Three problems in algebraic combinatorics

Berndsen, J.

Award date:
2012

Eindhoven University of Technology
Department of Mathematics and Computer Science

Master's thesis

Three problems in algebraic combinatorics

Jochem Berndsen

Eindhoven, Aug 2012

Supervisors:
dr. A. Blokhuis
prof.dr. A.E. Brouwer

Abstract

We consider three problems in algebraic combinatorics.

The first one is related to spectral graph theory. Let $\lambda_1 \geq \cdots \geq \lambda_n$ be the Laplacean eigenvalues of a finite, simple, undirected graph Γ. Then we prove that for all k we have $\sum_{i=1}^{k} \lambda_i \leq |E\Gamma| + \binom{k+1}{2}$ for some cases: if Γ is split, a cograph or regular. Furthermore we also show that this bound is sharper if Γ is nonsplit.

The second problem is the extended Alon–Tarsi conjecture. This conjecture states that the number of even and odd Latin squares of order n are different for all n. We show this for the case $n \leq 10$, the case where n is $p-1$, p or $p+1$ for odd prime p, or if $n = p+2$ where $n < 2^{20}$ and $n \neq 234781$.

The third problem is the Gossip problem. Suppose that there are n people, each knowing a secret. How many telephone calls are needed to distribute all secrets to all participants? It is known that the answer is $2n-4$. We do some more calculation in the gossip graph, which is the graph (V,A), where V is the set of distributions of knowledge, and A the telephone calls.

Contents

Chapter 1

Spectral graph theory

This chapter contains joint work with Mayank and the results can also be found in [24].

1.1 Spectrum of graphs

Spectral graph theory studies the relation between graphs and the eigenvalues of some of their associated matrices. So let Γ be a simple graph. We will only consider finite graphs. We can associate to Γ its *adjacency matrix* A. Its columns and rows are indexed by the vertices of Γ, and we have

$$A_{uv} = \begin{cases} 1 & \text{if } u \sim v, \\ 0 & \text{if } u \not\sim v \text{ or } u = v. \end{cases}$$

The eigenvalues of A are called the *eigenvalues* (or the *ordinary eigenvalues* if we want to distinguish them from other types of eigenvalues) of the graph Γ. These do not depend on the ordering of the vertices of Γ, since similar matrices have the same spectrum. If Γ is undirected, then A will be symmetric and hence will have real eigenvalues. From now on we will assume that Γ is undirected.

Another important matrix associated to the graph Γ is the *Laplacean matrix* L. It is defined as $L = \mathrm{diag}(\deg(v_1), \ldots, \deg(v_n)) - A$. Since it has constant row sum 0, it is singular. (The matrix A may or may not be singular.) Equivalently, we could as well define $L = NN^\top$, where N is an arbitrarily oriented vertex-edge incidence matrix, i.e. $N_{ie} = -1$ and $N_{je} = 1$ whenever $i \sim j$. From this we can easily see that L is positive semidefinite. The eigenvalues

of L are called the *Laplacean eigenvalues* of Γ. They are all nonnegative by the positive semidefiniteness of L.

Generally we will denote the ordinary eigenvalues of Γ by $\theta_1, \ldots, \theta_n$ and they are assumed to be in nonascending order. If an eigenvalue has a higher multiplicity, it occurs also the corresponding amount of times in the sequence. The Laplacean eigenvalues of Γ will be denoted by $\lambda_1, \ldots, \lambda_n$ and they are also in nonascending order. We always have $\lambda_n = 0$. Furthermore $\mathrm{tr}(A) = \sum_i \theta_i = 0$ and $\mathrm{tr}(L) = \sum_i \lambda_i = 2|E\Gamma|$. The ordinary (resp. Laplacean) eigenvalues are also called the *spectrum* (resp. *Laplacean spectrum*) of the graph.

We will give some examples now. The matrix J denotes the matrix with every entry equal to 1, and the matrix I denotes the identity matrix. (The size will be clear from the context.)

Example 1.1. Consider the complete graph K_n on n vertices. We have $A = J - I$. Since J and I commute, they can be simultaneously diagonalized and the ordinary eigenvalues of K_n are $\theta_1 = n-1 \geq \theta_2 = \cdots = \theta_n = -1$. We see that $\sum_i \theta_i = 0$. The Laplacean eigenvalues of K_n can be found by observing that $L = nI - J$, so L has eigenvalues n with multiplicity $n-1$ and 0 with multiplicity 1. We see that the sum of the Laplacean eigenvalues is $n(n-1)$, which equals the number of edges times two.

Example 1.2. Consider the complete bipartite graph $K_{n,m}$ on $n+m$ vertices. It has eigenvalues $\theta_1 = \sqrt{nm} \geq \theta_2 = \cdots = \theta_{n-1} = 0 \geq \theta_n = -\sqrt{nm}$. The Laplacean spectrum consists of $\lambda_1 = n+m$, $\lambda_n = 0$, the value m with multiplicity $n-1$, and the value n with multiplicity $m-1$. The sum of the Laplacean eigenvalues is $n+m+(n-1)m+(m-1)n = 2mn$, which is twice the number of edges.

It is easy to see that the spectrum of Γ is the union of the spectra of its components. This holds both for the Laplacean and for the ordinary spectrum.

1.2 Bounds on the eigenvalues

The number n will denote the number of vertices of the graph under consideration. Recall that the entries of the degree sequence $d_1(\Gamma), \ldots, d_n(\Gamma)$ of Γ correspond to the degrees of the vertices of Γ in nonascending order. The conjugate degree sequence is denoted by $d'_1(\Gamma), \ldots, d'_n(\Gamma)$ and defined as

$d_i'(\Gamma) = \#\{v \in V\Gamma : \deg(v) \geq i\}$. We use the convention $d_0'(\Gamma) = n$. We take the liberty of leaving out Γ if it is understood from the context.

Andries Brouwer conjectured the following [8, section 3.8].

Conjecture 1.3. *Let Γ be a graph. Then, for all $k = 1, \ldots, n$, we have*

$$\sum_{i=1}^{k} \lambda_i \leq |E\Gamma| + \binom{k+1}{2}. \tag{1.1}$$

Brouwer remarks in [8] that it is easy to see that the conjecture holds for threshold graphs. (A graph is called a *threshold graph* if $\lambda_i = d_i'$ for all i.) In private communication Brouwer mentions that he has checked the conjecture for all graphs on at most 10 vertices using the program geng in the **nauty** package [25] (to enumerate all isomorphism classes of graphs) combined with the GNU Scientific Library [35] (to calculate the Laplacean eigenvalues of a graph). We verified this result independently. W. Haemers, A. Mohammadian and B. Tayfeh-Rezaie proved Conjecture 1.3 for trees and for all graphs in the case $k = 2$ [16].

We will prove Conjecture 1.3 in the case where Γ is a split graph, a cograph or a regular graph. The definition of a graph being split or a cograph will be introduced in their respective sections. A graph is a threshold graph if and only if it is both a split graph and a cograph (see [7, Corollary 7.1.1]), so we will recover the result that all threshold graphs satisfy Conjecture 1.3.

1.2.1 The Brouwer conjecture for split graphs

Recently, Hua Bai [2] proved the Grone-Merris conjecture [15].

Theorem 1.4 ([2])**.** *Let Γ be a graph. Then, for all $k = 1, \ldots, n$, we have*

$$\sum_{i=1}^{k} \lambda_i \leq \sum_{i=1}^{k} d_i', \tag{1.2}$$

with equality if $k = n$.

We want to investigate in which cases the conjectured inequality (1.1) is sharper than the Grone-Merris bound (1.2). To this end, we define the sequence $f_0(\Gamma), \ldots, f_n(\Gamma)$ of integers by

$$f_k(\Gamma) = |E\Gamma| + \binom{k+1}{2} - \sum_{i=1}^{k} d_i'(\Gamma).$$

Note that f_k is the difference between the two bounds, and it is negative at precisely those k for which inequality (1.1) is sharper. We will collect some trivial but useful results in the following two lemmata.

Lemma 1.5. *For all $k = 1, \ldots, n$, we have $d_k \geq k$ if and only if $d'_k \geq k$.*

Proof. This follows from $d_k \geq j$ if and only if $d_1 \geq \cdots \geq d_k \geq j$ if and only if $d'_j \geq k$. □

Lemma 1.6. *The minimum of $f_0(\Gamma), \ldots, f_n(\Gamma)$ is attained at*

$$m(\Gamma) := \max\{k : 0 \leq k \leq n, d'_k \geq k\}. \tag{1.3}$$

Proof. Use the fact that d'_i is nonincreasing and $f_k - f_{k-1} = \binom{k+1}{2} - \binom{k}{2} - d'_k = k - d'_k$, so $f_k \leq f_{k-1}$ if and only if $k \leq d'_k$ if and only if $k \leq m$. □

We observe that $m(\Gamma) = 0$ implies that Γ is edgeless. Furthermore, note that $m(\Gamma)$ could equivalently have been defined as $\max\{k : 1 \leq k \leq n, d_k \geq k\}$ if Γ is not edgeless, due to Lemma 1.5.

A graph Γ is called *split* if its vertices can be partitioned into two sets A, B such that $\Gamma[A]$ is complete and $\Gamma[B]$ is edgeless. It turns out that for the class of split graphs, inequality (1.1) can be derived from the Grone-Merris bound.

Theorem 1.7. *If Γ is split, then $f_m(\Gamma) = 0$.*

Proof. Since Γ is split, we can partition its vertices into sets A and B such that $\Gamma[A]$ is complete and $\Gamma[B]$ is edgeless. Now take B maximal such that $\Gamma[B]$ is edgeless, and denote $N := |A|$ and $M := |B|$. We have that the degrees of the vertices A are all at least N by the maximality of B. Since $\Gamma[B]$ is edgeless, the vertices B all have degree at most N.

Now we can number the vertices of A, say $A = \{1, \ldots, N\}$, in nonascending order of degree, and we can number the vertices of B, say $B = \{N+1, \ldots, N+M\}$, in nonascending order of degree. But now all vertices of Γ are ordered in nonascending order of degree. We can see that $m(\Gamma) = N$, since $d_1 \geq \cdots \geq d_N \geq N \geq d_{N+1} \geq \cdots \geq d_{N+M}$. Denote by X the set of edges $\delta(A, B)$. We first observe

$$\begin{aligned}
\sum_{i=1}^{N} d'_i &= \sum_{i=1}^{N} \#\{v \in V\Gamma : \deg(v) \geq i\} \\
&= \sum_{i=1}^{N} \big(\#\{v \in A : \deg(v) \geq i\} + \#\{v \in B : \deg(v) \geq i\}\big) \\
&= N^2 + |X|.
\end{aligned}$$

Then

$$f_N = \binom{N+1}{2} + |E| - \sum_{i=1}^{N} d_i' = \binom{N+1}{2} + \binom{N}{2} + |X| - (N^2 + |X|),$$

which equals zero, as had to be shown. □

As an immediate corollary we have the following.

Corollary 1.8. *Conjecture 1.3 holds for split graphs.*

Proof. Combine the Grone-Merris bound (1.2) with Theorem 1.7. □

1.2.2 Splittance of graphs

We now established Conjecture 1.3 for split graphs. Before we proceed with the analysis of the value f_m, we will need to define the *splittance* of a graph. This concept was first defined by Hammer and Simeone [18].

Definition 1.9 (Splittance)**.** *Let $\Gamma = (V, E)$ be a (finite, undirected, simple) graph. The* splittance $\sigma(\Gamma)$ *of Γ is defined as the minimum cardinality of a set $F \subseteq \binom{V}{2}$ such that the graph $(V, E\Delta F)$ is split. (Here $E\Delta F$ is the symmetric difference between E and F.)*

Let us now review an important property of the splittance that can be found in the paper by Hammer and Simeone [18]. They use a slightly different convention for m, namely $\widehat{m} = \max\{i : d_i \geq i - 1\}$, but the proof carries over completely.

Theorem 1.10 ([18])**.** *Let Γ be a graph with adjacency relation $\sim$. The set F consisting of the edges*

$$F = \{\{u, v\} : u \not\sim v, 1 \leq u < v \leq m\} \cup \{\{u, v\} : u \sim v, m+1 \leq u < v \leq n\}$$

is of minimal cardinality such that $(V\Gamma, E\Gamma\Delta F)$ is split.

We see from this theorem that if we define $A := \{1, \ldots, m\}$ and $B := \{m+1, \ldots, n\}$, then adding an edge to $\Gamma[A]$ or removing an edge from $\Gamma[B]$ will decrease the splittance by 1.

1.2.3 The Brouwer bound is sharper for nonsplit graphs

Lemma 1.6 gives a value for k for which f_k is minimal. We saw in Theorem 1.7 that the minimum of f_k is zero for split graphs. For nonsplit graphs we can also compute the minimum value of f_k. Recall that the splittance of graph Γ is denoted by $\sigma(\Gamma)$. The next theorem will generalize Theorem 1.7, but use the result in its proof.

Theorem 1.11. *Let Γ be a graph. Then*

$$f_m(\Gamma) = -\sigma(\Gamma). \tag{1.4}$$

Proof. Use induction on the splittance of the graph. We may assume that the equation (1.4) holds for all graphs having splittance less than $\sigma(\Gamma)$. We may also assume without loss of generality that the vertices are numbered $1,\dots,n$ and they are ordered with nonascending degree, i.e. $\deg(i) = d_i$. Define sets $A := \{1,\dots,m\} \subseteq V\Gamma$ and $B := \{m+1,\dots,n\} \subseteq V\Gamma$. There are now three cases (not necessarily mutually disjoint):

1. the graph $\Gamma[A]$ is not complete;
2. the graph $\Gamma[B]$ is not edgeless;
3. the graph $\Gamma[A]$ is complete and the graph $\Gamma[B]$ is edgeless.

The third case means that Γ is split. Then $\sigma(\Gamma) = 0$, and we are done by Theorem 1.7. So we only need to consider the first two cases.

In the first case, the induced subgraph $\Gamma[A]$ is not complete. That means that there exist nonadjacent distinct vertices $v, w \in A$. Define $\Delta := \Gamma + \{v,w\}$. We know that $\sigma(\Delta) = \sigma(\Gamma) - 1$. We have $d'_m(\Gamma) \geq m(\Gamma) \geq d'_{m+1}(\Gamma)$ and $d_m(\Gamma) \geq m(\Gamma) \geq d_{m+1}(\Gamma)$ and from $d'_i(\Delta) = d'_i(\Gamma)$ if $i \leq m$ we see that $m(\Delta) \geq m(\Gamma)$. But $d_i(\Gamma) = d_i(\Delta)$ if $i > m$ since the added edge was between vertices of $A = \{1,\dots,m\}$. That means that $m(\Delta) = m(\Gamma)$. Now set $m := m(\Delta) = m(\Gamma)$ and $d'_i := d'_i(\Gamma) = d'_i(\Delta)$ as long as $i \leq m$. By the induction hypothesis we have in Δ that $f_m(\Delta) = |E\Delta| + \binom{m+1}{2} - \sum_{i=1}^{m} d'_i = -\sigma(\Delta)$. By counting the number of edges we see that $|E\Delta| = |E\Gamma| + 1$, hence $f_m(\Gamma) = f_m(\Delta) - 1 = -\sigma(\Delta) - 1 = -\sigma(\Gamma)$, and we are done with the first case.

Now we consider the case that $\Gamma[B]$ is not edgeless. So take an edge of $\Gamma[B]$, say $e = \{v,w\}$. Define $\Delta := \Gamma - e$. We again know that $\sigma(\Delta) = \sigma(\Gamma) - 1$. By a similar reasoning as in the proof of case 1, we have $m(\Delta) = m(\Gamma)$ so we set $m := m(\Delta) = m(\Gamma)$. On the other hand, since the degrees of vertices v and

w in Γ are both at most m, we can deduce $\sum_{i=1}^{m} d_i'(\Delta) = \sum_{i=1}^{m} d_i'(\Gamma) - 2$. We had $|E\Delta| = |E\Gamma| - 1$, hence

$$\begin{aligned} f_m(\Gamma) &= |E\Gamma| + \binom{m+1}{2} - \sum_{i=1}^{m} d_i'(\Gamma) \\ &= |E\Delta| + 1 + \binom{m+1}{2} - (\sum_{i=1}^{m} d_i'(\Delta) + 2) \\ &= f_m(\Delta) - 1 = -\sigma(\Delta) - 1 = -\sigma(\Gamma), \end{aligned}$$

and also the proof of this case is finished. □

From Theorem 1.11 we can deduce a new characterization of split graphs: a graph Γ is split if and only if $\sum_{i=1}^{m(\Gamma)} d_i'(\Gamma) = |E\Gamma| + \binom{m(\Gamma)+1}{2}$.

1.2.4 The Brouwer conjecture for cographs

A *cograph* is a graph that does not contain P_4 (the path on four vertices) as an induced subgraph. It is well-known (for example see [7, Thm. 11.3.3]) that cographs have the following inductive characterization.

1. K_1 is a cograph.
2. If Γ is a cograph, then $\overline{\Gamma}$ (the complement of Γ) is also a cograph.
3. If Γ and Δ are cographs, then their disjoint union $\Gamma \sqcup \Delta$ is also a cograph.

Every cograph can be constructed in this way.

We will prove Conjecture 1.3 for cographs inductively. The observation that K_1 satisfies inequality (1.1) is obvious. The following two lemmata will establish the Conjecture 1.3 for cographs. The proof of the first lemma can be found in [16].

Lemma 1.12 ([16])**.** *If Γ satisfies inequality (1.1) for all $k = 1, \ldots, n$, then $\overline{\Gamma}$ does as well.*

Proof (sketch). Use $\overline{\lambda}_i = n - \lambda_{n-i}$ for $i = 1, \ldots, n-1$. □

Lemma 1.13. *If Γ and Δ satisfy inequality (1.1) for all k, then $\Gamma \sqcup \Delta$ does as well.*

Proof. Let $\lambda_1 \geq \cdots \geq \lambda_n$ be the Laplacean eigenvalues of Γ, and $\mu_1 \geq \cdots \geq \mu_m$ be the Laplacean eigenvalues of Δ. The Laplacean eigenvalues of $\Gamma \sqcup \Delta$, say $\nu_1 \geq \cdots \geq \nu_{n+m}$, consist of the union $\{\lambda_i\} \cup \{\mu_i\}$. The k-th partial sum $\sum_{i=1}^{k} \nu_i$ of them is equal to $\sum_{i=1}^{p} \lambda_i + \sum_{i=1}^{q} \mu_i$ for some p, q satisfying $0 \leq p, q \leq k$ and $p + q = k$. We now claim that $p(p+1) + q(q+1) \leq k(k+1)$. But this is obvious, since it is equivalent to $p^2 + q^2 \leq k^2 = (p+q)^2 = p^2 + 2pq + q^2$. By assumption we have $\sum_{i=1}^{p} \lambda_i \leq |E\Gamma| + \binom{p+1}{2}$ and $\sum_{i=1}^{q} \mu_i \leq |E\Delta| + \binom{q+1}{2}$. Now combine these inequalities to obtain

$$\begin{aligned}\sum_{i=1}^{k} \nu_i = \sum_{i=1}^{p} \lambda_i + \sum_{i=1}^{q} \mu_i &\leq |E(\Gamma \sqcup \Delta)| + \frac{p(p+1) + q(q+1)}{2} \\ &\leq |E(\Gamma \sqcup \Delta)| + \frac{k(k+1)}{2},\end{aligned}$$

hence the lemma follows. □

Combining these lemmata gives us the following theorem.

Theorem 1.14. *Conjecture 1.3 holds for cographs.*

1.2.5 The Brouwer conjecture for regular graphs

It turns out that Conjecture 1.3 also holds for regular graphs.

Theorem 1.15. *Conjecture 1.3 holds for regular graphs.*

Proof. Let Γ be a regular graph on n vertices of valency r. We may assume without loss of generality that $r < n/2$ by Lemma 1.12. Denote the ordinary eigenvalues (*not* the Laplacean ones) of Γ by $\theta_1, \ldots, \theta_n$. Then the Laplacean eigenvalues of Γ are $r - \theta_i$ for $i = 1, \ldots, n$.

By Cauchy-Schwarz and Proposition 3.4.1 in [8] we have $(\sum_{i \in I} \theta_i)^2 \leq nr|I|$, where I is a subset of $\{1, \ldots, n\}$. This implies that $\sum_{i=1}^{k} \lambda_i \leq rk + \sqrt{nrk}$. Inequality (1.1) is now implied by showing that

$$nr + k^2 + k - 2rk - 2\sqrt{nrk} \geq 0, \tag{1.5}$$

for all $1 \leq r < n/2$ and $1 \leq k \leq n$. If we view the left hand side of (1.5) as a quadratic function (say f) in $\xi := \sqrt{r}$, we obtain

$$f(\xi) = (n - 2k)\xi^2 - 2\sqrt{nk}\xi + k^2 + k. \tag{1.6}$$

We now consider some cases, depending on the coefficient of ξ^2 in $f(\xi)$.

If $n = 2k$, then the equation for $f(\xi)$ reduces to $f(\xi) = n^2/4 + n/2 - \sqrt{2}n\xi$. Since $\xi < \sqrt{n/2}$, we have $f(\xi) \geq n^2/4 + n/2 - \sqrt{n^3}$, which is positive if $n \geq 12$. But we know that Conjecture 1.3 holds for all graphs on at most 10 vertices. Since n is even, we are done with this case.

For the other cases, we will rewrite (1.6) as

$$f(\xi) = (n-2k)(\xi - \frac{\sqrt{nk}}{n-2k})^2 - \frac{nk}{n-2k} + k^2 + k. \tag{1.7}$$

If $n > 2k+1$, then $k^2 + k - nk/(n-2k)$ is nonnegative, so $f(\xi) \geq 0$.

If $n = 2k+1$, then the roots of $f(\xi)$ are $\sqrt{n(n-1)/2} \pm (n-1)/2$, both of which are larger than $\sqrt{n/2}$ if $n \geq 10$. So if $n \geq 10$, then $f(\xi)$ is positive for $\xi < \sqrt{n/2}$. But if $n < 10$, Conjecture 1.3 is already known to hold.

The only case remaining is that $n < 2k$. From (1.7) we can see that $f(\xi)$ is an inverted parabola that attains its maximum at some negative ξ. This means that in the domain $[0, \sqrt{n/2}]$ for ξ, we have $f(\xi) \geq f(\sqrt{n/2})$. But $g(n) := 2f(\sqrt{n/2}) = n^2 - n(2k + 2\sqrt{2k}) + 2k + 2k^2$. The roots of $g(n)$ are $n = k + \sqrt{2k} \pm \sqrt{2k\sqrt{2k} - k^2}$. If $k > 8$, the roots are complex, so the convex quadratic g is everywhere positive. We investigate the cases $k \leq 8$ case by case.

1. If $k \leq 5$, then $n < 10$ and we are done by the observation that Conjecture 1.3 holds for all graphs on at most ten vertices.

2. If $k = 6$, then we should check $n = 11$. But $f(\sqrt{5}) \geq 0$, and we are done.

3. If $k = 7$, then we should check the cases $n = 11, 12, 13$. If $n = 11$, we see that $f(\sqrt{5}) \geq 0$. Similar computations for $n = 12, 13$ show that $f(\xi) \geq 0$.

4. If $k = 8$, then $g(n) = (n-12)^2$, which is always nonnegative.

□

Chapter 2

The Alon–Tarsi conjecture

2.1 Introduction

A *Latin square* (of order n) is an n by n array in which each entry is taken from an n-set (called the *symbols*), such that each symbol occurs exactly once in each row and exactly once in each column. We can assume without loss of generality that our n-set is $[n] := \{1, \ldots, n\}$. We will also assume throughout that $n \geq 1$.

Example 2.1. A very small example is given by the following:

$$L = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 1 \\ 4 & 1 & 2 & 3 \\ 3 & 4 & 1 & 2 \end{bmatrix}.$$

Equivalently, you can also view a Latin square as a function $f : [n] \times [n] \to [n]$ such that $j \mapsto f(i,j) \in S_n$ for all i and $i \mapsto f(i,j) \in S_n$ for all j. Here $S(X)$ denotes the symmetric group on X and $S_n = S([n])$.

Another way to look at a Latin square is to write $L = \sum_{s=1}^{n} sE_s$, where E_s is a permutation matrix and the E_s sum up to the all-one matrix J.

If we want to observe the symmetry of the roles of rows, columns and symbols, we can also view a Latin square L as an $n^2 \times 3$ *orthogonal array*. We will not introduce the most general definition of an orthogonal array, but only say the following. The orthogonal array associated to L has rows (i, j, k), where $L_{ij} = k$. We generally select the lexicographical ordering on the rows, but this is not essential. It has the property that if you select two columns of the orthogonal array, all pairs occur exactly once.

The Latin square L' of order m is called a *Latin subsquare* of L (where L is of order n) if there are subsets $R, C \subseteq [n]$ of cardinality m such that the Latin square L restricted to rows R and columns C equals L'. Of course, L' is called a proper Latin subsquare of L if $m < n$.

Lemma 2.2. *Suppose that L' is a proper Latin subsquare of L of order n. Then the order of L' is at most $n/2$.*

Proof. Trivial. □

2.2 The sign of a Latin square

The *sign* of a Latin square L is defined as follows. View L as a function $[n] \times [n] \to [n]$ again, and denote by r_i the i-th row of L, i.e. $r_i(j) = f(i,j)$. Similarly c_j is the j-th column of L, i.e. $c_j(i) = f(i,j)$. Recall that sgn : $S(X) \to \{\pm 1\}$ is the morphism that assigns -1 to odd bijections and 1 to even bijections. The *row sign* of L is $\prod_{i=1}^{n} \operatorname{sgn}(r_i)$. The *column sign* of L is $\prod_{j=1}^{n} \operatorname{sgn}(c_j)$. The sign of L is the product of its row sign and its column sign. By analogy, L is called *even* if its sign is 1, and *odd* otherwise.

Suppose that we interchange columns i and j of L. The column sign of L does not change. The sign of row permutations r_i all are multiplied by -1. This means that if n is odd, the even and odd Latin squares are in bijection. We denote the number of even Latin squares of order n by L_n^e, and the number of odd Latin squares of order n by L_n^o, and $L_n := L_n^e + L_n^o$. N. Alon and M. Tarsi conjectured the following in 1989. [1]

Conjecture 2.3 (Alon–Tarsi)**.** *The number of odd and even Latin squares of order n differ for even n.*

Similar to the row and column sign of a Latin square L, we can define its *symbol sign*. Write $L = \sum_{s=1}^{n} sE_s$, where the E_s are permutation matrices summing to J. The symbol sign of L is equal to $\prod_{s=1}^{n} \det(E_s)$. (Recall that the determinant of a permutation matrix is just the sign of the corresponding permutation.) The row, column and symbol sign are intimately related, as is apparent from the following theorem. This theorem was first proven by Richard Wilson and appeared in an article by J.C.M. Janssen. [20] Later another proof was given by David Glynn. [14] We follow a method by H. Derksen, which is essentially the same as Wilson's proof. [6]

Theorem 2.4. *The product of the column sign, row sign and symbol sign of a Latin square of order n is equal to $(-1)^{n(n-1)/2}$.*

Proof. By $[x]$ we will denote by "sign" of $x \in \mathbb{Z}$, i.e. $[x] = 1$ if $x \geq 0$, and $[x] = -1$ otherwise. We will use the orthogonal array representation of the Latin square, write $e \in L$ where $e = (a, b, c)$ is a row of the orthogonal array. Fix a total order on the rows of L. We will also write $e \wedge e'$ for the number of coordinates in which e and e' are equal. Recall that the sign of a permutation σ is related to the number of inversions via $\operatorname{sgn}(\sigma) = (-1)^m$ where $m = \#\{i, j : i < j, \sigma(i) > \sigma(j)\}$. Hence the product of the three signs of the Latin square is equal to

$$\prod_{\substack{e'>e \\ e'\wedge e=1}} [a'-a][b'-b][c'-c].$$

But this value is equal to

$$\begin{aligned}\prod_{\substack{e'>e \\ e'\wedge e=1}} [a'-a][b'-b][c'-c] = &\prod_{\substack{e'>e \\ a'=a}} [b'-b][c'-c] \\ &\cdot \prod_{\substack{e'>e \\ b'=b}} [a'-a][c'-c] \\ &\cdot \prod_{\substack{e'>e \\ c'=c}} [a'-a][b'-b] \\ &\cdot \prod_{\substack{e'>e \\ e'\wedge e=0}} [a'-a]^2[b'-b]^2[c'-c]^2.\end{aligned}$$

Now reorder to see that this equals

$$\prod_{e'>e} [b'-b][c'-c] \cdot \prod_{e'>e} [a'-a][c'-c] \cdot \prod_{e'>e} [a'-a][b'-b]. \tag{2.1}$$

Expression (2.1) is equal to $(-1)^{n(n-1)/2}$. Indeed, any two pairs b, b' and c, c' contribute twice to this expression, with opposite sign, namely once as $\{e, e'\} = \{(a, b, c), (a', b', c')\}$, and once as $\{e, e'\} = \{(a'', b, c'), (a''', b', c)\}$. But there are $\binom{n}{2}^2$ such pairs. The same holds for the other two factors. To conclude the proof, note that $3\binom{n}{2}^2 \equiv n(n-1)/2 \mod 2$. □

Example 2.5. Consider again the Latin square shown in Example 2.1. The row permutations are $r_1 = 1$, $r_2 = (1234)$, $r_3 = (1432)$, $r_4 = (13)(24)$. The column permutations are $c_1 = (34)$, $c_2 = (123)$, $c_3 = (1324)$, $c_4 = (142)$. The symbol permutations are $E_1 = (243)$, $E_2 = (12)$, $E_3 = (134)$, $E_4 = (1423)$, so the row sign is 1, the column sign is 1 and the symbol sign is also 1. This is not surprising in view of Theorem 2.4.

2.3 Symmetries

As is usual with algebraic and combinatoric objects, there are some groups associated to Latin squares that describe the symmetries. Consider the set $\mathfrak{L}_n$ of Latin squares of order n on symbol set $[n]$. We can permute the rows, the columns or the symbols of a Latin square to obtain another Latin square. More formally, the group $S_n \times S_n \times S_n$ acts on $\mathfrak{L}_n$. The group element (σ, τ, υ) maps $L = (a_{ij})$ to $(\upsilon(a_{\sigma^{-1}(i),\tau^{-1}(j)}))$. The verification that this is indeed a well-defined group action is left as an exercise for the reader. (An easy way to see this is to view the Latin square as an orthogonal array.) The orbits of this action are called *isotopy classes*, and two Latin squares are called *isotopic* if they are in the same orbit. There is also a way to let S_3 act on $\mathfrak{L}_n$, namely by permuting the columns of the associated orthogonal array. If L is in the same orbit as L' under the S_3-action, we say that L' is a *parastrophe* of L. If L is isotopic to some parastrophe of L', we call L and L' *paratopic*. The equivalence classes are called *main classes*. They are the orbits of a $(S_n \times S_n \times S_n) \rtimes S_3$-action, where S_3 is equipped with the obvious action on S_n^3.

We can use these definitions to state the following theorem, due to McKay, Meynert and Myrvold [26].

Theorem 2.6. *Let L be a Latin square of order n and let (σ, τ, υ) be an isotopy that fixes L. Then we have precisely one of the following.*

1. $\sigma = \tau = \upsilon = 1$.
2. *σ, τ, υ have the same cycle structure, and have at least one and at most $\lfloor n/2 \rfloor$ fixed points.*
3. *Exactly one of σ, τ, υ has at least one fixed point, and the other two have the same cycle structure without fixed points.*
4. *None of σ, τ, υ has fixed points.*

Proof. We follow [26]. Denote by $\mathrm{Fix}(\sigma)$ the fixed points of σ. Let F be the set of triples in the orthogonal array representation of L in $\mathrm{Fix}(\sigma) \times \mathrm{Fix}(\tau) \times \mathrm{Fix}(\upsilon)$. Since any two entries in L determine the third, we have that $|F|$ is equal to the product of the numbers of fixed points of any two of σ, τ, υ. This implies that $\mathrm{Fix}(\sigma)$, $\mathrm{Fix}(\tau)$, $\mathrm{Fix}(\upsilon)$ are all of the same cardinality or at least two of them are empty. Also note that for two permutations α,β that have different cycle structure, there is a positive integer m such that α^m and β^m have different numbers of fixed points. The statement is now proven except for the final constraint in in (ii). But this follows easily from Lemma 2.2. □

2.4 The extended Alon–Tarsi conjecture

We have seen that the number of even and odd Latin squares of order n are equal if n is odd. We can however restrict our attention to diagonal Latin squares. A Latin square L is called *diagonal* if $L_{ii} = 1$ for all $i = 1, \ldots, n$. We can let any $(1, 1, \upsilon)$ act on L to obtain a Latin square with the same sign. Therefore we can bring L to the form $L_{ii} = 1$ and $L_{1i} = i$ for all $i = 1, \ldots, n$. Latin squares of this form are called *normalized*. We define $AT(n)$ as the number of even normalized Latin squares of order n minus the number of odd normalized Latin squares of order n. We can easily see that $L_n^e - L_n^o = n!(n-1)!AT(n)$ if n is even, so the Alon–Tarsi conjecture boils down to $AT(n) \neq 0$ for even n. The extended Alon–Tarsi conjecture is its obvious generalization and was first stated by P. Zappa. [37]

Conjecture 2.7 (Extended Alon–Tarsi)**.** *For all $n > 1$, we have $AT(n) \neq 0$.*

2.5 Rota's basis conjecture

The motivation for the Alon–Tarsi conjecture actually comes from the following problem in linear algebra that was first stated by Rota. [19]

Conjecture 2.8 (Rota's basis conjecture)**.** *Let V be an n-dimensional vector space. Suppose that $B_1, \ldots, B_n \subseteq V$ are bases of V. Then there are n disjoint transversals of $B_1, \ldots, B_n$, each of which is again a basis.*

There is also a more general version of this conjecture in terms of matroids. It is easy to see that this problem might have something in common with Latin squares: n disjoint transversals correspond to a Latin square. Avoiding the word 'transversal', you can also state the conjecture as follows. Suppose that $B_1, \ldots, B_n \subseteq V$ are bases of V. Then their union (as multisets) can be partitioned into bases $C_1, \ldots, C_n \subseteq V$ such that for all i, j we have that $B_i \cap C_j$ has cardinality 1. It turns out that the Alon–Tarsi conjecture implies Rota's conjecture for certain fields.

Theorem 2.9. *Let V be a vector space of even dimension n over a field of characteristic zero. If $AT(n) \neq 0$, then Conjecture 2.8 holds for V.*

For the (self-contained) proof of this theorem, we refer to [31].

2.6 Counting Latin squares

2.6.1 Using the determinant of a matrix

We can count the number of Latin squares of order n in a roundabout way as follows. Let A be the $n \times n$ matrix given by $A_{ij} = x_{ij}$, where x_{ij} is indeterminate. We work over the ring $\mathbb{F}[x_{ij}]$ and for the time being we set $\mathbb{F} = \mathbb{Q}$. First we observe the following theorem, due to MacMahon. [22]

Theorem 2.10. *The number of Latin squares of order n is the coefficient of the monomial $\prod_{i=1}^{n}\prod_{j=1}^{n} x_{ij}$ in*

$$\left(\sum_{\sigma\in S_n}\prod_{i=1}^{n} x_{i\sigma(i)}\right)^n. \tag{2.2}$$

Proof. To see this, expand to obtain

$$\sum_{(\sigma_1,\ldots,\sigma_n)\in S_n^n}\prod_{i,j=1}^{n} x_{i\sigma_j(i)}.$$

This is a homogeneous polynomial of degree n^2. Consider the square-free terms. Any square-free term corresponds to n bijections $\sigma_1,\ldots,\sigma_n$ for which $\sigma_j(i)$ occurs exactly once for each i, and exactly once for each j. But then $(i,j) \mapsto \sigma_j(i)$ is a Latin square. Conversely, each Latin square determines n bijections $\sigma_1,\ldots,\sigma_n$ where $\prod_{j=1}^{n}\prod_{i=1}^{n} x_{i\sigma_j(i)}$ is square-free. □

Recall that the determinant of a square matrix A can be defined as $\det(A) = \sum_\sigma \operatorname{sgn}(\sigma)\prod_i A_{i\sigma(i)}$. Similary, the *permanent* of a square matrix A is defined as $\operatorname{per}(A) = \sum_\sigma \prod_i A_{i\sigma(i)}$. However, in contrast with the determinant, an efficient algorithm to compute the permanent is not known and is conjectured not to exist, and many useful properties that make the determinant easy to handle do not hold for the permanent. Both the determinant and the permanent are defined even if the matrices are not over a field, but a commutative ring. This means that we can also write equation (2.2) as $(\operatorname{per}(A))^n$. We can replace the permanent with the determinant, and we obtain the following. [33]

Theorem 2.11. *The expression $(L_n^e - L_n^o)(-1)^{n(n-1)/2}$ is equal to the coefficient of the monomial $\prod_{i=1}^{n}\prod_{j=1}^{n} x_{ij}$ in $(det(A))^n$.*

Proof. The proof is very similar to the proof of the previous theorem. □

n	$L_n/(n!(n-1)!)$	factorization
2	1	
3	1	
4	4	2^2
5	56	$2^3 \cdot 7$
6	9408	$2^6 \cdot 3 \cdot 7^2$
7	16942080	$2^{10} \cdot 3 \cdot 5 \cdot 1103$
8	535281401856	$2^{17} \cdot 3 \cdot 1361291$
9	377597570964258816	$2^{21} \cdot 3^2 \cdot 5231 \cdot 3824477$
10	75807214831601328114892800	$2^{28} \cdot 3^2 \cdot 5 \cdot 31 \cdot 37 \cdot 547135293937$

Table 2.1: The number of normalized Latin squares of order n.

2.6.2 Using graphs and their isomorphisms

A more practical method to compute the number L_n of Latin squares of order n is due to McKay and Rogoyski. [27] First we need a definition. A *factorization* of a graph G is a set of pairwise disjoint perfect matchings of G whose union is $E(G)$. We will only consider bipartite graphs and think of the color classes as 'rows' R and 'columns' C.

Consider the complete bipartite graph K_{nn}. A semi-normalized Latin square of order n now corresponds to a factorization of K_{nn}. Conversely, if we have n pairwise disjoint perfect matchings of K_{nn}, we can recover a semi-normalized Latin square by letting each perfect matching correspond to a symbol. The choice of the symbols is fixed since the first row of a semi-normalized Latin square is fixed. If we instead of K_{nn} consider the bipartite graph $G_n < K_{nn}$ where the edges connecting column i with row i are removed, we have essentially fixed the diagonal of the Latin square. So the number of factorizations of G_n is equal to the number of normalized Latin squares of order n.

We can use these considerations to find an algorithm to compute L_n. Define $N(G)$ as the number of factorizations of G. It is trivial to see that $N(G) = 1$ if G is edgeless. Otherwise, fix an edge e and compute the set $\mathscr{M}$ of all perfect matchings containing e. We will always take e to be the minimal edge with respect to some total order. Now $N(G) = \sum_{M \in \mathscr{M}} N(G \backslash M)$. This gives a well-defined recursion and we can use this to compute $N(G_n)$.

Note that $N(G)$ does only depend on the isomorphism class of G. So instead of G, compute a *canonical representative* of the isomorphism class of G and use that instead. Now we can use memoization to improve the effiency of our algorithm. [10] The computation of the canonical representative is done

by nauty. [25]

Computing $N(G_n)$ is now practically feasible for $n = 2, \ldots, 10$. The values are given in Table 2.1. Computing $N(G_{10})$ takes about nine hours on my machine.[1] McKay and Wanless [28] also computed

$$N(G_{11}) = 5363937773277371298119673540771840.$$

2.7 Computing $AT(n)$ for $n \leq 10$

With some bookkeeping we can adapt the method of section 2.6.2 to compute $AT(n)$ for some small values of n. Consider again bipartite graphs on two color classes of size n. We can define the sign of a perfect matching M and the sign of a factorization in the obvious way. We are now asked to count the number of factorizations of G, but taking into account their sign. So define $N^{\pm}(G)$ as the number of even factorizations of G minus the number of odd factorizations of G. We are interested in $AT(n) = (-1)^{n(n-1)/2} N^{\pm}(G_n)$. Note that $N^{\pm}(G_n)$ counts Latin squares as odd (resp. even) if they are symbol-odd (resp. symbol-even), and this must be corrected by a factor $(-1)^{n(n-1)/2}$ to find $AT(n)$ per Theorem 2.4.

Let us now forget for a moment about the isomorphism detection and the sign, and instead consider what the algorithm described in section 2.6.2 does to compute $N(G)$. We can construct a directed graph (the *call-graph*) in which nodes are graphs under consideration and there is an arrow from G to G' if $G' = G \backslash M$ for some perfect matching M of G containing the minimal edge e. We are now computing the number of paths in the call-graph from G_n to the edgeless graph. Note that these paths all have the same length, namely $n - 1$. Let G be a node in the call-graph. We call the distance $d(G)$ from G_n to G the *depth* of G.

Now we are going to introduce the sign and isomorphism detection. An *AT-isomorphism* ϕ is an isomorphism of graphs, such that $\phi(C) = C$ and $\phi(R) = R$ (i.e. ϕ sends columns to columns and rows to rows). Now we are ready to prove two lemmata.

Lemma 2.12. *Suppose that G has an edge e. Then we have*

$$N^{\pm}(G) = \sum_{M \in \mathcal{M}} N^{\pm}(G \backslash M)(-1)^{sgn(M)},$$

where M is the collection of all perfect matchings containing e.

[1] This machine has an Intel Core-2 Quad Q9550 processor running at 3.4 GHz.

n	$AT(n)$	factorization
2	1	
3	-1	-1
4	4	2^2
5	-24	$-1 \cdot 2^3 \cdot 3$
6	2304	$2^8 \cdot 3^2$
7	368640	$2^{13} \cdot 3^2 \cdot 5$
8	6210846720	$2^{17} \cdot 3^6 \cdot 5 \cdot 13$
9	2086844497920	$2^{21} \cdot 3^7 \cdot 5 \cdot 7 \cdot 13$
10	27369126116720640000	$2^{33} \cdot 3^9 \cdot 5^4 \cdot 7 \cdot 37$

Table 2.2: The values $AT(n)$ for small n.

Proof. Trivial. □

Lemma 2.13. *If $\phi : G \to G'$ is an AT-isomorphism, then*

$$N^{\pm}(G) = sgn(\phi)^{n-1-d(G)} N^{\pm}(G').$$

Proof. The paths from G to the edgeless graph are in bijection to the paths from G' to the edgeless graph. The sign of each arrow in these paths are flipped according to the sign of ϕ. Each of these paths has length $n-1-d(G)$. □

From Lemma 2.13 it follows readily that if G has an odd automorphism and $n-1-d(G)$ is odd, then $N^{\pm}(G) = 0$. This helps a bit in pruning. This observation is a generalization of the observation that the number of even Latin squares is equal to the number of odd Latin squares of odd order.

Nauty has the capability to find AT-isomorphisms, and also to find a canonical representative of each AT-isomorphism class. This means that using memoization, we can again find the values of $AT(n)$ for small n. They are given in Table 2.2. The values of $AT(n)$ for $n = 2,4,6,8$ were given by Janssen.[20] The cases $n = 3,5,7$ were computed by Marini and Pirillo[23] according to Zappa [37], who gives a table for $n = 2,\ldots,8$. The values of $AT(9)$ and $AT(10)$ where hitherto unknown.

2.8 Divisibility properties of $AT(n)$

McKay and Wanless [28] give some divisibility properties of $R_n := L_n/(n!(n-1)!)$. With some thought we can prove a similar result for $AT(n)$.

Theorem 2.14. *Suppose that n is a positive integer. We have the following.*

1. *$AT(2n+1)$ is a multiple of $gcd(n!(n-1)! \cdot AT(n), (n+1)!)$.*
2. *$AT(2n)$ is a multiple of $n!$.*
3. *R_{2n+1} is a multiple of $gcd(n!(n-1)! \cdot R_n, (n+1)!)$.*
4. *R_{2n} is a multiple of $n!$.*

Proof. We will partition the set of normalized Latin squares into equivalence classes. Let us first consider the case $m = 2n+1$. Let L be a normalized Latin square of order $2n+1$, and let L be its leading principal subarray of order n. There are two cases. If A is a Latin subsquare, the squares equivalent to L are those by possibly replacing A by another normalized Latin square of order n, permuting the $n-1$ partial rows $(l_{i,n+1}, \ldots, l_{i,2n+1})$ for $i = 2, \ldots, n$, permuting the n partial columns $(l_{n+1,j}, \ldots, l_{2n+1,j})$ for $j = 1, \ldots, n$. These operations are closed under composition and give different Latin squares. Replacing Latin subsquare A by B multiplies the sign by the signs of A and B. So this equivalence class has size $n!(n-1)! \cdot R_n$ and weight either $n!(n-1)! \cdot AT(n)$ or 0. If A is not a Latin subsquare, then the squares equivalent to L are those by taking a permutation σ of $\{n+1, \ldots, 2n+1\}$ and applying (σ, σ, σ) to the Latin square. The result will be a normalized Latin square with the same sign as L, and it will be different from L if $\sigma \neq 1$ by Theorem 2.6. So this equivalence class has size $(n+1)!$ and weight $\pm(n+1)!$. The result follows. The case $m = 2n$ is analogous. □

In fact we have seen that $AT(n)$ seems to have 'nicer' divisibility properties than R_n, for example 5 does not divide R_8 or R_9, whereas for $n \leq 10$ have $(n-2)! | AT(n)$.

2.9 Some cases proved

2.9.1 The case $n = p - 1$

In 2010, Glynn proved the Alon–Tarsi conjecture for the case $n = p-1$, where p is an odd prime. [14] It is a fairly straightforward corollary of a combinatorial lemma that he himself proved in 1998, twelve years earlier. [13] We will give a self-contained proof due to Blokhuis. [6]

So let p be an odd prime. First we introduce some notation. Let m be a positive integer and let $A = \mathbf{x} = (x_{ij})$ be an $m \times m$ matrix over the commutative ring $GF(p)[x_{ij}]$. The value $\det_p(A)$ is defined as

$$\det_p(A) = (-1)^m \sum_{\mathbf{e} \in E} \frac{\mathbf{x}^{\mathbf{e}}}{\mathbf{e}!},$$

where $\mathbf{x}^{\mathbf{e}} := \prod_{i,j} x_{ij}^{e_{ij}}$ and $\mathbf{e}! := \prod_{i,j} e_{ij}!$. The set E is the set of all $m \times m$ matrices having nonnegative integral entries and whose row and column sums are all equal to $p-1$. Note that if $\mathbf{e} \in E$, all entries of $\mathbf{e}$ are at most $p-1$, so $\mathbf{e}!$ is nonzero after reduction modulo p. We are going to overload the meaning of symbol σ. It will denote both a permutation on $\{1, \ldots, m\}$ and the corresponding permutation matrix. With this notation we can write $\det(A) = \sum_\sigma \operatorname{sgn}(\sigma)\mathbf{x}^\sigma$. Now we can state Glynn's lemma.

Lemma 2.15. *We have* $\det_p(A) = \det(A)^{p-1}$.

Proof. Observe that, by the freshman's dream,

$$\det(A)^p = \sum_{\sigma \in S_m} \operatorname{sgn}(\sigma)\mathbf{x}^{p\sigma}. \tag{2.3}$$

Also notice that

$$\det(A)\det_p(A) = \left(\sum_{\sigma \in S_m} \operatorname{sgn}(\sigma)\mathbf{x}^\sigma \right) (-1)^m \sum_{\mathbf{e} \in E} \frac{\mathbf{x}^{\mathbf{e}}}{\mathbf{e}!} \tag{2.4}$$

$$= (-1)^m \sum_{\mathbf{e} \in E} \sum_{\sigma \in S_m} \operatorname{sgn}(\sigma) \frac{\mathbf{x}^{\mathbf{e}+\sigma}}{\mathbf{e}!}. \tag{2.5}$$

Now compare the right hand side of (2.3) with the right hand side of (2.5). Consider the coefficient of $\mathbf{x}^{\mathbf{d}}$. The only terms that occur have the property that $\mathbf{d}$ has nonnegative entries and that $\mathbf{d}$ has column and row sums equal to p. If $\mathbf{d} = p\tau$ for some permutation $\tau \in S_m$, then there is exactly one choice of $\mathbf{e}$ and σ (namely $\mathbf{e} = (p-1)\tau$ resp. $\sigma = \tau$) that contributes to the coefficient. Since $\mathbf{e}!$ then equals $(-1)^m$, the coefficient of $\mathbf{x}^{\mathbf{d}}$ is $\operatorname{sgn}(\sigma)$ in both equations. Otherwise, define (for technical reasons) $Y := \{i : d_{ij} \neq p \text{ for all } j\}$

and $\mathbf{e}!! := \prod_{e_{ij} \neq p} e_{ij}!$. Now calculate the coefficient of $\mathbf{x}^{\mathbf{d}}$ in (2.5):

$$\begin{aligned}\sum_{\sigma, \mathbf{d}-\sigma \geq 0} \operatorname{sgn}(\sigma) \frac{1}{(\mathbf{d}-\sigma)!} &= \pm \sum_{\sigma, \mathbf{d}-\sigma \geq 0} \operatorname{sgn}(\sigma) \frac{\prod_{i \in Y} d_{i\sigma(i)}}{\mathbf{d}!!} \\ &= \pm \sum_{\sigma} \operatorname{sgn}(\sigma) \frac{\prod_{i \in Y} d_{i\sigma(i)}}{\mathbf{d}!!} \\ &= \pm \frac{\det \mathbf{d}_Y}{\mathbf{d}!!} = 0,\end{aligned}$$

where $\mathbf{d}_Y$ is the submatrix of $\mathbf{d}$ with rows and columns containing a p removed. (Note that by assumption $\mathbf{d}_Y$ is not an "empty matrix".) The matrix $\mathbf{d}_Y$ has zero determinant, since its row sums and column sums are all p. □

Theorem 2.16. *Suppose that p is an odd prime. Then*

$$AT(p-1) \equiv (-1)^{(p+1)/2} \mod p.$$

In particular, $AT(p-1)$ is nonzero.

Proof. Set $n = p-1$ and work in $GF(p)$. Consider the coefficient of $\prod_{i,j} x_{ij}$ in $\det(A)^n$. It is equal to $(-1)^n = 1$ by Lemma 2.15. By Theorem 2.11 this value is equal to $(-1)^{n(n-1)/2}(L_n^e - L_n^o)$. But $L_n^e - L_n^o = AT(n)n!(n-1)!$, and $n! = -(n-1)! = -1$. □

The reader is invited to check the values given in Table 2.2 using Theorem 2.16. This gives us a bit more faith in the calculations.

2.9.2 The case $n = p$

The case where n is an odd prime also readily follows from Glynn's lemma. This was first proven by Drisko using a different method. [12]

Theorem 2.17. *Let p be an odd prime. Then $AT(p) \equiv (-1)^{(p-1)/2} \mod p$. In particular, $AT(p)$ is nonzero.*

Proof. Set $n := p$. We can now define the $n \times n$ matrix A as

$$A = \begin{bmatrix} 0 & x_{12} & \cdots & x_{1n} \\ x_{21} & 0 & \cdots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & 0 \end{bmatrix},$$

i.e. the matrix where the entry at (i,j) is x_{ij} if $i \neq j$, and zero on the diagonal. The coefficient of $\prod_{i\neq j} x_{ij}$ in $\det(A)^{p-1}$ equals $(-1)^{p(p-1)/2}AT(p)$. Indeed, $AT(p)$ equals the weighted sum of Latin squares of order p with constant diagonal, each of which corresponds to a choice of $p-1$ disjoint traversals of A, each disjoint from the diagonal. (The argument is essentially the same as in the proof of Theorem 2.10.) But by Lemma 2.15 we have that $\det(A)^{p-1} = -1$ in $GF(p)$, whence the statement follows. □

2.9.3 The case $n = p+1$

If $n = p+1$, where p is an odd prime, the Alon–Tarsi conjecture also follows quite easily from Glynn's lemma. This case was already shown in 1997 by Drisko using a different method. [11]

Theorem 2.18. *Let p be an odd prime. Then*

$$AT(p+1) \equiv (-1)^{(p+1)/2} \mod p.$$

In particular, $AT(p+1)$ is nonzero.

Proof. Set $n := p+1$. Consider diagonal Latin squares of order n. It is clear that the difference between the number of even diagonal Latin squares and odd diagonal Latin squares equals $(n-1)!AT(n)$. Whenever σ is a derangement (a permutation without fixed points) on n points, we define $a(\sigma)$ to be the difference between the number of symbol-even and symbol-odd Latin squares in the set of all diagonal Latin squares such that $L_{i\sigma(i)} = 2$ for all i. We can now easily see using Theorem 2.4 that

$$(n-1)! \cdot AT(n) \cdot (-1)^{n(n-1)/2} = \sum_{\sigma} a(\sigma), \tag{2.6}$$

where the sum runs over all derangements on n points.

Now we invoke Glynn's lemma to compute $a(\sigma)$. Define the $n \times n$ matrix A as follows. We have $A = x_{ij}$ if $i \neq j$ and $\sigma(i) \neq j$, and 0 otherwise. Observe that a diagonal Latin square is obtained by choosing $p-1$ disjoint traversals in A each missing the zeros in A. Conversely, a diagonal Latin square where the symbol 2 is defined by σ corresponds to $p-1$ disjoint traversals in A each missing the zeros in A. The symbol-sign of the Latin square is equal to the sign of σ times the signs of the traversals. But by Glynn's lemma the unique squarefree monomial in $\det(A)^{p-1} = \det_p(A)$ (which corresponds to choosing $p-1$ disjoint traversals missing the zeros and weighing by the sign of the traversals) has coefficient 1 modulo p. So we have $a(\sigma) \equiv \operatorname{sgn}(\sigma) \mod p$.

Now we can calculate, using (2.6) and Wilson's theorem,

$$AT(n) = (-1)^{n(n-1)/2} \frac{\sum_\sigma a(\sigma)}{p \cdot (p-1)!} = (-1)^{1+n(n-1)/2} \frac{\sum_\sigma \operatorname{sgn}(\sigma)}{p},$$

where both sums run over the derangements on n points again. So now our problem has been reduced to computing the difference between the number of even and odd derangements on n points. By Lemma 2.19 this number equals $(-1)^{n-1}(n-1)$, hence the result has been proven. □

The only remaining part is the calculation of the difference between the number of even and odd derangements on n points. This is an old exercise that can be found in [30]. An alternative proof is given in [4]. The lemma has a cute proof that is illustrative of the power of algebraic methods in combinatorics.

Lemma 2.19. *The difference between the number of even and odd derangements on n points equals* $(-1)^{n-1}(n-1)$.

Proof. Expand $\det(J-I)$, where J is the $n \times n$ all-1 matrix. It equals $\sum_\sigma \operatorname{sgn}(\sigma)$, where σ runs through all derangements in S_n, which is the quantity that we are after. Recall from linear algebra that if two diagonalizable matrices commute, they are simultaneously diagonalizable. Also recall that the determinant of a matrix is equal to the product of its eigenvalues. But the spectrum of J is $n^{(1)}, 0^{(n-1)}$. □

2.9.4 A tragedy: the case $n = p + 2$

So far we have shown that Glynn's method can also be applied to the cases $n = p$ and $n = p + 1$. These cases were already settled, however. It is natural to ask the question: what about $n = p+2$? We have an aesthetically somewhat disappointing answer.

We start with two lemmata. Recall that a derangement is a permutation without fixed points. We call derangements σ and τ *disjoint* if they have no common image. In other words, σ and τ are disjoint if $\sigma^{-1}\tau$ is a derangement as well.

The first lemma

Lemma 2.20. *Suppose that σ is a derangement on an odd number n of points consisting of k even cycles and no odd cycles. Then we have that the difference*

between the number of even and odd derangements disjoint from σ equals

$$\det(J - I - \sigma) = 2^{k-1}(n-2).$$

Furthermore, suppose that $\sigma(1) = 2$. Then the difference between the number of even and odd derangements τ disjoint from σ that satisfy $\tau(1) = 3$ equals 2^{k-1}.

Proof. Note that $J - I$ and σ commute and are both diagonalizable. The spectrum of $J - I$ is $(n-1)^{(1)}, (-1)^{(n-1)}$. Denote by $c_1, \ldots, c_k$ the length of the k cycles in σ. The c_i are all odd and sum to n. (Note that from this it follows that k is necessarily odd.) The spectrum of σ is the multiset union of Z_i, where Z_i is the set of the c_i-th roots of unity. Now we have

$$\det(J - I - \sigma) = \prod_{i=1}^{k-1} \prod_{\zeta \in Z_i} (-\zeta - 1) \cdot \prod_{\zeta \in Z_k, \zeta \neq 1} (-\zeta - 1) \cdot (n-2). \tag{2.7}$$

Let i be in $\{1, \ldots, k\}$ and denote by ζ the primitive c_i-th root of unity. It is easy to see by comparing roots that $\prod_{j=1}^{c_i - 1}(x - \zeta^j) = (x^{c_i} - 1)/(x - 1)$. Now substitute $x = -1$ to find $\prod_{\zeta \in Z_i, \zeta \neq 1}(-\zeta - 1) = 1$ since $|Z_i| = c_i$ is odd. Substitute this in (2.7) to obtain

$$\det(J - I - \sigma) = \prod_{i=1}^{k-1} (-2) \cdot (n-2) = 2^{k-1}(n-2),$$

since k is odd. The rest of the statement is obvious. □

Interlude: exponential generating functions

Before we proceed with the next lemma, we will review some facts about combinatorial species and exponential generating functions. We will use the definitions in [5] and [21]. We will use some of the basic language of category theory. If you are not familiar with it, you can either choose to believe Theorem 2.23 or refer to [32]. We will not need these definitions later.

Definition 2.21. *A* species *is an endofunctor $\mathscr{F} : \mathbb{B} \to \mathbb{B}$, where $\mathbb{B}$ is the category with finite sets as objects and bijections as arrows. Whenever U is a finite set, the finite set $\mathscr{F}U$ is called the set of* $\mathscr{F}$-structures on U.

If $\sigma : U \to V$ is an arrow in $\mathbb{B}$, then $\mathscr{F}\sigma : \mathscr{F}U \to \mathscr{F}V$ is a bijection between the set of $\mathscr{F}$-structures on U and the set of $\mathscr{F}$-structures on V, also called the *transport of $\mathscr{F}$-structures along σ*.

Let $\mathscr{F}$ be a species. We can write the *exponential generating function* (abbreviated EGF) $F(x)$ of $\mathscr{F}$ as the formal power series $F(x) = \sum_{k=0}^{\infty} |\mathscr{F}[n]| x^n/n!$. It is obvious that $|\mathscr{F}U|$ only depends on the cardinality of U.

Let $\mathscr{F}$ and $\mathscr{G}$ be species with EGFs $F(x)$ and $G(x)$ respectively. We can define their sum $\mathscr{F}+\mathscr{G}$ as follows: $(\mathscr{F}+\mathscr{G})U$ will be the direct sum of $\mathscr{F}U$ and $\mathscr{G}U$. The definition of $(\mathscr{F}+\mathscr{G})\sigma$ is obvious. We can also define the product $\mathscr{F}\mathscr{G}$. An $\mathscr{F}\mathscr{G}$-structure on U is the pair (f,g) where f is an $\mathscr{F}$-structure on U_1, g is an $\mathscr{G}$-structure on U_2 and $\{U_1,U_2\}$ is a partition of U. The image of the functor $\mathscr{F}\mathscr{G}$ on arrows is obvious. It is easy to check that $\mathscr{F}+\mathscr{G}$ and $\mathscr{F}\mathscr{G}$ are well defined endofunctors on $\mathbb{B}$. It is also easy to verify that addition and multiplication are commutative and associative up to natural isomorphism. Finally it is quite easily seen that $F(x)+G(x)$ is the EGF of $\mathscr{F}+\mathscr{G}$ and $F(x)G(x)$ is the EGF of $\mathscr{F}\mathscr{G}$.

Definition 2.22. *Let $\mathscr{F}$ be a species such that $\mathscr{F}\emptyset$ is empty. The set of* $\exp\circ\mathscr{F}$*-structures on U is the set of pairs $(P,\{f_V\}_{V\in P})$, where P is a partition of U, and f_V is a $\mathscr{F}$-structure on $V\subseteq U$.*

We will use the following theorem, which is at heart of various applications that we will not go into. The theorem is a special case of a theorem given by Joyal. [21]

Theorem 2.23. *Let $\mathscr{F}$ be a species such that $\mathscr{F}\emptyset$ is empty. The species* $\exp\circ\mathscr{F}$ *is well-defined and has EGF* $\exp(F(x))$*, where $F(x)$ is the EGF of $\mathscr{F}$.*

Proof. Let U be a finite set of cardinality n. By definition and using the assumption that $\mathscr{F}\emptyset=\emptyset$, we have that the set of $\exp\circ\mathscr{F}$-structures on U equals

$$\sum_{\{U_i\}\text{ partition }U}\prod_i \mathscr{F}(U_i) = \sum_{j=0}^{n}\sum_{\{U_1,\ldots,U_j\}}\prod_{i=1}^{j}\mathscr{F}(U_i).$$

By counting the number of partitions of U we have that the coefficient of $x^n/n!$ in the EGF of $\exp\circ\mathscr{F}$ equals

$$\sum_{j=0}^{n}\sum_{n_1+\cdots+n_j=n,n_i>0}\frac{1}{j!}\binom{n}{n_1\cdots n_j}\prod_{i=1}^{j}|\mathscr{F}[n_i]|.$$

But it is easy to see that this equals $\sum_{j=0}^{n}\frac{1}{j!}F(x)^j = \exp(F(x))$. □

After this digression on combinatorial species, we can resume with the lemma. The proof of the lemma will use Theorem 2.23.

Lemma 2.24. *Denote by $b_k(n)$ the number of derangements on n elements consisting of exactly k cycles each of odd length. Let $S(x)$ be the exponential generating function $\sum_{n=0}^{\infty} s_n x^n/n!$ where $s_n = \sum_k b_k(n)2^k$. Then*

$$S(x) = e^{-2x}\frac{1+x}{1-x},$$

and

$$s_n = -(-2)^n + 2\sum_{k=0}^{n}\frac{(-2)^k \cdot n!}{k!}.$$

Proof. Define $T(x) = 2(\frac{1}{3}x^3 + \frac{1}{5}x^5 + \ldots)$. It is clear that $T(x)$ is the exponential generating function belonging to the sequence $t_1, t_2, \ldots$, where t_n denotes the number of derangements on n points consisting of exactly one even cycle plus a color (either black or white). Now $S(x) = \exp(T(x))$ by Theorem 2.23. Recall that $-\log(1-x) = x + \frac{x^2}{2} + \frac{x^3}{3} + \ldots$, hence $T(x) = -2x + \log(1+x) - \log(1-x)$. Substituting, we find $S(x) = \exp(-2x)(1+x)/(1-x)$.

To calculate s_n, rewrite $S(x) = \exp(-2x)(-1 + 2/(1-x))$, note that the power series of $\exp(-2x)$ is $\sum x^n(-2)^n/n!$, and that (via convolution) the power series of $-\exp(-2x)/(1-x)$ is $\sum x^n \sum_{k=0}^{n}(-2)^k/k!$. Now s_n equals $-(-2)^n + 2\sum_{k=0}^{n}(-2)^k n!/k!$, which had to be shown. □

The theorem

Now we can embark on the proof of the following theorem, which resolves $n = p + 2$ in a finite number of cases.

Theorem 2.25. *Let p be an odd prime. Then the following statements are true.*

1. *The value $AT(p+2)$ is a multiple of p.*
2. *Modulo p, we have*

$$\frac{AT(p+2)}{p} \equiv (-1)^{(p-1)/2}\left(1 - \sum_{k=0}^{p-1}\frac{k!}{2^k}\right).$$

3. *If $p < 1048576$ and $p \neq 234781$, then $AT(p+2) \neq 0$.*

Proof. Set $n = p + 2$. Whenever σ, τ are disjoint derangements on n points such that $\sigma(1) = 2$ and $\tau(1) = 3$, denote by $a(\sigma, \tau)$ the difference between the number of symbol-even and symbol-odd Latin squares in the set $S(\sigma, \tau)$

of diagonal Latin squares of order n such that $L_{i\sigma(i)} = 2$ and $L_{i\tau(i)} = 3$ for all i. We are interested in the value $AT(n) = (-1)^{n(n-1)/2}(n-3)! \sum_{\sigma,\tau} a(\sigma,\tau)$.

First note that $\sum_{\sigma,\tau} a(\sigma,\tau) = 0$ where the sum runs over all disjoint derangements σ, τ such that σ has a cycle of even length (in its disjoint cycle representation). Indeed, we can construct a bijection between even and odd such Latin squares. Suppose that σ is a derangement having smallest even-length (so odd!) cycle $(x_1, \ldots, x_s)$. We are going to transform any Latin square L in $S(\sigma,\tau)$ as follows. We only change the values in columns $c_{x_1}, \ldots, c_{x_s}$. Set $x_{s+1} = x_1$ and define column c'_{x_i} as $c_{x_{i+1}}$, except that the values 1 and 2 are interchanged. We claim that the sign of the Latin square has now been flipped. The column sign has not changed: we have $\mathrm{sgn}(c_{x_{i+1}}) = -\mathrm{sgn}(c'_{x_i})$, but s is even. However, the row sign has flipped. The rows $r_{x_1}, \ldots, r_{x_s}$ have not changed their sign, but the other rows (an odd number) have. So the sign of the Latin square has been flipped. The smallest even-length cycle of σ has been inverted by this operation. Now we can easily see that this defines a bijection between even and odd Latin squares having an even-length cycle in σ. This means that we can disregard such Latin squares, since their total contribution to $\sum_{\sigma,\tau} a(\sigma,\tau)$ will be zero.

Now consider $\sum_{\sigma,\tau} a(\sigma,\tau)$ again. Let i be in $\{3, \ldots, n\}$. We have that

$$\sum_{\sigma,\tau} a(\sigma,\tau) = (n-2) \sum_{\sigma,\tau,\sigma(2)=i} a(\sigma,\tau).$$

This is obvious from the fact that doing the same operation as described above on the columns $2, 3, \ldots, n$ and rows $2, 3, \ldots, n$ yields a Latin square with the same sign and from the fact that σ's satisfying $\sigma(2) = 1$ have zero contribution.

We can now use Lemma 2.15 to find the value of $a(\sigma,\tau)$ modulo p. Define the $n \times n$ matrix A as follows:

$$A_{ij} = \begin{cases} 0 & \text{if } i = j \text{ or } j = \sigma(i) \text{ or } j = \tau(i), \\ x_{ij} & \text{otherwise.} \end{cases}$$

Using a standard argument we see that $\det(A)^{p-1} \cdot \mathrm{sgn}(\sigma\tau)$ is the difference of symbol-even and symbol-odd diagonal Latin squares L satisfying $L_{i\sigma(i)} = 2$ and $L_{i\tau(i)} = 3$. Glynn's lemma gives $\det(A)^{p-1} = -1$. Hence we have $a(\sigma,\tau) \equiv -1 \cdot \mathrm{sgn}(\sigma\tau) \equiv -\mathrm{sgn}(\tau) \mod p$.

Using the considerations above and Lemma 2.20, calculate

$$\begin{aligned}(n-3)!\cdot(-1)^{n(n-1)/2}AT(n) &= \sum_{\sigma,\tau} a(\sigma,\tau)\\ &= (n-2)\sum_{\sigma,\tau,\sigma(2)=3} a(\sigma,\tau)\\ &\equiv -(n-2)\sum_{\sigma,\tau,\sigma(2)=3} \operatorname{sgn}(\tau) \mod p^2\\ &\equiv -(n-2)\sum_{\sigma,\sigma(2)=3} 2^{k(\sigma)-2} \mod p^2,\end{aligned}$$

where $k(\sigma)$ denotes the number of even cycles in σ. This implies that $AT(n)$ is a multiple of p, and

$$\frac{AT(n)}{p} \equiv \frac{(-1)^{(n-1)/2}}{4}\sum_{\sigma,\sigma(2)=3} 2^{k(\sigma)} \mod p. \tag{2.8}$$

What is left is calculating $\sum_{\sigma,\sigma(2)=3} 2^{k(\sigma)}$. But this expression equals $s_{p+2}/(p(p+1))$ with s_n as in the statement of Lemma 2.24. Splitting off $k = p, p+1, p+2$, expanding and simplifying yields

$$\frac{s_{p+2}}{(p+1)!} = 2\frac{(-2)^p}{(p-1)!} + 2\sum_{k=0}^{p-1}\frac{(-2)^k(p+2)}{k!}.$$

Now substitute this in (2.8), work modulo p, use $x^p = x$ and $(p-1)! = -1$, to see

$$\frac{AT(n)}{p} \equiv (-1)^{(n+1)/2}\left(1 + \sum_{k=0}^{p-1}\frac{(-2)^k}{k!}\right).$$

Now we calculate, still working modulo p,

$$\begin{aligned}\frac{AT(n)}{p}(-1)^{(n+1)/2} &\equiv 1 - \sum_{k=0}^{p-1}\frac{(p-1)!}{k!(-2)^{-k}}\\ &\equiv 1 - \sum_{k=0}^{p-1}\frac{(-1)(-2)\cdots(-(p-k-1))}{(-2)^{p-k-1}}\\ &\equiv 1 - \sum_{k=0}^{p-1}\frac{(p-k-1)!}{2^{p-k-1}}\\ &\equiv 1 - \sum_{k=0}^{p-1}\frac{k!}{2^k},\end{aligned}$$

which proves the second statement. The last statement of the theorem can be verified by executing the program given in Appendix B. □

The program in Appendix B informs us upon its execution that $AT(p+2) \equiv 0 \mod p^2$ for $p = 234781$. Of course this does not constitute a disproof of the Alon–Tarsi conjecture.

The proof technique for $AT(n)$ where $n = p-1, p, p+1$ only seems to resolve the case $AT(p+2)$ in a finite number of cases. Besides this, we have had to go through various contortions in order to prove even this finite number. This does not give us hope that $AT(p+3)$ is within reach using the same techniques (also because $p+3$ is even, which seems to be the harder case). Proving the $AT(p+3)$ case would imply $AT(26) \neq 0$, the smallest unknown case.

Chapter 3

The Gossip problem

3.1 Introduction

In this chapter we consider the following problem. There are n people, each knowing a set of secrets. Initially each participant knows exactly one unique secret. In a step, two participants call each other and exchange their secrets so that they both have the same knowledge. We can formalize this setting by seeing the 'state' as an $n \times n$ matrix with entries in $\{0,1\}$, and a transition by choosing two rows and replacing them by their maximum. The initial state will be the identity matrix. The directed graph having vertices all $n \times n$ matrices with entries in $\{0,1\}$ and arcs (u,v) whenever there is a transition that takes u to v will be denoted by G_n. We will identify the participants in our setting with the indices $1,\ldots,n$ of the rows. We can then annotate an arc (u,v) with a pair $\{i,j\}$ of people that communicated in this step.

3.2 The minimum number of calls to reach J

We are interested in the distance $m_n := d(I,J)$ in the graph G_n. (Recall that J is the all-1 matrix.) By direct calculation we can find $m_1 = 0$, $m_2 = 1$, $m_3 = 3$, $m_4 = 4$. We can easily see that $2n-4$ is an upper bound for m_n if $n \geq 4$. Indeed, if we want to find m_{n+1}, let P be a path $I \to J$ in G_n of length m_n. The path consisting of $(\{1,n+1\},P,\{1,n+1\})$ is a path $I \to J$ in G_{n+1} of length $2+m_n$, so by induction the result follows. The fact that $m_n = 2n-4$ if $n \geq 4$ is somewhat more difficult to establish. It has been proven by Tijdeman [36], by Hajnal, Milner and Szemerédi [17], by Baker and Shostak [3], by Bumby [9] and probably others as well.

Theorem 3.1. *If $n \geq 4$, the minimum number m_n of calls needed to spread all secrets amongst all n participants is equal to $2n - 4$.*

For the proof, see any of the mentioned references.

3.3 Counting in the Gossip graph

We can use the techniques in section 2.6.2 to answer some questions that might arise about G_n. We will only consider the vertices that are reachable from I. We can see a $\{0,1\}$ square matrix as a directed graph. As indicated in section 2.6.2, the computer package 'nauty' can calculate a canonical representative of this directed graph. (An isomorphism of directed graphs corresponds to a relabeling of the participants.) We can then restrict our attention to the canonical representatives. This will save needless calculation. The program 'nauty' can also calculate the size of the automorphism group. We can now find the number of vertices at distance d from I. We can also find the diameter of the graph.

Table 3.1 gives the number of graphs at distance i from I, for $n = 1, \ldots, 8$. The distance of J is given in boldface. For $n \geq 6$, there exist graphs that have distance larger than the distance from I to J. Unsurprisingly, it is possible to gossip inefficiently. Table 3.2 gives the number of isomorphism classes of graphs at distance i from I for $n = 1, \ldots, 8$.

n	$\lvert V\rvert$	d_0	d_1	d_2	d_3	d_4	d_5	d_6
		d_7	d_8	d_9	d_{10}	d_{11}	d_{12}	d_{13}
		d_{14}	d_{15}	d_{16}				
1	1	**1**						
2	2	1	**1**					
3	11	1	3	6	**1**			
4	189	1	6	27	76	**79**		
5	9152	1	10	75	430	1725	4180	**2731**
6	1092473	1	15	165	1475	10605	59145	229816
		477990	**295321**	16520	1420			
7	293656554	1	21	315	3920	41405	369180	2686411
		14916566	56809557	115163279	**86889419**	1578900	981540	
8	166244338221	1	28	546	8876	125475	1556660	16771566
		152871488	1133866839	6418872796	24974052664	55125490756	**5564801976**4	20243146392
		2455893090	73604160	57120				

Table 3.1: Number of graphs d_i at distance i in G_n

n	$\lvert V\rvert$	d_0	d_1	d_2	d_3	d_4	d_5	d_6
		d_7	d_8	d_9	d_{10}	d_{11}	d_{12}	d_{13}
		d_{14}	d_{15}	d_{16}				
1	1	**1**						
2	2	1	**1**					
3	4	1	1	1	**1**			
4	16	1	1	2	5	**7**		
5	111	1	1	2	6	19	46	**36**
6	1940	1	1	2	7	24	103	395
		850	**518**	34	5			
7	68300	1	1	2	7	25	119	656
		3437	13155	26959	**19958**	3716	263	1
8	4651805	1	1	2	7	26	124	734
		4865	32225	179804	702813	1550358	**1546271**	561917
		70430	2223	4				

Table 3.2: Number of isomorphism classes d_i at distance i in G_n

Appendix A

Source code for computing $AT(n)$

This is a C source file. It depends on nauty, the GNU Multiprecision Library, [34] and the file tree.h, which is included in the OpenBSD distribution and is written by Niels Provos. [29]

```
#ifndef N
#error Please define N as a compile-time constant.
#endif

#ifndef ODD_EVEN
#  ifndef REGULAR
#    error Please define either ODD_EVEN or REGULAR.
#  endif
#endif

#define NFACT (9*8*7*6*5*4*3*2*1)
#define MAXN (2*N)

#define MAX_MATCHES NFACT

#include <nauty.h>
#include <assert.h>
#include <gmp.h>

#include "tree.h"

struct tree_node {
    RB_ENTRY(tree_node) entry;
    graph g[MAXM*MAXN];
```

```
    mpz_t n_latin_squares;
};

RB_HEAD(tree_t, tree_node);
RB_PROTOTYPE_STATIC(tree_t, tree_node, entry, graph_compare);

static void dump(graph *g)
{
    for (unsigned int i = 0; i < MAXN; i++) {
        set *gv = GRAPHROW(g, i, MAXM);
        for (unsigned int j = 0; j < MAXN; j++) {
            if (j == N) putchar(' ');
            putchar(ISELEMENT(gv, j) ? '+' : '.');
        }
        putchar('\n');
    }
}

static int graph_compare(struct tree_node* n1,
                         struct tree_node* n2)
{
    for (unsigned int i = 0; i < MAXN; i++) {
        set *gu = GRAPHROW(n1->g, i, MAXM);
        set *gv = GRAPHROW(n2->g, i, MAXM);
#if MAXM != 1
#  error ":( you need to write more code here"
#endif
        if (*gu < *gv) {
            return -1;
        } else if (*gu > *gv) {
            return 1;
        }
    }
    return 0;
}

RB_GENERATE_STATIC(tree_t, tree_node, entry, graph_compare);

static void go(graph g[MAXN*MAXM], int matching[],
               int matches[], int *nmatches)
{
    for (int i = 0; i < MAXN; i++) {
        if (matching[i] != -1) continue;
        for (int j = 0; j < MAXN; j++) {
            if (matching[j] != -1) continue;
            set* gv = GRAPHROW(g, i, MAXM);
            if (ISELEMENT(gv, j)) {
```

```
                matching[i] = j;
                matching[j] = i;
                go(g, matching, matches, nmatches);
                matching[i] = -1;
                matching[j] = -1;
            }
        }
        return;
    }

    /* we are here if we found a new perfect matching */
    /* size of matching = N. */
    assert(*nmatches + 1 < MAX_MATCHES);
    memcpy(matches + MAXN * *nmatches, matching,
                MAXN * sizeof(int));
    *nmatches += 1;
}

static int sign(int matching[])
{
    /* we can do better than this */
    int result = 1;
    for (int i = 0; i < N; i++)
        for (int j = 0; j < i; j++)
            if (matching[i] < matching[j])
                result *= -1;
    return result;

}

/* Note: this is a ridiculous definition. */
static int label_sign(int label[])
{
    int result = 1;
    for (int i = 0; i < MAXN; i++) {
        assert( (i >= N) || (label[i] < N) );
        assert( (i < N) || (label[i] >= N) );
        for (int j = 0; j < i; j++) {
            if (label[i] < label[j])
                result *= -1;
        }
    }
    return result;
}

static void find_matchings(graph g[MAXN*MAXM], int u, int v,
            int matches[], int *nmatches)
```

```
{
    int matching[MAXN];
    int i;
    for (i = 0; i < MAXN; i++) {
        matching[i] = -1;
    }

    matching[u] = v;
    matching[v] = u;

    *nmatches = 0;
    go(g, matching, matches, nmatches);
}

static const unsigned int NAUTY_WORKSPACE_SIZE = MAXM * 1000;
static setword *nauty_workspace = NULL;

static int has_odd_automorphism;

static void my_automproc(int count, permutation *perm,
        int *orbits, int numorbits, int stabvertex, int n)
{
    has_odd_automorphism |= label_sign(perm) == -1;
}

static void go_number(mpz_t result, graph g[MAXN*MAXM],
        struct tree_t *seen_set)
{
    DEFAULTOPTIONS_GRAPH(nauty_options);
    nauty_options.getcanon = TRUE;
    nauty_options.digraph = FALSE;
    nauty_options.defaultptn = FALSE;

    static int depth = 0;
    static int* matches_coll = NULL;
    /* we allocate the matches arrays on the heap, because
       our stack is a bit too small otherwise. we have a
       pointer to our frame of the matches_coll object, and we
       keep a depth of the recursive calls. if we do a
       malloc()/free() in each call of go_number, it's a lot
       slower. */
    if (matches_coll == NULL)
        matches_coll = malloc(MAX_MATCHES * MAXN * N *
                              sizeof(int));
    assert(depth < N);
    int* matches = &matches_coll[MAX_MATCHES * MAXN * depth];
```

```
    /* find an edge */
    set *gv;
    int i, j;
    /* nota mal: this is not as efficient as possible. */
    for (i = 0; i < MAXN; i++) {
        gv = GRAPHROW(g, i, MAXM);
        for (j = 0; j < MAXN; j++) {
            if (ISELEMENT(gv, j))
                goto found_edge;
        }
    }
    /* no edge, we are the empty graph */
    mpz_set_ui(result, 1);
    return;

found_edge: {
        struct tree_node *tn;

        int nmatches;
        find_matchings(g, i, j, matches, &nmatches);
        assert(nmatches > 0);
        int k;

        depth++;

        for (k = 0; k < nmatches; k++) {
            int *m = &matches[MAXN*k];

            /* prepare recursive call */
            for (i = 0; i < MAXN; i++) {
                j = m[i];
                gv = GRAPHROW(g, i, MAXM);
                assert(ISELEMENT(gv, j));
                DELELEMENT(gv, j);
            }

            /* determine if we found the number already. */
            graph canon_graph[MAXM * MAXN];
            int lab[MAXN];
            int ptn[MAXN];
            int orbits[MAXN];
            statsblk stats;

            /* set initial labeling */
            for (i = 0; i < MAXN; i++) {
                lab[i] = i;
                ptn[i] = i == N-1 || i == MAXN-1 ? 0 : 1;
```

```
            }

            has_odd_automorphism = FALSE;
            nauty_options.writeautoms = FALSE;
            nauty_options.userautomproc = my_automproc;
            nauty(g, lab, ptn, NULL, orbits, &nauty_options,
                    &stats, nauty_workspace,
                    NAUTY_WORKSPACE_SIZE, MAXM, MAXN, canon_graph);
#ifdef ODD_EVEN
            if ((N-depth % 2 == 0) && has_odd_automorphism)
                continue;
#endif

            struct tree_node tn_find;
            memcpy(tn_find.g, canon_graph, sizeof(canon_graph));

#ifdef ODD_EVEN
            /* it is purely by accident that nauty always gives us
               an even labeling if we feed it a graph that is the
               canonical representative of its isomorphism class. */
            if (memcmp(g, canon_graph, sizeof(canon_graph)) == 0) {
                assert(label_sign(lab) == 1);
            }
            int sgn = sign(m);
            if (depth % 2 == N % 2) {
                sgn *= label_sign(lab);
            }
#endif
#ifdef REGULAR
            int sgn = 1;
#endif

            tn = RB_FIND(tree_t, seen_set, &tn_find);
            if (tn == NULL) {
                tn = malloc(sizeof *tn);
                mpz_init(tn->n_latin_squares);
                memcpy(tn->g, canon_graph, sizeof(canon_graph));
                go_number(tn->n_latin_squares, canon_graph, seen_set);
                RB_INSERT(tree_t, seen_set, tn);
            }

            if (sgn == 1) {
                /* gmp explicitly allows aliasing */
                mpz_add(result, result, tn->n_latin_squares);
            } else {
                mpz_sub(result, result, tn->n_latin_squares);
            }
```

```
            /* recover from recursive call */
            for (i = 0; i < MAXN; i++) {
                j = m[i];
                gv = GRAPHROW(g, i, MAXM);
                assert(!ISELEMENT(gv, j));
                ADDELEMENT(gv, j);
            }
        }

        depth--;
    }

}


int main(void)
{

    nauty_check(WORDSIZE, MAXM, MAXN, NAUTYVERSIONID);
    nauty_workspace = malloc(NAUTY_WORKSPACE_SIZE *
                             sizeof(*nauty_workspace));

    /* make K_nn */
    graph the_graph[MAXN*MAXM] = {};

    set *gv;
    for (int i = 0; i < 2*N; i++) {
        gv = GRAPHROW(the_graph, i, MAXM);
        EMPTYSET(gv, MAXM);
        for (int j = 0; j < 2*N; j++) {
            if ((i < N) ^ (j < N)) {
                if (i - N == j || j - N == i) continue;
                ADDELEMENT(gv, j);
            }
        }
    }

    struct tree_t seen_set = RB_INITIALIZER(tree_t);
    mpz_t result;
    mpz_init(result);
    go_number(result, the_graph, &seen_set);

#ifdef ODD_EVEN
    int correctfactor = (N*(N-1) / 2) % 2 ? -1 : 1;
    if (correctfactor == -1) {
        mpz_neg(result, result);
```

```
    }
#endif

    char* result_str = mpz_get_str(NULL, 10, result);
#ifdef ODD_EVEN
    printf("The value of AT(%d) is %s\n", N, result_str);
#endif
#ifdef REGULAR
    printf("There are %s reduced Latin squares of order %d\n",
            result_str, N);
#endif
    mpz_clear(result);
    free(result_str);

    /* cleanup */
    free(nauty_workspace);
    struct tree_node *tn, *next;
    for (tn = RB_MIN(tree_t, &seen_set); tn != NULL; tn = next) {
        next = RB_NEXT(tree_t, &seen_set, tn);

        RB_REMOVE(tree_t, &seen_set, tn);
        mpz_clear(tn->n_latin_squares);
        free(tn);
    }

    return 0;
}
```

Appendix B

Source code for verifying $AT(p+2)$

This is a C source file. It depends on the GNU Multiprecision Library. [34]

```
/*
 * Calculate sum j=0..p-1 j! / 2^j mod p, where p is an odd prime number.
 * Finds examples where this value equals 1.
 * Related to a "proof" of AT(p+2) != 0.
 * May 4 2012, Jochem Berndsen.
*/

#include <stdio.h>
#include <gmp.h>
#include <stdlib.h>

static void calc_s(mpq_t, unsigned int);

/* Apologies for the globals. Their introducted saves us some
 * initialization/freeing. Now calc_s is not reentrant unfortunately.
 */
static mpz_t f, t;
static mpq_t tmp;

/*
 * We can be a bit smarter and precompute sum j=0..N j!/2^j in Q for some large
 * number N.  (Note that "overshoot" is not a problem since j!/2^j will be a
 * multiple of p if j >= p.) This would save us some recomputations of the
 * partial sum over the range j=0..N. However, if N is too large the numbers get
 * unwieldy and it becomes beneficial to reduce mod p, I think.  So we could be
 * smart an figure out a "reasonable value" for N.
 */
```

```
static void calc_s(mpq_t s, unsigned int p)
{
    unsigned int k;

    mpz_set_ui(f, 1UL);
    mpz_set_ui(t, 1UL);

    mpq_set_ui(s, 0UL, 1UL);

    for (k = 0; k <= p-1; k++) {
        /* Invariant: f = i!, t = 2^i, s = sum i=0..k-1 i!/2^i */
        mpz_set(mpq_numref(tmp), f);
        mpz_set(mpq_denref(tmp), t);
        mpq_canonicalize(tmp);

        mpq_add(s, s, tmp);

        mpz_mul_ui(f, f, k + 1UL);
        mpz_mul_ui(t, t, 2UL);

        mpz_mod_ui(f, f, p);
        mpz_mod_ui(t, t, p);
        if (k % 5 == 0) {
            mpz_mod_ui(mpq_numref(s), mpq_numref(s), p);
            mpz_mod_ui(mpq_denref(s), mpq_denref(s), p);
            mpq_canonicalize(s);
        }
    }
}



int main(void)
{
    mpq_t s;
    mpq_init(s);

    mpz_init(f);
    mpz_init(t);
    mpq_init(tmp);

    mpz_t p;
    mpz_init_set_ui(p, 3);
    unsigned int i = 0;
    const char progress[] = "-\\|/";
```

```
    do {
        i++;
        if (i % 50 == 0) {
            fprintf(stderr, "\r%c p = %lu",
                    progress[(i/50) % (sizeof(progress)-1)],
                    mpz_get_ui(p));
        }

        calc_s(s, mpz_get_ui(p));
        if (mpq_cmp_ui(s, 1, 1) == 0) {
            fprintf(stderr, "\r");
            printf("Found counterexample! p = %lu\nn", mpz_get_ui(p));
        }
        mpz_nextprime(p, p);
    } while (mpz_fits_uint_p(p) && mpz_cmp_ui(p, 1048576) < 0);

    mpq_clear(s);
    mpz_clear(f);
    mpz_clear(t);
    mpq_clear(tmp);
    mpz_clear(p);
    fprintf(stderr, "\r \n");

    return 0;
}
```

Bibliography

[1] N. Alon and M. Tarsi. Colorings and orientations of graphs. *Combinatorica*, 12:125–134, 1992. 10.1007/BF01204715.

[2] Hua Bai. The Grone-Merris Conjecture. *Transactions of the American Mathematical Society*, 363:4463–4474, 2011.

[3] B. Baker and R. Shostak. Gossips and Telephones. *Discrete Mathematics*, 2:191–193, 1972.

[4] Arthur T. Benjamin, Curtis T. Bennett, and Florence Newberger. Recounting the Odds of an Even Derangement. *Mathematics Magazine*, 78(5):387–390, 2005.

[5] F. Bergeron, G. Labelle, and P. Leroux. *Combinatorial Species and Tree-like Structures*. Cambridge University Press, 1998. Translated from French. ISBN: 0-521-57323-8.

[6] A. Blokhuis. Personal communication.

[7] Andreas Brandstädt, Van Bang Le, and Jeremy P. Spinrad. *Graph Classes: A Survey*. SIAM monographs on discrete mathematics and applications, 1999. ISBN: 0-89871-432-X.

[8] A.E. Brouwer and W.H. Haemers. *Spectral graph theory*. Springer, 2012. ISBN: 978-1-4614-1938-9.

[9] R. Bumby. A problem with telephones. *SIAM Journal on Algebraic and Discrete Methods*, 2:13–18, 1981.

[10] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. The MIT Press, 2001. Second edition, ISBN: 0-262-53196-8.

[11] Arthur A. Drisko. On the Number of Even and Odd Latin Squares of Order $p+1$. *Advances in Mathematics*, 128(1):20–35, 1997.

[12] Arthur A. Drisko. Proof of the Alon-Tarsi Conjecture for $n = 2^r p$. *The Electronic Journal of Combinatorics*, 5, 1998.

[13] David G. Glynn. The modular counterparts of Cayley's hyperdeterminants. *Bulletin of the Australian Mathematical Society*, 57:479–492, 1998.

[14] David G. Glynn. The conjectures of Alon–Tarsi and Rota in dimension prime minus one. *SIAM J. Discrete Math.*, 24(2):394–399, 2010.

[15] Robert Grone and Russell Merris. The Laplacian Spectrum of a Graph II. *SIAM J. Discrete Math.*, 7(2):221–229, 1994.

[16] W. H. Haemers, A. Mohammadian, and B. Tayfeh-Rezaie. On the sum of Laplacian eigenvalues of graphs. *Linear Algebra Appl.*, 432(9):2214–2221, 2010.

[17] A. Hajnal, E.C. Milner, and E. Szemerédi. A cure for the telephone disease. *Canadian Mathematical Bulletin*, 15(3):447–450, 1972.

[18] Peter L. Hammer and Bruno Simeone. The splittance of a graph. *Combinatorica*, 1(3):275–284, 1981.

[19] Rosa Huang and Gian-Carlo Rota. On the relation of various conjectures on Latin squares and straightening coefficients. *Discrete Mathematics*, 128:225–236, 1994.

[20] Jeannette C.M. Janssen. On even and odd latin squares. *J. Combin. Theory Ser. A*, 69(1):173 – 181, 1995.

[21] André Joyal. Une théorie combinatoire des séries formelles. *Advances in Mathematics*, 42:1–82, 1981.

[22] P.A. MacMahon. A new method in combinatory analysis, with applications to latin squares and associated questions. *Trans. Cambridge Phil. Soc.*, 16:262–290, 1898.

[23] A. Marini and G. Pirillo. Signs on latin squares. *Advances in Applied Mathematics*, 15(4):490 – 505, 1994.

[24] Mayank. On variants of the Grone–Merris conjecture. Master's thesis, Eindhoven University of Technology, 2010.

[25] Brendan D. McKay. nauty User's Guide (Version 2.4). `http://cs.anu.edu.au/~bdm/nauty/`, 2009.

[26] Brendan D. McKay, Alison Meynert, and Wendy Myrvold. Small Latin Squares, Quasigroups and Loops. *Journal of Combinatorial Designs*, 15:98–119, 2007.

[27] Brendan D. McKay and Eric Rogoyski. Latin squares of order 10. *Electron. J. Combin.*, 2, 1995.

[28] Brendan D. McKay and Ian M. Wanless. On the number of Latin squares. *Ann. Combin.*, 9:335–344, 2005.

[29] Niels Provos. tree.h. `http://www.freebsd.org/cgi/cvsweb.cgi/src/sys/sys/tree.h`, 2002.

[30] C.D. Olds. Odd and even derangements, Solution E907. *American Mathematical Monthly*, 57:687–688, 1950.

[31] Shmuel Onn. A Colorful Determinantal Identity, a Conjecture of Rota, and Latin Squares. *American Mathematical Monthly*, 104(2):156–159, 1997.

[32] Benjamin C. Pierce. *Basic Category Theory for Computer Scientists*. MIT Press, 1991. ISBN: 0-262-66071-7.

[33] Douglas S. Stones. Formulae for the Alon–Tarsi conjecture. *SIAM J. Discrete Math.*, 26(1):65–70, 2012.

[34] The GNU Project. The GNU Multiprecision Library. `http://gmplib.org/`, 2010.

[35] The GSL Team. GNU Scientific Library – Reference Manual. `http://www.gnu.org/software/gsl/`, 2010.

[36] R. Tijdeman. On a Telephone Problem. *Nieuw Archief voor Wiskunde*, 19:188–192, 1971.

[37] Paolo Zappa. The Cayley determinant of the determinant tensor and the Alon–Tarsi conjecture. *Adv. in Appl. Math.*, 19(1):31 – 44, 1997.