Discrete Math

1 Logic & Proofs

1.1 Propositional Logic

Definitions:

A *proposition* is a declarative statement which is either true or false.
A *compound proposition* is a new proposition formed from existing ones.

Converse: $q \to p$. Isn't logically equivalent to $p \to q$, but is equivalent to the inverse.

Inverse: $\neg p \to \neg q$

Contrapositive: $\neg q \to \neg p$, logically equivalent to $p \to q$.

A *conditional statement*, denoted $p \to q$, is the proposition: "If p, then q". Here p is the *hypothesis* and q is the *conclusion.*

The *biconditional* of $p \to q$ is $p \leftrightarrow q$. It's read as "p if and only if q". Logically equivalent to $p \to q \wedge q \to p$.

Precedence of Logical Operations: Negation, and, or, conditional, biconditional.

Propositional operations, given propositions p and q,

- *Negation* of p, denoted $\neg p$, is the statement: "It is not the case that p".
- *Conjunction* is the proposition "p and q", denoted $p \wedge q$. True if p, q are both true, false otherwise.
- *Disjunction* is the proposition "p or q", denoted $p \vee q$. False if p, q are false, true otherwise.
- *Exclusive OR*, denoted $p \oplus q$, is true only if one of p, q is true, false otherwise.

Bit Operations: True/False represented by 1/0 respectively. OR means $\vee$, AND means $\wedge$, XOR means $\oplus$. They're applied bitwise to a bit string.

1.2 Propositional Equivalences

Definitions:

A *tautology* is a compound proposition that's always true, regardless of truth value of its variables.

A *contradiction*, however, is a compound proposition that's always false, regardless of truth value of its variables.

A *contingency* is a compound proposition that's neither a tautology or a contradiction.

Two statements are *logically equivalent*, denoted $p \equiv q$, if they always have the same truth values. Occurs when $p \leftrightarrow q$ is a tautology. Logical equivalence is often proven using truth tables.

Concepts:

De Morgan's Laws:
(a) $\neg(p \wedge q) \equiv \neg p \vee \neg q$
(b) $\neg(p \vee q) \equiv \neg p \wedge \neg q$

1.3 Predicates & Quantifiers

Definitions:

Predicate: A property the subject can have, denoted by $P(x)$, the propositional function. Once a value is assigned to x, $P(x)$ becomes a proposition. Note: propositional functions can have more than 1 variable.

Quantifiers: Create a proposition from propositional functions by specifying the extent to which the statement is true.

Domain (Universe) of Discourse: Set of allowed values for the proposition involved.

Universal Qualification: "$P(x)$ for all x in the domain", denoted $\forall x P(x)$. A *counterexample* proves $\forall x P(x)$ is false. If finite domain, $\forall x P(x) = P(x_1) \wedge P(x_2) \wedge \cdots \wedge P(x_n)$.

Existential Qualification: "There exists x such that $P(x)$, denoted $\exists x P(x)$. If finite domain, $\exists x P(x) = P(x_1) \vee P(x_2) \vee \cdots \vee P(x_n)$. Note: If domain $= \emptyset$, $\exists x P(x)$ is false.

Uniqueness Qualification: "There exists a unique x such that $P(x)$", denoted $\exists ! x P(x)$.

Concepts:

Qualifiers take precedence over $\wedge / \vee$.
The domain can be restricted within the qualification, e.g. $\forall x > 0 (x^2 > 0)$ or $\exists z ((z > 0) \wedge (z^2 = 2))$.

Negation of Qualifiers:
(a) $\neg \forall x P(x) \equiv \exists x \neg P(x)$

(b) $\neg \exists x P(x) \equiv \forall x \neg P(x)$

1.4 Nested Qualifications

Negation of nested qualifiers: e.g. $\neg \forall x \exists y (xy = 1) \equiv \exists x \neg \forall y (xy = 1) \equiv \exists x \exists y (xy \neq 1)$.

1.5 Introduction to Proofs

Methods of Proof:

- Direct proof: To prove $p \rightarrow q$, assume p, then prove q must follow.
- Proof by contrapositive: Prove $\neg q \rightarrow \neg p$.
- Proof by contradiction: Assume $p \wedge \neg q$, show this leads to a contradiction, thus proving assumption false.
- Proof by cases: aka exhaustive proof. Divide proof into finitely many cases and prove each.
- Existence Proof:
  - Constructive Proof: Explicitly find x such that $P(x)$.
  - Non-constructive Proof: Show $\exists x P(x)$ by some other manner, without finding an explicit x.

Note: To prove $p \leftrightarrow q$, need to prove $p \rightarrow q$ and $q \rightarrow p$. To prove $p_1 \leftrightarrow p_2 \leftrightarrow \cdots \leftrightarrow p_n$, need to prove $p_1 \rightarrow p_2$, $p_2 \rightarrow p_3$, ..., $p_{n-1} \rightarrow p_n$, and $p_n \rightarrow p_1$.

1.6 Proof Methods & Strategy

The phrase, ”without loss of generality” (WLOG), can be used in exhaustive proofs when two cases are essentially the same, except perhaps an issue with labeling.

2 Sets, Sequences, & Summations

A *set* is collection of objects known as *elements.*

Notation:

- For a set A, $a \in A$ means a is an element of A.
- $\mathbb{N}, \mathbb{Z}, \mathbb{Q}, \mathbb{R}, \mathbb{C}$ denote the sets of natural, integral, rational, real, and complex numbers, respectively.
- The *empty set*, denoted $\emptyset = \{\}$, is the set with no elements.

Definitions:

- Sets A, B are *equal* if and only if $x \in A \leftrightarrow x \in B$.
- Set A is a *subset* of B, denoted $A \subseteq B$, if and only if $x \in A \rightarrow x \in B$.
- The *cardinality* of a set A is the number of elements in A, denoted $|A|$.
- The *power set* of A, denoted $\mathcal{P}(A) = \{x | x \subseteq A\}$. If $|A| = n$, then $|\mathcal{P}(A)| = 2^n$.
- For sets A, B, the *Cartesian product* of A, B is $A \times B = \{(a, b) | a \in A \wedge b \in B\}$. In general, $X_1 \times \cdots \times X_n = \{(x_1, \ldots, x_n) | x_i \in X_i\}$.

Set Operations:

- *Union*: $A \cup B = \{x | x \in A \vee x \in B\}$.
- *Intersection*: $A \cap B = \{x | x \in A \wedge x \in B\}$.
- *Difference*: $A - B = \{x | x \in A \wedge x \notin B\}$.
- *Complement*: $A^c = \bar{A} = \{x \in \mathcal{U} | x \notin A\}$, where $\mathcal{U}$ is the domain.

De Morgan's Laws:

- $\overline{A \cap B} = \bar{A} \cup \bar{B}$
- $\overline{A \cup B} = \bar{A} \cap \bar{B}$

3 Algorithms

- All algorithms have an input and an output.
- The steps of an algorithm must be defined precisely.
- An algorithm should produce correct output for any input.
- An algorithm should produce output in a finite number of steps.
- Procedure should be generalized enough so it's applicable to all problems of a desire form.

3.0.1 Searching Algorithms

- *Linear*: Enumeratively search each element of the array for desired element. Return its position if present, 0 otherwise.Average performance: $O(n)$.
- *Binary*: Requires presorted list. Calculates index of middle element, determines if desired element is higher or lower in list. Repeats recursively until element is found, otherwise returns 0. Average performance: $O(\log n)$.

3.0.2 Sorting Algorithms

- *Bubble*: Compares adjacent elements in a pass swapping if necessary. After n passes, the last n elements are properly sorted. Requires at most, $n-1$ passes. Average performance: $O(n^2)$.
- *Insertion*: Sorts first two elements, then first three, and so on. Requires $n-1$ passes through array. Average performance: $O(n^2)$.

3.1 Algorithm Complexity

Big-O Notation: Let f, g be functions from $\mathbb{Z}$ or $\mathbb{R} \mapsto \mathbb{R}$. We say $f(x)$ is $O(g(x))$ if there are constants C, k, such that $|f(x)| \leq C|g(x)|$ for all $x > k$. We call C, k the *witnesses* to the relationship.

- If f is $O(g)$ and g is $O(h)$, then f is $O(h)$.
- A polynomial function with real coefficients of degree n is $O(x^n)$
- If f_1 is $O(g_1)$ and f_2 is $O(g_2)$, then $f_1 + f_2$ is $O(\max(|g_1|, |g_2|))$.
- If f_1 is $O(g_1)$ and f_2 is $O(g_2)$, then $f_1 f_2$ is $O(g_1 g_2)$.

Hierarchy of Big-O Functions: $n^n \to n! \to 2^n \to n^2 \to n \log n \to n \to \log n \to c$.

Big-Ω: Let f, g be functions from $\mathbb{Z}$ or $\mathbb{R} \mapsto \mathbb{R}$. We say $f(x)$ is $\Omega(g(x))$ if there are constants C, k, such that $|f(x)| \geq C|g(x)|$ for all $x > k$.

Big-Θ: If $f(x)$ is $O(g(x))$ and is also $\Omega(g(x))$, then we say that $f(x)$ is $\Theta(g(x))$. Hence, f and g are said to be of the same *order*.

4 Divisibility & Modular Arithmetic

Division Algorithm: Let $a \in \mathbb{Z}$ and $d \in \mathbb{Z}^+$. Then, there exists unique integers q, r with $0 \leq r < d$, such that $a = dq + r$. Note: $q = a$ div d and $r = a$ mod b.

Let $a, b \in \mathbb{Z}$ and $m \in \mathbb{Z}^+$. We say a and b are congruent modulo m, if m divides $a - b$. This is denoted $a \equiv b \pmod{m}$.

If $a \equiv b \pmod{m}$ and $c \equiv d \pmod{m}$, then $(a + c) \equiv (b + d) \pmod{m}$ and $ac \equiv bd \pmod{m}$.

Arithmetic Operations: Let $\mathbb{Z}_m = \{0, 1, \ldots, m-1\}$, then

- Define $+_m$ on $\mathbb{Z}_m$ such that for $a, b \in \mathbb{Z}_m,\ a +_m b = (a + b) \pmod{m}$.
- Define $\cdot_m$ on $\mathbb{Z}_m$ such that for $a, b \in \mathbb{Z}_m,\ a \cdot_m b = (ab) \pmod{m}$.

4.1 Alternate Number Bases

Let $b, n \in \mathbb{Z}^+$, where $b > 1$. Then, n can be expressed in the form:
$n = a_k b^k + a_{k-1} b^{k-1} + \cdots + a_1 b + a_0$, where k is a non-negative integer, and $0 \leq a_0, a_1, \ldots, a_k < b$ and $a_k \neq 0$.

4.2 Primality & GCD

An integer $p > 1$ is *prime* if the only positive factors of p are 1 and p. If p isn't prime, then it is *composite*.

Fundamental Theorem of Arithmetic: Every integer greater than 1 can be uniquely written as a prime or the product of primes, where the prime factors are written in order of non-decreasing size.

If n is composite, then n has a prime divisor less than or equal to $\sqrt{n}$.

Clearly, $ab = \text{lcm}(a,b) \cdot \gcd(a,b)$.

Euclidean Algorithm: The basic premise is that if $a = bq + r$, for integers a, b, q, r, such that $0 \leq r < b$, then, $\gcd(a,b) = \gcd(b,r)$.

Bézout's Theorem: If a and b are positive integers, then there exist integers s and t such that $\gcd(a,b) = sa + tb$. Such integers are known as *Bézout coefficients*, and can be determined by reverse engineering the Euclidean Algorithm.

From Bézout's, it follows that, if $a, b, c \in \mathbb{Z}^+$ such that $\gcd(a,b) = 1$ and $a|(bc)$, then $a|c$. Therefore, if p is a prime, and $p|a_1 a_2 \cdots a_n$, then $p|a_i$ for some i.

Let m be a positive integer, and let a, b, c be integers. If $ac \equiv bc \pmod{m}$ and $\gcd(c,m) = 1$, then $a \equiv b \pmod{m}$.

4.3 Solving Modular Congruences

A congruence of the form $ax \equiv b \pmod{m}$, where m is a positive integer, a, b are integers, and x is a variable, is a *linear congruence.*

We say an integer $\bar{a}$ is an *inverse* of a modulo m if $a\bar{a} \equiv 1 \pmod{m}$. If $\gcd(a,m) = 1$, then an inverse of a modulo m exists. Furthermore, this inverse is unique modulo m. If however a and m aren't relatively prime, then no such inverse exists.

Therefore, a method to solve a linear congruence is: $a\bar{a}x \equiv x \equiv \bar{a}b \pmod{m}$.

Chinese Remainder Theorem: Let $m_1, m_2, \ldots, m_n$ be pairwise relatively prime integers greater than 1 and $a_1, a_2, \ldots, a_n$ be arbitrary integers. Then the system

$$\begin{aligned} x &\equiv a_1 \pmod{m_1} \\ x &\equiv a_2 \pmod{m_2} \\ &\vdots \\ x &\equiv a_n \pmod{m_n} \end{aligned}$$

has a unique solution modulo $m_1 m_2 \cdots m_n$.

To solve systems of linear congruences via CRT, we define $\hat{m}_j = \frac{m_1 m_2 \cdots m_n}{m_j}$, and define $\hat{y}_j$ to be the inverse of $\hat{m}_j$ modulo m_j, which clearly exists since $\gcd(m_j, \hat{m}_j) = 1$. Therefore, we have $\hat{m}_j \hat{y}_j \equiv 1 \pmod{m_j}$, for all $1 \leq j \leq n$.

Now, consider

$$x = \sum_{j=1}^{n} a_j \hat{m}_j \hat{y}_j$$

Since $a_j \hat{m}_j \hat{y}_j \equiv a_j \pmod{m_j}$ and $a_j \hat{m}_j \hat{y}_j \equiv 0 \pmod{m_k}$ for all $k \neq j$, x is clearly our desired solution.

Fermat's Little Theorem: If p is prime and a is an integer such that $p \nmid a$, then

$$a^{p-1} \equiv 1 \pmod{p}$$

Furthermore, for every integer a,

$$a^p \equiv a \pmod{p}$$

Proof. Let $S = \{i\}_{i=1}^{p-1}$ and $a \cdot S = \{ai\}_{i=1}^{p-1}$. We claim $a \cdot S$ is simply a permutation of S modulo p, i.e.

$$S \equiv \{ai\}_{i=1}^{p-1} \pmod p$$

Clearly none of the ai for $1 \leq i \leq p-1$ are divisible by p, so it suffices to show that all of the elements in $a \cdot S$ are distinct. Suppose that $ai \equiv aj \pmod p$ for $i \neq j$. Since $\gcd(a, p) = 1$, by the cancellation rule, that reduces to $i \equiv j \pmod p$, which is a contradiction.

Thus, modulo p, we have that the product of the elements of S is

$$1a \cdot 2a \cdots (p-1)a \equiv 1 \cdot 2 \cdots (p-1) \pmod p.$$

Cancelling the factors $1, 2, 3, \ldots, p-1$ from both sides, we are left with $a^{p-1} \equiv 1 \pmod p$.

□

4.4 Cryptography

Shift Cipher: A text cipher is applied to a message M, which translates the characters into numbers. Then, the individual numbers are encrypted by $f(p) = (p + k) \mod 26$, for some integer k (key). Messages can then be decrypted via, $f^{-1}(p) = (p - k) \mod 26$.

Affine Cipher: Similar in nature to a shift cipher except the encryption function is of the form $f(p) = (ap + b) \mod 26$, and the decryption function will be $f^{-1}(q) = \bar{a}(q - b) \mod 26$, where $\gcd(a, 26) = 1$, and $\bar{a}$ is the canonical inverse of a modulo 26.

Both the aforementioned ciphers, are called **character ciphers**, and are vulnerable to attack by an analysis of letter frequency. A more effective encryption method is known as a **block cipher**, which encrypts whole blocks of letters at a time.

RSA Encryption: A type of public key, block cipher. The advantage of having a public key is that no previous communication is required between two individuals before an encrypted message can be sent. The encryption function for a message M is $C = M^e \mod n$, where the integers e, n are public knowledge.

Suppose $n = pq$, for large primes p, q and $\gcd(e, (p-1)(q-1)) = 1$. If d is the inverse of e modulo $(p-1)(q-1)$, then the decryption function is given by $M = C^d \mod n$. Notice, that

$$C^d \equiv (M^e)^d \equiv M^{de} \equiv M^{1+k(p-1)(q-1)} \mod n.$$

Furthermore, by Fermat's Little Theorem,

$$C^d \equiv M \cdot (M^{p-1})^{k(q-1)} \equiv M \mod p.$$
$$C^d \equiv M \cdot (M^{q-1})^{k(p-1)} \equiv M \mod q.$$

Therefore, by the Chinese Remainder Theorem, since $\gcd(p, q) = 1$,

$$C^d \equiv M \mod pq.$$

This is an effective cryptosystem because without knowledge of p, q, the integer d cannot be easily determined, since prime factorization is currently an NP problem.

5 Proof by Induction

Used to prove a propositional statement, $P(n)$, is true for all integers $n \geq b$.

Principle of Mathematical Induction:

- Base Case: Verify that $P(b)$ is true.
- Inductive Step: Using the inductive hypothesis, i.e. "assume that $P(k)$ holds for an arbitrary fixed integer $k \geq b$", prove $P(k) \Rightarrow P(k+1)$.

Strong Induction:
Similar to "weak" induction however, the inductive hypothesis becomes a stronger assumption, i.e. assume $[P(b) \wedge P(b+1) \wedge \cdots \wedge P(k)]$, then show this implies $P(k+1)$ holds. Notice that it may involve multiple base cases.

6 Counting

Principle of Inclusion-Exclusion: Given two sets A, B, where $|A|, |B| < \infty$, we have $|A \cup B| = |A| + |B| - |A \cap B|$.

Pigeonhole Principle: If $k \in \mathbb{Z}^+$, and $k+1$ or more objects are placed into k boxes, then there's at least one box containing two or more objects. In general, if N objects are placed into k boxes, then at least one box contains $\lceil \frac{N}{k} \rceil$ objects.

Pigeonhole Applications:

- A function f from a set of $k+1$ elements to a set of k elements cannot be a bijection.
- Every sequence of n^2+1 distinct real numbers contains a subsequence of length $n+1$ that is either strictly increasing or decreasing.

Permutation: An ordered arrangement of a set of distinct objects.
Combination: Number of permutations but without regard to order.

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

A *combinatorial proof* of an identity is a proof that uses counting arguments to prove that both sides of an identity count the same number of objects but it a different way (*double counting proof*), or that shows there's a bijection between the sets of objects counted by two sides of the identity.

$$(x+y)^n = \sum_{k=0}^{n} \binom{n}{k} x^{n-k} y^k$$

7 Recurrence Relations

A degree k linear homogeneous constant coefficent relation is of the form

$$a_n = c_1 a_{n-2} + c_2 a_{n-2} + \cdots + c_k a_{n-k}.$$

The characteristic equation for this relation is

$$r^k - c_1 r^{k-1} - \cdots - c_{k-1} r - c_k = 0.$$

Suppose characteristic equation has k distinct roots, $r_1, r_2, \ldots, r_k$. Then, the general solution for the recurrence relation is

$$a_n = \alpha_1 (r_1)^n + \alpha_2 (r_2)^n + \cdots + \alpha_k (r_k)^n,$$

where $\alpha_1, \alpha_2, \ldots, \alpha_n$ are fixed coefficients, uniquely determined by k initial conditions.

If all the roots are not distinct, say root r_j has multiplicity m_j, then $r_j^n, nr_j^n, n^2 r_j^n, \ldots, n^{m_j - 1} r_j^n$, will substituted in for the r_j terms in the general solution from above.

8 Graph Theory

A **graph**, $G = (V, E)$, consists of a non-empty set of vertices, V, and a set of edges E.

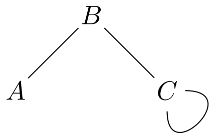


Figure 1: An undirected graph.

Figure 1 depicts an undirected graph given by $V = \{A, B, C\}$ and $E = \{\{A, B\}, \{B, C\}, \{C, C\}\}$. We call an edge that connects a vertex to itself a **loop**. A **simple** graph has no loops or multiple edges connecting two vertices.

A **directed graph** is a graph whose edges are associated with an order pair of vertices.

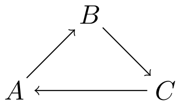


Figure 2: A directed graph.

Vertices sharing an edge are called **adjacent** or **neighbors**. We say an edge is **incident** with its vertices. For a vertex, v, the **neighborhood** of v, denoted $N(v)$, is the set of all its neighbors. The **degree** of a vertex, denoted $\deg(v)$, in an undirected graph, is the number of edges incident on it. We say a vertex is **isolated** if $\deg(v) = 0$ or **pendent** if $\deg(v) = 1$.

Handshaking Theorem: For an undirected graph, $G = (V, E)$, with m edges,

$$\sum_{v \in V} \deg(v) = 2m.$$

As a corollary, we see that an undirected graph must have an even number of vertices of odd degree.

Similarly, for a directed graph, if we define $\deg^-(v)$ as the number of edges with v as a terminal point, and $\deg^+(v)$ as the number of edges with v as an initial point, then,

$$\sum_{v \in V} \deg^-(v) = |E| = \sum_{v \in V} \deg^+(v).$$

8.1 Special Graphs

A **complete graph** on n vertices, denoted K_n, is a simple graph with 1 edge between each pair of vertices. In general, K_n will have $\binom{n}{2}$ edges.

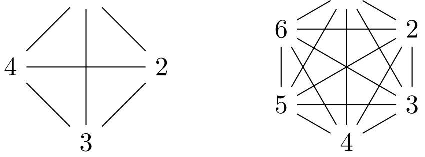


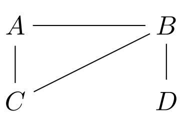
Figure 3: Depictions of K_4 and K_6 respectively.

A **cycle**, C_n, consists of $n \geq 3$ vertices, $v_1, \ldots, v_n$ and edges $\{v_1, v_2\}, \{v_2, v_3\}, \ldots, \{v_n, v_1\}$. Note that figure 2 is C_3.

A **wheel**, W_n, is a cycle graph (C_{n-1}) with one additional vertex which connects to all previous vertices.

8.2 Graph Isomorphism

An **adjacency matrix** for a simple undirected graph is a matrix whose $(i, j)^{th}$ entry is 1 if the vertices i and j are adjacent and 0 otherwise. For a non-simple graph, the entries are the number of edges associated with v_i, v_j.



$$\begin{array}{c} \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \end{array} \begin{array}{c} A \\ B \\ C \\ D \end{array}$$

Figure 4: An undirected graph and its corresponding adjacency matrix.

We say two graphs, $G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2)$ are **isomorphic** if there is a one-to-one function $f : V_1 \to V_2$, such that a, b are adjacent in V_1 if and only if $f(a)$ and $f(b)$ are adjacent in V_2. Such a function is called an isomorphism. Any two given graphs, G and H are isomorphic if and only if there is an ordering on the vertices of the graphs for which their adjacent matrices are equal.

The following properties are graph invariant and are necessary, but not sufficient for isomorphism.

- Need same number of vertices.
- Need same number of edges.
- Number of vertices of a given degree must be the same.

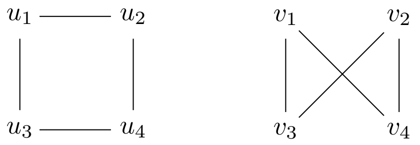


Figure 5: Example of isomorphic graphs.

8.3 Graph Connectedness

A **path** of length k in an undirected graph is a sequence of edges, $e_1, e_2, \ldots, e_k$, such that e_i, e_{i+1} share a vertex v_i. A **circuit** is a path that starts and ends at the same vertex. A simple path or circuit does not traverse the same edge more than once.

An undirected graph is **connected** if there is a path between each pair of distinct vertices.

A directed graph is **strongly connected** if there is a path from a to b and b to a for every pair of vertices in the graph. It's **weakly connected** if the underlying undirected graph is connected.

An **Euler circuit** in a graph G is a simple circuit containing every edge of G. An **Euler path** is a simple path containing every edge of G.

A connected graph with at least two vertices has an Euler circuit if and only if each of its vertices has even degree.

A connected graph has an Euler path but not an Euler circuit if and only if it has exactly two vertices of odd degree.

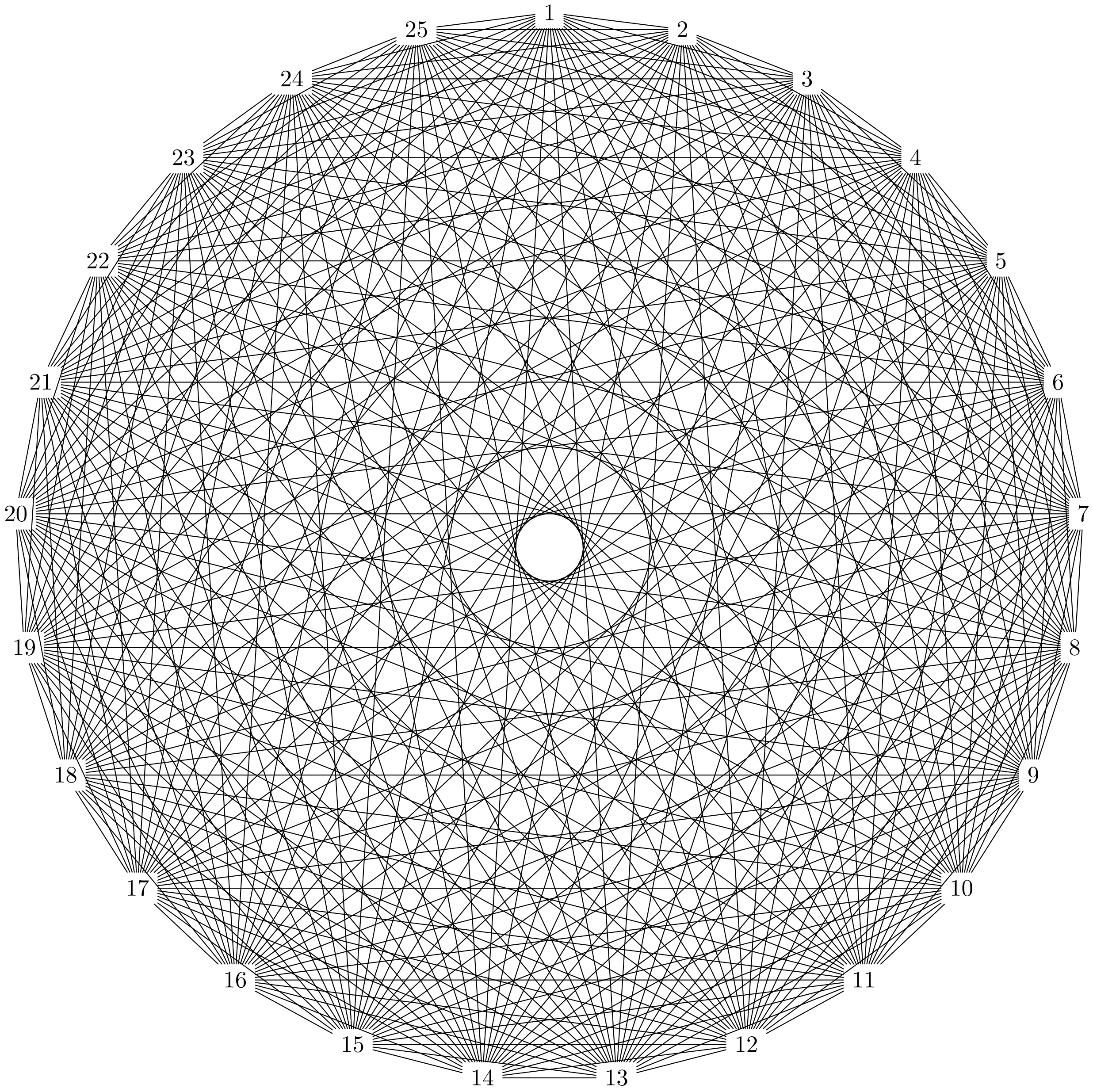


Figure 6: K_{25} (*300* edges.)