Search-Based Planning– Solution

1) Consider the formulation of the Tower of Hanoi puzzle as a planning problem (solution of Problem **??**). Using the delete relaxation, calculate the length of the action sequence for achieving each predicate that appears in the goal state from the start state, then calculate the heuristic value of the start state as the maximum of these values.

Answer:

There is only one operator, namely, `Move(X, Y, Z)`. With the delete relaxation, we obtain:

```
Action Move(X,Y,Z):
Preconditions = {Clear(X), On(X,Y), Clear(Z), Smaller(X,Z)};
Effects = {Clear(Y), On(X,Z)};
```

We use the following recursion to compute $h_s(p)$ for all propositions p that appear in the goal state:

$h_s(\text{set of predicates } s') = max_{predicate\ p\ in\ s'} h_s(p)$.
$h_s(\text{predicate } p) = 0$ if p appears in the state s,
$h_s(\text{predicate } p) = 1 + min_{relaxed\ operator\ o\ with\ p\ in\ add\ list} h_s(\text{precondition list of } o)$, otherwise.

That is, if a proposition p appears in the start state s, $h_s(p) = 0$. The following propositions appear in the start state (for brevity, we skip the propositions of the form `Smaller(X,Y)`):

```
Clear(D1), On(D1, D2), On(D2, D3), On(D3, P1),
Clear(P2), Clear(P3).
```

Let P_i denote the set of propositions p with $h_s(p) = i$. That is, P_0 contains exactly the propositions in the start state. Note that P_0 contains `On(D1, D2)` and `On(D2, D3)`, but not `On(D3, D3)`. We now incrementally compute P_1, P_2, ..., until we discover the P_i that contains `On(D3, D3)`.

We start with P_1. This is the set of propositions that can be added by an operator whose preconditions are satisfied by P_0. There are only two operators whose preconditions are satisfied by the propositions in P_0, namely, `Move(D1, D2, P2)` and `Move(D1, D2, P3)`. The add effects of these two operators have the propositions `On(D1, P2), On(D1, P3), Clear(D2)`. Since none of them are in P_0 and all preconditions of the operators that add them are in P_0, all of them must be in P_1 (because of the recursive rules listed above). Therefore, $P_1 = \{\texttt{On(D1,P2)}, \texttt{On(D1,P3)}, \texttt{Clear(D2)}\}$. Below, we list $P_0 \cup P_1$.

```
Clear(D1), On(D1, D2), On(D1, P2), On(D1, P3),
Clear(D2), On(D2, D3), On(D3, P1),
Clear(P2), Clear(P3).
```

We now identify P_2. This is the set of propositions that can be added by an operator whose preconditions are satisfied by $P_0 \cup P_1$. There are two new operators whose preconditions are satisfied by $P_0 \cup P_1$, namely `Move(D2, D3, P2)` and `Move(D2, D3, P3)`, which add the propositions `On(D2, P2), On(D2, P3), Clear(D3)` (note that, at this point, any operator that moves D1 does not add any new propositions, so we skip those operators). These propositions do not appear in $P_0 \cup P_1$ but can be added by operators whose preconditions are in $P_0 \cup P_1$. Therefore, $P_2 = \{\texttt{On}(\texttt{D2}, \texttt{P2}), \texttt{On}(\texttt{D2}, \texttt{P3}), \texttt{Clear}(\texttt{D3})\}$. Below, we list $P_0 \cup P_1 \cup P_2$.

```
Clear(D1), On(D1, D2), On(D1, P2), On(D1, P3),
Clear(D2), On(D2, D3), On(D2, P2), On(D2, P3),
Clear(D3), On(D3, P1),
Clear(P2), Clear(P3).
```

$P_0 \cup P_1 \cup P_2$ satisfy the preconditions of `Move(D3, P1, P3)`, which adds `On(D3, P3)`. Therefore, `On(D3, P3)` $\in P_3$ and, consequently $h_s(\texttt{On}(\texttt{D3}, \texttt{P3}) = 3$.

Taking the maximum of 0, 0 and 3, we obtain 3 as the heuristic value of the start state.