HAMeTONIAN-CONNECTED GRAPHS AND THEIR STRONG CLOSURES

PAK-KEN WONG
Department of Mathematics and Computer Science
Seton Hall University
South Orange, NJ 07079

(Received September 15, 1993 and in revised form November 2, 1993)

ABSTRACT. Let $G$ be a simple graph of order at least three. We show that $G$ is Hamiltonian-connected if and only if its strong closure $\text{sc}(G)$ is Hamiltonian-connected. We also give an efficient algorithm to compute the strong closure of $G$.

KEY WORDS AND PHRASES. Hamiltonian-connected graph, strong closure, degree sequence.

1991 AMS SUBJECT CLASSIFICATION CODE(S). 05C45, 05C99.

1. INTRODUCTION.

Let $G = (V,E)$ be a simple graph, $n \geq 3$ and $m \in E(G)$. $G$ is called Hamiltonian-connected if every two vertices of $G$ are connected by a Hamiltonian path. If $G$ is Hamiltonian-connected and $n \geq 4$, then $m \geq \frac{1}{2}(3n + 1)$ (see [2], p. 61).

In this paper, we define the strong closure $\text{sc}(G)$ of a simple graph $G$. We also show that $G$ is Hamiltonian-connected if and only if its strong closure $\text{sc}(G)$ is Hamiltonian-connected (Theorem 2.3). It follows immediately that if $\text{sc}(G)$ is a complete graph, then $G$ is Hamiltonian-connected (Corollary 2.4). As in the case of Hamiltonian graphs, there is no characterization of Hamiltonian-connected graphs. If we compute the strong closure of $G$ and find it is complete, then $G$ is Hamiltonian-connected. As another application, a result of O. Ore also follows from Corollary 2.4 (see Corollary 2.5).

In section 3, we give an efficient algorithm to compute the strong closure $\text{sc}(G)$ of any simple graph $G$. This algorithm can be executed in $O(n \log K)$ time, where $|K| = \frac{1}{2}(n^2 - n - 2m)$.

2. HAMILTONIAN-CONNECTED GRAPHS.

For each vertex $v$ of $G$, let $D(v) = \{u \in V(G) : u$ is adjacent to $v\}$. Then $d(v) = |D(v)|$ is the degree of $v$ in $G$.

We have the main result of this paper.

THEOREM 2.1. Suppose that $u$ is not adjacent to $v$ in $G$ and $d(u) + d(v) \geq n + 1$. Then $G$ is Hamiltonian-connected if and only if $G + (u,v)$ is Hamiltonian-connected.

PROOF. Suppose that $G + (u,v)$ is Hamiltonian-connected, but $G$ is not. Since $G$ is not Hamiltonian-connected, there exist two vertices $x$ and $y$ such that there is no Hamiltonian $x - y$ path in $G$. Since $G + (u,v)$ is Hamiltonian-connected, there is a Hamiltonian $u - v$ path in $G + (u,v)$ and hence in $G$. Therefore it follows that $(x,y) \neq (u,v)$. Let $P = \{w_1, w_2, \cdots, w_n\}$ be a Hamiltonian $x - y$ path in $G + (u,v)$, where $x = w_1$ and $y = w_n$. 
CASE 1. Assume that $x \neq u$ and $y \neq v$. Since $P$ is a Hamiltonian $x-y$ path in $G + (u, v)$ and $P$ is not a Hamiltonian $x-y$ path in $G$, $(u, v)$ must be an edge of $P$ in $G + (u, v)$. Therefore $u = w_k$ and $v = w_{k+1}$ for some $1 < k < n-1$. $(k \neq n-1$; for otherwise $k + 1 = n$ and $v = w_n = y)$. Since $(u, v)$ is not an edge of $G$, $u, v \notin D(u)$ and $u, v \notin D(v)$. Suppose $w_t \in D(u)$, where $t \neq k-1$ and $t \neq n$. Since $(u, v)$ is not an edge of $G$, $u, v \notin D(u)$ and $u, v \notin D(v)$. Suppose that this is not true, then $w_{t+1} \notin D(v)$. Suppose that this is not true, then $w_{t+1} \notin D(v)$. Suppose that this is not true, then $w_{t+1} \notin D(v)$. Therefore $d(u) + d(v) < n$, which is a contradiction. Therefore Case 1 is impossible.

CASE 2. Assume that $v = y$ ($= w_n$). Since $(x, y) \neq (u, v)$, it follows that $u \neq x$ and so $u = w_{n-1}$. Let $w_t \in D(u)$, where $t \neq n-2$. Then by the same argument as in Case 1, $w_{t+1} \notin D(v)$. Hence $d(v) \leq (n-2) - (d(u) - 1) = n - d(u) - 1$
and so $d(u) + d(v) \leq n - 1$, which is impossible. Therefore $G$ is Hamiltonian-connected. The converse of the theorem is clearly true. This completes the proof of the theorem.

Theorem 2.1 motivates the following definition.

**The stron closure of** $G$ **is the graph obtained from** $G$ **by recursively joining pairs of nonadjacent vertices whose degree sum is at least $n + 1$ until no such pair remains. We denote the strong closure of** $G$ **by** $sc(G)$.

**REMARK.** The closure $c(G)$ of $G$ is defined and studied in [2] and [4]. It is useful in the study of Hamiltonian graphs. The definition of $sc(G)$ is similar to that of $c(G)$.

**LEMMA 2.2.** $sc(G)$ is well-defined.

**PROOF.** This follows from the proof of ([2], p. 56, Lemma 4.4.2).

**THEOREM 2.3.** A graph is Hamiltonian-connected if and only if its strong closure is Hamiltonian-connected.

**PROOF.** This follows immediately from Theorem 2.1 and Lemma 2.2.

Theorem 2.3 gives some interesting results.

**COROLLARY 2.4.** If $sc(G)$ is a complete graph, then $G$ is Hamiltonian-connected.

**PROOF.** If $sc(G)$ is complete, then it is Hamiltonian-connected and so by Theorem 2.3, $G$ is also Hamiltonian-connected.

The following result was obtained by O. Ore (see [1], p. 136, Theorem 11.3 or [5]).

**COROLLARY 2.5.** If $d(u) + d(v) \geq n + 1$ for every pair of nonadjacent vertices $u$ and $v$, then $G$ is Hamiltonian-connected.

**PROOF.** Since $d(u) + d(v) \geq n + 1$ for every pair of nonadjacent vertices $u$ and $v$, it follows that $Sc(G)$ is a complete graph. Therefore by Corollary 2.4, $G$ is Hamiltonian-connected.

If $G$ has vertices $v_1, v_2, \cdots, v_n$, the sequence $(d(v_1), d(v_2), \cdots, d(v_n))$ is called a degree sequence of $G$. The following result is similar to a result obtained by Chvátal (see [2], p 57, Theorem 4.5).
COROLLARY 2.6. Let \((d_1, d_2, \ldots, d_n)\) be a degree sequence of \(G\) such that \(d_1 \leq d_2 \leq \cdots \leq d_n\). Suppose that there is no value of \(p\) less than \(\frac{1}{2} (n + 1)\) for which \(d_p \leq p\) and \(d_{n-p} < n - (p - 1)\). Then \(G\) is Hamiltonian-connected.

PROOF. By a similar argument as in the proof of ([2], p. 57, Theorem 4.5), we can show that \(sc(G)\) is a complete graph. Therefore by Corollary 2.4, \(G\) is Hamiltonian-connected.

3. AN ALGORITHM FOR FINDING STRONG CLOSURE.

In this section, we give an algorithm to find \(sc(G)\). Let \(V(G) = \{u_1, u_2, \ldots, u_n\}\).

STEP 1. For \(1 \leq i < j \leq n\), let

\[
 f(i, j) = \begin{cases} 
 d(u_i) + d(v_j), & \text{if } u_i \notin D(v_j) \\
 0, & \text{if } u_i \in D(v_j)
\end{cases}
\]

STEP 2. Choose \(f(I, J) = \max \{f(i, j): 1 \leq i < j \leq n\}\).

If \(f(I, J) < n + 1\), then go to Step 4.

STEP 3. Form \(f(I, J)\) as follows:

- If \(f(p, I) \neq 0\), then \(f(p, I) \leftarrow f(p, I) + 1 (1 \leq p < I)\).
- If \(f(I, p) \neq 0\), then \(f(I, p) \leftarrow f(I, p) + 1 (I < p \leq n)\).
- If \(f(q, J) \neq 0\), then \(f(q, J) \leftarrow f(q, J) + 1 (1 \leq q < J)\).
- If \(f(J, q) \neq 0\), then \(f(J, q) \leftarrow f(J, q) + 1 (J < q \leq n)\).

Go to Step 2.

STEP 4. Form \(sc(G)\) by joining \(u_i\) to \(u_j\) if \(f(i, j) = 0 (1 \leq i < j \leq n)\).

Let \(G\) be represented by an adjacency matrix. Steps 1 and 4 can be implemented in \(O(n^2)\) time. Clearly, Step 3 runs in \(O(n)\) time. Let \(K = \{(i, j): f(i, j) \neq 0 \ 1 \leq i < j \leq n\}\). Then

\[ |K| = 1 + 2 + \cdots + (n - 1) - m = \frac{1}{2} (n^2 - n - 2m). \]

By using \(F\)-heaps data structure [3], find \(\max\{f(i, j)\}\) takes \(O(log_2 |K|) = O(log_2 n)\) time. Hence Steps 2 and 3 take \(O(|K|(n + log_2 n)) = O(n |K|)\). Thus overall we have an \(O(n |K|)\) algorithm.

LEMMA 3.1. If \(n \geq 4\) and \(d(u) \leq 2\) in \(G\), then \(d(u) \leq 2\) in \(sc(G)\).

PROOF. Let \(v\) be a vertex of \(G\) which is not adjacent to \(u\). Then \(d(v) \leq n - 2\). Hence \(d(u) + d(v) \leq 2 + (n - 2) = n\) and so Lemma 3.1 is true.

Lemma 3.1 allows us not to consider \(u\) in the computation of \(sc(G)\) if \(d(u) \leq 2\) in \(G\).

REFERENCES