Using additional memory it is possible to overcome these difficulties.

A problem for a mathematician: To propose distinct (effective?) algorithms for finding collisions in pseudorandom graphs.

Cycles of large period and linear recurrent sequences

Cycles of large period and linear recurrent sequences

Linear recurrent sequences (LRS) form the most famous base for pseudo-random generators.

LRS is produced by **Linear feedback shift register** (LFSR).

LFSR and its generalization is widely used as a component in a stream cipher (for instance, A5/1, Grain, they will be considered later) and pseudo-random generators.

LFSR can produce a sequence with a **good** minimal period from the small initial state and its generalizations allow to sufficiently increase linear complexity of the output sequence.

Such sequences have **good** statistical properties.

Linear recurrent sequences

$\mathbf{u} = (u_0, u_1, u_2, \ldots)$, $u_i \in \mathbb{Z}_2$ is an infinite **binary** sequence.
An infinite sequence $\mathbf{u}$ is called linear recurrent (LRS), if for some $a_0, \ldots, a_{n-1} \in \mathbb{Z}_2$ it is right

$$u_{i+n} = a_0 u_i \oplus \ldots \oplus a_{n-1} u_{i+n-1} \ \forall i \in \mathbb{N}_0.$$

The expression above is also called linear recurrence relation of order n. Note that a linear recurrent sequence can be produced by linear recurrence relations of different orders.

The following polynomial is called a characteristic polynomial of the sequence $\mathbf{u}$: $c(\lambda) = \lambda^n \oplus a_{n-1}\lambda^{n-1} \oplus \ldots \oplus a_1\lambda \oplus a_0$.

- Minimal polynomial of a LRS is its characteristic polynomial of the minimal degree. Denote it by $\mu_{\mathbf{u}}(\lambda)$.
- Linear complexity of a LRS is the degree of its minimal polynomial. Denote it by $\ell(\mathbf{u})$.

Note that the minimal polynomial of a LRS is unique.

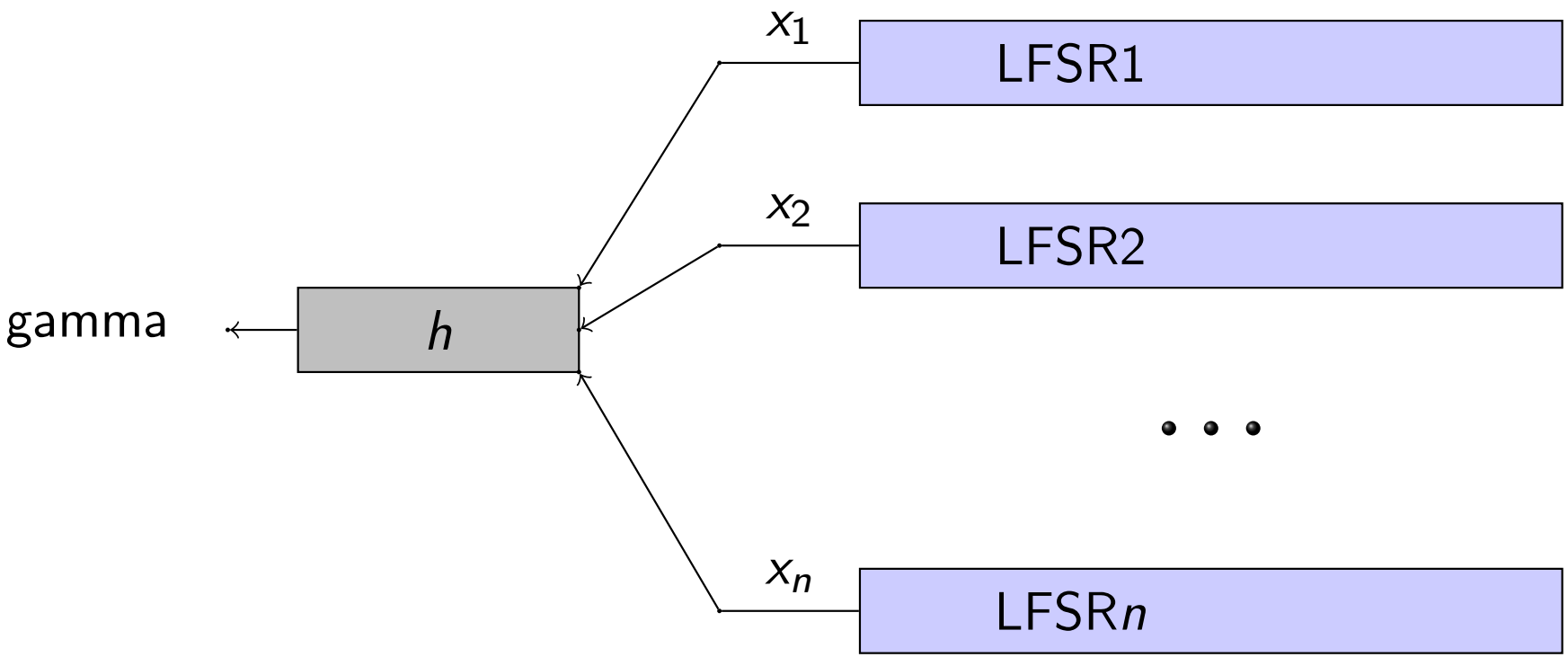
A sequence $\mathbf{u}$ is called

- ultimately periodic, if for some natural numbers T and s it holds $u_{i+T} = u_i,\ i \in \mathbb{N}_0,\ i \geq s$.
- periodic, if $s = 0$.

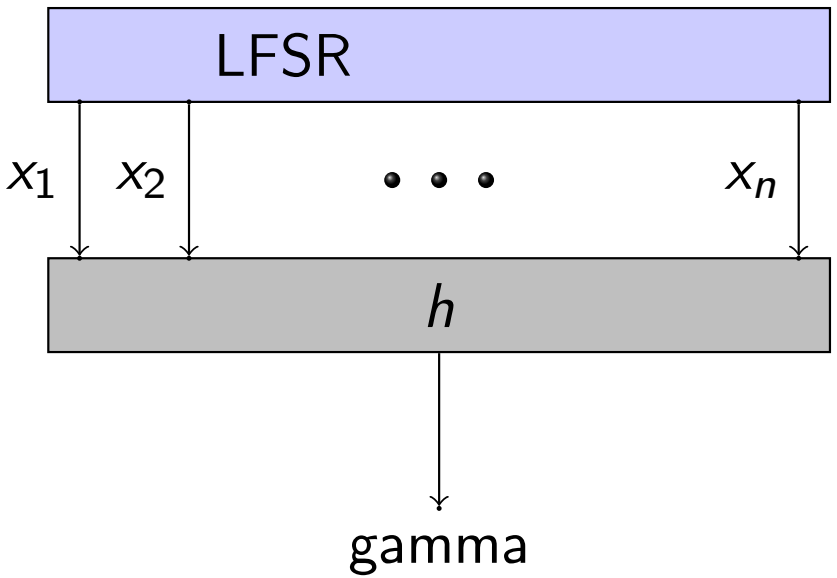
The number T is called period. Minimal possible period is called the minimal period and denoted by $per(u)$.
The minimal period of a sequence divides any its period.
It is known that any LRS $\mathbf{u}$ is periodic and $per(\mathbf{u}) \leq 2^{\ell(\mathbf{u})} - 1$, where $\ell(\mathbf{u})$ is linear complexity of $\mathbf{u}$.

A problem for a cryptographer: How to construct LRS such that for any initial values $u_0, u_1, \ldots, u_{n-1}$ the sequence $\mathbf{u}$ has high linear complexity and period? Large cycles of the state graph?

The same for several more complicated generators?



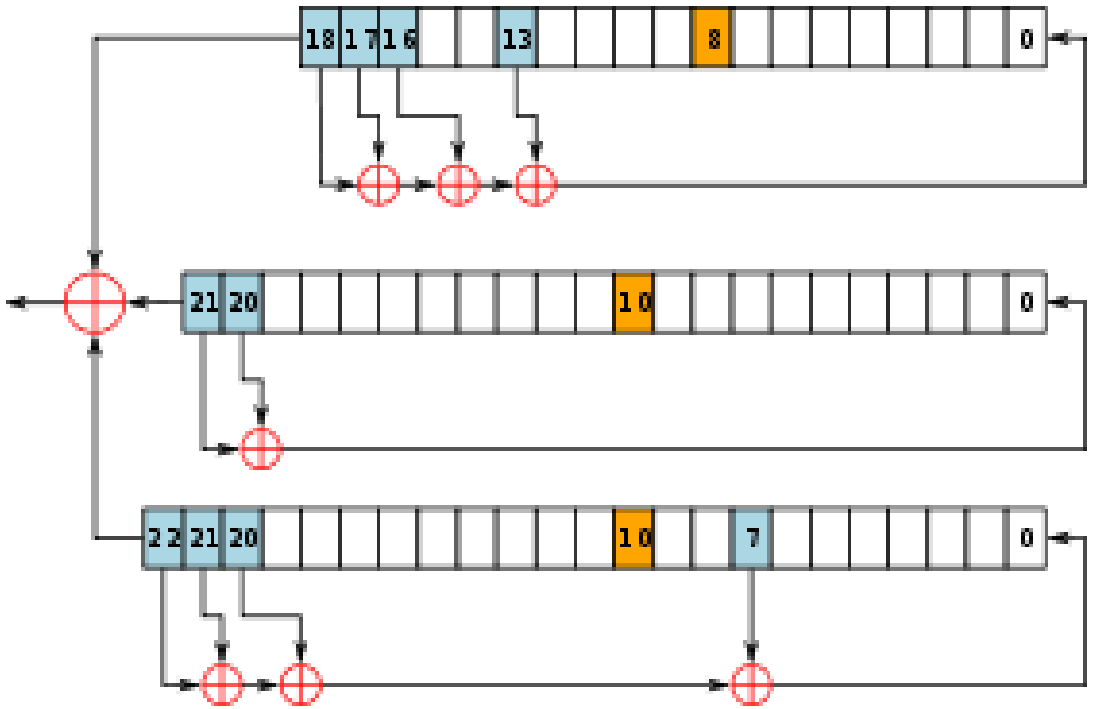
Combining model



Filtering model

A problem for a mathematician:

How to construct a function $f : M \to M$, $M = \{0, 1, 2, \ldots, N-1\}$, under several conditions, such that its associated graph has large cycles with short tails.



Cryptographic Boolean functions and graphs

Nonlinearity

Nonlinearity of a Boolean function f in n variables is the value N_f that is equal to the Hamming distance between function f and the set $\mathcal{A}_n$ of all affine functions in n variables, i.e. $N_f = dist(f, \mathcal{A}_n)$.

For arbitrary Boolean function f in n variables there is an upper bound for nonlinearity $N_f \leqslant 2^{n-1} - 2^{(n/2)-1}$.

A Boolean function is called maximal nonlinear if its nonlinearity achieves the maximum possible value. When n is **even** this value equals $2^{n-1} - 2^{(n/2)-1}$ and such function is called bent.

Bent functions if used for constructing S-boxes in block ciphers make them extremely resistant to linear cryptanalysis. Examples: DES, CAST, AES...

Bent functions

There is an approach to classification of bent functions proposed by A. Bernasconi, B. Codenotti, J. M. VanderKam (1999).

Let f be a Boolean function in n variables. Denote by $\mathrm{supp}(f)$ its support, i. e. the set of all binary vectors of length n on which function f takes the value 1. Consider the Cayley graph $G_f = G(\mathbb{Z}_2^n, \mathrm{supp}(f))$ of a Boolean function f. All vectors of length n are vertices of the graph. There is an edge between two vertices x and y if vector $x \oplus y$ belongs to $\mathrm{supp}(f)$.

A regular graph G is called strongly regular if there exist nonnegative integers λ, μ such that for any vertices x, y the number of vertices incident to x and y both is equal to λ or μ and it depends on the presence or absence of the edge between x and y.

Bent functions

Theorem. A Boolean function f is bent if and only if the graph Cayley G_f is strongly regular and $\lambda = \mu$.

Namely, graph Cayley G_f of a bent function f in n variables is strongly regular with $(2^n, 2^{n-1} \pm 2^{\frac{n}{2}-1}, \lambda = \mu = 2^{n-2} \pm 2^{\frac{n}{2}-1})$.

A problem for a mathematician: Propose new constructions of strongly regular graphs on 2^n vertices with $\lambda = \mu$.

Graph of minimal distances of bent functions

Let GB_n be the graph on bent functions in n variables as vertices with edges between functions that are on the minimal possible distance $2^{n/2}$ each other.

Nikolay Kolomeec studies such a graph in his PhD. He proved that

- Degrees of a vertex from GB_n is not more than $2^{n/2}\prod_{i=1}^{n/2}(2^i+1)$.
- Since for every even $n \geqslant 14$ there are found non weakly normal bent functions (A.Canteaut, et al. 2006), graph GB_n is not connected if $n \geqslant 14$. It is proven (N.Kolomeec, 2014) that GB_n is connected for $n = 2, 4, 6$.

A concrete problem for a mathematician: Is the graph GB_n connected / disconnected if $8 \leqslant n \leqslant 12$?

A problem for a mathematician: Let GB'_n be the graph obtained from GB_n after elimination of all pendant vertices (corresp. to non weakly normal bent func.s). Is GB'_n connected for all even $n \geqslant 2$?

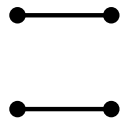
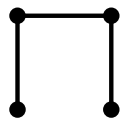
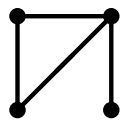
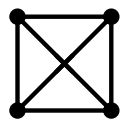
Graph classification of bent functions

In 2012 E.P.Korsakova obtained the graph classification of bent functions in 6 variables.

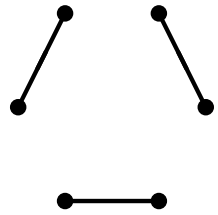
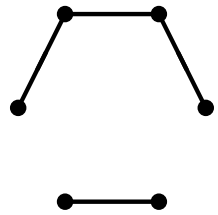
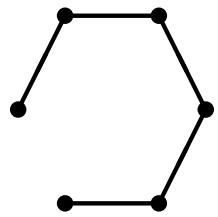
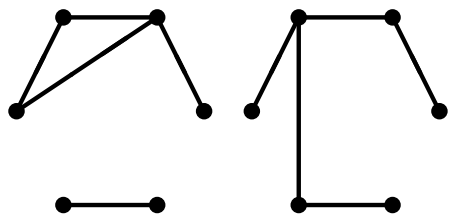
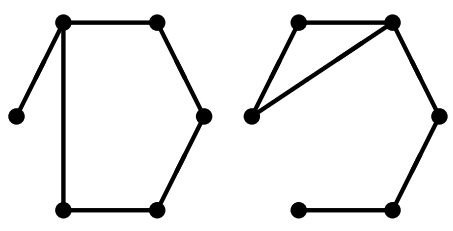
Graph of a quadratic function:
Vertices — variables x_i;
Edges — pairs (i, j) when $x_i x_j$ can be found in ANF of f.

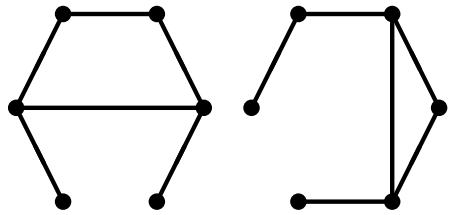
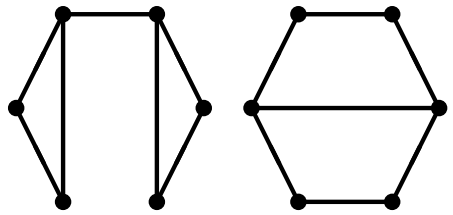
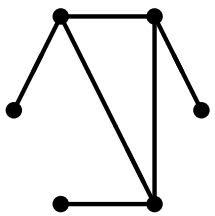
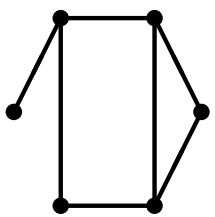
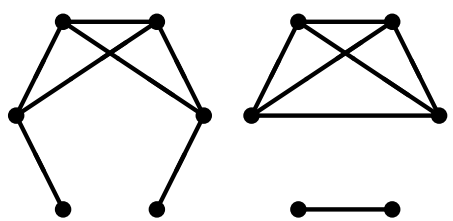
Quadratic Boolean functions are graph equivalent if their graphs are isomorphic.

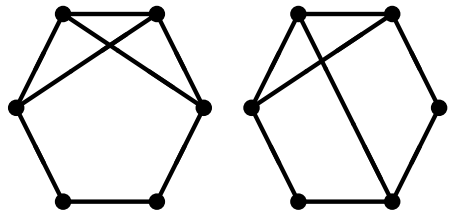
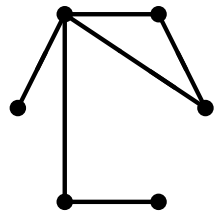
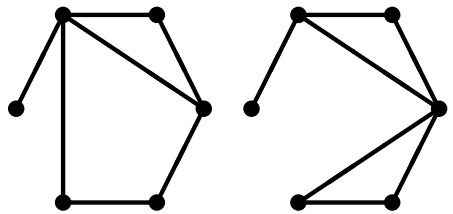
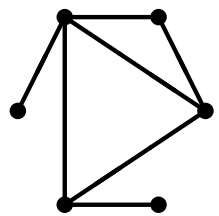
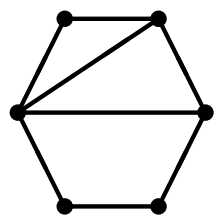
Bent functions in 4 variables (O.Rothaus, 1976):

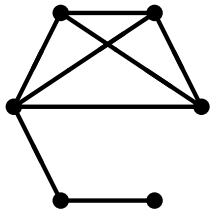
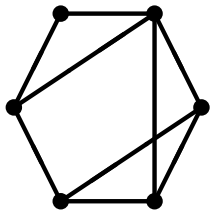
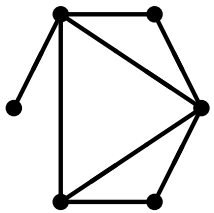
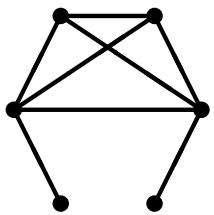
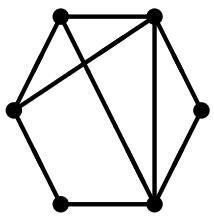
№	Type	Graph
1	1 1 1 1	
2	2 2 1 1	
3	3 2 2 1	
4	3 3 3 3	

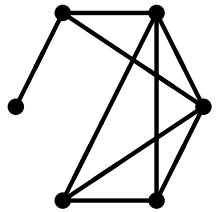
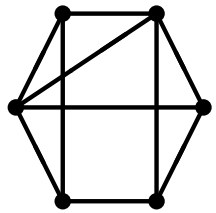
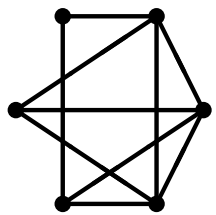
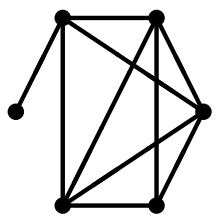
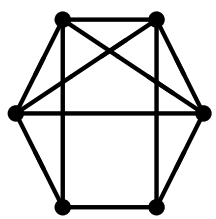
E.P.Korsakova (2013) has proved that there are 44 graph nonequivalent quadratic bent functions in 6 variables (and 37 types).

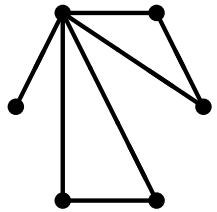
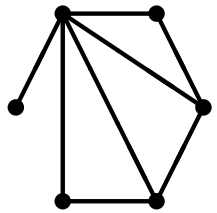
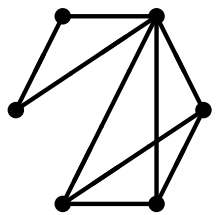
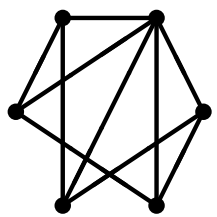
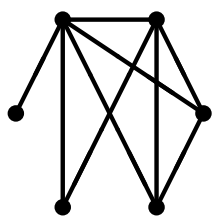
1	1 1 1 1 1 1	100000000100001	
2	2 2 1 1 1 1	100001000100001	
3	2 2 2 2 1 1	100001000100101	
4	3 2 2 1 1 1	110001000100001 100001001100001	
5	3 2 2 2 2 1	100001001100101 110001000100101	

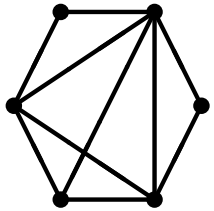
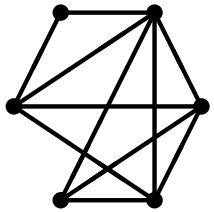
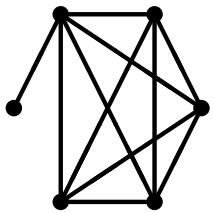
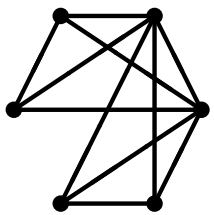
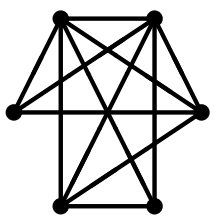
6	3 3 2 2 1 1	101011000100100 100001000110101	
7	3 3 2 2 2 2	100011001110100 101011000100101	
8	3 3 3 1 1 1	100001010110001	
9	3 3 3 2 2 1	100001001100111	
10	3 3 3 3 1 1	110011100100100 111001100100001	

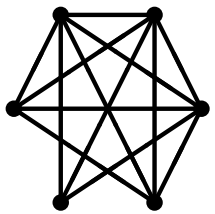
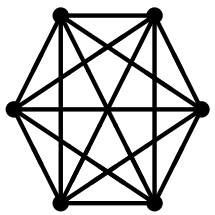
11	3 3 3 3 2 2	110011100100101 110011010100101	
12	4 2 2 2 1 1	100001101100001	
13	4 3 2 2 2 1	100001101100101 100001100100111	
14	4 3 3 2 1 1	100001101100011	
15	4 3 3 2 2 2	111011000100101	

16	4 3 3 3 2 1	111011100100001	
17	4 3 3 3 3 2	110011000110111	
18	4 4 3 2 2 1	100001101100111	
19	4 4 3 3 1 1	111011100100100	
20	4 4 3 3 2 2	110011010110101	

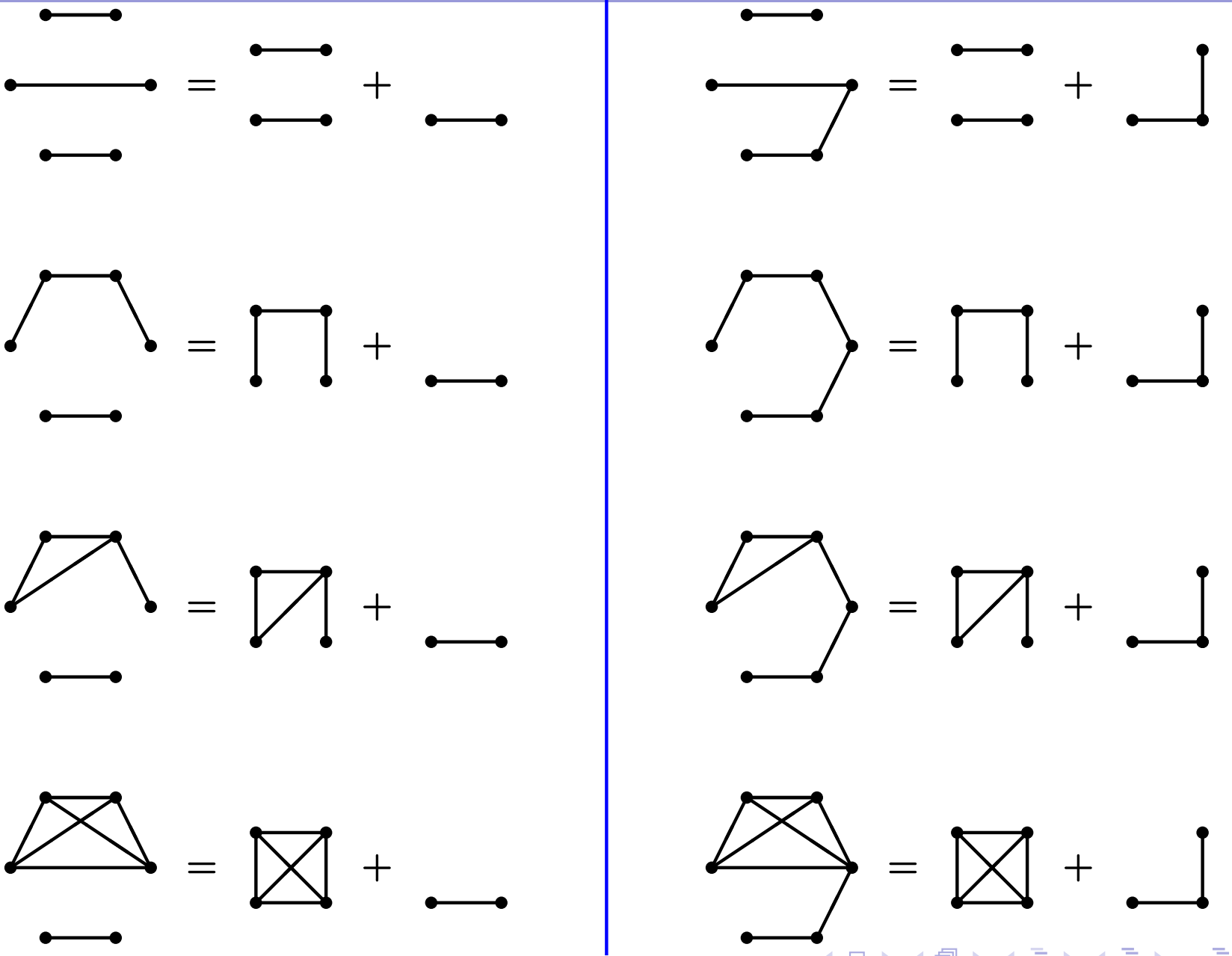
21	4 4 3 3 3 1	100001100111111	
22	4 4 3 3 3 3	111011001110101	
23	4 4 4 3 3 2	011101001110111	
24	4 4 4 4 3 1	100001101111111	
25	4 4 4 4 3 3	111011101110101	

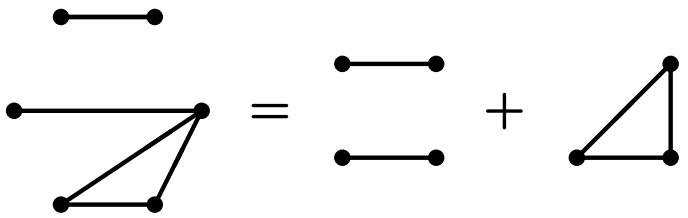
26	5 2 2 2 2 1	100001111100001	
27	5 3 3 2 2 1	100001111100101	
28	5 3 3 3 2 2	110001000111111	
29	5 3 3 3 3 3	110101001111110	
30	5 4 3 3 2 1	100001111111100	

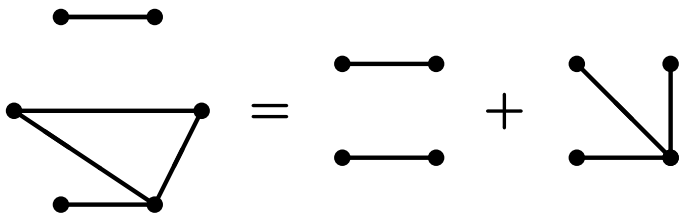
31	5 4 4 3 2 2	110111000111101	
32	5 4 4 4 3 2	111101000111111	
33	5 4 4 4 4 1	100001111111111	
34	5 5 3 3 3 3	111001100111111	
35	5 5 4 4 3 3	111001111111011	

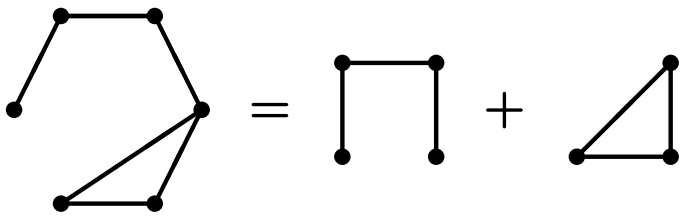
36	5 5 5 4 4 3	111101111111110	
37	5 5 5 5 5 5	111111111111111	

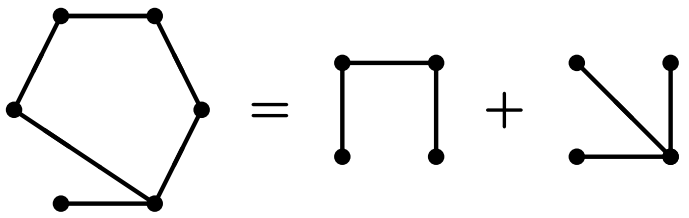
Based on this classification E.P.Korsakova has obtained several ideas for iterative constructions of bent functions.

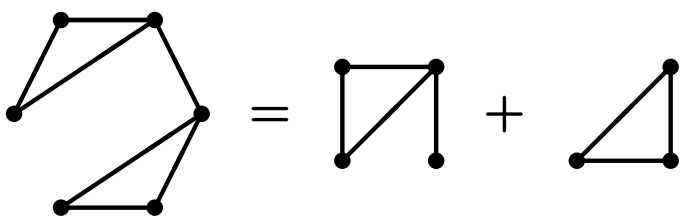


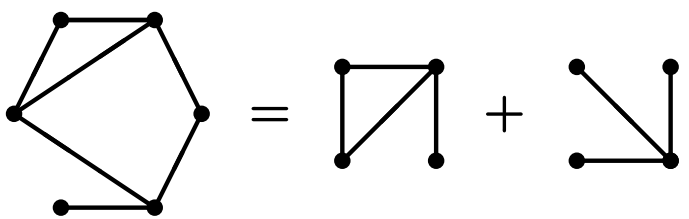


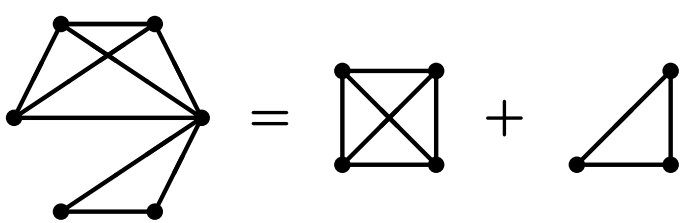


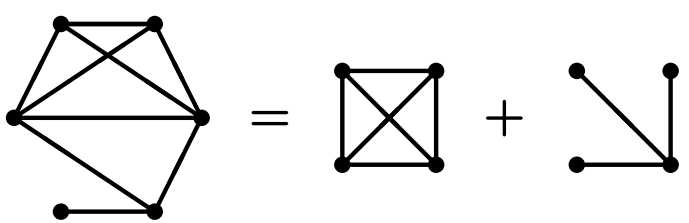












Thank you!