AICOMS
Applied Information Technology and Computer Science
e-ISSN: 2964-7703
Vol. 5 No. 1, Juni-2026, pp. 208-219

Perbandingan Algoritma Greedy dan Dynamic Programming Pada Optimasi Playlist Spotify Untuk Jogging

Fadhel Muhammad [1], Muhammad Radja Juang Jamemiko [2] and Yohannes [3,]

[1] Program Studi Informatika, Fakultas Ilmu Komputer dan Rekayasa, Universitas Multi Data Palembang; fadhelmuhammad_2327250070@mhs.mdp.ac.id; muhammadradjajuangjamemiko_2327250051@mhs.mdp ac.id; yohannesmasterous@mdp.ac.id
* Korespondensi: fadhelmuhammad_2327250070@mhs.mdp.ac.id

Info Artikel:
Dikirim: 26 April 2026
Direvisi: 26 Mei 2026
Diterima: 06 Juni 2026

Abstract: Spotify provides audio metadata that can be utilized to support physical activities such as jogging. This study compares the performance of Greedy and Dynamic Programming algorithms for Spotify playlist optimization modeled as a 0/1 Knapsack Problem. Song duration is treated as weight, while a score derived from popularity and energy is used as value. The dataset was obtained from Spotify Wrapped 2025 Top 50 Songs and Spotify All-Time Top 100 Songs, resulting in 31 candidate songs after preprocessing and filtering. Experiments were conducted on playlist durations of 30, 45, 60, 75, and 90 minutes. The results show that Dynamic Programming consistently achieved higher total scores than Greedy across all scenarios. For the 60-minute playlist, Dynamic Programming obtained a total score of 1897 compared to 1894 achieved by Greedy. However, Greedy required a lower execution time (4.244 ms) than Dynamic Programming (16.196 ms). The average optimality gap between the two methods was 1.89%, indicating that Greedy produced solutions that were close to the optimal solutions generated by Dynamic Programming while requiring less computation time.

Keywords: Spotify Playlist Optimization; Greedy Algorithm; Dynamic Programming; 0/1 Knapsack Problem; Music Recommendation.

Intisari: Spotify menyediakan metadata audio yang dapat dimanfaatkan untuk mendukung aktivitas olahraga seperti jogging. Penelitian ini membandingkan performa algoritma Greedy dan Dynamic Programming dalam optimasi playlist Spotify yang dimodelkan sebagai permasalahan 0/1 Knapsack. Durasi lagu direpresentasikan sebagai bobot, sedangkan skor yang dibentuk dari kombinasi popularitas dan energi digunakan sebagai nilai. Dataset diperoleh dari Spotify Wrapped 2025 Top 50 Songs dan Spotify All-Time Top 100 Songs yang setelah preprocessing dan filterisasi menghasilkan 31 lagu kandidat. Pengujian dilakukan pada lima skenario durasi playlist, yaitu 30, 45, 60, 75, dan 90 menit. Hasil penelitian menunjukkan bahwa Dynamic Programming secara konsisten menghasilkan total score yang lebih tinggi dibandingkan Greedy pada seluruh skenario pengujian. Pada playlist berdurasi 60 menit, Dynamic Programming memperoleh total score sebesar 1897, sedangkan Greedy memperoleh 1894. Namun, Greedy memiliki waktu eksekusi yang lebih rendah, yaitu 4,244 ms dibandingkan 16,196 ms pada Dynamic Programming. Nilai rata-rata optimality gap yang diperoleh sebesar 1,89%, menunjukkan bahwa solusi yang dihasilkan Greedy sangat mendekati solusi optimal Dynamic Programming dengan kebutuhan komputasi yang lebih rendah.

Kata Kunci: Optimasi Playlist Spotify; Algoritma Greedy; Dynamic Programming; 0/1 Knapsack Problem; Rekomendasi Musik.

1. Pendahuluan

Perkembangan teknologi digital telah mengubah cara masyarakat mengakses dan menikmati musik melalui layanan *music streaming* seperti Spotify. Platform tersebut menyediakan jutaan lagu yang dapat diakses secara instan serta dilengkapi dengan berbagai metadata audio, seperti tempo, energy, danceability, valence, dan popularity. Ketersediaan metadata tersebut membuka peluang pemanfaatan teknik komputasi dan optimasi dalam penyusunan playlist yang lebih sesuai dengan kebutuhan pengguna pada berbagai konteks aktivitas.

Dalam bidang olahraga, musik telah lama diketahui memiliki pengaruh terhadap performa fisik dan kondisi psikologis individu. Beberapa penelitian menunjukkan bahwa musik mampu meningkatkan motivasi, mengurangi persepsi kelelahan, memperbaiki suasana hati, serta membantu mempertahankan ritme gerakan selama aktivitas aerobik seperti jogging dan running [1], [2]. Selain itu, sistem musik yang disesuaikan dengan aktivitas fisik pengguna juga terbukti dapat meningkatkan keterlibatan dan pengalaman berolahraga secara keseluruhan [3].

Salah satu aktivitas olahraga yang paling populer adalah jogging karena relatif mudah dilakukan dan tidak memerlukan peralatan khusus. Dalam praktiknya, banyak pelari menggunakan playlist musik sebagai pendamping latihan. Namun, pemilihan lagu yang sesuai tidak selalu mudah mengingat jumlah lagu yang tersedia sangat besar dan setiap lagu memiliki karakteristik yang berbeda. Selain mempertimbangkan preferensi pengguna, playlist jogging juga perlu memperhatikan faktor seperti tingkat energi lagu, popularitas, dan durasi total playlist agar sesuai dengan target waktu latihan.

Berbagai penelitian pada bidang Music Information Retrieval (MIR) dan Music Recommendation System telah mengembangkan metode untuk menghasilkan playlist secara otomatis menggunakan pendekatan collaborative filtering, machine learning, knowledge graph, maupun reinforcement learning [4],[5],[6]. Meskipun demikian, sebagian besar penelitian tersebut berfokus pada personalisasi rekomendasi dan kontinuitas playlist, sedangkan aspek optimasi durasi playlist masih relatif jarang dibahas secara eksplisit.

Permasalahan pemilihan playlist dengan batasan durasi tertentu dapat dimodelkan sebagai permasalahan optimasi kombinatorial, khususnya 0/1 Knapsack Problem. Pada model ini, setiap lagu direpresentasikan sebagai item yang memiliki nilai (*value*) dan bobot (*weight*). Nilai lagu dapat dibentuk dari kombinasi atribut popularitas dan energi, sedangkan bobot direpresentasikan oleh durasi lagu. Dengan demikian, tujuan optimasi adalah memilih kombinasi lagu yang memberikan skor total maksimum tanpa melampaui batas waktu jogging yang telah ditentukan [7], [8]. Pendekatan ini memungkinkan proses penyusunan playlist dilakukan secara sistematis berdasarkan karakteristik lagu dan kendala durasi yang nyata dalam aktivitas olahraga.

Berbagai algoritma telah digunakan untuk menyelesaikan masalah knapsack, di antaranya Greedy dan Dynamic Programming. Algoritma Greedy memiliki keunggulan dalam efisiensi komputasi karena hanya memilih solusi terbaik pada setiap langkah berdasarkan kriteria tertentu. Namun, pendekatan ini tidak selalu menghasilkan solusi optimal global. Sebaliknya, Dynamic Programming mampu memperoleh solusi optimal dengan mengevaluasi seluruh kemungkinan kombinasi secara sistematis, meskipun membutuhkan sumber daya komputasi yang lebih besar [9], [10], [11].

Berdasarkan telaah literatur, masih terdapat keterbatasan penelitian yang menghubungkan optimasi playlist musik dengan pendekatan knapsack. Sebagian besar penelitian playlist berfokus pada rekomendasi musik, sedangkan penelitian knapsack umumnya menggunakan studi kasus alokasi sumber daya atau logistik. Selain itu, belum banyak penelitian yang secara khusus membandingkan algoritma Greedy dan Dynamic Programming dalam optimasi playlist Spotify berbasis atribut popularitas, energi, dan batasan durasi jogging. Kesenjangan penelitian tersebut menjadi dasar dilakukannya studi ini.

Oleh karena itu, penelitian ini bertujuan membandingkan performa algoritma Greedy dan Dynamic Programming dalam optimasi playlist Spotify untuk jogging yang dimodelkan sebagai permasalahan 0/1 Knapsack. Evaluasi dilakukan berdasarkan kualitas solusi, utilisasi kapasitas, waktu eksekusi, dan optimality gap pada berbagai skenario durasi jogging. Hasil penelitian diharapkan dapat memberikan kontribusi bagi pengembangan sistem rekomendasi playlist berbasis optimasi serta menjadi referensi penerapan algoritma strategi dalam permasalahan dunia nyata.

2. Tinjauan Pustaka

Bagian ini membahas konsep dan penelitian terdahulu yang relevan sebagai landasan teoritis dalam pengembangan sistem optimasi playlist Spotify untuk aktivitas jogging menggunakan pendekatan 0/1 Knapsack serta perbandingan algoritma Greedy dan Dynamic Programming.

2.1. Musik dan Performa Olahraga

Menurut Danso et al. (2024), sistem musik yang dipersonalisasi mampu meningkatkan keterlibatan pengguna dalam aktivitas fisik serta memberikan pengalaman olahraga yang lebih positif melalui penyesuaian karakteristik musik terhadap kebutuhan individu [1]. Penelitian oleh Delleli et al. (2023) menunjukkan bahwa musik sebelum aktivitas fisik dapat meningkatkan performa latihan sekaligus memperbaiki respons psikologis seperti motivasi dan suasana hati [2]. Selain itu, Brake et al. (2022) menemukan bahwa frekuensi ketukan (beat frequency) pada musik dapat membantu pelari mempertahankan running cadence yang lebih konsisten, sedangkan Wu et al. (2022) menunjukkan bahwa tempo musik memengaruhi persepsi kelelahan pada berbagai tingkat intensitas olahraga [3], [4]. Temuan-temuan tersebut menunjukkan bahwa karakteristik musik memiliki peran penting dalam mendukung aktivitas jogging sehingga relevan dijadikan dasar dalam pemilihan lagu pada penelitian ini.

2.2. Spotify dan Metadata Audio

Spotify merupakan salah satu platform *music streaming* yang menyediakan berbagai metadata audio seperti energy, danceability, tempo, valence, dan popularity yang dapat digunakan untuk merepresentasikan karakteristik sebuah lagu. Menurut Gabbolini dan Bridge (2024), metadata musik menjadi komponen penting dalam penelitian Music Information Retrieval (MIR) dan sistem rekomendasi playlist karena mampu menggambarkan atribut musikal yang relevan terhadap preferensi maupun konteks aktivitas pengguna [5]. Dalam penelitian ini, atribut energy dan popularity digunakan untuk membentuk nilai (value) lagu, sedangkan durasi lagu digunakan sebagai bobot (weight) dalam proses optimasi playlist.

2.3. Masalah Knapsack 0/1

Menurut Cacchiani et al. (2022), 0/1 Knapsack Problem merupakan salah satu permasalahan optimasi kombinatorial yang paling banyak digunakan dalam alokasi sumber daya dengan tujuan memperoleh nilai maksimum tanpa melebihi kapasitas yang tersedia [6], [7]. Setiap item hanya dapat dipilih satu kali atau tidak dipilih sama sekali. Secara matematis, permasalahan ini dirumuskan sebagai upaya memaksimalkan total nilai item dengan batasan kapasitas tertentu. Dalam penelitian ini, setiap lagu direpresentasikan sebagai item, di mana nilai lagu diperoleh dari kombinasi atribut energy dan popularity, sedangkan durasi lagu merepresentasikan bobot. Kapasitas knapsack ditentukan berdasarkan target durasi jogging yang diinginkan pengguna sehingga proses pemilihan playlist dapat dimodelkan sebagai masalah 0/1 Knapsack.

$$\text{Memaksimalkan } \sum_{i=1}^{n} v_i x_i \text{ dengan kendala } \sum_{i=1}^{n} w_i x_i \leq W, x_i \in \{0,1\}$$

2.4. Algoritma Greedy

Menurut Wang (2023), algoritma Greedy merupakan metode penyelesaian masalah yang membangun solusi secara bertahap dengan selalu memilih alternatif terbaik pada setiap langkah berdasarkan informasi lokal yang tersedia [14]. Pendekatan ini dikenal memiliki implementasi yang sederhana dan efisien dari sisi waktu komputasi. Chen (2022) menjelaskan bahwa pada permasalahan knapsack, strategi Greedy umumnya dilakukan dengan memilih item berdasarkan rasio nilai terhadap bobot (value-to-weight ratio) [11]. Dalam penelitian ini, pendekatan Greedy digunakan untuk memilih lagu dengan rasio skor terhadap durasi tertinggi secara berurutan hingga kapasitas waktu jogging terpenuhi. Meskipun cepat, metode ini tidak selalu menghasilkan solusi optimal global.

2.5. Dynamic Programming untuk 0/1 Knapsack

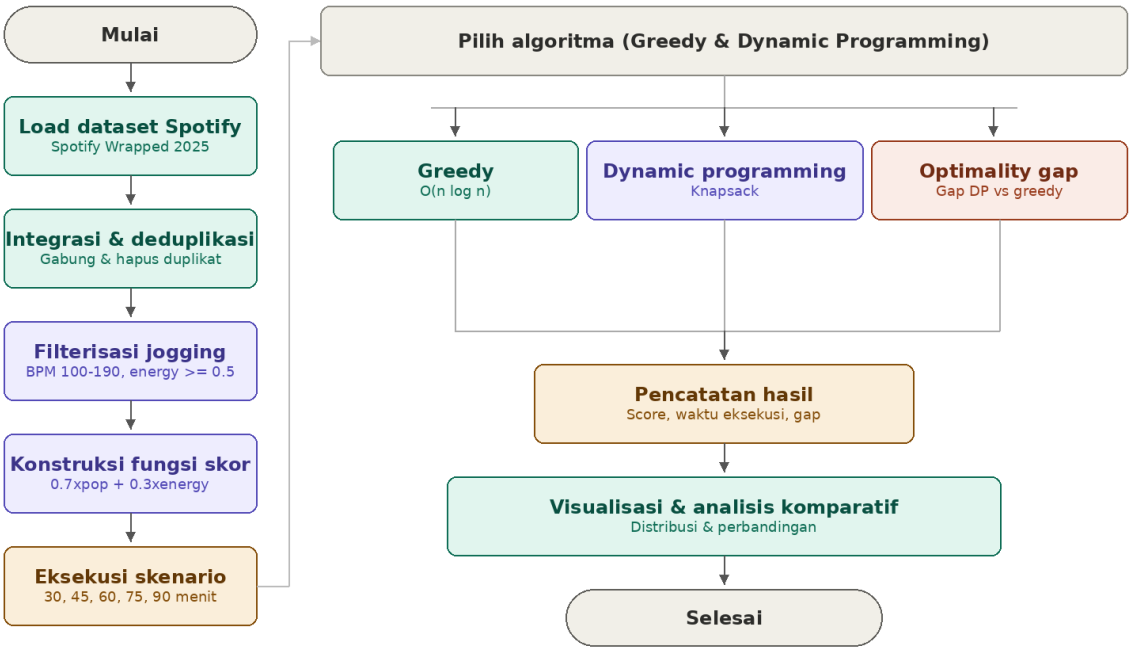
Menurut Zhang (2024), Dynamic Programming (DP) merupakan paradigma algoritmik yang menyelesaikan masalah kompleks dengan membaginya menjadi submasalah yang saling tumpang tindih dan menyimpan hasil perhitungan sebelumnya untuk menghindari komputasi berulang [10]. Pada kasus 0/1 Knapsack, DP mampu memperoleh solusi optimal dengan mengevaluasi seluruh kombinasi item secara sistematis melalui tabel penyimpanan nilai sementara. Wu (2023) menunjukkan bahwa pendekatan Dynamic Programming secara konsisten menghasilkan solusi yang lebih optimal dibandingkan Greedy pada permasalahan knapsack karena mempertimbangkan seluruh kemungkinan kombinasi item [11]. Oleh karena itu, dalam penelitian ini Dynamic Programming digunakan sebagai metode pembanding sekaligus acuan solusi optimal.

2.6. Sistem Rekomendasi Playlist Berbasis Konteks Aktivitas

Menurut Sakurai et al. (2022), sistem rekomendasi playlist modern tidak lagi hanya mempertimbangkan preferensi pengguna, tetapi juga konteks aktivitas yang sedang dilakukan melalui pendekatan *context-aware recommendation* [12]. Penelitian oleh Bendada et al. (2023) menunjukkan bahwa *automatic playlist continuation* pada layanan streaming dapat meningkatkan relevansi playlist dengan memanfaatkan karakteristik lagu secara simultan [13]. Selain itu, Tomasi et al. (2023) mengembangkan pendekatan generasi playlist otomatis menggunakan reinforcement learning untuk menghasilkan urutan lagu yang lebih sesuai dengan kebutuhan pengguna [14]. Meskipun berbagai penelitian telah mengembangkan metode rekomendasi playlist yang semakin canggih, kajian yang mengombinasikan optimasi playlist berbasis atribut lagu dengan pendekatan 0/1 Knapsack serta membandingkan algoritma Greedy dan Dynamic Programming masih relatif terbatas. Oleh karena itu, penelitian ini berfokus pada perbandingan kedua algoritma tersebut dalam menghasilkan playlist Spotify yang optimal untuk aktivitas jogging [15].

3. Metode Penelitian

Penelitian ini menggunakan pendekatan optimasi kombinatorial dengan memodelkan pemilihan playlist Spotify untuk aktivitas jogging sebagai permasalahan 0/1 Knapsack. Untuk menyelesaikan permasalahan tersebut, dilakukan perbandingan antara algoritma Greedy dan Dynamic Programming dalam memilih kombinasi lagu yang memberikan nilai maksimum tanpa melebihi batas durasi yang ditentukan. Tahapan penelitian dimulai dari pengumpulan dan pengolahan data lagu, dilanjutkan dengan pembentukan fungsi skor, seleksi kandidat lagu yang sesuai untuk aktivitas jogging, implementasi kedua algoritma, serta evaluasi hasil berdasarkan kualitas solusi dan efisiensi komputasi. Alur penelitian secara keseluruhan ditunjukkan pada Gambar 1.



Gambar 1. Alur Penelitian Optimasi Playlist Spotify Menggunakan Algoritma Greedy dan DP

3.1. Dataset dan Sumber Data

Penelitian ini menggunakan data sekunder yang diperoleh dari dataset publik Spotify Wrapped 2025 Top Songs and Artists yang tersedia pada platform Kaggle. Dari dataset tersebut, penelitian memanfaatkan dua berkas data lagu, yaitu Spotify Wrapped 2025 Top 50 Songs yang berisi 50 lagu dengan jumlah streaming tertinggi sepanjang tahun 2025 dan Spotify All-Time Top 100 Songs yang berisi 100 lagu dengan akumulasi streaming tertinggi sepanjang masa. Kedua dataset digabungkan dan dilakukan deduplikasi untuk membentuk satu katalog lagu yang lebih komprehensif. Atribut yang digunakan dalam penelitian meliputi song title, artist, bpm, energy, duration seconds, streams 2025 billions, dan total streams billions. Atribut-atribut tersebut digunakan sebagai dasar dalam proses penyaringan lagu jogging, pembentukan fungsi skor, serta optimasi playlist menggunakan algoritma Greedy dan Dynamic Programming.

3.2. Preprocessing dan Filterisasi

Tahap preprocessing dilakukan untuk memastikan kualitas data sebelum proses optimasi. Langkah yang dilakukan meliputi penggabungan kedua dataset, penghapusan data duplikat berdasarkan judul lagu dan artis, pembersihan data yang memiliki nilai durasi kosong atau tidak valid, serta penyesuaian tipe data pada atribut yang digunakan dalam analisis. Selanjutnya dilakukan perhitungan skor lagu berdasarkan kombinasi atribut popularitas dan energi, kemudian dilakukan filterisasi untuk memilih lagu yang sesuai dengan karakteristik jogging. Kriteria yang digunakan adalah BPM antara 100–190 dan nilai energy minimal 0,5 sehingga hanya lagu yang memiliki tempo dan tingkat energi yang mendukung aktivitas jogging yang dipertahankan sebagai kandidat playlist. Dataset hasil filterisasi inilah yang digunakan sebagai masukan bagi algoritma Greedy dan Dynamic Programming pada tahap optimasi.

3.3. Konstruksi Fungsi Skor

Setiap lagu diberikan nilai (value) yang merepresentasikan tingkat kelayakan lagu untuk dimasukkan ke dalam playlist jogging. Nilai tersebut dihitung berdasarkan kombinasi atribut popularitas dan energi lagu. Popularitas diperoleh dari akumulasi jumlah streaming pada dataset Spotify Wrapped 2025 dan Spotify All-Time Top 100 Songs, sedangkan energi merepresentasikan intensitas musikal yang relevan untuk aktivitas fisik. Perhitungan skor dilakukan menggunakan Persamaan (1) dan Persamaan (2).

$$popularity\_score = (streams\_2025\_billions + total\_streams\_billions) \times 100$$

$$final\_score = 0{,}7 \times popularity\_score + 0{,}3 \times (energy \times 100)$$

Pada penelitian ini, atribut popularitas diberikan bobot sebesar 70%, sedangkan atribut energi diberikan bobot sebesar 30%. Pembobotan ini didasarkan pada asumsi bahwa lagu yang memiliki tingkat popularitas tinggi cenderung lebih dikenal, lebih sering didengarkan, dan memiliki peluang lebih besar untuk diterima oleh sebagian besar pengguna. Selain itu, dataset yang digunakan berasal dari kumpulan lagu-lagu terpopuler Spotify, sehingga atribut popularitas dianggap sebagai indikator utama dalam merepresentasikan preferensi pengguna. Sementara itu, atribut energi tetap dipertimbangkan karena berhubungan dengan intensitas musikal yang dapat mendukung aktivitas fisik seperti jogging melalui ritme dan dinamika lagu yang lebih tinggi [1], [2]. Oleh karena itu, popularitas ditempatkan sebagai faktor utama dengan kontribusi sebesar 70%, sedangkan energi berfungsi sebagai faktor pendukung dengan kontribusi sebesar 30% untuk memastikan lagu yang dipilih tidak hanya populer, tetapi juga sesuai dengan karakteristik aktivitas jogging. Nilai FinalScore digunakan sebagai value (v_i) dalam formulasi 0/1 Knapsack, sedangkan atribut duration_seconds digunakan sebagai weight (w_i).

Pemilihan bobot 70:30 pada penelitian ini bersifat heuristik dan ditetapkan berdasarkan karakteristik dataset serta tujuan penelitian yang lebih menekankan tingkat popularitas lagu dibandingkan atribut audio lainnya. Analisis sensitivitas terhadap variasi bobot dapat menjadi agenda penelitian lanjutan untuk mengevaluasi pengaruh perubahan bobot terhadap hasil optimasi playlist.

3.4. Formulasi Masalah

Permasalahan optimasi playlist pada penelitian ini dimodelkan sebagai 0/1 Knapsack Problem, di mana setiap lagu direpresentasikan sebagai item yang memiliki nilai (value) dan bobot (weight). Nilai lagu dinyatakan sebagai *final score* hasil kombinasi atribut popularitas dan energi, sedangkan bobot direpresentasikan oleh durasi lagu dalam satuan detik. Tujuan optimasi adalah memilih kombinasi lagu yang menghasilkan total skor maksimum tanpa melebihi batas durasi jogging yang telah ditentukan. Secara matematis, formulasi masalah dinyatakan sebagai berikut:

Maksimalkan $\sum_{i=1}^{n} v_i x_i$

dengan kendala:

$$\sum_{i=1}^{n} w_i x_i \leq W,$$

$$xi \in \{0,1\}$$

dengan v_i menyatakan nilai lagu ke-i, w_i menyatakan durasi lagu ke-i, W merupakan kapasitas total durasi playlist, dan x_i adalah variabel keputusan yang bernilai 1 jika lagu dipilih dan 0 jika tidak dipilih. Pada penelitian ini, kapasitas W divariasikan ke dalam lima skenario durasi jogging, yaitu 30, 45, 60, 75, dan 90 menit atau setara dengan 1.800, 2.700, 3.600, 4.500, dan 5.400 detik.

3.5. Implementasi Algoritma Greedy

Algoritma Greedy digunakan untuk menyelesaikan permasalahan optimasi playlist dengan memilih lagu berdasarkan rasio nilai terhadap durasi (*value-to-weight ratio*). Nilai rasio setiap lagu dihitung menggunakan perbandingan antara final score dan duration_seconds, kemudian seluruh lagu diurutkan secara menurun berdasarkan rasio tersebut. Proses pemilihan dilakukan secara bertahap dengan menambahkan lagu berasio tertinggi ke dalam playlist selama total durasi yang terpilih belum melebihi kapasitas waktu jogging yang ditentukan. Pada penelitian ini, kapasitas durasi divariasikan pada lima skenario, yaitu 30, 45, 60, 75, dan 90 menit. Pendekatan Greedy dipilih karena memiliki proses komputasi yang sederhana dan cepat, dengan kompleksitas waktu sebesar O(nlogn) yang didominasi oleh proses pengurutan data.

Kemudian tampilkan persamaan:

$$r_i = \frac{v_i}{w_i}$$

dengan:
r_i = rasio lagu ke-i
v_i = final score lagu ke-i
w_i = durasi lagu ke-i (detik)

3.6. Implementasi Dynamic Programming

Algoritma Dynamic Programming digunakan untuk memperoleh solusi optimal pada permasalahan 0/1 Knapsack dengan mengevaluasi kombinasi lagu secara sistematis. Implementasi dilakukan menggunakan pendekatan *bottom-up*, di mana setiap lagu diproses berdasarkan nilai (final score) dan durasinya untuk memperbarui nilai maksimum yang dapat diperoleh pada setiap kapasitas playlist. Untuk setiap lagu ke-I dengan nilai vi dan durasi wi, dilakukan iterasi mundur pada kapasitas w sehingga setiap lagu satu kali sesuai karakteristik 01 Knapsack. Persamaan pembaruan nilai dinyatakan sebagai:

$$dp[w] = max(dp[w], dp[w - wi] + vi)$$

dengan dp[w] menyatakan nilai maksimum yang dapat diperoleh pada kapasitas w. Setelah proses optimasi selesai, lagu-lagu yang terpilih direkonstruksi menggunakan mekanisme *backtracking* melalui matriks keep, sehingga diperoleh kombinasi lagu yang membentuk playlist optimal tanpa melebihi batas durasi yang ditentukan. Pendekatan ini menjamin diperolehnya solusi optimal untuk setiap skenario durasi jogging yang diuji.

3.7. Evaluasi dan Pengukuran Performa

Perbandingan algoritma Greedy dan Dynamic Programming dilakukan berdasarkan tiga metrik evaluasi. Pertama, Total Score, yaitu jumlah final score dari seluruh lagu yang terpilih dalam playlist sebagai indikator kualitas solusi. Kedua, Waktu Eksekusi, yaitu waktu yang dibutuhkan algoritma untuk menghasilkan solusi, yang diukur menggunakan modul timeit pada Python dengan 30 kali pengulangan (run) untuk memperoleh nilai rata-rata dan standar deviasi yang lebih stabil. Ketiga, Optimality Gap, yaitu persentase selisih kualitas solusi Greedy terhadap solusi optimal yang dihasilkan Dynamic Programming. Nilai optimality gap dihitung menggunakan persamaan berikut:

$$Gap(\%) = \frac{Score_{DP} - Score_{Greedy}}{Score_{DP}} \times 100$$

dengan $Score_{DP}$ menyatakan total skor yang diperoleh Dynamic Programming dan $Score_{Greedy}$ menyatakan total skor yang diperoleh algoritma Greedy. Semakin kecil nilai optimality gap, semakin mendekati

kualitas solusi Greedy terhadap solusi optimal. Selain itu, rasio waktu eksekusi kedua algoritma juga dihitung untuk mengukur tingkat efisiensi komputasi antara pendekatan Greedy dan Dynamic Programming.

4. Hasil dan Pembahasan

Bagian ini menyajikan hasil implementasi dan evaluasi algoritma Greedy dan Dynamic Programming dalam optimasi pemilihan playlist musik untuk jogging. Pembahasan diawali dengan proses persiapan data melalui penggabungan dataset, preprocessing, dan filterisasi lagu berdasarkan kriteria BPM 100–190 serta energy ≥ 0,5. Selanjutnya dilakukan analisis hasil optimasi pada berbagai skenario durasi jogging, meliputi perbandingan total score, waktu eksekusi, optimality gap, dan tingkat kesamaan (overlap) playlist yang dihasilkan oleh kedua algoritma. Hasil yang diperoleh kemudian dianalisis untuk mengevaluasi trade-off antara kualitas solusi dan efisiensi komputasi pada masing-masing metode.

4.1. Persiapan Data dan Seleksi Lagu Jogging

Dataset awal terdiri atas 150 lagu yang berasal dari penggabungan dataset Spotify Wrapped 2025 Top 50 Songs dan Spotify All-Time Top 100 Songs. Setelah dilakukan proses deduplikasi berdasarkan kombinasi judul lagu dan nama artis, diperoleh 135 lagu unik. Selanjutnya dilakukan seleksi atribut yang digunakan dalam penelitian, yaitu BPM, energy, duration_seconds, streams_2025_billions, dan total_streams_billions, serta perhitungan skor akhir (final score) menggunakan Persamaan (2).

Hasil proses tersebut menghasilkan 50 lagu yang memiliki atribut lengkap dan dapat digunakan untuk analisis lebih lanjut. Selanjutnya dilakukan filterisasi berdasarkan karakteristik jogging, yaitu BPM antara 100–190 dan nilai energy minimal 0,5. Kriteria tersebut diterapkan untuk memastikan lagu yang dipilih memiliki tempo dan tingkat energi yang sesuai untuk aktivitas jogging. Hasil filterisasi menghasilkan 31 lagu yang memenuhi seluruh kriteria dan digunakan sebagai kandidat playlist pada tahap optimasi. Karakteristik dataset hasil filterisasi ditunjukkan pada Tabel 1.

Tabel 1. Karakteristik Dataset Hasil Filterisasi

Parameter	**Nilai**
Jumlah lagu	31
BPM rata-rata	122,1
Energy rata-rata	0,721
Final score rata-rata	89,0

4.2. Hasil Optimasi Menggunakan Algoritma Greedy

Pengujian algoritma Greedy dilakukan pada kapasitas playlist sebesar 3600 detik (60 menit). Algoritma ini memilih lagu berdasarkan rasio final score terhadap durasi (*score-per-second*) secara menurun hingga kapasitas playlist tidak memungkinkan penambahan lagu berikutnya.

Hasil optimasi menunjukkan bahwa algoritma Greedy menghasilkan playlist yang terdiri atas 19 lagu dengan total durasi 3582 detik atau sekitar 59,7 menit. Playlist yang dihasilkan mampu memanfaatkan 99,5% dari kapasitas yang tersedia dan memperoleh total score sebesar 1894. Ringkasan hasil optimasi menggunakan algoritma Greedy ditunjukkan pada Tabel 2.

Tabel 2. Hasil Optimasi Algoritma Greedy

Parameter	**Nilai**
Jumlah lagu	19
Total durasi	3582 detik
Utilisasi kapasitas	99,5%
Total score	1894

Hasil tersebut menunjukkan bahwa algoritma Greedy mampu menghasilkan playlist dengan tingkat pemanfaatan kapasitas yang sangat tinggi. Selain itu, proses seleksi berlangsung secara efisien karena hanya

memerlukan pengurutan berdasarkan rasio score-per-second dan pemilihan lagu secara berurutan. Namun, karena keputusan pemilihan dilakukan berdasarkan informasi lokal pada setiap langkah, kombinasi lagu yang dihasilkan belum tentu memberikan total score maksimum secara global.

4.3. Hasil Optimasi Menggunakan Dynamic Programming

Pengujian algoritma Dynamic Programming dilakukan pada kapasitas playlist sebesar 3600 detik (60 menit). Algoritma ini memodelkan permasalahan sebagai kasus 0/1 Knapsack dan mengevaluasi kombinasi lagu secara sistematis untuk memperoleh total score maksimum tanpa melampaui kapasitas playlist yang ditentukan.

Hasil optimasi menunjukkan bahwa algoritma Dynamic Programming menghasilkan playlist yang terdiri atas 19 lagu dengan total durasi 3593 detik atau sekitar 59,9 menit. Playlist yang dihasilkan mampu memanfaatkan 99,8% dari kapasitas yang tersedia dan memperoleh total score sebesar 1897. Ringkasan hasil optimasi menggunakan algoritma Dynamic Programming ditunjukkan pada Tabel 3.

Tabel 3. Hasil Optimasi Algoritma Dynamic Programming

Parameter	**Nilai**
Jumlah lagu	19
Total durasi	3593 detik
Utilisasi kapasitas	99,8%
Total skor	1897

Hasil tersebut menunjukkan bahwa algoritma Dynamic Programming mampu menghasilkan kombinasi lagu dengan total score yang maksimum sesuai formulasi 0/1 Knapsack. Selain memiliki tingkat pemanfaatan kapasitas yang sangat tinggi, pendekatan ini menjamin diperolehnya solusi optimal karena seluruh kemungkinan kombinasi lagu dievaluasi secara sistematis selama proses optimasi.

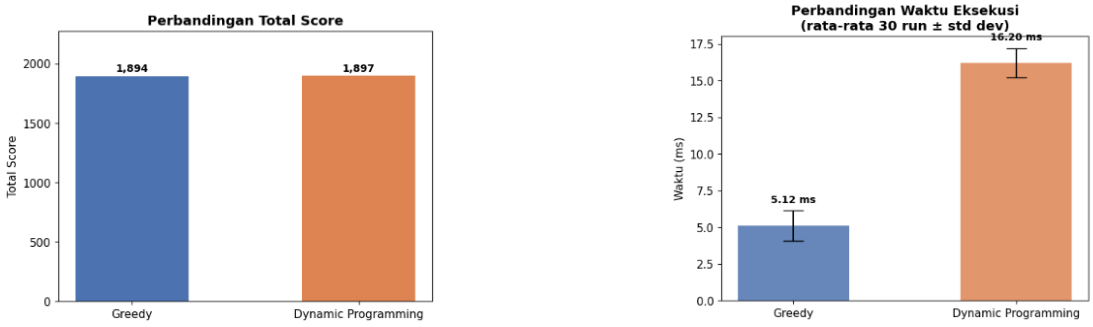
4.4. Perbandingan Kinerja Algoritma

Untuk mengetahui perbedaan performa kedua metode, dilakukan perbandingan berdasarkan jumlah lagu terpilih, total durasi playlist, utilisasi kapasitas, total score, dan waktu eksekusi pada kapasitas playlist 3600 detik (60 menit). Hasil perbandingan ditunjukkan pada Tabel 4.

Tabel 4. Perbandingan Kinerja Algoritma Greedy dan Dynamic Programming

Metode	**Jumlah Lagu**	**Durasi (detik)**	**Utilisasi (%)**	**Total Skor**	**Waktu Eksekusi (ms)**
Greedy	19	3582	99,5	1894	4,244
Dynamic Programming	19	3593	99,8	1897	16,196

Berdasarkan Tabel 4, kedua algoritma menghasilkan jumlah lagu yang sama, yaitu 19 lagu. Meskipun demikian, Dynamic Programming mampu memanfaatkan kapasitas playlist secara lebih optimal dan menghasilkan total score yang lebih tinggi dibandingkan Greedy. Hasil ini menunjukkan bahwa evaluasi seluruh kombinasi solusi pada Dynamic Programming memberikan kualitas playlist yang lebih baik dibandingkan pendekatan pemilihan lokal yang digunakan oleh Greedy. Dari sisi efisiensi komputasi, Greedy menunjukkan waktu eksekusi yang jauh lebih rendah sehingga lebih sesuai untuk kebutuhan rekomendasi secara *real-time*.



(a) **(b)**

Gambar 2. Perbandingan Total Score dan Waktu Eksekusi Greedy dan DP pada Kapasitas 60 Menit

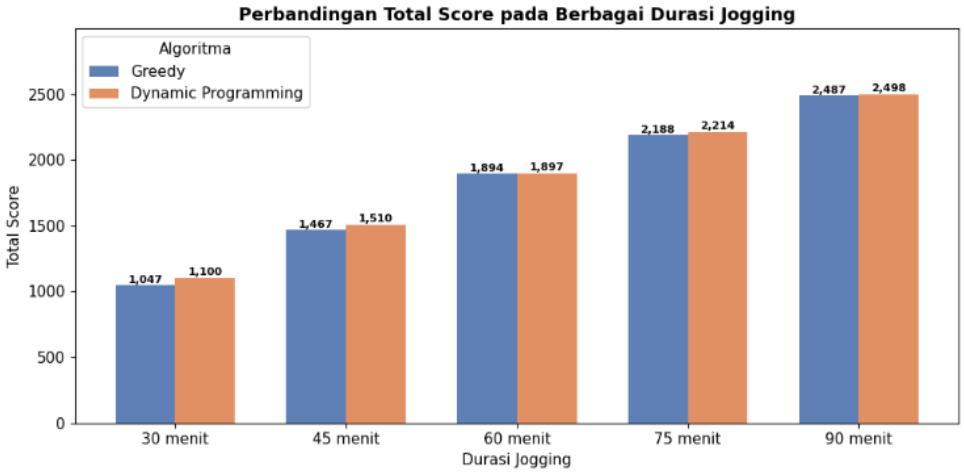
Visualisasi pada Gambar 2(a) menunjukkan bahwa Dynamic Programming menghasilkan total score yang sedikit lebih tinggi dibandingkan Greedy. Meskipun peningkatan skor yang diperoleh relatif kecil, metode ini tetap menghasilkan solusi terbaik pada kapasitas yang diuji. Sebaliknya, Gambar 2(b) memperlihatkan bahwa waktu eksekusi Dynamic Programming lebih besar dibandingkan Greedy. Kondisi ini sejalan dengan karakteristik teoritis kedua algoritma, di mana Greedy memiliki kompleksitas waktu O(n log n), sedangkan Dynamic Programming memiliki kompleksitas pseudo-polynomial O(n × W). Dengan demikian, terdapat trade-off antara kualitas solusi dan efisiensi komputasi dalam pemilihan metode optimasi playlist.

4.5. Eksperimen Multi-Durasi

Untuk mengevaluasi konsistensi performa kedua algoritma pada berbagai kondisi, pengujian dilakukan pada lima skenario durasi jogging, yaitu 30, 45, 60, 75, dan 90 menit. Setiap durasi jogging dikonversi menjadi kapasitas knapsack dalam satuan detik yang digunakan sebagai batas maksimum total durasi playlist. Hasil eksperimen ditunjukkan pada Tabel 5.

Tabel 5. Hasil Eksperimen pada Berbagai Durasi Jogging

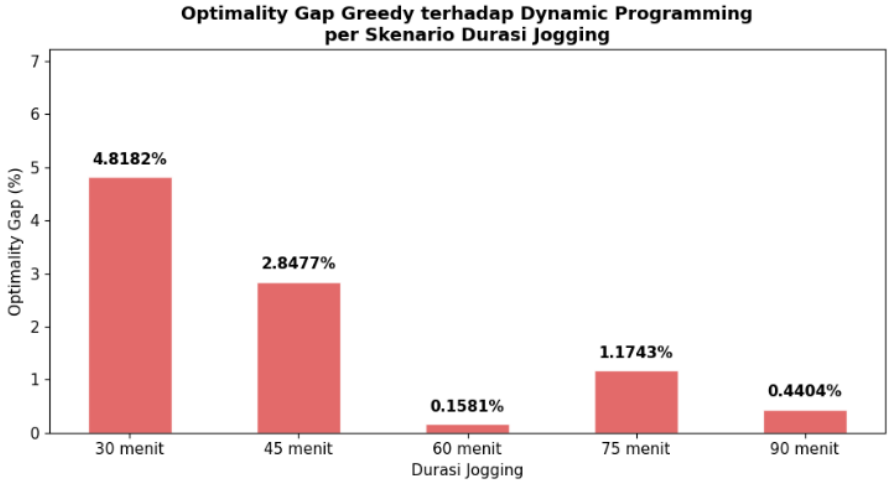
Durasi	Greedy	Dynamic Programming	Optimality Gap
30 menit	1047	1100	4,818%
45 menit	1467	1510	2,848%
60 menit	1894	1897	0,158%
75 menit	2188	2214	1,174%
90 menit	2487	2498	0,440%
Rata-rata	-	-	1,89%



Gambar 3. Perbandingan Total Score pada Berbagai Durasi Jogging

Hasil pada Tabel 5 dan Gambar 3 menunjukkan bahwa Dynamic Programming secara konsisten menghasilkan total score yang lebih tinggi dibandingkan Greedy pada seluruh skenario durasi yang diuji, menunjukkan kemampuannya dalam menemukan kombinasi lagu yang lebih optimal. Seiring bertambahnya kapasitas playlist, kedua algoritma sama-sama menghasilkan peningkatan total score karena lebih banyak lagu dapat dimasukkan ke dalam playlist.

Selain itu, nilai optimality gap menunjukkan bahwa kualitas solusi Greedy relatif mendekati solusi optimal yang diperoleh Dynamic Programming. Gap terbesar terjadi pada durasi 30 menit, yaitu sebesar 4,818%, yang menunjukkan bahwa keterbatasan kapasitas menyebabkan strategi pemilihan lokal pada Greedy lebih berpotensi menghasilkan solusi yang kurang optimal. Sebaliknya, gap terkecil terjadi pada durasi 60 menit sebesar 0,158%, menandakan bahwa solusi Greedy hampir identik dengan solusi optimal Dynamic Programming pada kapasitas tersebut.



Gambar 4. Optimality Gap Greedy terhadap Dynamic Programming

Berdasarkan hasil eksperimen, nilai optimality gap berfluktuasi pada setiap skenario durasi dan tidak menunjukkan pola peningkatan maupun penurunan yang konsisten. Namun demikian, seluruh nilai gap berada di bawah 5%, yang menunjukkan bahwa algoritma Greedy mampu menghasilkan solusi dengan kualitas yang sangat mendekati optimal, meskipun menggunakan waktu komputasi yang jauh lebih rendah dibandingkan Dynamic Programming.

4.6. Analisis Overlap Playlist

Untuk mengetahui tingkat kesamaan solusi yang dihasilkan oleh kedua algoritma, dilakukan analisis overlap terhadap lagu-lagu yang terpilih pada kapasitas playlist 3600 detik (60 menit). Analisis ini bertujuan untuk mengidentifikasi sejauh mana algoritma Greedy dan Dynamic Programming memilih lagu yang sama dalam membentuk playlist.

Tabel 6. Hasil Analisis Overlap Playlist pada Kapasitas 60 Menit

Kategori	Jumlah Lagu	Persentase
Dipilih keduanya	18	58,1%
Hanya Greedy	1	3,2%
Hanya Dynamic Programming	1	3,2%
Tidak dipilih	11	35,5%

Hasil pada Tabel 6 menunjukkan bahwa sebagian besar lagu yang dipilih oleh algoritma Greedy juga dipilih oleh Dynamic Programming. Dari total 31 lagu kandidat, sebanyak 18 lagu (58,1%) dipilih oleh kedua algoritma secara bersamaan. Sementara itu, hanya terdapat satu lagu yang dipilih secara eksklusif oleh Greedy dan satu lagu yang dipilih secara eksklusif oleh Dynamic Programming.

Tingginya jumlah lagu yang dipilih secara bersamaan menunjukkan bahwa kedua algoritma memiliki kecenderungan yang serupa dalam mengidentifikasi lagu-lagu dengan nilai yang tinggi dan sesuai dengan batasan durasi playlist. Perbedaan kombinasi lagu yang sangat kecil tersebut menyebabkan total score yang dihasilkan kedua algoritma juga relatif berdekatan, yaitu 1894 untuk Greedy dan 1897 untuk Dynamic Programming.

4.7. Pembahasan

Berdasarkan hasil pengujian, algoritma Dynamic Programming menghasilkan solusi terbaik pada permasalahan optimasi playlist musik untuk jogging dengan total score yang lebih tinggi dibandingkan algoritma Greedy. Hal ini menunjukkan bahwa pendekatan Dynamic Programming mampu menemukan kombinasi lagu yang lebih optimal sesuai formulasi 0/1 Knapsack.

Di sisi lain, algoritma Greedy menunjukkan keunggulan dari sisi efisiensi komputasi. Waktu eksekusi Greedy lebih cepat dibandingkan Dynamic Programming, sementara kualitas solusi yang dihasilkan tetap mendekati optimal. Hal ini terlihat dari nilai optimality gap yang relatif kecil pada seluruh skenario durasi yang diuji.

Hasil penelitian menunjukkan adanya trade-off antara kualitas solusi dan waktu komputasi. Dynamic Programming lebih sesuai digunakan ketika optimalitas solusi menjadi prioritas utama, sedangkan Greedy lebih cocok untuk sistem rekomendasi yang memerlukan proses komputasi cepat. Dengan demikian, kedua algoritma memiliki keunggulan masing-masing dan dapat dipilih sesuai kebutuhan implementasi..

5. Kesimpulan

Penelitian ini telah membandingkan algoritma Greedy dan Dynamic Programming dalam optimasi pemilihan playlist musik untuk jogging yang dimodelkan sebagai permasalahan 0/1 Knapsack. Berdasarkan hasil preprocessing dan filterisasi dataset Spotify Wrapped 2025, diperoleh 31 lagu yang memenuhi kriteria jogging, yaitu BPM 100–190 dan nilai energy minimal 0,5. Lagu-lagu tersebut kemudian digunakan sebagai kandidat playlist pada lima skenario durasi jogging, yaitu 30, 45, 60, 75, dan 90 menit.

Hasil pengujian menunjukkan bahwa algoritma Dynamic Programming menghasilkan total score tertinggi pada seluruh skenario durasi yang diuji. Pada kapasitas 60 menit, Dynamic Programming memperoleh total score sebesar 1897 dengan utilisasi kapasitas 99,8%, sedangkan Greedy memperoleh total score sebesar 1894 dengan utilisasi kapasitas 99,5%. Analisis overlap playlist juga menunjukkan bahwa kedua algoritma memiliki tingkat kesamaan yang tinggi, dengan 18 dari 31 lagu kandidat dipilih secara bersamaan.

Dari sisi efisiensi komputasi, algoritma Greedy menunjukkan kinerja yang lebih baik dengan waktu eksekusi sekitar 4,244 ms, sedangkan Dynamic Programming memerlukan waktu sekitar 16,196 ms. Meskipun demikian, nilai optimality gap Greedy terhadap Dynamic Programming pada seluruh skenario pengujian berada di bawah 5%, yang menunjukkan bahwa kualitas solusi Greedy tetap sangat mendekati solusi optimal.

Berdasarkan hasil tersebut, Dynamic Programming lebih sesuai digunakan ketika optimalitas solusi menjadi prioritas utama. Namun, apabila mempertimbangkan keseimbangan antara kualitas solusi dan efisiensi komputasi, algoritma Greedy menjadi alternatif yang lebih praktis untuk diterapkan pada sistem rekomendasi playlist jogging secara real-time karena mampu menghasilkan solusi yang mendekati optimal dengan waktu komputasi yang lebih rendah.

Sebagai saran untuk penelitian selanjutnya, model optimasi dapat dikembangkan dengan menambahkan atribut audio lainnya seperti danceability, valence, dan acousticness, serta mempertimbangkan preferensi pengguna agar playlist yang dihasilkan lebih personal dan adaptif terhadap kebutuhan masing-masing pengguna.

Daftar Pustaka

[1] K. S. Park, D. M. Williams, and J. L. Etnier, "Exploring the use of music to promote physical activity: From the viewpoint of psychological hedonism," *Front. Psychol.*, vol. 14, p. 1021825, Jan. 2023, doi: 10.3389/fpsyg.2023.1021825.

[2] S. Delleli *et al.*, "The effects of pre-task music on exercise performance and associated psycho-physiological responses: a systematic review with multilevel meta-analysis of controlled studies," *Front. Psychol.*, vol. 14, p. 1293783, Nov. 2023, doi: 10.3389/fpsyg.2023.1293783.

[3] A. Danso *et al.*, "Personalized Interactive Music Systems for Physical Activity and Exercise: Exploratory Systematic Review and Meta-Analysis," *JMIR Hum. Factors*, vol. 12, pp. e70372–e70372, Sep. 2025, doi: 10.2196/70372.

[4] G. Gabbolini and D. Bridge, "Surveying More Than Two Decades of Music Information Retrieval Research on Playlists," *ACM Trans. Intell. Syst. Technol.*, vol. 15, no. 6, pp. 1–68, Dec. 2024, doi: 10.1145/3688398.

[5] F. Tomasi, J. Cauteruccio, S. Kanoria, K. Ciosek, M. Rinaldi, and Z. Dai, "Automatic Music Playlist Generation via Simulation-based Reinforcement Learning," in *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, Long Beach CA USA: ACM, Aug. 2023, pp. 4948–4957. doi: 10.1145/3580305.3599777.

[6] W. Bendada, G. Salha-Galvan, T. Bouabça, and T. Cazenave, "A Scalable Framework for Automatic Playlist Continuation on Music Streaming Services," in *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, Taipei Taiwan: ACM, Jul. 2023, pp. 464–474. doi: 10.1145/3539618.3591628.

[7] V. Cacchiani, M. Iori, A. Locatelli, and S. Martello, "Knapsack problems — An overview of recent advances. Part I: Single knapsack problems," *Comput. Oper. Res.*, vol. 143, p. 105692, Jul. 2022, doi: 10.1016/j.cor.2021.105692.

[8] V. Cacchiani, M. Iori, A. Locatelli, and S. Martello, "Knapsack problems — An overview of recent advances. Part II: Multiple, multidimensional, and quadratic knapsack problems," *Comput. Oper. Res.*, vol. 143, p. 105693, Jul. 2022, doi: 10.1016/j.cor.2021.105693.

[9] J. R. Lee, "Spectral Hypergraph Sparsification via Chaining," in *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, Orlando FL USA: ACM, Jun. 2023, pp. 207–218. doi: 10.1145/3564246.3585165.

[10] Y. Wu, "Comparison of dynamic programming and greedy algorithms and the way to solve 0-1 knapsack problem," *Appl. Comput. Eng.*, vol. 5, no. 1, pp. 631–636, May 2023, doi: 10.54254/2755-2721/5/20230666.

[11] X. Chen, "A Comparison of Greedy Algorithm and Dynamic Programming Algorithm," *SHS Web Conf.*, vol. 144, p. 03009, 2022, doi: 10.1051/shsconf/202214403009.

[12] M. Te Brake, N. Stolwijk, B. Staal, and B. Van Hooren, "Using beat frequency in music to adjust running cadence in recreational runners: A randomized multiple baseline design," *Eur. J. Sport Sci.*, vol. 23, no. 3, pp. 345–354, Mar. 2023, doi: 10.1080/17461391.2022.2042398.

[13] J. Wu, L. Zhang, H. Yang, C. Lu, L. Jiang, and Y. Chen, "The Effect of Music Tempo on Fatigue Perception at Different Exercise Intensities," *Int. J. Environ. Res. Public. Health*, vol. 19, no. 7, p. 3869, Mar. 2022, doi: 10.3390/ijerph19073869.

[14] Y. Wang, "Review on greedy algorithm," *Theor. Nat. Sci.*, vol. 14, no. 1, pp. 233–239, Nov. 2023, doi: 10.54254/2753-8818/14/20241041.

[15] Y. Zhang, "A survey of dynamic programming algorithms," *Appl. Comput. Eng.*, vol. 35, no. 1, pp. 183–189, Feb. 2024, doi: 10.54254/2755-2721/35/20230392.