CONWAY'S 99-GRAPH PROBLEM: A BOOLEAN SATISFIABILITY APPROACH

NATHANIEL SELUB

Abstract. In this paper, we present a framework for solving Conway's 99-graph problem using a Boolean satisfiability problem (SAT) solver. We begin by encoding the problem into conjunctive normal form (CNF). Then, we append symmetry-breaking clauses to the CNF encoding in order to improve SAT solver performance by preventing it from exploring isomorphic regions of the search space. Finally, we describe several basic substructures that must be contained in any possible Conway 99-graph, which aid in fixing variables in our SAT problem and simplifying the CNF encoding.

Contents

1. Introduction

Conway's 99-graph problem poses the following question: Does there exist an undirected graph with 99 vertices such that each edge belongs to a unique triangle and each non-adjacent vertex pair belongs to a unique quadrilateral?

A **Boolean formula** is an expression built from Boolean variables (variables that are either `TRUE` or `FALSE`), Boolean operators (`AND`, `OR`, `NOT`), and parentheses. An assignment of truth values to these variables is called an **interpretation**. The **Boolean Satisfiability Problem** (SAT) is to determine if a given Boolean formula can be made true. If there exists an interpretation that results in the formula being `TRUE`, the formula is called **satisfiable**, and if every possible interpretation results in the formula being `FALSE`, the formula is called **unsatisfiable**. A program designed to check the satisfiability of a SAT instance is called a SAT solver.

Conway's 99-graph problem can be expressed as a SAT problem. Consider $4851 = \binom{99}{2}$ Boolean variables $e_{1,2}, e_{1,3}, \ldots, e_{98,99}$. Here, $e_{i,j}$ indicates the presence (`TRUE`) or absence (`FALSE`) of an edge between vertices i and j. Let F be a Boolean formula whose arguments are the aforementioned Boolean variables such that F is `TRUE` if and only if each edge belongs to a unique triangle and each non-adjacent vertex pair belongs to a unique quadrilateral. Then, Conway's 99-graph problem is equivalent to determining the satisfiability of F.

A decision problem yields a "yes" or "no" answer for an input. In computational complexity theory, the nondeterministic polynomial time class (NP) is the set of all decision problems such that "yes" instances have solutions whose correctness can be verified in polynomial time. A problem is NP-complete if it is in NP and can emulate any other problem in NP with a solution that can be verified quickly. It is known that SAT is NP-complete.

In general, NP-complete problems are hard to solve. However, modern SAT solvers are very effective. As of 2023, they can process instances with nearly one billion variables and billions of clauses [2]. Conway's 99-graph problem can be encoded as a SAT instance whose number of variables and clauses is well within these bounds, which makes it a candidate for resolution with a SAT solver.

This paper is structured as follows: In Section 2, we define conjunctive normal form (CNF) and show how to encode Conway's 99-graph problem as a Boolean formula in CNF. Section 3 introduces the concept of a symmetry of a Boolean formula and demonstrates how to introduce symmetry-breaking clauses to improve SAT solver performance. In Section 4, we identify a substructure in a putative Conway's 99-graph, which allows us to fix certain variables in the CNF encoding and simplify the final CNF formula. Finally, Section 5 outlines potential methods to solve the Conway's 99-graph problem using the techniques described in this paper.

2. Encoding the Problem in Conjunctive Normal Form

2.1. **Conjunctive Normal Form.** A **truth function** is a function that takes truth values as input and yields a unique truth value as output. For example:

- The `AND` operator, represented as $\wedge$, is the truth-functional operator of conjunction.
- The `OR` operator, represented as $\vee$, is the truth-functional operator of disjunction.

Furthermore, we denote the inverse of a Boolean variable x as $\bar{x} \equiv \neg x$, where $\neg$ is the logical `NOT` operator.

We define a **clause** as a disjunction of literals. A formula is said to be in **conjunctive normal form** (CNF) if it consists of a conjunction of one or more clauses. Thus, one can view a CNF formula as an "`AND` of `OR`'s."

Given the variables A, B, C, D, E, F, the following are examples of formulas in conjunctive normal form:

(1) $(A \vee \bar{B} \vee \bar{C}) \wedge (\bar{D} \vee E \vee F)$
(2) $(A \vee B) \wedge (C)$
(3) $(A \vee B)$
(4) (A)

By convention, a SAT instance needs to be in CNF in order to be processed by modern SAT solvers.

2.2. **Encoding the problem as a Boolean formula.** Consider $\binom{n}{2}$ Boolean variables $e_{1,2}, e_{1,3}, \ldots, e_{n-1,n}$. Here, $e_{i,j}$ indicates the presence (`TRUE`) or absence (`FALSE`) of an edge between vertices v_i and v_j. The ordering of the indices is unimportant, i.e., $e_{i,j} = e_{j,i}$. For each variable $e_{i,j}$, there are two Boolean formulae associated with it

- If $e_{i,j}$, then there must exist a unique triangle containing $e_{i,j}$.
- If $\bar{e}_{i,j}$, then there must exist a unique quadrilateral containing v_i and v_j.

We refer to the first of the above Boolean formulae as the triangle condition for $e_{i,j}$, which we denote as $T_{i,j}$, and the second of the above formulae as the quadrilateral condition for $e_{i,j}$, which we denote as $Q_{i,j}$.

First, we encode the triangle condition for $e_{i,j}$ as a Boolean formula. Observe that if $e_{i,j}$ is `TRUE`, then there exists a triangle containing $e_{i,j}$ if and only if there exists a vertex v_k such that k is distinct from i, j and both $e_{i,k}$ and $e_{j,k}$ are `TRUE`. Therefore, if $e_{i,j}$ is `TRUE`, then a triangle containing $e_{i,j}$ exists if and only if

$$T_{i,j,E} \equiv \bigvee_{\substack{1 \le k \le n \\ k \neq i,j}} e_{i,k} \wedge e_{j,k}$$

is `TRUE`. We refer to $T_{i,j,E}$ as the triangle existence condition for $e_{i,j}$.

Observe that if $e_{i,j}$ is `TRUE`, then there does not exist more than one triangle containing $e_{i,j}$ if and only if there does not exist two vertices v_{k_1}, v_{k_2} such that k_1, k_2, i, j are distinct and $e_{i,k_1}, e_{j,k_1}, e_{i,k_2}, e_{j,k_2}$ are `TRUE`. Therefore, if $e_{i,j}$ is `TRUE`, then there does not exist more than one triangle containing $e_{i,j}$ if and only if

$$\begin{aligned} T_{i,j,U} &\equiv \bigwedge_{\substack{1 \le k_1 < k_2 \le n \\ k_1, k_2 \neq i,j}} \neg \left(e_{i,k_1} \wedge e_{j,k_1} \wedge e_{i,k_2} \wedge e_{j,k_2} \right) \\ &= \bigwedge_{\substack{1 \le k_1 < k_2 \le n \\ k_1, k_2 \neq i,j}} \bar{e}_{i,k_1} \vee \bar{e}_{j,k_1} \vee \bar{e}_{i,k_2} \vee \bar{e}_{j,k_2}, \end{aligned}$$

where we have used De Morgan's Law. We refer to $T_{i,j,U}$ as the triangle uniqueness condition for $e_{i,j}$.

Using the definition of $a \implies b$, the triangle condition for $e_{i,j}$ can be written as

$$\begin{aligned} T_{i,j} &= \bar{e}_{i,j} \vee (T_{i,j,E} \wedge T_{i,j,U}) \\ &= (\bar{e}_{i,j} \vee T_{i,j,E}) \wedge (\bar{e}_{i,j} \vee T_{i,j,U}). \end{aligned}$$

Next, we encode the quadrilateral condition as a Boolean formula. Observe that there exists a quadrilateral containing v_i and v_j if and only if there exist two vertices k_1, k_2 such that k_1, k_2, i, j are distinct and e_{i,k_1}, e_{j,k_1}, e_{i,k_2}, e_{j,k_2} are `TRUE`. Therefore, a quadrilateral containing $e_{i,j}$ exists if and only if

$$Q_{i,j,E} \equiv \bigvee_{\substack{1 \le k_1 < k_2 \le n \\ k_1, k_2 \neq i,j}} e_{i,k_1} \wedge e_{j,k_1} \wedge e_{i,k_2} \wedge e_{j,k_2}$$

is `TRUE`. We refer to $Q_{i,j,E}$ as the quadrilateral existence condition for $e_{i,j}$.

Observe that there does not exist more than one quadrilateral containing $e_{i,j}$ if and only if there does not exist three vertices $v_{k_1}, v_{k_2}, v_{k_3}$ such that k_1, k_2, k_3, i, j are distinct and e_{i,k_1}, e_{j,k_1}, e_{i,k_2}, e_{j,k_2}, e_{i,k_3}, e_{j,k_3} are `TRUE`. Therefore, there does not exist more than one quadrilateral containing $e_{i,j}$ if and only if

$$\begin{aligned} Q_{i,j,U} &\equiv \bigwedge_{\substack{1 \le k_1 < k_2 < k_3 \le n \\ k_1, k_2, k_3 \neq i,j}} \neg (e_{i,k_1} \wedge e_{j,k_1} \wedge e_{i,k_2} \wedge e_{j,k_2} \wedge e_{i,k_3} \wedge e_{j,k_3}) \\ &= \bigwedge_{\substack{1 \le k_1 < k_2 < k_3 \le n \\ k_1, k_2, k_3 \neq i,j}} \bar{e}_{i,k_1} \vee \bar{e}_{j,k_1} \vee \bar{e}_{i,k_2} \vee \bar{e}_{j,k_2} \vee \bar{e}_{i,k_3} \vee \bar{e}_{j,k_3}, \end{aligned}$$

where we have used De Morgan's Law. We refer to $Q_{i,j,U}$ as the quadrilateral uniqueness condition for $e_{i,j}$. Using the definition of $a \implies b$, we can write the quadrilateral condition for $e_{i,j}$ as

$$\begin{aligned} Q_{i,j} &= e_{i,j} \vee (Q_{i,j,E} \wedge Q_{i,j,U}) \\ &= (e_{i,j} \vee Q_{i,j,E}) \wedge (e_{i,j} \vee Q_{i,j,U}) \end{aligned}$$

Define the Boolean formula

$$F_n \equiv \bigwedge_{1 \le i < j \le n} T_{i,j} \wedge Q_{i,j}.$$

Then, a graph satisfies the conditions of Conway's 99-graph problem if and only if the Boolean formula F_{99} evaluates to `TRUE` for the Boolean variables corresponding to the graph's edges. Furthermore, a graph satisfying the conditions of Conway's 99-problem exists if and only F_{99} is satisfiable.

2.3. **Converting the Boolean formula to Conjunctive Normal Form.** Observe that in order to transform F_n into CNF, it suffices to transform each $\bar{e}_{i,j} \vee T_{i,j,E}$ and $e_{i,j} \vee Q_{i,j,E}$ into CNF. Fix $i < j$ and consider

$$T_{i,j,E} = \bigvee_{\substack{1 \le k \le n \\ k \neq i,j}} e_{i,k} \wedge e_{j,k}.$$

If we attempt to convert this to CNF by expanding this formula, the result is

$$(e_{i,1} \vee e_{i,2} \vee \cdots \vee e_{i,n}) \wedge (e_{j,1} \vee e_{i,2} \vee \cdots \vee e_{i,n}) \wedge (e_{i,1} \vee e_{j,2} \vee \cdots \vee e_{i,n}) \wedge (e_{j,1} \vee e_{j,2} \vee \cdots \vee e_{i,n}) \wedge \cdots \wedge (e_{j,1} \vee e_{j,2} \vee \cdots \vee e_{j,n}),$$

where we note that all terms of the form $e_{i,j}$, $e_{i,i}$, and $e_{j,j}$ are absent in the above formula. The result is a formula with 2^{n-2} clauses, which, for $n = 99$, is far too large for any computer to handle.

Let F and F' be two Boolean formulae. Then, F and F' are **equisatisfiable** if F is satisfiable if and only F' is satisfiable, and **equivalent** if $F = F'$ for all possible input variables. There exist transformations into CNF that avoid an exponential increase in size by transforming the original formula into a new formula such that the original formula and the new formula are equisatisfiable rather than equivalent.

First, introduce $\binom{n}{3}$ auxiliary variables $t_{i,j,k}$ for all $1 \le i < j < k \le n$, where the ordering of the subscript indices is unimportant, e.g., $t_{i,j,k} = t_{j,i,k}$, such that $t_{i,j,k}$ is `TRUE` if and only if the triangle containing the sides $e_{i,j}$, $e_{j,k}$, $e_{k,i}$ exists, i.e., $t_{i,j,k} \equiv e_{i,j} \wedge e_{j,k} \wedge e_{k,i}$. For each auxiliary variable $t_{i,j,k}$, we enforce the condition $t_{i,j,k} \equiv e_{i,j} \wedge e_{j,k} \wedge e_{k,i}$ by appending the clauses

$$T_{i,j,k} \equiv (\bar{t}_{i,j,k} \vee e_{i,j}) \wedge (\bar{t}_{i,j,k} \vee e_{j,k}) \wedge (\bar{t}_{i,j,k} \vee e_{k,i}) \wedge (\bar{e}_{i,j} \vee \bar{e}_{j,k} \vee \bar{e}_{k,i} \vee t_{i,j,k})$$

to our CNF encoding.

For each $T_{i,j,E}$, we define a new CNF formula

$$T'_{i,j,E} \equiv \bigvee_{\substack{1 \le k \le n \\ k \neq i,j}} t_{i,j,k}.$$

Let

$$\begin{aligned} \tilde{T}_{i,j,E} &\equiv \bar{e}_{i,j} \vee T'_{i,j,E} \\ &= \bar{e}_{i,j} \vee \left[\bigvee_{\substack{1 \le k \le n \\ k \neq i,j}} t_{i,j,k} \right] \end{aligned}$$

and

$$\tilde{T}_{i,j,U} \equiv \bar{e}_{i,j} \vee T_{i,j,U}.$$

Let

$$\tilde{T}_{i,j} \equiv T_{i,j,U} \wedge T_{i,j,E}.$$

Then, by construction,

$$\left[\bigwedge_{1 \le i < j \le n} \tilde{T}_{i,j} \right] \wedge \left[\bigwedge_{1 \le i < j < k \le n} T_{i,j,k} \right]$$

is equisatisfiable with

$$\bigwedge_{1 \le i < j \le n} T_{i,j}$$

For the analogous formula $e_{i,j} \vee Q_{i,j,E}$ resulting from the quadrilateral existence condition, we can avoid the corresponding exponential growth in clauses that results from a naive conversion to CNF by proceeding in exactly the same manner.

First, introduce $3 \cdot \binom{n}{4}$ auxiliary variables $q_{(a,b),(c,d)}$ for all $((a,b),(c,d))$ such that $1 \le a,b,c,d \le n$ and a,b,c,d are distinct, where the ordering of the tuples and the ordering of the subscripts in each individual tuple is unimportant, e.g., $q_{(a,b),(c,d)} = q_{(c,d),(a,b)}$ and $q_{(a,b),(c,d)} = q_{(b,a),(c,d)}$, such that $q_{(a,b),(c,d)}$ is `TRUE` if and only if the quadrilateral containing the sides $e_{a,c}$, $e_{c,b}$, $e_{b,d}$, $e_{d,a}$ exists, i.e., $q_{(a,b),(c,d)} \equiv e_{a,c} \wedge e_{c,b} \wedge e_{b,d} \wedge e_{d,a}$. For each auxiliary variable $q_{(a,b),(c,d)}$, we enforce the condition $q_{(a,b),(c,d)} \equiv e_{a,c} \wedge e_{c,b} \wedge e_{b,d} \wedge e_{d,a}$ by appending the clauses

$$\begin{aligned} Q_{(a,b),(c,d)} \equiv\ & (\bar{q}_{(a,b),(c,d)} \vee e_{a,c}) \wedge (\bar{q}_{(a,b),(c,d)} \vee e_{c,b}) \wedge (\bar{q}_{(a,b),(c,d)} \vee e_{b,d}) \wedge (\bar{q}_{(a,b),(c,d)} \vee e_{d,a}) \\ & \wedge (\bar{e}_{a,c} \vee \bar{e}_{c,b} \vee \bar{e}_{b,d} \vee \bar{e}_{d,a} \vee q_{(a,b),(c,d)}) \end{aligned}$$

to our CNF encoding.

For each $Q_{i,j,E}$, we define a new CNF formula

$$Q'_{i,j,E} \equiv \bigvee_{\substack{((a,b),(c,d)) \\ 1 \le a,b,c,d \le n \\ a,b,c,d \text{ distinct}}} q_{(a,b),(c,d)}.$$

Let

$$\begin{aligned} \tilde{Q}_{i,j,E} &\equiv e_{i,j} \vee Q'_{i,j,E} \\ &= e_{i,j} \vee \left[\bigvee_{\substack{((a,b),(c,d)) \\ 1 \le a,b,c,d \le n \\ a,b,c,d \text{ distinct}}} q_{(a,b),(c,d)} \right]. \end{aligned}$$

and

$$\tilde{Q}_{i,j,U} \equiv e_{i,j} \vee Q_{i,j,U}.$$

Let

$$\tilde{Q}_{i,j} \equiv Q_{i,j,E} \wedge Q_{i,j,U}.$$

Then, by construction,

$$\left[\bigwedge_{1\leq i<j\leq n} \tilde{Q}_{i,j}\right] \wedge \left[\bigvee_{\substack{((a,b),(c,d))\\ 1\leq a,b,c,d\leq n\\ a,b,c,d \text{ distinct}}} Q_{(a,b),(c,d)}\right]$$

is equisatisfiable with

$$\bigwedge_{1\leq i<j\leq n} Q_{i,j}.$$

Let

$$\tilde{F}_n = \left[\bigwedge_{1\leq i<j\leq n} \tilde{T}_{i,j} \wedge \tilde{Q}_{i,j}\right] \wedge \left[\bigwedge_{1\leq i<j<k\leq n} T_{i,j,k}\right] \wedge \left[\bigwedge_{\substack{(a,b),(c,d)\\ 1\leq a,b,c,d\leq n\\ a,b,c,d \text{ distinct}}} Q_{(a,b),(c,d)}\right].$$

By construction, $\tilde{F}_n$ is a Boolean formula in CNF that is equisatisfiable with F_n.

3. Breaking Symmetries in the Conjunctive Normal Form

A Boolean formula F of Boolean variables $x_1, \ldots, x_n$ is a function from $\{0,1\}^n \to \{0,1\}$. Let $\mathrm{Aut}(\{x_1, \ldots, x_n\})$ be the set of all automorphisms of the set $\{x_1, \ldots, x_n\}$. Then, we define a symmetry of F as any automorphism $\phi \in \mathrm{Aut}(\{x_1, \ldots, x_n\})$ such that the following diagram commutes

$$\begin{array}{ccc} \{x_1,\ldots,x_n\} & \xrightarrow{\phi} & \{x_1,\ldots,x_n\} \\ \downarrow{\scriptstyle \psi} & \swarrow{\scriptstyle \psi} & \\ \{0,1\}^n & \xrightarrow{F} & \{0,1\} \end{array}$$

for functions $\psi : \{x_1, \ldots, x_n\} \to \{0,1\}^n$.

The space of interpretations of $\tilde{F}_n$ is highly symmetric. In particular, let A be the following assignment of truth values:

$$\begin{aligned} e_{i,j} &= x_{i,j} \text{ for each } e_{i,j}, \\ t_{i,j,k} &= y_{i,j,k} \text{ for each } t_{i,j,k}, \\ q_{(a,b),(c,d)} &= z_{(a,b),(c,d)} \text{ for each } q_{(a,b),(c,d)}, \end{aligned}$$

where each $x_{i,j}, y_{i,j,k}, z_{(a,b),(c,d)}$ is either `TRUE` or `FALSE`. For $\sigma \in S_n$, let $\sigma(A)$ be the following assignment of truth values:

$$\begin{aligned} e_{\sigma(i),\sigma(j)} &= x_{\sigma(i),\sigma(j)} \text{ for each } e_{\sigma(i),\sigma(j)}, \\ t_{\sigma(i),\sigma(j),\sigma(k)} &= y_{\sigma(i),\sigma(j),\sigma(k)} \text{ for each } t_{\sigma(i),\sigma(j),\sigma(k)}, \\ q_{(\sigma(a),\sigma(b)),(\sigma(c),\sigma(d))} &= z_{(\sigma(a),\sigma(b)),(\sigma(c),\sigma(d))} \text{ for each } q_{(\sigma(a),\sigma(b)),(\sigma(c),\sigma(d))}. \end{aligned}$$

Then, $\tilde{F}_n(A) = \tilde{F}_n(\sigma(A))$ for all $\sigma \in S_n$. These symmetry transformations correspond to the S_n permutation symmetry on the graph's n vertices and they partition the space of interpretations of $\tilde{F}_n$ into distinct orbits. In principle, there may be additional symmetries of the CNF encoding other than the transformations described above; however, we restrict our attention to these transformations since they are the only ones we are aware of.

Without further modifications to the CNF encoding of our problem, the presence of this symmetry greatly reduces the performance of a SAT solver because without additional constraints, the SAT solver will spend a substantial amount of time exploring isomorphic parts of the search space. To determine the satisfiability of $\tilde{F}_n$, it suffices to consider a single representative interpretation from each orbit. To prevent the SAT solver from considering multiple interpretations from the same orbit, we can add additional **symmetry-breaking clauses** to the CNF encoding. Our method follows [1].

An irredundant generating set S of a group G is a set of group elements such that S generates G and no proper subset of S generates G. For CNF encodings with large symmetry groups, it is often advantageous

to add symmetry-breaking clauses that break the symmetries corresponding to an irredundant generating set instead of attempting to break every symmetry in the group. There are several reasons for this. The first is that the number of symmetries is often extremely large, so attempting to break all of them by adding symmetry-breaking clauses would add such a large number of new clauses to the CNF encoding that all potential performance gains from symmetry breaking would be lost. Furthermore, a theorem from elementary group theory guarantees that the size of an irredundant generating set is at most logarithmic in the size of the group, so as long as the transformations corresponding to these generators are not extremely complicated, the symmetries corresponding to the irredundant generating set can be broken with a reasonably small number of clauses. The disadvantage of this approach is that not all symmetries are broken.

For the symmetries of the formula that we are examining, it is sufficient to consider symmetries of the form $\sigma = (ab)(cd)(ef)...$, where each letter is a variable in the CNF encoding. First, we consider a symmetry of the form (ab). To break such a symmetry, we introduce a clause of the form $\bar{a} \vee b$. To see how this clause works, observe the following. Suppose (ab) is a symmetry of the CNF encoding. If $a = 0, b = 1$ is a satisfying assignment and (ab) is a symmetry of the CNF encoding, then there is a symmetric equivalent satisfying assignment with $a = 1, b = 0$ with no other variables changed. The addition of the clause $\bar{a} \vee b$ ensures that only the first assignment is allowed.

To break a symmetry of the form $\sigma = (ab)(cd)(ef)...$, we proceed as follows. First, choose an ordering of the variables present in the cycle decomposition of the transformation. Then, for each individual cycle, sort the variables in the cycle according to this ordering, and after doing so for each cycle, sort cycles by their first variable. These steps are necessary to break the symmetries in a consistent manner that preserves satisfiability. To break the symmetry σ, we first add the symmetry breaking clause $\bar{a} \vee b$. Then, we add $(a = b) \implies (\bar{c} \vee d)$, then $((a = b)(c = d)) \implies (\bar{e} \vee f)$, etc. This construction can be efficiently implemented by introducing one additional auxiliary variable per cycle that indicates the equality of all variables in the cycle. For example, a clause with these new auxiliary variables would be of the form $(\bar{x}_{a=b} \vee \bar{x}_{c=d} \vee \bar{e} \vee f)$. By an analysis similar to that described before, one can see that the symmetry-breaking clauses introduced for the symmetry σ allow only one satisfying assignment from the set of satisfying assignments that are related by σ.

An irredundant set of generators for S_n is

$$S = \{(1, 2), (2, 3), \ldots, (n - 1, n)\}.$$

Any element $\sigma \in S_n$ induces the following transformation ϕ_σ in the CNF encoding

$$\begin{aligned} e_{i,j} &\leftrightarrow e_{\sigma(i),\sigma(j)} \\ t_{x,y,z} &\leftrightarrow t_{\sigma(x),\sigma(y),\sigma(z)} \\ q_{(a,b),(c,d)} &\leftrightarrow q_{(\sigma(a),\sigma(b)),(\sigma(c),\sigma(d))}, \end{aligned}$$

so each ϕ_σ is a product of disjoint transpositions of CNF variables of the form

$$\begin{aligned} \phi_\sigma = &(e_{i_1,j_1}, e_{\sigma(i_1),\sigma(j_1)}) \cdots (e_{i_{m_e},j_{m_e}}, e_{\sigma(i_{m_e}),\sigma(j_{m_e})}) \cdot \\ &(t_{x_1,y_1,z_1} t_{\sigma(x_1),\sigma(y_1'),\sigma(z_1')}) \cdots (t_{x_{m_t},y_{m_t},z_{m_t}} t_{\sigma(x_{m_t}),\sigma(y_{m_t}),\sigma(z_{m_t})}) \cdot \\ &(q_{(a_1,b_1),(c_1,d_1)}, q_{(\sigma(a_1),\sigma(b_1)),(\sigma(c_1),\sigma(d_1))}) \cdots (q_{(a_{m_q},b_{m_q}),(c_{m_q},d_{m_q})} q_{(\sigma(a_{m_q}),\sigma(b_{m_q})),(\sigma(c_{m_q}),\sigma(d_{m_q}))}), \end{aligned}$$

where m_e, m_t, and m_q are the number of edge variable transpositions, triangle variable transpositions, and quadrilateral variable transpositions in ϕ_σ, respectively. For each such transposition, we introduce symmetry breaking clauses using the methodology described above.

4. Substructures of the Conway 99-Graph

A **regular graph** is a graph such that each vertex has the same degree. A regular graph with n vertices, each of degree k, is a **strongly regular graph** with parameters (v, k, λ, μ) if there exist integers λ and μ such that every two adjacent vertices have λ common neighbors and every two non-adjacent vertices have μ common neighbors. It is known that a putative Conway 99-graph must be a strongly regular graph of parameters $(99, 14, 1, 2)$.

Next, we determine several substructures of the Conway 99-graph. Let v_1 be an arbitrary vertex and choose any neighboring vertex, denoting it as v_2. Since v_1 and v_2 are adjacent, they share precisely one mutual neighbor, which we denote as v_3. This formation produces a triangle consisting of vertices v_1, v_2, and v_3. Furthermore, neither v_2 nor v_3 can have any other neighbours in the neighborhood $N(v_0)$ of v_0.

Repeating this process, we can sequentially introduce and pair the vertices $v_4 - v_5$, $v_6 - v_7$, $v_8 - v_9$, $v_{10} - v_{11}$, $v_{12} - v_{13}$, and $v_{14} - v_{15}$ in $N(v_0)$.

Any two vertices v_i, v_j in $N(v_0)$ that are not part of a triangle must have two mutual neighbors. One of these neighbors must be v_0, but the other vertex must be a new vertex, which we denote as $v_{i,j}$, where the ordering of the subscripted indices is unimportant, that is not in $N(v_0)$. Suppose $v_{i,j}$ is adjacent to more than two vertices in $N(v_0)$. Then, this would imply that $v_{i,j}$ has more than two mutual neighbors with v_0, which would violate the requirement that $v_{i,j}$ and v_0 have exactly two mutual neighbors. Therefore, each pair of vertices in $N(v_0)$ that are not adjacent must have a unique mutual neighbor outside $N(v_0)$. For each vertex v_i in $N(v_0)$, there exist twelve other vertices in $N(v_0)$ that v_i is not adjacent to, so upon adding a vertex $v_{i,j}$ for each pair of non-adjacent vertices in $N(v_0)$, we find that all 14 of the neighbors for each vertex in $N(v_0)$ have been determined. This allows us to determine all 14 neighbors of vertices $v_1, v_2, \ldots, v_{15}$. Doing so allows us to determine all edge variables $e_{i,j}$ such that at least one of i, j is in $\{1, 2, \ldots, 15\}$. This information can then be used to simplify the CNF encoding of our problem.

5. Next Steps

Here, we outline several next steps that could help refine our approach and leverage SAT solvers to tackle Conway's 99-graph problem.

(1) **Incorporating Regularity Constraints:** Our current CNF encoding can be enhanced by adding regularity constraints, i.e., constraints that enforce the condition that every vertex of a Conway 99-graph must have degree 14. Such constraints can potentially prune large portions of the search space, thus improving the efficiency of the SAT solver.

(2) **Refining the At-Most-One Constraint:** The constraint for uniqueness of triangles between adjacent vertices and quadrilaterals between non-adjacent vertices is an at-most-one constraint. In our encoding, we implemented a naive version of this at-most-one constraint; however, there exist more efficient encodings of the at-most-one constraint which can substantially reduce the complexity of the encoding and increase SAT solver performance.

(3) **Fixing Edges to Simplify CNF:** In principle, there exists some large number of edges such that, if fixed, the resulting simplified CNF encoding can be determined as satisfiable or unsatisfiable by a SAT solver in a reasonable amount of time. If the number of edges is sufficiently small, it is possible that determining the satisfiability of the entire CNF encoding is computationally feasible by reducing the problem to the task of determining the satisfiability of a large number of SAT instances that result from fixing many edges. Much is known about the spectra of strongly regular graphs, which places restrictions on the spectra of induced subgraphs of strongly regular graphs. These restrictions on spectra of subgraphs can be used to rule out configurations that must be checked by the SAT solver.

6. Acknowledgements

First and foremost, I would like to sincerely thank my mentor Michael Klug for his insight, help, and guidance throughout the summer. I am also grateful to Peter May for organizing the 2023 University of Chicago REU during which this paper was written.

References

[1] Fadi A. Aloul et al. "Solving Difficult SAT Instances in the Presence of Symmetry". In: *Proceedings of the 39th Annual Design Automation Conference*. DAC '02. New York, NY, USA: Association for Computing Machinery, 2002, pp. 731–736. ISBN: 1581134614. DOI: 10.1145/513918.514102. URL: https://doi.org/10.1145/513918.514102.

[2] Alexander Nadel. "Solving Huge Instances with Intel® SAT Solver". In: *26th International Conference on Theory and Applications of Satisfiability Testing*. 2023.