# COORDINATING PEBBLE MOTION ON GRAPHS, THE DIAMETER OF PERMUTATION GROUPS, AND APPLICATIONS

by

DANIEL MARTIN KORNHAUSER

submitted to the

DEPARTMENT OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

in partial fulfillment
of the requirements for the degree of

MASTER OF SCIENCE

at

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May, 1984



Signature of Author________________________________________

Department of Electrical Engineering and Computer Science
May 11, 1984

Certified by________________________________________

Prof. Gary L. Miller
Thesis Supervisor

Accepted by________________________________________

Prof. Arthur C. Smith
Chairman, Department Committee

COORDINATING PEBBLE MOTION ON GRAPHS,
THE DIAMETER OF PERMUTATION GROUPS, AND APPLICATIONS

by

Daniel Martin Kornhauser

Submitted to the Department of Electrical Engineering and Computer Science on May 11, 1984, in partial fulfillment of the requirements for the degree of Master of Science

## ABSTRACT

The problem of memory management in totally distributed computing systems leads to the following movers' problem on graphs:

Let $G$ be a graph with $n$ vertices with $k < n$ pebbles numbered $1,\ldots,k$ on distinct vertices. A move consists of transferring a pebble to an adjacent unoccupied vertex. The problem is to decide whether one arrangement of the pebbles is reachable from another, and to find the shortest sequence of moves to find the rearrangement when it is possible.

In the case that $G$ is biconnected and $k = n - 1$, Wilson (1974) gave an efficient decision procedure. However, he did not determine whether solutions require at most polynomially many moves. We generalize by giving a P-time decision procedure for all graphs and any number of pebbles. Further, we prove matching $O(n^3)$ upper and lower bounds on the number of moves required, and show how to efficiently plan solutions.

It is hoped that the algebraic methods introduced for the graph puzzle will be applicable to special cases of the general geometric movers' problem, which is PSPACE-hard (Reif (1979)).

We consider the related question of permutation group diameter. Driscoll and Furst (1983) obtained a polynomial upper bound on the diameter of permutation groups generated by cycles of bounded length. By making effective some standard results in permutation group theory, we obtain the following partial extension of their result to unbounded vcycles:

If $G$ (on $n$ letters) is generated by cycles, one of which has prime length $p < 2n/3$, and $G$ is primitive, then $G = A_n$ or $S_n$ and has diameter less than $2^{6\sqrt{p}+4}n^8$. This is a moderately exponential bound.

Thesis Supervisor: Prof. Gary L. Miller

Title: Associate Professor of Applied Mathematics

## Acknowledgements

I wish to thank M.I.T. for its financial support during the research and writing of this thesis.

I thank my advisor Gary Miller for his helpful suggestions and inspiration, and for teaching me to be persistent and to think in pictures.

TABLE OF CONTENTS

# 1. Introduction

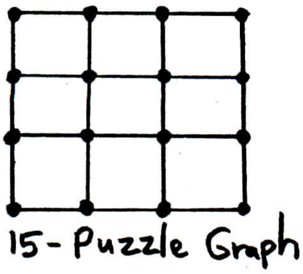
The management of memory in totally distributed computing systems is an important issue in hardware and software design. On an existing hardware network of devices, there is the problem of how to coordinate the transfer of one or more indivisible packets of data from device to device without ever exceeding the memory capacity of a device. Depending on the severity of the memory capacity, a considerable number of intermediate transfers may be necessary to clear a "path" for the movement of a data packet along a network. A combination of almost filled devices and a network configuration with few paths can, in fact, make impossible the transfer of the data packets intact.

Suppose we consider a simplified version of the above problem, where each device has unit capacity and each packet occupies one unit of memory. Then at any moment in time, any given device is either empty or is totally filled. Suppose also that at any time each data packet resides in some device. It is also assumed that only one packet may be moved at a time, from its current device to any empty immediately adjacent device. Under these assumptions, it is interesting to know whether it is possible to start from one given distribution of the packets in the network, and end with another given rearrangement, and to know how many moves are required when the rearrangement is possible.

This version of the network problem immediately translates into the following movers' problem on graphs:

Let $G$ be a graph with $n$ vertices with $k < n$ pebbles numbered $1,\dots,k$ on distinct vertices. A move consists of transferring a pebble to an adjacent unoccupied vertex. The problem is to decide whether one arrangement of the pebbles is reachable from another, and to find the shortest sequence of moves to find the rearrangement when it is possible.

It is seen that this latter problem is a generalization of Sam Loyd's famous "15-puzzle". In this puzzle, 15 numbered unit squares are free to move in a 4x4 area with one unit square blank. The problem is to move from one arrangement of the squares to another. One can easily show that this puzzle is equivalent to the graph puzzle on the square grid in Figure 1-1, with 15 numbered pebbles on the vertices and one blank vertex.



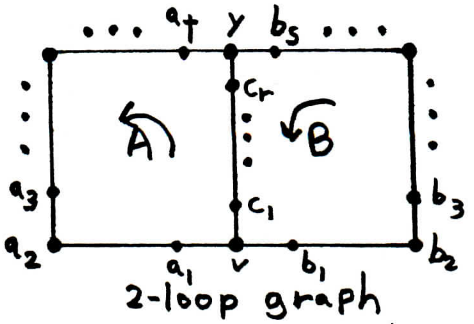




Figure 1-1 and Figure 1-2

In the case that $G$ is biconnected and $k = n - 1$, Wilson (1974) gave an efficient decision procedure. However, he did not determine whether solutions require at most polynomially many moves. His approach involved deriving a 3-cycle and 2-transitivity (these terms are defined later). This basis is known to generate all possible even permutations on the pebbles, but it is not immediately obvious that the basis is efficient. Driscoll and Furst [DF] showed that this basis is efficient; we will show how to use their result to obtain a $O(n^5)$ upper bound on the number of moves needed for solution. However, we achieve a sharper upper bound by deriving 3-transitivity. This is trickier to prove than 2-transitivity, but it enables us to obtain an $O(n^3)$ upper bound for the number of moves required in the Wilson case.

Then we generalize by giving a polynomial time decision procedure for all graphs and any number of pebbles, and we show that again at most $O(n^3)$ moves are needed and can be efficiently planned. Finally, we find an infinite family of graph puzzles for which it is proved that $O(n^3)$ moves are needed for solutions. Thus the upper and lower bounds match to within a constant factor.

A topic of related interest, and the second main part of the thesis, is the subject of permutation groups and their diameter with respect to a set of generators. Briefly, the diameter of a permutation group $G$ with respect to a set $S$ of generators for $G$ is defined to be the smallest positive integer $k$ such that all elements of $G$ are expressible as products of the generators of length at most $k$.

Consideration of the pebble coordination problem leads naturally to questions about permutation groups. Consider the graph in Figure 1-2, with vertex $v$ blank and pebbles $a_1, ..., a_t, c_1, ..., c_r, b_1, ..., b_s$, and $y$ on the other vertices. It is seen that any sequence of moves from this position will, upon the first return of the blank to $v$, net one of the following permutations on the pebbles: $A = (c_1c_2...c_rya_t...a_2a_1)$ or $B = (yc_r...c_2c_1b_1b_2...b_s)$ or $C = (b_1b_2...b_sya_t...a_2a_1)$ or $A^{-1}, B^{-1}, C^{-1}$ or the identity permutation. Hence the set of rearrangements of the pebbles (with $v$ blank) is the group of permutations generated by $S = \{A, B, C, A^{-1}, B^{-1}, C^{-1}\}$. Deciding whether a rearrangement is solvable amounts to testing membership of the corresponding permutation in the group generated by $S$; minimum number of moves is clearly related to the shortest product of generators yielding the permutation.

We view the introduction of algebraic methods as useful for the solution of movers' problems. Whereas general geometric movers' problems are PSPACE-hard (Reif (1979)), it is hoped that the techniques introduced for the solution of the pebble coordination problem may be applicable to special cases of the general geometric problem.

We now briefly discuss the state of the art in permutation group membership and diameter questions. Furst, Hopcroft and Luks [FHL] give a polynomial time algorithm for deciding whether a given permutation $g$ is in $G(S)$, the group generated by $S$. Thus the analogue of the graph decision problem is in $P$. One also immediately has a P-time criterion for deciding solvability of the Rubik's Cube and the Hungarian Rings puzzles. The situation is not as fortunate when one tries to find the length of the shortest generator sequence for a given permutation: Jerrum [J] has recently shown this to be PSPACE-complete! The difficulty may be related to the fact that some groups may have superpolynomial diameter. For example, the group $G$ generated by the single permutation (12)(345)(6789 10)...(...$s$) where $s$ is the sum of the first $n$ prime numbers, can be shown to have diameter roughly on the order of $2^{O(\sqrt{n})}$ (see figure 1-3 for a mechanical realization of this group). This contrasts with the analogous question for the pebble coordination problem, where no solution can ever require more than $O(n^3)$ moves. Therefore the group diameter question is in some sense more general, and probably more difficult, than the corresponding question for pebble motion.

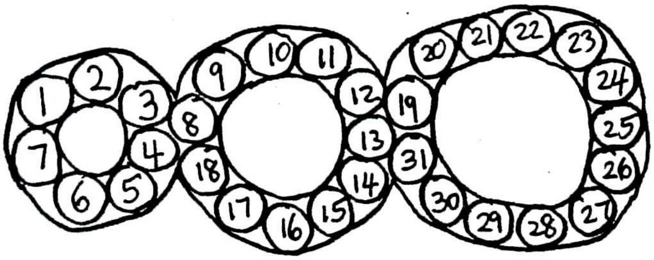



Figure 1-3 A geometrical movers' problem requiring exponentially many moves

There are nonetheless some interesting recent results concerning upper bounds on group diameter, for special generating sets. Driscoll and Furst [DF] have shown that if all the generators are cycles of bounded length, then the group has $O(n^2)$ diameter where $n$ is the number of letters that the group acts on. More recently, McKenzie [M] obtained the upper bound $O(n^k)$ on diameter for groups, each of whose generators moves at most $k$ letters. This is polynomial if $k$ is bounded.

The foregoing results leave open the question of a group's diameter when the generators are arbitrary (not of bounded length) cycles. In Chapter 4 we informally discuss certain generalizations of the Hungarian Rings puzzle, and find sufficient conditions for the required number of moves to be polynomial. Examples which do not meet these sufficient conditions are offered as possible candidates for groups with superpolynomial diameter. However, this part of the chapter is speculative. The rest of Chapter 4 consists of a number of new results in permutation groups, which extend classical theorems by providing upper bounds on diameter. We obtain the following theorem as a corollary (all definitions will be given later):

If $G$ (on $n$ letters) is generated by cycles, one of which has prime length $p < 2n/3$, and $G$ is primitive, then $G = A_n$ or $S_n$ and has diameter less than $2^{6\sqrt{p}+4}n^8$.

This upper bound is only moderately exponential, but is nonetheless superpolynomial. It remains of interest to know whether the bound can be significantly improved, or whether the diameter really can be this large.

At the end of the thesis we present conjectures, open problems, and suggestions for further research in movers' problems and permutation group diameter.

## 2. Basic Definitions and Results about Permutation Groups

We introduce some basic definitions and concepts in permutation groups which will be needed to facilitate later discussions.

A *permutation* is a one-to-one mapping from a finite set to itself. The set may contain any objects, which are often referred to as *letters*. We often denote a set of $n$ objects by $\{1, 2, ..., n\}$, the first $n$ positive integers.

We think of the integer $i$ as representing a *position* which is occupied by a letter, and a permutation as a mapping of positions. Thus a permutation which maps $i$ to $j$ is thought of as moving the letter at position $i$ over to position $j$. This distinction between letter and position is assumed throughout, but usually will not be voiced. Thus we will often speak of a "permutation on $n$ letters" rather than a "permutation on $n$ positions".

Suppose that a permutation $p$ maps $1, 2, ..., n$ to $r_1, r_2, ..., r_n$ respectively. We can represent the action of $p$ by the notation

$$p = \begin{pmatrix} 1\,2\,...n \\ r_1 r_2 ... r_n \end{pmatrix}.$$

This indicates that a letter $i$ is sent to the letter indicated directly below it. If a letter $i$ is fixed by the permutation, it is often omitted for brevity of notation. For example,

$$\begin{pmatrix} 1\,2\,3\,4\,5\,6 \\ 2\,3\,1\,4\,5\,6 \end{pmatrix} \text{ is abbreviated as } \begin{pmatrix} 1\,2\,3 \\ 2\,3\,1 \end{pmatrix}.$$

The permutation on $n$ letters which fixes all the letters is called the *identity permutation*. We will abbreviate it here by () or $ID$.

If a permutation $p$ moves only letters at positions $a_1, a_2, ..., a_k$ and sends $a_i$ to $a_{i+1}$ for $i = 1, ..., k-1$ and $a_k$ to $a_1$, then $p$ is called a $k-cycle$. Its notation

$$\begin{pmatrix} a_1 a_2 ... a_{k-1} a_k \\ a_2 a_3 ....... a_k a_1 \end{pmatrix}$$

is often abbreviated by

$(a_1 a_2 ... a_k)$. This new notation means that each letter that appears is to be mapped to the letter appearing cyclically to its right, and letters not appearing are to be fixed. A $k$-cycle is also called a *cyclic permutation* of $k$ letters, or a *cycle of length* $k$. A 2-cycle is often called a *transposition* or a *swap*.

If $p$ and $q$ are two permutations, then the *product* $pq$ is defined as the composition of the permutation $p$ followed by $q$. It is easy to show that any permutation can be expressed as a product of cycles, such that no letter appears in more than one cycle. This is called a *product of disjoint cycles*. For example,

$\begin{pmatrix}1\,2\,3\,4\,5\,6\,7\\4\,6\,5\,2\,3\,1\,7\end{pmatrix}$ can be written as the following product of disjoint cycles: (1426)(35).

This product notation is succinct, and it tends to give a clearer idea of the action of a permutation on the letters.

It is not hard to show that any permutation on $n$ letters can be written as a product of at most $n-1$ transpositions (not necessarily disjoint). For example, (1 2 3) = (1 2)(1 3). A permutation is said to be *even* if it is expressible as a product of an even number of transpositions, otherwise it is said to be an *odd* permutation. For example, any 3-cycle $(a_1a_2a_3)$ is an even permutation since $(a_1a_2a_3) = (a_1a_2)(a_1a_3)$ is a product of two transpositions. If $p$ is an even permutation, we can abbreviate this by saying that "p is even". It can be shown that a permutation is either even or odd, but not both.

Let $S_n$ denote the set of all permutations on $n$ letters. It is a group under the binary operation of composition. Similarly, let $A_n$ denote the group of all even permutations on $n$ letters. (It is a group because the product of two even permutations is even, and the inverse of an even permutation is even.)

A group $G$ of permutations on $n$ letters (i.e. a subgroup of $S_n$) is said to be $k-$*transitive* if, for any $k$ distinct letters $a_1, ..., a_k$ and any other (possibly overlapping set of) $k$ distinct letters $b_1, ..., b_k$, there exists a permutation $p \in G$ which sends $a_i$ to $b_i$, $i = 1, ..., k$ (and may or may not move other letters). A 1-transitive group is simply said to be *transitive*. For example, $G = \{(123), (132), ()\}$ is a transitive group on three letters but is not 2-transitive.

A useful fact is that for $G$ to be $k$-transitive, it is sufficient that there exist a (fixed) set of letters $c_1, ..., c_k$ such that any set $a_1, ..., a_k$ can be sent to $c_1, ..., c_k$ respectively. Proof: to send $a_1, ..., a_k$ to $b_1, ..., b_k$, first send $a_1, ..., a_k$ to $c_1, ..., c_k$; then perform the inverse of a permutation which sends $b_1, ..., b_k$ to $c_1, ..., c_k$. This sends $c_1, ..., c_k$ to $b_1, ..., b_k$, so the composition sends $a_1, ..., a_k$ to $b_1, ..., b_k$.

It is immediate that $S_n$ is $n$-transitive. It is also easy to show that $A_n$ is $n-2$-transitive: to send $a_1, ..., a_{n-2}$ to $b_1, ..., b_{n-2}$, observe that as

$$\begin{pmatrix}a_1 \dots a_{n-2}\; a_{n-1}\; a_n\\ b_1 \dots b_{n-2}\; b_{n-1}\; b_n\end{pmatrix} \text{ and } \begin{pmatrix}a_1 \dots a_{n-2}\; a_{n-1}\; a_n\\ b_1 \dots b_{n-2}\; b_n\; b_{n-1}\end{pmatrix}$$

differ by a transposition, one of them must be even, and so must be in $A_n$.

Let $S = \{p_1, ..., p_k\}$ be a finite set of permutations on $n$ letters. Let $G$ be the set of permutations formed by products of any elements of $S$ with any number of repetitions. Then $G$ is clearly a group, called the *permutation group generated by* $S$. We write $G = G(S)$. The elements of $S$ are called the *generators*. If $g \in G$, then $g$ can be written as a product of generators. Such a product is called a *word*, and the *length* of the word is the number of terms in the product (including repetitions). The *diameter* of $G(S)$, written $diam(G(S))$, is the smallest positive integer $k$ such that every $g \in G$ is expressible as a word of length $k$ or less.

We now prove the following important fact used extensively later on:

$A_n$ is generated by the set of all 3-cycles on $n$ letters, and any element of $A_n$ is expressible as a product of at most $n-2$ 3-cycles.

Proof. Let $S$ be the set of all 3-cycles on $n$ letters. A 3-cycle is even, so $G(S)$ is a subgroup of $A_n$. To show that $G(S) = A_n$, we show that $A_n$ is a subset of $G(S)$, as follows. Let $p \in A_n$ send $a_1, ..., a_n$ to $b_1, ..., b_n$ respectively. The 3-cycle $p_1 = (a_1 b_1 c_1)$ (for any letter $c_1$ besides $a_1, b_1$) sends $a_1$ to $b_1$. If $a_2$ is not sent to $b_2$ by $p_1$, we multiply $p_1$ with $p_2 = (p_1(a_2) b_2 c_2)$ where $c_2$ is not $b_1$, and then $p_1 p_2$ takes $a_1$ to $b_1$ and $a_2$ to $b_2$. Continuing in this way, let $p_k = (p_1 * ... * p_{k-1}(a_k) b_k c_k)$ where $c_k$ is not $b_1, ..., b_{k-1}$. Then by induction $p_1 * ... * p_k$ sends $a_1, ..., a_k$ to $b_1, ..., b_k$ respectively. This succeeds up to and including $k = n-2$, after which $c_{k+1}$ cannot avoid $b_1, ..., b_k$. But we have already have moved all but two letters to their destinations, by a product of $n-2$ or fewer 3-cycles. Now, this product must also take the last two letters to their proper spots, for if instead they were interchanged we would have an odd permutation, which is impossible.

This completes the proof, which is in fact an efficient algorithm for representing a permutation in $A_n$ as a product of $\leq n-2$ 3-cycles. In an exactly similar way we can show that any element of $S_n$ is a product of at most $n-1$ swaps.

Let us now define *conjugation* and give a few of its basic properties. If $S$ and $T$ are permutations, we define $S$ *conjugated by* $T$ as the permutation $T^{-1}ST$. Property: If $S$ takes letter $a$ to $b$, then $T^{-1}ST$ takes $T(a)$ to $T(b)$. For $T^{-1}ST$ takes $T(a)$ first to $a$, then to $b$, and finally to $T(b)$. From this property it follows that if

$$S = (a_{i_1} ... a_{i_2})(a_{i_3} ... a_{i_4}) ... (a_{i_k} ... a_{i_{k+1}})$$

then

$$T^{-1}ST = (T(a_{i_1}) ... T(a_{i_2}))(T(a_{i_3}) ... T(a_{i_4})) ... (T(a_{i_k}) ... T(a_{i_{k+1}})) \ .$$

That is, to conjugate $S$ by $T$, simply replace each letter in the cycle structure of $S$ by its image under $T$. Hence conjugation may change the underlying set of letters moved, but does not change the cycle structure of a permutation.

From the property of conjugation just described, it is immediate that the conjugate of a $k$-cycle is again a $k$-cycle. An important fact which we will use often in what follows is:

If $G$ has a permutation which is a $k$-cycle, and $G$ is $k$-transitive, then $G$ contains all $k$-cycles.

For if $S = (a_1 ... a_k) \in G$, then for any distinct letters $b_1, ..., b_k$ we can find $T \in G$ which maps $a_1, ..., a_k$ to $b_1, ..., b_k$ respectively (by $k$-transitivity). Then by the property of conjugation given in the previous paragraph, $T^{-1}ST = (b_1 ... b_k)$. As $T^{-1}ST$ is in

$G$, so is $(b_1...b_k)$. As $b_1, ..., b_k$ are arbitrary letters, we conclude that $G$ contains all $k$-cycles.

When $k = 3$ we get the useful result that, if $G$ contains a 3-cycle and $G$ is 3-transitive, then $G = A_n$ or $S_n$. For $G$ contains all 3-cycles, and therefore contains $A_n$; it follows that $G = A_n$ or $S_n$, since it is easy to show that $A_n$ is a subgroup only of $A_n$ and $S_n$.

We now define the notion of primitivity. A *block* of a group $G$ on $n$ letters is a subset $B$ of the letters with the property that for all $g$ in $G$, $g(B) = B$ or $g(B)$ and $B$ have empty intersection. Informally, a block is a set of letters that always maps either exactly onto itself or completely outside of itself. Clearly the empty set, the set of all $n$ letters, and each set of a single letter, are all blocks of $G$, no matter what the group $G$. For this reason, these are called *trivial* blocks. A group $G$ with no nontrivial blocks is called *primitive*.

Some properties of primitive groups:

1. $G$ primitive implies $G$ is transitive. For if $G$ were not transitive, then consider the set $G(a)$ of places that the letter $a$ is mapped by various permutations of $G$ (where $a$ is not fixed by all of $G$). $G(a)$ is clearly a block of $G$. However, $G(a)$ contains at least two letters, and $G(a)$ cannot be all $n$ letters because $G$ is assumed intransitive. Hence $G(a)$ is a nontrivial block, so $G$ is imprimitive, contrary to hypothesis.

2. $G$ 2-transitive implies $G$ is primitive. For let $B$ be any set of two or more, but less than $n$ letters. Let $a_1, a_2$ be in $B$, and let $a_3$ be outside of $B$. Then by 2-transitivity there is a $g \in G$ which maps $(a_1, a_2)$ to $(a_1, a_3)$. Hence this $g$ maps $B$ onto part of itself, so $B$ is not a block. Since $B$ was arbitrary, we conclude that $G$ is primitive.

3. The following is an especially useful property:

$G$ is primitive if and only if, for any nonempty proper subset $B$ of the $n$ letters, and any distinct letters $a, b$, there is a permutation in $G$ which maps one letter into $B$ and the other letter outside of $B$.

Proof.

The right side of the "if and only if" clearly implies that $G$ is primitive, in the same way as we proved property 2. For the converse, let $G$ be primitive. Let $B$ be any nonempty proper subset of the $n$ letters. Say that $a$ is equivalent to $b$ if for all $g \in G$ we have that $g(a) \in B$ iff $g(b) \in B$. This is easily checked to be an equivalence relation on the $n$ letters. Let $B_1 = \{a_1, ..., a_k\}$ be an equivalence class. We will show that the right hand side of the "iff" holds by proving that $k$ must equal 1. For any $g \in G$, $g$ maps all of $B_1$ into $B$ or all of $B_1$ outside of $B$. Since $B$ is a nonempty proper subset of the $n$ letters, we see that $B_1$ cannot be the full set of $n$ letters. Also observe that for any equivalence class $B_1$ and any $g \in G$, $g(B_1) = B_2$ is also an equivalence class. For the letters of $B_1$ are equivalent, so the letters of $B_2$ are also equivalent. If $B_2$ were not an equivalence class but rather a proper subset of an equivalence class $B_3$, then the letters of $g^{-1}(B_3)$ would be equivalent and contain $B_1$ as a proper subset; this contradicts the assumption that $B_1$ is an equivalence class. Thus any image of $B_1$ is an equivalence class, and since equivalence classes are identical or disjoint, we conclude that $B_1$ is a block. Since $G$ is primitive, $B_1$ must be a trivial block; since $B_1$ is not all $n$ letters, $B_1$ must consist of a single letter. This completes the proof of property 3.

In fact the following slightly stronger property holds, but we do not need it in what follows.

$G$ is primitive if and only if, for any nonempty proper subset $B$ of the $n$ letters, and any distinct letters $a, b$, there is a permutation in $G$ which maps $a$ into $B$ and $b$ outside of $B$.

The proof may be found in [Wielandt , p.15].

A group which is not primitive is called *imprimitive*.

We now derive one more useful fact which will be used later, namely:

2-transitivity can be accomplished in wordlength $< n^2$, independent of the generator set. More precisely, for any generating set $S$, and any $a_1 \neq a_2, b_1 \neq b_2$, we can find a $g \in G$ which moves $a_1, a_2$ to $b_1, b_2$ respectively and has wordlength $< n^2$.

Proof

Form a directed graph with a vertex for each ordered pair of distinct letters, and a directed edge from $(c_1, c_2)$ to $(d_1, d_2)$ iff some generator $s$ takes $c_1, c_2$ to $d_1, d_2$ respectively.

Now, starting on the vertex corresponding to $(a_1, a_2)$, we want to find a directed path to the vertex corresponding to $(b_1, b_2)$. Such a path exists since we are assuming that $S$ generates a 2-transitive group. The path can clearly be taken to be simple, i.e. no vertex is visited more than once; for a nonsimple path can be made simple by removing the closed "loops" from the path. Since there are $n(n-1) < n^2$ vertices on the graph, the path must have length $< n^2$. This yields a word of length $< n^2$ which effects the desired 2-transitive operation, and the proof is complete.

We can effectively find a word shorter than $n^2$ which gives 2-transitivity, as follows: begin at the start vertex, and follow each directed edge which leads to another vertex. Then starting at each of these new vertices, follow each directed edge which leads to a vertex not previously visited. Repeat this until we reach the target vertex. Since no edge is examined more than once, the running time of this algorithm is at most $|E| < |V|^2 < (n^2)^2 = n^4$.

In a similar way, we can do primitive operations efficiently. That is, for a proper subset $B$ of $\{1, 2, ..., n\}$, and distinct letters $a, b$, we can find a $g \in G$ with wordlength $< n^2$ which moves one letter into $B$ and the other letter outside of $B$. The method is the same as for 2-transitivity, but now any vertex corresponding to a desired image of $(a, b)$ is an acceptable target vertex. With more than one target vertex, the length of the path can only get shorter; similarly the search time can only get less.

The above result on 2-transitivity can immediately be generalized to $k$-transitivity: a $k$-transitive operation can be performed in wordlength $< n^k$, and the search time for finding such a word is $< n^{2k}$. [DF] gives a proof of this result which is similar to our proof.

# 3. Coordinating Pebble Motion on Graphs

## 3.1. Preliminary Remarks

In this chapter we will tackle the pebble coordination problem given in the introduction:

Let $G$ be a graph with $n$ vertices with $k < n$ pebbles numbered $1,\ldots,k$ on distinct vertices. A move consists of transferring a pebble to an adjacent unoccupied vertex. The problem is to decide whether one arrangement of the pebbles is reachable from another, and to find the shortest sequence of moves to find the rearrangement when it is possible.

Before describing the details of the decision algorithm and $O(n^3)$ upper and lower bounds on number of moves, we first make the simplifying assumption that the set of unoccupied vertices of $G$ is the same in both the initial and final arrangements. One reason for this assumption is that the analysis becomes simpler. Move sequences which do not change the set of unoccupied vertices induce a permutation on the tokens in a natural way. The set of permutations induced by such move sequences is clearly a group, and we can apply notions of permutation groups to the problem. On the other hand, if the unoccupied vertices are not the same from start to end position, then it is less natural to assign permutations to actions on the tokens. The other justification is that any graph puzzle can easily be reduced to a graph puzzle satisfying the assumption. This follows from the following Lemma.

### Lemma

For any connected graph $G$ with $n$ vertices, and a placement of $k < n$ pebbles on $k$ distinct vertices (with the other vertices blank), it is possible to reach a position where any $k$ desired vertices are covered by pebbles. The number of moves required is at most $O(n^2)$.

Suppose we want to get from position $P_1$ to position $P_2$. Use the Lemma to move from $P_2$ to a position $P_2'$ which has the same vertices unoccupied as in $P_1$. Clearly we can reach $P_2$ from $P_1$ if and only if we can reach $P_2'$ from $P_1$. Hence we decide arbitrary puzzles by reducing them to the assumed form, then deciding the puzzle in this form. Upper and lower bounds $O(n^3)$ are valid for arbitrary puzzles, once we establish these bounds for puzzles satisfying the assumption, because at most $O(n^2)$ moves (which is absorbed in the $O(n^3)$ term) are needed to change to the assumed form.

Hence from now on, we assume without loss of generality that the initial and final positions of the puzzle have the same vertices unoccupied.

### Proof of the Lemma

Perform the following procedure.

A. Find a minimum spanning tree $T$ for $G$. We will obtain the desired position by only making moves along edges of $T$.

B. If $T$ is empty, terminate the procedure. Otherwise, select any "branch end" of $T$, i.e. a vertex $v$ with valence 1 in $T$. If $v$ is desired to have a pebble on it and $v$

already has a pebble, or if $v$ is desired to unoccupied and $v$ is currently unoccupied, then "prune" $v$ from $T$, and go to step B.

Otherwise, suppose $v$ is unoccupied and is desired to have a pebble. Select a pebble which is the minimum distance from $v$ on $T$. Then the path from the pebble to $v$ is clear of pebbles. Move the pebble along the path to $v$, prune $v$ from $T$, and go to step B.

Or, suppose $v$ is occupied and is desired to be unoccupied. Find an unoccupied vertex $u$ which is the minimum distance from $v$ on $T$. Then the path from $u$ to $v$ has pebbles on each vertex except $u$. Move each pebble on the path one edge towards $u$ (start with the pebble next to $u$, then move each pebble in turn until the pebble on $v$ moves off of $v$). This makes $v$ unoccupied. Prune $v$ from $T$, and go to step B.

Note that each application of step B puts another pebble or space into place, without disturbing the placements made during previous applications of step B. Therefore the procedure terminates with tokens on the desired vertices, and the other vertices blank. Since clearly at most $n$ moves are made at each application of step B, and step B is executed $n$ times, the total number of moves required is $< n^2$. This proves the Lemma.

## The Graph Puzzle

We now solve the decision aspect of the pebble coordination problem. It turns out to be natural to divide the analysis into two cases: 1. Biconnected graphs with all but one vertex occupied (the Wilson case) 2. All other cases (unconnected graphs, separable graphs, and the biconnected case with at least two blanks).

We remark that the Wilson case is the more interesting from the group theoretical point of view. It can be shown that graphs without closed paths need at most $O(n^2)$ moves to solve; it is the loop structure of biconnected graphs which results in some puzzles requiring $O(n^3)$ moves.

In what follows, we will assume that all graphs are *simple*, that is, no two vertices are directly joined by more than one edge, and no vertex is joined to itself by an edge. It is clear that if a graph $G$ is nonsimple, we can remove the "extraneous" edges to get a simple graph $G'$, and the graph puzzle on $G'$ is exactly equivalent to that on $G$, both with respect to solvability and the number of moves needed to solve it. For the extra edges neither help nor hinder solutions. Hence there is no loss of generality in making this assumption.

Before we proceed, let us describe precisely how the transformation from one position to another defines a permutation on the pebbles. Denote the vertices on which pebbles initially reside by $v_1, \ldots, v_k$. Suppose that after a sequence of moves, the pebble initially on $v_i$ ends up at $v_{j_i}$, $i = 1, \ldots, k$, where $v_{j_1}, \ldots, v_{j_k}$ is the same vertex set as $v_1, \ldots, v_k$ but possibly in a different order. Then we define the permutation induced by the sequence of moves to be

$$\begin{pmatrix} 1\ 2\ 3 \ldots k \\ j_1\ j_2 \ldots j_k \end{pmatrix}.$$

This is a natural way to define permutations on the pebbles induced by a change in configuration.

## 3.2. Biconnected Graphs, one Blank

### 3.2.1. Introduction

We introduce Wilson's theorem, and prove it in a way which we believe is simpler than the original proof. This new proof enables us to obtain a $O(n^3)$ upper bound on the number of moves needed for solution.

A connected graph $G$ is said to be *separable* if, by removing some vertex $v$ and all edges incident to $v$, one obtains two or more connected graphs. Separable graphs are also called $1-$*connected* because the deletion of one vertex will disconnect the graph. A vertex which separates the graph in this way is called a *cutpoint*. Observe that the property that vertex $v$ is a cutpoint, is equivalent to the property that there exist $v_1, v_2$ in $G$ such that all paths between them pass through $v$. A *biconnected* or *nonseparable* graph is defined to be a connected graph which is not separable. At least two vertices must be removed to disconnect a biconnected graph. Any biconnected graph, except for the graph consisting of two vertices joined by an edge, has the property that the removal of a single edge cannot disconnect the graph. For if an edge $e$ separated the graph, then either of the two endpoints of $e$ with valence $> 1$ (at least one has valence $> 1$ unless we have the graph just mentioned) is a cutpoint.

A *path* $p$ in a graph $G$ is a sequence $p = [x_0, x_1, ..., x_n]$ of vertices of $G$ s.t. $x_{i-1}$ and $x_i$ are adjacent in $G$, $i = 1, 2, ..., n$. Such a path $p$ is said to be "from $x_0$ to $x_n$". The path $p$ is *simple* when $x_0, x_1, ..., x_n$ are distinct, with the possible exception that $x_0 = x_n$, in which case $p$ is a *simple closed path*. Define a *polygon* to be a graph consisting of a simple closed path containing at least two vertices. A polygon looks like a "loop" containing two or more vertices. Let $G_0$ be the graph consisting of two vertices joined by an edge. Let $T_0$ be the graph shown in Figure 3-1.

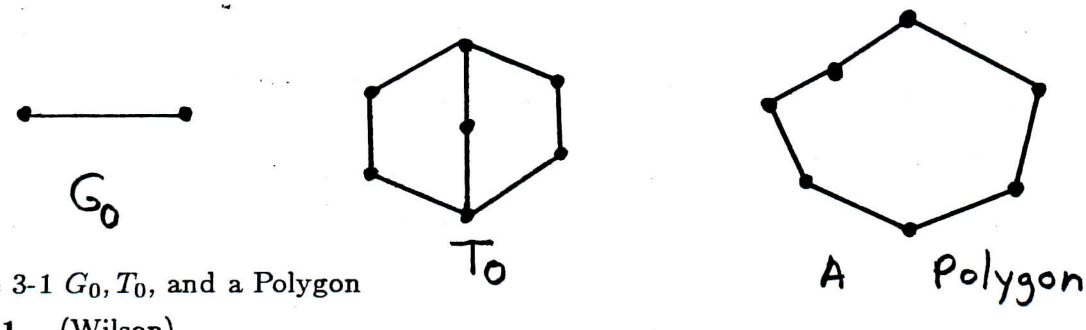



Figure 3-1 $G_0, T_0$, and a Polygon

**Theorem 1** (Wilson)

Let $G$ be a biconnected graph on $n$ vertices, other than a polygon or $T_0$, with one blank vertex. If $G$ is not bipartite, then the puzzle is solvable. If $G$ is bipartite, then the puzzle is solvable iff the permutation induced by the initial and final positions is even.

Bipartitism can be tested in $O(E)$ time, $E$ the number of edges of $G$. The permutation can be calculated in product of cycles form in $O(n)$ time on a random access machine; then the parity can be checked in an additional $O(n)$ time. Hence Wilson's criterion takes $O(E)$ time.

For $G$ a polygon, only cyclical rearrangements of the tokens are possible, so it is easy to check reachability in this case in $O(n)$ time. For the special graph $T_0$, we can simply precalculate (by exhaustive search) a table of all pairs of positions, indicating which pairs are mutually reachable. Table lookup is constant time, hence we have a $O(E)$ decision algorithm for all biconnected graphs with one blank.

It will turn out as a special case of the next section, that the biconnected case with two or more blanks is always solvable.

### Theorem 2

Let $G$ be a biconnected graph. Let $n = |V(G)|$. If labeling $g$ can be reached from labeling $f$ at all, then this can be done within $O(n^3)$ moves, and such a sequence of moves can be efficiently generated.

First we sketch the proof of Theorem 1; then the full proof will be given. We will then prove Theorem 2.

### Sketch of Proof of Theorem 1

It is a well-known fact in graph theory that a biconnected graph, other than $G_0$, can be viewed as being "grown", by starting with a polygon graph and successively adding zero or more "handles". To be precise, let $G$ be a biconnected graph besides $G_0$. Select any vertex $v$ in $G$, and any vertex $w$ adjacent to $v$ in $G$ via an edge $e$. Since $G$ is biconnected and not $G_0$, removing $e$ will not disconnect $G$, hence there must be a path (which can be chosen to be simple) from $w$ to $v$ which does not traverse $e$. Combining the path from $v$ to $w$ along $e$, with a simple return path from $w$ to $v$ which avoids $e$, yields a polygon. If this polygon $H$ does not contain all vertices of $G$, then let $w_1$ be a vertex in $G - H$ which is adjacent to a vertex $v_1$ in $H$ via an edge $e_1$. Then, reasoning as before, there is a simple path from $w_1$ to a vertex in $H$, which avoids the edge $e_1$. The augmentation of this path to $H$ looks like "adding a handle to H". Proceeding in this way, we continue adding handles until the augmented graph contains all vertices of $G$. At this point, we have all of $G$ except perhaps for some edges. Add these edges one by one to the augmented graph till we have the entire graph $G$ (we can think of these final edges as vertex-free handles).

A biconnected graph which can be "grown" by adding $i$ handles to a polygon, appears pictorially to consist of $i+1$ simple loops joined together in some way. This number of loops is called the *Betti number* of the graph, and we denote it by $B(G)$. It can be shown that for a general graph $G$, $B(G) = |E(G)| - |V(G)| + 1$ . We will often denote a biconnected graph with Betti number $i$ by the term *$T_i$-graph*. Wilson's theorem will be proved by induction on the Betti number of the graph. We skip the $T_1$-graphs (the polygons) and begin the induction with the $T_2$-graphs (except $T_0$).

The main step is to show that the group of possible induced permutations always contains the alternating group $A_{n-1}$ on the $n-1$ pebbles. The final step is to determine whether the group is $A_{n-1}$ or $S_{n-1}$. The group will be $S_{n-1}$ iff it contains an odd permutation, and it is easy to see that there is an odd permutation iff the graph has a closed path of odd length. As a graph has a closed path of odd length iff it is not bipartite, we see that the group is $A_{n-1}$ if the graph is bipartite, and $S_{n-1}$ if the graph is not bipartite. Therefore, to check solvability on a bipartite graph, it is necessary and sufficient that the induced permutation be even; on a nonbipartite graph, the puzzle is always solvable.

To show that the group of induced permutations contains the alternating group, we show how to obtain a 3-cycle and how to obtain 3-transitivity. We know how to generate the alternating group from this basis, by Chapter 2.

A 3-cycle is obtained roughly as follows. A $T_2$-graph looks like that pictured in Figure 3-2. Let vertex v be blank, and pebbles $a_1, ..., a_t, c_1, ..., c_r, b_1, ..., b_s$, and $y$ on the other vertices.

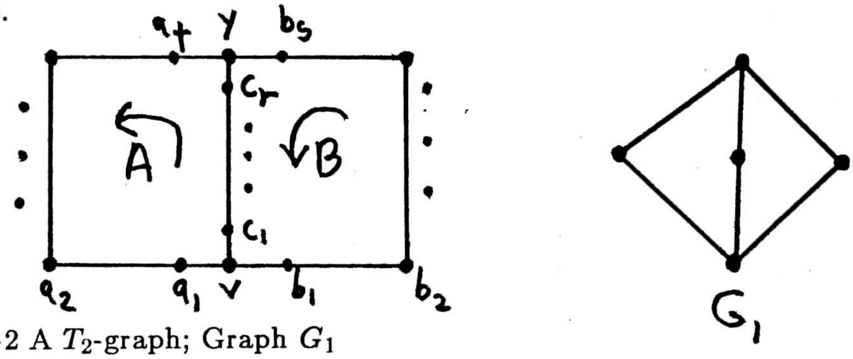



Figure 3-2 A $T_2$-graph; Graph $G_1$

Assume first that $r = 0$, i.e. the center arc has no internal vertices. $A = (ya_t...a_1)$ and $B = (b_1...b_s y)$ are permutations induced by moving pebbles around, respectively, the left or right loops. Then $ABA^{-1}B^{-1} = (yb_s a_1)$, a 3-cycle. If $r > 0$, then $ABA^{-1}B^{-1}$ is a product of two swaps; we will show in the full proof how to obtain a 3-cycle from this, if the graph induces 4-transitivity. It turns out that the graph $T_0$ is the only $T_2$-graph which does not induce a 3-cycle. We then show that all $T_i$-graphs, $i > 2$ give a 3-cycle, because they are formed by adding handles to a $T_2$-graph which can induce the 3-cycle. The hole in the induction due to $T_0$ will be taken care of with no difficulty.

3-transitivity will also be shown by induction. It will be shown that all $T_2$-graphs except the graph $G_1$ shown in Figure 3-2 are 3-transitive, by a simple Lemma. Then we show how adding a handle to a 3-transitive graph yields a 3-transitive graph. The hole in the induction due to $G_1$ will be handled without trouble.

Putting 3-cycle and 3-transitivity together, we will conclude that all $T_i$-graphs, $i \geq 2$, generate at least the alternating group, except $T_0$.

In Wilson's proof, a 3-cycle and 2-transitivity are derived. This basis generates $A_n$, but Wilson did not examine how efficiently it does so. [DF] showed that the diameter is $O(n^5)$ if the 3-cycle is considered to be one of the generators. The proof is essentially this: 3-cycle + 2-transitivity generates $A_n$ which, for $n \geq 5$, is 3-transitive (if $n < 5$, the diameter is $O(1)$, so we're done in this case). Now, 3-transitivity can be accomplished in wordlength $O(n^3)$ (see Chapter 2), so by conjugation, any 3-cycle has wordlength $O(n^3)$. (On a biconnected graph, we can choose the generators to be cycles around loops, and their inverses. So the introduction of inverses, due to conjugation, does not increase the wordlength.) Since any element of $A_n$ is a product of $O(n)$

3-cycles, this gives $A_n$ a diameter of $O(n^4)$. Since (as just mentioned) the generators can be chosen to be cycles, and cycles take $O(n)$ moves on a graph, the total number of moves is $O(n^5)$.

Incidentally, Theorem 2 of Chapter 4 implies ($p = 3$ and $Diam(H(S)) = O(1)$) that a 2-transitive group with a 3-cycle as one of its generators has diameter $O(n^5)$; but this bound is not as good as the $O(n^4)$ just obtained.

We achieve the sharper upper bound of $O(n^3)$ moves by obtaining 3-transitivity directly from the graph in $O(n^2)$ moves. The above derivation obtains 3-transitivity in wordlength $O(n^3)$, which means $O(n^4)$ moves. That explains why our bound is better. Our proof of 3-transitivity on graphs is slightly tricky, especially in dealing with graph $G_1$, but it gives the sharp upper bound; we will see later in this chapter that the bound is optimal to within a constant factor.

### 3.2.2. Proof of Theorem 1

We will prove Theorem 1 by induction on the Betti number.

#### 3.2.2.1. Proof of Theorem 1 for $T_2$-graphs

As the basis step, we prove Theorem 1 for biconnected graphs $G$ with $B(G) = 2$, the $T_2$-graphs (except $T_0$).

Let $G$ be a $T_2$-graph other than $T_0$.

Let $x$ and $y$ be the vertices with valence 3. Let the three arcs from $x$ to $y$ have respectively $t$, $r$, $s$ internal nodes where $t \geq s \geq r$, drawn here with the loop having $t$ on the left and the loop having $s$ on the right. Note that $s \geq 0$ as $G$ is simple. Put the blank token at $x$ (see Figure 3-2).

We wish to show that $A_{n-1}$ is a subset of the possible permutations on the $n-1$ pebbles. The strategy will be to generate a 3-cycle and to obtain 3-transitivity, so that all other 3-cycles are obtained. This will give us all of $A_{n-1}$, since as is well known, the 3-cycles generate the alternating group.

Let $A$ be the permutation induced by moving pebbles around the path $[xc_1...c_r y a_t...a_1 x]$ . This in effect clockwise "rotates" the pebbles on the left loop. Similarly let $B$ be the permutation induced by the path $[xb_1...b_s y c_r...c_1 x]$; i.e. the clockwise rotation of the pebbles on the right loop.

Claim: A and B generate at least $A_{n-1}$.

**Proof of Claim**

Let $K$ be the group generated by $A$ and $B$.

a) If $r = 0$, then $ABA^{-1}B^{-1}$ gives the 3-cycle $(a_t y b_1)$. $(t \geq 0)$ b) If $r > 0$, then $ABA^{-1}B^{-1}$ gives the permutation $(a_t y)(c_1 b_1)$.

Now if $K$ is 4-transitive, then in case a) we get (using just 3-transitivity) all 3-cycles and so the alternating group. In case b), we can use 4-transitivity to get all pairs of disjoint transpositions. Using (1 2)(3 4) and (1 2)(4 5) so obtained, their product gives us (3 5 4), a 3-cycle. Then use 3-transitivity to obtain all 3-cycles and so the alternating group.

By Lemma 1 (stated and proved at the end of this section), we have 4-transitivity if $t \geq 3$. Hence all that remains is to prove the claim for those $T_2$-graphs with $t < 3$. These have $t, s, r$ respectively:

i) (2 2 2)

ii) (2 2 1) $= T_0$

iii) (2 2 0)

iv) (2 1 1)

v) (2 1 0)

vi) (1 1 1)

vii) (1 1 0)

It will be seen presently that the claim is true for each of these cases; this will complete the proof of the claim.

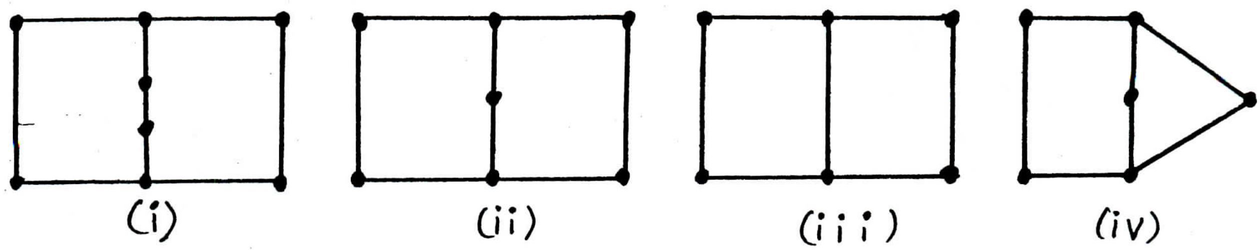


Figure 3-3 i,ii,iii,iv

i) (2 2 2) :

$ABA^{-1}B^{-1}$ gives $(a_2y)(c_1b_1)$. Let $X = A^2(BA)^2A^{-2}$. $X$ fixes $a_2, y$ and takes $b_1$ to $c_1$, $a_1$ to $b_1$ (as well as moving other things). So $X(a_2y)(c_1b_1)X^{-1} = (a_2y)(a_1b_1)$. Then $(a_2y)(c_1b_1)^*(c_2y)(a_1b_1) = (c_1a_1b_1)$, a 3-cycle. As $t = 2$, Lemma 1 implies we have 3-transitivity; hence we can get all 3-cycles and so the alternating group.

ii) (2 2 1):

There is nothing to prove here, since we assume $G$ is not $T_0$. However, it can be shown by exhaustive calculation that in this case $K$ does not contain a 3-cycle.

iii) (2 2 0):

$A = (a_1a_2y)$ is a 3-cycle. $t = 2$ implies by Lemma 1 that $K$ is 3-transitive. So we get all 3-cycles and so the alternating group.

iv) (2 1 1):

$A^{-1}(B^{-1}A^{-1}BA)B = (a_2y)$, a transposition. $t = 2$, so Lemma 1 gives 3-transitivity. Using just 2-transitivity, we get the symmetric group, and a fortiori the alternating group.

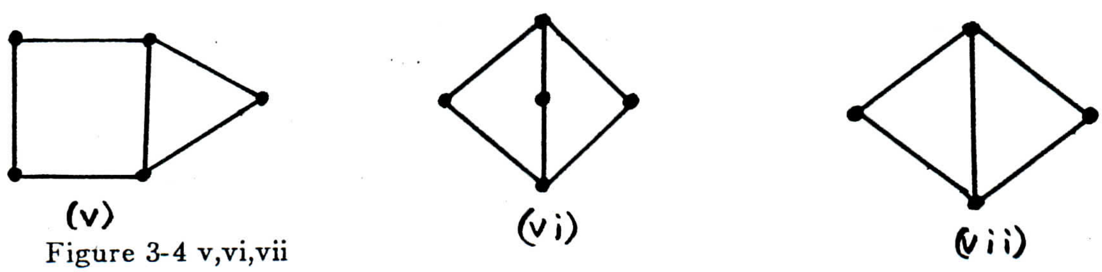


Figure 3-4 v,vi,vii

v) (2 1 0):

$B = (b_1y)$, a transposition. $t = 2$ gives 3-transitivity by Lemma 1. Using just 2-transitivity, we get the symmetric group, and a fortiori the alternating group.

vi) (1 1 1):

$A = (a_1yc_1)$; $B = (yb_1c_1)$; $BA = (a_1yb_1)$; $BAB = (a_1c_1b_1)$. These four 3-cycles and their powers comprise all possible 3-cycles on $\{a_1, b_1, c_1, y\}$, so we have the alternating group.

vii) (1 1 0):

$A = (a_1 y)$ is a transposition. $t = 1$ gives 2-transitivity by Lemma 1; this yields the symmetric group, and a fortiori the alternating group.

Hence Theorem 1 is true for $T_2$-graphs.

### 3.2.2.2. Proof of Theorem 1 for $T_k$-graphs, $k \geq 3$

We will show by induction that biconnected graphs $G$ with $B(G) > 2$ have a 3-cycle and that we have 3-transitivity. This will yield all 3-cycles and so the alternating group. Since we have already proved that the alternating group is generated by $T_2$-graphs other than $T_0$, the theorem will be completely proved.

We will use the result for $T_2$-graphs as a basis for proving the result for $T_3$-graphs. Then we use induction to prove the theorem for all higher graphs. The reason we don't base the induction on $T_2$-graphs is that our basic tool, 3-transitivity, does not hold for all $T_2$-graphs (the exception is $G_1 = (1\ 1\ 1)$).

**Obtaining a 3-cycle**

As we have seen above, all $T_3$-graphs are obtained by adding a "handle" to a $T_2$-graph. Since $T_2$-graphs except $T_0$ yield 3-cycles, a fortiori $T_3$-graphs formed by adding a handle to a $T_2$-graph other than $T_0$ yield 3-cycles (just move tokens within the $T_2$-graph, to get the 3-cycle). Now, it can be seen by adding a handle to a $T_2$-graph in all possible ways, that any $T_3$-graph is a subdivision of one of the four graphs shown in Figure 3-5.

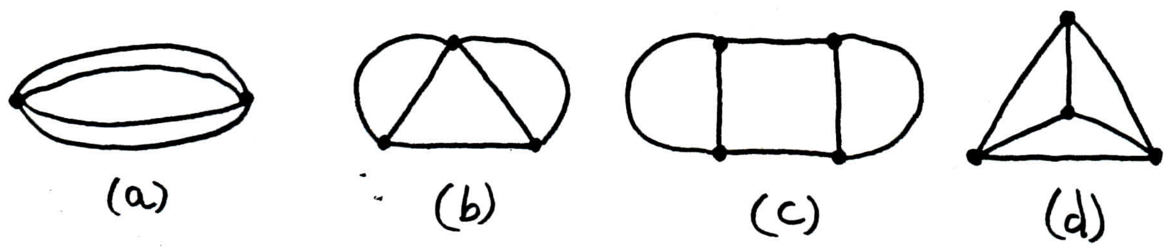



Figure 3-5 a,b,c,d

It can easily be shown by exhaustive case analysis that in each of these four graphs, no matter what the subdivision, we can pick a handle so that the remainder of the graph is a $T_2$-graph other than $T_0$. Hence any $T_3$-graph can be obtained by adding a handle to a $T_2$-graph other than $T_0$. It follows that all $T_3$-graphs yield 3-cycles.

Using the $T_3$-graphs as a basis for induction, it then clearly follows that all biconnected graphs $G$ with $B(G) \geq 3$ yield 3-cycles. For if $B(G) > 3$, $G$ is obtained by adding a handle to a graph $H$ with $B(H) = B(G) - 1$. By the induction hypothesis, $H$ has a 3-cycle; hence a fortiori so does $G$.

**Obtaining 3-transitivity:**

We will use an inductive argument to obtain 3-transitivity for $T_{k+1}$-graphs, given 3-transitivity for $T_k$-graphs. This induction has a hole when $k = 2$, because the graph $G_1 = (1\ 1\ 1)$ is not 3-transitive. However, we will patch this hole.

Induction: suppose that all $T_k$-graphs are 3-transitive. Let $G$ be a $T_{k+1}$-graph. $G$ = a $T_k$-graph $H$ + a handle $A$. There are three cases to consider.

1. The handle has two or more internal nodes. Lemma 1 gives 3-transitivity.

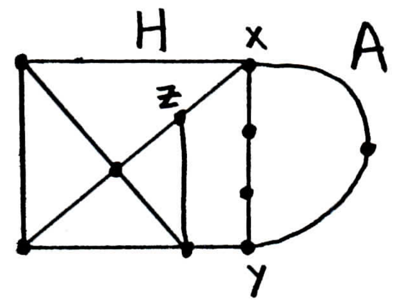



Figure 3-6 One internal node

2. The handle has exactly one internal node (see Figure 3-6). Let $A$ join $H$ at vertices $x$ and $y$. Let $z$ be in $H$, $z$ not $x, y$. We will show that any three pebbles can be moved to x,y,z respectively; 3-transitivity then follows easily. Let the three pebbles start at $a, b, c$ ; call these pebbles the start pebbles. If $a, b$, or $c$ is the internal vertex of $A$, first use transitivity of $H$ to move a non-start pebble to $x$; consider a simple path from $x$ to $y$ in $H$, not passing through $z$ (exists by nonseparability of $H$). Then rotate the loop formed by $A$ plus this path so that the start pebble on the internal vertex of $A$ moves to $y$ (and the nonstart pebble moves onto the internal node of $A$). Now, since all three start pebbles are in $H$, use the 3-transitivity of $H$ to get them to $x, y, z$ respectively.

3. There are no internal nodes on the handle. Use the induction hypothesis to immediately get 3-transitivity.

As we said before, there is a hole in the induction: graph (1 1 1) is not 3-transitive.

We resolve this as follows. A $T_3$-graph is a subdivision of one of the graphs (a)-(d) in Figure 3-5. Let us see whether we can decompose the graph into a $T_2$-graph other than (1 1 1), plus a handle.

Graph (b):

we can avoid (1 1 1) because some handle must have 2 or more internal nodes (otherwise $G$ would not be simple); include this handle in the $T_2$-subgraph.

Graph (c):

we can avoid (1 1 1) because, as in graph (b), some handle will have 2 or more internal nodes; include this handle in the $T_2$-graph.

Graph (d):

we can avoid (1 1 1) because graph (d) itself does not have (1 1 1) as a subgraph, and any refinement of (d) will cause some handle to have at least 2 internal nodes (include this handle in the $T_2$-graph).

Graph (a):

The refinement of graph (a) shown in Figure 3-7 is the only one for which we cannot avoid (1 1 1). Call this graph $G_2$.

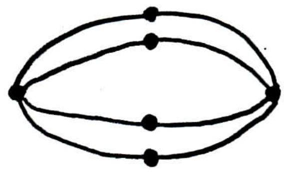


Figure 3-7 Graph $G_2$

Hence the graph $G_2$ is the only $T_3$-graph for which we cannot deduce 3-transitivity by using the above induction argument on $T_2$-graphs. However, upon inspection we see directly that $G_2$ is 3-transitive: pebbles can be moved one at a time to internal nodes of handies, without disturbing pebbles on the internal nodes of the other handles.

Therefore all $T_3$-graphs are 3-transitive; using this as a basis, the induction method proves that all $T_k$-graphs, $k \geq 3$, are 3-transitive.

Combining the 3-cycle plus 3-transitivity, it follows that Theorem 1 holds for all $T_k$-graphs, $k \geq 3$. Since we already proved Theorem 1 for $T_2$-graphs, the proof of Theorem 1 is complete.

### 3.2.3. Proof of Theorem 2

**Sketch of Proof**

First we sketch the proof. We can show that a 3-cycle can always be obtained in $O(n^2)$ moves (either $ABA^{-1}B^{-1}$ gives a 3-cycle in $O(n)$; or we get a product of two swaps, in which case we can do 4-transitivity in $O(n^2)$ moves to get a 3-cycle), and that 3-transitivity requires at most $O(n^2)$ moves. Then by conjugation we obtain any 3-cycle within $O(n^2)$ moves. Since any element of $A_n$ is a product of $O(n)$ 3-cycles, the total for $A_n$ is $O(n^3)$. If the group is $S_n$, then any permutation is a product of an odd permutation and an element of $A_n$. An odd permutation is generated by a closed path of odd length in $O(n)$ moves. Hence $S_n$ also requires at most $O(n^3)$ moves.

**Complete Proof**

**Obtaining a 3-cycle:**

For $T_2$-graphs other than the exceptional cases i)-vii), we had a 3-cycle for $r = 0$ via $ABA^{-1}B^{-1}$. Since $A, B$ are permutations induced by simple closed paths of length $O(n)$, we get a 3-cycle in $O(n)$ and so a fortiori $O(n^2)$ moves. For $r > 0$, we used the pair of transpositions due to $ABA^{-1}B^{-1}$ ($O(n)$ moves), plus 4-transitivity which can be done in $O(n^2)$ moves by Lemma 2 (proved at the end of this section). Hence also for $r > 0$ we get a 3-cycle in $O(n^2)$ moves. Cases i - vii (except $T_0$) give a 3-cycle in $O(1)$ moves. Therefore all $T_2$-graphs aside from $T_0$ give a 3-cycle in $O(n^2)$ moves.

Now any $G$ is either 1) a $T_2$-graph other than $T_0$ or 2) obtained by adding successive handles to a $T_2$-graph not equal to $T_0$. In the first case, we have a 3-cycle in $O(n^2)$ moves. In the second case, let the $T_2$-subgraph have $k \leq n$ vertices. Then as we have a 3-cycle in $O(k^2)$ moves on the subgraph, this is a fortiori a 3-cycle on $G$ in $O(n^2)$ moves.

Hence any $G$ other than $T_0$ can generate a 3-cycle in $O(n^2)$ moves.

**Obtaining 3-transitivity:**

By Lemma 2, we see that for $T_2$-graphs besides i-vii (Figures 3-3 and 3-4) we have 3-transitivity in $O(n^2)$ moves. i-vii besides vi are 3-transitive in $O(1)$ moves. So T2-graphs besides (1 1 1) are 3-transitive in $O(n^2)$ moves.

Then, using the induction method, we get for $T_3$-graphs besides $G_2$:

case 1. $O(n^2)$ 3-transitivity by Lemma 2.

case 2. For some $c_1 > 0$, it takes at most $c_1 n$ moves to get a nonstart pebble to $x$, plus rotate the loop to get the start pebble to $y$. And for some $c_2 > c_1$, we can obtain 3-transitivity in $H$ in at most $c_2(n-1)^2$ moves because $H$ is a $T_2$-graph other than (1 1 1), with $n-1$ vertices. Then a total of at most $c_2 n^2$ moves are involved, which is $O(n^2)$.

case 3. $O(n^2)$ by the induction hypothesis

Hence $T_3$-graphs besides $G_2$ are 3-transitive in $O(n^2)$ moves. But $G_2$ is 3-transitive in $O(1)$ moves, hence all $T_3$-graphs are $O(n^2)$ 3-transitive.

Repeating the induction analysis for all higher graphs, we conclude that all biconnected graphs $G$ with $B(G) \geq 2$, except (1 1 1), are 3-transitive in $O(n^2)$ moves.

Generating the alternating group in $O(n^3)$ moves:

Using the 3-cycle plus 3-transitivity, each in $O(n^2)$ moves, we get that each 3-cycle on is obtainable in $O(n^2)$ moves, except for the graph (1 1 1). But we obtained each of the 3-cycles for this case in $O(1)$ moves. Hence for any $G$, each 3-cycle is obtainable in $O(n^2)$ moves. Since any permutation in the alternating group is expressible as a product of $O(n)$ 3-cycles, at most $O(n^3)$ moves are required altogether.

If we have the full symmetric group: any permutation is a product of an odd permutation and an element of the alternating group. An odd permutation is generated by a closed path of odd length (which exists, because the graph is not bipartite when we have the symmetric group) in $O(n)$ moves. Hence the symmetric group requires at most $O(n^3)$ moves altogether.

This completes the proof of Theorem 2.

We note here that Wilson showed how to generate a 3-cycle, and how to get 2-transitivity. It is well known that this generates all of $A_n$; but we do not know if this gives a polynomial upper bound.

### 3.2.4. Statements and Proofs of the Lemmas

**Lemma 1**

Let $H$ be a nonseparable graph, and $G$ be the result of adding a handle to $H$. If the handle has $k$ internal nodes, then $G$ is $k+1$-transitive.

**Proof**

Let the handle (call it $A$) join $H$ at vertices $x$ and $y$. Let the vertices of the path from $x$ to $y$ along $A$ be, in order, $x, a_1, a_2, ..., a_k, y$. Let $t \in V(H)$, $t \neq x, y$. Now as $H$ is nonseparable, there is a simple path $p$ from $x$ to $y \in H$ which does not pass through $t$. Let the simple closed path $[xa_1a_2...a_ky]p^{-1}$ ($p^{-1}$ is the reverse of path p) from $x$ to $x$ be called $L$.

We now indicate how to move any $k+1$ pebbles to respective positions $a_1, ..., a_k, x$. This will imply $k+1$-transitivity, since to get pebbles from vertices $b_1, ..., b_{k+1}$ to $c_1, ..., c_{k+1}$ resp., first move $b_1, ..., b_{k+1}$ to $a_1, ..., a_k, x$, and then do the inverse of any permutation that takes $c_1, ..., c_{k+1}$ to $a_1, ..., a_k, x$.

First we claim that $G$ is transitive. To prove this, we use induction on the Betti number of a graph. $T_2$-graphs are easily seen to be transitive, by rotating one or other of the loops to get the pebble to where one wants it to go. Induction: If $B(G) > 2$, look at the last handle added to obtain $G$. Let it join nonseparable subgraph $H$ at $x$ and $y$. To move pebble to $x$: if pebble is in $H$, just use the inductive hypothesis. If pebble is not in $H$, rotate the loop consisting of the handle and a simple path from $x$ to $y$ in $H$, until the pebble on the handle moves to $x$. This completes the proof of the claim.

Now to show $k+1$-transitivity, we show that we can move $i$ pebbles to $a_{i-1}, ..., a_1$, $x$ respectively, $1 \leq i \leq k+1$. We have 1-transitivity by the claim above. Induction: suppose that we have $i$-transitivity. To get $i+1$-transitivity: move the first $i$ pebbles to targets $a_{i-1}, ..., a_1$, $x$ resp. Suppose the next pebble is not on $A$. Then rotate $L$ so that the $i$ pebbles are at $a_i, ..., a_1$ respectively. Use 1-transitivity of $H$ graph to get the next pebble to $x$. On the other hand, if the next pebble is on $A$, then rotate $L$ until the pebble is at $y$. Then use transitivity of $H$ to get the pebble to $t$ (this doesn't disturb $A$'s internal nodes). Then rotate $L$ back, so that the first $i$ pebbles are at $a_i, ..., a_1$ respectively (this does not disturb the pebble at $t$, since p does not pass through $t$). Finally, move the pebble at $t$ to $x$, using transitivity of $H$. This completes the inductive step, and the Lemma is proved.

**Lemma 2**

For any bounded $k$, the $k+1$-transitivity guaranteed by Lemma 1 can be done in $O(n^2)$ moves.

Proof

We basically make estimates on the number of moves used at each stage of the proof of Lemma 1.

First we show $O(n^2)$ for 1-transitivity, by induction on the number of loops in $G$. $T_2$-graphs are 1-transitive in $O(n^2)$, by rotating loops. Induction: if $B(G) > 2$, look at last handle $L$ added. Say $L$ has $l$ internal nodes. To move pebble to $x$: if not on $L$, use the induction hypothesis. If the pebble is on $L$, use $O(l^2)$ moves to rotate $L$ to bring the pebble to $x$. This is $O(n^2)$ a fortiori.

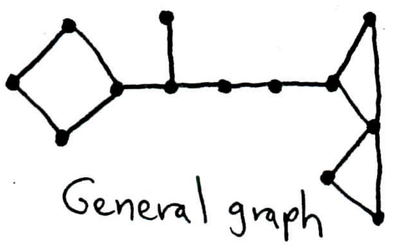
Higher transitivity: each successive pebble moved into place takes, in case it is not on $A$: rotation of $L$ ($O(n^2)$), then 1-transitivity ($O(n^2)$). This is a total of $O(n^2)$. In case the pebble is on $A$: let $c > 1$ be such that 1-transitivity can be accomplished in at most $cn^2$ moves. Then rotate $L$ in at most $l^2$, move pebble to $t$ in at most $c(n-l)^2$, rotate $L$ back in at most $l^2$, and finally move pebble from $t$ to $x$ in at most $c(n-1)^2$ moves. This totals to less than $2cn^2$, so we have $O(n^2)$ in this case also. Hence to move $k$ pebbles takes $k*O(n^2)$, which is $O(n^2)$ for $k$ bounded. This completes the proof of Lemma 2.

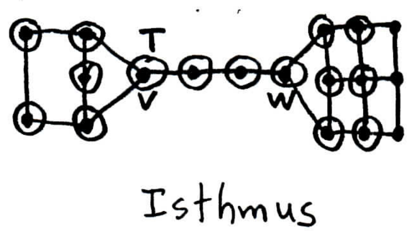
### 3.3.1. Introduction

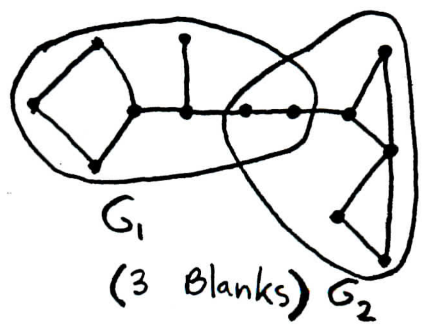
We now consider all cases of the graph puzzle not covered by the Wilson case, and will combine our results with those just obtained.

We begin with an informal discussion to motivate the rigorous analysis which follows.

The basic element which distinguishes separable graphs from biconnected graphs is the existence of isthmuses (of length $\geq 0$), which if severed will separate the graph. One can think of a separable graph as being a tree, or a tree structure connecting one or more biconnected graphs (see Figure 3-8 for an example).



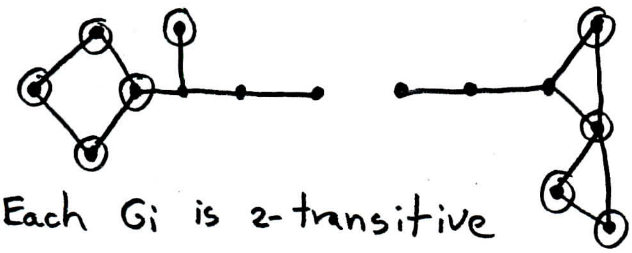



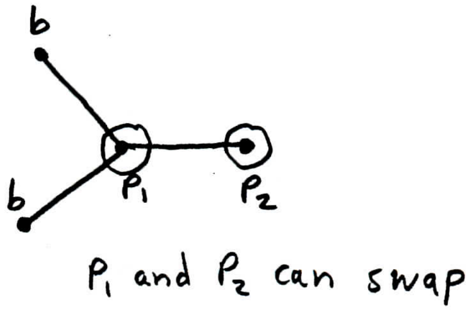

Figures 3-8, 3-9, and 3-10

Much of what follows is motivated by the example shown in Figure 3-9. This graph consists of a simple nonclosed path of length $m$ ($= 3$ in the figure) which connects subgraphs $A$ and $B$. Suppose we wish to move pebble $T$ from $v$ to $w$. Since $A$ has no blank vertices, it is clear that $T$ can reach $w$ if and only if $B$ has $m$ or more blank vertices. Therefore, the number of blanks has a direct effect on the ability of pebbles to cross isthmuses. Conversely, the lengths of the isthmuses will determine whether or not certain pebbles can cross from one component into another.

It turns out that we can naturally divide a graph $G$ in this way into subgraphs $G_i$ connected by isthmuses, with the property that pebbles can move freely within each $G_i$ but cannot leave $G_i$. Each pebble in its initial position is assigned in a natural way to the $G_i$ (if any) to which it is confined, otherwise it is confined to an isthmus. This decomposition induces subpuzzles on the $G_i$'s and their pebbles, and it will be shown that the original puzzle is solvable iff all the subpuzzles are solvable and the tokens trapped on isthmuses do not change order. Figure 3-10 shows the $G_i$ subgraphs for the graph of Figure 3-8 with 3 blanks (exactly how the $G_i$'s are determined will be explained below).

The final step in the analysis is to study solvability of subpuzzles on the $G_i$. When there is one blank, the $G_i$'s turn out to be biconnected and so the Wilson criterion applies. We will show that when there are two or more blanks, the $G_i$ subpuzzles are always solvable (subject to the condition that the $G_i$ contains the same pebble set before and afterwards). Informally, one reason is that the $G_i$'s were defined in such a way that pebbles can cross all isthmuses in $G_i$, and so get from any vertex to any other vertex (see Figure 3-11 for an illustration); we will show how to achieve 2-transitivity in this way, by moving one pebble after another to its destination. The other reason is that two blanks are sufficient to achieve a swap of a pair of pebbles near a vertex of valence three (see Figure 3-12). Combining 2-transitivity and the swap yields all permutations of the pebbles.




Figures 3-11 and 3-12

## The General Criterion

Here is an outline of the general solvability criterion for graph puzzles, which combines the Wilson case of the previous section with the results for the remaining cases considered in this section. Details of how to determine the subpuzzles and the pebbles confined to isthmuses will be given later.

Let $G$ be a graph with $k$ tokens and $m = n - k$ blanks. Let $f, g$ be the starting and ending positions. Move blanks in $f$ to form a labeling $f'$ whose blanks are in the same locations as in $g$. Then $f$ and $g$ are mutually reachable iff $f'$ and $g$ are mutually reachable.

So without loss of generality assume that $f$ and $g$ have blanks in the same places.

If $G$ is nonsimple, remove extra edges. This will neither hurt nor help solvability.

If $G$ is not connected, check that the token partition induced naturally by the connected subgraphs is consistent, and that each connected subgraph puzzle is solvable.

If $G$ is connected:

> If $G$ is nonseparable: if $m = 1$ use the criterion in Theorem 1; if $m > 1$, then the puzzle is solvable, unless $G$ is a polygon, in which case only cyclic rearrangements are possible.
>
> Otherwise, determine the subpuzzles and check that the pebble sets in each subpuzzle match before and after. Also determine the pebbles confined to isthmuses, and check that they are the same before and after, and in the same order. If all this is satisfied, then if $m = 1$ check that each subpuzzle is solvable (for $m = 1$, all components are nonseparable, so use Wilson's criterion). If $m > 1$, then each subpuzzle will be solvable, so the puzzle is solvable in this case.

This completes the criterion.

It will be easy to show based on the analysis of case 1, that solutions with at most $O(n^3)$ moves exist and can be efficiently planned. In the final section of this chapter, we construct an infinite family of graph puzzles which are proved to require $O(n^3)$ moves for solution.

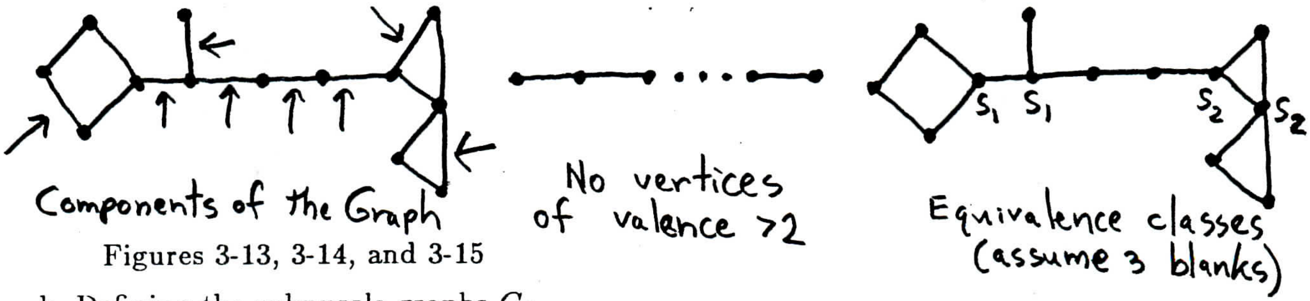
### 3.3.2. Details of the General Decision Algorithm and the Proofs

Suppose $G$ is a simple connected graph with $n$ vertices, and $k < n$ pebbles on distinct vertices. $m = n - k$ vertices will be unoccupied ("blank").

a. Dividing a graph into maximal biconnected components

First assume $G$ is a connected simple graph. We will divide $G$ into its maximal biconnected components as follows: Say that edges $e_1$ and $e_2$ of $G$ are "equivalent" iff $e_1 = e_2$ or there is a simple closed path in $G$ containing $e_1$ and $e_2$. This is seen to be an equivalence relation; the equivalence classes, together with incident vertices, are the maximal biconnected components ("components" for short) of $G$.

We will call components with one edge "trivial"; otherwise they are called "nontrivial". See Figure 3-13 for the components of the graph of Figure 3-8.




Figures 3-13, 3-14, and 3-15

b. Defining the subpuzzle graphs $G_i$

1) If the number of blanks $m$ is 1: define the $G_i$'s to be the nontrivial components of $G$ in some order.

2) If there are $m > 1$ blanks:

If there is only one component, and it is nontrivial, then assign $G_1 = G$. If there is only one component, and it is trivial, then assign no subpuzzle graphs $G_i$.

If no vertex has valence $> 2$, then we have the graph pictured in Figure 3-14. Assign no subpuzzle graphs $G_i$.

Otherwise, consider the set $S$ of vertices with valence $> 2$ which are common to more than one component (i.e. a join of valence $> 2$ between two or more components). $S$ is nonempty: For, if all components are trivial, then any vertex with valence $> 2$ is a join of components; if there is a nontrivial component, any vertex which joins with the rest of the graph has valence $> 2$ and joins components.

Say that elements $s_1, s_2$ of $S$ are equivalent iff $s_1 = s_2$ or $s_1$ and $s_2$ are in the same nontrivial component, or there is a unique path between $s_1$ and $s_2$, and its length is $\leq m - 2$. The transitive closure of this relation is an equivalence relation (which roughly describes reachability of one "critical" vertex from another by a pebble). Let the equivalence classes be $S_1, \ldots, S_l$. Figure 3-15 shows the equivalence classes.

For each $1 \leq i \leq l$ let $X_i$ be the set of $v \in G$ such that $v$ is in the same nontrivial component as some element of $S_i$ or there is an element $s_i$ of $S_i$ such that there is a unique path from $v$ to $s_i$, and its length is $\leq m - 1$. Let $G_i$ be the subgraph of

$G$ consisting of vertices $X_i$ and edges incident only on these vertices. Remark: $X_i$ contains $S_i$, and may be thought of as the "completion" of $S_i$ to include all points of the subgraph $G_i$. Figure 3-10 shows the $G_i$ subgraphs for the graph of Figure 3-8.

c. A look at how the $G_i$ are interconnected

Suppose it turns out that the $G_i$'s number more than one. We will now see that if $G_i$ and $G_j$ are joined, it is by means of a simple nonclosed path. This is done by analyzing the connections of $G_i$ with the rest of the graph.

Let $w$ be a vertex not in $G_i$, but adjacent to a vertex $v$ in $G_i$.

If $v \in S_i$, then as $m \geq 2$ we have $w \in G_i$ (For if the path from $v$ to $w$ is not unique, then $v$ and $w$ are in the same nonseparable component. If the path is unique, then as it has length 1 and $m \geq 2$, we have by definition that $w \in G_i$). This is a contradiction!

So $v$ is not in $S_i$. If $v$ has valence $> 2$, then $v$ not in $S_i$ implies $v$ is not a join between components, so all edges incident to $v$ must be in the same nontrivial component. So $v$ gained membership in $X_i$ by being in the same nontrivial component as some $s_i \in S_i$. (For if there was a unique path from some $s_i$ to $v$, then the vertex $x$ on the path just before $v$ would also have a unique path to $v$. But this contradicts the fact that $x$ and $v$ are in the same nontrivial biconnected component.) Hence $w$, being in the same nontrivial component with $v$, is in the same nontrivial component with $s_i$, so $w \in G_i$. This is a contradiction!

So $v$ has valence $\leq 2$. $v$ cannot have valence 1, because then $G_i$ is a single vertex of valence 1; by the definition of $G_i$, some vertex must have valence $> 2$.

Hence $v$ has valence 2.

The "Plank"

Now, $v$ cannot be in a nontrivial component. (For if $v$ was, then there would be a simple closed loop through $v$, and the neighbor $w$ would have to be in the loop also. $v$ must have gained membership in $X_i$ by being in the same nontrivial component as some $s_i$, because in a simple closed loop, paths are not unique. So $w$ is in the component with $s_i$, implying $w \in X_i$. This is a contradiction.)

Hence $w$ must be one step off a "plank", that is, a simple nonclosed path of length at least 1, none of whose edges belong to a loop. A plank is part of all of an isthmus; we use the term to avoid confusion with isthmuses. The internal vertices of a plank have valence 2.

Length of the Plank

Going from $w$ through $v$ and along the plank until we reach a vertex $s$ of valence $> 2$ ($G_i$ has at least one such vertex), $s$ must be in $S_i$ since the plank is not in the same component as other edges out of $s$. The distance from $v$ to $s$ must be exactly $m-1$. (For there is only one path from $s$ to $w$. If $s$ to $v$ is less than $m-1$, then by definition $w \in G_i$, a contradiction. If $s$ to $v$ is more than $m-1$, then $v$ could not have gained membership in $X_i$ via a path from $s$, so $v$ must have gained membership via a unique path from an $s_i$ on the $w$ side. But then $w$ was also approached uniquely from $s_i$ and was closer than $v$, so $w \in X_i$, a contradiction.)

Summing up, we can say that the "exits" from $G_i$, i.e. the vertices of $G_i$ which are adjacent to vertices not in $G_i$, are precisely the ends of $m-1$ long planks which begin with a $s_i \in S_i$. The vertices adjacent to the ends of these planks are the connections of $G_i$ with the rest of the graph. Note that therefore the $G_i$'s connect with each other via their planks, and it is possible for the $G_i$'s to overlap on part or all of the isthmus joining them (see Figure 3-16). Points not in any $G_i$ must reside on a plank, because they are reached by leaving a $G_i$.

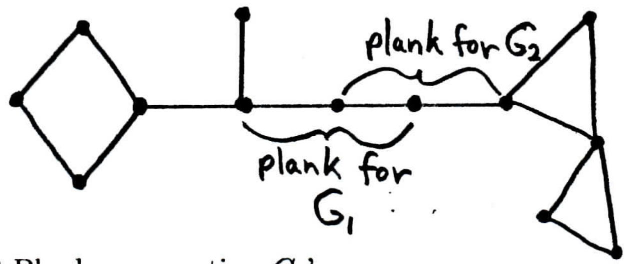



Figure 3-16 Planks connecting $G_i$'s

d. Assigning pebbles to the confining $G_i$ or plank

Let $P$ be a pebble in the start position, to be assigned to the appropriate part of $G$.

1) If $P$ is not on a plank, then $P$ is in some $G_i$ and is confined to $G_i$ (although $P$ may be able to visit the intersection with another $G_i$). For to leave $G_i$, $P$ has to get onto the inside end (i.e. the end attached to a $s_i \in G_i$) of a plank, "walk the plank", and "jump off". To get to the inside end requires at least one blank, which is left behind when $P$ gets onto this end. Now $m$ additional blanks are required for $P$ to walk the plank and exit (since the plank is $m-1$ long). But since there are only m blanks altogether, this cannot be done.

2) If $P$ is on a plank:

a) If enough blanks are on the subgraph to one side of $P$ to take $P$ off the plank in that direction, then $P$ would be trapped in the $G_i$ which it enters, by 1).

We define the set $U_i$ to be the set of all pebbles in the initial position of the puzzle which are confined to $G_i$. $U_i$ contains pebbles satisfying condition 1) or 2a).

b) If $P$ cannot leave the plank as in a), then $P$ is clearly confined to the plank. Furthermore, $P$ is not confined to any $G_i$. (For if $P$ is a distance $r \geq 0$ from one end of the plank, and cannot leave the plank from that end, then there must be at most $r$ blanks in the subgraph to that side of $P$. So there are at least $m-r$ blanks on the other side of $P$, which is enough to carry $P$ the distance $m-1-r$ and one further, to leave that $G_i$. Similarly $P$ can leave the $G_i$ on the other end of the plank.)

For a pebble $P$ confined to a plank, we observe which vertex $v_j$ $P$ was on in the start position, and define a set $V_j = \{P\}$. For those plank vertices $v_i$ which are initially blank, or whose initial pebble is not confined to the plank, we define the set $V_i$ to be the empty set.

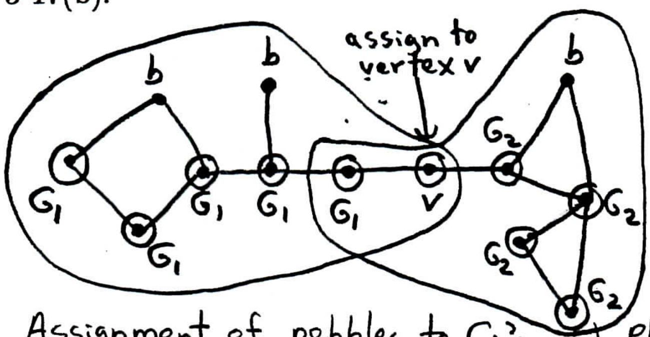
3) Observation: a pebble $P$ cannot be confined simultaneously to two $G_i$'s (remember that the $G_i$'s can partially overlap on a plank). For then $P$ would be

trapped on the intersection of the two $G_i$'s, which is part or all of the plank joining them. So $P$ is confined to the plank, which by 2b) implies that $P$ belongs to no $G_i$, a contradiction.

In summary, nonplank pebbles are confined to exactly one $G_i$. Plank tokens are either confined to exactly one $G_i$, or else they are confined to the plank and are confined to no $G_i$. Figure 3-17(a) shows the assignment of pebbles in a typical start position to $G_i$'s and to plank vertices.

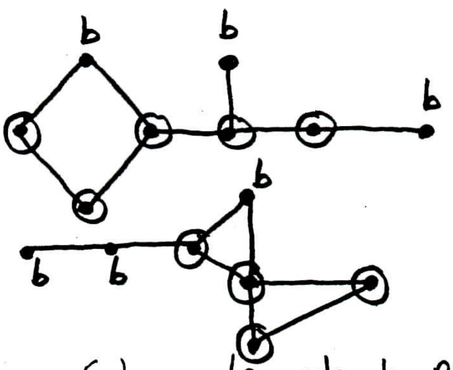
Assume for the moment that the pebbles in the end position are the same set as the pebbles in the start position, and that they reside on the same set of vertices as at the start (but probably in a different order). Then we define sets $U'_i$ for each $G_i$ and $V'_j$ for each plank vertex $v_j$ for the end position, in the same way that we defined the $U_i$'s and $V_j$'s for the start position. We will show how to easily handle the case where the assumption is not true.

A "subpuzzle" consists of a starting position on $G_i$ and an end position on $G_i$. The start position consists of $G_i$, with pebble $P$ on vertex $v_j$ iff $P$ is on $v_j$ at start of the whole puzzle on $G$, and $P$ is in $U_i$. Another way to say this, is that in the start position, $G_i$ has as pebbles those which reside on $G_i$ at the start of the whole puzzle, and which are also confined to $G_i$. Those vertices in $G_i$ which are occupied by pebbles at the start of the whole puzzle, but whose pebbles can leave $G_i$, are considered blank at the start of the subpuzzle. Similarly we define the end position for the subpuzzle on $G_i$. The subpuzzle consists of $G_i$ and its start and end positions; the questions on the subpuzzle are whether the subpuzzle as defined has a solution, and in how many moves if possible. These questions are asked, now that the subpuzzle has been defined, as if $G_i$ were the entire playing space; the relation of $G_i$ to $G$ is ignored for the moment. The subpuzzle start positions for the pebbling in Figure 3-17(a) are shown in Figure 3-17(b).




Assignment of pebbles to $G_i$'s and Plank

(a)




Subpuzzle start positions for pebbling in 3-17(a)

(b)

Figure 3-17(a),(b)

We now present the complete decision algorithm for the pebble coordination problem on general graphs.

**Theorem 3**

Let $G$ be a graph with $n$ vertices, with $k < n$ pebbles numbered $i_1, ..., i_k$ on distinct vertices $v_1, ..., v_k$ respectively. To decide whether a legal sequence of moves will result in pebbles $j_1, ..., j_l$ on distinct vertices $w_1, ..., w_l$ respectively, perform the following algorithm:

1. If $G$ is not simple, remove the excess edges from $G$ to get $G'$, and test solvability of the puzzle on the simple graph $G'$. The original puzzle is solvable iff the new one is solvable.

2. If $G$ is not connected, check that the subpuzzles induced in the obvious way on the connected components of $G$ are solvable. The puzzle is solvable iff the subpuzzles are all solvable.

3. Check that $k = l$ and that the sets $\{i_1, ..., i_k\}, \{j_1, ..., j_l\}$ are equal. If not, output "not solvable".

4. If the sets $\{v_1, ..., v_k\}, \{w_1, ..., w_k\}$ are not equal, then move the final position to a position where $i_1, ..., i_k$ are on $v_1, ..., v_k$ in some order. The modified final position has the same vertices occupied as the starting position. The puzzle is solvable iff the modified puzzle is solvable.

5. Compute the $G_i$'s and $U_i$'s to get the subpuzzles, and the $V_j$'s.

a. If there are not any $G_i$'s, then we have the graph pictured in Figure 3-14. The puzzle is solvable iff $V_j = V'_j$ for each $j$ (this is equivalent to requiring that the start and end position be identical).

b. If there are subpuzzles, check whether each subpuzzle is solvable. The puzzle is solvable iff the subpuzzles are all solvable and $V_j = V'_j$ for each $j$.

6. To test the solvability of a subpuzzle:

a. Perform step 3. If it passes, then continue to b,c. Otherwise the subpuzzle is not solvable.

b. If $m = n - k$ is 1, then all $G_i$'s will be nontrivial biconnected components of $G$. Use the Wilson criterion to decide solvability.

c. If $m = n - k$ is $>1$: the subpuzzle is solvable, unless the subpuzzle graph is a polygon (this can only happen if the whole graph $G$ is a polygon). In this case, the subpuzzle is solvable iff the start and end positions differ only by a cyclic permutation.

7. This completes the algorithm.

### 3.3.3. Proof of the Main Theorem

Most of the above steps are almost trivial to prove; 5b and 6c are more substantial.

1\. : the removal of extra edges clearly does not help or hinder the solution of the puzzle. Therefore this step is valid.

2\. : if $G$ is not connected, then pebbles cannot travel from one connected component to another. Therefore the puzzle is solvable iff the natural projections of the puzzle onto the connected components are all solvable subpuzzles.

3\. : this merely checks that the set of pebbles on the graph does not change. Clearly this is necessary for solvability.

4\. : this step has the purpose of putting the puzzle into the form assumed earlier in this chapter. Justification was given then.

5a. : if there are no $G_i$'s, then we get the graph of Figure 3-14, as discussed earlier. It is clear that no rearrangements are possible on this graph, hence the procedure of checking $V_j = V'_j$ gives the correct decision.

5b. : this is one of the essential points of the algorithm. It states that the puzzle is solvable iff the $V_j$'s match and the subpuzzles are all solvable. This is the justification for the careful definition of the subpuzzles. The proof will be given below.

6a. : this is clearly needed, for the same reason as for step 3.

6b. : it is easy to show that the $G_i$'s are nontrivial biconnected graphs when $m = 1$. Then Wilson's criterion clearly applies.

6c. : this is the other essential point of the algorithm, and gives justification for defining the subpuzzles in the way that they were defined. The proof will be given below.

### 3.3.3.1. Proof of 5b

If the puzzle is solvable, then solve the puzzle and watch what happens in the various $G_i$'s. Sets $U_i$ must remain in $G_i$, with perhaps also some pebbles visiting and leaving $G_i$. When the solution is complete, each subpuzzle is in its end position. Clearly if the subpuzzle can be solved even with extra visitors around, then the subpuzzle can be solved in its isolated form. Hence the subpuzzles are all solvable. Also note that the pebbles confined to planks cannot change their ordering. Hence $V_j = V'_j$ for each $j$.

The converse is a bit less trivial. Suppose $V_j = V'_j$ for each $j$, and each subpuzzle is solvable. We will solve the puzzle by solving each subpuzzle in turn, without disturbing the other subpuzzles. The pebbles confined to planks cannot change in their ordering, and so all parts of the puzzle will match the final position.

To solve a subpuzzle on $G_i$ without disturbing another subpuzzle: Note that the pebbles on $G_i$ which can leave $G_i$ reside on planks, and reside closer to the ends of the planks than any plank tokens which cannot leave $G_i$. Also note, because of the overall tree structure of $G$, that removing pebbles from one plank of $G_i$ does not effect the ability to remove pebbles from any other plank of $G_i$ (by using blanks outside $G_i$). Hence we can remove all pebbles in $G_i$ which are not in $U_i$, by moving blanks from outside $G_i$ onto the planks of $G_i$; and this does not disturb the other pebbles of $G_i$. Remember the sequence of moves which accomplishes this. Now the configuration on $G_i$ looks exactly like the start position of the subpuzzle on $G_i$. For use in the proof of 6c, note that a subpuzzle always has exactly $m$ blanks. (For it follows by the definition of an escapable plank pebble of $G_i$, that if all pebbles on a plank cannot leave, then all blanks in the part of $G$ off that plank had to be used to remove the plank pebbles that could leave. Hence either all blanks outside of $G_i$ enter $G_i$, or some plank of $G_i$ becomes devoid of pebbles (i.e. $m$ blanks). In either case, the subpuzzle has all $m$ blanks in it.) Solve the subpuzzle on $G_i$ (assumed to be solvable), then reverse the above memorized sequence of moves to return escaped tokens to their planks and otherwise restore any change to the configuration outside of $G_i$. In this way, we can solve each subpuzzle in turn without disturbing the rest of the puzzle, and so get each $U_i$ set in the desired order. With all $U_i$'s in order and the $V_j$'s matching, the entire puzzle is solved.

3.3.3.2. Proof of 6c

We now prove that when there are $m > 2$ blanks, then each subpuzzle is solvable. First note by the remark in the proof of 5b, that each subpuzzle has exactly $m$ blanks in it.

The result will be proved in stages. First we prove it for the case that $G_i$ is a tree having exactly one vertex of valence $> 2$ (called a "radiating tree" because "free branches", i.e. paths with all internal vertices of valence 2, and an end vertex of valence 1, radiate out from a central vertex; see Figure 3-18). Then the result is extended to general trees. Finally the most general case, where $G_i$ is a tree structure connecting nontrivial biconnected components, is handled.

A. Radiating Trees

We will show how to get a swap, and 2-transitivity. This gives all possible orderings of the pebbles.

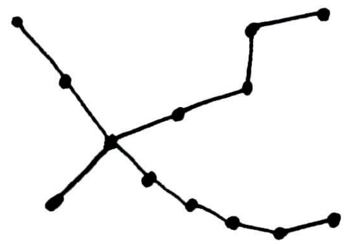


Figure 3-18 A "radiating tree"

1. Swap

"Cram" pebbles as far as possible to the ends of the branches. Move the pebble (if any) onto any branch, crammed against the other branch pebbles. Remember the sequence of moves thus far. Now the central vertex and at least one adjacent vertex are blank (since $m \geq 2$); if at least two branches are nonempty, we can perform a swap with the nearest pebble of one branch and the nearest pebble of the other. If only one branch is nonempty, we can swap the two pebbles of that branch which are nearest the central vertex. Then reverse the memorized sequence of moves. The net result is a swap.

2. 2-transitivity

As in 1., cram the pebbles to the ends of the branches. We will move any two pebbles $P_1$,$P_2$ to two furthermost occupied branch ends, in order. By a remark in Chapter 2 on permutations, this suffices to get 2-transitivity.

Move $P_1$: First get $P_1$ from its current branch onto the nearest (to center) vertex on another branch (this is possible, because there are enough spaces to clear off the branch on which $P_1$ resides). Now move all pebbles off the branch with the first target vertex. If this cannot be done, it is only because $P_1$'s current branch contains some blanks. In this case, move $P_1$ to the nearest vertex on another branch, if one such vertex is blank (if not, then swap $P_1$ with a pebble on the nearest vertex of another branch), fill in the blanks in the branch $P_1$ just left, using pebbles from the destination branch, then return $P_1$ to the branch it just left. In any case, we can now vacate the first destination branch; do so, and move $P_1$ to the destination vertex at the end of this blank branch.

Move $P_2$: Prune the first destination vertex, along with $P_1$, off of $G_i$, to get a new $G_i$. Now move $P_2$ to its destination, just as done for $P_1$. The only possible problem occurs in the special case where the pruned vertex was on a branch of length 1, and the central vertex had valence 3. Then the resulting tree no longer has a center of valence $> 2$. However, this cannot happen since this situation implies that the furthest occupied branch end was on a branch of length 1; then the original radiating tree must have had three branches, two of length 1 with pebbles on their ends. But then the third branch must be blank because it contains all $m$ blanks. Hence the two pebbles on the short branches must be $P_1$ and $P_2$; either they are on their destination vertices, or swapping $P_1$ and $P_2$ will achieve this. Hence 2-transitivity holds in this special case also.

Finally, restore the blanks to their positions when the branches were "crammed"; this can clearly be done without moving $P_1$, $P_2$ from their destinations. Then reverse the sequence of moves which crammed the branches. The net result is a desired 2-transitive permutation on the pebbles.

B. General Trees

In this case, more than one vertex has valence $> 1$. We can think of the general tree as being made up of radiating trees connected by isthmuses. An "end" radiating tree is one which is connected directly to only one other radiating tree. Clearly a general tree contains at least one end radiating tree; otherwise it would contain a cycle. Define a "free branch" to be a path with one end of valence 1, inner vertices of valence 2, and the other end of valence $> 2$; define a "free end" to be the vertex of valence 1 on a free branch. See Figure 3-19 for examples of these objects.

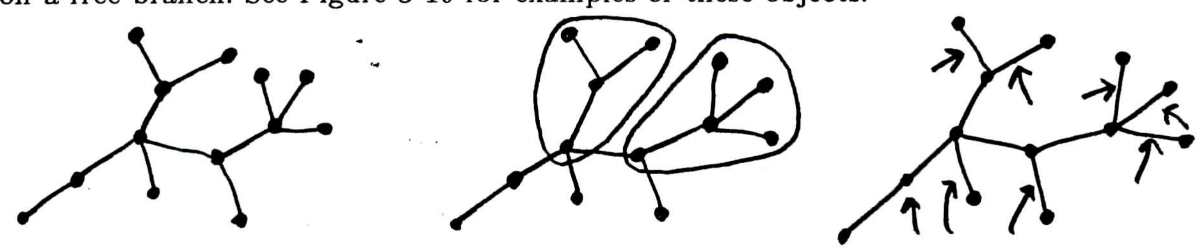


Figure 3-19 General tree; end radiating trees; free branches

1. Swap

Move blanks so as to vacate the nearest vertices (to the central vertex) of two free branches of an end radiating tree. Remember the move sequence which accomplished this. Now we can swap the two pebbles in $G_i$ nearest the central vertex of this radiating tree, by moving them to this central vertex, swapping, and returning them. Finally, reverse the memorized sequence of moves; the net result is a swap.

2. 2-transitivity

We will show how to move any two pebbles $P_1$, $P_2$ to the spots currently occupied by any two pebbles $P_3$, $P_4$ respectively. First we describe in general terms how this will be carried out. Select any end radiating tree, and any two free branches. We will move any pebble $P_1$ to the end of the longer free branch (if lengths are unequal), then any pebble $P_2$ to the end of the other free branch. Call the position we reach the "intermediate position". Similarly, we could have instead moved $P_3$ and $P_4$ to these two free ends, and then moved the blanks (without disturbing $P_3$ and $P_4$) so that the same vertices are occupied as in the intermediate position. Then the following procedure produces the desired 2-transitive permutation: Move $P_1$, $P_2$ to the two free ends (intermediate position). Then reverse the sequence of moves which takes $P_3$, $P_4$ to the same two free ends and puts the blanks at the same vertices as in the intermediate position.

Now we must show how to move $P_1$, $P_2$ to the above-mentioned free ends. We will first move $P_1$ to its destination, prune the destination vertex off of $G_i$, then move $P_2$ in the same way. (The only situation to watch for, is when the pruning reduces the valence of the center vertex (of the end radiating tree) to 2. However, in this case all free branches were of length 1, so the free branch created by this operation is 1 longer than than the isthmus connecting to this end tree, or at most length $m-1$, so that its end is reachable; hence the pruned tree preserves reachability of all its vertices.)

To move $P_1$, we will move it from radiating tree to radiating tree, until we reach the target radiating tree. This is done as follows: move $P_1$ to the nearest vertex to the center, of an isthmus besides the one connecting to the neighbor we want to reach. Clear all pebbles off this connecting isthmus and one vertex farther, so $P_1$ can now be moved to a nearest-to-center vertex of an isthmus (other than this connecting isthmus) of the neighbor. Now we can clear the connecting isthmus to the next neighbor radiating tree, and continue to advance $P_1$ towards its goal. (If there are not enough blanks to clear the isthmus, then $P_1$'s isthmus must have some blanks. We can move $P_1$ aside momentarily, fill some of these blanks, then put $P_1$ back; now the connecting isthmus can be cleared). Once $P_1$ reaches the target radiating tree, we can move enough blanks to this tree so that $P_1$ can be moved to the target vertex (by case A. : radiating trees).

Similarly, after pruning, we can move $P_2$ into place.

C. General $G_i$

This is the most general form of a subpuzzle graph, and consists of a tree structure connecting one or more nontrivial biconnected components. An example was shown in Figure 3-8. The procedure for obtaining a swap and 2-transitivity is almost the same as for trees, with a few modifications to account for the biconnected component(s).

1. Swap

If there is an end radiating tree, perform the same swap as done in the tree case. Otherwise, pick any end biconnected component and vacate a vertex which joins to the rest of the graph, and an adjacent vertex. Then we can perform a swap in the same way as done for the tree case.

2. 2-transitivity

If there is an end radiating tree, then we use the tree case to move $P_1$, $P_2$ to furthermost free ends, followed by the inverse of the moves which would take $P_3$, $P_4$ there. The only modification is that, along the way, the pebble we are moving may pass from a radiating tree into a biconnected component. But we saw in the Wilson case that biconnected components are transitive, so the pebble can move through the component to the next radiating tree, and so on.

If there is no end radiating tree, then select an end biconnected component $B$. We will move $P_1$ to a vertex of $B$ farthest from the junction by which $P_1$ entered, and $P_2$ to the next most distant vertex (distance here means length of shortest path). Then we perform the inverse of the sequence of moves which take $P_3$ and $P_4$ to these two locations (and check correspondence of blanks). Because $P_1$, $P_2$ are most distant from the junction, moving $P_3$, $P_4$ need not disturb them. First move $P_1$ to a vertex of $B$, just past the junction with the rest of the graph. Then move $P_2$ to the junction vertex, after making sure $B$ contains at least one blank. Then use 2-transitivity of $B$ to get $P_1$, $P_2$ to the two most distant vertices just described. (The only exception to this is when $B$ is a polygon. Now $B$ is not 2-transitive. However, in this case, once $P_1$, $P_2$ are on the junction and one vertex away from the junction, we can cycle the pebbles around B until $P_1$, $P_2$ reach the desired most distant vertices.) Similarly we can move $P_3$, $P_4$ to these locations, and so obtain 2-transitivity.

This completes the demonstration of case C, and thus 6c is entirely proved. Hence the main theorem has been proved.

### 3.3.4. $O(n^3)$ Upper Bound

Another main result is the following upper bound on the number of moves required to solve a graph puzzle. The proof is relatively short, because the main work was establishing this result for the Wilson case.

**Theorem 4**

If a pebble coordination problem on a graph $G$ of $n$ vertices has a solution, then there is a solution requiring at most $O(n^3)$ moves, and it can be efficiently planned.

**Proof**

We solve the puzzle by solving each subpuzzle. To solve a subpuzzle, we (as in the proof of 5b of the main theorem) "set up" the subpuzzle by removing pebbles from $G_i$ which are not in $U_i$, actually solve the subpuzzle, then return the removed pebbles.

The pebbles were removed by moving in blanks; by the Lemma at the beginning of this chapter, this takes $O(n^2)$ moves. Since there are clearly at most $O(n)$ subpuzzles, the total number of moves for removing and restoring pebbles is at most $O(n^3)$.

This leaves us the task of proving that the subpuzzle solutions themselves total at most $O(n^3)$ moves. We will first show that if a subpuzzle graph has $n_i$ vertices, then it can be solved in $O(n_i^3)$ moves. Then we claim that the sum of the $n_i$'s totals at most $4n$. Using this claim, the total number of moves is $O(\sum_i n_i^3) = O(\sum_i n_i)^3 = O(n^3)$.

If there is one blank, then subpuzzles are on biconnected graphs, so by the Wilson case we have an $O(n_i^3)$ upper bound. If there are more than one blank: As we saw in the proof of the decision algorithm, a subpuzzle can be solved by a swap plus 2-transitivity. The swap is seen to take $O(n_i)$ moves ($O(n_i)$ to bring two blanks to a swap site, $O(n_i)$ to do the swap, and $O(n_i)$ to restore the blanks to their original places; total is $O(n_i)$). 2-transitivity: This is easily seen to be $O(n_i)$ for a radiating tree. For general trees, we get $O(n_i^2)$. (To move $P_1$, $P_2$ to two destination vertices requires $O(n_i)$; moving $P_3$, $P_4$ to the same two locations is $O(n_i)$; putting the blanks in corresponding locations is $O(n_i^2)$ by the Lemma at the beginning of this chapter. Hence, total is $O(n_i^2)$.) For general graphs, the analysis is almost the same, but total passage of pebbles through biconnected components takes as much as (but no more than) $O(n_i^2)$ moves. However, the total is still $O(n_i^2)$. Armed with a swap and 2-transitivity, conjugation yields any swap in $O(n_i^2)$ moves. Since any permutation is a product of at most $n_i - 1$ swaps, we get a total upper bound of $O(n_i^3)$ moves for the subpuzzle.

Proof of Claim: The sum of the $n_i$ equals $n$ plus the total number of times vertices are counted more than once. If there is one blank: the biconnected components overlap pairwise by at most one vertex. Form a minimum spanning tree connecting the components, thinking of the components as points for now. Each edge represents two components meeting at a vertex, and so represents a vertex being counted one extra time. Since the tree contains at most $n$ edges, the total "overcount" is at most $n$; hence $\sum_i n_i \leq 2n < 4n$. If there are more than one blank: overcounting occurs on isthmuses connecting the subpuzzle graphs. Interior points of isthmuses are counted at most twice, and since the total number of such points is $\leq n$, the interior points contribute at most $n$ to the overcount. End points of isthmuses: Each isthmus contributes at most

2 to the overcount, at its endpoints. The number of isthmuses is $\leq n$, because the minimum spanning tree has at most $n$ edges, and isthmuses are part of the minimum spanning tree. Hence the overcount due to the endpoints is at most $2n$. Therefore $\sum_i n_i \leq n+n+2n = 4n$. Hence the claim holds no matter what the number of blanks.

Solutions with $O(n^3)$ moves therefore exist and, by following the constructions outlined above and detailed in the proof of Theorem 2, they are efficiently plannable.

### 3.4. $O(n^3)$ lower bound

We now complement the above result with a lower bound which matches, to within a constant factor.

**Theorem 5**

There exists a constant $c > 0$ and an infinite sequence of graph puzzles $Puz_i$ on increasingly large graphs $G_i$ with $n_i$ vertices, such that for each $i$, $Puz_i$ requires at least $cn_i^3$ moves for solution.

**Proof**

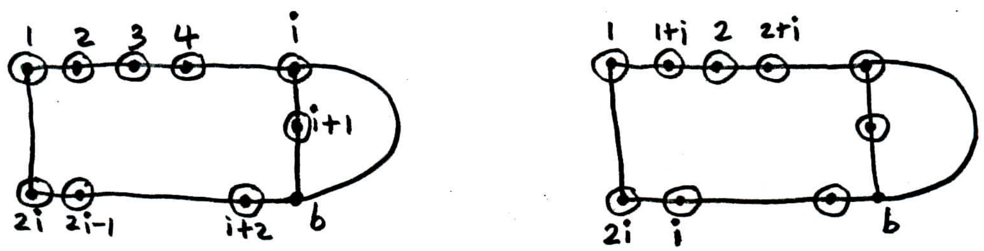



Figure 3-20 Start and end positions for lower bound $Puz_i$

Let $Puz_i$ consist of graph $G_i$ shown in Figure 3-20, with $2i+1$ vertices and $2i$ pebbles, and starting and ending positions as shown. We will show that $Puz_i$ requires $O(i^3)$ moves, as follows. A move sequence that does not waste moves (by retracing move sequences just made) is seen to consist of cycles $A, B$ and their inverses, interspersed in some order (e.g. $ABAAAABA^{-1}B$). It would be wasteful to do $B$ twice in succession, since this would cancel itself. Hence a move sequence can be represented by the form $A^{i_1}BA^{i_2}B...A^{i_k}BA^{i_{k+1}}$ where $i_j$ is a nonzero integer (positive or negative), except $i_1$ and $i_{k+1}$ may be 0.

Now consider the "entropy function" of position

$E = \sum_{j=0}^{i}$ (shortest circular distance from pebbles $j$ to $j+i$)

where circular distance is either clockwise or counterclockwise. Initially, $E = i^2$; at the end, $E = i$. Change in $E$ is $i^2 - i$.

It is seen that $A$ does not change $E$, and $B$ changes $E$ by 0 or by 2. Hence to effect the change in $E$ requires $O(i^2)$ occurrences of $B$ in the move sequence. But because occurrences of $A^{i_j}$ and $B$ alternate, this implies that $A$ occurs at least $O(i^2)$ times. Since the number of moves to perform the cycle $A$ is $O(i)$, we need at least $O(i^3)$ moves for solution.

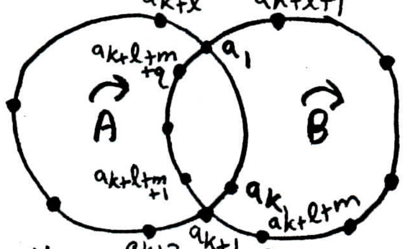
This completes the proof of the lower bound.

# 4. The Diameter of Permutation Groups

As mentioned in the introduction, this chapter is concerned with the diameter of permutation groups generated by sets of cyclic permutations. We begin with some examples of generator sets which yield groups of polynomial diameter, then speculate on some conditions on the generator set which might give groups of superpolynomial diameter. The main part of the chapter consists of theorems which give information about the diameter of a group under various conditions. They imply the result given in the introduction, which is a moderately exponential upper bound on the diameter of groups generated by cycles which satisfy a few conditions.

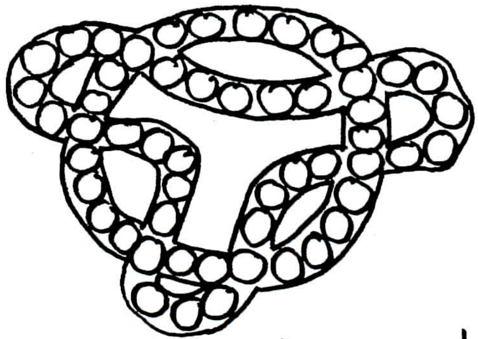
## 4.1. What is not of Exponential Diameter, and what might be

The Hungarian Rings puzzle consists of two intersecting circular rings in which distinguished marbles circulate. The problem is to obtain a desired rearrangement of the marbles by a sequence of operations, where an operation consists of circulating the marbles in one of the rings. This problem immediately translates into the permutation problem of determining membership in the group generated by two intersecting cyclic permutations. By [HFL], we can decide membership in polynomial time; however, it is of interest to know how many "moves" are required, i.e. the length of the shortest word which gives the desired permutation.



Hungarian Rings

Figures 4-1 and 4-2


Candidate "Exponential" Puzzle

In Figure 4-1 is shown schematically two cyclic permutations which intersect at two points. This corresponds to the commercial version of the Hungarian Rings. Note that this is not like a pebble puzzle on a $T_3$-graph, because only $A$ and $B$ are possible, and not the third loop; the Hungarian rings is a physical movers' problem which imposes this restriction mechanically. This gives reason to expect that the number of moves may need to be larger in some permutation puzzles than in the pebble puzzles.

What is the diameter of the group generated by these two cycles? It is first useful to observe that, if some arc $C$ contains at least $r$ internal nodes, and an arc $D$ on the other cycle contains at least one internal node, then we can get $r+1$-transitivity in $O(rn)$-long moves. This is done, roughly speaking, by moving one desired marble after another to $a_1$, then rotating it onto arc $C$. The cycle not containing arc $C$ is rotated to bring the next desired marble to $a_1$, leaving the contents of $C$ undisturbed. Arc $D$ serves as temporary "storage" of a marble which, already on arc $C$, needs to be removed from $C$ and then placed onto $C$ at the right place.

Suppose that in the Figure, $l \geq 6$ and $m \geq 1$. Then we have efficient 6-transitivity. Now $ABA^{-1}B^{-1} = P = (a_1 a_{k+l+m+q} a_{k+l})(a_k a_{k+1} a_{k+l+m})$. Using 6-transitivity, we can find a permutation $P_1$ which sends $a_1, a_{k+l+m+q}, a_{k+l}$ to $a_1, a_{k+l}, a_{k+l+m+q}$ respectively and fixes $a_k, a_{k+1}, a_{k+l+m}$. Then conjugating $P$ by $P_1$ gives $P_2 = (a_1 a_{k+l} a_{k+l+m+q})(a_k a_{k+1} a_{k+l+m})$. $P_2$ is a product of two 3-cycles, one the inverse of the one in $P$, the other the same as the other in $P$. So $PP_2 = (a_k a_{k+l+m} a_{k+1})$, a 3-cycle. Then, using 3-transitivity, we get the alternating group. Hence $l \geq 6$ and $m \geq 1$ implies a polynomial diameter for the Hungarian Rings puzzle with the rings intersecting at two places.

What happens if the number of intersections of the two cycles is some number $k$ greater than 2? By similar reasoning to the above, we get $ABA^{-1}B^{-1}$ to be a product of $k$ 3-cycles. Then a conjugation argument similar to the above yields that, if we have $3k$-transitivity, then we can get a single 3-cycle. How do we get efficient $3k$-transitivity? Well, an arc of $3k-1$ nodes and another arc with one node would suffice for efficient $3k$-transitivity. Without this assumption, $3k$-transitivity can be done in words of length $O(n^{3k})$ (see Chapter 2), which is polynomial if $k$ is bounded. However if $k$ is large, then this bound is exponential. If no arc has enough nodes in it, there might be no efficient way to get the desired degree of transitivity.

The foregoing considerations suggest that a good candidate for a Hungarian Rings puzzle with superpolynomial diameter is one with lots of crossings and no long arcs (see Figure 4-2). To be more quantitative, suppose that there are k equally spaced crossings. Then the arcs have length on the order of $n/k$. We want this to be less than $3k$. So: $n/k < 3k$, i.e. $k > \sqrt{n/3}$. This suggests that we should use at least on the order of $\sqrt{n}$ crossings to create a likely exponential puzzle. It would be of great interest to establish an exponential or moderately exponential lower bound for some of these "candidate" puzzles.

We now leave these examples and speculations, and prove some results about the diameter of permutation groups.

## 4.2. Results about the Diameter of Permutation Groups

The following are classical theorems in the theory of permutation groups.

**Theorem A**

If the group $G$ on $n$ letters is $k$-transitive and $k > n/3 + 1$, then $G = A_n$ or $S_n$.

It is actually the case that 6-transitivity implies that the group is $A_n$ or $S_n$. This surprising result was a long-standing conjecture until very recently [C]; the proof makes use of the recent classification of the finite simple groups, which required about 30 years of effort by hundreds of mathematicians, in about 500 journal articles totaling approximately 10,000 pages!

**Theorem B**

If $G$ is primitive on $n$ letters, and a subgroup $H$ moves only $m < n$ letters and is primitive on them, then $G$ is $n - m + 1$-transitive.

We prove the following versions of these theorems, which give information about the diameter:

**Theorem 1**

If group $G$ on $n$ letters is $k$-transitive in words of length $\leq L$, the generator set $S$ is closed under inverses, and $k > n/3 + 1$, then $G = A_n$ or $S_n$ and $\mathrm{Diam}(G(S)) < 4n^2 L$.

We do not know how to make effective the aforementioned result on 6-transitivity.

**Theorem 2**

If $G$ is primitive on $n$ letters, and $H$ is the primitive subgroup generated by a cyclic permutation of prime length $p < n$, and the generator set $S$ is closed under inverses, then $G$ is $n - p + 1$-transitive using words of length $< 2^{6\sqrt{p}+1} n^3 (n^2 + diam(H(S)))$.

**Theorem 3**

If $G$ is primitive on $n$ letters, and $H$ is a 2-transitive subgroup which moves only $2 \leq m < n$ letters, and the generating set $S$ is closed under inverses, then $G$ is $n - m + 1$-transitive using words of length $< 2^{9\sqrt{m}+1} n^3 (n^2 + diam(H(S)))$.

We were not able to prove an effective version of theorem B for arbitrary primitive $H$, but did obtain the special cases contained in theorems 2 and 3.

These theorems imply the following corollaries

**Theorem 2′**

If $G$ is primitive on $n$ letters, and $H$ is the primitive subgroup generated by a cyclic permutation of prime length $p < 2n/3$, and the generator set $S$ is closed under inverses, then $G$ is $A_n$ or $S_n$, and $Diam(G(S)) < 2^{6\sqrt{p}+3} n^5 (n^2 + diam(H(S)))$.

**Theorem 3′**

If $G$ is primitive on $n$ letters, and $H$ is a 2-transitive subgroup which moves only $2 \leq m < 2n/3$ letters, and the generating set $S$ is closed under inverses, then $G$ is $A_n$ or $S_n$, and $Diam(G(S)) < 2^{9\sqrt{m}+3} n^5 (n^2 + diam(H(S)))$.

Theorem

If a primitive group $G$ on $n$ letters is generated by a set $S$ of cyclic permutations, one of prime length $p < 2n/3$, then $G$ is $A_n$ or $S_n$, and $\mathrm{Diam}(G(S)) < 2^{6\sqrt{p}+4}n^8$.

This last theorem provides a partial extension of [DF]'s upper bound for bounded cycles to unbounded cycles. It would be desirable to generalize the result to apply to all cycles, and to find a matching lower bound on diameter.

## 4.3. Proofs of Theorems 1,2,3 and the Corollaries

### 4.3.1. Proof of Theorem 1

Following the classical proof, we find a nonidentity permutation which moves at most $k$ letters. Then using $k$-transitivity, we will get a 3-cycle and then easily get $A_n$ or $S_n$. Our contribution is an estimate of the wordlength at each step of the derivation.

Suppose $S$ is a nonidentity permutation which moves $r > k$ letters. Write $S$ as a product of disjoint cycles, and number the letters in the order that they appear in the expression: $S = (b_1 ... b_{i_1})(b_{i_1+1} ... b_{i_2}) ... (b_{i_{m-1}+1} ... b_{i_m})$. If no $i_j$ is equal to $k-1$, then we have case 1, otherwise case 2.

case 1

$S = (b_1 ... b_{i_1}) ... (b_{i_{l-1}+1} ... b_{k-1} b_k ... b_{i_l})$.... Since we have $k$-transitivity, let $T$ be a permutation which fixes $b_1, ..., b_{k-1}$ and moves $b_k$ to $c_k$, where $c_k$ is a letter moved by $S$, but $c_k$ is not $b_1, ...$,or $b_k$. Then $T^{-1}ST = (b_1 ... b_{i_1}) ... (b_{i_{l-1}+1} ... b_{k-1} c_k ... b_l)$....

Note that $T^{-1}ST$ has the same effect as $S$ on the first $k-2$ letters. However, $T^{-1}ST$ is not equal to $S$, so $T^{-1}STS^{-1}$ is nontrivial. The letters $P$ moved by $T^{-1}STS^{-1}$ are those letters moved by $T^{-1}ST$ which are not "cancelled" by $S^{-1}$, plus those letters not moved by $T^{-1}ST$ which are moved by $S^{-1}$, so

$P =$ ((domain $T^{-1}ST$)-(domain $T^{-1}ST$ cancelled by $S^{-1}$)) $\cup$ (domain $S^{-1}$ - domain $T^{-1}ST$)

$=$ ((domain $T^{-1}ST$ $\cup$ (domain $S^{-1}$ - domain $T^{-1}ST$))-(domain $T^{-1}ST$ cancelled by $S^{-1}$)

$=$ (domain $T^{-1}ST$ $\cup$ domain $S^{-1}$)-(domain $T^{-1}ST$ cancelled by $S^{-1}$)

$= (A \cup B) - C$.

Now, $|A| = |B| = |S| = r$. But $A$ and $B$ overlap at least on the letters $b_1, , , .b_{k-1}, c_k$. So $|A \cup B| \leq 2r - k$. $|C| \geq k-2$ since $T^{-1}STS^{-1}$ fixes $b_1, ..., b_{k-2}$. So $|P| \leq (2r-k)-(k-2) = 2r-2k+2$.

case 2

In this case, some $i_l$ is equal to $k-1$. $S = (b_1 ... b_{i_1}) ... (b_{i_{l-1}+1} ... b_{k-1})(b_k ...)$.... Using $k$-transitivity, let $T$ fix $b_1, ..., b_{k-1}$ and move $b_k$ to $d_k$, where $S$ does not move $d_k$. Then $T^{-1}ST = (b_1 ... b_{i_1}) ... (b_{i_{l-1}+1} ... b_{k-1})(d_k ...)$....

Note that $T^{-1}ST$ and $S$ do the same thing to $b_1, ..., b_{k-1}$. However, $T^{-1}ST$ is not equal to $S$.

As in case 1, the set $P$ of letters moved by $T^{-1}STS{-}1$ is

(domain $T^{-1}ST$) $\cup$ domain $S^{-1}$) - (domain $T^{-1}ST$ cancelled by $S^{-1}$) $= (A \cup B) - C$.

$|A| = |B| = |S| = r$. But $A$ and $B$ overlap at least in $b_1, ..., b_{k-1}$, so $|A \cup B| \leq 2r-(k-1)$. $C$ contains at least $b_1, ..., b_{k-1}$ so $|C| \geq k-1$. Therefore $|P| \leq (2r-(k-1))-(k-1) = 2r-2k+2$.

Under what conditions is $|P| < r$? Well, $2r-2k+2 < r$ iff $r < 2k-2$. By $k$-transitivity, we can select a nonidentity permutation $S$ which fixes $k-1$ letters and moves another letter. Then the number of letters $r$ moved by $S$ satisfies $r \leq n-(k-1) = n-k+1 < 2k-2$ (since $k > n/3+1$ by hypothesis). Hence for this $S$, the condition is satisfied for $T^{-1}STS^{-1}$ to move less letters than $S$.

Repeating the procedure by setting $S := T^{-1}STS^{-1}$, we finally obtain a permutation $S$ which moves at most $k$ letters $b_1, \ldots, b_{i_l}, i_l \leq k$. $S = (b_1 \ldots b_{i_1}) \ldots (b_{i_{l-1}+1} \ldots b_{i_l})$. Let $T$ fix $b_1, \ldots, b_{i_{l+1}-1}$ and move $b_{i_{l+1}}$ to a different letter $c$. Then $T^{-1}ST = (b_1 \ldots b_{i_1}) \ldots (b_{i_{l-1}+1} \ldots c)$, and $S^{-1}T^{-1}ST = (b_{i_l} c b_{i_{l-1}+1})$, a 3-cycle. Since $k > n/3+1$, we have $k \geq 2$, so using the 3-cycle and 2-transitivity, it is known that we have all of $A_n$, and perhaps $S_n$.

This completes the classical portion of the theorem, and its proof.

Quantitative analysis

Let us first determine how many iterations were used to get $|P| \leq k$. We started with a permutation $S$ moving $r \leq n-k+1 < 2k-2$ letters. Let $t = 2k-2$. Then $r < t$. Now after one iteration, the number of letters moved is $r_1 = |P| \leq 2r-2k+2 = 2r-t = r-(t-r)$. After another iteration, the number of letters moved is $r_2 \leq 2r_1 - t = 4r-3t = r-3(t-r)$. In general, after $i$ iterations $r_i \leq r-(2^i-1)(t-s)$ (as long as $r, s_1, \ldots, r_{i-1}$ are all $>k$). Note that this is an exponential rate of decline: $r - r_i > 2^i - 1$. Since $S$ started by moving $r \leq n-k+1$ letters, we have $r-k \leq n-2k+1 < n/3-1$. If after $i$ iterations, $2^i - 1 \geq n/3-1$, then we attain $r_i \leq k$, which is the goal. Hence $|P| \leq k$ is achieved in $i$ or fewer iterations, where $i$ is the smallest integer such that $2^i \geq n/3$; this is logarithmic in $n$.

Now to compute the diameter. Suppose, as in the hypothesis, that we get $k$-transitivity in wordlength $\leq L$. So the starting $S$ can be picked to have wordlength $\leq L$. We next form $S_1 = T^{-1}STS^{-1}$, where $T$ has length $\leq L$. Using the assumption that the generator set is closed under inverses, $S_1$ has length $\leq 2L+2|S|$ (where $|S|$ is the length of $S$). On the next iteration,

$$|S_2| \leq 2L+2|S_1| \leq 2L+2(2L+2|S|) = 6L+4|S|.$$

In general, after $i$ iterations,

$$|S_i| \leq (2^{i+1}-2)L + 2^i|S|.$$

We go for $i$ iterations, where $i$ is the least integer s.t. $2^i \geq n/3$; then $2^i < 2n/3$. We get

$$|S_i| < (4n/3-2)L + (2n/3)|S|$$

$$\leq (4n/3-2)L + (2n/3)L = (2n-2)L.$$

Then the final step $S_i^{-1}T^{-1}S_iT$ gives a 3-cycle, and wordlength is $< (4n-4)L + 2L = (4n-2)L$.

Now suppose $n \geq 5$,

so that (as $k > n/3+1$) $k \geq 3$.

So, using 3-transitivity and the 3-cycle, we get any 3-cycle (by conjugation) in wordlength $< L + (4n-2)L + L = 4nL$. Then we generate anything in $A_n$ using at most $n-2$ 3-cycles, so anything in $A_n$ has wordlength

$$< 4n(n-2)L < 4n^2L - L \text{ , when } n \geq 5.$$

(For $n < 5$, the group has

diameter $\leq n! < 4n^2 - 1 \leq 4n^2L - L$,

so the wordlength of any element satisfies the same bound in this case also.)

If the group is $S_n$, then some generator $s_i$ is odd. Express an element of $S_n$ by dividing by $s_i$, then expressing the resulting element of $A_n$ by a word of length $< 4n^2L - L$. Hence total wordlength is $\leq 1 + (4n^2L - L) \leq 4n^2L$. This is then an upper bound on the diameter of the resulting group, whether it be $A_n$ or $S_n$, and the proof is complete.

### 4.3.2. Proofs of Theorems 2 and 3

We recall the statements of Theorems 2 and 3.

**Theorem 2**

If $G$ is primitive on $n$ letters, and $H$ is the primitive subgroup generated by a cyclic permutation of prime length $p < n$, and the generator set $S$ is closed under inverses, then $G$ is $n-p+1$-transitive using words of length $< 2^{6\sqrt{p}+1}n^3(n^2 + diam(H(S)))$.

**Theorem 3**

If $G$ is primitive on $n$ letters, and $H$ is a 2-transitive subgroup which moves only $2 \leq m < n$ letters, and the generating set $S$ is closed under inverses, then $G$ is $n-m+1$-transitive using words of length $< 2^{9\sqrt{m}+1}n^3(n^2 + diam(H(S)))$.

First we will prove the following preliminary Lemmas.

**Lemma 2a**

If $G$ is primitive on $n$ letters, and $H$ is the primitive subgroup generated by a cyclic permutation of prime length $p < n$, and the generator set $S$ is closed under inverses, then there exists a $g \in G$ which takes $D = \text{Domain}(H)$ to $D'$, such that $D$ and $D'$ overlap on exactly $\cancel{m}$ p $-1$ letters, and $g$ has wordlength $< 2^{6\sqrt{p}}(n^2 + diam(H(S)))$.

**Lemma 3a**

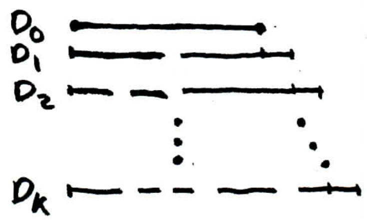
If $G$ is primitive on $n$ letters, and $H$ is a 2-transitive subgroup which moves only $2 \leq m < n$ letters, and the generating set $S$ is closed under inverses, then there exists a $g \in G$ which takes $D = \text{Domain}(H)$ to $D'$, such that $D$ and $D'$ overlap on exactly $m-1$ letters, and $g$ has wordlength $< 2^{9\sqrt{m}}(n^2 + diam(H(S)))$.

The purpose of the Lemmas is roughly as follows. By making a set of letters overlap itself by all but one letter, and repeating this process, it is possible to build a tower of conjugates of $H$ whose domains look like the diagram in Figure 4-3. It is then possible to achieve $n-k+1$ transitivity by using the fact that the domains intersect, to move any letter to the right end of the bottom row (as pictured in the figure), then move any letter to the right end of the next to bottom row without disturbing the previous element, and so on to get $n-k+1$ transitivity. The details will be given later, as well as an analysis of the wordlength needed to do these operations. This, combined with Lemma 2a, gives Theorem 2, and with Lemma 3a it gives Theorem 3.

### 4.3.2.1. Proofs of the Lemmas

We first motivate the proofs of the Lemmas. Roughly speaking, we will first find a permutation which maps $D$ to a $D_1$ which overlaps $D$ partially but not totally. Then a clever device is repeated, which increases the overlap with each iteration, but never reaches total overlap. Naturally, we must reach a $D'$ where overlap is all but one letter.

Let us be more explicit. Let $g \in G$ take $D$ partly, but not entirely, to itself. Let $H_0 = g^{-1}Hg$, with $D_0 = \text{dom}(H_0) = g(D)$. Let $A = D \cap D_0, B = D_0 - D, C = D - D_0$ (see Figure 4-4 for a crude Venn diagram). Then $|A| + |B| = m$ and $|B| = |C|$.




A tower of Conjugates

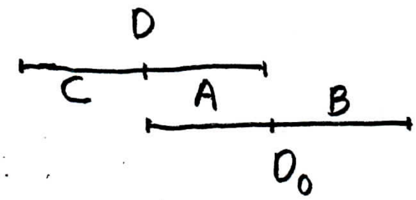



Fig 4-4

Figures 4-3 and 4-4

Now if $g_0 \in H_0$ takes $B$ to $B_0$, such that $B \cap B_0$ is nonempty but not entire, then $H_1 = g_0^{-1} H g_0$ has domain

$D_1 = g_0(C \cup A) = g_0(C) \cup g_0(A) = C \cup g_0(A)$.

$A_1 = D \cap D_1 = C \cup (A \cap g_0(A))$,

$B_1 = D_1 - D = g_0(A) - A = g_0(A) \cap B$,

and $C_1 = D - D_1$ has, as before, the same size as $B_1$. As $B$ is not a subset of $g_0(A)$, we have that $B_1$ is a proper subset of $B$.

And $B - B_1 = B - (g_0(A) \cap B)$

$=$ the part of $B$ not mapped into by $A$

$=$ the part of $B$ mapped into by $B = B \cap g_0(B)$.

Hence the amount by which $B$ decreases in size by going to $B_1$ is the size of $B \cap g_0(B)$, which is nonempty. Therefore the overlap has been increased by $|B \cap g_0(B)| \geq 1$ and the overlap is not entire since $|B \cap g_0(B)| < |B|$.

Our aim will be to pick $g_0$ to map $B$ as much at possible into $B$ (and so decrease the size of $B$), but not entirely (so that there is a new nonempty $B$). Then we repeat the scenario with $H$ and $H_1$, picking $g_1$ in $H_1$ to get $H_2$, and so on, decreasing $B$ till it contains just one letter. The object is to use as few iterations as possible, to keep word length from growing too much (it approximately doubles with each iteration). In the proofs which follow, we will show that the properties of $H$ are sufficient to ensure that only $O(\sqrt{|B|})$ iterations are needed to get $B$ down to size 1. The upper bounds on wordlength will follow in a straightforward way.

Proof of Lemma 2a

We want to see how much $B$ can be decreased when $H$ is generated by a cyclic permutation $h_1$ of prime order $p$.

For simplicity, let the domain of $H$ be $\{1, 2, ..., p\}$ and let $h_1 = (12...p)$. Let $B$ be a proper subset of dom($H$), $|B| = r < p$. We wish to find a power $k$ $(0 < k < p)$ of $h_1$ which maps $B$ onto much of itself, but not entirely. (Note that the latter cannot happen, so we do not need to be careful to avoid this case. For any $b \in B$ has orbit (the set of iterated images of $b$ under a mapping) under $h_1^k$ of length $l$ where $l$ divides $p$. $l$ is not 1 because $h_1^k(b)$ is not $b$. Hence $l = p$. If $h_1^k$ took $B$ to $B$, then it follows that $b$'s orbit is all in $B$. But then $|B| = p$, a contradiction.)

Let $S_k =$ the number of unordered pairs of elements of $B$ whose elements are $k$ apart in the shortest "circular" direction. Then $h_1^k$ maps exactly $S_k$ elements of $B$ into $B$, and so $|B|$ decreases by $S_k$.

Let us determine how large $max(1 \leq k \leq (p-1)/2)S_k$ must be. Note that

$\sum_{k=1}^{(p-1)/2} S_k$

is equal to the number of unordered pairs of elements of $B$, which is $r(r-1)/2$. So some $S_k$ must be at least

$(r(r-1)/2)/((p-1)/2) = r(r-1)/(p-1)$.

Hence in one step we can decrease $B$'s size from $r$ to $r - r(r-1)/(p-1)$, or perhaps an even faster decrease.

Let us examine the mapping $r := r - r(r-1)/(p-1)$.

$(r-1)/(p-1) > r/(2p)$

iff $2rp - 2p > rp - r$

iff $rp > 2p - r$

iff $r > 2 - r/p$.

We have $r \geq 2$, which implies the latter inequality and so the first inequality holds. This implies that

$r := r - r^2/(2p)$

is a mapping which reduces more slowly. However, we will show that repetitions of even this mapping will reduce $r$ to $O(\sqrt{r})$ in only $O(\sqrt{r})$ iterations; once $r$ is this small, we can afford $O(\sqrt{r})$ additional conjugations, each of which reduce $r$ by at least 1, to finally reach $r = 1$ (the goal).

Therefore let us examine the mapping $r_{k+1} = r_k - r_k^2/(2p)$. First make the scale change of variables $z_k = r_k/\sqrt{p}$. Then we get

$$z_{k+1}\sqrt{p} = z_k\sqrt{p} - (z_k\sqrt{p})^2/(2p)$$

so $z_{k+1} = z_k - z_k^2/(2\sqrt{p})$.

Note that $z_0 = r_0/\sqrt{p} < \sqrt{p}$ (since $r_0 < p$), so the start value $z_0$ is down at level $\sqrt{p}$ rather than as high as $p$ (in the original variable).

Let us see how many iterations it takes to decrease $z_0$ by 1. Now during the range of iterations for which $z_i > z_0 - 1, i = 0, ..., k-1$, we have

$$z_k = z_{k-1} - z_{k-1}^2/(2\sqrt{p}) \leq z_{k-1} - (z_0 - 1)^2/(2\sqrt{p}).$$

So $z_k \leq z_0 - k(z_0 - 1)^2/(2\sqrt{p})$.

Then $z_k$ becomes $\leq z_0 - 1$ when $k(z_0 - 1)^2 \geq 2\sqrt{p}$, i.e. when $k \geq 2\sqrt{p}/(z_0 - 1)^2$. So $\lfloor(2\sqrt{p}/(z_0 - 1)^2)\rfloor + 1$ iterations reach $z_0 - 1$ or lower.

Therefore we can reduce $z_0$ successively by 1 until $\lfloor z_0 \rfloor = 0$ (corresponding to $r < \sqrt{p}$) in at most

$$(\lfloor 2\sqrt{p}/(\sqrt{p}-1)^2\rfloor + 1) + (\lfloor 2\sqrt{p}/(\sqrt{p}-2)^2\rfloor + 1) + ... + (\lfloor 2\sqrt{p}/1^2\rfloor + 1)$$

$$< 2\sqrt{p}(pi^2)/6 + \sqrt{p} < 5\sqrt{p} \text{ steps,}$$

since $1/1^2 + 1/2^2 + 1/3^2 + ... = pi^2/6$.

We have thus proved the following Lemma

**Lemma**

The mapping $r := r - r^2/(2p)$ started at $r_0 < p$, requires less than $5\sqrt{p}$ iterations to reach a value of $r$ less than $\sqrt{p}$.

Using this result, we can now take at most $\sqrt{p}$ additional conjugation steps (reducing $B$ by at least one letter per iteration) to reach $|B| = 1$ in a total of $< 6\sqrt{p}$ iterations.

Wordlength analysis

We described above how, by forming a suitable conjugate $g^{-1}Hg$ of $H$, we could get the domains of $H$ and of $g^{-1}Hg$ to overlap by all but one letter. What is the wordlength of the permutation g obtained by conjugating as many as $6\sqrt{p}$ times? Well, let

$$H_0 = g_0^{-1} H g_0$$

be the first conjugate, to get some overlap in domains. Then we picked $h_0 \in H_0$, so

$$h_0 = g_0^{-1} h_0' g_0$$

for some $h_0' \in H$, to get

$$H_1 = (g_0^{-1} h_0' g_0)^{-1} H (g_0^{-1} h_0' g_0) = (g_0^{-1} h_0'^{-1} g_0) H (g_0^{-1} h_0' g_0).$$

Then we picked $h_1 \in H_1$, so

$$h_1 = g_0^{-1} h_0'^{-1} g_0 h_1' g_0^{-1} h_0' g_0$$

for some $h_1' \in H$, to get $H_2 = h_1^{-1} H h_1$ which is a cumbersome expression when written out! And so on.

Now the permutation $g_0$ which conjugates $H$ to get $H_0$ can be chosen to have wordlength $L_0 < n^2$ (see Chapter 2). $h_0$ had length

$L_1 =$ wordlength of $(g_0^{-1} h_0' g_0) \leq 2L_0 + diam(H)$.

$h_1$ had length

$L_2 =$ wordlength of $g_0^{-1} h_0'^{-1} g_0 h_1' g_0^{-1} h_0' g_0$

$\leq 2L_1 + diam(H)$. (Note that we are using the assumption that the generators are closed under inverses.)

In general, $h_i$ has wordlength less than $2^i L_0 + (2^i - 1) diam(H)$, which is less than $2^i (n^2 + diam(H))$. Since only $i = 6\sqrt{p}$ iterations are needed at most, we get the upper bound length $< 2^{6\sqrt{p}}(n^2 + diam(H(S)))$. This proves **Lemma 2a.**

Proof of Lemma 3a

Now we want to see how much $B$ can be decreased when $H$ is 2-transitive. We will use an interesting probabilistic argument which yields, in the end, a similar bound to that obtained in the proof of Lemma 2a.

Let $H_0$ be a conjugate of $H$, with domain $D_0$ which partially overlaps with domain $D$ of $H$. Let $A$, $B$, and $C$ be the sets defined in the proof of Lemma 2a.

$|D| = |D_0| = m.$

As $D_0 = A \bigcup B$, where $A$ and $B$ are disjoint, let

$|A| = Lm, |B| = (1-L)m, 0 < L < 1.$

We will define some functions which formally look like probability random variables, distribution functions, expectations, and variances. However, we will treat them formally so that there is no suggestion that chance enters the discussion; it is nevertheless enlightening to informally think in terms of probability.

For $a \in A$ and $p \in H_0$, let

$X_a(p) = 1$ if $p(a) \in B$, 0 otherwise.

Let $X(p) = \sum_{a \in A} X_a(p)$.

Define $E(X_a)$ to be $\sum_{p \in H_0} 1/|H_0| X_a(p)$.
This is equal to
$| \{p \in H_0$ which takes $a$ into $B\}| \,/\, |\{p \in H_0\}|$.

As there are $m$ cosets (of equal size) of $H_0$, each of which takes $a$ to a different place, we have that

$E(X_a) = |\{$cosets moving $a$ into $B\}| \,/\, |\{$ cosets $\}|$

$= |B|/m = 1 - L.$

Define $E(X)$ to be $\sum_{p \in H_0} 1/|H_0| {}^* X(p)$.
This evaluates to

$\sum_{p \in H_0} 1/|H_0| {}^* \sum_{a \in A} X_a(p)$

$= \sum_{a \in A} \sum_{p \in H_0} 1/|H_0| X_a(p)$

$= \sum_{a \in A} E(X_a)$

$= |A|(1-L) = L(1-L)m.$

In general, we will use the symbol $E(Z)$ to mean

$\sum_{p \in H_0} 1/|H_0| Z(p)$, if $Z$ is a function of $p$.

Define $Var(X)$ to be $E((X - E(X))^2)$. This evaluates to
$E(X^2 - 2XE(X) + E(X)^2)$
$= E(X^2) - 2E(X)^2 + E(X)^2 = E(X^2) - E(X)^2$.

The latter term, $-E(X)^2$, is equal to $L^2(1-L)^2m^2$.
The first term is $E(X^2)$ which evaluates to

$E(\sum_{a_1, a_2 \in A} X_{a_1} X_{a_2})$

$= E(\sum_{a \in A} X_a^2) + E(\sum_{distinct a_1, a_2 \in A} X_{a_1} X_{a_2})$

$= T_1 + T_2$.

$T_1 = \sum_{a \in A} E(X_a^2)$

$= \sum_{a \in A} E(X_a)$

$= E(X) = L(1-L)m$.

$T_2 = \sum_{a_1, a_2 \in A, a_1 \neq a_2} E(X_{a_1} X_{a_2})$.
Now, for distinct a1,a2 we have

$E(X_{a_1} X_{a_2})$

$= \sum_{p \in H_0} 1/|H_0| * X_{a_1}(p) X_{a_2}(p)$

$= |\{p \in H_0$ taking both $a_1$ and $a_2$ out of $A\}|/|H_0|$.

Since $H$ is 2-transitive, there are $m(m-1)$ equal sized cosets, each of which takes $(a_1, a_2)$ to a different pair of places. $(1-L)m((1-L)m-1)$ of these take $(a_1, a_2)$ out of $A$.

So $E(X_{a_1} X_{a_2}) = ((1-L)m((1-L)m-1))/(m(m-1))$

$= (1-L)((1-L)m-1)/(m-1)$

$= ((1-L)^2m - (1-L))/(m-1)$.

So $T_2 = Lm(Lm-1)((1-L)^2m - (1-L))/(m-1)$.

Then $Var(X) = T_1 + T_2 - L^2(1-L)^2m^2$

$= L(1-L)m + Lm(Lm-1)((1-L)^2m - (1-L))/(m-1) - L^2(1-L)^2m^2$

$= F_1.$

We claim that $F_1$ reduces to $L^2(1-L)^2(m+1+1/(m-1)) = G_1$.

Proof:

$F_1/(Lm(1-L)) = F_2 = 1 + (Lm-1)((1-L)m-1)/(m-1) - L(1-L)m.$

$G_1/(Lm(1-L)) = G_2 = L(1-L)(m+1+1/(m-1))/m$

$= L(1-L)(m^2/(m-1))/m$

$= L(1-L)m/(m-1).$

$F_2(m-1) = F_3 = (m-1) + (Lm-1)((1-L)m-1) - L(1-L)m(m-1),$

and $G_2(m-1) = G_3 = L(1-L)m.$

We claim that $F_3 = G_3$ is an identity,

i.e. $C = F_3 - G_3 = (m-1) + (Lm-1)((1-L)m-1) - L(1-L)m^2$ is identically zero.

Well, expanding $C$ yields

$C = (m-1) + L(1-L)m^2 - (1-L)m - Lm + 1 - L(1-L)m^2$

$= (m-1) - (1-L)m - Lm + 1$

which is identically zero. Hence $F_3 = G_3$, so $F_1 = G_1$ and we conclude that

$Var(X) = L^2(1-L)^2(m+1+1/(m-1)).$

Now, if $(X - E(X))^2$ were $> s^*Var(X)$ for a fraction $0 < f < 1$ of the permutations of $H_0$ and $s > 1$, then

$E(X-E(X))^2 \geq fs^*Var(X),$

which is a contradiction if $fs \geq 1$. So, for example (pick $f = .5$, $s = 2$) at most half of the $p \in H_0$ can make the value of $(X - E(X))^2$ over $2Var(X)$. So, for at least half of the $p \in H_0$,

$(X - L(1-L)m)^2 \leq 2L^2(1-L)^2(m+1+1/(m-1)),$

i.e. $|X - L(1-L)m| \leq \sqrt{2}L(1-L)\sqrt{(m+1+1/(m-1))},$

i.e. (as $m \geq 2$ implies that $m+1+1/(m-1) \leq 2m$)

$|X - L(1-L)m| \leq L(1-L)\sqrt{4m},$

so we conclude that

$$L(1-L)(m-2\sqrt{m}) \leq X \leq L(1-L)(m+2\sqrt{m}) \tag{1}$$

for at least half of the $p \in H_0$.

Now, recall that $X(p) =$ number of letters of $A$ leaving $A$ under $p$

$=$ number of letters of $B$ leaving $B$ under $p$.

Then $(1-L)m - X =$ number of letters of $B$ mapping into $B$

$=$ amount by which $B$ becomes smaller on next iteration of conjugation.

We want this latter quantity to be large, but not to be all of $B$. By the above inequality (1) on $X$, we have that $B$ decreases by at least

$$(1-L)m - X > (1-L)m - L(1-L)(m+2\sqrt{m})$$

$$= (1-L)^2 m - 2L(1-L)\sqrt{m}$$

$$= |B|^*|B|/m - (1-|B|/m)^* 2\sqrt{m}|B|/m$$

$$\geq |B|^*|B|/m - 2\sqrt{m}|B|/m$$

$$= (|B| - 2\sqrt{m})(|B|/m)$$

for some $p \in H_0$ (in fact, for at least half of the $p \in H_0$). Hence some $p \in H_0$ causes $M = |B|$ to reduce at least as fast as the mapping

$$M := M - (M - 2\sqrt{m})(M/m)$$

$$= M(1 + 2/\sqrt{m} - M/m).$$

Let us examine the behavior of this mapping in the range $M \geq 4\sqrt{m}$. Then

$2/\sqrt{m} - M/m \leq -M/(2m)$, so

$$M(1 + 2/\sqrt{m} - M/m) \leq M(1 - M/(2m)) = M - M^2/(2m).$$

Hence in the range $M \geq 4\sqrt{m}$, $M := M - M^2/(2m)$ decreases at least as slowly as the original mapping. If we show that this new mapping decreases at some rate, then we know that the original mapping decreases at that rate or faster. But the Lemma proved in the proof of Lemma 2a applies precisely to this mapping. Therefore, after less than $5\sqrt{m}$ iterations, we have $M \leq 4\sqrt{m}$. Since we can reduce $M$ to 1 in another $4\sqrt{m}$ or fewer iterations, we get the result that less than $9\sqrt{m}$ iterations will reduce $B$ to exactly one letter.

We must take care to check that $B$ does not vanish entirely. Inequality (1) says that $M$ decreases by at most

$$(1-L)^2 m + 2L(1-L)\sqrt{m}$$

$$= M/mM + 2\sqrt{m}(1-M/m)M/m$$

$$= M(M/m + 2/\sqrt{m}(1-M/m))$$

$< M$ (so that decrease is not entire)

if and only if $M/m + 2/\sqrt{m}(1-M/m) < 1$

if and only if $M + 2\sqrt{m}(1-M/m) < m$

if and only if $M(1-2/\sqrt{m}) < m - 2\sqrt{m}$

if and only if $M < (m-2\sqrt{m})/(1-2/\sqrt{m}) = m$.

But since $M < m$, all these inequalities hold, so the decrease is not entire for those $p \in H_0$ which satisfy the inequality (1).

Therefore less than $9\sqrt{m}$ iterations will reduce $B$ to exactly one letter. Then, proceeding to analyze the wordlength (of the composite permutation which achieves this) in exactly the same way as done for Lemma 2a, we get that the wordlength is $< 2^{9\sqrt{m}}(n^2 + diam(H(S)))$. This completes the proof of Lemma 3a.

### 4.3.2.2. Proofs of Theorems 2 and 3

Theorems 2 and 3 will follow in precisely the same way, now that the important step has been accomplished of getting $B$ to overlap itself by all but one letter.

We started with $H$ with domain $D$, and showed in the Lemmas 2a and 3a how to (in semiexponential wordlength) get a conjugate $H_1$ with domain $D_1$ which contains exactly one letter not in $D$.

Now, suppose (induction hypothesis) that we have built a tower of such conjugates $H, H_1, ..., H_k$, with domains $D_0, D_1, ..., D_k$ such that $D_i$ has exactly one letter not in $(D_0 \cup ... \cup D_{i-1})$ for each $i = 1, ..., k$ (the above Lemmas give us a tower up to $H_1$). See Figure 4-5 for a schematic picture of this situation.

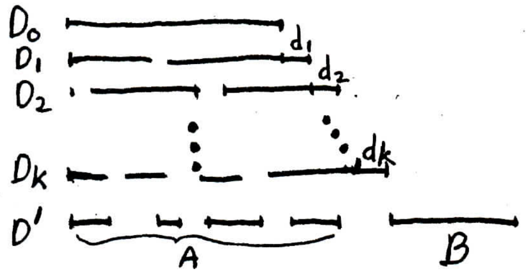



Figure 4-5 A Tower of Conjugates

Let $g \in G$ map $D_0$ partially into $D = (D_0 \cup ... \cup D_k)$ and partly outside $D$. This is possible because $|D_0| \geq 2$, so we can pick distinct $a, b \in D_0$ and use the primitivity of $G$ to map one of $a, b$ into $D$ and the other outside of $D$.

Let $D' = G(D_0)$. Define $A = D' \cap D$, and for $i = 0, ..., k$ define $A_i = D' \cap D_i$. Also define $B = D' - D$.

Now let $H' = g^{-1}H_0 g$. By the Lemmas, there exists an $h$ in $H'$ of "moderately exponential" wordlength which maps $A$ partly to $B$, and causing a significant reduction in the size of $B$.

$h(D_0) = h(D_0 - A_0) \cup h(A_0) = D_0 - A_0 \cup h(A_0)$.

As $(D_0 - A_0) \cap B$ is null, we see that $A_0$ is the only part of $D_0$ that can map into $B$. If in fact $h$ takes $A_0$ partly outside of $A$, then $h^{-1}H_0 h$ gives a new $H'$ with a significantly reduced $B$ (new $B$ is $h(A_0) \cap($ old $B))$, which is at most as large as $h(A) \cap($ old $B)$; the latter we know can be made small, by the Lemmas).

Keep repeating this step until $B$ reduces to exactly one letter, or until the condition (that $h$ take $A_0$ partly out of $A$) is not satisfied. If the condition is always satisfied, then we get $H_{k+1}$ , the next level of the tower.

Let us look at the case that some point in the iteration, $h(A_0)$ is entirely in $A$. Then find the first $A_i$ which has something that leaves $A$ (something in $A = (A_0 \cup ... \cup A_k)$ leaves $A$). Then nothing in $(A_0 \cup ... \cup A_{i-1})$ leaves $A$, so something in

$$A_i - (A_0 \cup ... \cup A_{i-1})$$
$$= (D_i \cap A) - ((D_0 \cap A) \cup ... \cup (D_{i-1} \cap A))$$
$$= (D_i \cap A) - ((D_0 \cup ... \cup D_{i-1}) \cap A)$$

$= (D_i - (D_0 \cup ... \cup D_{i-1})) \cap A$

leaves A.

But $(D_i - (D_0 \cup ... \cup D_{i-1}))$ by hypothesis is a single letter $b_i$, say. So since the intersection in the last expression is not empty, it must be $b_i$. Hence $A_i - (A_0 \cup ... \cup A_{i-1})$ is a single letter $b_i$, and $h(b_i) \in B$. Therefore, if we fall out of the iteration, the single additional step of conjugating $A_i$ by $h$ brings $B$ down to exactly one letter. Hence this case yields $H_{k+1}$ with little extra work.

### How to get n-m+1 transitivity

Build the tower till we reach $H_{n-m}$ with domain $D_{n-m}$. As the tower is being built, construct the graph consisting of a point for each $D_i$, and an edge connecting $D_i, D_j$ if and only if $D_i \cap D_j$ is nonempty. As the next level $D_i$ is added, add a point labeled $D_i$ and draw appropriate edges to previous points. Clearly this graph is connected at each stage of its construction, and remains connected when each new point $D_i$ is added, since $D_i$ intersects with at least one previous domain.

Let $d_0$ be any letter in $D_0$, and let $d_1, ..., d_{n-m}$ be respectively the letter in $D_1, ..., D_{n-m}$ which is not in any previous domain. We will show how to get $n-m+1$ transitivity by showing how to send any $a_0, ..., a_{n-m}$ to $d_0, d_1, ..., d_{n-m}$ respectively. First move $a_{n-m}$ from a domain which it is currently in, to domain $D_{n-m}$, by moving along edges of the graph. At most $n-m$ edges need be traversed. Each edge is traversed by moving $a_{n-m}$, currently in some domain $D_i$, to the intersection of $D_i$ with another domain $D_j$, using a permutation $h_i \in H_i$. Now $a_{n-m}$ is in the domain $D_j$. Once $a_{n-m}$ is in $D_{n-m}$, use a permutation $h_{n-m} \in H_{n-m}$ to put $a_{n-m}$ at location $d_{n-m}$. In the same way, we get the other letters into place. Suppose $a_{n-m}, a_{n-m-1}, ..., a_{j+1}$ have all been put into place without disturbing any of the previous placements. Then put $a_j$ into place as follows: $a_j$ is not in any of the locations $d_{j+1}, ..., d_{n-m}$ (since they have been filled already), so $a_j$ is in one of the domains $D_0$,..., or $D_j$. Since the subgraph involving only $D_0, ..., D_j$ is connected, we can get $a_j$ to $D_j$ in at most $j$ edges, and then finally move $a_j$ to $d_j$. Since edge traversals mean using permutations in $H_0$,..., or $H_j$, and since these groups do not move any of the letters $d_{j+1}, ..., d_{n-m}$, we do not disturb the previous placements in the process of placing aj. Thus we have demonstrated how to obtain $n-m+1$-transitivity.

### Quantitive analysis of wordlength of n-m+1 transitivity

We will now analyze the wordlength of the permutation contructed above, which sends $a_0, ..., a_{n-m}$ to $d_0, ..., d_{n-m}$ respectively. Then the wordlength for $n-m+1$-transitivity will be at most twice this amount (send $a_0, ..., a_{n-m}$ to $d_0, ..., d_{n-m}$; then perform the inverse of the permutation which sends the target letters $b_0, ..., b_{n-m}$ to $d_0, ..., d_{n-m}$).

For simplicity of notation, we will derive the bound on wordlength, in the case where $H = H_0$ is 2-transitive (Theorem 3). The bound for Theorem 2 is proved in exactly the same way.

Since

$H_1 = h^{-1} H_0 h$

where $domain(H_1) - domain(H_0) = 1$ letter, and Lemma 3a gives an upper bound of

$$L = 2^{9\sqrt{m}}(n^2 + diam(H_0))$$

for the wordlength of $h$, we get that (using the assumption that the generator set is closed under inverses)

$$Diam(H_1) < 2L + Diam(H_0).$$

Now to get additional levels of the tower:

to get $H_i$, first we make $D_0$ overlap itself partially, using the primitivity of $G$; this takes at most wordlength $n^2$ (see Chapter 2). This gives $H'$. Then after less than $9\sqrt{m}$ iterations, $|B|$ becomes 1.

In the case that $h(A_0)$ always mapped partly outside of $A$, then we are performing the exact same type of construction as for $H_1$, so $Diam(H_i) < 2L + Diam(H_0)$.

If this is not the case, then after $< 9\sqrt{m} - 1$ iterations we perform a special final step. Before the final step, $H'$ has diameter $< L + Diam(H_0)$. The final step consists of conjugating one of $H_1, ..., H_{i-1}$ by a properly chosen element of $H'$. So in this case,

$$Diam(H_i) < 2Diam(H') + max(Diam(H_1), ..., Diam(H_{i-1})).$$

Letting $L_1 = 2L + Diam(H_0)$ be an upper bound on the diameter of $H_1$, and similarly letting $L_i$ be an upper bound on the diameter of $H_i$, then

$$Diam(H_2) < 2(L + Diam(H_0)) + max(Diam(H_0), Diam(H_1))$$

$$= L_1 + Diam(H_0) + max(Diam(H_0), Diam(H_1))$$

$$< L_1 + Diam(H_0) + L_1$$

$$= 2L_1 + Diam(H_0).$$

This upper bound for $Diam(H_2)$ is higher than the upper bound $2L + Diam(H_0)$ for the first case (when $h(A_0)$ always has something outside of $A$), so to be conservative we use this higher bound, and set $L_2 = 2L_1 + Diam(H_0)$.

In the same way, $Diam(H_3)$ is in either case

$$< (L_1 + Diam(H_0)) + L_2$$

$$= 3L_1 + 2Diam(H_0).$$

By induction,

$$Diam(H_{i+1}) < (L_1 + Diam(H_0)) + L_i$$

$$< (L_1 + Diam(H_0)) + (iL_1 + (i-1)Diam(H0))$$

$$= (i+1)L_1 + iDiam(H_0)$$

$= (i+1)2L + (2i+1)Diam(H_0).$

Now, to move $a_0, ..., a_{n-m}$ to $d_0, ..., d_{n-m}$:

Moving $a_i$ into place at most requires using one permutation from each of $H_0, ..., H_i$. So total wordlength is less than

$\sum_{i=0}^{n-m} \sum_{j=0}^{i} [(j+1)2L + (2j+1)Diam(H_0)]$

$= \sum_{i=0}^{n-m} [(i(i+1)/2 + (i+1))*2L + (i(i+1) + (i+1))*Diam(H_0)]$

which is (since $i(i+1)/2 + (i+1) < (i+1)^2$ and $i(i+1) + (i+1) = (i+1)^2$)

$< \sum_{i=0}^{n-m} (i+1)^2 (2L + Diam(H_0)).$

Now,

$\sum_{i=0}^{n-m} (i+1)^2$

$= (n-m+1)(n-m+2)(2n-2m+3)/6$

$< n*n*2n/6 = n^3/3$ since $m \geq 2$.

So wordlength is $< n^3/3*(2L + Diam(H_0))$

$< n^3/3*(3L)$

$= n^3 L.$

Finally, to get $n-m+1$-transitivity, as explained above we double this to get the upper bound $n^3*2L = 2^{9\sqrt{m}+1} n^3 (n^2 + diam(H(S)))$, and Theorem 3 is proved. In exactly the same way, we get the upper bound in Theorem 2.

### 4.3.3. Proofs of the Corollaries

Theorem 2′ and Theorem 3′ are proved immediately by substituting $n-m+1$ for $k$ and the upper bound on wordlength for $n-m+1$-transitivity for $L$ in Theorem 1 (whose hypothesis $k > n/3+1$ is satisfied because $m < 2n/3$ implies $n-m+1 > n/3+1$).

The proof of the final corollary is as follows. The generator $h$ of the cyclic subgroup $H = H_0$ of order $p$ is (by hypothesis) in the generator set of $G$. So $Diam(H_0) \leq p$. We are not assuming that the generators are closed under inverses, but because they are cyclic of order $\leq n$, the inverse of a generator is at most the $n$-th power of that generator. Hence the wordlength is at most a factor of $n$ longer than obtained previously, where we assumed closure under inverses. Therefore, $n-m+1$-transitivity requires wordlength

$$< 2^{6\sqrt{p}+1} n^4 (n^2 + p)$$
$$< 2^{6\sqrt{p}+2} n^6.$$

Then, as $m < 2n/3$, we have $n-m+1 > n/3+1$, so using Theorem 1, we get an additional factor of $4n^2$, giving $Diam(G) < 2^{6\sqrt{p}+4} n^8$, which proves the corollary.

## 5. Conclusion and Open Problems

We have obtained some results in pebble coordination problems and the diameter of permutation groups. Specifically, we derived:

1. An efficient decision algorithm for the general pebble coordination problem on graphs.

2. $O(n^3)$ matching upper and lower bounds on the number of moves to solve pebble coordination problems.

3. $2^{6\sqrt{p}+3}n^8$ upper bound on diameter of $A_n$ or $S_n$ when generated by cycles, one of which has prime length $p < 2n/3$.

We see 1. as being a complete and satisfactory result as it stands. It would be of interest to apply the algebraic methods used in the pebble movers' problem to special cases of the general geometric movers' problem which may admit an algebraic approach.

2. could stand a number of refinements.

a. Find exact constants in the $O$-terms.

b. It would be useful to at least have an efficient algorithm which approximates the number of moves required. For it seems that only a small fraction of the graph puzzles actually require $O(n^3)$ moves. As an example, it is not hard to show that the "15-puzzle" generalized to square grids of arbitrary size (with one blank) requires only $O(n^{3/2})$ moves (where $n$ is the number of vertices).

c. We do not even know how to solve b. for the case of $T_2$-graphs with one blank. It is easy to generalize the lower bound result to show that, if one loop has bounded length, then $O(n^3)$ moves are required. However, it is not clear what happens when both loops are about the same length: Since there are two generators $A$, $B$ which yield $A_n$ or $S_n$ (groups of order-$O(n!)$), it follows that some orderings will require $O(nlogn)$ words in $A, B$, which yields a $O(n^2 logn)$ lower bound on number of pebble moves. We do not know whether or not there is a matching upper bound.

3. is only a first step towards understanding the diameter of groups generated by arbitrary cycles. A number of related questions are open:

a. Is the upper bound in 3. tight? Is there a corresponding lower bound of $O(2^{O\sqrt{p}})$ for some instances of 3. ? This would settle the following well-known open problem:

b. Can a transitive group have larger than polynomial diameter for some generator set? Can this be the case for $A_n$ or $S_n$?

c. Can the upper bound in 3. be generalized to less restrictive conditions on the generating cycles? Is it even true that the following conjecture holds?:

d. Is the diameter of a group, relative to any generating set, always bounded above by $O(n^{\sqrt{n}})$? E.g. the group generated by $S = \{(12)(345)...(...[\text{sum of first } n \text{ primes}])\}$ has diameter $O(2^{\sqrt{n}})$, which satisfies the conjecture.

# Bibliography


[C] Peter J. Cameron, "Finite permutation groups and finite simple groups", Bull. London Math. Soc., vol. 13, part 1, Jan. 1981, pp. 1-22.

[DF] J.R. Driscoll and M.L. Furst, "On the diameter of permutation groups", 15th STOC, 1983, pp. 152-160.

[DSY] C.O'Dunlaing, M. Sharir, C.K. Yap, "Retraction: A new approach to motion-planning", 1983 ACM pp. 207-220.

[FHL] M. Furst, J. Hopcroft, E. Luks, "Polynomial-time algorithms for permutation groups", 21st FOCS, 1980, pp. 36-41.

[HJW1] J.E. Hopcroft, D.A. Joseph, S.H. Whitesides, "Movement problems for 2-Dimensional linkages", Tech. Rep. 82-515, Comp. Sci. Dept., Cornell Univ., August 1982.

[HJW2] J.E. Hopcroft, D.A. Joseph, S.H. Whitesides, "On the movement of robot arms in 2-dimensional bounded regions", Tech. Rep. 82-486, Comp. Sci. Dept., Cornell Univ., March 1982. Also, 23rd FOCS, 1982, pp. 280-289.

[HJW3] J.E. Hopcroft, D.A. Joseph, S.H. Whitesides, "Determining points of a circular region reachable by joints of a robot arm", Tech. Rept. 82-516, Comp. Sci. Dept., Cornell Univ., October 1982.

[J] Mark Jerrum, "The Complexity of Finding Minimum-Length Generator Sequences", Internal Report CSR-139-83, Dept. of Comp. Sci., Univ. of Edinburgh, Aug. 1983.

[M] Pierre McKenzie, "Permutations of Bounded Degree Generate Groups of Polynomial Diameter", manuscript, Dept. of Comp. Sci., Univ. of Toronto, Jan. 1984.

[R] J. Reif, "Complexity of the mover's problem and generalizations", 20th FOCS, 1979, pp. 421-427.

[SS1] J.T. Schwartz and M. Sharir, "On the piano movers' problem: I. The special case of a rigid polygonal body moving amidst polygonal barriers", to appear in Comm. Pure Appl. Math.

[SS2] J.T. Schwartz and M. Sharir, "On the piano movers' problem: II. General techniques for computing topological properties of real algebraic manifolds", to appear in Adv. Appl. Math.

[SS3] J.T. Schwartz and M. Sharir, "On the piano movers' problem: III. Coordinating the motion of several independent bodies: The special case of circular bodies moving amidst polygonal barriers", Tech. Rept., Courant Institute, N.Y.U., 1983.

[SY] Paul Spirakis and Chee Yap, "On the combinatorial complexity of motion coordination", Tech. Rept., Courant Institute, N.Y.U., April 1983.

[SY1] Paul Spirakis and Chee Yap, "Moving many pebbles in a graph is polynomial time", Tech. Rept., Courant Institute, N.Y.U., September 1983.

[Wie] Wielandt, "Finite permutation groups", Academic Press, New York, 1964.

[W] R.M. Wilson, "Graph puzzles, homotopy, and the alternating group", Journal of Comb. Theory (B) 16, 86-96 (1974).

[Y] C.K. Yap, "Motion coordination for two discs", Tech. Rept., Courant Institute, N.Y.U., January 1983.