For \( n \geq 2 \), \( n \) prime \( \Rightarrow n \) odd.

So \( g(n) \leq f(n) \).

So \( g(n) = O(f(n)) \).

Converse not true:

There are arbitrarily large \( n \) for which \( f(n) = \Omega(g(n)) \).

So for any constants \( c, k \), there exist an \( n \) with

\[ f(n) \geq c \cdot g(n). \]
Let $A_1$ compute the length of the shortest TSP tour.

An algorithm for finding the tour on graph $G = (V,E)$:

1. Run $A_1$ on original graph. Get length $L$.

2. For each edge $e$ in $E$:
   
   Change length of edge to 1 if its original value.
   Call $A_1$.
   If length of tour $> L$, $e$ must be on the optimal tour.
   Return length of $e$ to its original value.
   Else delete $e$ from the graph.
Endif

Note: Need to delete $e$ in case there are multiple optimal tours. Eg, if $G$ is $\Box$ and every edge has length 1, $A_1$ will return that every call of edges are not deleted.

This algorithm only calls $A_1$ $|E|$ times, so it is a polynomial algorithm.
3. Matching with bonds

In NP: Give list of edges. Do the edges constitute a matching? Do they satisfy the bond conditions?

NP-complete:
Reduce 3-SAT to matching with bonds.
Clause $x_i + x_j + x_k$ goes to

This component contains a matching of size 4. Only matchings of size 4 contain the three rightmost edges.

Any combination of the three rightmost edges can be extended to a matching of size 4. Hence clause being true $\iff$ matching of size 4 is component.

For each variable introduce component

$\forall x_i$ Ensures consistency. Then get 2n bonds, one each for $x_i$ and $\neg x_i$.

3-SAT instance true $\iff$ 3 matching of size $\geq n + 4n$ (n clauses, n variables)
A clause is made up of three literals. If \( C_j = x_1 + x_2 + \overline{x}_3 \), \( C_j \) is true if and only if \( y_1 + y_2 + (\neg y_3) \geq 1 \), \( y_j \) being binary.

Thus, we get \( m \) inequalities from the \( m \) clauses. These inequalities can all be satisfied simultaneously if and only if we can satisfy, for example,

\[ y_1 + y_2 + (\neg y_3) - \omega_1^j - \omega_2^j = 1, \]

where all variables are binary.

Notice that the l.h.s. of this inequality can only equal -1, 0, 1, 2, or 3. Thus, by choosing appropriate weights, we can sum the inequalities:

\[ \sum 10^j \text{ (equality } j) \]

This can be satisfied by binary variables if and only if all the original equalities can.

E.g.: If the three equalities are:

\[ y_1 + y_2 + (\neg y_3) - \omega_1^1 - \omega_2^1 = 1 \]
\[ y_1 + (\neg y_2) + (\neg y_3) - \omega_1^2 - \omega_2^2 = 1 \]
\[ (\neg y_1) + (\neg y_2) + y_3 - \omega_1^3 - \omega_2^3 = 1, \]

we get the equality:

\[ y_1 + y_2 + (\neg y_3) - \omega_1^1 - \omega_2^1 + 10(y_1 + (\neg y_2) + (\neg y_3) - \omega_1^2 - \omega_2^2) + 100((\neg y_1) + (\neg y_2) + y_3 - \omega_1^3 - \omega_2^3) = 111 \]

Then \( a^T x = b \) is equivalent to \( a^T x \leq b \) and \( a^T x \geq b \).