Three Properties of Heuristics Computed by Problem Relaxation

1. Heuristics computed by problem relaxation are admissible: \( h(n) \leq h^*(n) \)

Proof sketch:
see figure

\( h(n) \) goes through shortcuts in relaxed problem

\( h^*(n) \): shortest path in the base graph
2. Heuristics computed by problem relaxation are monotone:
Theorem. If \( h(m) \) and \( h(n) \) are computed as costs of shortest path in a relaxed subproblem, then

\[
h(m) - h(n) \leq d(m, n).
\]

Proof sketch:

See figure (showing relaxed graph) and note that \( d(m, n) \geq d'(m, n) \) on base on relaxed problem, so

\[
h(m) \leq d'(m, n) + h(n) \leq d(m, k) + h(n)
\]
3. Consider the algorithm that solves a state space search problem in two phases:
   (a) compute $h(\cdot)$ by problem relaxation using Dijkstra's algorithm (on a relaxed subproblem).
   (b) use $h(\cdot)$ as $A*$ does on the base problem.

Such algorithm (call it $M$) expands at least every node expanded by Dijkstra's algorithm on the same problem.

**Proof.**

In order to save on expanded nodes, there must be
at least one node (say, \( m \)) whose heuristic \( h(n) \) is large enough to prevent some nodes (say, \( g_i \), \( i = 1, \ldots, k, \ k \geq 1 \)) from being expanded in phase (b).

Example:

\[
\begin{array}{ccc}
S & 1 & 0 \\
\downarrow & & \\
1 & 0 & 1 \\
\downarrow & & \\
2 & 0 & 2 \\
\downarrow & & \\
3 & 0 & 3 \\
\downarrow & & \\
m & 2 & 4 \\
\downarrow & & \\
g_i & 3 & * \\
\downarrow & & \\
d & 4 & 0 \\
\downarrow & & \\
k & 5 & 5 \\
\end{array}
\]

*: not computed, 
\( b \) i.e. \( m \) is not expanded in phase (b).
The largest set of nodes of the $g_i$ kind is characterized by the following inequality:

$$g(m) + d(m, q_i) < h^*(s) \quad (1)$$

This inequality characterizes the largest set of nodes that would be expanded by Dijkstra's algorithm (because their $g$ value is less than $h^*(s)$), but are not expanded in phase (b) of algorithm M, because $m$ itself is not expanded. Inequality (1) can be rewritten as
\[ d(m, g_i) < h^*(s) - g(m) \] (2)

But in order for \( m \) not to be expanded in phase (b), it must be that

\[ g(m) + h(m) \geq h^*(s) \] (3). This can be rewritten as

\[ h(m) \geq h^*(s) - g(m) \] (4).

But in order to compute \( h(m) \), one needs to expand, i.e., the relaxed subproblem, using Dijkstra's algorithm, at least all nodes closer to \( m \) than \( h(m) \), i.e., all nodes \( h_i \leq h(s) \).
\[ d(m, h_i) < h^*(s) - g(m), \text{ i.e., at least the nodes } h_i \text{ s.t.} \]
\[ d(m, h_i) < h^*(s) - g(m) \quad (5). \]

Comparing (5) with (2), one sees that at most the nodes for which \( d(m, h_i) < h^*(s) - g(m) \) are not expanded in phase (b).

Therefore, the set of nodes that are expanded in phase (a) to compute \( h^* \) is a (non-necessarily strict) superset of the nodes that are not expanded
in phase (b) by using the heuristic. This is what the theorem claims.