For each of the following graph search strategies, work out the order in which states are expanded, as well as the path returned by graph search. In all cases, assume ties resolve in such a way that states with earlier alphabetical order are expanded first. The start and goal state are S and G, respectively. Remember that in graph search, a state is expanded only once.

(a) Depth-first search.
States Expanded: Start, A, C, D, B, Goal
Path Returned: Start-A-C-D-Goal

(b) Breadth-first search.
States Expanded: Start, A, B, D, C, Goal
Path Returned: Start-D-Goal

(c) Uniform cost search.
States Expanded: Start, A, B, D, C, Goal
Path Returned: Start-A-C-Goal

(d) Greedy search with the heuristic $h$ shown on the graph.
States Expanded: Start, D, Goal
Path Returned: Start-D-Goal

(e) $A^*$ search with the same heuristic.
States Expanded: Start, A, D, B, C, Goal
Path Returned: Start-A-C-Goal
2 Burrito Search Problem

You manage a taqueria. Making a burrito requires a set of tasks $T_1, \ldots, T_n$. Each task $T_i$ takes $t_i$ minutes to complete. Also, some tasks require other tasks to be completed before they can begin. You may schedule tasks among your employees. How fast can you make a burrito?

Concrete example: You have two ($e = 2$) employees and the following tasks, with requirements and times listed below:

1. steam tortilla, 1
2. cook steak, 3
3. add beans (steam), 1
4. add cheese (steam), 1
5. add steak (steam, cook steak), 1
6. wrap burrito (add all), 1
7. prepare to-go bag, 3
8. collect money from customer, 3
9. put burrito in a prepared bag (wrap, prepare), 1

(a) Formulate the problem of preparing a burrito as a search problem where you have an arbitrary number of employees $e$. In particular:

1. What is the state space?

   A set of finished tasks $F$, and for each employee $E_k$, a pair of $(T, t)$ indicating that $E_k$ started performing task $T$ at time $t$. Note that $T$ might be the empty task $T_{empty}$.

2. What is the initial search state?

   $F = \{\}, E_k = (T_{empty}, 0)$ for all $E_k$

3. What actions are available and what are the costs associated with an action?

   Actions are scheduling a task $T_i$ for $E_k$ at time $t$. Let $t_{last}$ be the time at which the last task in the current state will finish, and let $t_k$ be the time at which the last task for $E_k$ will finish. Then cost of an action is $\max(0, (t_k + t_i) - t_{last})$. In words, this is the amount by which the current action extends the last time at which any task will finish.

4. What is the successor function?

   Let $E_k$ be the employee whose current task $T_{first}$ will finish the earliest, and let $t_{first}$ be the time at which their task will finish. Let $U$ be the set of tasks which are neither in $F$ nor currently scheduled, $C$ be the set of tasks that are either in $F$ or currently scheduled to complete at time $t_{first}$, and $r(T)$ be the set of required tasks for $T$. Then the successor function is scheduling any $\{T \in U : r(T) \subseteq C\}$ at time $t_{first}$, and also placing $T_{first}$ in $F$. In words, we can schedule any task that is unfinished, and whose required tasks are either already finished or will finish at the same time as $E_k$’s current task.

5. What is the goal test?

   Check that the union of $F$ and all currently scheduled tasks contains all of $T_1, \ldots, T_n$. 
Propose an admissible heuristic for this problem. Explain why your heuristic is admissible.

A very simple heuristic comes from relaxing the constraint that tasks have dependencies. Let $t_{\text{max}}$ be the length of the longest remaining unscheduled and unfinished task $T_{\text{max}}$, $t_{\text{first}}$ be the time that the earliest currently scheduled task will finish, and $t_{\text{last}}$ be the time that the last currently scheduled task will finish. Then an admissible heuristic is given by $\max(0, (t_{\text{max}} + t_{\text{first}}) - t_{\text{last}})$, i.e. the additional time required to complete $T_{\text{max}}$ if it is delegated to the employee with the earliest available capacity. Another heuristic comes from recognizing that every remaining task must be done by some employee, and in the best case, every employee will be active at all times, so the remaining time is at least the sum of all times $t_i$ for unfinished and unscheduled tasks divided by $e$. 