The University of Michigan
Electrical Engineering & Computer Science
EECS 281: Data Structures and Algorithms
Spring 2024

Lab 10: Algorithm Families and Dynamic Programming

Instructions:
Work on this document with your group, then enter the answers on the canvas quiz.

15/15

Note:
This lab assignment contains a survey, multiple choice quiz, and coding portion. Submit your answers to the Lab 10 Canvas quiz and the coding portion to the autograder. You will have **three** attempts on the quiz.

You MUST include the following assignment identifier at the top of every file you submit to the autograder as a comment. This includes all source files, header files, and your Makefile (if there is one). If there is no autograder assignment, you may ignore this.

Project Identifier: D7E20F91029D0CB08715A2C54A782E0E8DF829BF

---

2 Algorithm Families and Dynamic Programming

3. Which algorithm seeks to break problems into smaller, non-overlapping subproblems then recursively combine the answers to those problems for the solution?

    A. Greedy

    B. Branch and Bound

    C. Divide and Conquer

    D. Dynamic Programming

    E. Brute Force

4. Which algorithm exhaustively tries every possible solution to a problem in order to find the optimal solution?

    A. Greedy

    B. Branch and Bound

    C. Divide and Conquer

    D. Dynamic Programming

    E. Brute Force

5. Which algorithm will break a problem into subproblems and combine the solutions to those while recording the solutions to those subproblems in order to avoid solving the same subproblem twice?

    A. Greedy

    B. Branch and Bound

    C. Divide and Conquer

    D. Dynamic Programming

    E. Brute Force

6. Which algorithm will take a locally optimal choice at each step of the algorithm, but might not necessarily be globally optimal?

    A. Greedy

    B. Branch and Bound

    C. Divide and Conquer

    D. Dynamic Programming

    E. Brute Force

7. Which algorithm will discard potential solutions which are worse than the current best solution?

    A. Greedy

    B. Branch and Bound

    C. Divide and Conquer

    D. Dynamic Programming

    E. Brute Force

8. What kind of algorithms are Prim's and Kruskal's?

    A. Brute Force

    B. Greedy

    C. Divide and Conquer

    D. Branch and Bound

    E. None of the above

9. What kind of algorithm is selection sort?

    A. Brute Force

    B. Greedy

    C. Divide and Conquer

    D. Branch and Bound

    E. None of the above

10. What kind of algorithms are quicksort and mergesort?

    A. Brute Force

    B. Greedy

    C. Divide and Conquer

    D. Branch and Bound

    E. None of the above

11. Your company is working on a map application but has run into a roadblock in their algorithm — finding the shortest route from point A to point B takes too long for practical use! Upon further analysis, you realize that the algorithm often does unnecessary work by traversing paths that are worse than the current optimal path. What algorithm can you use to fix this problem?

    A. Brute Force

    B. Greedy

    C. Divide and Conquer

    D. Branch and Bound

    E. None of the above

12. Which of the following statements is/are TRUE? Select all that apply.

    A. A brute force solution will always give you the optimal solution.

    B. Because backtracking avoids looking at large portions of the search space by pruning, the asymptotic complexity of backtracking is always better than that of brute force.

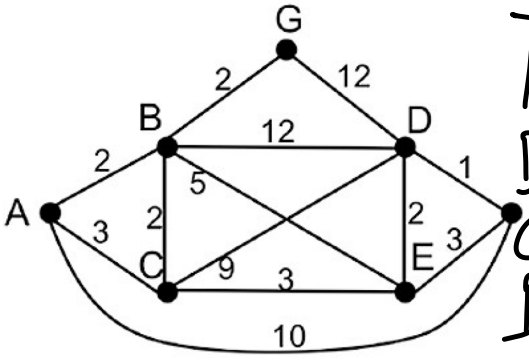
    C. The greedy algorithm guarantees an optimal solution to the 0-1 knapsack problem.

    D. Branch and bound will not speed up your program if it takes just as long to determine the bounds than to test all possible choices.

    E. Dynamic programming will always reduce both the time and memory needed to solve a problem with multiple overlapping subproblems.

    F. given n items and a knapsack capacity of m, the dynamic programming solution to the 0-1 knapsack problem runs in $\Theta(mn)$ time.

13. You are using Dijkstra's algorithm to find the shortest path from vertex A to every other vertex in the graph above. S is the set of vertices for which the minimum path weight from A has been found. A is the first vertex you add to S. Which of the following gives the correct order in which vertices are added to S?



	1	2	3	4
A	0	0	0	0
B	2	2	2	2
C	3	min(2+12,3) = 3	3	3
D	∞	2+12=14	min(14, 3+9)=12	min(12, 4+12)=12
E	∞	2+5=7	min(7, 3+3)=6	min(6, 4+∞)=6
F	∞	∞	∞	∞
G	∞	2+2=4	min(4, ∞)=4	4

    A. A, B, C, E, G, D, F

    B. A, B, G, C, E, D, F

    C. A, C, B, G, E, D, F

    D. A, B, G, D, C, E, F

    E. A, B, C, G, E, D, F

14. Consider the recurrence relation

$$T(n) = T(n-1) + T(n-3) + T(n-4)$$

    $[n > 4]$ with $T(n) = n$ for $1 <= n <= 4$. What is the time complexity of the best algorithm for calculating $T(n)$ with $O(n)$ additional space?

    A. $O(1)$

    B. $O(\log(n))$

    C. $O(n)$

    D. $O(n^2)$

    E. $O(3^n)$

15. Which of the following is **true** about dynamic programming?

    A. A dynamic programming solution for calculating the N^{th} fibonacci number can be implemented with $O(1)$ additional memory

    B. Dynamic programming is mainly useful for problems with disjoint subproblems

    C. A bottom-up DP solution to a problem will **always** use the same amount of stack space as a top-down solution to the same problem

    D. A top-down DP solution to a problem will **always** calculate every single subproblem.