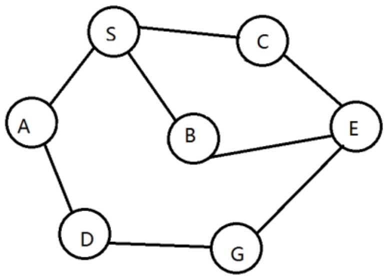
1. Search Algorithm
a. BFS:
UNIV->BRNG->STEW->HAAS->REC->PMU->LWSN->WTHR->HEAV->WTHR->ELLT

b. iterative deepening DFS:
h = 0: HEAV
h = 1: HEAV->PMU->WTHR
h = 2: HEAV->PMU->STEW->WTHR->ELLT->REC->SC
h = 3: HEAV->PMU->STEW->BRNG->UNIV->WTHR->ELLT->LWSN->REC->SC
h = 4: HEAV->PMU->STEW->BRNG->HAAS
c. Greedy search:
LWSN->ELLT->WTHR->HEAV->PMU

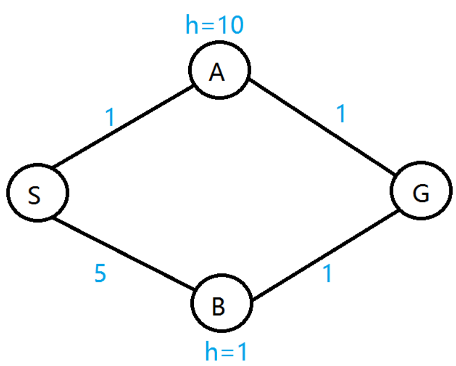
d. A*
LWSN->SC->WTHR->HEAV->PMU

2. Search and Heuristics
a.
suppose we have a graph like this:



From Start to Goal, if using BFS, the path will be: S->A->B->C->D->E->G
If using DFS, the path will be much simpler, which is S->A->D->G
The reason is that the breath of the search tree is wide, using BFS to search each tier which cost more time to approach to the goal. However, for DFS, the depth of this search tree is not deep. So that in this case, DFS is much better than BFS.

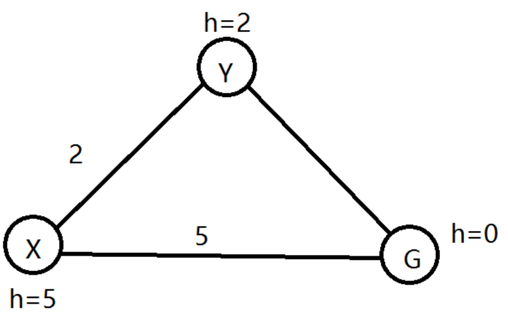
b.



This is a graph with an overestimating heuristic. In this case, from S to G, for heuristic search, the path will be S->B->G. the actual cost is 6. However, S->A->G will cost 2.
First, compare the heuristic costs between A and B, h(B)<h(A), so that will choose B but not A. the path is S->B->G is not optimal. The actual optimal goal is S->A->G which is 2.

c.
there are k values for h, which can be written as $h_1, h_2, h_3, \ldots, h_k$. $h^*(n)$ is the maximum of $h_1, h_2, h_3, \ldots, h_k$, which means that $h^*(n)$ is larger or equal to any values in $h_1, h_2, h_3, \ldots, h_k$.
Which $h_1 \leq h^*(n), h_2 \leq h^*(n), h_3 \leq h^*(n), \ldots h_k \leq h^*(n)$. It satisfies the admissible.
It cannot imply h^* is consistent.
Assume this is a graph which is a single path with all k nodes. All edge costs are equal to 1. So that $h^*(n_1) = k$, and make $h(n_1) = k - 1$, $h(n_i) = 1$ for $1 \leq i < k$ and $h(n_k) = 0$.
For every $h^*(n)$, which $h^*(n) \leq h(n)$.
It is not consistent because $h(n_1) = k - 1$, and cost of n_1 to n_2 is 1 and $h_1 = 1$. $k - 1 > 1 + 1 = 2$. Which $h(n_1) > C(h_1, h_2) + h(n_2)$

d.



Here is an example.

$$h(X) = 5, h^*(X) = 5$$

so that:

$$h(X) \leq h^*(X)$$

it satisfies admissible.
For X to Y

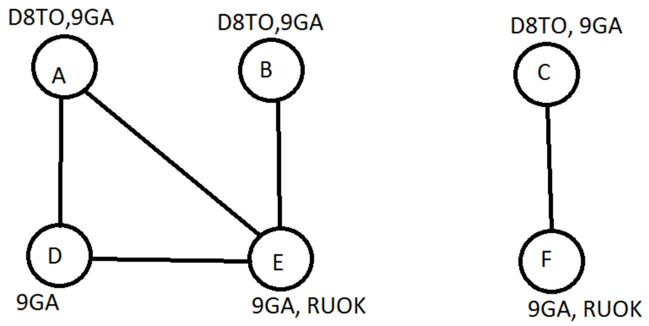
$$h(X) = 5, h(Y) = 2$$
$$cost(X\ to\ Y) = 2$$
$$cost(X\ to\ Y) + h(Y) = 4$$
$$h(X) > cost(X\ to\ Y) + h(Y)$$

which is not consistent.

3.CSP
a.
variables: $\{A, B, C, D, E, F\}$
Domain: $\{D8TO, 9GA, RUOK\}$
Constraints: $A \neq D, A \neq E, D \neq E, B \neq E, F \neq C$
Variable domains:

$$A: \{D8TO, 9GA\}$$
$$B: \{D8TO, 9GA\}$$
$$C: \{D8TO, 9GA\}$$
$$D: \{9GA\}$$
$$E: \{9GA, RUOK\}$$
$$F: \{9GA, RUOK\}$$

b.



c.

Assign D first because it has only one choice which is 9GA. A and E's domain will be decided which is D8TO and RUOK. B will not be influenced. It still has 2 choices (D8TO and 9GA). Assigned D8TO to C and F will not be influenced. In this way, when assigned variables to each domain, we do not need to traceback to the previous nodes to assign variables again. It will save much time to solve this problem.

In this way, the result for this problem will be:

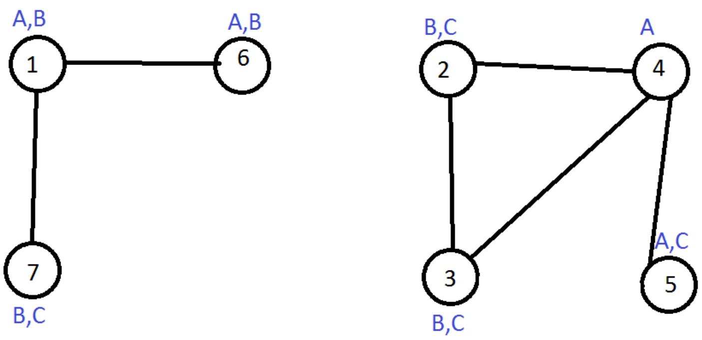
$$D: \{9GA\}, A: \{D8TO\}, E: \{RUOK\}, B: \{D8TO, 9GA\}, C: \{D8TO\}, F: \{9GA, RUOK\}$$

4. Arc Consistency

a.

$$c_1 \neq c_6$$
$$c_1 \neq c_7$$
$$c_2 \neq c_3$$
$$c_2 \neq c_4$$
$$c_4 \neq c_5$$

b.



$c_5 \rightarrow c_4$ is not arc consistency. For c_5, it can only choose C. For c_4, it can only choose A.
The variable domains after enforcing arc consistency will be:

$$c_1: \{A, B\}$$
$$c_2: \{B, C\}$$
$$c_3: \{B, C\}$$
$$c_4: \{A\}$$
$$c_5: \{C\}$$
$$c_6: \{A, B\}$$
$$c_7: \{B, C\}$$

c.
c_4 will be assigned first. The domain for c_4 is A. assigned c_4 first will only influence c_5, but will not influence c_2 and c_3.