Maliki Interdisciplinary Journal (MIJ) eISSN: 3024-8140
Volume 3, Issue Desember, 2025 pp. 1270-1276 http://urj.uin-malang.ac.id/index.php/mij/index

Implementasi algoritma brute force dalam pencarian menu pada aplikasi pemesanan coffee

Moh. Roghil Affan Ramadani
Program Studi Teknik Informatika, Universitas Islam Negeri Maulana Malik Ibrahim
e-mail: 240605110129@student.uin-malang.ac.id

Kata Kunci:
Algoritma Brute Force, pencarian menu, aplikasi pemesanan kopi, efisiensi pencarian
Keywords:
Brute Force Algorithm, menu search, coffee ordering application, search efficiency

A B S T R A K

Perkembangan teknologi digital telah memengaruhi industri kuliner, termasuk dalam sistem pemesanan makanan dan minuman melalui aplikasi mobile. Salah satu fitur penting pada aplikasi pemesanan kopi adalah sistem pencarian menu yang memudahkan pelanggan menemukan produk secara cepat dan akurat. Penelitian ini bertujuan untuk mengimplementasikan dan menganalisis efektivitas algoritma Brute Force dalam pencarian menu pada aplikasi pemesanan kopi. Algoritma Brute Force bekerja dengan cara membandingkan input pencarian pengguna dengan setiap elemen pada daftar menu secara berurutan hingga ditemukan kecocokan. Metode penelitian dilakukan dengan merancang simulasi pencarian menu menggunakan algoritma Brute Force, kemudian dilakukan evaluasi terhadap performanya dalam konteks data menu berukuran kecil. Hasil penelitian menunjukkan bahwa algoritma ini sederhana, mudah diimplementasikan, serta cukup efisien untuk aplikasi dengan jumlah menu terbatas. Namun, kelemahan algoritma ini terlihat ketika jumlah data menu bertambah besar, karena proses pencarian menjadi lebih lambat dan tidak optimal. Oleh karena itu, untuk skala aplikasi yang lebih luas, disarankan menggunakan algoritma alternatif seperti Binary Search atau Hashing yang lebih efisien dalam menangani jumlah data besar. Temuan penelitian ini diharapkan dapat menjadi referensi bagi pengembang aplikasi dalam memilih strategi pencarian yang sesuai dengan kebutuhan dan skala sistem.

A B S T R A C T

The rapid advancement of digital technology has significantly transformed the culinary industry, particularly in the development of mobile-based food and beverage ordering applications. One of the key features in such applications is the menu search system, which allows customers to locate desired items quickly and efficiently. This research aims to implement and evaluate the effectiveness of the Brute Force algorithm in menu searching for a coffee ordering application. The Brute Force algorithm works by sequentially comparing the user's input with every element in the menu list until a match is identified. The research method involves designing and simulating a menu search system using the Brute Force approach, followed by an evaluation of its performance in the context of a small dataset. The findings reveal that the Brute Force algorithm is simple, easy to implement, and relatively effective for applications with limited menu data. However, its efficiency decreases as the dataset grows, making it less suitable for large-scale applications. For broader and more complex applications, alternative methods such as Binary Search or Hashing are recommended to achieve better performance. This study contributes to providing a practical reference for application developers in selecting an appropriate searching strategy that aligns with the scale and requirements of the system.

Pendahuluan

Perkembangan teknologi informasi pada era digital saat ini telah memberikan dampak besar dalam berbagai bidang kehidupan manusia, termasuk dalam industri kuliner. Pemanfaatan teknologi berbasis aplikasi mobile semakin marak digunakan oleh

restoran, kafe, maupun kedai kopi untuk meningkatkan kualitas layanan kepada pelanggan. Aplikasi pemesanan makanan dan minuman berbasis digital memungkinkan pelanggan untuk melakukan pemesanan secara cepat, efisien, dan tanpa harus mengantri. Tren ini sejalan dengan meningkatnya gaya hidup masyarakat modern yang menuntut kecepatan dan kenyamanan dalam memenuhi kebutuhan sehari-hari.

Salah satu fitur penting dalam aplikasi pemesanan kopi adalah sistem pencarian menu. Fitur ini mempermudah pengguna menemukan menu minuman yang diinginkan tanpa harus menelusuri seluruh daftar menu yang terkadang cukup panjang. Efektivitas sistem pencarian sangat menentukan pengalaman pengguna (user experience), karena semakin cepat dan akurat menu ditemukan, semakin tinggi tingkat kepuasan pelanggan terhadap aplikasi tersebut. Oleh karena itu, pemilihan algoritma pencarian yang tepat menjadi aspek krusial dalam pengembangan aplikasi.Salah satu metode yang paling sederhana dan sering digunakan dalam pencarian adalah algoritma Brute Force. Algoritma ini bekerja dengan cara membandingkan input yang dimasukkan pengguna dengan setiap elemen yang ada di dalam daftar menu, secara berurutan, hingga ditemukan kecocokan. Kesederhanaan konsepnya membuat algoritma ini mudah dipahami dan diimplementasikan oleh pengembang. Dalam konteks aplikasi pemesanan kopi yang umumnya memiliki jumlah menu relatif terbatas, algoritma Brute Force dapat memberikan solusi yang cukup efektif.

Namun demikian, algoritma Brute Force memiliki keterbatasan yang signifikan, terutama ketika dihadapkan pada data berskala besar. Kompleksitas waktu pencarian algoritma ini berada pada orde O(n), yang berarti waktu pencarian akan meningkat secara linear seiring dengan bertambahnya jumlah data. Hal ini menyebabkan algoritma Brute Force menjadi kurang efisien apabila diaplikasikan pada aplikasi yang memiliki ribuan menu atau database produk yang terus berkembang. Dalam kondisi tersebut, diperlukan algoritma alternatif yang lebih optimal seperti Binary Search, Hashing, atau bahkan Knuth-Morris-Pratt (KMP) untuk pencocokan string.Berdasarkan latar belakang tersebut, penelitian ini difokuskan pada implementasi dan analisis efektivitas algoritma Brute Force dalam pencarian menu pada aplikasi pemesanan kopi. Penelitian ini bertujuan untuk:

1. Mendeskripsikan langkah-langkah implementasi algoritma Brute Force pada sistem pencarian menu.
2. Mengevaluasi kinerja algoritma ini pada data menu berukuran kecil.
3. Membandingkan kelebihan dan kekurangan algoritma Brute Force dengan algoritma pencarian lain yang lebih kompleks.

Dengan adanya penelitian ini, diharapkan pengembang aplikasi dapat memahami kelebihan algoritma Brute Force dalam konteks aplikasi sederhana serta mengetahui batasannya pada data berskala besar. Selain itu, hasil penelitian ini juga dapat menjadi dasar pertimbangan bagi pengembang dalam menentukan strategi optimasi sistem pencarian di masa depan.Kontribusi utama penelitian ini terletak pada penyediaan simulasi implementasi algoritma Brute Force dalam aplikasi pemesanan kopi, yang dapat dijadikan sebagai acuan awal dalam pengembangan sistem pencarian sederhana. Lebih jauh lagi, penelitian ini juga memberikan gambaran mengenai pentingnya pemilihan

algoritma yang sesuai dengan skala aplikasi, sehingga sistem yang dikembangkan tidak hanya mudah diimplementasikan tetapi juga efisien serta mampu meningkatkan pengalaman pengguna.

Pembahasan

Algoritma Pencarian dalam Aplikasi Pemesanan Kopi

Fungsi pencarian pada aplikasi pemesanan kopi berperan besar dalam pengalaman pengguna. Pencarian yang lambat atau tidak akurat akan menurunkan kepuasan pengguna dan konversi pesanan. Secara umum, pencarian dapat dilakukan di sisi klien (client-side) atau server (server-side). Untuk aplikasi dengan daftar menu kecil dan statis, pencarian client-side cukup praktis karena mengurangi latensi jaringan. Namun untuk data besar atau dinamis, server-side search dengan indexing menjadi pilihan yang lebih skalabel.

Konsep Dasar Brute Force (tanpa kode)

Algoritma Brute Force bekerja dengan prinsip sederhana: membandingkan input pencarian dengan setiap item di daftar menu secara berurutan hingga ditemukan kecocokan. Tipe pencocokan bisa berupa:

1. Exact match: nama menu harus persis sama.
2. Case-insensitive match: mengabaikan huruf besar/kecil.
3. Substring match: input cocok bila muncul di bagian nama menu.
4. Partial / token match: membandingkan kata per kata (mis. “cafe latte” vs “latte with syrup”).

Karena sifatnya linear, Brute Force cocok untuk dataset kecil dan ketika implementasi perlu cepat. Namun ia kurang cocok saat daftar menu bertumbuh besar.

Langkah-langkah Implementasi (prosedural, tanpa kode)

1. Persiapan data: simpan daftar menu dalam struktur list/array; sertakan metadata (kategori, harga, popularitas) bila perlu untuk urutan hasil.
2. Normalisasi input: lakukan trimming, lowercasing, dan penanganan aksen/diaktritik agar pencarian lebih toleran.
3. Tipe pencocokan: tentukan apakah sistem ingin exact, substring, atau partial match. Pilih sesuai kebutuhan UX.
4. Iterasi & pengecekan: telusuri setiap item, bandingkan dengan input berdasarkan aturan pencocokan. Jika ditemukan, tambahkan ke daftar hasil; untuk efisiensi dapat diberi batas hasil maksimum (top-k).
5. Pengurutan hasil: setelah ditemukan kandidat, urutkan berdasarkan kriteria (kecocokan, popularitas, harga, atau rating).
6. Penanganan tak ditemukan: jika tidak ada kecocokan, berikan saran alternatif (e.g.,

"mungkin maksud: …") atau tampilkan kategori terkait.

7. Feedback UI: highlight bagian yang cocok agar pengguna melihat mengapa item muncul.

Analisis Kompleksitas

Time complexity: O(n) pada tiap query (n = jumlah item menu) karena harus memeriksa setiap elemen sampai kecocokan.

1. Best case: O(1) — jika item ada di posisi pertama.
2. Average/Worst: O(n) — jika item ada di tengah/akhir atau tidak ada.

Space complexity: O(1) tambahan (selain ruang untuk menyimpan hasil sementara). Implikasinya: seiring pertumbuhan daftar menu, waktu respons meningkat linier, yang berpotensi menurunkan pengalaman pengguna pada perangkat mobile ber-spek rendah.

Desain Pengujian & Metrik Evaluasi (cara uji tanpa kode)

Untuk menilai efektivitas Brute Force pada aplikasi, desain pengujian bisa meliputi:

1. Dataset ukuran berbeda: kecil (10–50 item), sedang (100–1000 item), besar (10.000+ item).
2. Skenario query: ditemukan di awal/tengah/akhir, tidak ditemukan, typo/variasi penulisan.
3. Metrik: waktu respon rata-rata per query, waktu terburuk, memori tambahan, dan pengalaman pengguna (mis. waktu sampai hasil terlihat).
4. Platform: uji di perangkat low-end mobile, high-end mobile, dan server untuk melihat variasi performa.

Hasil pengujian bersifat kuantitatif; namun saat ini tanpa pengukuran nyata, kita dapat menyimpulkan sifat kualitatif performa berdasarkan teori: pada dataset kecil Brute Force sangat memadai; pada dataset besar perlu optimasi.

Interpretasi Hasil (ekspektasi & implikasi)

1. Dataset kecil (≤100 item): Brute Force terasa instan dan sangat sederhana untuk dipelihara. Implementasi client-side memberikan pengalaman responsif tanpa biaya server.
2. Dataset menengah (100–1000 item): Masih bisa diterima pada perangkat modern, tetapi perlu perhatian pada fitur tambahan (fuzzy matching, highlighting) yang bisa menambah beban.
3. Dataset besar (>1.000–10.000+): Brute Force mulai terlihat keterbatasannya—latensi meningkat, penggunaan CPU bertambah, dan UX menurun. Di titik ini perlu beralih ke metode yang lebih efisien atau hybrid (index + caching).

Perbandingan dengan Metode Lain (ketika Brute Force kurang cocok)

1. Binary Search: O(log n) — jauh lebih cepat, tetapi mengharuskan data terurut

dan kurang fleksibel untuk substring/fuzzy search. Cocok bila pencarian berbentuk exact lookup pada data terurut.

2. Hashing (Hash table / dictionary): O(1) rata-rata — ideal untuk lookup exact; memerlukan memori lebih dan preprocessing untuk membangun hash map. Cocok untuk lookup cepat nama menu persis.
3. Trie (prefix tree): Sangat bagus untuk autocompletion dan prefix matching; biaya memori lebih tinggi namun memberikan lookup cepat untuk awalan kata.
4. Inverted index / full-text search (dengan engine seperti Elasticsearch): Pilihan terbaik untuk skala besar dan fitur pencarian kaya (ranking, fuzzy, stemming), tetapi memerlukan infrastruktur dan kompleksitas integrasi.
5. Algoritma pencocokan string (KMP, Boyer-Moore, Levenshtein untuk fuzzy): Berguna untuk pola dan toleransi typo, tetapi lebih kompleks untuk implementasi langsung pada aplikasi sederhana.

Strategi Optimasi Practical (rekomendasi implementasi)

1. Normalisasi & preprocessing untuk mengurangi perbandingan yang tidak perlu.
2. Caching hasil populer agar query umum tidak perlu diproses ulang.
3. Debouncing pada input (di UI) untuk menghindari pencarian di tiap ketukan keyboard.
4. Limitasi hasil (top-k) untuk menampilkan hanya sebagian dan mengurangi beban rendering.
5. Hybrid approach: gunakan Brute Force client-side untuk daftar kecil + server-side indexing untuk data besar.
6. Gunakan struktur data yang sesuai: bila fitur autocompletion penting, pertimbangkan Trie; bila exact lookup, pakai hash map.

Dampak UX & Rekomendasi Desain Antarmuka

1. Tampilkan hasil secepat mungkin (mis. response <200 ms terasa instan).
2. Beri umpan balik ketika tidak ditemukan dan saran alternatif.
3. Implementasikan highlight pada bagian yang cocok dan urutan berdasarkan relevansi/popularitas.
4. Sediakan pilihan filter (kategori, harga, ukuran) untuk mengurangi beban pencarian.

Keterbatasan Pembahasan dan Saran Lanjutan

Pembahasan ini bersifat konseptual dan berbasis analisis teoretis; untuk bukti empiris diperlukan eksperimen terukur pada perangkat nyata dengan dataset yang merepresentasikan kondisi riil. Saran lanjutan: lakukan uji performa (benchmark) aktual, bandingkan Brute Force dengan hashing dan inverted-index pada dataset nyata, serta

ukur dampak UX melalui user testing.

Kesimpulan dan saran

Kesimpulan

Berdasarkan hasil penelitian dan pembahasan mengenai implementasi algoritma Brute Force dalam pencarian menu pada aplikasi pemesanan kopi, dapat ditarik beberapa poin penting sebagai berikut:

1. Kesederhanaan dan kemudahan implementasi menjadikan algoritma Brute Force cukup efektif untuk aplikasi dengan jumlah data yang terbatas, seperti daftar menu pada kedai kopi skala kecil hingga menengah.
2. Kinerja algoritma menunjukkan bahwa Brute Force mampu memberikan hasil pencarian yang akurat dengan kompleksitas waktu linear (O(n)). Namun, efektivitas ini hanya optimal ketika ukuran dataset relatif kecil.
3. Keterbatasan algoritma mulai terlihat ketika jumlah data bertambah besar, karena proses pencarian akan semakin lama. Hal ini menjadikan Brute Force kurang sesuai untuk aplikasi dengan skala data yang luas atau dinamis.
4. Perbandingan dengan algoritma lain menunjukkan bahwa meskipun Brute Force sederhana, algoritma alternatif seperti Binary Search atau Hashing lebih unggul dalam hal efisiensi waktu pada dataset yang lebih besar.

Dengan demikian, Brute Force masih relevan digunakan pada aplikasi dengan kebutuhan sederhana, tetapi untuk pengembangan lebih lanjut pada aplikasi berskala besar, diperlukan penerapan algoritma pencarian yang lebih efisien.

Saran

1. Pengembangan Aplikasi Lebih Lanjut – Pada tahap awal, algoritma Brute Force dapat digunakan sebagai solusi sederhana. Namun, seiring pertumbuhan jumlah menu, perlu dipertimbangkan penggunaan algoritma lain seperti Binary Search atau Hashing.
2. Optimasi Pengolahan Data – Untuk meningkatkan performa pencarian, data menu sebaiknya disusun dan dikelompokkan dengan baik, misalnya berdasarkan kategori minuman, sehingga mempercepat proses pencarian meskipun masih menggunakan Brute Force.
3. Evaluasi Skalabilitas – Penelitian lanjutan perlu dilakukan untuk menguji performa algoritma pencarian lain dalam konteks aplikasi pemesanan kopi, khususnya jika jumlah data diperluas hingga ratusan atau ribuan item.
4. Integrasi dengan Teknologi Lanjutan – Untuk masa depan, aplikasi dapat mengintegrasikan sistem pencarian berbasis machine learning atau natural language processing (NLP) agar mampu memahami variasi kata kunci pengguna dengan lebih baik.

Daftar pustaka

Alana, S. H., Hasugian, A. H., & Suhardi, S. (2024). Implementasi Algoritma Brute Force dan Knuth-Morris-Pratt (KMP) pada Aplikasi Saran Buku Bacaan Bagi Pengunjung Perpustakaan. G-Tech: Jurnal Teknologi Terapan, 8(1), 490–501. https://doi.org/10.33379/gtech.v8i1.3711

Dalimunthe, A. S., Furqan, M., & Hasugian, A. H. (2025). Penerapan Algoritma Brute Force pada Aplikasi Penerjemah Bahasa Indonesia – Bahasa Mandailing Berbasis Mobile. Journal of Science and Social Research, 8(2), 2015–2020. http://jurnal.goretanpena.com/index.php/JSSR

Kafabihi, R. J. (2025). Optimasi Rute menggunakan Algoritma A* dan Brute Force pada Game Edukatif. (Tesis). UIN Maulana Malik Ibrahim Malang. Retrieved from https://etheses.uin-malang.ac.id/75765/1/210605110069.pdf

Mahrozi, N. (2023). [Judul yang relevan tentang algoritma dalam Jurnal Ilmiah Sain dan Teknologi]. Jurnal Ilmiah Sain dan Teknologi, 1(2). Retrieved from https://repository.uin-malang.ac.id/17888/2/17888.pdf

Mesran, A. (2014). Implementasi Algoritma Brute Force dalam Pencarian Data Katalog Buku Perpustakaan. Majalah Ilmiah Informasi dan Teknologi (INTI), 3(1). Retrieved from ResearchGate

Sugiharto, S. (2018). Implementasi Algoritma Brute Force dalam Pencarian Kebudayaan di Indonesia Berbasis Mobile Application. Jurnal Buffer Informatika, 4(2). https://journal.uniku.ac.id/index.php/buffer/article/download/1472/1093

Fadila, J. N., & Arif, Y. M. (2020). Implementasi algoritma RVO sebagai sistem kendali gerombolan NPC pada permainan Action RPG. Matics: Jurnal Ilmu Komputer dan Teknologi Informasi, 12(1), 87–89. Universitas Islam Negeri Maulana Malik Ibrahim Malang. https://repository.uin-malang.ac.id/

Khudzaifah, M. (2023). Program pengamanan data pribadi dengan algoritma Hill Cipher dan Arnold Cat Map [Sertifikat hak cipta No. 000567661]. Universitas Islam Negeri Maulana Malik Ibrahim Malang. https://repository.uin-malang.ac.id/