In the proof of Theorem 5.15, the construction that starts from \( \langle M, w \rangle \) and results in an encoding \( \langle P \rangle \) of an “equivalent” instance of the Post Correspondence Problem is quite long, but every step in the construction is completely algorithmic and could be performed by a Turing machine as a preprocessor. The goal of the construction is to ensure that \( P \) has a match if, and only if, \( M \) accepts \( w \). Remember that the match, if it exists, is an accepting computation history for \( M \) on \( w \). Every aspect of the construction of \( P \) is designed to ensure that this Boolean equality holds for any Turing machine \( M \) and any input \( w \).

Just to review the reduction strategy: Assume (as the hypothesis in a proof by contradiction) that there is a Turing machine \( C \) that decides \( PCP \). \( C \) takes as input any instance \( \langle P \rangle \) of the Post Correspondence Problem. \( C \) accepts \( \langle P \rangle \) if \( P \) has a match and rejects \( \langle P \rangle \) if it does not.

Using \( C \) as a subroutine, we could then construct a decider for \( A_{TM} \). Given the input \( \langle M, w \rangle \), this decider would first construct \( \langle P \rangle \), as described in the proof, and then run \( C \) on \( \langle P \rangle \), accepting if \( C \) accepts and rejecting if \( C \) rejects.

But Theorem 4.1 tells us that there is no decider for \( A_{TM} \), so there cannot be a decider for \( PCP \) either.