Phase – 2 :
Analysis of Algorithms that Solve Cutting Stock Problem
(Design & Analysis of Algorithms)

ABSTRACT

When given unlimited materials of a set length and orders placed for materials of varying smaller lengths along with their quantities, a problem arises on how to divide the given material into the smaller length ones along such that the all the orders get satisfied along with an optimization function. That can be in the form of minimum wastage, cost, number of items or maximum profit, etc. Termed as the Cutting Stock Problem (CSP), this can be seen during various industrial production and supply planning stages for cutting paper, glass, steel and other materials. It can be 1-dimensional or multidimensional. For example, 1-Dimensional CSP (1D-CSP) would mean cutting rods of a certain length from a larger metal rod and 2-Dimensional CSP (2D-CSP) could be cutting rectangular shapes from a larger rectangular sheet. In the classical 1D-CSP, all the different combinations of cuts, named “patterns", are listed and each one is assigned a decision variable. Such a typical model has two constraints and an optimization function. The first constraint is for calculating wastage at each individual cutting step and the second constraint is to ensure that the items cut from the stock match the demand. So, 1D-CSP can be considered as an Integer Linear Program (ILP).

Due to the number of patterns growing exponentially as the number of ordered materials increase, it becomes impractical to manually list them all. A proposed solution to this problem by Gilmore and Gomory is to use delayed column generation. Start by analyzing a few patterns and introduce new ones when needed. For 1D-CSP, new patterns are generated by an auxiliary optimization technique, known as the knapsack problem, by referencing variable information from the LP. However, this method doesn’t support integrality and could suggest a fractional solution. Rounding it off to the previous or next number would lead to underproduction or overproduction. Luckily, modern algorithms like cutting planes or branch-and-bound can overcome this limitation and provide optimal integer solutions even for large orders. Some other general limitations are that most algorithms assume no defects in the stock or that there are only rectangular items (for multi-dimensional CSP). These can lead to low quality results, poor actual utilization and limited usage in real-life. However, if these limitations were dealt with, they would require additional decision variables, pattern formations and eventually increase computational time. Thus, overcoming those challenges come at the cost of increased time.

INTRODUCTION

Cutting Stock Problem refers to the problem of cutting the stock material into standard sizes such that the waste is minimized and the demand is satisfied. CSP is an NP Hard problem which is reducible to the Knapsack problem. It is formulated as an integer linear programming problem with one decision variable. [1] The possible combinations of the material being cut plays an important role.

The Cutting Stock Problem was first formulated in 1939 by Kantorvinch. He proposed to solve the problem by using the material economically with the help of linear programming.
CSP can be classified based on dimensionality, assortment of large objects, assortment of small items and kind of assignment. [3]

1. Dimensionality - Based on dimensionality CSP can be classified into 1D and 2D problems
2. Assortment of Large Objects - One, Many and Different large objects.
3. Assortment of Small Items - Many identical and different items.
4. Kind of assignment - Selection of large objects along with all the small items and vice versa.

The most common ones are 1D and 2D Cutting Stock Problems. 2D problems are harder to solve because of the cutting feasibility issue. Generating 2D patterns is difficult and the time complexity is more. [2]
The mathematical formulation of the CSP is as follows:
Sets:
Parameters:
Decision Variable:

The CSP problem is formulated as:
Some common approaches used in solving CSP are Linear Programming, Column Generation, Sequential Heuristic and Hybrid Solution procedures. There's high degeneracy in the problem owing to the different ways in which the items can be moved around without any change in the amount of waste generated. This degeneracy of the CSP gives rise to a set of problems which include the minimum pattern cut problem, minimum stack problem, minimum number of knife changes problem. [1]

CSP has a wide range of real-world applications. Some of them are discussed below.
1. Assortment Problem - Determining the best piece out of all the stock materials that was cut from the masterpiece,
2. Winder Constraints - A certain number of slitting knives are available and the stock pieces should be cut in such a way that the maximum limit is not exceeded.
3. Guillotine Problem – The stock pieces should be cut in rectangular shapes and the cut should extend all the way across the sheet.

In this paper, we focus on the cutting rod problem. According to the problem, given a rod of length n and a list of prices of all the pieces of size lesser than n, we need to find a maximum obtainable price. The solution to this problem can be obtained by considering all the possible combinations in which the rod can be cut and then computing the price of the pieces [4]. However, the time complexity of this solution is exponential. Some of the optimal solutions to solve the cutting rod problem is discussed below.

1. Optimal Substructure: This approach uses recursive call of the cutting rod function to solve the problem. Arbitrary cuts are made in the rod given and the price obtained is calculated after each cut to compare the values.
The syntax of the recursive function is as follows:
2. Overlapping Subproblems: A decision tree is created; at every node the problem gets branched leading to subproblems. Each subproblem is solved in the recursive call. This problem has the subproblem and the dynamic programming property.

3. Unbounded Knapsack: The unbounded knapsack problem has multiple occurrences of the same item. The cutting rod problem can be solved using the same idea. The length and price variables of the cutting rod problem is analogous to the size and value of the item in the Knapsack problem.

PROBLEM DEFINITION

Cutting-stock problem is a NP-hard optimization problem of cutting standard-sized pieces of stock material into specified sized pieces while minimizing cost and maximizing stock utilization. Hence, due to the NP-hardness of this problem, it is important to solve it efficiently. The problem can be classified based on the dimensions of the stock like 1-D CSP, 2-D CSP etc. and the 1-D CSP problem can be formulated as a linear programming problem. It can be represented mathematically with constraints like so:

Where m is the number of orders, with q_j pieces each, n is the number of all possible combinations of cuts or patterns, x_i represents how many times a pattern i is used, c_i is the cost/waste of the pattern and a_{ij} is number of times order j appears in a pattern i.

OBJECTIVES:

1. To analyze the algorithms proposed to solve the Cutting Stock Problem (CSP), the methods, objectives, limitations and performance evaluation metrics of the same.
2. To critically evaluate the algorithms and present the parameters used by the algorithms.
3. Determine the effect of parameters in the efficiency rate and compare the performance of the other algorithms that use the same parameters.
4. Study the methods of the works that have been successful in the past and incorporate the same with improvisations in the current work.
5. The efficiency of methods when used together and the enhancement of the traditional algorithms is discussed.
6. Identify possible gaps in the literature for future work.

LITERATURE TABLE

REFERENCES	OBJECTIVES	PROBLEM STATEMENT	METHODOLOGY	DATASET	ALGORITHM	ADVANTAGE	DISADVANTAGE	PERFORMANCE

								MEASURE VALUES
1. Bonnevay, Stéphane, et al. "A Genetic Algorithm to Solve a Real 2-D Cutting Stock Problem with Setup Cost in the Paper Industry." Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation, 2015. Crossref, doi:10.1145/2739480.2754660.	1. Implement a genetic algorithm to solve the combinatorial part of the 2D Cutting Stock Problem with Setup Cost. 2. Experiments are to be realized on real and artificial datasets. 3. Satisfy customer demands while minimizing the total production cost (stock sheet cost and setup cost).	The Cutting Stock Problem with Setup Cost requires the cost of setup and the cost of the stock sheet printed so that it can minimize the total production cost. Most existing papers deal with one dimensional problems. This paper gives a solution to the Two-Dimension al Cutting Stock Problem with Setup Cost.	The goal consists of satisfying customer demands and minimizing stock sheet cost and setup cost by automatically fixing: the number of patterns, the number of rectangular images and stock sheets to be printed for each pattern and the positioning of each rectangular image on patterns. The number of objects (rectangular images) to be set on each pattern is unknown (only the demands are known), hence the objective is to find the best number of each image on each pattern (and to find the optimal number of patterns). Genetic algorithm used to find the proper number of each image on each pattern. a 2-D bin packing algorithm and a classical linear programming solver were used to find the placement of all images and stock sheets to be printed for each pattern	20 real datasets of the Seripress company	GA-2D CSP-S, combines the Crossover and the Mutation operators in a classical genetic algorithm.	Automatizati on of the process, the reduction of the manufacturin g time and the global cost reduction.	Limitation of the computation time and the algorithm has very small memory footprint.	The mean gain of stock sheets printed was 26.1%, and mean percentage of cost improvement was 22.2%

2. M. Fathy, M. Osama and M. S. El-Mahallawy, "Evolutionary-based hybrid algorithm for 2D cutting stock problem," 2015 IEEE Seventh International Conference on Intelligent Computing and Information Systems (ICICIS), Cairo, Egypt, 2015, pp. 454-457, doi: 10.1109/IntelCIS.2015.7397260 .	1. Design an algorithm that minimizes trim loss. 2. To detect and isolate defective stock sheets using Image Processing. 3. Compare the existing methods with the proposed algorithm to evaluate the results obtained.	Algorithms proposed so far involve the use of traditional methods such as evolutionary programming(EP), linear programming (LP), etc. The efficiency of these algorithms when used along with any other algorithm is yet to be explored.	Image of the stock sheet is captured and converted into a black and white version. Check the image obtained to see if it is a defective or a regular sheet using image processing. If the image is a defective sheet then exit. If not, the image is processed to get the maximum area of the rectangle and the data is stored in the database. The variables length, width and count store the dimensions of the sheet and the amount of sheet available. Repetitive waste and last sheet waste are calculated using the fitness formula. The fitness value is compared to the fitness function value to evaluate the results. In each iteration a new population is randomly selected. The process is repeated till the solution satisfies the minimum waste percentage allowed condition.	Dataset of a local Dream Stone plant	Greedy Knapsack Algorithm	Capable of minimizing trim loss of regular sheets.	The algorithm doesn't work for irregular and defective sheets.	Results obtained from Evolutionary Programming (EP)
3. K. B. Parmar, H. B. Prajapati and V. K. Dabhi, "Cutting stock problem: A solution based on novel pattern based chromosome representation using modified GA," 2015 International Conference on Circuits, Power and Computing Technologies [ICCPCT-2015], Nagercoil, India,2015, pp. 1-7, doi: 10.1109/ICCPCT.2015.7159318 .	1. Implement a Modified Genetic Algorithm that solves single and multiple stock size cutting stock problems (CSP). 2. Build chromosome representations for Single and Multiple stock size. 3. Perform a comparative study on the other algorithms that solve the CSP problem	Most of the works take length, width and the amount of stock required as parameters but fail to filter out the negatives . A model that filters at every iteration is needed to minimize the trim loss.	Initialize a feasible pattern and choose the best 10% of the pattern, store it in the best pattern set. Two population sets are formed from the patterns. Population set1 contains the best_pattern and the feasible_pattern. Population set2 contains the feasible_pattern and a randomly selected set of patterns. Patterns are extracted from these two population sets to get the solution_pattern. A newly generated feasible pattern set is formed by choosing the fittest pattern from the feasible_pattern set. Mutation is applied on the best_pattern set	Gene pattern.	Modified Genetic Algorithm.	Accuracy rate is higher compared to the other traditional methods.	The algorithm analyzes and builds the chromosome representation for each stock which is time consuming.	Accuracy and Speed Of Convergence.

			and the new_feasible_pattern set.					
4. T. Y. Lin, S. M. Chen and M. T. Yu, "Solving the cutting-stock problem by using the Sequential Quadratic Programming optimization method," 2016 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM), Bali, Indonesia, 2016, pp. 1699-1702, doi: 10.1109/IEEM.2016.7798167.	1. Arrange the stock based on minimum material wasted using SQP. 2. Increase the utility rate and reduce the cost of the stock. 3. To avoid local minimum point for Multiple Stock Size CSP.	Previous methods do not consider irregular sheets and multistock size CSP but real datasets contain defective and irregular sheets. An algorithm that solves this problem is presented in this paper.	Represent the stock on a 2D plane and apply the optimization inequality. Divide the problem into n-sub problems. Lagrange function is applied to each sub problem. Equal step size is calculated. The algorithm is repeated till a minimum waste rate is achieved.	ESICUP database	Sequential Quadratic Programming (SQP)	The proposed algorithm works for multiple stock size CSP as well as irregular sheets. It outperforms the genetic algorithm.	Only the utility rate of the stock is considered.	Dagli, Swim, Albano and Blaz2 patterns.
5. Cui, Yaodong, et al. "New Model and Heuristic Solution Approach for One-Dimensional Cutting Stock Problem with Usable Leftovers." *Journal of the Operational Research Society*, vol. 68, no. 3, 2017, pp. 269–80. *Crossref*, doi:10.1057/s41274-016-0098-y.	1. Implement a new model and a 2 phase algorithm for 1D-CSP with usable leftovers to obtain minimal trim loss and low computational time. 2. Prove its superiority by testing the new algorithm with 2 previous ones using the same dataset. 3. Add an extra constraint on no.of leftovers for better pattern reduction.	Previous papers have proposed algorithms to solve 1D-CSP but fail to acknowledge the effect unused leftovers can have on the cutting process complexity, storage area and trim loss. The heuristic TPBNT algorithm uses an integer linear programming model with a column generation process to combat that.	Phase-1: Solve the linear relaxation of the current residual problem. Allow some patterns into the Phase-1 solution and update the remaining demands. Add all the generated patterns that have positive frequency to the total. Phase-2: When all demands are met, use an MILP solver for the linear programming model to obtain Phase-2 solution. Select the better one of the Phase-1 and Phase-2 solutions.	Cherri's 16 generated classes of instances.	TPBNT which is an algorithm that considers an upper bound on no.of leftovers used	Multiple leftovers generation leads to better material utilization, considering an upper bound on number of leftovers keeps inventory level of leftovers under control.	Cutting workload slightly increases due to multiple leftovers being generated.	Trim-loss length, material utilization, computation time, no.of leftovers in stock.
6. "A New Heuristic Algorithm for Two-Dimensional Defective Stock Guillotine Cutting Stock Problem with Multiple Stock Sizes." *Tehnicki Vjesnik - Technical Gazette*, vol. 22, no. 5, 2015. *Crossref*, doi:10.17559/tv-20150731113849.	1. Model an algorithm that takes rectangular cuts of various lengths and counters defectives like shrinkage, holes, etc. in stock. 2. Demonstrate its improvement as per utilization and reuse rate.	Usually, most algorithms assume that there are no defects in the stocks. However, this is unrealistic and there has to be an algorithm that takes this into account.	Stocks are arranged in ascending order of largest edge and waiting-pack-items in descending order. Select minimum stock that's available. Keep inserting till there's no items or waiting-pack-items for 1st type cavity merging. The 2nd type cavity is merged and the algorithm moves the cavity to a corner to find a waiting-pack-item that can be inserted there. Keep going till there's no waiting-pack-items or items that can fit in a cavity. A hit-test identifies defect location.That defective stock is eliminated and the algorithm runs again to fill items in that cavity and places defective	Vassiliadis's aircraft manufacturing enterprise pieces.	MS2DDSGCSP algorithm which	By reducing remainder fragmentation, stock utilization rate and remainder reuse rate improves. It considers single and multi-defected stocks.	Still assumes all shapes are rectangular. The algorithm can recognize guillotine cuts only.	Utilization ratio percentage, reuse value percentage, quantity of remainders.

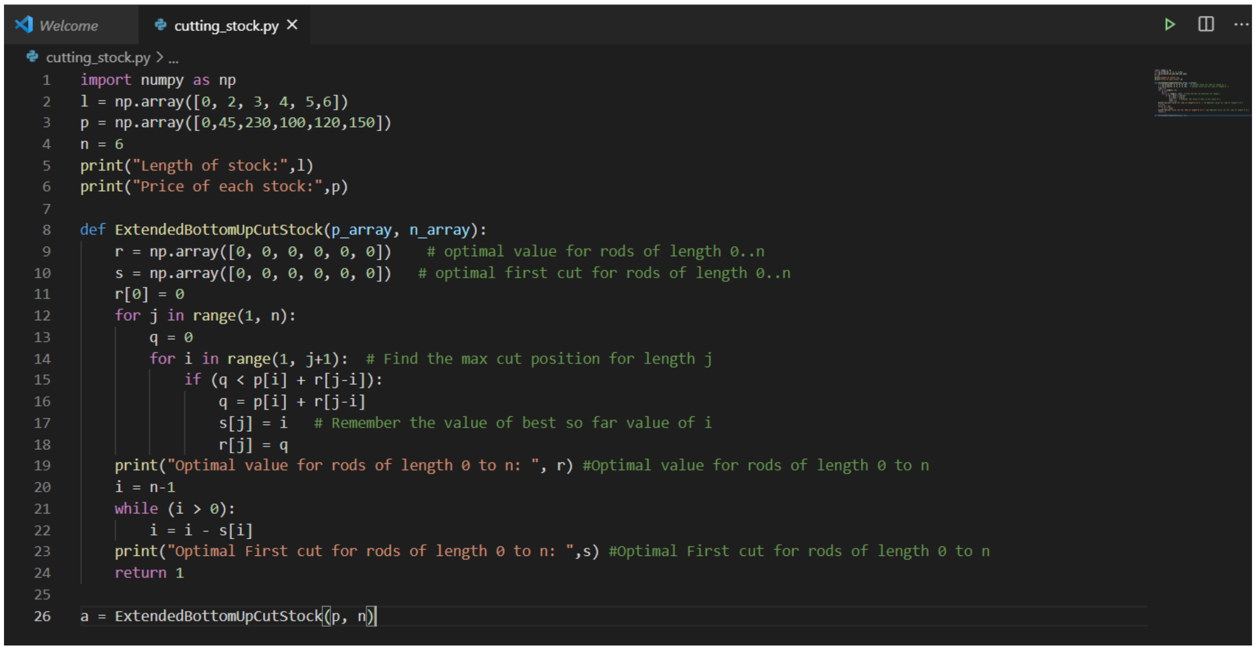
			items far away from others.					
7. MirHassani, S. A., and A. Jalaeian Bashirzadeh. "A GRASP Meta-Heuristic for Two-Dimensional Irregular Cutting Stock Problem." *The International Journal of Advanced Manufacturing Technology*, vol. 81, no. 1–4, 2015, pp. 455–64. *Crossref*, doi:10.1007/s00170-015-7107-1 .	1. Propose an algorithm for 2D-CSP with irregular shapes that minimizes material wastage using the dotted-board model. 2. Experiment to find the parameter combination that gives the most optimal pattern with minimum wastage.	Most works have algorithms dealing with only large rectangular shaped cuts and not irregular ones. This cannot be applied in real-life and so, there must be an algorithm dealing with irregular cuts as well.	Greedy Randomized Construction (GRC) generates the dataset. Pre-processing step checks for overlapping by comparing the 0-1 matrices of both shapes. Then, each shape's dotted mapping is shown in a linear program form and solved by the algorithm.	3 classes of instances generated by GRC.	GRASP and Reactive GRASP (RGRASP) algorithms.	With RGRASP, there's no need to manually adjust the Restricted Candidate List (RCL) parameter, saving time during the GRC process.	Larger shaped pieces indirectly increase the number of decision variables, causing the algorithm to have more iterations and thus, more computational time.	Average gap percentage, computational time, success rate.
8. Clautiaux, François, et al. "Pattern-Based Diving Heuristics for a Two-Dimensional Guillotine Cutting-Stock Problem with Leftovers." *EURO Journal on Computational Optimization*, vol. 7, no. 3, 2019, pp. 265–97. *Crossref*, doi:10.1007/s13675-019-00113-9.	1. Design an algorithm that maximizes leftover material length so that it could be reused for the next cutting stage. 2. Discuss the partial enumeration technique which helps in reducing suggestion of improper patterns as solutions.	Any extra material left from a cutting stage is regarded as waste and discarded. Is there a way to make use of that extra material such that wastage production is reduced?	Apply Dantzig-Wolfe decomposition and column generation to get extended pseudo-polynomial size formulation. The hypergraph based heuristic algorithm solves the bounded case of the pricing problem. Then, constructive heuristics are derived and used in computational comparisons with diving heuristics.The first heuristic is based on a hypergraph exploration and the second is an evolutionary algorithm. Partial enumeration with dynamic programming is introduced for quality assurance.	ESICUP industrial dataset, UNIBO 2D-Bin Packing problem dataset.	2DGCSPL and hypergraph based heuristic algorithm.	Reduces time and improves quality as the technique gives only proper patterns and strengthens lower bounds obtained by column generation.	Slower column generation due to larger plate size. Improved lower bound of column generation is at the expense of longer solution time.	Average primal dual- gap percentage, computational time.
9. M. Hähnel, J. Martinovic, G. Scheithauer, A. Fischer, A. Schill and W. Dargie, "Extending the Cutting Stock Problem for Consolidating Services with Stochastic Workloads," in IEEE Transactions on Parallel and Distributed Systems, vol. 29, no. 11, pp. 2478-2488, 1 Nov. 2018, doi: 10.1109/TPDS.20182819680.	1. Extend the cutting stock problem to combine workloads. 2. Compare the algorithm with four scheduling algorithms. 3. Demonstrate the correlation between resource consumption and performance.	Previous works consider only the resource requirement and compare the traditional methods with the same. This paper takes into account the job completion time, energy consumed and workload to determine the efficiency of the algorithm.	Services are sorted in decreasing order of resource utilisation by FFD. A subset is created and the first service for the same is assigned. If the resource demands for another service the process of creating a subset is repeated. This process repeats till the resource demand is met.	32000 real world Virtual Machines (VM) from Google data centre.	First Fit Decreasing Algorithm.	The proposed method minimizes CPU utilization and can be extended to IO and MEM operations. Power consumption can also be reduced.	Energy cost of data centre networks is yet to be explored.	Average and Standard Deviation.

10. Tandabani, Asvany, et al. "A Comparative Study of Meta Heuristic Approach for Cutting Stock Problem." *2016 International Conference on Communication and Electronics Systems (ICCES)*, 2016. *Crossref*, doi:10.1109/cesys.2016.7889930.	1.Implement Cuckoo search and particle search optimization algorithms to optimize Cutting stock problem. 2. Compare the computational efficiency and convergence speed of both algorithms 3. Identify the performance factors that give minimum cost.	The Cutting Stock Problem focuses on minimizing cost and maximising stock utilization. However, this has to be done in a way that is computationally efficient and has high convergence speed. Cuckoo Search and Particle Optimization Algorithms are compared to see which one would achieve these two objectives better.	In Cuckoo Search Algorithm, at a time, a cuckoo lays an egg in a randomly chosen nest, and then leaves the nest. The next generation processes the best eggs from the nest. The probability that a present host nest carries a cuckoo egg is pa ∈ [0, 1]. Levy flights and random walks generate new solutions. In Particle Swarm Optimization, a particle moves to find the nearest optimal solution. For every iteration, particle moves and updates current velocity and its position of swarm based search experience and their results found by particle and its neighbours.	Dataset consisting of stock roll width, demand width and number of required width.	Cuckoo Search and Particle Swarm Optimization Algorithms	1. The Cuckoo Search algorithm has a good fitness frequency, and at the same time, fitness can be improved. 2. Particle Swarm Optimization has better cutting stock time and a high utilization rate of stock. It is easy to implement.	1. In Cuckoo Search, the greater number of nests and iterations increase the computational time. 2.Particle Swarm Optimization require high CPU time	Computational speed was in seconds for both Cuckoo Search and Particle Swarm Optimization. Results show a good convergence ratio of the overall stock and a 96.84% average ratio of other stocks utilization.
11.S. Fairee, C. Khompatraporn, B. Sirinaovakul and S. Prom-On, "Trim Loss Optimization in Paper Production Using Reinforcement Artificial Bee Colony," in IEEE Access, vol. 8, pp. 130647-130660, 2020, doi: 10.1109/ACCESS.2020.3008922.	1.Optimize cutting stock problem using a reinforced version of the Artificial Bee Colony algorithm 2.Reduce total trim loss of CSP 3. To minimize leftover stock in inventory by having only the sufficient number of universal intermediate roll sizes.	Jumbo reels in a paper factory are cut into smaller intermediate rolls before they are sheeted. Therefore, it is necessary to calculate the patterns that best fit the width of the reels so trim loss at the two edges are minimized.	A new solution is first sampled and its feasibility is checked through a feasibility matrix,a loss matrix calculates the trim loss and then the optimal cutting patterns at the winder are determined by the linear programming subroutine. The total trim loss is then calculated and the reinforcement vector and solutions are updated. These new solutions are sampled by the Artificial Bee Colony approach which reduces the problem space. All steps are repeated until the condition is satisfied.	The dataset of 1,055 orders of 127 unique sheet sizes was obtained from a paper manufacturer in Thailand.	Artificial Bee Colony algorithm with an ILP subroutine.	The algorithm tends to perform well in high-dimensional problems.	The Artificial Bee Colony algorithm shows a premature convergence late search.The accuracy of optimal value does not meet the requirements sometimes.	Total trim losswas 3.51% as compared to the original 5%, ranking by Friedma's test.
12. .P. Wang, Y. Rao and Q. Luo, "An Effective Discrete Grey Wolf Optimization Algorithm for Solving the Packing Problem," in IEEE Access, vol. 8, pp. 115559-115571, 2020, doi: 10.1109/ACCESS.2020.3004380.	1. Grey Wolf Algorithm is used to solve the 2DSP problem. grey wolf algorithm 2. Redesign searching and attacking operators of the algorithm 3. Propose a novel approach to measure distance between wolves	There have been many meta-heuristic algorithms that optimize CSP. In this paper, we redesign discrete grey wolf optimization to optimize CSP better than the state of the art algorithms.	The Grey Wolf Optimization is modified so that the problem of the prey searching step is not broken off and jumped into the encircling step. Instead, the discrete grey wolf optimization makes sure all the wolves find the prey and increases the probability of avoiding local minima and	Five sets of data from Computational Logistics Organization	Discrete grey wolf optimization	The algorithm is easy to implement and has only 3 parameters. Outperforms most meta-heuristic algorithms.	Large time complexity. Hence, data structure and framework of the algorithm needs to be improved.The Exploration ability of the algorithm is poor.	Computational results, best gap

			simplifies hunting behavior.					
13. Arenales, Marcos Nereu. “A NEW MATHEMATICAL MODEL FOR THE CUTTING STOCK/LEFTOVER PROBLEM.” *Scielo*, Dec. 2015, www.scielo.br/scielo.php?pid=S0101-74382015000300509&script=sci_arttext.	1. Build a mathematical model for CSP that utilizes the leftovers for future demands. 2. The proposed algorithm solves 1D CSP.	Most works consider the leftovers as waste material. This paper presents a method that considers the leftover as materials to satisfy the resource demands.	Three models for standard, reduced and leftover objects are built. Continuous solution is obtained from the column generation method. The proposed method is coded using Optimization Programming Language (OPL).	ESICUP dataset.	Restricted Master Problem.	Leftover size is calculated.	Rrelaxation of the problem is to be solved.	Lengths and demands of items are used to evaluate the performance.
14. Delorme, Maxence, et al. “Bin Packing and Cutting Stock Problems: Mathematical Models and Exact Algorithms.” *European Journal of Operational Research*, vol. 255, no. 1, 2016, pp. 1–20. *Crossref*, doi:10.1016/j.ejor.2016.04.030.	1. Present papers discussing CSP and Bin Packing Problem (BPP) along with the models and proposed algorithms. 2. Evaluate all the algorithms against their respective datasets and a new class of instances.	With so many proposed algorithms for CSP, they all need to be tested against the same dataset and performance metrics in order to actually find a suitable and effective algorithm.	Subjugate all the algorithms based on heuristics, pseudo-polynomial formulation, arc-flow formulation, branch-and-bound, branch-and-price, etc. to the datasets and note down the CPU time taken and list down the algorithms that were solvable in below 10 minutes.	Beasley’s OR library, Scholl’s 3 sets, Wascher’s 2 sets of 17 instances, Schoenfield’s 28 instances.	MTP, BISON, CVRPSEP, VANCE, BELOV, SCIP-BP, ONECUT, DPFLOW, VPSOLVER, BASIC ILP and CSTRPROG algorithms.	All the algorithms are tested against a new dataset with the same performance metric, thus providing us with an actual answer on what the best algorithm is.	Failure to take defective stocks, higher-dimensional CSP with regular and irregular shapes.	Standard deviation, CPU time.
15.S. Octarina, P. B. J. Bangun, S. Pebrina, D. A. Zayanti and L. Hanum, "The N-Sheet Model in Capacitated Multi-Period Cutting Stock Problem with Pattern Set-Up Cost," 2020 3rd International Seminar on Research of Information Technology and Intelligent Systems (ISRITI), Yogyakarta, Indonesia, 2020, pp. 315-320, doi: 10.1109/ISRITI51436.2020.9315440.	1. Use the N-sheet model for solving capacitated multi-period CSP. The goal is minimal pattern formation cost. 2. Make use of the Pattern Generating (PG) algorithm. This considers rectangular and guillotine shaped cuts as well.	The N-sheet model which is used in 2-D CSP has so far has only used single periods and not multi periods. Furthermore, there has been limited studies on capacitated multi-period CSP. Hence, this paper formulates N-sheet models on Capacitated multi period CSP to minimize trim loss.	First, a search tree helps in obtaining feasible cutting patterns Then the PG algorithm produces optimal patterns based on both length and width. Next, those patterns are fed to the N-Sheet model which selects the patterns giving minimal trim loss.	Paper raw material data of a rectangle with a length of 3,000 mm and a width of 3,500 mm with three items.	Pattern Generation Algoritm	N-sheet models yield consistent quality and less wasted materials.	Algorithm results did not give optimal solutions because it yielded too many surplus items.	Trim loss.

IMPLEMENTATION

Code



Welcome   cutting_stock.py ×

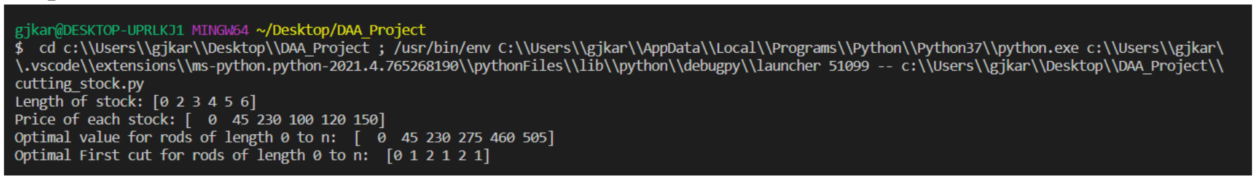
cutting_stock.py > ...

```python
import numpy as np
l = np.array([0, 2, 3, 4, 5,6])
p = np.array([0,45,230,100,120,150])
n = 6
print("Length of stock:",l)
print("Price of each stock:",p)

def ExtendedBottomUpCutStock(p_array, n_array):
    r = np.array([0, 0, 0, 0, 0, 0])    # optimal value for rods of length 0..n
    s = np.array([0, 0, 0, 0, 0, 0])   # optimal first cut for rods of length 0..n
    r[0] = 0
    for j in range(1, n):
        q = 0
        for i in range(1, j+1):  # Find the max cut position for length j
            if (q < p[i] + r[j-i]):
                q = p[i] + r[j-i]
                s[j] = i   # Remember the value of best so far value of i
                r[j] = q
    print("Optimal value for rods of length 0 to n: ", r) #Optimal value for rods of length 0 to n
    i = n-1
    while (i > 0):
        i = i - s[i]
    print("Optimal First cut for rods of length 0 to n: ",s) #Optimal First cut for rods of length 0 to n
    return 1

a = ExtendedBottomUpCutStock(p, n)
```

Output



```
gjkar@DESKTOP-UPRLKJ1 MINGW64 ~/Desktop/DAA_Project
$  cd c:\\Users\\gjkar\\Desktop\\DAA_Project ; /usr/bin/env C:\\Users\\gjkar\\AppData\\Local\\Programs\\Python\\Python37\\python.exe c:\\Users\\gjkar\
\.vscode\\extensions\\ms-python.python-2021.4.765268190\\pythonFiles\\lib\\python\\debugpy\\launcher 51099 -- c:\\Users\\gjkar\\Desktop\\DAA_Project\\
cutting_stock.py
Length of stock: [0 2 3 4 5 6]
Price of each stock: [  0  45 230 100 120 150]
Optimal value for rods of length 0 to n:  [  0  45 230 275 460 505]
Optimal First cut for rods of length 0 to n:  [0 1 2 1 2 1]
```

ARCHITECTURE DIAGRAM

DETAILED ANALYSIS OF PROBLEM & ALGORITHM

PROBLEM

1D Cutting Stock problem is one where a stock has to be cut into standard sizes to minimize the stock wastage and maximize the price. Given n stocks, each stock would have a corresponding length as well as price. To find the best number of cuts to get the maximum profit from said stock, each cut would be of integer length, the number of cuts may be from between 0 to n-1 and there would be no extra charges to cut a stock. We would be required to find the solutions to multiple subproblems in order to find out an optimal solution to the problem.

ALGORITHM

The dynamic programming approach reduces the time complexity from exponential to polynomial. There are 2 approaches to solve the problem using dynamic programming. They are

1. Top-Down Approach - This method is similar to the optimal substructure solution that calls the function recursively.
2. Bottom-Up Approach – Along with the maximum obtainable value, the segment size is also stored in an array. Then we examine the cuts by going backwards till we reach the root node.

Top-Down approach gives us only the maximum stock revenue generated and not the optimal number of cuts. Thus, we come up with a hybridized approach which satisfies both requirements. The function CSPalgo (price_array, stock_len) creates two arrays of size stock_len that store the optimal first value and the optimal first cut for each rod. Then the code runs through a loop that goes from 1 to n. If the sum of the price of that particular piece of rod and its optimum value are greater than the minimum assigned value, then the sum is assigned to the minimum value stored in the variable q. For every iteration, the best value explored so far is stored in the array opcut_array. Then the code prints the optimal value for each rod by exploring the last node first and then moving up the order like the Optimal Substructure problem.

The pseudocode of the aforementioned algorithm is given below:

```
Algorithm CSPalgo (price_array, stock_len)
        Let opval_array and opcut_array be two arrays of size stock_len.
        opval_array will contain the maximum stock revenue for rods of length 0 . . stock_len
        opcut_array will contain the optimal first cut for rods of length 0 . . stock_len
        opval_array (0) ← 0
        for j ←1 to stock_len do
        min_val ← 0
           for i ←1 to j do                              // Find the max cut position for length j
                if min_val < price_array (i) + opval_array (j-i) then
                min_val ← price_array (i) + opval_array (j-i)
              opcut_array (j) ← i                       // Remember the value of best so far value
             of i
                end if-loop
        end for-loop
        opval_array (j) ← min_val
        end for-loop
     print opval_array
         while stock_len > 0 do
           stock_len ← stock_len - opcut_array (n)
         end while-loop
        print opcut_array
return 1
```

RESULTS & DISCUSSION

In the implementation of the code, a collection of n stocks, of varying lengths and prices, need to be analyzed so that the optimal value or maximum cost that can be obtained from the stocks and the optimal first cut of the stock can be determined. In the program, we have a collection of stocks as per the given table:-

Length per stock piece (m)	0	2	3	4	5	6

Price per stock piece (AED)	0	45	230	100	120	150

Here, for n=6 stocks, the length of each stock piece and the price per stock piece is input as numpy arrays l and p. The bottom approach fills up the solution array by figuring out the optimum order. In the *ExtendedBottomUpCutStock()* function, we create the empty numpy arrays r and s, which will hold the optimal value and the optimal first cuts for the rods. For j of range (1,n), we find the maximum cut position for length j as well as remember the best value for i, which is in the range (1, j+1).

The top down while loop is used to calculate the optimal value or the maximum price of the stock. This lop solves each subproblem only once. It solves subproblems for size 0 to n, and the loop iterates n times to solve a subproblem of size n. This gives us the solution of the problem as follows:

Optimal value for rods	0	45	230	275	460	505
Optimal First cut for rods	0	1	2	1	2	1

CONCLUSION

The above dynamic programming algorithm combines top down method and bottom down method to give the optimum value or maximum cost and the optimal first cut for the stock. Both of these methods require Θ(n^2) time. Hence, the algorithm provides an optimal solution to the cutting stock problem in polynomial time.

REFERENCES

1.“Cutting Stock Problem.” *Wikipedia*, 6 Mar. 2021, en.wikipedia.org/wiki/Cutting_stock_problem.

2.“The Cutting Stock Problem | NEOS.” *Winconsin*, 2020, neos-guide.org/content/cutting-stock-problem.

3. Robert W. Haessler. “Cutting Stock Problem and Solution Procedures.” *Cutting Stock Problem and Solution Procedures*, 1991, deepblue.lib.umich.edu/bitstream/handle/2027.42/29128/0000167.pdf.

4. GeeksforGeeks. “Cutting a Rod | DP-13.” *GeeksforGeeks*, 20 Apr. 2021, www.geeksforgeeks.org/cutting-a-rod-dp-13