Enhancing Financial Management: Algorithm Strategy Approach to Capital Budgeting Optimization

Felix Chandra - 13523012

Program Studi Teknik Informatika

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia

E-mail: mainaccfelixchandra@gmail.com , 13523012@std.stei.itb.ac.id

***Abstract*— Capital budgeting, a critical pillar of financial management, dictates the allocation of scarce resources to long-term investment projects to maximize firm value. Selecting the optimal project portfolio under strict budgetary constraints presents a significant combinatorial challenge that can overwhelm traditional evaluation methods. This paper explores the application of algorithm strategies to provide a more systematic and optimized approach to these crucial decisions.**

The study investigates two distinct algorithmic models for two different investment scenarios. First, the paper analyzes the Greedy algorithm as a solution for the Fractional Knapsack Problem, applicable when projects are divisible. This approach, which prioritizes investments by their profitability-to-cost ratio is proven to yield the optimal solution for this specific context. Second, it examines the Backtracking algorithm for the 0/1 Knapsack Problem, where projects are indivisible. This method systematically explores all valid project combinations to guarantee the identification of the portfolio yielding the absolute maximum profit.

By analyzing these methods, this paper demonstrates how different algorithmic strategies are suited for different financial constraints, contrasting the efficient, optimal approach of the Greedy algorithm for divisible problems with the comprehensive search of Backtracking for indivisible ones. The research concludes that these frameworks provide a robust, data-driven methodology that enhances capital budgeting.

***Keywords*— Capital Budgeting, Financial Management, Algorithm Strategy, Optimization, Greedy, Fractional Knapsack Problem, Backtracking, 0/1 Knapsack Problem.**

I. Introduction

As a computer science undergraduate pursuing a minor in business management, the author's studies in financial management have provided a unique interdisciplinary view to examine one of its most important functions, capital budgeting. This process, by which a firm strategically evaluates and selects its long-term investments, serves as a fundamental of corporate prosperity. Such decisions are consequential, dictating the allocation of substantial capital to assets with multi-year lifespans and thereby fundamentally shaping the firm's future profitability. The primary objective is to undertake ventures that increase the firm value, typically identified as projects whose discounted future cash inflows (PV) surpass the initial outlay.

This theoretical pursuit of value maximization however is constrained by the reality of resource scarcity. Firms operate within finite capital budgets, frequently encountering more viable investment opportunities than they can feasibly fund. This issue transforms capital budgeting from a straightforward evaluation of individual projects into a complex combinatorial optimization problem. The essential question thus becomes "given a fixed budget and a set of prospective projects each with a distinct cost and expected return, which specific portfolio will yield the maximum aggregate return?" A brute-force assessment of every permutation is computationally intractable as the number of projects expands.

It is at this point where finance and computation meet that the principles of algorithm design present a potent and systematic resolution. Algorithmic strategies offer a rigorous methodology for navigating intricate decision spaces to find optimal or near-optimal outcomes efficiently. By modeling the capital budgeting challenge as a well defined computational problem, financial managers can transcend intuition-based selections and leverage algorithms for a data-driven, objective, and ultimately optimized approach.

Accordingly, this paper will analyze two fundamental algorithmic paradigms applied to capital budgeting. First, it will analyze the Greedy algorithm as a robust solution to the Fractional Knapsack Problem, relevant to contexts where partial project funding is permissible. Second, it will analyze the Backtracking algorithm within the framework of the 0/1 Knapsack Problem, a model that accurately mirrors investment scenarios characterized by indivisible projects. Through this examination, the research demonstrates how algorithm strategy thinking can be seamlessly integrated into

financial strategy to elevate the outcome and efficiency of capital allocation decisions.

II. Literature Review

A. Time Value of Money

The Time Value of Money is the foundational axiom of finance, establishing that a sum of money available at the present time is worth more than the identical sum in the future. This preference for present over future cash is not merely psychological. It is based on sound economic reasoning and is quantified through the use of interest rates. A dollar in hand today is more valuable than a dollar promised a year from now because of its potential earning capacity. This core principle underpins nearly every aspect of financial valuation and corporate finance, including the capital budgeting techniques central to this paper.

There are three primary reasons for the time value of money:

1) **Opportunity Cost**: Money received today can be invested to earn a return. By forgoing the money now, one also forgoes the potential earnings (interest) that could have been generated during the waiting period. This lost earning potential is a critical opportunity cost.
2) **Inflation**: In most economies, the general price level of goods and services tends to rise over time, a phenomenon known as inflation. Consequently, the purchasing power of money diminishes. A dollar today can purchase more than a dollar will be able to in the future, making present cash inherently more valuable.
3) **Risk and Uncertainty**: A promise of future payment carries inherent risk. The entity promising the payment could default, or unforeseen circumstances could prevent the payment from materializing. Possessing the cash now eliminates this uncertainty. Therefore, a premium, in the form of interest, is required to compensate for bearing this risk over time.

To put the concept of time value of money into operation, finance uses two primary calculations: compounding to find future value and discounting to find present value.

While future value calculates the growth of money forward in time, Present Value (PV) does the inverse and more critical task for investment decisions: it determines the current worth of a future sum of money or stream of cash flows, given a specified rate of return. This process is known as discounting.

Discounting is the cornerstone of capital budgeting. When a firm evaluates a project, it anticipates receiving cash inflows many years into the future. To make a rational decision today, the firm must ascertain what those future earnings are worth in today's terms. As Gitman (2003) emphasizes, this is the only way to compare the initial investment (a current cash outflow) with the project's anticipated returns (future cash inflows) on a like for like basis.

The present value of a single future cash flow is calculated using the following formula:

$$PV = \frac{FV}{(1+r)^n}$$

Figure 1. Present Value Formula
(https://www.inchcalculator.com/future-value-calculator/)

Where:

- **PV = Present Value**: The value of the future cash flow in today's dollars. This is the amount you would need to invest today at the specified interest rate to have the future value amount at the end of the period.
- **FV = Future Value**: The amount of cash that is expected to be received at a future date.
- **r = Discount Rate or Rate of Return**: This is the interest rate used to discount the future cash flows. In capital budgeting, this rate is typically the firm's cost of capital (e.g., WACC), which represents the return required by investors to compensate them for the risk of the project. It is the opportunity cost of the investment.
- **n = Number of Periods**: The number of years or other time periods until the future cash flow is received.

The denominator of the formula, $(1+r)^n$, is the Present Value Interest Factor. This factor grows exponentially as either the discount rate (r) or the number of periods (n) increases. This leads to two critical inverse relationships:

- Higher Discount Rate, Lower Present Value: As the discount rate (r) increases, the present value of a future cash flow decreases. This is logical: if the opportunity cost of capital is higher, or the investment is riskier (requiring a higher return), then a future payment is worth significantly less today.
- Longer Time Horizon, Lower Present Value: As the number of periods (n) increases, the present value also decreases. A payment promised 10 years from now is worth far less today than the same payment promised in 3 years, due to the extended period of forgone interest and increased uncertainty.

By applying this formula, we can translate any future cash flow into its equivalent value today. This is the essential first step in calculating Net Present Value (NPV), where the

present value of all future inflows is summed and then compared against the initial investment. Without the concept of Present Value, a meaningful evaluation of long-term investments would be impossible.

B. Capital Budgeting

Capital budgeting is the process a firm undertakes to evaluate potential major projects or long-term investments. It is a cornerstone of corporate financial management, concerned with the allocation of a company's scarce financial resources to ventures that are expected to generate returns over a multi-year period. These decisions are of paramount strategic importance because they often involve substantial capital outlays, are difficult and costly to reverse, and fundamentally dictate a firm's future operational capacity, market position, and profitability. Examples of capital budgeting decisions include constructing a new factory, purchasing new machinery, launching a new product line, or entering a new market. As articulated by Gitman (2003) in "Principles of Managerial Finance," the primary goal of these decisions is to select investments that maximize the value of the firm, and by extension, the wealth of its shareholders. A well-executed capital budgeting process ensures that capital is deployed efficiently, funding projects that promise returns greater than the cost of the capital used to finance them.

A systematic capital budgeting process typically involves five distinct stages:

1) **Proposal Generation**: Investment ideas are generated at all levels of an organization, from employees on the factory floor to senior executives. These proposals align with the firm's strategic objectives, such as expansion, replacement of assets, or compliance with regulations.
2) **Review and Analysis**: This is the most critical stage. The financial viability of each proposal is meticulously analyzed. This involves estimating the project's cash flows both inflows and outflows and assessing the associated risks.
3) **Decision Making**: Based on the financial analysis, a decision is made. Firms use various evaluation techniques, which will be discussed below, to determine whether to accept or reject a project. For smaller investments, authority may rest with departmental managers, while major strategic projects typically require board approval.
4) **Implementation**: Once a project is approved, funds are appropriated, and the project is implemented. This phase involves managing project timelines, costs, and execution to ensure the planned objectives are met.
5) **Post Audit**: After the project has been operational for some time, its performance is compared against the initial projections. This post-audit is crucial for continuous improvement; it helps identify systematic biases in forecasting, improves future investment decisions, and determines if a project should be continued or abandoned.

The entire capital budgeting framework rests on two foundational elements: estimating relevant cash flows and determining the appropriate discount rate.

Relevant Cash Flows: The focus is exclusively on incremental cash flows, the change in the firm's total future cash flows that results directly from undertaking the project. This includes:

1) **Initial Investment**: The total cash outflow required to initiate the project, including the purchase price of assets, installation costs, and any required increase in net working capital.
2) **Operating Cash Inflows**: The incremental after-tax cash flows the project is expected to generate over its life. This must account for revenues, expenses, and the tax effects of depreciation.
3) **Terminal Cash Flow**: The net after-tax cash flow received at the conclusion of a project, typically from the sale of the asset and the recovery of net working capital. It is critical to exclude sunk costs (expenses already incurred that cannot be recovered) and include opportunity costs (cash flows a firm foregoes by accepting one project over another).

Cost of Capital: Since project returns are received in the future, they must be discounted to their present value to be comparable with the initial investment. The appropriate discount rate is the firm's cost of capital, often calculated as the Weighted Average Cost of Capital (WACC). The WACC represents the minimum rate of return a project must earn to cover the cost of the funds used to finance it. It reflects the riskiness of the firm's overall operations. For projects with significantly different risk profiles, a risk-adjusted discount rate may be used.

C. Capital Budgeting Evaluation Technique

Firms employ several techniques to evaluate investment proposals, which can be broadly classified into two categories.

1) **Non-Discounted Cash Flow Techniques (Traditional Methods)**

Payback Period: This is the length of time required for a project's cumulative cash inflows to equal its initial investment. While simple to calculate and intuitive, it is widely criticized. As Gitman (2003) points out, its primary weaknesses are that it (1) ignores the time value of money and (2) completely disregards any cash flows generated after the payback period, thereby failing to measure true profitability.

This method does not explicitly account for the time value of money.

2) Discounted Cash Flow (DCF) Techniques

These methods are theoretically superior as they incorporate the time value of money and risk into the analysis.

- **Net Present Value (NPV)**: The NPV is calculated by subtracting the project's initial investment from the present value of its future operating cash inflows, discounted at the firm's cost of capital.

  Decision Rule: Accept projects with an NPV greater than or equal to zero

  Interpretation: A positive NPV signifies that the project is expected to generate returns in excess of its financing costs, thereby creating value for the firm and increasing shareholder wealth. NPV is widely considered the most robust capital budgeting technique.

- **Internal Rate of Return (IRR)**: The IRR is the discount rate that equates the present value of a project's future cash inflows with its initial investment, effectively making the NPV equal to zero.

  Decision Rule: Accept projects where the IRR is greater than the cost of capital.

  Interpretation: The IRR represents the project's expected percentage rate of return. While popular due to its intuitive, percentage-based nature, it can be problematic when comparing mutually exclusive projects of different scales or cash flow patterns, sometimes leading to rankings that conflict with the NPV rule.

- **Profitability Index (PI)**: The PI is the ratio of the present value of future cash flows to the initial investment.

  Decision Rule: Accept projects with a PI greater than 1.0.

  Interpretation: A PI of 1.5, for example, means that for every rupiah invested, the project is expected to generate Rp1.50,- in present value. The PI is particularly useful for ranking projects under conditions of capital rationing.

D. Greedy Algorithm

The Greedy algorithm is a simple yet powerful algorithmic paradigm for solving optimization problems, which are problems that seek to find a maximum or minimum value under a set of constraints. Based on the explanation of Munir (2025), the defining principle of the Greedy method is to make the choice that appears to be the best at each step. It constructs a solution incrementally, and at each stage, it makes a locally optimal choice with the hope that this sequence of "take what you can get now" decisions will lead to a globally optimal solution. Any problem solvable with a Greedy approach can be broken down into several key elements:

1) **A Set of Candidates (C):** The pool of items from which a solution is built. This could be a set of coins, investment projects, activities, or edges in a graph.
2) **A Set of Solutions (S):** The set containing the candidates that have been chosen. The algorithm starts with an empty set and incrementally adds candidates to it.
3) **A Selection Function:** The core of the greedy strategy. This function selects the most promising candidate from the remaining pool at each step (e.g., the coin with the highest value, the activity that finishes earliest).
4) **A Feasibility Function:** This function determines if a candidate can be added to the solution set without violating the problem's constraints (e.g., checking if adding an object's weight exceeds the knapsack's capacity).
5) **A Solution Function:** A function that determines if the set S constitutes a complete solution to the problem.
6) **An Objective Function:** The function that the algorithm aims to maximize or minimize.

The general process involves iteratively selecting the best candidate, checking if it's feasible to add to the solution, and continuing until a complete solution is formed.

A critical aspect of the Greedy algorithm is that it does not always produce a globally optimal solution. The locally optimal choices it makes are irrevocable. Where the algorithm never backtracks to reconsider a decision. Munir (2025) demonstrates this with the Coin Exchange Problem. While the greedy strategy of picking the largest denomination of coin works for certain currency systems like USD, EUR, and IDR, it fails for others. For instance, given coins of values {10, 7, 1}, making change for 15 with a greedy approach yields {10, 1, 1, 1, 1, 1} (six coins), whereas the optimal solution is {7, 7, 1} (three coins). This shows that while Greedy algorithms can provide a fast approximation, their optimality must be mathematically proven for each specific problem.

Applying Greedy Algorithm to The Knapsack Problem

The Knapsack Problem is a classic optimization problem in computer science. As described in the lecture materials by Munir (2025), the general scenario is as follows: given a set of items, each with a weight and a value or profit, determine the number of each item to include in a collection so that the total weight is less than or equal to a given limit and the total value is as large as possible. This problem serves as an excellent model for resource allocation scenarios where one must select

from a set of choices to maximize benefit under a fixed budget or constraint. The key to solving it depends entirely on a critical distinction: whether the items are divisible or not. This leads to two major variants.

1) The 0/1 Knapsack Problem (Indivisible Items)

In this version of the problem, each item is indivisible; you must either take the entire item (1) or leave it behind (0). You cannot take a fraction of an item. The most intuitive greedy strategy is Greedy by Density. This involves prioritizing items with the highest value-to-weight ratio (p_i/w_i). However, for the 0/1 problem, this strategy is not guaranteed to be optimal.

Example:

Knapsack Capacity (K): 50 kg

Item	Weight (w_i)	Profit (p_i)	Density (p_i/w_i)
A	10 kg	$60	$6 / kg
B	20 kg	$100	$5 / kg
C	30 kg	$120	$4 / kg

Greedy by Density Walkthrough:

1) Select Item A (highest density). Remaining Capacity: 50 - 10 = 40 kg. Current Profit: $60.
2) Select Item B (next highest density). Remaining Capacity: 40 - 20 = 20 kg. Current Profit: $60 + $100 = $160 .
3) Evaluate Item C. Its weight (30 kg) exceeds the remaining capacity (20 kg), so it cannot be chosen.

The algorithm terminates, resulting in a total profit of $160. However, the true optimal solution for this problem is to select Items B and C, for a total weight of 50 kg and a total profit of $220. It fails because the greedy choice to take Item A, while locally optimal, was irrevocable. This choice "blocked" the ability to select the more valuable combination of Items B and C, which perfectly utilized the knapsack's capacity. For the 0/1 problem, the empty space left by a greedy choice can be too small for other valuable items, leading to a suboptimal result.

2) The Fractional Knapsack Problem (Divisible Items)

In this variant, you are allowed to take any fraction of an item. This changes the problem fundamentally and allows the greedy strategy to succeed.

Greedy by Density Walkthrough (using the same example):

1) **Select Item A** (highest density). Take all of it. Remaining Capacity: 40 kg. Current Profit: $60.
2) **Select Item B** (next highest density). Take all of it. Remaining Capacity: 20 kg. Current Profit: $60 + $100 = $160.
3) **Select Item C** (next highest density). The remaining capacity is 20 kg, but the item weighs 30 kg. We can take a fraction. Fraction to take: 20 kg / 30 kg = **2/3** of Item C. Profit from this fraction: (2/3) * $120 = $80. The knapsack is now full.

The algorithm terminates, resulting in a total profit of $160 + $80 = $240. As stated in Theorem 2 from Munir's lecture (2025), this method is provably optimal for the fractional case. Because we can take fractions, there is no wasted space or blocked opportunity. The greedy choice ensures that every single unit of capacity, from the first to the last, is filled with the item that provides the absolute highest profit-per-unit-of-weight available at that moment. This guarantees a maximum total profit.

E. Backtracking Algorithm

Since the Greedy method can fail for the 0/1 Knapsack problem, a more robust technique is required to guarantee an optimal solution. The Backtracking algorithm provides such a guarantee. As detailed by Munir (2025), backtracking is a refined version of an exhaustive search that systematically explores all potential solutions and discards large subsets of fruitless candidates by using a bounding function.

The algorithm works by building a state-space tree, which represents all possible choices. For the 0/1 Knapsack problem with n items, this is a binary tree of depth n. At each level i of the tree, the algorithm makes a decision for item i: The left branch represents including the item ($x_i = 1$). The right branch represents excluding the item ($x_i = 0$). A path from the root to any leaf node represents a complete potential solution. A brute-force search would explore all 2^npaths. Backtracking improves on this significantly. As it traverses the tree (typically using a Depth-First Search), it checks at each node whether the current path is still promising.

The key to backtracking's efficiency is its bounding function. For the 0/1 Knapsack problem, the primary constraint is the knapsack's capacity K.

1) **The Bounding Condition**: The algorithm checks if the cumulative weight of the items included so far exceeds K.
2) **Pruning**: If at any point the cumulative weight is greater than K, that node becomes a dead node. The algorithm abandons this path and backtracks to a previous decision point, effectively pruning the entire subtree below the dead node. This is because any further additions along that path will also exceed the capacity.

Applying Backtracking to 0/1 Knapsack Problem:

The algorithm keeps track of the best profit found so far with maximum profit starting from $0.

1. Start at the root. Explore the path where Item A is included (Weight=10, Profit=$60). This is valid.
2. From there, explore where Item B is included (Total Weight=30, Total Profit=$160). This is valid.
3. From there, try including Item C (Total Weight=60). This exceeds the capacity. Prune this path and backtrack.
4. Now, try excluding Item C. The solution {A, B} is valid (Weight=30, Profit=$160). We update maximum profit to $160 and backtrack.
5. Now, explore excluding Item B but keeping Item A (Weight=10, Profit=$60). From here, include Item C (Total Weight=40, Total Profit=$180). This is a valid solution. Since $180 is greater than the current maximum profit, we update maximum profit to $180 and backtrack.
6. The algorithm continues to backtrack to the root and explores the path where Item A is excluded.
7. From there, it includes Item B (Weight=20, Profit=$100). This is valid.
8. From there, it includes Item C (Total Weight=50, Total Profit=$220). This is a valid solution. Since $220 is greater than current maximum profit, we update maximum profit to $220.

The algorithm continues exploring all other valid branches. By systematically exploring only the valid paths and pruning invalid ones, backtracking eventually confirms that $220 is the highest possible profit, correctly identifying {Item B, Item C} as the optimal solution. This proves its superiority over the greedy method for the 0/1 problem.

III. Implementation of Algorithm Strategy in Financial Management : Capital Budgeting

The core challenge of capital budgeting under capital rationing is one of constrained optimization. A firm possesses a set of valuable investment opportunities but is limited by a finite capital budget. The objective is to select a portfolio of these projects that maximizes the total value, typically measured by the sum of the projects' Net Present Values (NPVs), without exceeding the budgetary constraint.

As established in the literature review, this financial dilemma maps perfectly to the classic Knapsack Problem. To apply algorithmic solutions, we must first translate the financial terminology into the components of a Greedy algorithm, as defined by Munir (2025):

1) **Candidate Set (C):** The set of all available, independent investment proposals that the firm is considering.
2) **Knapsack Capacity (K):** The firm's total capital budget for the period.
3) **Item Weight (w_i):** The initial investment or cost required for project i.
4) **Item Profit (p_i):** The expected value generated by project i, which is its Net Present Value (NPV).
5) **Objective Function:** To maximize the total NPV of the selected projects.

The critical factor that determines the effectiveness and optimality of the Greedy algorithm is the nature of the projects themselves, specifically, whether they are divisible or indivisible. This distinction directly corresponds to the Fractional Knapsack and 0/1 Knapsack problems, respectively.

A. Scenario 1 : Divisible Projects - P2P Lending for Indonesian SMEs

Let's consider a practical scenario involving a Jakarta-based financial technology firm, "PT. Dana Indonesia". This firm specializes in connecting investors with Indonesian Small and Medium Enterprises (UMKM) through a Peer-to-Peer (P2P) lending platform. For the upcoming quarter, the firm has a capital pool of Rp 2.000.000.000,- (two billion Rupiah) to allocate across various loan proposals from promising UMKM.

P2P lending is an ideal real-world example of the Fractional Knapsack Problem because the investments are inherently divisible. PT. Dana Indonesia can choose to fully fund a loan or fund only a fraction of the total amount requested by an UMKM, with other investors on the platform funding the rest.

Valuating the Profit (NPV) of Each Loan

Before applying the Greedy algorithm, the firm's analysts must determine the "profit" of each loan opportunity. This is not just the total interest collected; it is the Net Present Value (NPV) of the future cash flows from the loan. The process is as follows:

1. Projecting Cash Flows: For each loan, analysts project the stream of principal and interest payments the UMKM is expected to make over the loan's term (e.g., monthly payments for 24 months).
2. Determining the Discount Rate: A risk-adjusted discount rate is assigned to each UMKM. This rate reflects the firm's required rate of return and the perceived risk of the borrower. A loan to an established coffee shop might have a lower discount rate than a loan to a brand-new craft business.
3. Calculating Present Value: Each projected cash inflow is discounted back to its present value using the formula $PV = FV/ (1+r)^n$.
4. Calculating NPV: The NPV is the sum of all the present values of the cash inflows minus the initial

investment / the loan principal. A positive NPV indicates the investment is expected to generate returns above the required rate, thus adding value to the firm.

Implementation Walkthrough

After analysis, PT. Dana Indonesia has the following five loan opportunities. The NPV for each has been pre-calculated by their analysts.

UMKM	Loan Required (w_i) (Rp 000.000)	Calculated NPV (p_i) (Rp 000.000)	Profitability Index (p_i+w_i) / w_i	Rank
Toko Kopi Bahagia	600	210	1.35	4
Bengkel Jaya Motor	400	220	1.55	1
Garmen Maju Bersama	800	360	1.45	2
Kerajinan Rotan	300	105	1.35	4
Warung Nasi Ibu Susi	500	200	1.40	3

The Greedy algorithm is applied to allocate the Rp 2.000.000.000 budget

Step 1 (Rank by PI): The projects are ranked by their PI: Bengkel Jaya Motor, Garmen Maju Bersama, Warung Nasi Ibu Susi, Toko Kopi Bahagia, Kerajinan Rotan.

Step 2 (Select Bengkel Jaya Motor): The top-ranked loan is fully funded.
- Cost: Rp 400.000.000
- Remaining Budget: Rp 1.600.000.000
- Cumulative NPV: Rp 220.000.000

Step 3 (Select Garmen Maju Bersama): The next loan is fully funded.
- Cost: Rp 800.000.000
- Remaining Budget: Rp 1.600.000.000 - Rp 800.000.000 = Rp 800.000.000
- Cumulative NPV: Rp 220.000.000 + Rp 360.000.000 = Rp 580.000.000

Step 4 (Select Warung Nasi Ibu Susi): The next loan is fully funded.
- Cost: Rp 500.000.000
- Remaining Budget: Rp 800.000.000 - Rp 500.000.000 = Rp 300.000.000
- Cumulative NPV: Rp 580.000.000 + Rp 200.000.000 = Rp 780.000.000

Step 5 (Select Fraction of Toko Kopi Bahagia): The next loan opportunity is Toko Kopi Bahagia, which requires Rp 600.000.000. However, only Rp 300.000.000 remains in the budget. Since the loan is divisible, we fund a fraction.

- Fraction to fund: Rp 300.000.000 / Rp 600.000.000 = 0.5 or 50%.
- Cost: Rp 300.000.000
- Remaining Budget: Rp 300.000.000 - Rp 300.000.000 = Rp 0
- NPV from fraction: 0.5 * Rp 210.000.000 = Rp 105.000.000
- Final Cumulative NPV: Rp 780.000.000 + Rp 105.000.000 = Rp 885.000.000

The budget is now exhausted. The optimal investment portfolio is {Bengkel Jaya Motor, Garmen Maju Bersama, Warung Nasi Ibu Susi, and 50% of the loan for Toko Kopi Bahagia}, yielding the maximum possible NPV of Rp 885.000.000.

B. Scenario 2 : Indivisible Projects - Manufacturing Expansion

Now let's examine a scenario that maps to the 0/1 Knapsack Problem. Consider "PT. Furnitur Indonesia," a successful furniture export company based in Bandung. To meet rising international demand and improve efficiency, the management plans to upgrade its production facility. The company has allocated a capital budget of Rp 10.000.000.000,- (Ten Billion Rupiah) for this expansion.

The proposed investments are for large, specialized machines. Each machine is a single, all-or-nothing purchase; the company cannot buy half a CNC machine or 30% of a finishing line. This makes the projects strictly indivisible.

The finance department has already calculated the NPV for each potential project, factoring in the initial purchase and installation cost, future labor savings, increased production capacity, and maintenance costs, all discounted by the company's WACC.

Machine	Cost (w_i) (Rp 000.000.000)	NPV (p_i) (Rp 000.000.000)
A: CNC Wood Carver	5	6
B: Automated Finishing Line	4	5
C: Industrial Dust Collector	3	3

While a simple Greedy heuristic might be tempting, it does not guarantee finding the most profitable combination. To ensure the optimal allocation of their Rp 10 Billion budget, the company must use a more exhaustive method like Backtracking.

As detailed by Munir (2025), the Backtracking algorithm works by systematically exploring a state-space tree of all possible solutions. For this 0/1 problem, the tree represents the decision to either include or exclude each of the three machines. The algorithm's power comes from its bounding function, which prunes any branch of the tree that is no longer promising. Here, the bounding function is simple: if the cumulative cost of the projects selected along a path exceeds the Rp 10 Billion budget, that path is abandoned.

Implementation Walkthrough

The algorithm explores the decision tree, keeping track of the best combination found so far (max_profit, initially Rp 0).

Step 1 (Explore path {A}) :

- Include A (Cost: 5 Billion, Profit: 6 Billion). Path is valid. Continue.

Step 2 (Explore path {A, B}) :

- Include B (Total Cost: 9 Billion, Profit: 11 Billion). Path is valid. Continue.

Step 3 Explore path {A, B, C}) :

- Try including C. Total Cost becomes 12 Billion. This is over budget.
- Prune this path. Backtrack.

Step 4 Finalize path {A, B}):

- Since C was pruned, {A, B} is a complete solution. Profit = 11M.
- Update max_profit to 11 billion. Backtrack.

Step 5 (Explore path {A, C}) :

- This path results from excluding B. Total Cost = 8 Billion, Profit = 9 Billion.
- This is a valid solution, but its profit (9 Billion) is less than max_profit (11 Billion). No update. Backtrack.

Step 6 (Explore path {B, C}) :

- Backtrack to the root and explore excluding A. Include B (Cost: 4 Billion) and C (Cost: 3 Billion).
- Total Cost = 7 Billion, Profit = 8 Billion.
- Valid, but less than max_profit. No update.

After systematically exploring all other valid combinations ({A only}, {B only}, {C only}, etc.) and finding none with a profit greater than 11 billion rupiah, the algorithm terminates. It has confirmed that the optimal solution is to invest in Project A and Project B, for a total cost of Rp 9.000.000.000,- and a maximum NPV of Rp 11.000.000.000 ,- .

IV. Summary and Conclusion

This paper has explored the application of computational algorithm strategies to enhance one of the most critical functions of corporate finance: capital budgeting. The central challenge addressed is the optimization of investment decisions under capital rationing, a scenario where a firm has more value-adding projects than its budget can support. Traditional financial metrics like Net Present Value (NPV), while essential for evaluating individual projects, fall short in identifying the optimal portfolio of projects from a combinatorial perspective.

To address this, the capital budgeting problem was framed as the classic Knapsack Problem from computer science. This powerful analogy allows us to model the firm's budget as the knapsack's capacity and the investment projects as items to be selected based on their cost (weight) and NPV (profit). The analysis focused on two distinct scenarios, each requiring a different algorithmic approach.

The first scenario involved divisible projects, analogous to the Fractional Knapsack Problem. This was illustrated with a real-world case of a P2P lending firm in Indonesia allocating its capital pool across various SME loan proposals. For this type of problem, the Greedy Algorithm, which makes locally optimal choices at each step, was shown to be a provably optimal strategy. By prioritizing investments based on their Profitability Index (NPV divided by cost), the Greedy method efficiently allocates every unit of the budget to the most value-dense opportunities available, guaranteeing the maximum possible total NPV for the portfolio.

The second scenario focused on indivisible projects, which correspond to the 0/1 Knapsack Problem. This was demonstrated through the case of an Indonesian furniture

manufacturer selecting large, all-or-nothing machinery for a factory upgrade. It was shown that a simple Greedy heuristic can fail in this context, as its irrevocable, short-sighted choices can block more profitable combinations. For this challenge, the Backtracking Algorithm was presented as the definitive solution. By systematically building a state-space tree of all possible combinations and using a bounding function to prune paths that violate the budget constraint, Backtracking performs an intelligent, exhaustive search. While more computationally intensive than the Greedy method, it is guaranteed to find the true, globally optimal portfolio of indivisible projects.

In conclusion, this paper demonstrates that a purely financial approach to capital budgeting can be significantly enhanced by integrating the structured problem-solving frameworks of computer science. By correctly identifying whether an investment problem is divisible (Fractional Knapsack) or indivisible (0/1 Knapsack), a financial manager can select the appropriate algorithm, be it the fast and efficient Greedy method or the robust and exhaustive Backtracking algorithm, to move beyond simple project ranking and toward a truly optimized, value-maximizing allocation of capital. This interdisciplinary approach provides a more rigorous, objective, and powerful methodology for modern financial management.

Explanation Video

Link to author's youtube video explanation on this research paper: https://youtu.be/zqwZZZ-itxY

Acknowledgement

The author would like to express sincere gratitude to Almighty God, whose grace and guidance enabled the completion of this paper. The author also wishes to thank the lecturers of the Algorithm Strategy course (IF2211), especially Dr. Nur Ulfa Maulidevi, S.T, M.Sc., the lecturer of the author's class, for her guidance and teaching, which greatly supported the development of this work. Deep appreciation is extended to the author's parents for their unwavering financial and emotional support, without which this paper could not have been completed. Finally, the author is grateful to the readers for taking the time to read this paper.

References

[1] Gitman, L. J. (2003). Principles of managerial finance (10th ed.). Addison-Wesley.

[2] L. J. Gitman and C. J. Zutter, Principles of Managerial Finance, 14th ed. Boston, MA: Pearson, 2015. [Slide presentation, School of Business and Management, ITB, Chapter 5: Time Value of Money].

[3] L. J. Gitman and C. J. Zutter, Principles of Managerial Finance, 14th ed. Boston, MA: Pearson, 2015. [Slide presentation, School of Business and Management, ITB, Chapter 10: Capital Budgeting Techniques].

[4] L. J. Gitman and C. J. Zutter, Principles of Managerial Finance, 14th ed. Boston, MA: Pearson, 2015. [Slide presentation, School of Business and Management, ITB, Chapter 11: Capital Budgeting Cash Flows].

[5] https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2024-2025/04-Algoritma-Greedy-(2025)-Bag1.pdf, accessed on June 18th 2025 13 o'clock.

[6] Ellis Horrowitz & Sartaj Sahni, Computer Algorithms, 1998[https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2024-2025/05-Algoritma-Greedy-(2025)-Bag2.pdf].

[7] Roger Crawfis, CSE 680 Introduction to Algorithms: GreedyAlgorithms[https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2024-2025/06-Algoritma-Greedy-(2025)-Bag3.pdf].

[8] https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2024-2025/15-Algoritma-backtracking-(2025)-Bagian1.pdf , accessed on June 18th 2025 14 o'clock.

[9] https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2024-2025/16-Algoritma-backtracking-(2025)-Bagian2.pdf , accessed on June 18th 2025 14 o'clock.

Statement

I hereby declare that this paper I've written is the result of my own independent work, It is neither a translation of another person's work nor derived from any form of plagiarism.

Bandung, June 22nd 2025

Felix Chandra (13523012)