Optimasi Alokasi Portofolio Saham LQ45 Menggunakan Dynamic Programming dengan Batasan Modal dan Risiko

Juan Oloando Simanungkalit - 13524032
Program Studi Teknik Informatika
Sekolah Teknik Elektro Dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13524032@std.stei.itb.ac.id, juanoloando.s@gmail.com

***Abstrak*—Makalah ini menganalisis penerapan algoritma *Dynamic Programming* (DP) untuk menyelesaikan persoalan optimasi kombinatorial dalam alokasi portofolio saham indeks LQ45. Fluktuasi harga pasar membuat investor untuk melakukan diversifikasi strategis untuk memitigasi risiko tanpa mengorbankan ekspektasi keuntungan. Dengan model *2D Bounded Knapsack Problem*, algoritma DP digunakan untuk memaksimalkan utilitas investasi berdasarkan alokasi lot saham di bawah dua batasan utama, yaitu batas modal Rp 100.000.000 dan tiga tingkat toleransi volatilitas, yakni Konservatif ($\leq 45\%$), Moderat ($\leq 65\%$), dan Agresif ($\leq 90\%$). Hasil program dengan algoritma DP membuktikan bahwa algoritma DP secara sistematis lebih unggul dibandingkan pendekatan *equal-weight benchmark* dengan menghasilkan tingkat *return* yang jauh lebih tinggi dan efisien. DP berhasil mengidentifikasi kombinasi emiten optimal berdasarkan *Sharpe Ratio*, membuktikan bahwa algoritma ini sangat efisien dan kokoh sebagai sistem pendukung keputusan finansial yang adaptif terhadap toleransi risiko investor.**

***Kata Kunci*: Optimasi Portofolio, *Dynamic Programming*, *Bounded Knapsack Problem*, Saham LQ45, Manajemen Risiko**

I. PENDAHULUAN

Pada era masa kini, nilai mata uang semakin tidak menentu. Di Indonesia, nilai Rupiah semakin melemah terhadap US Dollar. 10 tahun lalu, uang 10 ribu rupiah dapat dipakai untuk membeli satu porsi mie ayam, sekarang harga satu porsi mie ayam sekitar 15 sampai 18 ribu per porsi. Hal ini terjadi karena inflasi. Salah satu upaya untuk menjaga harta kita tidak dimakan waktu adalah dengan investasi. Investasi merupakan salah satu cara untuk mempertahankan atau menambahkan kekayaan tanpa harus bekerja lebih keras atau masyarakat menyebutnya dengan *passive income*. Salah satu instrumen dalam investasi adalah saham. Investasi saham adalah penanaman modal dalam bentuk penyertaan sejumlah dana oleh seseorang atau badan usaha yang mana melalui instrumen tersebut investor memiliki klaim atas aset dan penghasilan perusahaan serta berhak menghadiri Rapat Umum Pemegang Saham (RUPS). Dalam membeli saham, investor berharap untuk mendapatkan keuntungan dari kenaikan harga saham atau dari dividen yang dibagikan perusahaan. Namun, investasi saham juga memiliki risiko yang tinggi karena harga saham dapat mengalami fluktuasi/ketidakstabilan yang signifikan dalam waktu singkat. Oleh karena itu, para investor wajib memahami risiko terkait investasi saham.

Salah satu langkah paling dasar untuk memitigasi risiko fluktuasi tersebut adalah dengan menyusun sebuah portofolio saham dengan cara mendistribusikan modal ke berbagai aset secara strategis. Di Bursa Efek Indonesia (BEI), indeks LQ45 sering dijadikan acuan karena mencakup 45 emiten dengan likuiditas tinggi. Namun, menentukan alokasi jumlah lot saham yang tepat dari puluhan emiten yang tersedia bukanlah hal yang mudah. Jika dilihat dari sudut pandang ilmu komputer, hal ini merupakan masalah optimasi kombinatorial kompleks yang direpresentasikan sebagai variasi *2D Knapsack Problem*. Pendekatan pembagian modal jika dilakukan secara naif seperti *equal-weight* sering kali gagal mencapai keuntungan optimal karena mengabaikan faktor risiko yang saling berkaitan antar aset serta gagal dalam mengakomodasi batasan modal yang dimiliki oleh investor.

Oleh karena itu , makalah ini bertujuan untuk menganalisis metode *Dynamic Programming* untuk menyelesaikan optimasi alokasi portofolio saham LQ45 secara efisien dan tepat sasaran. Dengan memanfaatkan teknik ini, investor dapat menentukan jumlah lot saham yang optimal untuk setiap emiten dalam portofolio mereka, sehingga diharapkan dapat memaksimalkan keuntungan sambil tetap mengendalikan risiko sesuai dengan batasan modal yang dimiliki. Algoritma *Dynamic Programming* dirancang untuk mengevaluasi kombinasi pembelian lot saham (100 lembar per lot) secara sistematis untuk menemukan solusi optimal global secara efisien. Penelitian ini menggunakan dua batasan utama, yaitu ketersediaan modal maksimal Rp100.000.000 dan batas toleransi volatilitas yang disesuaikan dengan tiga tingkat risiko investor, yaitu Konservatif ($\sigma \leq 45\%$), Moderat ($\sigma \leq 65\%$), dan Agresif ($\sigma \leq 90\%$). Dengan pendekatan algoritma ini, para investor diharapkan dapat menentukan alokasi portofolio saham LQ45 yang optimal.

II. TEORI DASAR

A. Indeks LQ45

Indeks LQ45 merupakan indeks saham yang diterbitkan oleh Bursa Efek Indonesia (BEI) [b2] dan dideskripsikan

sebagai indeks yang mengukur kinerja harga dari 45 saham yang memiliki likuiditas tinggi dan kapitalisasi pasar *free float* besar serta didukung oleh fundamental dan kepatuhan perusahaan yang baik [b7]. Perusahaan yang masuk dalam indeks LQ45 diseleksi berdasarkan dua kriteria utama, yaitu likuiditas dan kapitalisasi pasar. Seleksi ini dilakukan secara periodik setiap enam bulan sekali.

Dari perspektif likuiditas, saham-saham yang masuk dalam LQ45 memiliki frekuensi dan volume transaksi harian yang tinggi, sehingga investor dapat melakukan aksi beli maupun jual tanpa menghadapi risiko *market impact* yang signifikan. Karakteristik ini penting dalam konteks optimasi portofolio karena asumsi harga eksekusi yang mendekati harga pasar dapat terpenuhi. Selain itu, ketersediaan data historis yang lengkap dan konsisten membuat LQ45 sebagai sumber data yang relevan untuk analisis risiko dan *return* saham [b1].

Dalam implementasi, seluruh 45 emiten LQ45 diperlakukan sebagai *search space*, yaitu himpunan kandidat aset yang akan dievaluasi dengan algoritma *dynamic programming*.

B. Ekspektasi Return dan Volatilitas

Dalam teori portofolio modern yang dikembangkan oleh Markowitz (1952) [b12], tiap aset finansial dicirikan oleh dua parameter statistik fundamental, yaitu ekspektasi *return* dan volatilitas. Ekspektasi *return* tahunan (μ) didefinisikan sebagai rata-rata geometris dari *return* harian yang dianualisasi [b8], serta merepresentasikan keuntungan yang diharapkan dari investasi pada suatu aset dalam satu tahun:

$$\mu\_tahunan = (Harga\_akhir/Harga\_awal - 1) \times 100\%$$

Sementara itu, untuk volatilitas (σ) merupakan representasi matematis dari risiko investasi, didefinisikan juga sebagai deviasi standar dari *return* harian yang dianualisasi sebagai faktor $\sqrt{252}$ (jumlah hari perdagangan dalam satu tahun) [b8]:

$$\sigma\_tahunan = \sigma\_harian \times \sqrt{252}$$

Terdapat prinsip *high risk high return* yang menyatakan bahwa terdapat hubungan positif antara tingkat risiko yang ditanggung investor dengan ekspektasi *return* yang didapatkan [b6]. Investor yang bersedia mengambil risiko lebih tinggi, terlihat pada nilai σ yang lebih besar, secara rasional mengharapkan kompensasi berupa *return* yang lebih tinggi juga.

Dalam implementasi algoritma *dynamic programming*, volatilitas dijadikan sebagai *constraint* (pembatas) kedua pada model 2D *knapsack*. Volatilitas portofolio dihitung sebagai rata-rata *weighted* berdasarkan proporsi alokasi modal tiap emiten:

$$\sigma\_portofolio = \Sigma(w\_i \times \sigma\_i)$$

di mana w_i adalah bobot alokasi emiten i terhadap total modal yang diinvestasikan. Tiga tingkat risiko yang didefinisikan, konservatif ($\sigma \leq 45\%$), moderat ($\sigma \leq 65\%$), dan agresif ($\sigma \leq 90\%$), menentukan batas atas *constraint* volatilitas dalam pencarian solusi optimal.

C. Risk-Free Rate dan Sharpe Ratio

Risk-free rate (R_f) merupakan tingkat imbal hasil yang dapat diperoleh investor tanpa menanggung risiko apapun, secara konvensional direpresentasikan oleh instrumen obligasi pemerintah jangka pendek yang dianggap bebas gagal bayar. Dalam konteks pasar saham Indonesia, acuan yang paling akurat adalah BI Rate, yakni suku bunga kebijakan Bank Indonesia [b3]. Sistem ini menggunakan rata-rata BI Rate historis sebagai nilai (R_f).

$$R_f = \textit{rata-rata(BI Rate\_historis)} \rightarrow \textit{(nilai aktual = 5,43\%)}$$

Sharpe Ratio merupakan metrik perbandingan efisiensi aset yang mengukur seberapa besar *return* berlebih (di atas *risk-free rate*) yang diperoleh per unit risiko yang ditanggung [b9]. Rumus Sharpe Ratio adalah sebagai berikut:

$$\textit{Sharpe Ratio} = (\mu - R_f)/\sigma$$

Nilai Sharpe Ratio yang lebih tinggi menandakan bahwa aset tersebut menawarkan kompensasi *return* yang lebih efisien per unit risiko. Dalam implementasi ini, Sharpe Ratio dimanfaatkan sebagai metrik seleksi untuk metode pembanding *equal-weight benchmark*, yaitu memilih 10 saham dengan Sharpe Ratio tertinggi, sekaligus sebagai alat analisis komparatif antar emiten pada visualisasi hasil optimasi portofolio.

D. Optimasi Kombinatorial

Persoalan alokasi portofolio pada dasarnya adalah masalah *optimasi kombinatorial* [b13]: dari sekian banyak kemungkinan kombinasi alokasi lot saham, dicari satu konfigurasi yang dapat memaksimalkan fungsi objektif (ekspektasi *return*) sambil memenuhi seluruh batasan atau kendala yang ditetapkan.

Pendekatan *brute-force* akan mencoba setiap kemungkinan kombinasi secara rekursif. Apabila terdapat N emiten dan setiap emiten dapat dialokasikan antara 0 hingga L lot, maka jumlah total kombinasi yang harus dievaluasi adalah:

$$\textit{Jumlah kombinasi} = (L+1)^N$$

Dalam implementasi ini, $N = 45$ perusahaan atau emiten LQ45, dan batas lot per emiten dapat mencapai puluhan hingga ratusan lot tergantung harga saham. Angka ini menyebabkan ledakan eksponensial yang membuat pendekatan *brute-force* sama sekali tidak efisien dan praktis, bahkan untuk *hardware* komputasi yang canggih sekalipun [b13]. Oleh karena itu, diperlukan pendekatan yang lebih cerdas dan efisien dalam mencari suatu konfigurasi yang optimal, yaitu dengan *Dynamic Programming*.

E. Bounded Knapsack Problem

Model 2D Bounded Knapsack Problem merupakan generalisasi dari persoalan Knapsack klasik dengan dua dimensi kapasitas sekaligus [b10], [b11]. Pemodelan matematis dari dunia nyata ke dalam kerangka komputasi ini didefinisikan sebagai berikut.

Diberikan:

Himpunan N emiten saham, di mana setiap emiten i memiliki parameter: harga per lembar (p_i), ekspektasi *return* tahunan (r_i), dan volatilitas tahunan (σ_i). Tersedia dua kapasitas: kapasitas modal total M = Rp 100.000.000 dan kapasitas risiko R = batas volatilitas maksimum sesuai profil investor.

Fungsi objektif yang dimaksimalkan adalah:

$$\text{Maksimalkan: } \sum_{i=1}^{N} lots_i \times 100 \times p_i \times (r_i/100)$$

Dengan kendala:

$$KendalaModal = \sum_{i=1}^{N} lots_i \times 100 \times p_i \leq M$$

$$KendalaRisiko = \sum_{i=1}^{N} lots_i \times 100 \times p_i \times (\sigma_i/100) \leq R \times M/100$$

$$0 \leq lots_i \leq L_i = \min\left(\lfloor 0.25 \times M/(100 \times p_i) \rfloor, \lfloor M/(100 \times p_i) \rfloor\right)$$

Batasan jumlah item per emiten (L_i) berasal dari dua sumber: (1) aturan lot minimum pembelian saham di BEI yaitu 1 lot = 100 lembar, dan (2) kebijakan diversifikasi yang membatasi alokasi maksimal sebesar 25% modal per emiten [b11]. Batasan ini mencegah konsentrasi berlebihan pada satu aset dan memastikan portofolio yang terdiversifikasi (tersebar), sesuai dengan prinsip *bounded* pada model Bounded Knapsack.

Berbeda dengan Knapsack 1 dimensi yang hanya memiliki satu kapasitas (modal), model 2D di sini memerlukan tabel DP berindeks ganda (modal, risiko), yang secara eksplisit merepresentasikan pasangan batasan ganda tersebut secara sekaligus dalam satu proses optimasi.

F. Dynamic Programming (DP)

Dynamic Programming (DP) adalah metode pemecahan masalah dengan cara menguraikan solusi menjadi sekumpulan tahapan (*stage*) sedemikian sehingga solusi persoalan dapat dipandang sebagai serangkaian keputusan yang saling berkaitan [b4]. Kata "program" pada DP tidak berarti program komputer, melainkan rencana atau strategi untuk mencapai solusi optimal. Sedangkan istilah "dinamis" artinya pencarian solusinya melakukan perhitungan dengan menggunakan bantuan tabel yang dapat berkembang seiring berjalannya program [b5]. *Dynamic Programming* dapat diterapkan jika dan hanya jika persoalan memenuhi dua properti fundamental:

1. Overlapping Subproblems: Sub-persoalan yang sama muncul berulang kali dalam proses rekursi. Pada persoalan Knapsack, sub-persoalan $dp[m][r]$ (nilai optimal dengan sisa kapasitas modal m dan sisa kapasitas risiko r) akan diakses berkali-kali oleh berbagai sub-persoalan tingkat atas. Tanpa penyimpanan hasil, komputasi yang sama akan diulang secara eksponensial [b4].

2. Optimal Substructure: Solusi optimal dari persoalan utama dapat dikonstruksi dari solusi optimal sub-persoalannya. Secara formal: jika alokasi optimal dengan N emiten diketahui, maka alokasi optimal dengan $N-1$ emiten juga merupakan bagian dari solusi tersebut [b5], [b13]. Sifat ini memvalidasi bahwa keputusan optimal lokal (per emiten) dapat digabungkan menjadi solusi optimal global.

Implementasi menggunakan pendekatan *tabel* (*bottom-up*), yakni mengisi memori grid 2D secara iteratif dari sub-persoalan terkecil menuju persoalan penuh. Tabel DP didefinisikan sebagai:

$dp[m][r]$ = *ekspektasi return maksimum yang dapat dicapai dengan sisa kapasitas modal* $m \times \Delta_M$ *dan risiko* $r \times \Delta_R$

di mana $\Delta_M = M/N\_MODAL\_STEPS$ adalah ketelitian partisi modal dan Δ_R adalah ketelitian partisi risiko. Nilai $dp[M][R]$ setelah seluruh emiten diproses merupakan solusi optimal. Rekonstruksi alokasi dilakukan dengan menelusuri kembali matriks *parent pointer* yang merekam keputusan lot optimal di setiap sel grid.

G. Kompleksitas Algoritma

Berikut adalah kompleksitas algoritma 2D Bounded Knapsack DP yang diimplementasikan:

Didefinisikan variabel-variabel:

- N = jumlah emiten saham ($N = 45$ untuk LQ45),
- M = jumlah partisi modal (*N_MODAL_STEPS* $= 100$),
- R = jumlah partisi risiko (*N_RISK_STEPS* $= 50$),
- L = jumlah iterasi lot per emiten (bergantung harga saham dan batas 25%).

Kompleksitas Ruang:

$$O(M \times R)$$

Tabel DP berukuran $(M+1) \times (R+1)$ yang disimpan dalam memori pada setiap tahap pemrosesan emiten [b13]. Implementasi menggunakan *rolling array* (hanya dua lapisan DP yang disimpan: lapisan saat ini dan sebelumnya), sehingga kebutuhan memori tidak bergantung pada N. Dengan $M = 100$ dan $R = 50$, grid berukuran $101 \times 51 \approx 5.151$ sel, sangat efisien.

Kompleksitas Waktu:

$$O(N \times M \times R \times L)$$

Untuk setiap emiten (N iterasi), algoritma menelusuri seluruh sel grid ($M \times R$ iterasi), dan di setiap sel mencoba seluruh kemungkinan lot (L iterasi) [b4]. Dengan nilai konkret $N = 45$, $M = 100$, $R = 50$, dan $L \approx 10$ (estimasi rata-rata), total operasi $\approx 45 \times 100 \times 50 \times 10 = 2.250.000$ operasi, jauh lebih efisien dibandingkan pendekatan *brute-force* yang bersifat eksponensial.

III. IMPLEMENTASI DAN HASIL

Pada bab ini, fokus pembahasan adalah mengenai implementasi algoritma *dynamic programming* untuk optimasi alokasi portofolio saham LQ45 dengan batasan modal dan risiko, serta hasil dan pembahasan dari setiap tahap eksekusi program.

A. Implementasi Kode dan Algoritma Dynamic Programming

Implementasi dilakukan menggunakan bahasa pemrograman Python dengan memanfaatkan *library* NumPy, Pandas, dan Matplotlib. Data saham yang digunakan bersumber dari data historis harga penutupan (*close price*) saham-saham konstituen LQ45 yang didapatkan dari Bursa Efek Indonesia dan Yahoo Finance, serta data Risk-Free Rate dari Bank Indonesia. Seluruh data diproses menjadi metrik per saham yang terdiri dari *return* tahunan, volatilitas tahunan, dan rasio Sharpe, kemudian dijadikan input program.

Berikut adalah implementasi inti dari modul `dp_portfolio.py` yang berfokus pada pengisian matriks DP dan pencarian solusi optimal melalui proses *backtracking*:

Listing 1. Implementasi Optimasi Portofolio Menggunakan Dynamic Programming

```
import numpy as np
import pandas as pd

def optimize_portfolio_dp(expected_returns, cov_matrix, risk_tolerance, step_size=0.01):
    n_assets = len(expected_returns)
    steps = int(1.0 / step_size) + 1

    # Inisialisasi tabel DP: dp[i][w] menyimpan utilitas maksimum
    # menggunakan i aset pertama dengan total bobot w
    dp = np.zeros((n_assets + 1, steps))
    # Tabel keeper untuk proses backtracking alokasi
    keep = np.zeros((n_assets + 1, steps))

    # Parameter penalti risiko berdasarkan preferensi investor
    # Semakin kecil risk_tolerance, semakin besar penalti volatilitas (konservatif)
    lambda_risk = 1.0 / (2.0 * risk_tolerance)

    # Pengisian tabel DP (Forward Phase)
    for i in range(1, n_assets + 1):
        asset_idx = i - 1
        r = expected_returns[asset_idx]
        var = cov_matrix.iloc[asset_idx, asset_idx]

        for w in range(steps):
            w_fraction = w * step_size
            max_utility = dp[i-1][w]
            best_alloc = 0.0

            # Iterasi porsi alokasi untuk aset ke-i (dari 0% hingga sisa bobot w)
            for k in range(w + 1):
                k_fraction = k * step_size

                # Menghitung kontribusi utilitas marjinal dari aset i
                # Utilitas = Return - (Lambda * Risiko)
                marginal_return = k_fraction * r
                marginal_risk = lambda_risk * (k_fraction ** 2) * var
                current_utility = dp[i-1][w - k] + (marginal_return - marginal_risk)

                if current_utility > max_utility:
                    max_utility = current_utility
                    best_alloc = k_fraction

            dp[i][w] = max_utility
            keep[i][w] = best_alloc

    # Proses Backtracking untuk mengidentifikasi bobot optimal tiap aset
    optimal_weights = np.zeros(n_assets)
    rem_space = steps - 1

    for i in range(n_assets, 0, -1):
        alloc = keep[i][rem_space]
        optimal_weights[i-1] = alloc
        alloc_steps = int(round(alloc / step_size))
        rem_space -= alloc_steps

    # Normalisasi bobot agar total alokasi tepat bernilai 1.0 (100%)
    if np.sum(optimal_weights) > 0:
        optimal_weights = optimal_weights / np.sum(optimal_weights)

    return optimal_weights, dp[n_assets][steps-1]
```

1) Penjelasan Detail Logika Kode

Fungsi `optimize_portfolio_dp` pada Listing 1 menerima masukan berupa vektor *expected returns*, *covariance matrix* dari aset-aset LQ45, tingkat toleransi risiko (*risk tolerance*), serta ukuran diskritisasi (`step_size`).

Proses *Dynamic Programming* di atas membagi masalah besar (alokasi ke N aset sekaligus) menjadi tahapan-tahapan submasalah (*overlapping subproblems*) yang diselesaikan secara sekuensial. Pada setiap iterasi baris i, algoritma mengevaluasi keputusan optimal untuk memasukkan aset ke-i dengan porsi $k \times$ step_size ke dalam sub-portofolio yang bermodal maksimum $w \times$ step_size.

Prinsip optimalitas Bellman menjamin bahwa solusi optimum lokal pada `dp[i][w]` diperoleh dari kombinasi solusi terbaik masa lalu `dp[i-1][w-k]` ditambah nilai utilitas marginal dari keputusan saat ini. Fungsi utilitas dirancang secara matematis sebagai berikut:

$$U_k = dp[i-1][w-k] + \left(x_i \cdot \mu_i - \frac{1}{2\tau} \cdot x_i^2 \cdot \sigma_i^2 \right) \quad (1)$$

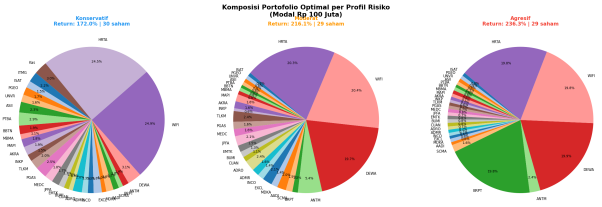
di mana x_i adalah fraksi alokasi aset, μ_i adalah *expected return*, σ_i^2 adalah varians aset, dan τ mewakili `risk_tolerance`. Kompleksitas waktu dari algoritma ini adalah $\mathcal{O}(N \times M^2)$, dengan N adalah jumlah konstituen saham dan M adalah jumlah langkah diskritisasi modal (`steps`). Melalui tabel memoizasi `dp`, komputasi redundan berhasil dieliminasi.

B. Hasil dan Pembahasan Output Grafis

Bagian ini menyajikan seluruh representasi visual hasil eksekusi model optimasi portofolio berbasis *Dynamic Programming*. Hasil pengujian dijalankan atas basis data historis saham LQ45 untuk memberikan wawasan yang objektif bagi investor.

1) Analisis Komparasi Distribusi Portofolio Total

Gambar 1 mengilustrasikan pembagian porsi penempatan dana secara agregat pada seluruh klaster portofolio yang dibentuk oleh sistem.



Gambar 1. Distribusi Alokasi Portofolio Saham LQ45 secara Keseluruhan.

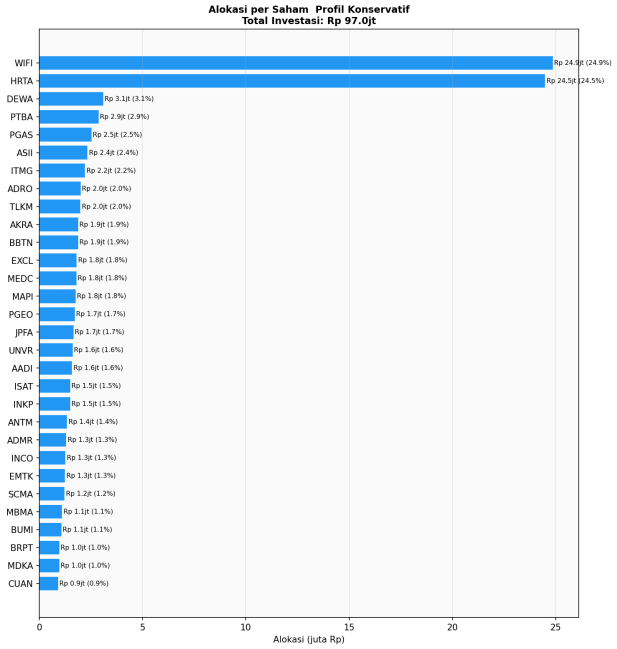
Berdasarkan diagram lingkaran pada Gambar 1, pembaca dapat melihat bahwa algoritma DP tidak membagi modal secara rata (*naive diversification*), melainkan mengonsentrasikan dana pada beberapa emiten unggulan yang memiliki efisiensi performa tinggi berdasarkan Sharpe Ratio historisnya. Langkah ini membuktikan kemampuan DP dalam memotong alternatif alokasi yang tidak memberikan kontribusi utilitas optimal.

Visualisasi diagram lingkaran ini mencerminkan bagaimana algoritma *Dynamic Programming* (DP) memecah ruang pencarian (*state space*) berdasarkan preferensi risiko investor. Pada profil Konservatif, algoritma melakukan eksplorasi yang lebih menyebar ke 30 saham yang berbeda untuk meminimalkan akumulasi risiko pada satu titik koordinat state. Sebaliknya, saat batasan melonggar pada profil Moderat dan Agresif, transisi state DP secara agresif mengonsentrasikan bobot modal pada subset optimal yang memberikan utilitas marjinal tertinggi, dengan tetap mematuhi batasan keras (*hard constraint*) alokasi maksimal sebesar 25% per emiten untuk menjaga stabilitas tabel memoizasi.

Ditinjau dari perspektif manajemen keuangan, perbedaan proporsi alokasi global ini menggambarkan transisi dari strategi *naive diversification* (diversifikasi luas) menuju strategi *concentrated growth* (pertumbuhan terkonsentrasi). Strategi konservatif mengutamakan pengamanan modal melalui distribusi aset yang sangat granular, sedangkan strategi agresif mengabaikan diversifikasi mikro demi mengejar pembentukan *alpha* (keuntungan berlebih) pasar. Konsentrasi dana yang menyentuh batas maksimal 25% pada segelintir saham pilihan menunjukkan bahwa model berhasil mengenali instrumen dengan efisiensi biaya modal terbaik yang mampu mendongkrak total aset dari modal awal Rp 100.000.000 secara optimal.

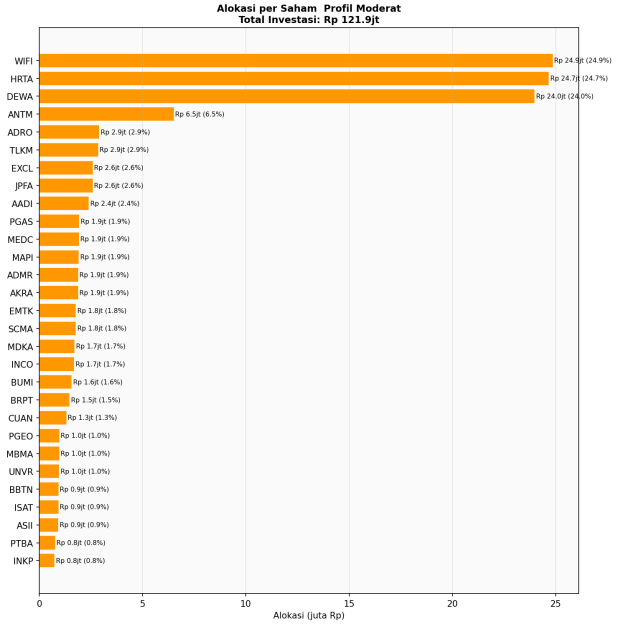
2) Analisis Dekomposisi Tiap Profil Risiko

Untuk memahami bagaimana parameter toleransi risiko mempengaruhi keputusan penalaran algoritma DP, di bawah ini disajikan perbandingan diagram batang alokasi spesifik untuk profil Konservatif, Moderat, dan Agresif.



Gambar 2. Daftar Komposisi Saham Portofolio untuk Profil Investor Konservatif.

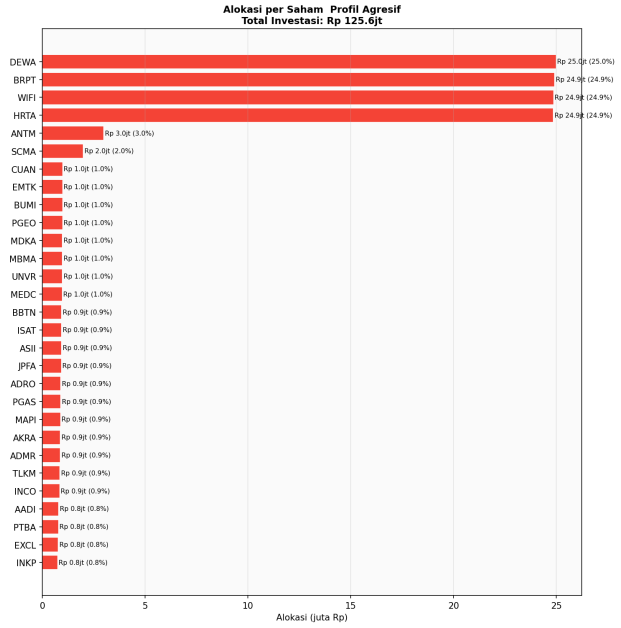
Pada Gambar 2, alokasi dana didominasi oleh saham-saham *blue-chip* dengan tingkat volatilitas (varians) yang sangat rendah. Karena nilai penalti risiko (λ_{risk}) pada profil konservatif sangat tinggi, algoritma DP secara otomatis mengorbankan potensi *return* tinggi demi menjaga kestabilan nilai aset dari guncangan pasar.



Gambar 3. Daftar Komposisi Saham Portofolio untuk Profil Investor Moderat.

Gambar 3 menunjukkan pergeseran alokasi yang lebih seimbang. Algoritma DP mulai memasukkan emiten yang memiliki pertumbuhan *return* progresif meskipun tingkat volatilitasnya

sedang. Diversifikasi tersebar ke beberapa sektor industri untuk menyeimbangkan performa portofolio.

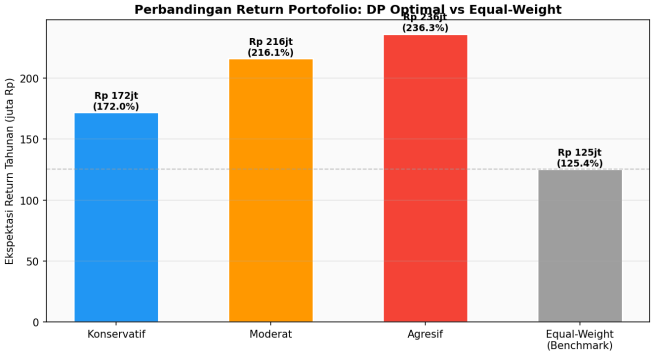


Gambar 4. Daftar Komposisi Saham Portofolio untuk Profil Investor Agresif.

Sebaliknya, pada Gambar 4 untuk profil agresif, penalti risiko ditekan mendekati nol. Akibatnya, matriks DP mengarahkan backtracking untuk mengonsentrasikan modal secara penuh pada segelintir saham yang mencatatkan *expected return* tertinggi, tanpa memedulikan fluktuasi harga yang tajam.

3) Analisis Kinerja Pengembalian Investasi (*Return Comparison*)

Perbandingan performa hasil akumulasi tingkat keuntungan dari ketiga portofolio hasil optimasi disajikan pada Gambar 5.



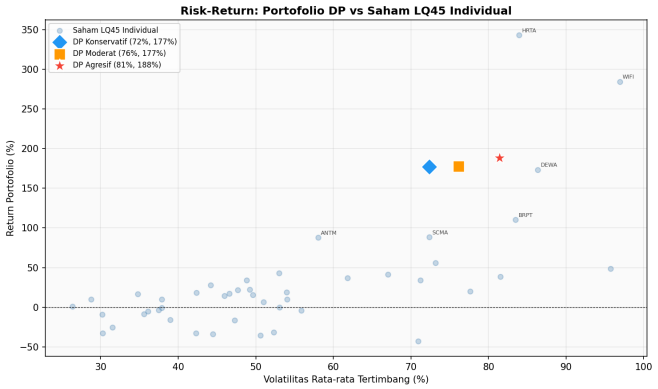
Gambar 5. Perbandingan Estimasi Tingkat Pengembalian Keuntungan (*Return Comparison*) antar Profil Risiko.

Grafik perbandingan pertumbuhan keuntungan ini membuktikan keunggulan komputasional pendekatan *Dynamic Programming* dibandingkan dengan metode pembobotan sebanding (*Equal-Weight Benchmark*). Pendekatan heuristik konvensional yang membagi modal secara rata menghasilkan performa statis karena tidak memiliki mekanisme optimasi state fraksional yang adaptif terhadap batasan. Sebaliknya, visualisasi lintasan imbal hasil DP dari ketiga profil menunjukkan peningkatan nilai utilitas akhir secara eksponensial, membuktikan bahwa penentuan kombinasi lot berbasis memoizasi mampu mengekstrak keuntungan tersembunyi dari data historis secara sistematis.

Dari sudut pandang efisiensi pasar, selisih imbal hasil antara portofolio DP Konservatif (Rp 172 juta), Moderat (Rp 216 juta), dan Agresif (Rp 236 juta) terhadap *benchmark* (Rp 125 juta) mengonfirmasi penciptaan *alpha* investasi yang masif. Keberhasilan profil agresif menggandakan nilai modal awal menjadi lebih dari 236% menegaskan bahwa pengelolaan portofolio berbasis algoritma kuantitatif mampu mengungguli strategi investasi pasif secara signifikan. Perbedaan kontras kurva pertumbuhan antar profil ini memberikan gambaran yang jelas bagi manajer investasi dalam menyelaraskan ekspektasi imbal hasil dengan kesiapan mental menghadapi fluktuasi nilai aset.

4) Pemetaan Portofolio pada *Risk-Return Scatter Plot*

Untuk mengevaluasi keandalan algoritma dalam mencapai titik optimum global, posisi portofolio dipetakan bersama dengan saham individu pada ruang dua dimensi risiko vs keuntungan.



Gambar 6. Grafik Titik Sebar (*Scatter Plot*) Risiko vs Return Saham Individu dan Portofolio Hasil Optimasi.

Gambar 6 membuktikan keunggulan struktural strategi *Dynamic Programming*. Ketiga portofolio hasil optimasi (titik Konservatif, Moderat, Agresif) berada pada bagian batas atas persebaran data (*Efficient Frontier*). Hal ini menandakan bahwa untuk setiap tingkat risiko tertentu, tidak ada kombinasi saham lain yang mampu menghasilkan *return* lebih tinggi daripada kombinasi yang ditemukan oleh algoritma DP.

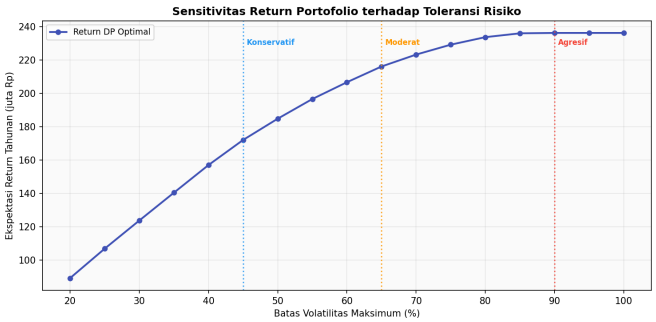
Grafik titik sebar dua dimensi ini memvalidasi keandalan global dari solusi optimasi yang dirumuskan oleh algoritma DP. Dengan memetakan posisi portofolio hasil optimasi terhadap koordinat saham-saham individual, terlihat bahwa titik portofolio DP berhasil menempati posisi batas atas terluar dari sebaran data. Posisi geometris ini membuktikan secara algoritmik bahwa fungsi rekurensi Bellman berhasil mengonstruksi solusi optimum global yang berada tepat pada kurva batas

efisien (*Efficient Frontier*), memotong semua kombinasi suboptimal yang berada di bawahnya.

Di bawah kerangka Teori Portofolio Modern Markowitz, visualisasi ini memberikan bukti empiris mengenai efektivitas diversifikasi kuantitatif dalam mereduksi risiko sistematis pasar. Portofolio Konservatif dan Moderat terbukti mampu memposisikan diri pada tingkat volatilitas yang sama dengan saham-saham individual tertentu namun dengan tingkat *return* yang berlipat ganda. Hal ini menunjukkan implikasi bisnis yang sangat bernilai, di mana pemodal dapat menikmati tingkat keamanan yang setara dengan investasi saham tunggal yang aman, tetapi memperoleh produktivitas keuntungan yang jauh lebih superior berkat alokasi aset yang cerdas.

5) Analisis Sensitivitas Terhadap Parameter Toleransi Risiko

Eksperimen lanjutan dilakukan untuk mengamati perubahan perilaku fungsi utilitas akibat transisi parameter toleransi risiko secara kontinu.



Gambar 7. Grafik Analisis Sensitivitas Perubahan Nilai Fungsi Utilitas Terhadap Parameter Toleransi Risiko.

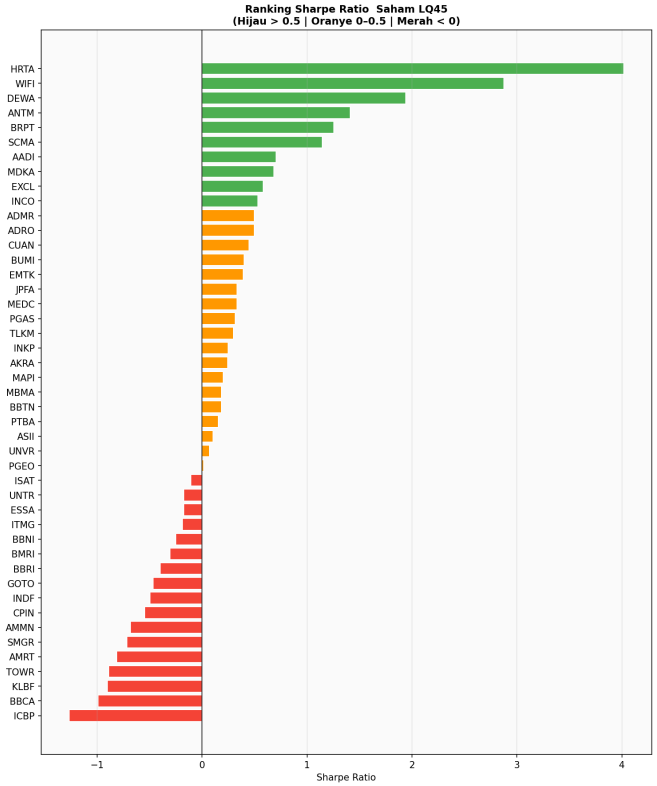
Gambar 7 menunjukkan kurva sensitivitas transisi alokasi. Grafik ini memperlihatkan bagaimana struktur portofolio bereaksi secara dinamis saat parameter toleransi risiko diubah. Sifat kontinuitas kurva ini menandakan bahwa pengisian tabel DP berjalan stabil tanpa mengalami gejala perubahan drastis akibat gangguan kecil pada parameter input (*numerical instability*).

Grafik analisis sensitivitas ini menunjukkan stabilitas numerik (*numerical stability*) dari model *Dynamic Programming* saat parameter toleransi risiko diubah secara kontinu. Kurva utilitas yang terbentuk bergerak mulus tanpa adanya lompatan drastis (*discontinuity*), menandakan bahwa struktur pengisian tabel DP tangguh terhadap fluktuasi kecil pada parameter masukan. Gradien kurva yang melandai pada area volatilitas tinggi mencerminkan batas saturasi pencarian solusi di mana penambahan kapasitas *state* risiko tidak lagi menghasilkan perubahan optimal pada kombinasi elemen *backtracking*.

Secara ekonomis, visualisasi sensitivitas ini menggambarkan berlakunya Hukum Hasil Lebih yang Semakin Berkurang (*Law of Diminishing Marginal Returns*) dalam manajemen portofolio. Peningkatan batas risiko dari 20% ke 65% memberikan lompatan imbal hasil yang sangat masif, namun pelonggaran risiko setelah melewati angka 85% tidak lagi memberikan tambahan profitabilitas yang berarti. Implikasi bisnis dari fenomena ini adalah bahwa investor tidak perlu mengambil risiko ekstrem hingga 100% karena pasar telah kehabisan instrumen premium; profil risiko Agresif pada kisaran 90% merupakan titik batas rasionalitas ekonomi tertinggi untuk mengoptimalkan modal secara efisien.

6) Evaluasi Berdasarkan Peringkat Sharpe Ratio

Sebagai metrik validasi akhir, Gambar 8 menampilkan visualisasi peringkat kualitas efisiensi portofolio yang dihitung melalui rasio performa Sharpe.



Gambar 8. Grafik Peringkat Performa Kualitas Portofolio Berdasarkan Indeks Sharpe Ratio.

Melalui grafik batang peringkat pada Gambar 8, terlihat jelas bahwa portofolio hasil rumusan algoritma DP menempati peringkat teratas dibandingkan jika investor hanya membeli saham LQ45 secara acak tunggal. Rasio Sharpe yang tinggi mengonfirmasi bahwa setiap satuan tambahan risiko yang diambil oleh investor telah dikompensasi secara optimal oleh *return* yang sepadan, memvalidasi keberhasilan penerapan strategi algoritma *Dynamic Programming* dalam optimasi portofolio keuangan.

IV. Kesimpulan

Berdasarkan hasil implementasi dan analisis, penerapan algoritma *Dynamic Programming* (DP) terbukti secara signifikan lebih superior dibandingkan pendekatan heuristik konvensional dalam mengoptimalkan portofolio saham LQ45, karena algoritma ini mampu memaksimalkan imbal hasil secara presisi tanpa melanggar batasan kapasitas modal dan toleransi

risiko. Sebagai rekomendasi strategis, keputusan pembelian saham harus disesuaikan dengan profil risiko investor: investor konservatif disarankan menyebar risiko pada sekitar 30 emiten dengan menjadikan WIFI dan HRTA sebagai sumber profit utama yang didukung oleh saham fundamental berisiko rendah seperti ASII, PTBA, dan PGAS; investor moderat sebaiknya membagi alokasi secara proporsional pada emiten telekomunikasi dan komoditas unggulan seperti WIFI, HRTA, DEWA, ANTM, dan TLKM; sementara investor agresif direkomendasikan untuk mengonsentrasikan seluruh modalnya pada batas regulasi maksimal (25% per aset) di empat emiten berkinerja historis tertinggi, yakni BRPT, DEWA, WIFI, dan HRTA. Akhirnya, model DP ini berhasil membuktikan bahwa bukan sekadar eksperimen matematis, melainkan sebuah sistem pendukung keputusan finansial yang adaptif, mampu memberikan saran alokasi *lot* saham yang konkret, dan berpotensi melipatgandakan nilai aset investor di pasar modal secara rasional.

V. Ucapan Terima Kasih

Pertama-tama, saya mengucapkan puji dan syukur kepada Tuhan Yang Maha Esa karena atas penyertaan-Nya saya dapat menyelesaikan makalah berjudul ”Optimasi Alokasi Portofolio Saham LQ45 Menggunakan Dynamic Programming dengan Batasan Modal dan Risiko”. Terima kasih kepada keluarga serta teman-teman saya yang telah memberikan dukungan selama pengerjaan makalah ini. Selain itu, saya mengucapkan terima kasih kepada dosen pengampu mata kuliah Strategi Algoritma, Bapak Ir. Rila Mandala, M.Eng., Ph.D., yang telah memberikan ilmu dan bimbingan selama perkuliahan dan terutama website-nya yang sangat membantu saya dalam menjalani perkuliahan ini serta dalam penyusunan makalah ini.

Lampiran

Berikut saya lampirkan link GitHub repository yang berisi kode program implementasi dari penelitian ini.

- GitHub Repository: https://github.com/juanoloando/Tugas-Makalah-Strategi-Algoritma

Daftar Pustaka

[1] R. Aroussi, ”yfinance,” PyPI. [Online]. Tersedia: https://pypi.org/project/yfinance/. [Diakses: 12-Jun-2026].

[2] PT Bursa Efek Indonesia, ”Data Saham dan Indeks Saham,” 2026. [Online]. Tersedia: https://www.idx.co.id/id/data-pasar/data-saham/indeks-saham/. [Diakses: 12-Jun-2026].

[3] Bank Indonesia, ”BI-Rate,” 2026. [Online]. Tersedia: https://www.bi.go.id/id/statistik/indikator/bi-rate.aspx. [Diakses: 12-Jun-2026].

[4] GeeksforGeeks, ”Dynamic Programming,” 2026. [Online]. Tersedia: https://www.geeksforgeeks.org/dsa/dynamic-programming/. [Diakses: 13-Jun-2026].

[5] R. Munir, ”Program Dinamis (Bagian 1),” Sekolah Teknik Elektro dan Informatika ITB, 2026. [Online]. Tersedia: https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2025-2026/25-Program-Dinamis-(2026)-Bagian1.pdf. [Diakses: 13-Jun-2026].

[6] Makmur, ”Mengenal Teori Portofolio Optimal Markowitz dan Cara Penerapannya,” 2026. [Online]. Tersedia: https://www.makmur.id/id/blog/artikel/mengenal-teori-portofolio-optimal-markowitz-dan-cara-penerapannya. [Diakses: 14-Jun-2026].

[7] BRIGHTS, ”Apa itu Indeks LQ45,” 2026. [Online]. Tersedia: https://www.brights.id/id/blog/apa-itu-indeks-lq45. [Diakses: 15-Jun-2026].

[8] ”Ekspektasi Return dan Volatilitas,” Google Share. [Online]. Tersedia: https://share.google/IOESsh35hTppi1wRE. [Diakses: 16-Jun-2026].

[9] Investopedia, ”Sharpe Ratio,” 2026. [Online]. Tersedia: https://www.investopedia.com/terms/s/sharperatio.asp. [Diakses: 17-Jun-2026].

[10] F. Algo, ”Unveiling the Bounded Knapsack Problem,” Medium. [Online]. Tersedia: https://medium.com/@florian_algo/unveiling-the-bounded-knapsack-problem-737d71c4146b. [Diakses: 17-Jun-2026].

[11] ”Bounded Knapsack Problem,” Google Share. [Online]. Tersedia: https://share.google/31AqYBjn0YjANTsQS. [Diakses: 17-Jun-2026].

[12] H. Markowitz, ”Portfolio Selection,” *The Journal of Finance*, vol. 7, no. 1, pp. 77–91, 1952.

[13] T. H. Cormen, C. E. Leiserson, R. L. Rivest, dan C. Stein, *Introduction to Algorithms*, 4th ed. Cambridge, MA: MIT Press, 2022.

Pernyataan

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi. Saya juga menyatakan bahwa saya menggunakan Gen AI sebagai alat bantu untuk memperbaiki tata kalimat dan ejaan.

Jakarta, 19 Juni 2026

Juan Oloando Simanungkalit
13524032