A Census of Vertex-Transitive Graphs

by Gary Amende[1]
Beloit College
700 College St
Beloit, WI 53511
amendeg@stu.beloit.edu

Abstract. We provide the census of all vertex-transitive graphs with no more than 12 vertices. While the computer generated the vertex-transitive graphs, we prove some properties about the transitivity of two graph products that help illustrate exactly which graphs were generated. Both the Cartesian product and cardinal product of any two vertex-transitive graphs is vertex-transitive. The cardinal product of two dart-transitive graphs is dart-transitive, while the cardinal product of an edge-transitive graph with a dart-transitive graph is edge-transitive.

Introduction.

An undirected graph G consists of a finite set of vertices V(G) and a finite set of unordered pairs (edges) $E(G) = \{(a,b) \mid a,b \in V(G)\}$. A *dart* is an edge together with a vertex that the dart points to. Clearly, each edge (a,b) contains two darts: the dart (a,b), coming from a going into b, and the dart (b,a), coming from b going into a. The set of all darts is denoted D(G). The graphs that we are interested in contain no multiple edges (no two elements of E(G) are equal), contain no loops (for any edge $(a,b) \in E(G)$, $a \neq b$) and are connected (for any $a \in V(G)$ there exists $(a,x) \in E(G)$ for some $x \in V(G)$).

The number of vertices a graph contains is denoted p or |V(G)|. The symmetric group S_p is the group of all permutations of these p vertices. The elements of S_p that preserve adjacency in the graph G, where |V(G)| = p, form a subgroup Γ called the *group of the graph* G. Formally, for each $\sigma \in S_p$, define the function ϕ_σ: $E(G) \to V(G) \times V(G)$ as $\phi_\sigma(a,b) = (\sigma a, \sigma b)$ for $(a,b) \in E(G)$. Now define $\Gamma(G) = \{\sigma \in S_p \mid \phi_\sigma[E(G)] = E(G)\}$. Note that if $\phi_\sigma[E(G)] = E(G)$, then $\phi_\sigma[E(\overline{G})] = E(\overline{G})$ and $\Gamma(G) = \Gamma(\overline{G})$ where $\overline{G}$ is the complement of G. The complement of G has the same vertices as G, but the edges are defined as $E(\overline{G}) = \{(a,b) \mid (a,b) \notin E(G)\}$.

If for any $a,b \in V(G)$ there exists $\sigma \in \Gamma(G)$ such that $\sigma a = b$ then Γ is *transitive* and G is *vertex-transitive*. Note that since $\Gamma(G) = \Gamma(\overline{G})$, the complement of a vertex-transitive graph is also vertex-transitive. If for any (a,b), (c,d) $\in E(G)$ there exists $\sigma \in \Gamma(G)$ such that $\phi_\sigma(a,b) = (c,d)$, then G is *edge-transitive*. Since the edge (c,d) is the same as the edge (d,c), either $\sigma a = c$ and $\sigma b = d$ or $\sigma a = d$ and $\sigma b = c$. Finally, if for any two darts (a,b), (c,d) $\in D(G)$ there exists $\sigma \in \Gamma(G)$ such that $\phi_\sigma(a,b) = (c,d)$, then G is *dart-transitive*. Since darts are ordered pairs, $\sigma a = c$ and $\sigma b = d$.

Clearly, vertex-transitivity does not imply edge-transitivity and edge-transitivity does not imply vertex-transitivity. However, dart-transitivity does imply vertex- and edge-transitivity. Take any dart $(a,b) \in D(G)$. From dart-transitivity, there exists $\sigma \in \Gamma(G)$ such that $\phi_\sigma(a,b) = (\sigma a, \sigma b) = (a,x)$ for any $(a,x) \in E(G)$ where $x \neq a$. Also, there exists $\sigma \in \Gamma(G)$ such that $\phi_\sigma(a,b) = (b,a)$. Therefore, for any $b,x \in V(G)$ there exists $\sigma \in \Gamma(G)$ such that $x = \sigma b$. Finally, dart-

---

[1]Research sponsored by NSF REU at Northern Arizona University, Summer 1994, under supervising professor Steve Wilson.

transitivity implies edge-transitivity.

In this paper, we are interested in collecting together as many vertex-transitive graphs as possible. We explore three avenues of thought. First, we look at existing families of graphs and prove transitivity properties about families of graphs. Second, we study products of transitive graphs and try to prove transitivity properties about the resulting products. Last, we implement a computer program that generates vertex-transitive graphs for small p. This last process helps fill in the gaps in our knowledge of the first two. However, due to size of the problem, generating graphs is practical only for small p.

Families of Vertex-Transitive Graphs.

Now that the definitions of transitivity have been made clear, which graphs exhibit these properties? The complete graphs K_n are dart-transitive, since $\Gamma(G) = \Gamma(\overline{G})$ and $\overline{K}_n$ consists of n vertices without any edges. It is clear that $\overline{K}_n$ has automorphism group S_n and so K_n has automorphism group S_n. Now, for any two darts (a,b), (c,d) $\in K_n$, there exists two transpositions (b d), (a c) $\in S_n$ such that (b d)(a c)(a,b) = (b d)(c,b) = (c,d). Therefore, the complete graphs K_n are dart-transitive and hence vertex- and edge-transitive.

The n-cycles C_n are dart-transitive. Note that $\Gamma(C_n) = D_n$. The permutations of D_n are rotations and reflections of the regular n-gon. We know that any edge can be mapped into any other edge by some rotation of D_n. By applying a reflection first and then rotations, one can see that any dart can be mapped into any other dart. As a result, C_n is dart-transitive.

The partition graphs, $K_{n,n}$, $K_{n,n,n}$, etc. are dart-transitive. Clearly, any dart coming from one partition can be mapped into any other dart coming from the same partition. Since both vertices the dart points from are in the same partition, they are adjacent to the same vertices in the other partitions. Now any partition can be mapped into any other partition, since each vertex in both partitions is adjacent to the same vertices in the other partitions. Any dart coming from a vertex in one partition can be mapped into a dart coming from a vertex in another partition. Therefore any dart can be mapped into any other dart.

Another family of graphs that are vertex-transitive is the n-cycles with edges added in a deterministic way. For lack of a better term, we have been merely referring to these as *Hamiltonian graphs* since they contain a *Hamiltonian cycle*. We denote these $HG(p;x_1, x_2, ..., x_n)$ where $1 < x_i \le \lfloor p/2 \rfloor$ for $1 \le i \le n$, $x_i \ne x_j$ for all $i \ne j$ and define them as follows:

$$V(HG(p;x_1, x_2, ..., x_n)) = \{i \mid 0 \le i \le p-1\}$$
$$E(HG(p;x_1, x_2, ..., x_n)) = \{(a,(a+1) \bmod p) \mid 0 \le a \le p-1\} \cup \{(a,b) \mid |a-b| \equiv x_i \pmod p \text{ for some } 1 \le i \le n\}$$

We write $HG(p;x_i)$ for short. Now, the permutation $\sigma = (0\ 1\ 2\ ...\ p-1) \in \Gamma(HG(p;x_i))$. Clearly, $\sigma a = a+1 \pmod p$. For any $(a,b) \in E(HG(p;x_i))$, $\phi_\sigma(a,b) = (\sigma a, \sigma b) = (a+1 \pmod p, b+1 \pmod p)$. Now either $b = a+1 \pmod p$ or $|a-b| \equiv x_i \pmod p$ for some $1 \le i \le n$. If $b = a+1$, then $\phi_\sigma(a,b) = (a+1 \pmod p, a+2 \pmod p) \in E(HG(p;x_i))$. If $|a-b| \equiv x_i \pmod p$ for some $1 \le i \le n$, then $(a+1-b-1) = (a-b)$. Therefore, $\phi_\sigma(a,b) \in E(HG(p;x_i))$ and $\sigma \in \Gamma(HG(p;x_i))$. Since all the powers of this permutation are also in the group, these Hamiltonian graphs are vertex-transitive.

Graph Constructions.

While numerous graph products exist, most of them combine the graphs in such a way that yield few graphs with inherent transitivity properties. However two of them, the Cartesian product and the cardinal product, do exhibit these properties.

Definition 1. The *cardinal* product, as introduced by Čulik [1], of two graphs G_1 and G_2, written as $G_1 \otimes G_2$ is defined as follows.

$V(G_1 \otimes G_2) = \{(a,b) \mid a \in V(G_1), b \in V(G_2)\}$

$E(G_1 \otimes G_2) = \{(a,b), (c,d) \mid (a,c) \in E(G_1) \text{ and } (b,d) \in E(G_2)\}$

This product, also called the Kronecker product, was also introduced independently by Weichsel [3] and named for the Kronecker product of matrices.

Definition 2. The *Cartesian* product, as introduced by Sabidussi [2], of two graphs G_1 and G_2, written as $G_1 \times G_2$ is defined as follows.

$V(G_1 \times G_2) = \{(a,b) \mid a \in V(G_1), b \in V(G_2)\}$

$E(G_1 \times G_2) = \{(a,b), (c,d) \mid (a = c \text{ and } (b,d) \in E(G_2)) \text{ or } (b = d \text{ and } (a,c) \in E(G_1))\}$

The following theorems establish results that guarantee the construction of transitive graphs. First, we must show that each element in the group of a graph is imbedded within the group of the product of the graph with another graph.

Lemma 1. (Cardinal product) For any two graphs G_1 and G_2, $\sigma \in \Gamma(G_1)$ and $\mu \in \Gamma(G_2)$ Define

$\sigma': V(G_1 \otimes G_2) \to V(G_1 \otimes G_2)$ by $\sigma'(a,b) = (\sigma a,b)$ for any $(a,b) \in V(G_1 \otimes G_2)$ and

$\mu': V(G_1 \otimes G_2) \to V(G_1 \otimes G_2)$ by $\mu'(a,b) = (a,\mu b)$ for any $(a,b) \in V(G_1 \otimes G_2)$.

Then $\sigma', \mu' \in \Gamma(G_1 \otimes G_2)$.

Proof: If $\phi_\sigma[E(G_1 \otimes G_2)] = E(G_1 \otimes G_2)$, then $\sigma' \in \Gamma(G_1 \otimes G_2)$.
If $((a,b), (c,d)) \in E(G_1 \otimes G_2)$, then $(a,c) \in E(G_1)$ and $(b,d) \in E(G_2)$.
Since $\sigma \in \Gamma(G_1)$, $\phi_\sigma(a,c) = (\sigma a, \sigma c) \in E(G_1)$.
Therefore $((\sigma a,b), (\sigma c,d)) \in E(G_1 \otimes G_2)$ and $\sigma' \in \Gamma(G_1 \otimes G_2)$.
The proof that $\mu' \in \Gamma(G_1 \otimes G_2)$ is similar.

Theorem 1. If G_1 and G_2 are vertex-transitive, then $G_1 \otimes G_2$ is vertex-transitive.

Proof: Since G_1 is vertex-transitive, for any a, $a' \in V(G_1)$ there exists $\sigma \in \Gamma(G_1)$ such that $a' = \sigma a$. Likewise, since G_2 is vertex-transitive, for any b, $b' \in V(G_2)$ there exists $\mu \in \Gamma(G_1)$ such that $b' = \mu b$.
By Lemma 1, $\sigma', \mu' \in \Gamma(G_1 \otimes G_2)$. Since $\Gamma(G_1 \otimes G_2)$ is a group, $\sigma'\mu' \in \Gamma(G_1 \otimes G_2)$.
For any $(a,b), (a',b') \in V(G_1 \otimes G_2)$, $\sigma'\mu'(a,b) = \sigma'(a,\mu b) = (\sigma a,\mu b) = (a',b')$.
Therefore, $G_1 \otimes G_2$ is vertex-transitive.

Theorem 2. If G_1 is edge-transitive and G_2 is dart-transitive, then $G_1 \otimes G_2$ is edge-transitive.

Proof: For any two edges (a,c), $(a',c') \in E(G_1)$ there exists $\mu \in \Gamma(G_1)$ such that $(a',c') = \phi_\mu(a,c)$. Either $a' = \mu a$ and $c' = \mu c$ or $a' = \mu c$ and $c' = \mu a$.
For any two darts $(b,d), (b',d') \in D(G_2)$ there exists $\sigma \in \Gamma(G_2)$ such that

$(b',d') = \phi_\sigma(b,d)$. Since (b',d') is a dart, (d',b') is a dart and there exists $\tau \in \Gamma(G_2)$ such that $(d',b') = \phi_\tau(b,d)$.
By Lemma 1, μ', σ', $\tau' \in \Gamma(G_1 \otimes G_2)$.

For any two edges $((a,b), (c,d)), ((a',b'), (c',d')) \in E(G_1 \otimes G_2)$,
there are two possible mappings by μ:
(i) $a' = \mu a$ and $c' = \mu c$.
$\phi_{\mu'\sigma'}((a,b), (c,d)) = (\mu'\sigma'(a,b), \mu'\sigma'(c,d)) = (\mu'(a,\sigma b), \mu'(c,\sigma d)) =$
$((\mu a,\sigma b), (\mu c,\sigma d)) = ((a',b'), (c',d'))$.
(ii) $a' = \mu c$ and $c' = \mu a$.
$\phi_{\mu'\tau'}((a,b), (c,d)) = (\mu'\tau'(a,b), \mu'\tau'(c,d)) = (\mu'(a,\tau b), \mu'(c,\tau d)) =$
$((\mu a,\tau b), (\mu c,\tau d)) = ((c',d'), (a',b')) = ((a',b'), (c',d'))$.
Therefore, $G_1 \otimes G_2$ is edge-transitive.

Theorem 3. If G_1 and G_2 are dart-transitive, then $G_1 \otimes G_2$ is dart-transitive.
Proof: For any two darts (a,c), $(a',c') \in D(G_1)$, there exists $\mu \in \Gamma(G_1)$ such that $(a',c') = \phi_\mu(a,c)$. For any two darts (b,d), $(b',d') \in D(G_2)$, there exists $\sigma \in \Gamma(G_2)$ such that $(b',d') = \phi_\sigma(b,d)$.
By Lemma 1, μ', $\sigma' \in \Gamma(G_1 \otimes G_2)$.

For any two darts $((a,b), (c,d)), ((a',b'), (c',d')) \in D(G_1 \otimes G_2)$,
$\phi_{\mu'\sigma'}((a,b), (c,d)) = (\mu'\sigma'(a,b), \mu'\sigma'(c,d)) = (\mu'(a,\sigma b), \mu'(c,\sigma d)) =$
$((\mu a,\sigma b), (\mu c,\sigma d)) = ((a',b'), (c',d'))$.
Therefore, $G_1 \otimes G_2$ is dart-transitive.

Now we need to show similar results for the Cartesian product of graphs. Unfortunately, further conditions must be placed upon the graphs to guarantee edge- and dart-transitivity. The Cartesian product of an edge-transitive graph with itself does produce another edge-transitive graph. However, this condition is sufficient but not necessary. The study of primal graphs may prove helpful in proving further results.

Lemma 2. (Cartesian product) For any two graphs G_1 and G_2, $\sigma \in \Gamma(G_1)$ and $\mu \in \Gamma(G_2)$, define
$\sigma': V(G_1 \times G_2) \to V(G_1 \times G_2)$ by $\sigma'(a,b) = (\sigma a,b)$ for any $(a,b) \in V(G_1 \times G_2)$ and
$\mu': V(G_1 \times G_2) \to V(G_1 \times G_2)$ by $\mu'(a,b) = (a,\mu b)$ for any $(a,b) \in V(G_1 \times G_2)$.
Then σ', $\mu' \in \Gamma(G_1 \times G_2)$.
Proof: If $\phi_{\sigma'}[E(G_1 \times G_2)] = E(G_1 \times G_2)$, then $\sigma' \in \Gamma(G_1 \times G_2)$.
If $((a,b), (c,d)) \in E(G_1 \times G_2)$, then either
(i) $a = c$ and $(b,d) \in E(G_2)$.
$\phi_{\sigma'}((a,b), (a,d)) = (\sigma'(a,b), \sigma'(a,d)) = ((\sigma a,b), (\sigma a,d))$.
Since $(b,d) \in E(G_2)$, we have $((\sigma a,b), (\sigma a,d)) \in E(G_1 \times G_2)$ and $\sigma' \in \Gamma(G_1 \times G_2)$.

(ii) $b = d$ and $(a,c) \in E(G_1)$.
$\phi_{\sigma'}((a,b), (c,b)) = (\sigma'(a,b), \sigma'(c,b)) = ((\sigma a,b), (\sigma c,b))$.
Since $(a,c) \in E(G_1)$ and $\sigma \in \Gamma(G_1)$, $(\sigma a, \sigma c,d) \in E(G_1)$ and

$((\sigma a,b), (\sigma c,b)) \in E(G_1 \times G_2)$ and $\sigma' \in \Gamma(G_1 \times G_2)$.
The proof that $\mu' \in \Gamma(G_1 \times G_2)$ is similar.

Theorem 4. If G_1 and G_2 are vertex-transitive, then $G_1 \times G_2$ is vertex-transitive.

Proof: Since G_1 is vertex-transitive, for any a, $a' \in V(G_1)$ there exists $\sigma \in \Gamma(G_1)$ such that $a' = \sigma a$. Likewise, since G_2 is vertex-transitive, for any b, $b' \in V(G_2)$ there exists $\mu \in \Gamma(G_1)$ such that $b' = \mu b$.
By Lemma 1, $\sigma', \mu' \in \Gamma(G_1 \times G_2)$. Since $\Gamma(G_1 \times G_2)$ is a group, $\sigma'\mu' \in \Gamma(G_1 \times G_2)$.
For any (a,b), $(a',b') \in V(G_1 \times G_2)$, $\sigma'\mu'(a,b) = \sigma'(a,\mu b) = (\sigma a,\mu b) = (a',b')$.
Therefore, $G_1 \times G_2$ is vertex-transitive.

Now that we know that products of vertex-transitive graphs are vertex-transitive, we can construct new vertex-transitive graphs from our stock of vertex-transitive graphs. For instance, we know that K_2 and K_4 are vertex-transitive, so both $K_2 \times K_4$ and $K_2 \otimes K_4$ are vertex-transitive. In fact, we know that $K_2 \otimes K_4$ is dart-transitive. Both have 8 vertices, but $K_2 \times K_4$ is regular of degree 4, while $K_2 \otimes K_4$ is regular of degree 3. Also $K_2 \times C_4$ and $K_2 \otimes C_4$ are vertex-transitive. However, since K_2 and C_4 are both bipartite, $K_2 \otimes C_4$ is disconnected [5], while $K_2 \times C_4$ is isomorphic to $K_2 \otimes K_4$. For graphs with 8 vertices, we have C_8, K_8, $K_2 \times K_4$, $K_2 \otimes K_4$, $K_{2,2,2,2}$, $K_{4,4}$. Since the complement of a vertex transitive is also vertex-transitive, we also have the complements of those six graphs, but the complements of K_8, $K_{2,2,2,2}$, and $K_{4,4}$ are disconnected and the complements of $K_2 \times K_4$ and $K_2 \otimes K_4$ are isomorphic to two of the first six graphs. The Hamiltonian graphs yield three more non-isomorphic graphs: HG(8;4), HG(8;2), and HG(8;3,4). HG(8;3,4) is isomorphic to the complement of $2C_4$, two copies of C_4. These three bring the total to 10 vertex-transitive graphs with 8 vertices. For graphs with 8 vertices, there are only 10 vertex-transitive graphs.

Using the first two methods of finding vertex-transitive graphs, we find all the vertex-transitive graphs up to 10 vertices. However, when searching for vertex-transitive graphs with 10 vertices, we leave out two: the Petersen graph and its complement. The methods work for graphs with 11 vertices, but for graphs with 12 vertices, we find only 48 of the 66. Two of them are the icosahedron and its complement; the computer program found the other 16.

The Computer Program.

The problem of generating all the vertex-transitive graphs and only the vertex-transitive graphs is rather difficult. We restrict each run of the program to finding all the vertex-transitive graphs with p vertices and degree r. Then, we run the program multiple times on the graphs we want to generate.

We could generate all the graphs with p vertices and then check each graph for vertex-transitivity. That approach is needlessly wasteful. Since the graphs we are interested in are connected and regular, we generate only those graphs. Then, since we want vertex-transitive graphs, we utilize a *complete code* for graphs described by Foster [4]:

> Select any one vertex as a starting point, and designate this vertex as of grade zero. Then the vertices at a distance 1 from this vertex are designated as of grade 1, at a distance of 2 as of grade 2, and, in general, vertices at a distance r as of grade r. Then classify each vertex as of type "abc" where a + b + c = 3, a being the number of edges connecting this vertex to vertices of the next lower grade, b being the number of edges connecting it to vertices of the same grade and c being the number of edges connecting it to vertices of the next higher grade.

Thus the initially selected vertex is of type 003. Then, in tabular form, the complete code shows the number of vertices of each grade and of each type.

Foster was interested in only trivalent graphs, thus a + b + c = 3. Since we are interested in regular graphs in general, a + b + c = r, where r is the degree and the initially select vertex is of type 00r. For vertex-transitive graphs, the complete code generated by choosing any vertex initially will be the same as choosing any other vertex. This observations helps in "weeding out" unwanted graphs. During the process of generating graphs, at convenient intervals, the algorithm checks the partial complete codes for each of the vertices. If any code for a vertex is different from the other codes for the other vertices, then the partial graph is not vertex-transitive and it is thrown away.

Algorithm 1.

1. The user inputs p and r.
2. Push an empty graph with p vertices labeled 0, 1, ..., p−1 onto a stack.
3. Pop a graph off stack.
4. Search the vertices in numerical order to find the first vertex that doesn't have degree r.
5. For the vertex found in step 4, create a list of vertices that can be adjacent to it. If two vertices are adjacent to the same vertices, then those two are virtually identical for the purpose of generating graphs.
6. The vertex found in step 4 needs r less the degree of the vertex edges added to the graph. The list generated in step 5 contains all the possible vertices the vertex can be adjacent to. Each vertex on the list has a different adjacency list from any other vertex on the list. Using the list of vertices not currently adjacent to the vertex, follow the procedure below on all the possible ways to add the needed edges to the graph. For example, let there be two different types of vertices in the list and let there be two vertices of the first type and one of the second. Then, if a vertex needs two more edges adjacent to it to be degree r, then the two edges can be added to the graph in two different ways. An edge can be added between each of the first type or an edge can be added between each of the two types. Notice that it doesn't matter which of the first type an edge is added to. Within each type of vertex on the list, each of the vertices are adjacent to the same vertices.
    a. Copy the graph.
    b. Add the edges to the graph.
    c. If the graph is regular (i.e. all the edges have been added), then check the complete code. If the complete code is the same for every vertex, then output the graph. Continue with step e. If the graph is not regular, then proceed to step d.
    d. If the graph is connected, then check the complete code. At this point, a complete code for each vertex does not exist. The code generated by choosing a particular vertex as the start may not be complete. However, the first few columns of the code may be complete and this partial code can be compared to the code of the first graph. If the partial complete code is the same for every vertex, then push the graph onto the stack. Go to step e.
    e. Copy the original graph back into the current graph. Finish the iterations.

7. If the stack is empty, end the algorithm. Otherwise, continue with step 3.

The graphs generated by this algorithm have the same complete code for every vertex. While this condition is necessary, it is not sufficient. The algorithm also outputs too many isomorphic graphs. To guarantee that graphs are vertex-transitive, the output from this algorithm is filtered through Algorithm 2 which checks for vertex-transitivity and isomorphism. The output of this second algorithm is all non-isomorphic vertex-transitive graphs of p vertices and degree r.

Algorithm 2.

1. Get a graph to check.
2. Check for vertex-transitivity.
    a. Select the first vertex. Every vertex has to be able to be mapped into the first vertex. If all the vertices can be mapped into the first vertex, then any vertex can be mapped into any other vertex.
    b. Partition the vertices by grade with the first vertex as the starting point.
    c. For one vertex at time, partition the other vertices by grade. Now any vertex in grade 1 of this vertex has to map into a vertex in grade 1 of the first vertex. In general, any vertex in grade r has to map into a vertex in grade r of the first vertex.
    d. Iterate through all the possible permutations of the vertices using the grade information. For each permutation, either it is an element of the group of that graph or it is not. If the set of edges is preserved, then the permutation is an element of the group. If it is an element of the group, the current vertex can be mapped into the first vertex. If no permutation is found, this graph is not vertex transitive.
    e. If there exist permutations such that all vertices can be mapped into the first vertex, then the graph is vertex-transitive.
3. If the graph is vertex-transitive, then test the graph for isomorphism to previously checked graphs.
    a. Compare the graph to each previously checked graph one at a time. Call the graph the first graph and the previously checked graph the second graph.
    b. If the graphs are isomorphic, then we can choose any two vertices, one in each graph, and there exists a permutation that maps the first graph into the second graph. Since both graphs are vertex-transitive, any vertex in a graph can be mapped into any other vertex in that graph and any vertex in one graph can be mapped into any vertex in the other graph.
    c. Select the first vertex of each graph.
    d. For each graph, partition the vertices by grade.
    e. Iterate through all the possible permutations of the vertices of the second graph using the grade information. If the set of edges in

the second graph is preserved, then the permutation is an element of the group. If it is an element of the group, the two graphs are isomorphic. If no permutation is found, these graphs are non-isomorphic.

f. If the graphs are non-isomorphic, add the first graph to the list of previously checked graphs.. If the graphs are isomorphic, continue with the next graph to be checked.

4. If another graph exists, continue with step 1.
5. Output the list of non-isomorphic vertex-transitive graphs.

Having algorithms may seem unnecessary and, indeed, they may be unnecessary. Further research is being done to eliminate the use of Foster's complete code. Currently, Foster's complete code test is very fast to implement and the codes can be generated and maintained very rapidly. The code provides a simple criterion for a graph to be vertex-transitive that eliminates many graphs. The older algorithm for the vertex-transitivity and isomorphic checker was very slow. Hence we utilized Foster's code to generate fewer graphs to check. The newer algorithm for the checker is very fast. We believe a similar improvement upon the first algorithm will render the use of Foster's code unnecessary.

The Census.

The graphs are listed in tabular order. The headings "p" and "r" are the number of vertices and the degree respectively. The decimal place in r is used for numbering purposes. The number 3.2 means the second graph with degree 3. "Name" is the primary name of the graph. The "Alternative Name" is self-explanatory. The only exception is that Hamiltonian graphs are put under the alternative name. The c() that surround some graph names indicate the complement of that graph inside the parentheses. The notation $G_1[G_2]$ designates the composition of two graphs (see Sabidussi [**6**] or Harary [7]). An | between the number in the Hamiltonian graphs designates "or". HG(11;3|4) designates HG(11;3) or HG(11;4). The two graphs are isomorphic. The HG(11;*,*,*,*) means any four numbers produce the same graph.

If the entry under the edge-transitive column is not a "Y", then it designates the orbits of edges in the graph. An entry of (5,3-10) means that one orbit of 5 edges and 3 orbits of 10 edges exist in the graph. For all the graphs in this table, if a graph is edge-transitive, it is also dart-transitive.

The group order is simply the number of elements in the automorphism group of the graph. The group description is a brief summary of some groups that are subgroups of the automorphism group. More study needs to be done to determine the exact structure of the automorphism group.

Each of the graphs were entered into the program "Groups and Graphs" [**8**], where they were checked for edge-transitivity, dart-transitivity, and group order.

Table 1.

p	r	Name	Alternative Name	Edge-transitive	|G|	Group Description
1	0	K_1		Y	1	{e}
2	1	K_2		Y	2	Z_2
3	2	K_3		Y	6	D_3
4	2	C_4	$K_{2,2}$	Y	8	D_4
4	3	K_4	HG(4;2)	Y	4!	S_4
5	2	C_5	$\overline{C}_5$	Y	10	D_5
5	4	K_5	HG(5;2)	Y	5!	S_5
6	2	C_6	$K_2 \otimes K_3$	Y	12	D_6
6	3.1	$K_2 \times K_3$	$\overline{C}_6$	(3,6)	12	D_6
6	3.2	$K_{3,3}$	HG(6;3)	Y	72	$2\text{-}D_3, Z_2$
6	4	$K_{2,2,2}$	HG(6;2)	Y	48	$3\text{-}Z_2, S_3$
6	5	K_6	HG(6;2,3)	Y	6!	S_6
7	2	C_7		Y	14	D_7
7	4	$\overline{C}_7$	HG(7;2|3)	(2-7)	14	D_7
7	6	K_7	HG(7;2,3)	Y	7!	S_7
8	2	C_8		Y	16	D_8
8	3.1		HG(8;4)	(4,8)	16	D_8
8	3.2	$K_2 \times K_2 \times K_2$	$K_2 \otimes K_4$	Y	48	$3\text{-}Z_2, S_3$
8	4.1	c(HG(8;3))	HG(8;2)	(2-8)	16	D_8
8	4.2	$K_2 \times K_4$	$c(K_2 \times K_2 \times K_2)$	(4,12)	48	$3\text{-}Z_2, S_3$
8	4.3	$K_{4,4}$	HG(8;3)	Y	1152	$2\text{-}S_4, Z_2$
8	5.1	$\overline{C}_8$	HG(8;2,4)	(4,2-8)	16	D_8
8	5.2	$c(2C_4)$	HG(8;3,4)	(4,16)	128	$2\text{-}D_4, Z_2$
8	6	$K_{2,2,2,2}$	HG(8;2,3)	Y	384	$4\text{-}Z_2, S_4$
8	7	K_8	HG(8;2,3,4)	Y	8!	S_8
9	2	C_9		Y	18	D_9
9	4.1	c(HG(9;3))	HG(9;2|4)	(2-9)	18	D_9
9	4.2	c(HG(9;2|4))	HG(9;3)	(2-9)	18	D_9
9	4.3	$K_3 \times K_3$	$K_3 \otimes K_3$	Y	72	$2\text{-}S_3, Z_2$
9	6.1	HG(9;2,3)	HG(9;3,4)	(3-9)	18	D_9

9	6.2	$K_{3,3,3}$	HG(9;2,4)	Y	1296	4-S_3
9	8	K_9	HG(9;2,3,4)	Y	9!	S_9
10	2	C_{10}		Y	20	D_{10}
10	3.1		HG(10;5)	(5,10)	20	D_{10}
10	3.2	$K_2 \times C_5$		(5,10)	20	D_{10}
10	3.3	Petersen		Y	5!	S_5
10	4.1		HG(10;2)	(2-10)	20	D_{10}
10	4.2	$K_2 \circledcirc K_5$	HG(10;3)	Y	240	S_5, Z_2
10	4.3		HG(10;4)	Y	320	D_{10}, 5-Z_2
10	5.1		HG(10;2,5)	(5,2-10)	20	D_{10}
10	5.2	$K_2 \times K_5$		(20,5)	240	S_5, Z_2
10	5.3	c(HG(10;4))	HG(10;4,5)	(20,5)	320	D_{10}, 5-Z_2
10	5.4	$K_{5,5}$	HG(10;3,5)	Y	28800	2-S_5, Z_2
10	6.1		HG(10;2,4)	(3-10)	20	D_{10}
10	6.2	c($K_2 \times C_5$)	HG(10;3,4|2,3)	(3-10)	20	D_{10}
10	6.3	c(Petersen)		Y	5!	S_5
10	7.1	$\overline{C}_{10}$	HG(10;2,4,5)	(5,3-10)	20	D_{10}
10	7.2	HG(10;3,4,5)	HG(10;2,3,5)	(25,10)	200	2-D_5, Z_2
10	8	$K_{2,2,2,2,2}$	HG(10;2,3,4)	Y	3840	S_5, 5-Z_2
10	9	K_{10}	HG(10;2,3,4,5)	Y	10!	S_{10}
11	2	C_{11}		Y	22	D_{11}
11	4.1		HG(11;2|5)	(2-11)	22	D_{11}
11	4.2		HG(11:3|4)	(2-11)	22	D_{11}
11	6.1	HG(11;2,4)	HG(11;3,5|2,5)	(3-11)	22	D_{11}
11	6.2	HG(11;4,5)	HG(11;2,3|3,4)	(3-11)	22	D_{11}
11	8		HG(11;*,*,*,*)	(4-11)	22	D_{11}
11	10	K_{11}		Y	11!	S_{11}
12	2	C_{12}		Y	24	D_{12}
12	3.1		HG(12;6)	(6,12)	24	D_{12}
12	3.2	$C_6 \times K_2$		(6,12)	24	D_{12}
12	3.3			(6.12)	48	
12	3.4			(6,12)	24	D_{12}
12	4.1	$C_4 \circledcirc K_3$	HG(12;5)	Y	768	

12	4.2		HG(12;3)	(2-12)	24	D_{12}
12	4.3		HG(12;2)	(2-12)	24	D_{12}
12	4.4	$C_4 \times K_3$	c(HG(12;2,5,6))	(2-12)	48	
12	4.5		c(HG(12;4,5,6))	(2-12)	48	
12	4.6		HG(12;4)	(2-12)	24	D_{12}
12	4.7	$K_{3,3} \times K_2$		(6,18)	144	$2\text{-}S_3$, $2\text{-}Z_2$
12	4.8			(2-6,12)	48	
12	4.9			Y	48	
12	4.10			(12,12)	24	D_{12}
12	5.1		HG(12;2,6)	(6,2-12)	24	D_{12}
12	5.2		HG(12;3,6)	(6,2-12)	24	D_{12}
12	5.3		HG(12;4,6)	(6,2-12)	24	D_{12}
12	5.4	$C_6[K_2]$	HG(12;5,6)	(6,24)	768	
12	5.5		c(HG(12;4,5))	(6,2-12)	48	
12	5.6	$K_3 \times K_4$	c(HG(12;2,5))	(12,18)	144	
12	5.7	$c(K_6 \times K_2)$	$K_2[K_{2,2,2}]$	Y	1440	S_6, Z_2
12	5.8	$K_{2,2,2} \times K_2$		(6,24)	96	$4\text{-}Z_2$,S_3
12	5.9	Icosahedron		Y	120	
12	5.10			(3-6,12)	12	
12	5.11			(3-6,12)	12	
12	5.12			(6,24)	48	
12	6.1	$K_4 \circledast K_3$	HG(12;2,5)	Y	144	
12	6.2		HG(12,2,4)	(3-12)	24	D_{12}
12	6.3		c(HG(12;5,6))	(12,24)	768	
12	6.4		HG(12;2,3)	(3-12)	24	D_{12}
12	6.5		HG(12;4,5)	(12,24)	48	
12	6.6	$K_{6,6}$	HG(12;3,5)	Y	1036800	$2\text{-}S_6$, Z_2
12	6.7		HG(12;3,4)	(3-12)	24	D_{12}
12	6.8	$K_6 \times K_2$		(6,30)	1440	S_6, Z_2
12	6.9	$c(K_{2,2,2} \times K_2)$		(2-6,24)	96	$4\text{-}Z_2$,S_3
12	6.10	c(Icosahedron)		(6,30)	120	
12	6.11			(4-6,12)	12	
12	6.12			(4-6,12)	12	

12	6.13			(12,24)	48	
12	7.1		HG(12;2,3,6)	(6,3-12)	24	D_{12}
12	7.2		HG(12;2,4,6)	(6,3-12)	24	D_{12}
12	7.3		HG(12;2,5,6)	(6,12,24)	48	
12	7.4		HG(12;3,4,6)	(6,3-12)	24	D_{12}
12	7.5	$K_4[K_3]$	HG(12;3,5,6)	(6,36)	4608	
12	7.6		HG(12;4,5,6)	(6,12,24)	48	
12	7.7	$\overline{C}_6[K_2]$	c(HG(12;5))	(6,12,24)	768	
12	7.8	$c(K_{3,3}\times K_2)$		(2-12,18)	144	$2\text{-}S_3$, $2\text{-}Z_2$
12	7.9			(6,12,24)	48	
12	7.10			(6,12,24)	48	
12	7.11			(6,12,24)	24	D_{12}
12	8.1		HG(12;2,3,4)	(4-12)	24	D_{12}
12	8.2	$c(C_6\times K_2)$		(2-6,3-12)	24	D_{12}
12	8.3			(6,12,24)	48	
12	8.4			(2-12,24)	24	D_{12}
12	8.5	$K_3[C_4]$	HG(12;2,3,5)	(12,36)	288	$2\text{-}S_3$, $3\text{-}Z_2$
12	8.6	$K_{4,4,4}$	HG(12;2,4,5)	Y	82944	$3\text{-}S_4$, S_3
12	8.7	$C_4[K_3]$	HG(12;3,4,5)	(12,36)	10368	$4\text{-}S_3$, $3\text{-}Z_2$
12	9.1	$\overline{C}_{12}$	HG(12;2,3,4,6)	(6,4-12)	24	D_{12}
12	9.2	$K_{3,3,3,3}$	HG(12;2,3,5,6)	Y	31104	$4\text{-}D_3$, S_4
12	9.3	$K_{2,2,2}[K_2]$	HG(12;2,4,5,6)	(6,48)	3072	$3\text{-}D_4$, S_3
12	9.4	$c(2C_6)$	HG(12;3,4,5,6)	(6,12,36)	288	$2\text{-}D_6$, Z_2
12	10	$K_{2,2,2,2,2,2}$	HG(12;2,3,4,5)	Y	46080	$6\text{-}Z_2$, S_6
12	11	K_{12}		Y	12!	S_{12}

Acknowledgement

The material in this paper would not have been possible without the guidance and encouragement of Dr. Steve Wilson. I am indebted to his patience and understanding.

References.

1. K. Čulik, Zur Theorie der Graphen. *Časopis Pěst.* Mat. **83** (1958), 133-155.
2. G. Sabidussi, Graph multiplication. *Math. Zeit.* **72** (1960), 446-457.
3. P.M. Weichsel, The Kronecker product of graphs. *Proc. Amer. Math. Soc.* **13** (1962), 47-52.
4. R.M. Foster, A Census of Trivalent Symmetrical Graphs. Unpublished (1966).
5. D.J. Miller, The Categorical Product of Graphs. *Canad. J. Math.* **20** (1968), 1511-1521.
6. G. Sabidussi, The lexicographic product of graphs. *Duke Math. J.* **28** (1961), 573-578.
7. F. Harary, On the group of the composition of two graphs. *Duke Math. J.* **26** (1959), 29-34.
8. William Kocay, Group and Graphs. Computer software.