No, he has not proved that $P = NP$. He proved that $L \in P$, and that $L$ reduces to an $NP$-complete problem $M$. To prove that $P = NP$, he would need to prove that $L$ is $NP$-complete.

**R-13.3**

$SAT \in NP$:

There is an obvious polynomial-time verification algorithm. It takes as input an assignment of 0/1 to variables and verifies that using that assignment results in satisfying the formula.

$SAT$ is $NP$-hard:

We show a reduction from $3SAT$ (which is known to be $NP$-complete) to $SAT$. Let $\varphi$ be an instance of $3SAT$. It is a 3-CNF Boolean formula. Since instances of $SAT$ are arbitrary Boolean formulas $\varphi$ is also an instance of $SAT$. Our reduction therefore simply outputs $\varphi$. The equivalence:

$\varphi \in 3SAT \iff \varphi \in SAT$

is obvious.

**C-13.7**

$SUBGRAPH-ISOMORPHISM \in NP$:

The verification algorithm takes the description of the isomorphism and verifies that it is in fact an isomorphism. It runs in polynomial time.

$SUBGRAPH-ISOMORPHISM$ is $NP$-hard:

We show a reduction from $CLIQUE$ (which is known to be $NP$-complete) to $SUBGRAPH-ISOMORPHISM$. Let $(G, k)$ be an instance of $CLIQUE$. We construct an instance $(G', H')$ of $SUBGRAPH-ISOMORPHISM$ in the following way: we set $G' = G$ and $H' = C_k$ (clique of $k$ vertices).

If $G$ contains a clique of size at least $k$, then it also contains a clique of size exactly $k$ (if a clique has $k + l$ vertices, removing any $l$ vertices produces a clique of $k$ vertices). Let us call it $C$. So, $C$ is isomorphic to $H'$.

Conversely, if $G' = G$ contains a subgraph $C$ isomorphic to $H'$, then $C$ is a clique of size $k$.

Therefore,

$(G, K) \in CLIQUE \iff (G', H') \in SUBGRAPH-ISOMORPHISM$. 


C-13.8

\textbf{INDEPENDENT-SET} \in \textbf{NP}:

The verification algorithm takes the description of an independent set, checks that it has at least \(k\) vertices, and that it is in fact an independent set. It runs in polynomial time.

\textbf{INDEPENDENT-SET} is \textbf{NP}-hard:

We show a reduction from \textbf{CLIQUE} (which is known to be \textbf{NP}-complete) to \textbf{INDEPENDENT-SET}. Let \((G, k)\) be an instance of \textbf{CLIQUE}. We construct an instance \((G', k')\) of \textbf{INDEPENDENT-SET} in the following way: we set \(G' = G^c\) (complement of \(G\)) and \(k' = k\).

If there is a clique \(C\) of size at least \(k\) in \(G\), then none of the edges in \(C\) is in \(G'\), so the vertices of \(C\) form an independent set (of size at least \(k\)) in \(G'\).

Conversely, if \(C'\) is an independent set in \(G'\) of size at least \(k\), then in \(G\) there are edges between all pairs of vertices of \(C'\), so they form a clique (of size at least \(k\)) in \(G\).

Therefore,

\((G, K) \in \text{CLIQUE} \iff (G', k') \in \text{INDEPENDENT-SET} \). 

C-13.12

Use the algorithm for problem \textbf{C-5.10} from Homework #5, but this time look for a subset of sum \(k\), instead of \(N/2\) (\(N\) is the sum of all elements of the set). Its running time is \(O(n'k) = O(nN)\) (as \(k \leq N\)), where \(n'\) is the number of items in the set.

Let \(n\) denote the size of the input to \textbf{SUBSET-SET} that is written in unary. Obviously, \(n' \leq n\). Additionally, \(N \leq n\) \((N = \sum_{i=0}^{n'} x_i\), and unary representation of \(x_i\) takes \(x_i\) bits).

Therefore, the algorithm runs in time \(O(n'N) = O(n \cdot n) = O(n^2)\), so it is polynomial, and the \textbf{SUBSET-SUM} problem is not strongly \textbf{NP}-hard.