Chapter3

Heuristic Search Techniques

(Cont….)

Steepest-Ascent Hill Climbing (Gradient search)

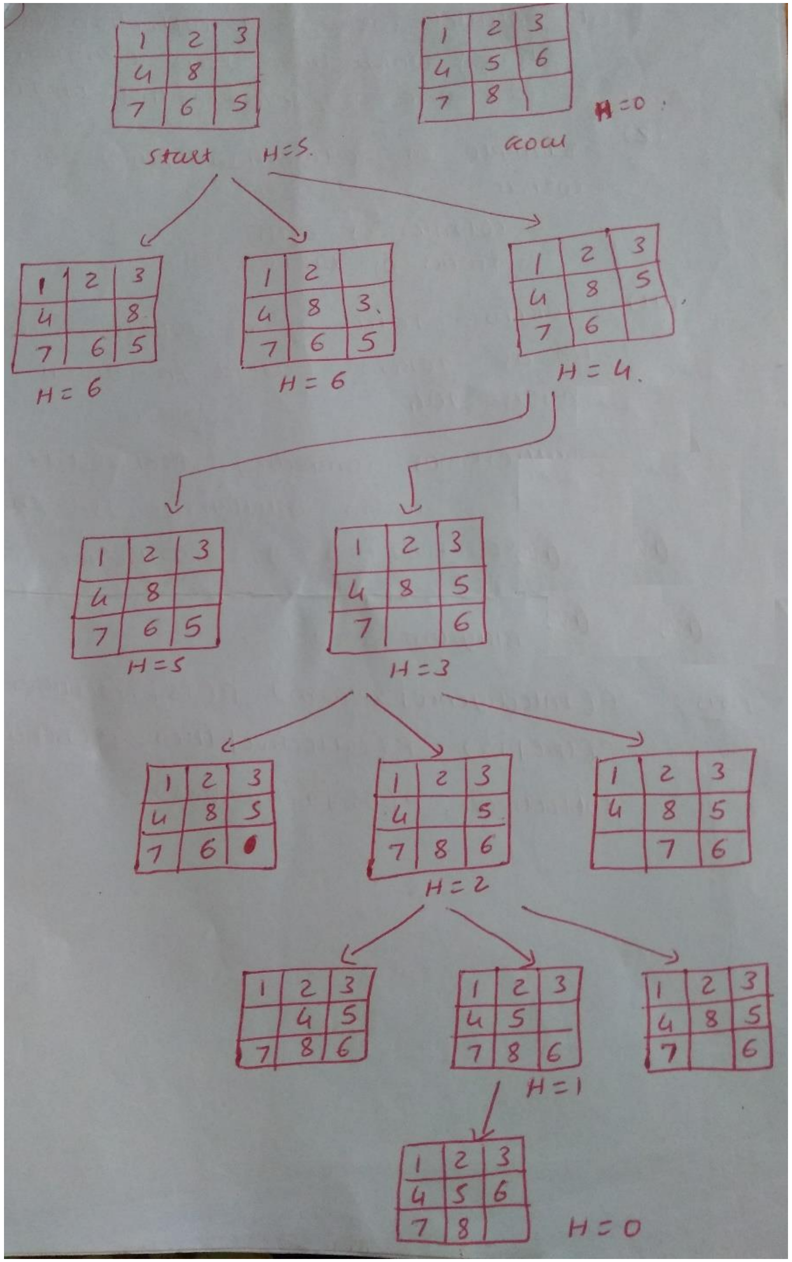
- Variation on simple hill climbing
- Considers all the moves from the current state and selects the best one as the next state
- Contrasts with the basic method in which the first state that is better than the current state is selected

Steepest-Ascent Hill Climbing

Algorithm: Steepest-Ascent Hill Climbing

1. Evaluate the initial state. If it is also a goal state, then return it and quit. Otherwise, continue with the initial state as the current state.
2. Loop until a solution is found or until a complete iteration produces no change to current state:
   (a) Let *SUCC* be a state such that any possible successor of the current state will be better than *SUCC*.
   (b) For each operator that applies to the current state do:
      i. Apply the operator and generate a new state.
      ii. Evaluate the new state. If it is a goal state, then return it and quit. If not, compare it to *SUCC*. If it is better, then set *SUCC* to this state. If it is not better, leave *SUCC* alone.
   (c) If the *SUCC* is better than current state, then set current state to *SUCC*.

Steepest-Ascent Hill Climbing



Steepest-Ascent Hill Climbing

Heuristic = sum of the (Manhattan) distance of every numbered tile to its goal position

= 0 + 0 + 0 + 0 + 1 + 0 + 2 + 2

Start = 5

Steepest-Ascent Hill Climbing

- **Simple Hill Climbing: number of moves required to get to a solution is longer**

  ↕ **trade-off**

- **Steepest-Ascent Hill Climbing: Time required to select a move is longer**

- **Both basic and steepest-ascent hill climbing may fail to find a solution, either algorithm may terminate not by finding a goal state but by getting to a state from which no better states can be generated**

- **This will happen if the program has reached either a <u>local maximum, a plateau or a ridge</u>**

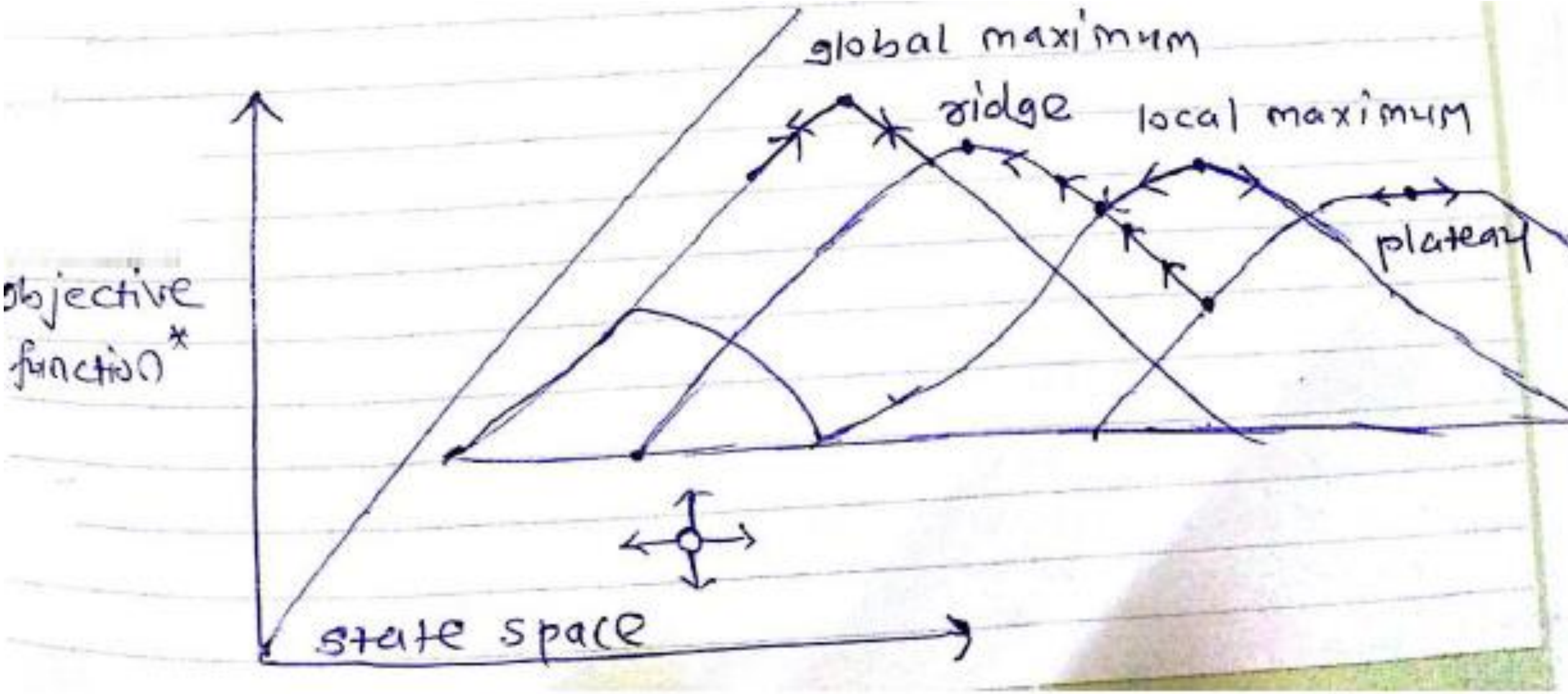
Disadvantage of Hill Climbing

- **Local maximum:**
  - **Is a state that is better than all its neighbours but is not better than some other states farther away**
- **Plateau:**
  - **Is a flat area of search space in which whole set of neighboring states have the same value**
  - **Not possible to determine the best direction in which to move by making local comparisons**

Disadvantage of Hill Climbing

- **Ridge:**
  - **Special kind of local maximum**
  - **It is an area of the search space that is higher than surrounding areas and that itself has a slope (Which one would like to climb). But the orientation of the high region, compared to the set of available moves and the directions in which they move, makes it impossible to traverse a ridge by single moves**

Disadvantage of Hill Climbing



Ways to solve Hill Climbing problems

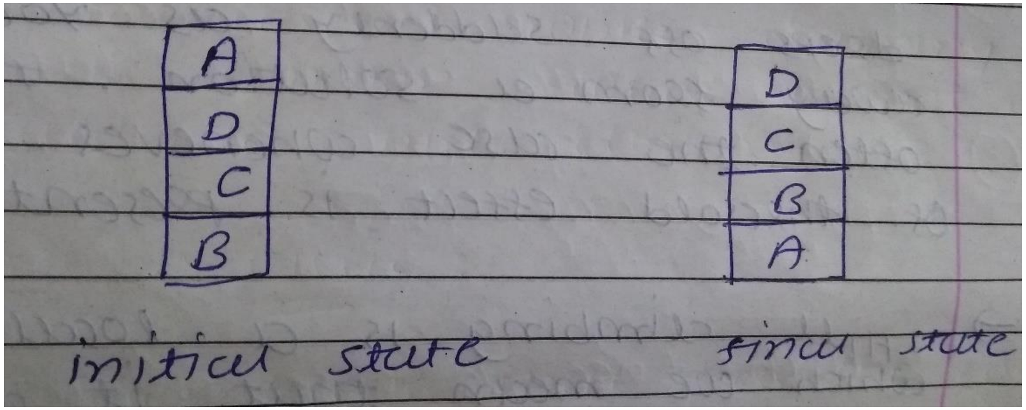
- **Local maximum:**
  - **Backtrack to some earlier node and try going in a different direction**
- **Plateau:**
  - **Make a big jump in some direction to try to get to a new section of the search space**
  - **If the only rules available describe single small steps, apply them several times in the same direction**
- **Ridge:**
  - **Apply two or more rules before doing the test**
  - **This corresponds to moving in several directions at once**

- **Even with these first-aid measures, hill climbing is not always very effective**
- **It is particularly unsuited to problems where the value of heuristic function drops off suddenly as you move away from a solution. These is often the case whenever any sort of threshold effect is present**
- **Hill climbing is a local method by which we mean that it decides what to do next by looking only at the “immediate” consequences of its choice rather than by exhaustively exploring all the consequences**
- **Advantage of being local heuristic: less combinatorially explosive**
- **Disadvantage of being local heuristic: lack of a guarantee that it will be effective**

- **How to make hill climbing use global information?**

  - provide global information in heuristic function

<u>Example:</u> Blocks world problem



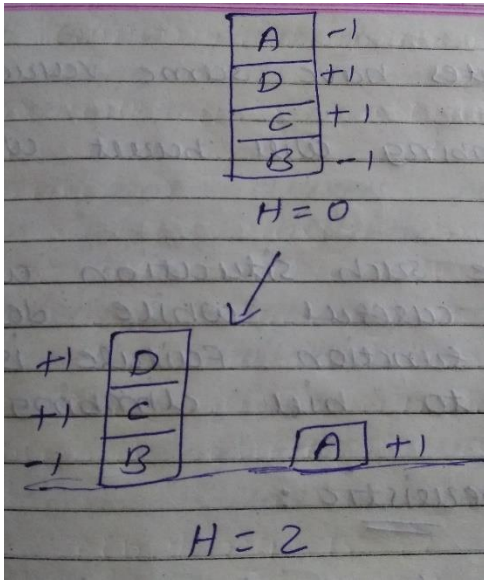
- **Local heuristic:**
  - **+1 for each block that is resting on the thing it is suppose to be resting on**
  - **-1 for each block that is resting on wrong thing**

Blocks world problem

Start state : (-1) + (+1) + (+1) + (-1) = 0

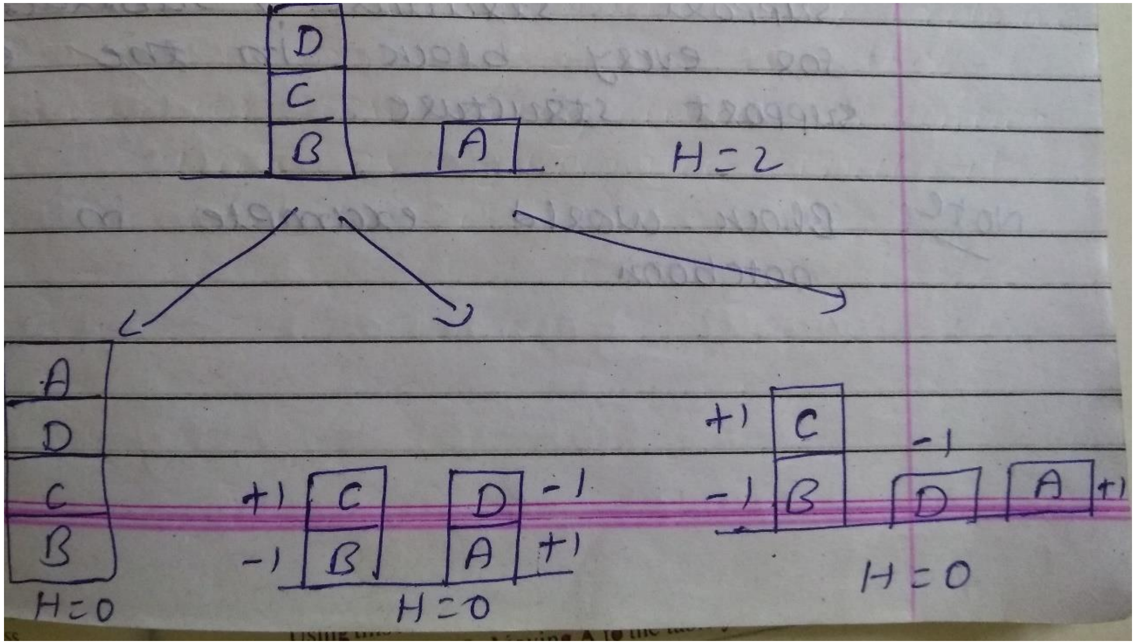
Goal state: (+1) + (+1) +(+1) + (+1) = 4

- **Apply operator to current state**



- **If new state is better than current state than new state becomes current state**

Blocks world problem



- All the states have same value: Plateau

Blocks world problem

- **Global heuristic:**
  - **For each block that has the correct support structure(i.e. the complete structure underneath it is exactly as it should be) , add one point for every block in the support structure**
  - **For each block that has an incorrect support structure, subtract one point for every block in the existing support structure**

Blocks world problem

Initial	
A	−3
D	−2
C	−1
B	0
	−6

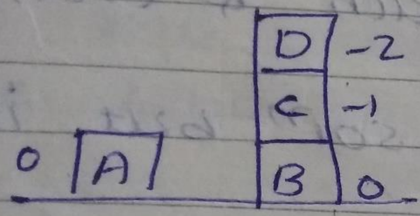
Goal	
D	3
C	2
B	1
A	0
	6

FOR A: −1 −1 −1 = −3
" D: −1 −1 = −2
" C, −1 = −1
" B, = 0
−6

FOR D, +1 +1 +1 = 3
" C, +1 +1 = 2
" B +1 = +1
" A, = 0
6

Blocks world problem

Next possible state from Initial state



H = -3

For A (No support structure) = 0

For D, -1 -1 = -2

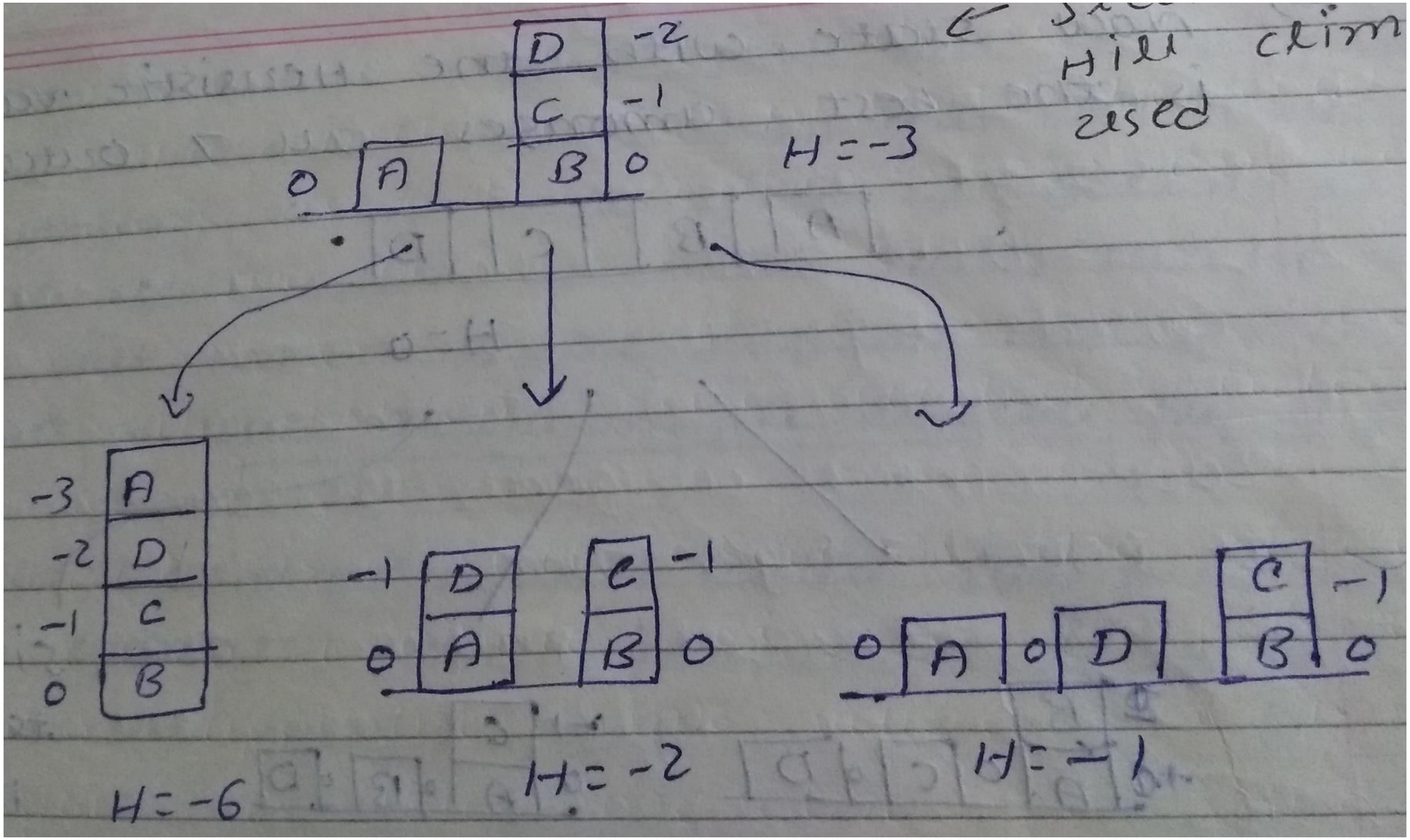
For C, -1 = -1

For B = 0

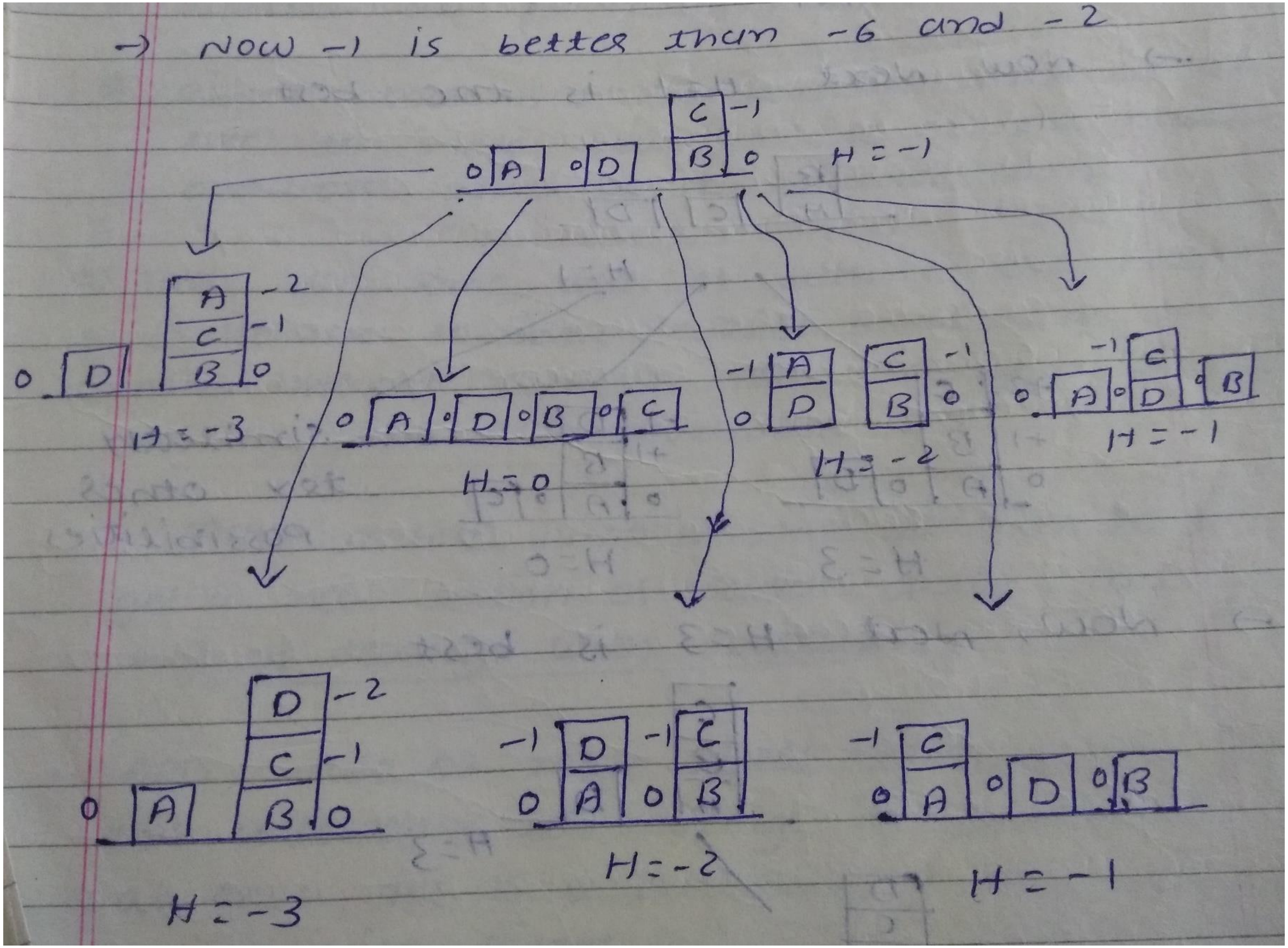
So total = -3

Hence it is better than Initial state

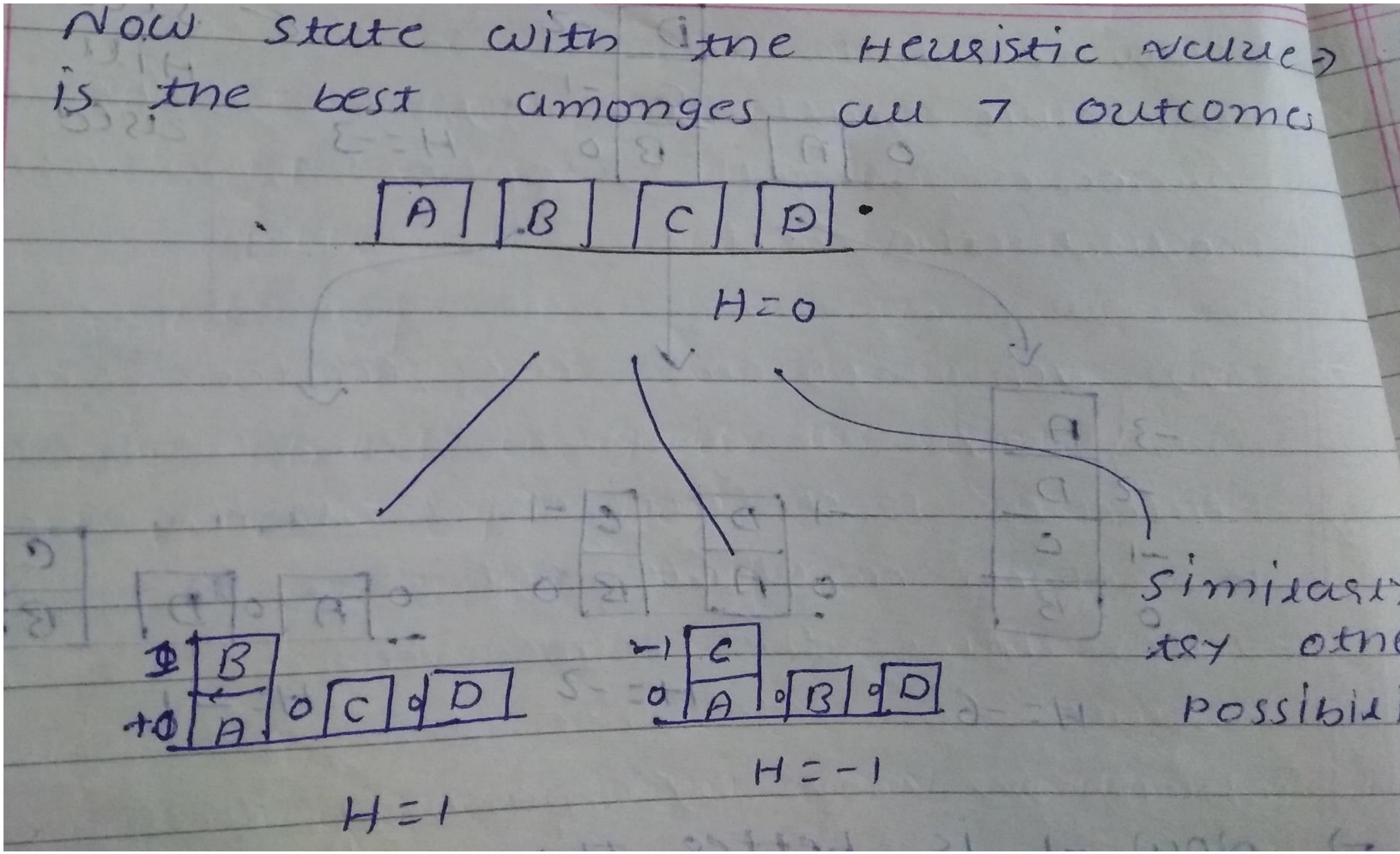
Blocks world problem



Blocks world problem

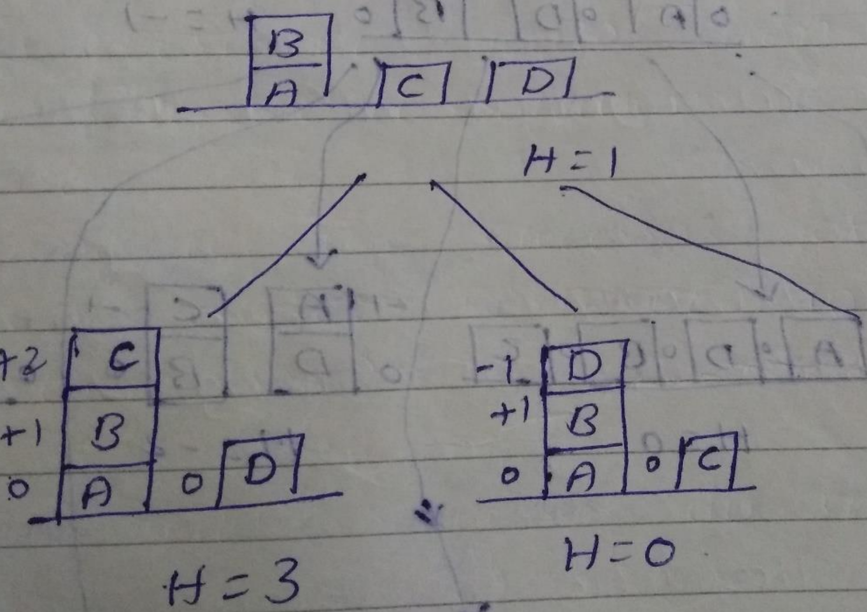


Blocks world problem



Blocks world problem

Now Next H=1 is the best



similarly try other possibilities

Now, Next H=3 is best