SIAM J. DISCRETE MATH.
Vol. 23, No. 2, pp. 634–646

PEBBLING ALGORITHMS IN DIAMETER TWO GRAPHS*

AIRAT BEKMETJEV† AND CHARLES A. CUSACK‡

Abstract. Consider a connected graph and a configuration of pebbles on its vertices. A pebbling step consists of removing two pebbles from a vertex and placing one on an adjacent vertex. A configuration is called solvable if it is possible to place a pebble on any given vertex through a sequence of pebbling steps. A smallest number t such that any configuration with t pebbles is solvable is called the pebbling number of the graph. In this paper, we consider algorithms determining the solvability of a pebbling configuration on graphs of diameter two. We prove that if k is the vertex connectivity of a diameter two graph G, then a configuration is solvable if there are at least $c(k) = \min\{k+4, 3k-1\}$ vertices in G with two or more pebbles. We use this result to construct an algorithm that has complexity $O(c(k)! \cdot n^{2c(k)-3}m)$, where n is the number of vertices and m is the number of edges. We also present an algorithm for diameter two graphs with pebbling number $n+1$, known as Class 1 graphs, which takes $O(nm)$ time.

Key words. graph pebbling, diameter, connectivity, algorithms

AMS subject classifications. 05C85, 05C40, 68R05

DOI. 10.1137/080724277

1. Introduction.

Let G be a connected graph with vertex set V and edge set E, with $n = |V|$ and $m = |E|$. 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 with $C(v) \geqslant 2$, a *pebbling step* consists of placing one pebble on an adjacent vertex 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.

This paper considers an algorithmic approach to the pebbling problem. Watson [8] and Milans and Clark [6] showed that determining the solvability of a pebbling configuration on a general graph is an NP-complete problem. We will consider graphs of diameter two and show the existence of an algorithm whose running time depends on the vertex connectivity and the size of the graph. In particular, we will show that in a diameter two graph with connectivity k, any configuration that contains at least $c(k) = \min\{3k-1, k+4\}$ vertices with two or more pebbles is solvable. Based on this result, we will establish an algorithm that determines the solvability of a given configuration in $O(c(k)! \cdot n^{2c(k)-3}m)$ time, which is polynomial when k is constant.

We begin by presenting a backtracking algorithm (Algorithm 1.1, which uses Algorithm 1.2) that determines the solvability of a pebbling configuration on any graph. The method ADJACENTPEBBLE(u, v) performs a pebbling step from u to v, assuming that u and v are adjacent, and that $C(u) \geqslant 2$. UNDOPEBBLE(u, v) reverses

*Received by the editors May 13, 2008; accepted for publication (in revised form) November 3, 2008; published electronically February 6, 2009. This work was supported in part by a grant to Hope College from the Howard Hughes Medical Institute through the Undergraduate Science Education Program.

http://www.siam.org/journals/sidma/23-2/72427.html

†Department of Mathematics, Hope College, 27 Graves Place, Holland, MI 49422 (bekmetjev@hope.edu).

‡Department of Computer Science, Hope College, 27 Graves Place, Holland, MI 49422 (cusack@hope.edu).

a pebbling move. The algorithm maintains a set L of vertices which can be pebbled, returning TRUE if $|L| = n$ (since then all of the vertices have been covered), and returning FALSE if $|L| < n$ at the end of the algorithm.

The algorithm is based on the following ideas. If C is a solvable configuration, and r is a vertex with no pebbles, it follows from [6] that there is an acyclic r-solution, and all moves from valid vertices (i.e., vertices that contain at least two pebbles) can be made in an arbitrary order. Also, if M is an r-solution, then any pebbling sequence N with $M \subseteq N$ is also an r-solution.

Algorithm 1.1. IsSolvable(G, C).

```
global Set L
for u ← 0 to n − 1                          }
  do { if C(u) ⩾ 1                           } 1.1.1
     {   then add u to L                     }
if |L| = n
  then return (TRUE)
  else return (IsSolvableRecursive(G, C))
```

Algorithm 1.2. IsSolvableRecursive(G, C).

```
comment: Determine first vertex with at least 2 pebbles
u ← 0                                       }
while u < n and C(u) ⩽ 1                    }
  do u ← u + 1                              } 1.2.1
if u = n                                    }
  then return (FALSE)                       }

comment: Now try all possible moves from u
for each v adjacent to u
       { add v to L                                   }
       { if |L| = n                                   }
       {   then return (TRUE)                         }
       { C.AdjacentPebble(u, v)                       }
  do   { solvable = IsSolvableRecursive(G, C')        } 1.2.2
       { C.UndoPebble(u, v)                           }
       { if solvable                                  }
       {   then return (TRUE)                         }
return (FALSE)
```

Let $T(t)$ be the worst-case time it takes to determine the solvability of a graph with t pebbles using Algorithm 1.1, and let d be the maximum degree of G. Notice that step 1.1.1 in Algorithm 1.1 and step 1.2.1 in Algorithm 1.2 each take $O(n)$ time. A graph with a single pebble will end at step 1.2.1, so $T(1) = O(n)$. Step 1.2.2 executes at most d times, each time requiring $O(1)$ time plus making a recursive call on a graph with one fewer pebble. Thus,

$$T(t) = d(T(t-1) + O(1)) + O(n) = d \cdot T(t-1) + O(n) = O(nd^{t-1}).$$

Since t may depend on n, Algorithm 1.1 is not polynomial time in general. One source of inefficiency in this algorithm is that there may be many ways of moving a pebble from one vertex to another along paths of vertices which contain a single pebble, and it tries all of them. It turns out that, for graphs of diameter two with connectivity k, we can avoid such an exhaustive search.

2. Diameter and connectivity. Let $\mathcal{G}_2$ represent the set of all graphs of diameter two, and let $\mathcal{G}_{2,k} \subset \mathcal{G}_2$ be the set of diameter two graphs which have vertex connectivity k. It is clear that if Q is a vertex cut set in a diameter two graph, then any vertex in $V \backslash Q$ must be adjacent to at least one vertex in Q. This observation leads to the following.

LEMMA 2.1. *Let $G \in \mathcal{G}_2$ and let Q be a vertex cut set. Then a configuration C is solvable if it is possible to place at least two pebbles on each vertex in Q.*

Let $C_m = \{v \in V \mid C(v) \geqslant m\}$. Note that for a graph $G \in \mathcal{G}_2$, any configuration is solvable if C_4 is nonempty. We are going to establish two upper bounds on $|C_2|$ for members of $\mathcal{G}_{2,k}$ which are unsolvable.

LEMMA 2.2. *Let $G \in \mathcal{G}_{2,k}$. Then a configuration C is solvable if $|C_2| \geqslant 3k-1$.*

Proof. Let Q be a minimal cut set of G that contains k vertices. For any sequence of pebbling moves in an unsolvable configuration, none of the vertices in Q can accumulate four or more pebbles and, by Lemma 2.1, at least one vertex in Q can accumulate at most one. Therefore, at most $3(k-1)+1 = 3k-2$ pebbles can be placed on vertices in Q without the configuration being solvable. However, at least $3k-1$ pebbles can be placed on Q from C_2, and therefore C is solvable. ∎

The last result is tight for $k = 1, 2$. If $G \in \mathcal{G}_{2,1}$, then it has a vertex u that is not adjacent to all other vertices, and placing two pebbles on u and zero on all other vertices creates an unsolvable configuration. Further, Figure 2.1 represents a graph in $\mathcal{G}_{2,2}$ and a configuration C with $|C_2| = 4$, which is not solvable. As it is shown in the following result, the upper bound on $|C_2|$ can be improved for $k \geqslant 3$.

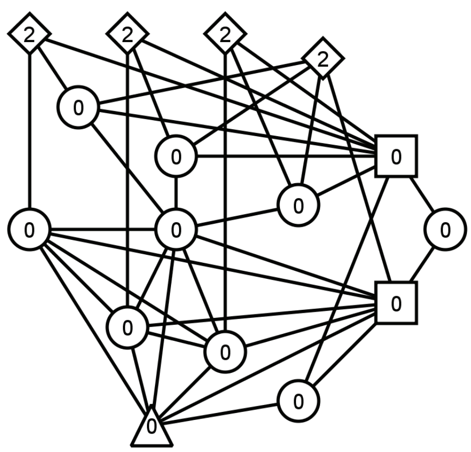

FIG. 2.1. *An unsolvable configuration for a graph from $\mathcal{G}_{2,2}$ with $|C_2| = 4$. The squares represent the cut set, the diamonds are the vertices containing two pebbles, and the triangle is the root.*

THEOREM 2.3. *Let $G \in \mathcal{G}_{2,k}$. Then a configuration C is solvable if $|C_2| \geqslant k+4$.*

Proof. Let C be an unsolvable configuration with $|C_2| = k+4+i$, where $i \geqslant 0$. Let Q be a vertex cut set of size k in G, $Q_2 = C_2 \cap Q$, and $Q_{0,1} = Q \setminus Q_2$. Also, let X be the set of vertices in the component of $V \setminus Q$ such that $|C_2 \cap X|$ is the smallest, and let $Y = V \setminus (Q \cup X)$. Let $X_2 = C_2 \cap X$, $Y_2 = C_2 \cap Y$, $|Q_2| = q$, and $|X_2| = x$.

Then $|Y_2| = k + 4 + i - q - x$. By construction, $|X_2| \leqslant |Y_2|$, so $x \leqslant \frac{k+4+i-q}{2}$. Finally, let $Q' \subseteq Q_{0,1}$ be the set of vertices that are adjacent to vertices in both X_2 and Y_2. By Lemma 2.1, at least one vertex in $Q_{0,1}$ must be adjacent to at most one vertex in $X_2 \cup Y_2$ (or the configuration will be solvable), so $|Q'| \leqslant k - q - 1$.

There are three cases to consider depending on the value of x.

1. ($x \geqslant 2$). Let $v_2 \in X_2$ and $u_2 \in Y_2$. Any vertex $u \in Q$ that is adjacent to both v_2 and u_2 must be in $Q_{0,1}$, since otherwise two more pebbles can be placed on u from v_2 and u_2, giving it four. For the same reason, each $u \in Q_{0,1}$ can be adjacent to at most three vertices in $X_2 \cup Y_2$. This implies that each $u \in Q_{0,1}$ can be adjacent to at most two distinct pairs in $X_2 \times Y_2$. Therefore $|X_2||Y_2| = x(k + 4 + i - q - x) \leqslant 2(k - q - 1)$ or, equivalently,

   $$(x - 2 + q - k - i)(x - 2) \geqslant 6 + 2i. \tag{2.1}$$

   Since $x \leqslant \frac{k+4+i-q}{2}$, and $q \leqslant k$,

   $$\begin{aligned}(x - 2 + q - k - i)(x - 2) &\leqslant \left(\frac{k + 4 + i - q}{2} - 2 + q - k - i\right)(x - 2)\\ &= \left(\frac{q - k - i}{2}\right)(x - 2)\\ &\leqslant 0,\end{aligned}$$

   contradicting (2.1).
2. ($x = 1$). Let $X = \{u\}$. Then there is a path of length two from each of the $k - q + 3 + i$ vertices in Y_2 to u with some vertex in Q' as the intermediate vertex. Thus, it is possible to move at least $k - q + 3 + i$ pebbles onto the vertices of Q'. Since $|Q'| \leqslant k - q - 1$, then $k - q + 3 + i \geqslant |Q'| + 4$, so either one of these vertices accumulates four or more, or at least two of them accumulates two or more. In either case, two pebbles can be moved onto u, giving it four pebbles.
3. ($x = 0$). Let $u \in X$. Then there is a path of length two from each of the $k + 4 + i - q$ vertices in Y_2 to u with some vertex in Q as the intermediate vertex. Let $S \subseteq Q$ be the set of vertices that are adjacent to both u and a vertex in Y_2, and let $S_2 = S \cap Q_2$. Then $|S| \leq k - q + |S_2|$, and it is possible to accumulate at least $|Y_2| + 2|S_2| = k + 4 + i - q + 2|S_2| \geqslant |S| + 4 + |S_2| + i \geqslant |S| + 4$ pebbles onto the vertices in S. There are four cases to consider, each of which leads to a solvable configuration.
   (a) Some vertex $v \in S$ can accumulate four pebbles.
   (b) There are four vertices in S which can accumulate two pebbles, in which case four pebbles can be moved to u.
   (c) Some vertex $v_1 \in S$ can accumulate three pebbles, and $v_2, v_3 \in S$ can accumulate at least two each. Then two pebbles can be moved from v_2 and v_3 onto u, and then one can be moved from u to v_1, so that v_1 accumulates four.
   (d) $v_1, v_2 \in S$ can each accumulate three pebbles, and every other vertex in S can accumulate only one pebble. In this case, $S = Q_{0,1}$, and every vertex in $S \setminus \{v_1, v_2\}$ can accumulate exactly one pebble. Then one pebble can be moved from each of v_1 and v_2 onto u, and then one pebble can be moved from u onto any vertex in S. Thus, any vertex in Q can

accumulate at least two pebbles and, by Lemma 2.1, the configuration is solvable. □

COROLLARY 2.4. *Let $G \in \mathcal{G}_{2,k}$. Then a configuration C is solvable if $|C_2| \geqslant \min\{3k-1, k+4\}$.*

3. 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 $\mathbf{b} = \{b_1, b_2, \ldots, b_l\}$, a **b**-*island* is an island that contains distinct vertices $v_1, v_2, \ldots, v_l$ such that $C(v_j) \geqslant b_j, 1 \leqslant j \leqslant l$. A *bridge* is a vertex which has zero pebbles. Notice that if a bridge v is adjacent to a 2-island, then one pebble can be added to v. If every vertex on an island contains precisely one pebble, we call it a *desert*. For an island I, the *surplus* $s_I(C)$ of I is the difference between the number of pebbles placed on I and the number of vertices in I, i.e.,

$$s_I(C) = \sum_{v \in I} C(v) - |I|.$$

We also define the surplus of the graph $s_G(C)$ as the sum of the surpluses of all islands. That is,

$$s_G(C) = \sum_{v \in G} C(v) - |C_1|.$$

Note that a pebbling move from an island to a bridge always reduces the surplus of the graph. In a graph with surplus s, for any vertex r, any r-solution can move pebbles onto at most s bridges, including r.

A vertex in G that is adjacent to at least one vertex of a subgraph H is called *adjacent to H*. Note that a vertex containing two or more pebbles allows the movement of at least one pebble to any other vertex on its island or any bridge adjacent to its island.

Every vertex of a graph is either a bridge or a member of a single island, and a pebbling configuration is solvable if and only if every bridge can be pebbled. In any sequence of pebbling steps, we say a bridge is *filled* if two pebbles are moved onto it, and *emptied* if two pebbles are removed from it. In a minimal r-solution for any root r, every bridge that is used must be filled and emptied, except r.

Recall that for any two vertices $u, v \in V$, the *distance* $d(u,v)$ between u and v is the number of edges on the shortest path connecting them. For a subset $S \subseteq V$, let $d_{\min}(v, S)$ be the smallest distance between vertex v and any vertex in S. Let

$$D_m(S) = \{v \in V \mid d_{\min}(v, S) = m\}.$$

In particular, for an island I, $D_1(I)$ is the set of vertices adjacent to at least one vertex in I.

LEMMA 3.1. *Let $G \in \mathcal{G}_2$ and let C be a configuration which contains an island I with $s_I(C) \geqslant 3$. Then C is solvable.*

Proof. If $s_I(C) \geqslant 3$, then I is a 4-island, a $\{2,3\}$-island, or a $\{2,2,2\}$-island. Clearly, a 4-island guarantees solvability. Given a $\{2,3\}$-island, we can move a pebble from the vertex with two pebbles to the vertex with three pebbles along a path in I, creating a solvable configuration. If I is a $\{2,2,2\}$-island, let a, b, and c be vertices in $C_2 \cap I$. Consider any path P between a and b. If $c \in P$, we can accumulate four pebbles on c from a and b. Otherwise, choose a shortest path P' between c and the vertices in P. Let $u = P \cap P'$. If $u = a$ (or $u = b$), we can accumulate four 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 move three pebbles from a, b, and c onto u, giving it four pebbles. ☐

In light of Lemma 3.1, we call an island I with $s_I(C) \geqslant 3$ an *empire*. If a graph does not contain an empire, then the only possible islands are deserts, 2-islands, 3-islands, and $\{2,2\}$-islands. Note that the latter three cases of islands are not mutually exclusive.

Islands and pebbles can be maintained as part of the graph data structure, and each vertex can store a reference to the island to which it belongs. The following basic procedures are needed in the construction of an algorithm to determine the solvability of graphs in $\mathcal{G}_{2,k}$. An upper bound for their running time is provided. 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. Thus UPDATEISLANDS is called at the end of any procedure that moves pebbles.

- UPDATEISLANDS() updates the data structure representing the islands and records whether or not an empire is present. This can be implemented using a standard BFS/DFS algorithm in $O(n+m)$ time (see [3], for instance).
- CONTAINSEMPIRE() returns TRUE if and only if G contains an empire. This takes $O(1)$ time.
- ISADJACENTTWOISLAND(u) returns TRUE if and only if u is adjacent to a 2-island. This can be implemented in $O(d)$ time, where d is the maximum degree of G, by checking if any of the vertices adjacent to u belong to a 2-island.
- PEBBLEFROMISLAND(I, u) performs a sequence of pebbling steps required to move a pebble from I to u using only vertices in I in intermediate steps, assuming that $u \in D_1(S)$ and I is a 2-island. It calls UPDATEISLANDS() and returns the resulting graph configuration. It takes $O(n+m)$ time.
- CANDOUBLEPEBBLE(G', I, u, v) returns TRUE if and only if pebbles can be moved from I (assuming that I is a $\{2,2\}$-island) to vertices u and v (where $u=v$ is possible) simultaneously, using only vertices from I in intermediate moves. It assigns to G' the resulting graph configuration and calls UPDATE ISLANDS() on G'. It takes $O(nm)$ time (see Corollary 3.3 below).

The next result by Shiloach [7] is used to determine the pebbling solvability in the presence of a $\{2,2\}$-island.

THEOREM 3.2 (see [7]). *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 admits two vertex-disjoint paths connecting s_1 to t_1 and s_2 to t_2.*

COROLLARY 3.3. *Let I be a $\{2,2\}$-island that is not an empire.*

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 $O(nm)$ time whether u can be filled from I.*

Proof. Let G' be the subgraph of G induced by the vertex set $I \cup u \cup v$, and s_1 and s_2 be elements of $C_2 \cap I$. Then the first condition follows from Theorem 3.2 by determining disjoint paths connecting either s_1 to u and s_2 to v, or s_1 to v and s_2 to u in G'.

For the second condition, notice that if u is adjacent only to one vertex in I, then it cannot be filled from I. Otherwise, vertex u can be filled from I if and only if there are disjoint paths to u from $\{s_1, s_2\} = C_2 \cap I$. Let G' be a graph induced by the vertex set $I \cup u$ with an added vertex u' that is adjacent to everything that u is. Then there are disjoint paths from s_1 to u and s_2 to u in G if and only if there are disjoint paths from s_1 to u and s_2 to u' in G'. The result follows from Theorem 3.2. ☐

4. Algorithms for $\mathcal{G}_{2,k}$. The goal of this section is to present a polynomial time algorithm to determine the solvability of a pebbling configuration for graphs in $\mathcal{G}_{2,k}$.

Let C be a configuration on $G \in \mathcal{G}_2$ and I an island in C. In any sequence of pebbling moves, if one or more pebbles are moved from the vertices of I, and no pebbles are moved onto I, then I is called an *origin*.

THEOREM 4.1. *Let C be a solvable configuration which does not contain an empire and contains some vertex r with $C(r) = 0$ that is not adjacent to any 2-island. Then, any minimal r-solution contains an origin.*

Proof. Let b be the number of bridges in C used in a minimal r-solution, counting repeated use of any bridge. Note that $b \geqslant 1$, since r is not adjacent to a 2-island. Further, in a minimal r-solution, exactly $2b$ pebbles are moved to the bridges, and exactly b pebbles are moved from the bridges to adjacent vertices. Let j be the number of these b pebbles that is moved onto bridges or deserts in C. Clearly, $j \geqslant 1$, since one pebble must be moved either to r or a desert adjacent to r. Thus, at most $b-j$ pebbles can be moved onto 2-islands, since 2-islands in C can be pebbled only from bridges in C. If there is no origin, the number of 2-islands used in the pebbling is at most $b - j$. Since there is no empire, the only possible islands in C are 2-islands (including 3-islands) and $\{2, 2\}$-islands, so at most two pebbles can be moved from each island. Therefore, at most $2(b - j)$ pebbles can be moved from 2-islands to bridges. Hence, at most $2(b - j) + j = 2b - j < 2b$ pebbles can be moved onto bridges, contradicting the fact that $2b$ pebbles were moved onto bridges. □

Theorem 4.1 allows us to construct a recursive algorithm for determining the solvability of graphs in $\mathcal{G}_2$. We will show that this algorithm is polynomial for graphs in $\mathcal{G}_{2,k}$ when k is constant.

Consider a configuration C that satisfies the condition of Theorem 4.1. Let r be a vertex of G and let I be an island which is an origin in a minimal r-solution. If I is a 2-island or 3-island, then the vertices of I are used to pebble to some adjacent bridge u, and then never used again. Therefore, instead of considering all possible pebbling sequences from I to u, we can just choose one of them.

If the origin I is a $\{2, 2\}$-island, then things are slightly different. If only one pebble is moved from I, the situation is the same as if I were a 2-island. If two pebbles are moved from I, either onto the same adjacent bridge u, or two adjacent bridges u and v, then the vertices of I are never used again, so, as before, we can choose just one such sequence of moves.

Algorithm 4.1 uses these ideas to determine whether every root r in the graph can be pebbled. It maintains a set L of vertices which can be pebbled, which is initialized with every vertex in C_1. It then calls Algorithm 4.2, which recursively tries pebbling moves from every 2-island to every adjacent bridge until every vertex can be pebbled or no moves are possible. At each recursive step, the algorithm checks whether r is adjacent to a 2-island or C' contains an empire, in which case the configuration is solvable. Otherwise, we need to consider configuration C' with fewer pebbles than C.

The following result proves that for every $\{2, 2\}$-island I, the recursive call will be made at least once for each vertex in $D_1(I)$ at step 4.2.4 in Algorithm 4.2.

LEMMA 4.2. *Let C be a configuration on a graph $G \in \mathcal{G}_{2,k}$ which contains some vertex r that is not adjacent to any 2-island. If C contains a $\{2, 2\}$-island I, then for any vertex $u \in D_1(I)$ there is a vertex $v \in D_1(I)$ such that algorithm* CANDOUBLEPEBBLE(G', I, u, v) *returns* TRUE.

Proof. Let u_1 and u_2 be vertices in I with two or more pebbles and $u \in D_1(I)$. Let

P be a shortest path from $\{u_1, u_2\}$ to u with vertex set $V(P) \subseteq I \cup u$. Without loss of generality, we can assume that $u_1 \in V(P)$. Since P is a shortest path, $u_2 \notin V(P)$. We can move a pebble from u_1 to u using vertices of P and, since $G \in \mathcal{G}_2$, there is a path u_2vr with $v \in D_1(I)$. Therefore, we can move a pebble from u_2 to v and algorithm CanDoublePebble(G', I, u, v) returns TRUE. ∎

Algorithm 4.1. IsSolvableDiamTwo(G).

```
global Set L
for u ← 0 to n − 1                                    ⎫
   do { if u ∈ C_1                                    ⎬ 4.1.1
          then add u to L                             ⎪
G.UpdateIslands()                                     ⎭

return (IsSolvableDiamTwoRec(G))
```

Algorithm 4.2. IsSolvableDiamTwoRec(G).

```
if G.ContainsEmpire()                                 } 4.2.1
  then return (TRUE)

for each 2-island I                                   ⎫
   do { for each u ∈ D_1(I)                           ⎪
          do { L.Add(u)                               ⎬ 4.2.2
if |L| = n                                            ⎪
  then return (TRUE)                                  ⎭

for each 2-island I                                   ⎫
   do { for each u ∈ D_1(I)                           ⎪
          do { G' = G.PebbleFromIsland(I, u)          ⎬ 4.2.3
               if IsSolvableDiamTwoRec(G')            ⎪
                 then return (TRUE)                   ⎭

for each {2,2}-island I                               ⎫
   do { for each u ∈ D_1(I) and v ∈ D_1(I)            ⎪
          do { if G.CanDoublePebble(G', I, u, v)      ⎬ 4.2.4
                 then { if IsSolvableDiamTwoRec(G')   ⎪
                          then return (TRUE)          ⎭

return (FALSE)
```

The next result provides an upper bound on the time required to determine the solvabilty of a configuration with $|C_2| = l$.

THEOREM 4.3. *Let G be a diameter two graph with a configuration C of pebbles such that $|C_2| = l$. Then the solvability of G can be determined in $O(l! \cdot n^{2l-1}m)$ time.*

Proof. We will use induction on $|C_2|$.

Base case. Let C be a configuration with $|C_2| = 1$. Then C is solvable if and only if either every bridge is adjacent to the 2-island or some vertex $v \in C_2$ contains at least four pebbles. Thus, the solvability of C can be determined in $O(n+m)$ time.

Induction step. Let us assume that the theorem is true for $|C_2| \leqslant l$, and let C be a configuration with $|C_2| = l + 1$.

If C contains an empire or every bridge is adjacent to a 2-island, then its solvability can be verified in $O(n+m)$ time. These conditions are checked in steps 4.2.1 and 4.2.2 of Algorithm 4.2.

Otherwise, C does not contain an empire and there is some bridge r that is not adjacent to any 2-island. If there is a minimal r-solution S_r in C, then by Theorem 4.1, there is an island I in G that is an origin for this solution. If I is a 2-island or 3-island, then a subsequence of steps in S_r required to move a pebble to some bridge $u \in D_1(I)$ results in a new configuration C' with $|C_2'| = |C_2| - 1 = l$. If I is a $\{2,2\}$-island that is used in S_r to pebble to bridges u and v (with $u = v$ a possibility), the result of the move(s) from I to u and v is a new configuration C' with $|C_2'| \leqslant |C_2| - 1 = l$. In either case, the resulting graph is solvable by induction.

Since it is unknown which island I is an origin, or which vertex u (or u and v) it pebbles to, we need to consider all of them. For every 2-island and 3-island, we will pebble to each bridge u adjacent to I. There are at most l such islands, and each is adjacent to at most $n-l$ bridges. For each island I and each $u \in D_1(I)$, we need to make the pebbling move and then make a recursive call. This requires $l(n-l)(O(n+m)+T(l-1))$ time, and corresponds to step 4.2.3.

Similarly, for every $\{2,2\}$-island I and pair of bridges u and v (including $u = v$) adjacent to I, we make the pebbling moves required to pebble to both u and v, assuming it is possible, and then make a recursive call. There are at most $l/2$ such islands, each adjacent to at most $(n-l)^2$ pairs of bridges. For each of these, we need to attempt to pebble to both u and v and make a recursive call. This requires $\frac{l}{2}(n-l)^2(O(nm)+T(l-1))$ time, and corresponds to step 4.2.4.

From this, we can see that the complexity of Algorithm 4.1 is

$$T(l) = O(n+m) + l(n-l)(O(n+m)+T(l-1)) + \frac{l}{2}(n-l)^2(O(nm)+T(l-1))$$
$$\leqslant O(l \cdot n^3 m) + l \cdot n^2 T(l-1),$$

where $T(1) = O(n+m)$. It follows that $T(l) = O(l! \cdot n^{2l-1} m)$. ∎

Corollary 4.4. *Let $G \in \mathcal{G}_{2,k}$, where k is a constant. Then the solvability of G can be determined in $O(c(k)! \cdot n^{2c(k)-3} m)$ time.*

Proof. Algorithm 4.3 extends Algorithm 4.1 by first determining the connectivity of the graph, and then applying Corollary 2.4. The vertex connectivity of a graph with n vertices and m edges can be determined in $O((n+\min\{k^{5/2}, kn^{3/4}\})m)$ time (see [4]). By Corollary 2.4, the configuration is solvable if $|C_2| \geqslant c(k) = \min\{3k-1, k+4\}$. If $|C_2| < c(k)$, the solvability of the pebbling configuration can be determined by the algorithm IsSolvableDiamTwo in $O(c(k)! \cdot n^{2(c(k)-1)-1} m)$ time. The total time is thus $O(c(k)! \cdot n^{2c(k)-3} m + (n+\min\{k^{5/2}, kn^{3/4}\})m) = O(c(k)! \cdot n^{2c(k)-3} m)$. ∎

Algorithm 4.3. IsSolvableDiamTwoComplete(G).

```
k = G.ComputeConnectivity()
if |C_2| ⩾ min{3k − 1, k + 4}
  then return (TRUE)
  else return (IsSolvableDiamTwo(G))
```

5. Class 1 graphs. The smallest number t, such that any configuration of t pebbles on G is solvable, is called the *pebbling number* of G. Clarke, Hochberg, and Hurlbert [2] provided a complete classification of graphs in $\mathcal{G}_2$. The pebbling number of any graph in $\mathcal{G}_2$ is either n (these graphs are called *Class* 0) or $n+1$ (called *Class* 1). Figure 5.1 introduces various subsets of $\mathcal{G}_2$ graphs according to class and connectivity. Algorithm 4.3 will determine the solvability of any diameter two graph of Class 0 or 1 in $O(c(k)! \cdot n^{2c(k)-3}m)$ time. For graphs in $\mathcal{D}$ and $\mathcal{F}$, this takes $O(nm)$ and $O(n^7m)$ time, respectively. In this section, we describe a more efficient technique for determining the solvability of a configuration for graphs of Class 1 using their structural properties. We will show that this approach requires $O(n+m)$ time for graphs in $\mathcal{D}$ and $O(nm)$ time for graphs in $\mathcal{F}$.

	Class	
k	0	1
1	$\emptyset$	$\mathcal{D}$
2	$\mathcal{E}$	$\mathcal{F}$
≥ 3	$\mathcal{H}$	$\emptyset$

FIG. 5.1. *Categorization of diameter two graphs.*

LEMMA 5.1. *Let G have a vertex of degree $n-1$. Then the solvability of G can be determined in $O(n+m)$ time.*

Proof. Let v be a vertex of degree $n-1$. If $C_2 \geqslant 2$, then it is possible to move two pebbles to v, making the configuration solvable. If $C_2 \leqslant 1$, then a configuration is solvable if and only if it contains a vertex with four or more pebbles, every bridge is adjacent to the single 2-island, or every vertex contains one pebble. All of these can be checked in $O(n+m)$ time. ∎

COROLLARY 5.2. *Let $G \in \mathcal{D}$. Then the solvability of G can be determined in $O(n+m)$ time.*

Proof. If $G \in \mathcal{D}$, then G contains a vertex of degree $n-1$. The result follows immediately from Lemma 5.1. ∎

Clarke, Hochberg, and Hurlbert [2] gave a description of the structure of graphs in $\mathcal{F}$, which was corrected by Blasiak and Schmitt [1]. Figure 5.2 shows the structure of all graphs in $\mathcal{F}$. At least two of the edges $p'q'$, $p'r'$, and $q'r'$ must be present. The possibly empty subgraph H_p (similarly for q and r) has all of its vertices adjacent to both q' and r', and each component of H_p has at least one vertex adjacent to p. Finally, each vertex of the subgraph H_c (which also may be empty) must be adjacent to at least two of p', q', and r'. Except for edges within the subgraphs H_p, H_q, H_r, and H_c, no other edges are permitted. Let $H_p' = H_p \cup p$ (similarly for q and r). Note that H_p' is connected if and only if each component of H_p has at least one vertex adjacent to p. Thus we can replace the conditions above with H_p' being nonempty, connected, and each of its vertices being adjacent to both q' and r'.

The existence of a $O(n^5)$ algorithm to determine whether a diameter two graph is Class 1 is implied in [2, 5], but no details are given.

LEMMA 5.3. *Membership in $\mathcal{F}$ can be determined in $O(n^3m)$ time.*

Proof. We first attempt to identify p', q', and r' by considering all triples of vertices in G. For each triple, we proceed with the following test, quitting if any step fails:

1. Verify that at least two of the edges $p'q'$, $p'r'$, and $q'r'$ are present.
2. Identify the subgraph H_p' by choosing it to be everything that is not in the connected component of $G \setminus \{q', r'\}$ which contains p'. Verify that H_p' is nonempty, connected, and each of its vertices is connected to both q' and r'. Repeat the same process for q and r.

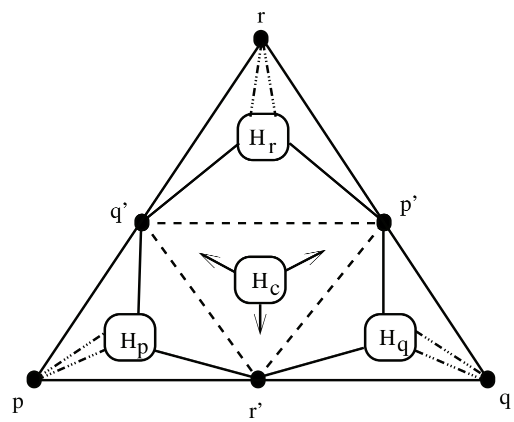

FIG. 5.2. *Family $\mathcal{F}$.*

3. Let H_c' be the set of all vertices not already accounted for. Verify that each one is connected to at least two of p', q', and r'.

Each step takes $O(n + m)$ time. If the test passes for a given choice of p', q', and r', then $G \in \mathcal{F}$. Otherwise, we try the next triple. If this test fails for all triples of vertices in G, $G \notin \mathcal{F}$. Since there are $\binom{n}{3} = O(n^3)$ ways of selecting p', q', and r', the total running time is $O(n^3(n + m)) = O(n^3 m)$. ∎

By [4], it can be determined in polynomial time whether $G \in \mathcal{G}_{2,1}$ or $G \in \mathcal{G}_{2,2}$ and, by Lemma 5.3, membership in $\mathcal{F}$ can be determined in polynomial time.

COROLLARY 5.4. *Membership in $\mathcal{D}$, $\mathcal{E}$, $\mathcal{F}$, and $\mathcal{H}$ can be determined in polynomial time.*

The next result describes properties of an unsolvable configuration in $\mathcal{F}$.

LEMMA 5.5. *Let $G \in \mathcal{F}$. If C is an unsolvable configuration on G, then all of the following are true:*

1. *At most one of p', q', and r' has two or more pebbles.*
2. *At most one vertex in each of H_r', H_p', H_q', and H_c has two or more pebbles.*
3. *At most two of H_r', H_p', H_q', and H_c have a vertex with two or more pebbles.*
4. *At most two vertices in G have two or more pebbles.*

Proof. Note that any pair of vertices from p', q', and r' forms a cut set, so if statement 1 or 2 is not met, then it is possible to move two pebbles to two of these three vertices. Therefore, by Lemma 2.1, the configuration is solvable. If all three of H_p, H_q, and H_r have a vertex containing two pebbles, it is possible to place two pebbles on any of p', q', and r'. If H_p, H_q, and H_c each have a vertex with two pebbles, then two pebbles can be placed on r', and on either q' or p' (or both). Applying symmetry and Lemma 2.1, the third condition is true. If the first three conditions are satisfied and G contains at least three vertices with two or more pebbles, then one of them must be p', q', or r', and the other two must be from H_r', H_p', H_q', and H_c' with at most one vertex from each. Without loss of generality, assume r' contains two pebbles. Due to symmetry there are four cases to consider:

1. If $u \in H_p$ and $v \in H_q$ each contain two pebbles, four pebbles can be moved to r'.
2. If $u \in H_p$ and $v \in H_r$ each contain two pebbles, two pebbles can be moved to q'.
3. If $u \in H_p$ and $v \in H_c$ each contain two pebbles, either four pebbles can be

moved to r' or two pebbles can be moved to q', depending on which edges among $p'q'$, $p'r'$, and $q'r'$ are present.

4. If $u \in H_r$ and $v \in H_c$ each contain two pebbles, two pebbles can be moved to q' or p', depending on which edges among $p'q'$, $p'r'$, and $q'r'$ are present.

In any of these cases the configuration is solvable. ∎

COROLLARY 5.6. *If $G \in \mathcal{F}$ has a configuration with $|C_2| \geqslant 3$, then C is solvable.*

We need to consider the cases $|C_2| = 1$ and $|C_2| = 2$. The result for $|C_2| = 1$ is obvious, so we state it without proof.

LEMMA 5.7. *Let $G \in \mathcal{G}_2$ and $C_2 = \{u\}$. Then C is solvable if and only if every bridge in C is adjacent to the 2-island or $u \in C_4$.*

We say that a bridge has *potential s* if s pebbles can be moved to it from adjacent islands.

LEMMA 5.8. *Let $G \in \mathcal{G}_2$ and $|C_2| = 2$. Then C is solvable if and only if every bridge is*

1. *adjacent to a 2-island,*
2. *adjacent to a bridge with potential two, or*
3. *adjacent to a desert which is adjacent to a bridge with potential two.*

Proof. The reverse implication is easy to check. For the forward implication, assume C is solvable, and that some bridge r is not adjacent to a 2-island. We need to show that r is adjacent to a bridge with potential two or a desert which is adjacent to a bridge with potential two.

If C contains an empire I, then it is possible to move four pebbles onto some vertex $v \in I$. Since $G \in \mathcal{G}_2$, there is some bridge u adjacent to r and v. Therefore, r is adjacent to a bridge of potential two, and condition 2 is satisfied.

If C does not contain an empire, $|C_2| = 2$ implies that the configuration consists of a $\{2, 2\}$-island, or two 2-islands, either or both of which may be 3-islands. Notice that moving a pebble from a 2-island to a bridge reduces the surplus of the graph by 1, and moving from a 3-island to a bridge reduces the surplus of the graph by 2. In any of these cases, moving pebbles to two different bridges reduces the surplus of the graph to 0, and r cannot be pebbled. Thus, in any r-solution, two pebbles must be moved from the vertices in C_2 onto the same bridge u, reducing the surplus to 1. This implies that r is the only other bridge that can be used in this solution, and u has potential two. Since the configuration is solvable, then r must be adjacent to u or some desert adjacent to u. ∎

The above discussion leads to the following.

THEOREM 5.9. *Let $G \in \mathcal{F}$. Then the solvability of G can be determined in $O(nm)$ time.*

Proof. The conditions of Corollary 5.6 and Lemmas 5.7 and 5.8 can be checked in $O(nm)$ time. ∎

REFERENCES

[1] A. BLASIAK AND J. SCHMITT, *Degree sum conditions in graph pebbling*, Australa. J. Combin., 42 (2008), pp. 83–90.

[2] T. A. CLARKE, R. A. HOCHBERG, AND G. H. HURLBERT, *Pebbling in diameter two graphs and products of paths*, J. Graph Theory, 25 (1997), pp. 119–128.

[3] T. H. CORMEN, C. E. LEISERSON, R. L. RIVEST, AND C. STEIN, *Introduction to Algorithms*, 2nd ed., MIT Press, Cambridge, MA, 2001.

[4] H. N. GABOW, *Using expander graphs to find vertex connectivity*, J. ACM, 53 (2006), pp. 800–844.

[5] G. HURLBERT, *On Graph Pebbling, Threshold Functions, and Supernormal Posets*, manuscript,

2000.

[6] K. Milans and B. Clark, *The complexity of graph pebbling*, SIAM J. Discrete Math., 20 (2006), pp. 769–798.

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

[8] N. G. Watson, *The Complexity of Pebbling and Cover Pebbling*, preprint, http://www.arxiv.org/abs/math.CO/0503511, 2005.