# Useful Results for Pebbling in Diameter Two Graphs

Charles Cusack and Airat Bekmetjev

March 5, 2008

## 1 Introduction

Let $G$ be a connected graph with the vertex set $V$ and the edge set $E$, where $|V| = n$ and $|E| = m$. Define a *pebbling configuration* as a function $C : V \to Z^+$ where $C(v)$ represents the number of pebbles placed on vertex $v$. For any vertex $v$ such that $C(v) \geq 2$ a *pebbling step* consists of placing a pebble on one of the vertices adjacent to $v$ and discarding two pebbles from $v$. A configuration is called *$r$-solvable* if there is a sequence of pebbling steps that places at least one pebble on vertex $r$. Any such sequence is called an *$r$-solution*. A configuration is called *solvable* if it is $r$-solvable for any $r \in V$. We call an $r$-solution *minimal* if it contains the smallest number of pebbling steps.

The *pebbling number* of a graph $G$, denoted $\pi(G)$, is the minimum number of pebbles such that the configuration is solvable no matter how the pebbles are distriibuted on the vertices.

For any two vertices $u, v \in V$, the *distance* between $u$ and $v$ (denoted $d(u, v)$) is the the number of edges on the shortest path from $u$ to $v$. The *diameter* of a graph is the maximum distance between any pair of vertices in a graph. For a subset $S \subseteq V$, let $d_{\min}(v, S)$ be the smallest distance between vertex $v$ and any other vertex in $S$. Let

$$D_m(S) = \{v \in V \setminus S, \text{ such that } d_{\min}(v, S) = m\}$$

In particular, for any vertex $r \in V$, $D_1(r)$ is the set of vertices adjacent to $r$.

Let $C_m = \{v \in V \,|\, C(v) \geq m\}$. Clearly, a pebbling move is possible from any vertex in $C_2$. Let $\mathcal{G}_2$ represent the set of all graph of diameter 2. For the remainder of the paper, we shall assume all graphs are diameter two unless otherwise stated.

The purpose of this paper is to present the reader with several ways of thinking about pebbling in diameter two graphs. More specifically, we hope that new algorithms for determining the solvability of diameter two graphs will be found based on one or more of these ways of thinking about diameter two graphs.

We begin in Section 2 by providing a few elemetary results, and we give a simple, but inefficient, algorithm that determines whether or not a configuration is solvable in Section 3. In Section 4 we discuss a few results related the connectivity of a graph. Section 5 is devoted to a discussion of the concept of pebbling islands and how this might be useful in determining solvability of diameter two graphs. Finally, in Section 6 we discuss several facts that might be used to determine the $r$-solvability of a graph by partitioning the graph into 3 tiers.

## 2 Basic Results

**Lemma 2.1** *For any graph $G$, $\pi(G) \geq \max\{n, 2^{diam(G)}\}$.*

*Proof:* If there are less than $n$ pebbles, one can place each pebble on a different vertex, leaving at least one vertex with no pebbles and no moves possible, so $\pi(G) \geq n$. Also, let $u$ and $v$ be vertices in $G$ such that $d(u, v) = diam(G)$. Then if $2^{diam(G)} - 1$ pebbles are placed on $u$, and no other pebbles are placed on the graph, it is impossible to move a pebble onto $v$. Therefore, $\pi(G) \geq 2^{diam(G)}$. ■

**Lemma 2.2** *Let $K_n$ be the complete graph on $n$ vertices. Then $\pi(K_n) = n$.*

*Proof:* Let $C$ be a configuration with $n$ pebbles. If some vertex has two pebbles, then any vertex can be pebbled since every vertex is adjacent to every other vertex. If no vertex contains two pebbles, then every vertex contains one, and the graph is solvable. By Lemma 2.1, $\pi(K_n) \geq n$, and we have shown any configuration of $n$ pebbles is solvable. Thus $\pi(K_n) = n$. ■

**Lemma 2.3** *Let $S_n$ be the star on $n$ vertices (the graph in which one vertex is connected to the other $n - 1$ vertices, and there are no other edges). Then $\pi(S_n) = n + 1$.*

*Proof:* Let $c$ be the center of the graph and $r$ and $v$ be any other vertices. If we place 0 pebbles on $c$ and $r$, three on $v$, and one on the remaining vertices, it is easy to see that no pebble can be moved to $r$, and that a total of $n$ pebbles are on the graph. Thus $\pi(S_n) > n$.

Let $C$ be a configuration with $n+1$ pebbles. If every vertex has one pebble, $C$ is clearly solvable. Assume some vertex $r$ does not contain a pebble. Then some vertex $v$ contains two pebbles. If $r$ is the center of the star, a pebble can be moved from $v$ to $r$. If $r$ is not the center, there are two cases to consider. In the first case, the center contains at least one pebble, in which case a pebble can be moved from $v$ to the center to $r$. On the other hand, if the center contains no pebble, there are at least two vertices with no pebbles. In this case, there are either two vertices with two or more pebbles, or one vertex with four or more. In either case, two pebbles can be moved to the center, and then one to $r$. Thus, $\pi(S_n) = n + 1$. ■

We omit the proof for the following.

**Lemma 2.4** *Let $P_n$ be the path on $n$ vertices. Then $\pi(P_n) = 2^{n-1}$.*

In the exercises, we ask the reader to compute/prove the pebbling number of wheels, the Lemke graph, and the Petersen graph.

The following is a trivial but important result about pebbling in diameter two graphs.

**Lemma 2.5** *Let $G \in \mathcal{G}_2$ have a configuration $C$ such that it is possible to place 4 pebbles on some vertex. Then $C$ is a solvable configuration.*

For a more complete introduction to graph pebbling and recent results in graph pebbling, see [1].

# 3 Basic Algorithm

In this section, we assume that $G$ is a graph with vertices labeled $0, 1, \ldots, n - 1$, and $C$ is a configuration of pebbles associated with the graph. $C$ might simply be an array of integers such that $C[i]$ is the number of pebbles on vertex $i$. We can implement a method AdjacentPebble$(u, v)$ on the configuration which performs a pebbling step from $u$ to $v$ (by removing two pebbles from $u$ and adding one to $v$) and returns the resulting configuration $C'$, assuming $u$ and $v$ are adjacent,

and that $u \in C_2$. This can be implemented in several ways–we can either assume that $u$ and $v$ are connected (because we checked before we called the method), or we can pass the graph $G$ into the method and check. Similarly, we can either assume that $u$ has at least two pebbles, or we can check before we make the move. Below, we assume that none of this checking is done in the method, leaving it to the user to verify that these two conditions are true before calling the method. This is done for several reasons–first, it makes the method more efficient. Second, any reasonable algorithm from which we call this method should be only attempting legal pebbling moves, so the check should be unnecessary. We leave it to the reader to verify that this is the case in the following algorithms.

It should be clear that, in theory, AdjacentPebble can be implemented so that it takes constant time. However, the configuration it returns is a copy of the original, and it takes $O(n)$ time to make the copy.

Algorithm 3.1 (which uses Algorithm 3.2) is a simple exhaustive search algorithm that determines whether or not a graph is solvable by making all possible pebbling moves in lexicographical order. The algorithm maintains a set $L$ of vertices which can be pebbled, returning *true* if this set ever has $n$ elements on it (since then all of the vertices have been covered), and returning *false* if all possible moves have been exhausted and some vertex remains unpebbled. It is important to note that for the sake of efficiency, $G$ and $C'$ should be passed by reference, not by value.

For each recursive call of the algorithm, the graph has one less pebble, and since every vertex is adjacent to at most $n-1$ other vertices, the worst case time required to determine the solvability of a graph with $t$ pebbles is

$$T(t) = (n-1)(T(t-1) + O(n)) + O(n) = (n-1)T(t-1) + O(n^2),$$

where $T(1) = O(n)$. The reader is asked to prove in the exercises that the solution to the recurrence relation is $T(t) = O(n^t)$. Since $t$ might depend on $n$, this algorithm is not polynomial time in general. The reader is asked to verify that there is a polynomial-time algorithm to determine the solvability of some classes of graphs, including $K_n$ and $S_n$.

**Algorithm 3.1:** IsSolvableWrapper$(G, C)$

```
global Set L
for u ← 0 to n − 1
  do { if u ∈ C_1
       then add u to L
if |L| = n
  then return (true)
  else return (IsSolvable(G, C))
```

When implementing an exhaustive search algorithm like Algorithm 3.2, it is important that pebbling moves can be undone so that different moves can be tried. There are generally two ways to accomplish this. In Algorithm 3.2 we chose to simply copy the data structure (in this case, just the configuration $C$, since the graph doesn't change) every time a change is made.

Another technique is to store the moves that are made so they can be undone and the next move tried. Often this involves the use of a stack, although if recursion is involved, a stack may not be necessary. Algorithm 3.3 is identical to Algorithm 3.2 except that it uses this approach. In this case, we use AdjacentPebble2$(u, v)$, which works the same way as AdjacentPebble$(u, v)$ except

rather than returning a copy of the configuration after the move, it simply modifies the configuration itself and returns nothing. Then, after the recursive call is make, a call to UndoPebble$(u, v)$ is made to undo the move.

We leave it to the reader to decide which of these algorithms is preferable.

**Algorithm 3.2:** IsSolvable$(G, C)$

**comment:** Determine first vertex with at least 2 pebbles

$u \leftarrow 0$
**while** $u < n$ **and** $u \notin C_2$
  **do** $u \leftarrow u + 1$
**if** $u = n$
  **then return** $(false)$

**comment:** Now try all possible moves from $u$

**for each** $v \in D_1(u)$
  **do**
    add $v$ to $L$
    **if** $|L| = n$
      **then return** $(true)$
    $C' =$ C.AdjacentPebble$(u, v)$
    **if** IsSolvable$(G, C')$
      **then return** $(true)$
**return** $(false)$

**Algorithm 3.3:** IsSolvable2$(G, C)$

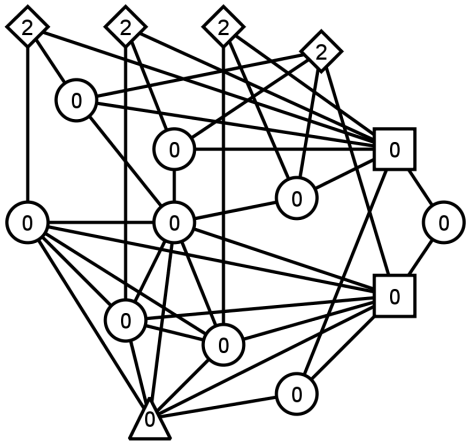
**comment:** Determine first vertex with at least 2 pebbles

$u \leftarrow 0$
**while** $u < n$ **and** $u \notin C_2$
  **do** $u \leftarrow u + 1$
**if** $u = n$
  **then return** $(false)$

**comment:** Now try all possible moves from $u$

**for each** $v \in D_1(u)$
  **do**
    add $v$ to $L$
    **if** $|L| = n$
      **then return** $(true)$
    C.AdjacentPebble2$(u, v)$
    **if** IsSolvable2$(G, C)$
      **then return** $(true)$
    C.UndoPebble$(u, v)$
**return** $(false)$

Figure 1: An unsolvable configuration for a graph from $\mathcal{G}_{2,2}$ with $|C_2| = 4$ .




# 4 Connectivity

The *connectivity* of a graph is the minimal number of vertices that must be removed from a graph in order to make it unconnected. For instance, if a graph has connectivity 3, then there is some set of three vertices whose removal will cause the graph to become unconnected, and there is no set of 2 vertices whose removal will disconnect the graph. A set of vertices whose removal causes a graph to become disconnected is called a *cut set*.

We leave the proof of the following as exercises.

**Lemma 4.1** *Let $G \in \mathcal{G}_2$ and let $Q$ be a cut set of vertices in $G$. Then each vertex in $V \setminus Q$ is adjacent to a vertex in $Q$.*

**Corollary 4.2** *Let $G \in \mathcal{G}_2$ and let $Q$ be a cut set of vertices in $G$. If at least two pebbles can be placed on each of the vertices in $Q$, then the configuraton is solvable.*

Let $\mathcal{G}_{2,k} \subset \mathcal{G}_2$ be the subset of diameter two graphs which are $k$-connected. We are going to establish two upper bounds on the size of $C_2$ for members of $\mathcal{G}_{2,k}$. The first bound is based on Corollary 4.2 and uses vertices from a vertex cut set to perform pebbling steps.

**Lemma 4.3** *Let $G \in \mathcal{G}_{2,k}$. Then a configuration $C$ is solvable if $|C_2| \geq 3k - 1$.*

*Proof:* Assume that $|C_2| \geq 3k-1$, but that $C$ is unsolvable. Let $Q$ be a minimal cut set of $k$ vertices. Since $C$ is unsolvable, none of the cut vertices can accumulate 4 or more pebbles, and by corollary 4.2 at least one of the cut vertices can only accumulate 1. Thus, at most $3(k-1)+1 = 3k-2$ pebbles can be accumulated on the cut. However, lemma 4.1 implies that it is possible to accumulate at least $3k-1$ pebbles onto the $k$ cut vertices (more if $C_2 \cap Q$ is not empty). ■

This result is tight for $k = 1, 2$. It is easy to check that if $k = 1$, then there are configurations with $|C_2| = 1$ which are unsolvable. For $k = 2$, Figure 1 gives a $2-$connected graph with $|C_2| = 4$ which is not solvable.

We end this section with a result that provides a better lower bound on the size of $C_2$ for $k \geq 3$. We will omit the proof. The tightness of this bound is unknown for all values of $k$.

**Theorem 4.4** *Let $G \in \mathcal{G}_{2,k}$, where $k \geq 3$. Then a configuration $C$ is solvable if $|C_2| \geq k + 4$.*

# 5 Islands and Bridges

An $i-$*island* is a maximal connected subgraph of a graph in which every vertex has at least one pebble and one vertex has at least $i$ pebbles. Given a set of integers $\mathbf{b} = \langle b_1, b_2, ..., b_m \rangle$, a $\mathbf{b}$-*island* is an island that contains distinct vertices $v_1, v_2, ..., v_m$ such that $C(v_j) \geq b_j, 1 \leqslant j \leqslant m$. If every vertex on an island contains precisely one pebble, we call it a *desert*. A *bridge* is a vertex which has no pebbles. Clearly, every vertex of a graph is either a bridge or a member of a single island.

We say a bridge is *filled* if two pebbles are moved onto it and *emptied* if two pebbles are removed from it. Note that any bridge that is used in a minimal $r$-solution, except $r$, must be filled and emptied. Milans et al. [2] showed that, for any vertex $r$, a minimal $r$-solution contains no directed cycle.

A vertex of $G$ that is adjacent to at least one vertex of a subgraph $G'$ is called *adjacent to* $G'$. Note that a vertex containing two or more pebbles allows the movement of at least one pebble to any other vertex on the island or any bridge adjacent to the island. We say that a bridge has *potential* $s$ if $s$ pebbles can be moved to it from adjacent islands.

## 5.1 Results Using Islands

In this section we present a few results about pebbling moves on graphs with diameter two. Most of these are given without proof. The reader should attempt to verify the correctness of each of the results.

**Lemma 5.1** *Let $C$ be a configuration which contains an island $I$ with at least 3 more pebbles than vertices. Then $C$ is solvable.*

*Proof:* If $I$ contains three or more pebbles than vertices, then $I$ is a 4-island, a $\langle 2, 3 \rangle$-island, or a $\langle 2, 2, 2 \rangle$-island. Clearly, a 4-island already guarantees solvability. Given a $\langle 2, 3 \rangle$-island, we can move a pebble from the vertex containing two pebbles to the vertex containing three pebbles, creating a solvable configuration. If the island is a $\langle 2, 2, 2 \rangle$-island, let $a$, $b$, and $c$ be vertices on $I \cap C_2$. Consider any path $P$ on the island between $a$ and $b$. If $c \in P$ we can accumulate 4 pebbles on $c$. Otherwise choose a shortest path $P'$ on the island between $c$ and any vertex on $P$. Let $u = P \cap P'$. If $u = a$ (or $u = b$), we can accumulate 4 pebbles on $u$ by moving one from $b$ (or $a$), and the other from $c$. If $u$ is different from $a$ and $b$, we can accumulate 4 on $u$ by moving one from each of $a$, $b$, and $c$. ■

Thus, the only islands that can be present in an unsolvable configuration are deserts, 2-islands (which may be 3-islands), and $\langle 2, 2 \rangle$-islands. In the light of Lemma 5.1, we call an island $I$ which contains at least 3 more pebbles than vertices an *empire*.

**Lemma 5.2** *If $v$ is a bridge adjacent to a 2-island, then one pebble can be added to $v$.*

**Corollary 5.3** *If every bridge is adjacent to a 2-island, then the configuration is solvable.*

**Lemma 5.4** *Let $v$ be a vertex on a 2-island. Then one pebble can be added to $v$ unless it is the only vertex which contains 2 or more pebbles.*

**Lemma 5.5** *Let $u$ and $v$ be adjacent bridges and let $u$ be adjacent to two 2-islands. Then one pebble can be added to $v$.*

**Lemma 5.6** *Let $u$ and $v$ be any two vertices on a desert. Then one pebble can be added to $u$ iff one pebble can be added to $v$.*

**Lemma 5.7** *Let $I$ be an island in a configuration containing $s$ more pebbles than vertices. Then in any minimal $r$-solution, at most $3 - s$ pebbles are ever added to $I$.*

**Lemma 5.8** *Let $C$ be a configuration. Then in any minimal $r$-solution, at most one vertex is pebbled from twice.*

**Corollary 5.9** *Let $C$ be a configuration. Then in any minimal $r$-solution, at most one vertex $v \in C_2$ is pebbled to. Further, if such a vertex exists and $v \notin C_3$, then it is pebbled to exactly twice, and in either case, it is pebbled from exactly twice.*

*Proof:* In a minimal $r$-solution, the only reason to add a pebble to a vertex in $C_2$ or $C_3$ is in order to pebble from it twice. According to Lemma 5.8, there is at most one such vertex. If we are to pebble from it twice, we must pebble to it twice if $v \notin C_3$. ∎

For any vertex subset $M \subseteq V$ the *surplus of* $M$ is

$$s_M(C) = \sum_{v \in M} \lfloor C(v)/2 \rfloor$$

The surplus of an island represents the maximum number of pebbles that can be moved onto adjacent bridges. Note that if a pebbling step moves a pebble onto a vertex containing an even number of pebbles, the surplus of the graph decreases by one, and if it moves a pebble onto a vertex containing an odd number of pebbles, the surplus of the graph remains unchanged. In particular, a pebbling move from an island to a bridge always reduces the surplus by one.

**Lemma 5.10** *Let $C$ be a configuration with surplus $s$, and let the surplus be $s'$ after a minimal $r$-solution. Then $s - s'$ or $s - s' - 1$ different bridges, including the root, are used in the pebbling.*

*Proof:* Clearly the $r$-solution reduced the surplus by $s - s'$. Since the surplus is only reduced by a move onto a vertex with an even number of pebbles, the only way to reduce the surplus is to pebble to a bridge or a vertex with exactly two pebbles (that is, in $C_2 \setminus C_3$), since if a vertex has 4 or more pebbles, there is no reason to move any more onto it. According to Corollary 5.9, at most one vertex in $C_2$ is pebbled to.

If one move is made onto a vertex $v \in C_2 \setminus C_3$, then by Corollary 5.9, it is the unique vertex that is pebbled from twice. Therefore, $s - s' - 1$ moves were made onto $s - s' - 1$ different bridges in $C$, including the root.

If no moves are made onto a vertex $v \in C_2 \setminus C_3$, then $s - s'$ moves were made onto bridges. If one of these bridges is pebbled from twice, then $s - s' - 1$ different bridges were used. Otherwise, $s - s'$ different bridges were used. ∎

**Corollary 5.11** *Let $C$ be a solvable configuration with surplus $s$. Then at most $s$ bridges are needed to pebble to any root, including the root itself.*

Corollary 5.11 may help lead to an efficient algorithm to determine the solvability of diameter two graphs, since it gives us an upper bound on the number of bridges that can possibly be used pebble to any root in a graph.

**Lemma 5.12** *Let $C$ be a configuration with exactly one vertex $u$ with 2 or more pebbles. Then $C$ is solvable iff every bridge is adjacent to the 2-island or $u$ contains at least 4 pebbles.*

**Lemma 5.13** *Let $C$ be a configuration on $G$ with exactly two vertices with 2 or more pebbles. Then $C$ is solvable iff one of the following is true:*

1. *$G$ contains an empire, or*
2. *Every bridge is*
   (a) *adjacent to a 2-island,*
   (b) *adjacent to a bridge with potential 2, or*
   (c) *adjacent to desert which is adjacent to a bridge with potential 2.*

*Proof:* The reverse implication is easy to check. For the forward implication, assume $C$ is solvable, contains no empires, and that some bridge $r$ is not adjacent to a 2-island or a bridge with potential 2. We need to show that $r$ is adjacent to a desert which is adjacent to a bridge with potential 2. Notice that $G$ contains either two 2-islands (with one or both possibly being a 3-island) or a $\langle 2,2\rangle$-island. In either case, it is clear that the graph has surplus 2. Thus, at most two bridges, including $r$, can be used to pebble $r$. Since $r$ is not adjacent to either of the islands, a second bridge, $u$, must be used to pebble to $r$. Since $u$ is the only other bridge used, it must be filled from the two 2-islands or the $\langle 2,2\rangle$-island, which means it has potential two, and is not adjacent to $r$. Filling $u$ reduces the potential of the graph to 1. Thus, $u$ is the only vertex with more than 1 pebble on it. Since $C$ is solvable, there must be a move from $u$ to $r$. The only possibility is that both $u$ and $r$ are adjacent to a desert. ■

The following result may be helpful in creating an efficient algorithm to determine the solvability of diameter two graphs.

**Lemma 5.14** *Let $C$ be a solvable configuration which contains $k$ vertices with 2 or more pebbles, where $k \geq 2$. If $C$ contains no empire and there is some vertex $r$ which is not adjacent to a 2-island, then some bridge with potential 2 is needed in the pebbling of $r$.*

*Proof:* Since $C$ contains no empire, $C$ consists of bridges, deserts, and 2-islands (which might be 3-islands or $\langle 2,2\rangle$-islands). Assume that no bridge with potential 2 is needed to pebble to $r$.

If $r$ is pebbled from an adjacent desert, then some vertex on the desert was pebbled from a bridge. Otherwise $r$ is pebbled from an adjacent bridge. In either case there a bridge which must be filled in order to pebble to $r$.

Let $u$ be the first bridge that is filled while pebbling to $r$. Then $u$ received at least one pebble from an adjacent bridge or desert, since it cannot have potential 2. If one pebble comes from a desert adjacent to $u$, then some vertex on the desert received a pebble, which must come from an adjacent filled bridge. Otherwise, one pebble must come from a bridge adjacent to $u$ which is filled. In either case, this contradicts the fact that $u$ was the first filled bridge. ■

## 5.2 Algorithms

The goal of this section is to present several algorithms which might prove useful in the construction of a polynomial-time algorithm to determine the solvability of all diameter two graphs.

As before, we will let $G \subset \mathcal{G}_{2,k}$ be a graph with $|V| = n$ and $|E| = m$, we will label the vertices $0, 1, \ldots, n-1$. The next result by Shiloach [3] is used to determine the pebbling solvability in the presence of a $\langle 2, 2\rangle$-island.

**Theorem 5.15** *[3] For any distinct vertices $s_1$, $s_2$, $t_1$, and $t_2$, it can be determined in $O(nm)$ time whether or not $G$ has two vertex-disjoint paths connecting $s_1$ to $t_1$ and $s_2$ to $t_2$.*

**Corollary 5.16** *Let $I$ be a $\langle 2, 2\rangle$-island.*

1. *For any two distinct vertices $u, v \in D_1(I)$, it can be determined in $O(nm)$ time whether or not both $u$ and $v$ can be pebbled from $I$ simultaneously.*
2. *For any $u \in D_1(I)$, it can be determined in $O(n^3m)$ time whether $u$ can be filled from $I$.*

*Proof:* Let $G'$ be a subgraph of $G$ induced by the vertex set $I \cup u \cup v$. Let $s_1$ and $s_2$ be elements of $C_2 \cap I$. Then the first part of the theorem follows from Result 5.15 by determining disjoint paths connecting $s_1$, $s_2$, $t_1 = u$ and $t_2 = v$. This clearly takes time $O(nm)$.

For the second part, notice that if $u$ is only adjacent to one vertex in $I$, then it can be filled from $I$ iff $I$ is an empire. If $I$ is not an empire then $u$ can be filled from $I$ iff there are disjoint paths from $u$ to $s_1$ and $s_2$. In this case, given any $u \in D_1(I)$, we can apply Theorem 5.15 to each pair of vertices $t_1, t_2 \in I$ which are both adjacent to $u$ until the answer is "yes" or we exhaust them all (and answer "no"). Since there are no more than $n-1$ vertices adjacent to $u$, we must check at most $\binom{n-1}{2} = O(n^2)$ pairs of vertices, so this will take time $O(n^3m)$. ∎

The following list contains descriptions of several basic procedures that can be easily implemented on a graph data structure, and might be useful. Since islands are tied to both the structure of the graph and the configuration of pebbles, we will assume that the configuration is stored as part of the graph data structure, and we will refer to this data structure as the *graph configuration*. We assume the islands are maintained as part of the graph configuration as well, and that each vertex can determine which island it resides on in constant time (since each vertex can store a reference to its island).

Notice that whenever a pebbling move is performed, the islands must be updated, since a single pebbling move might significantly change the configuration of islands.

- UpdateIslands() updates the data structure representing the islands. This can be implemented using a BFS/DFS algorithm in time $O(n+m)$. In addition, it is possible to compute and store the surplus and the number of "extra" pebbles an island contains (these are not quite the same), as well as having each vertex store which island it belongs to, without changing the complexity. We assume that this information about islands is stored and maintained.
- GetIslandContaining($u$) returns the vertex set of the island $I$ such that $u \in I$. Can be implemented in $O(1)$ time.
- ContainsEmpire() returns TRUE iff $G$ contains an empire. Can be implemented in $O(n)$ time.
- IsOnTwoIsland($u$) returns TRUE iff $u$ is on a $2-$island. Can be implemented in $O(1)$ time.
- IsOnTwoTwoIsland($u$) returns TRUE iff $u$ is on a $\langle 2, 2\rangle$-island. Can be implemented in $O(1)$ time.

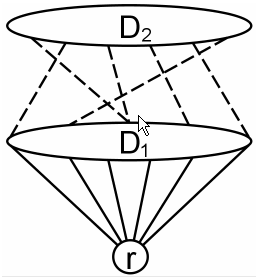
- IsAdjacentTwoIsland($u$) returns TRUE iff $u$ is adjacent to a 2−island. Can be implemented in $O(n)$.
- AdjacentPebble($u, v$) performs a pebbling step from $u$ to $v$ and returns the resulting graph configuration, assuming $u$ and $v$ are adjacent, and that $u \in C_2$. Takes $O(n+m)$ time since we need to call UpdateIslands(). The actual pebbling move takes $O(1)$ time.
- PebbleFromIsland($S, v$) performs a sequence of pebbling steps required to move a pebble from $S$ to $v$ using only vertices in $S$ in intermediate steps, assuming $v \in D_1(S)$ and $S$ is a 2-island. Returns the resulting graph configuration. Takes $O(n+m)$ time since we need to call UpdateIslands(). The actual pebbling move takes $O(n)$ time.
- CanDoublePebble($C', S, u, v$) returns TRUE iff pebbles can be moved from $S$ (assuming $S$ is a $\langle 2, 2\rangle$-island) to vertices $u$ and $v$ (where $u = v$ is possible) simultaneously, using only vertices from $S$ in intermediate moves. Assigns to $C'$ the resulting graph configuration. Takes time $O(n^3 m)$.

The last three methods can be implemented so that they either modify the graph configuration, or return a copy of the graph configuration after the move, leaving the original configuration unchanged. As in Section 3, which choice is made has an impact on the algorithms that call these methods. It should be noted that undoing a move from PebbleFromIsland or CanDoublePebble is not as easy as undoing AdjacentPebble since the exact set of pebbling moves made is uncertain unless we somehow store them. However, the beauty of the islands and bridges representation of diameter two graphs is that we can make moves from islands without worrying about exactly what moves are made, so having to keep track of the exact moves made in order to undo them is unappealing.

# 6 3-Tier Representation

In this section we examine a few results related to the *3-tier representation* of diameter two graphs. Let $r$ be a root we wish to pebble to. Notice that $|D_k(r)|=0$ for $k \geq 3$ for diameter two graphs. This allows us to partition the graph into 3 tiers: the root ($r$), the vertices adjacent to the root ($D_1(r)$), and the vertices not adjacent to the root ($D_2(r)$)–See Figure 2.

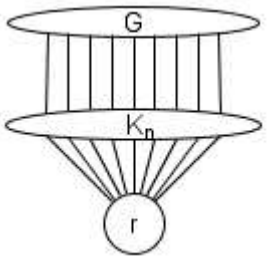
Figure 2: The 3-tier view of a diameter two graph




**Lemma 6.1** *Let $C$ be an unsolvable configuration and $r$ be the root vertex.*

1. $C_4 = \emptyset$.

Figure 3: Diagram of Lemma 6.4




2. $C_2 \cup C_3 \subseteq D_2(r)$.

**Lemma 6.2** *Let $u \in D_2(r)$ and $v \in V \setminus r$. Then at least one of the following is true:*

1. *$u$ and $v$ are adjacent.*
2. *$u$ and $v$ are both adjacent to some $w \in D_1(r)$.*
3. *$u$ and $v$ are both adjacent to some $w \in D_2(r)$.*

**Lemma 6.3** *Let $C$ be an unsolvable configuration.*

1. *No two vertices in $C_2$ are adjacent to the same vertex in $D_1(r)$.*
2. *No vertex in $C_2$ is adjacent to a vertex in $D_1(r) \cap C_1$.*
3. *Every vertex in $C_2$ is adjacent to at most 1 other vertex in $C_2$.*
4. *No vertex in $C_3$ is adjacent to any vertex in $C_2$.*
5. *For every pair of vertices $u \in C_3$ and $v \in C_2$, there is a different empty vertex in $D_2(r)$ which is adjacent to both $u$ and $v$.*
6. *Let $\{x, y, z\} \subset C_2$ be three distinct vertices such that there is some $p$ which is adjacent to $x, y$ and $z$. Then for any other triple of distinct points $\{u, v, w\} \subset C_2$ all of which are adjacent to some $q$ then $|\{x, y, z\} \cap \{u, v, w\}| \neq 1$.*

The following lemma can be easily proven, and can be useful in constructing unsolvable configurations. See Figure 3 for a graphical representation of the theorem.

**Lemma 6.4** *Let $G = (V, E)$ be a diameter 2 graph with $|V| = n$. Let $U$ be a set of $n$ vertices, and label the elements of $U$ (resp. $V$) as $u_1, \ldots, u_n$ ($v_1, \ldots, v_n$). Let $G' = (V', E')$ be the graph with $V' = V \cup U \cup r$, and $E' = E \cup \{(u_i, v_i)\}_{i=1}^n \cup \{(u_i, u_j)\}_{i \neq j} \cup \{(u_i, r)\}_{i=1}^n$. Then $G'$ is diameter two.*

Notice that $U = D_1(r)$ and $V = D_2(r)$, and if we place zero pebbles on $r$ and all vertices in $D_1(r)$, then $G$ is solvable if and only if we can place 4 pebbles on some vertex in $V$. We can use this fact to construct diameter two graphs which are not solvable by just constructing $D_2(r)$ such that no vertex can accumulate 4 pebbles.

# 7 Exercises

1. Determine $\pi(W_n)$, where $W_n$ is the *wheel graph* on $n \geq 4$ vertices. ($W_n$ is a cycle of $n-1$ vertices with 1 vertex in the center connected to each of the $n-1$ vertices on the cycle.) Prove your assertion. (Hint: To prove that the pebbling number is $k$, you have to show that there is some configuration of $k-1$ pebbles which is unsolvable, and that *any* distribution of $k$ pebbles is solvable.)

2. Prove that $\pi(L) = 8$, where $L$ is the *Lemke graph*. (Hint: See hint for exercise 1.)

3. Prove that $\pi(P) = 10$, where $P$ is the *Petersen graph*. (Hint: See hint for exercise 1.)

4. Prove that the solution to the recurrence relation $T(t) = (n-1)T(t-1)+O(n^2)$, $T(1) = O(n)$, is $O(n^t)$. Hint: Notice that the recurrence depends on $t$, not $n$.

5. Give a polynomial-time algorithm to determine whether a configuration of pebbles on $K_n$ is solvable. Do the same for $S_n$. Give the computational complexity of each algorithm.

6. Prove Lemma 2.5.

7. What type of algorithm (brute force, divide-and-conquer, decrease-and-conquer, etc.) is Algorithm 3.2?

8. In Algorithm 3.2, how do we know that the call to AdjacentPebbble$(u, v)$ is legal? That is, how do we know that $u$ and $v$ are adjacent, and that $u$ has at least two pebbles?

9. Asymptotically, is Algorithm 3.3 any faster than Algorithm 3.2?

10. Prove Lemma 4.1. Hint: Use the fact that $G$ is diameter two.

11. Prove Corollary 4.2. Hint: Use Lemma 4.1.

12. Prove Lemma 5.5.

13. Prove Lemma 5.12.

14. Prove Lemma 6.3.

15. Prove that there is no limit on the number of islands a graph of diameter two can have. In other words, for any positive integer $k$, show that there is some diameter two graph with a configuration of pebbles which contains $k$ islands.

16. Prove that there is no limit on the number of islands an *unsolvable* graph of diameter two can have.

17. Give an algorithm for ContainsEmpire$(G)$ and verify that its complexity is $O(n)$. Use the basic graph operations and the procedures from Section 5.2.

18. Give an algorithm for IsAdjacentTwoIsland$(G, u)$ and verify that its complexity is $O(n)$. Use the basic graph operations and the procedures from Section 5.2.

19. Implement Algorithms 3.1 (and 3.2) and determine how large $n$ and $t$ can be until the algorithm becomes unreasonably slow. (Hint: For increasing values of $n$ and $t$, generate a random graph on $n$ vertices, and then randomly place $t$ pebbles on the vertices and run the algorithm.

20. Come up with an algorithm that is more efficient than Algorithm 3.1. Give and prove estimates on the running time of the algorithm. Does your algorithm run in polynomial time? (Hint: The answer to the last question is almost certainly "no".)

21. Come up with an algorithm to determine the solvability of diameter two graphs. Give and prove estimates on the running time of the algorithm. Does your algorithm run in polynomial time?

22. It is possible to parallelize Algorithm 3.1. Does this increase the size of the problems that can be solved significantly? Explain.

23. Can a randomized algorithm be used to determine the solvability of a configuration? Explain.

24. Explain how you can use simulations to determine upper and lower bounds on the pebbling number of a graph. Be precise. It is important to note that the answer may be different for upper and lower bounds. Think about what it means to be an upper bound versus what it means to be a lower bound.

# 8 Open Questions

There are several open questions which may help lead to an efficient (polynomial-time) algorithm to determine the solvability of diameter two graphs. We list a few here.

1. For what values of $k$ is the bound in Theorem 4.4 tight? In particular, if $k = 3$, is 6 enough, or do we need 7?

2. Is there a limit on the number of 2-islands an *unsolvable* configuration of diameter two can have?

3. Is there an upper bound on $|C_2|$ for an unsovable configuration? In other words, is there some constant $c$ such that if $|C_2| \geq c$, then $C$ is always solvable? (We think this may be true for $c = 8$, but have been unable to prove it. We can construct a configuration with $|C_2| = 6$ which is unsolvable, but have been unable to contruct any example with $|C_2| \geq 7$ which is unsolvable.)

4. Is it possible to prove that there is a maximum number of bridges needed to pebble to any root, or is this unbounded?

5. In a minimal $r$-solution, is there at least one island which is pebbled from but never pebbled to in a minimal pebbling? (If the answer to this is "yes", then a polynomial-time algorithms exists to determine the solvability of graphs in $G_{2,k}$, as long as $k$ is a constant.)

6. Is there a efficient algorithm to determine the solvability of graphs in $\mathcal{G}_{2,k}$ for fixed values of $k$? Or is this problem NP-Complete? (We think the answer is that there is an efficient algorithm.)

7. Is there an efficient algorithm to determine the solvability of graphs in $\mathcal{G}_2$ (diameter two graphs)? Or is this problem NP-Complete? (We think the answer is that there is an efficient algorithm, but are less certain about this one.)

## References


[1] G. Hurlbert. Recent progress in graph pebbling, 2005.

[2] K. Milans and B. Clark. The complexity of graph pebbling. *SIAM J. Discret. Math.*, 20(3):769–798, 2006.

[3] Y. Shiloach. A polynomial solution to the undirected two paths problem. *J. ACM*, 27(3):445–456, 1980.