18k-0205.pdf

by

Submission date: 24-Jan-2021 09:51AM (UTC+0500)
Submission ID: 1493073908
File name: 18k-0205.pdf (152.66K)
Word count: 1549
Character count: 7213

PROJECT REPORT

DESIGN AND ANALYSIS OF ALGORITHM

DEPARTMENT OF COMPUTER SCIENCE

Submitted to:

SIR ZEESHAN KHAN

Submitted by:

SARAH FATIMA (18K-0205)

IRZA KAMRAN (18K-1097)

Sec: A

National University of Computer and Emerging Sciences-FAST

Karachi Campus

Abstract:

In this project, we have used 10 different dynamic programming algorithms to produce optimal solutions for various problems. These include algorithms to find the longest common sequence and the shortest common supersequence or to solve the rod cutting and the coin change making problems. The user can choose an input file from the random datasets generated and a valid algorithm for that particular dataset. This algorithm is then executed to give optimal solution.

Introduction:

Dynamic programming is used to solve complex problems, by dividing them into similar sub-problems, which are solved just once and their optimal solutions are stored. Before solving the in-hand sub-problem, dynamic algorithms will try to examine the results of the previously solved sub-problems. The solutions of sub-problems are then combined in order to achieve the best overall solution. Mostly, these algorithms are used for optimization.

1. **Longest Common Subsequence**[1]: This algorithm finds the longest subsequence that is present in any two given sequences in the same order, but not necessarily contiguous.

2. **Shortest common subsequence**[2]: Given two strings and, this algorithm finds the length of the shortest string that has both strings as subsequences.

3. **Longest increasing subsequence**[3]: Finds the length of the longest subsequence of a given sequence such that all elements of the subsequence are sorted in increasing order.

4. **The levenshtein edit distance**[4]: Finds minimum number of edits (operations) required to convert one string into another.

5. **Matrix chain multiplication**[5]: Given a sequence of matrices, this algorithm finds the most efficient way to multiply these matrices together.

6. **0/1 knapsack problem**[6]: Given a set of items, each with a weight and a value, we need to determine the maximum total value obtainable by choosing items so that the total weight is less than or equal to a given limit

7. **Partition problem**[7]: This algorithm determines whether a given set can be partitioned into two subsets such that the sum of elements in both subsets is the same.

8 . **Rot cutting problem**[8]: This algorithm determines the maximum value obtainable by cutting up the rod and selling the pieces, given the price of each length.

9. **Coin changing problem**[9]: Given an unlimited supply of coins of given denominations, this

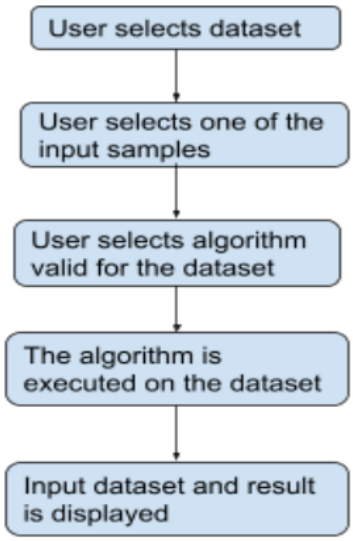
algorithm finds the minimum number of coins required to get a desired change

10. **Word break problem**: Given an input string and a dictionary of words, this algorithm finds out if the input string can be segmented into a space-separated sequence of dictionary words.

We will implement these algorithms to read input data from files, process it and display the optimal solution to the user. We have used Python 3 for implementation. A cross-platform GUI toolkit PyQt is used to create a sleek user interface. C++ is used to produce input datasets of random values that satisfy the given criteria. The algorithms used to generate the datasets are attached in the 'Generate Datasets' folder.

Proposed System:

We ask the user to choose one of the four kinds of datasets and to choose an input file containing one of the ten sample datasets generated. The user then chooses one from a list of valid algorithms that can be run on this input file. These choices are made through drop down menus. The chosen algorithm executes and the output is displayed to the user. The input set is also displayed to the user.



Experimental Setup:

Sample input data sets were created to test the aforementioned dynamic algorithms. The data sets were produced by algorithms designed to generate sample data according to the specifications defined in the Project Description document. The dynamic algorithms then took the data sets as input and processed them to give optimal results.

There are 4 different kinds of data sets, each valid:

1. For LCS, SCS and Edit Distance
2. For LIS, MCP, Coin Change and Partition Problem
3. For the Word Break Problem
4. For the Knapsack and Rod Cutting Problem

One set of values generated by each (all 10 that were generated are in the folder submitted):

1. kzkrmznmkrnkzaaanniarakaarrmnranrrzrizanaaazzaakaakna atsaaiataaairhhafamiairfimrraattarataamaaamtsahaaaaht
2. 61 61 89 2 39 25 15 21 7 46 47 100 43 35 44 6 91 30 78 17 56 26 83 17
3. nerrixbg, yupdnhcpc, fskhe, xnzm, yfzpedn, rke, wu, rtvuqbato, ei, beryfqo, t, ujvmos, js
4. V = [22 19 10 15 50 89 15 77 69 15 1 82 47 71 10 15 15 85 23 88 55 65 91 25 88 78]
   W = [58 28 29 8 86 35 22 61 24 2 40 64 23 32 7 90 54 24 77 13 26 75 96 43 78 1]

Results/Discussion:

	1	2	3	4	5	6	7	8	9	10
LCS	ariaaraa aaaaaaa riararaa aarrrm mmiaaa araa	arraiaaa rmaara maam	aaaiaaaa rmarrraa aaaaaaa	maaaara araiaaaa aamaam ama	amaaarii miaama ariaaa	amaaam aaaaaaa aaaimrra amaraa mamaaa aaaa	rammrar aaaarair aaaaara aamama aaaaam aaaram	aaaaaaai raaaam mrimaia rmaaaar amamaa arraramr aa	aaraaaa aarriaaa aaaaaaa aaaaaari rmamaa rra	raamaaa amaaraa amaaaai amaama aaaaaa
SCS	143	64	83	94	78	123	144	156	123	121
Edit	71	33	40	45	41	59	71	72	74	60
LICS	2 5 7 9 12 14 37 41 43 50 58 65 66 83 85 90	0 14 21 23 30 33 34 40 53 75 81 88 99	2 4 8 10 16 20 29 37 41 47 52 57 62 73 83	0 1 2 13 22 26 27 70 82 89 95 99	2 11 15 17 22 32 36 44 61 71 75 88 94	1 5 8 13 24 33 39 40 42 47 49 56 57 59 68 85 92	40 43 61 69 83 89 94 96 99	2 15 21 35 44 56 83	3 4 7 22 26 37 52 53 62 68 76 84	9 11 16 24 36 41 43 48 54 58 76 80

MCP	((A(B(C(D(E(F(G(H(I(J(K(L(M(N(O(P(Q(R(S(T(U(V(W(X(Y(Z([\))))))))))))))))))))))))))	((A(B(C(D(E(F(G(H(I(J(K(L(M(N(O(PQ))))))))))))))))	((A(B(C(D(E(F(G(H(I(J(K(L(M(N(O(P(Q(R(S(T(U(V(W(X(YZ)))))))))))))))))))))))))))	((A(B(C(D(E(F(G(H(I(J(K(L(M(N(O(P(Q(R(S(T(U(V(W(X))))))))))))))))))))))))	((A(B(C(D(E(F(G(H(I(J(K(L(M(N(O(P(Q(R(S(T(U(V(W(X(Y(Z([(\(](^_)))))))))))))))))))))))))))))(((((((((`a)b)c)d)e)f)g)h)i))	((A(BC))(((((((((((((DE)F)G)H)I)J)K)L)M)N)O)P)Q)R)S)T)U)V)W))	((A(B(C(D(E(F(G(H(I(J(K(L(M(N(O(P(Q(R(S(T(U(V(W(X(Y(Z([(\(](^(_(`a))))))))))))))))))))))))))))))))))((((((((((((bc)d)e)f)g)h)i)j)k)l)m)n)o)p)q))	((A(B(C(D(E(F(G(H(I(J(K(L(M(N(O(P(Q(R(S(T(U(V(W(X(Y(Z([(\(](^(_(`)))))))))))))))))))))))))))))))))))	((((((((((((((((((((ab)c)d)e)f)g)h)i)j)k)l)m)n)o)p)q)r)s)t)u)v)w)x)y)z){)|))	((A(BC))(((((((((((((((((((DE)F)G)H)I)J)K)L)M)N)O)P)Q)R)S)T)U)V)W))
Coin	2	1	1	1	2	2	2	2	1	2
Part	yes	no	no	no	yes	no	no	yes	no	yes
Word	No	No	No	No	no	No	No	no	no	No
Knap	3697	8536	4365	4273	4844	2703	8536	3880	5723	2134
Rod	3679	8536	4365	4273	4844	2307	8536	3880	5723	2134

The table shows the output produced by the algorithms when they are tested on each of the ten datasets. Each algorithm correctly produces the optimal solution within time constraints.

Conclusion:

It can be concluded that dynamic programming algorithms that are used, operate correctly and produce the optimal results when executed on valid data, as shown in the table in the Results section.

References:

1. "Longest Common Subsequence | DP-4 ...." 30 Sep. 2019, https://www.geeksforgeeks.org/longest-common-subsequence-dp-4/. Accessed 16 Jan. 2021.
2. "Shortest Common Supersequence ...." 19 Nov. 2020, https://www.geeksforgeeks.org/shortest-common-supersequence/. Accessed 16 Jan. 2021.
3. "Longest Increasing Subsequence | DP-3 - GeeksforGeeks." 18 May. 2020, https://www.geeksforgeeks.org/longest-increasing-subsequence-dp-3/. Accessed 16 Jan. 2021.
4. "Edit Distance | DP-5 - GeeksforGeeks." https://www.geeksforgeeks.org/edit-distance-dp-5/. Accessed 16 Jan. 2021.
5. "Matrix Chain Multiplication using Dynamic Programming." https://www.techiedelight.com/matrix-chain-multiplication/. Accessed 16 Jan. 2021.
6. "0-1 Knapsack Problem - Techie Delight." https://www.techiedelight.com/0-1-knapsack-problem/. Accessed 16 Jan. 2021.
7. "Partition problem | DP-18 - GeeksforGeeks." 23 Dec. 2020, https://www.geeksforgeeks.org/partition-problem-dp-18/. Accessed 16 Jan. 2021.
8. "Rod Cutting Problem - Techie Delight." https://www.techiedelight.com/rot-cutting/. Accessed 16 Jan. 2021.
9. "Coin change-making problem (unlimited supply of coins ...." https://www.techiedelight.com/coin-change-making-problem-unlimited-supply-coins/. Accessed 16 Jan. 2021.
10. "Word Break Problem | DP-32 ...." https://www.geeksforgeeks.org/word-break-problem-dp-32/. Accessed 16 Jan. 2021

18k-0205.pdf

ORIGINALITY REPORT

34%	30%	9%	9%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	nuokui.com Internet Source	10%
2	ispylog.blogspot.com Internet Source	5%
3	pt.scribd.com Internet Source	5%
4	zeth.ciencias.uchile.cl Internet Source	3%
5	Shahpasand, Maryam, and Sayed Alireza Hashemi Golpayegani. "Optimum Countermeasure Portfolio Selection", Emerging Trends in ICT Security, 2014. Publication	3%
6	Submitted to Higher Education Commission Pakistan Student Paper	3%
7	www.tutorialspoint.com Internet Source	2%
	www.geeksforgeeks.org	

8	Internet Source	2%
9	Krishnendu Chatterjee, Andreas Pavlogiannis, Viktor Toman. "Value-centric dynamic partial order reduction", Proceedings of the ACM on Programming Languages, 2019 Publication	1%
10	www.coursehero.com Internet Source	1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	On		

18k-0205.pdf

GRADEMARK REPORT

FINAL GRADE

/0

GENERAL COMMENTS

Instructor

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5