ANALISIS PERBANDINGAN ALGORITMA *DEPTH FIRST SEARCH* DAN ALGORITMA *BACKTRACKING* DALAM PENYELESAIAN PERMAINAN SUDOKU

ADINDA BORU SARAGIH
H1011201065

SKRIPSI

PROGRAM STUDI MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS TANJUNGPURA
PONTIANAK
2025

ANALISIS PERBANDINGAN ALGORITMA *DEPTH FIRST SEARCH* DAN ALGORITMA *BACKTRACKING* DALAM PENYELESAIAN PERMAINAN SUDOKU

ADINDA BORU SARAGIH
H1011201065

Skripsi

Sebagai salah satu syarat untuk memperoleh gelar

Sarjana Matematika pada Program Studi Matematika

PROGRAM STUDI MATEMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS TANJUNGPURA

PONTIANAK

2025

LEMBAR PENGESAHAN

Judul Tugas Akhir	: Analisis Perbandingan Algoritma *Depth First Search* dan *Backtracking* dalam Penyelesaian Permainan Sodoku
Nama Mahasiswa	: Adinda Boru Saragih
NIM	: H1011201065
Jurusan/Program Studi	: Matematika/Matematika
Tanggal Lulus	: 31 Juli 2025
SK Pembimbing	: No. 2539/UN22.8/TD.06/2024/2 September 2024
SK Penguji	: No. 2510/UN22.8/TD.06/2025/29 Juli 2025

Dosen Pembimbing

Pembimbing I	Pembimbing II
Dr. Bayu Prihandono, M.Sc	Meliana Pasaribu, M.Sc
NIP 197911152005011003	NIP 198710192019032006

Dosen Penguji

Ketua Penguji	Anggota Penguji
Drs. Helmi, M.Si	Dr. Evi Noviani, M.Si,CFrA
NIP 196410171998021001	NIP 199411142020122014

Pimpinan Sidang (merangkap anggota penguji)	Sekretaris Sidang (merangkap anggota penguji)
Dr. Bayu Prihandono, M.Sc	Meliana Pasaribu, M.Sc
NIP 197911152005011003	NIP 198710192019032006

Mengesahkan
Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Tanjungpura

Prof. Dr. Gusrizal, S.Si., M.Si
NIP 197108022000031001

PERNYATAAN INTERGRITAS AKADEMIK

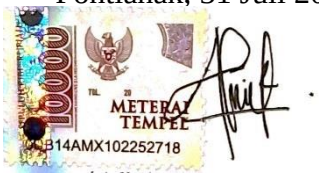
Saya yang bertanda tangan di bawah ini:

Nama	: ADINDA BORU SARAGIH
NIM	: H1011201065
Program Studi/ Jurusan	: Matematika/Matematika
Fakultas	: Matematika dan Ilmu Pengetahuan Alam

dengan ini menyatakan bahwa dokumen ilmiah Tugas Akhir yang disajikan ini tidak mengandung unsur pelanggaran integritas akademik sesuai Peraturan Menteri Pendidikan, Kebudayaan, Riset, Dan Teknologi Republik Indonesia Nomor 39 Tahun 2021. Apabila di kemudian hari dokumen ilmiah Tugas Akhir ini mengandung unsur pelanggaran integritas akademik sesuai ketentuan perundangan tersebut, maka saya bersedia menerima sanksi akademik dan/atau sanksi hukum yang berlaku.

Demikian pernyatan ini untuk digunakan sebagaimana mestinya.

Pontianak, 31 Juli 2025



Adinda Boru Saragih
H1011201065

Analisis Perbandingan Algoritma *Depth First Search* dan Algoritma *Backtracking* dalam Penyelesaian Perminan Sudoku

Abstrak

Sudoku adalah permainan dengan jenis teka-teki logika berbentuk n × n, yang terdiri dari baris, kolom, grid dan kotak. Kotak-kotak tersebut diisi dengan angka tanpa ada pengulangan dalam satu grid, baris atau kolom. Sudoku memiliki tingkat kesulitan yang berbeda, mulai dari tingkat sangat mudah hingga sangat sulit. Petunjuk angka yang diberikan setidaknya memiliki minimal 17 angka. Untuk menyelesaikan permainan sudoku dengan waktu yang lebih sedikit, di perlukan algoritma pencarian yang tepat. Dalam penelitian ini digunakan algoritma *Depth First Search* (DFS) dan *Backtracking* untuk melakukan penelusuran solusi dengan bantuan pohon berakar. Tujuan penelitian ini untuk membandingkan algoritma DFS dan Backtracking dalam menyelesaikan permainan sudoku. Penelitian ini menggunakan Sudoku berukuran 9 × 9 pada tingkat sulit, dari aplikasi Sudoku versi 4.6.0 dengan 54 kotak kosong yang harus diselesaikan. Penelusuran solusi menunjukkan DFS membutuhkan sebanyak 249 langkah dan 70 iterasi, sedangkan *Backtracking* membutuhkan 173 langkah dan 25 iterasi untuk mendapatkan solusi secara keseluruhan. Setelah dilakukan sebanyak 20 kali percobaan dengan *Pyhton*, hasil percobaan menunjukkan, DFS membutuhkan rata-rata 61.443 langkah dan 6.828 iterasi dengan waktu eksekusi 0,408114 detik, sedangkan algoritma *Backtracking* hanya memerlukan rata-rata 17.970 langkah dan 2.021 iterasi, dengan waktu eksekusi 0,024750 detik. Perbedaan ini menunjukkan algoritma *Backtracking* lebih efisien dibandingkan algoritma DFS untuk menyelesaikan permasalahan sudoku pada penelitian ini. Dilakukan percobaan tambahan pada beberapa permasalahan sudoku berdasarkan tingkat kesulitannya, sebanyak 10 kali. Hasilnya menunjukkan algoritma *Backtracking* konsisten lebih cepat dalam eksekusi waktu dibandingkan DFS, meskipun DFS memerlukan lebih sedikit langkah dan iterasi. Ini menunjukkan *Backtracking* lebih efisien dalam menyelesaikan sudoku.

Kata Kunci: penelusuran solusi, pohon berakar, solusi

Comparative Analysis of Depth First Search and Backtracking Algorithm In Solving Sudoku

Abstract

Sudoku is a classic $n \times n$ logic puzzle game, consisting of rows, columns, grids and boxes. The boxes are filled with numbers with no repetition within a grid, row or column. Sudoku has different levels of difficulty, ranging from very easy to very difficult. The number clue given in sudoku has at least 17 numbers. The more difficult the sudoku game, the more time it takes to complete it. To solve the sudoku game with less time, the right search algorithm is needed. In this research, Depth First Search (DFS) and *Backtracking* algorithms are used to search for sudoku problem solutions with the help of rooted trees. The purpose of this research is to compare Depth First Search (DFS) and *Backtracking* algorithm in solving sudoku game. In this research, the sudoku used is a 9×9 sudoku at a difficult level with 54 empty boxes that must be solved. The solution search shows that the DFS algorithm requires 249 steps and 70 iterations, while the *Backtracking* algorithm requires 173 steps and 25 iterations to get the overall sudoku solution. After 5 experiments in Python, the results show that the DFS algorithm takes an average of more than 0,5 seconds to solve the sudoku game and the *Backtracking* algorithm takes an average of less than 0,1 seconds to solve the sudoku game. The research shows that the *Backtracking* algorithm is more efficient in solving the 9×9 sudoku game problem at a difficult level compared to the DFS algorithm.

Keywords: solution search, rooted trees, solution

PRAKATA

Puji dan syukur Kepada Tuhan Yang Maha Esa, atas kasih dan karunia-Nya, sehingga penulis dapat menyelesaikan skripsi ini dengan baik. Penyelesaian skripsi ini tidak lepas dari dukungan berbagai pihak. Oleh karena itu, dengan penuh kasih, penulis mengucapkan terimakasih kepada:

1. Tuhan Yesus Kristus, atas kekuatan dan penyertaan-Nya dalam masa perkuliahan dan dalam penyelesaian penulisan skripsi ini.
2. Bapak Iwan Saragih dan ibu Amor Fatria serta dek Manda, dek Nanda, dek Ucok dan kak Dita atas doa dan dukungan dana dalam penyelesaian penulisan skripsi ini.
3. Bapak Dr. Bayu Prihandono, M.Sc, selaku dosen pembimbing pertama dan ibu Meliana Pasaribu, S.Pd., M.Sc selaku dosen pembimbing kedua, atas arahan dan bimbingan yang diberikan selama proses penyusunan skripsi ini.
4. Bapak Drs. Helmi, M.Sc, selaku dosen penguji pertama dan ibu Evi Noviani, M.Si, CFrA selaku dosen penguji kedua atas saran dan masukannya pada penyelesaian skripsi ini
5. Saudari Eva, Orpa, Maria, Elin, Rika, Anggun, Anggel, Abda, Cinda yang terus mendukung dalam penulisan skripsi ini dan kepada Chintya Arisan Noveny yang selalu setia menemai di setiap masa dalam penyelesaian penulisan skripsi ini.

Penulis menyadari bahwa skripsi ini masih memiliki banyak kekurangan, sehingga kritik dan saran yang membangun sangat diharapkan untuk penyempurnaan penulisan ini. Akhir kata, penulis mengucapkan terima kasih kepada semua pihak yang terlibat.

Pontianak, 31 Juli 2025

Penulis

DAFTAR ISI

LEMBAR PENGESAHAN ..... Error! Bookmark not defined.

PERNYATAAN INTERGRITAS AKADEMIK..... ii

Abstrak ……..... iv

***Abstract* ..... v**

PRAKATA..... vi

DAFTAR ISI..... vii

DAFTAR TABEL ix

DAFTAR GAMBAR..... xi

DAFTAR SIMBOL..... xiii

BAB I PENDAHULUAN..... 1

1.1 Latar Belakang 1

1.2 Rumusan Masalah 2

1.3 Tujuan Penelitian..... 3

1.4 Batasan Masalah..... 3

1.5 Tinjauan Pustaka 3

1.6 Metodologi Penelitian 5

BAB II LANDASAN TEORI 8

2.1 Teori Graf..... 8

2.2 Jenis-jenis Graf..... 9

2.3 Pohon..... 10

2.4 Permainan Sudoku..... 11

2.5 Algoritma *Depth First Search* (DFS)..... 13

2.6 Algoritma *Backtracking* 14

2.7 Program Linear..... 16

BAB III STUDI KASUS ALGORITMA *DEPTH FIRST SEARCH* DAN *BACKTRACKING* UNTUK PENYELESAIAN PERMAINAN SUDOKU 20

3.1 Permasalahan Sudoku 20
3.2 Pemodelan Matematika 22
3.3 Penyelesaian permainan sudoku dengan *Depth First Search* 24
3.4 Penyelesaian permainan sudoku dengan *Backtracking* 35
3.5 Hasil Simulasi 46

BAB IV KESIMPULAN 49

DAFTAR PUSTAKA xiv

LAMPIRAN xvi

DAFTAR TABEL

Tabel 2. 1 Tingkat Kesulitan Sudoku 13
Tabel 3. 1 Indeks Sudoku 21
Tabel 3. 2 Solusi DFS $r = 1$ 24
Tabel 3. 3 Solusi DFS $r = 2$ 25
Tabel 3. 4 Solusi DFS $r = 3$ 26
Tabel 3. 5 Solusi DFS r $= 4$..... 26
Tabel 3. 6 Solusi DFS $r = 5$ 27
Tabel 3. 7 Solusi DFS $r = 6$ 28
Tabel 3. 8 Solusi DFS $r = 5$ Dengan *Backtrack*..... 28
Tabel 3. 9 Solusi DFS $r = 4$ Dengan *Backtrack*..... 29
Tabel 3. 10 Solusi DFS $r = 5$ Setelah *Backtrack* 29
Tabel 3. 11 Solusi DFS $r = 2$ Dengan *Backtrack*..... 30
Tabel 3. 12 Solusi DFS $r = 3$ Setelah *Backtrack* 31
Tabel 3. 13 Solusi DFS $r = 4$ Setelah *Backtrack* 31
Tabel 3. 14 Solus DFS Lanjutan $r = 5$ 32
Tabel 3. 15 Solusi DFS Lanjutan $r = 6$ 32
Tabel 3. 16 Solusi DFS $r = 7$ 33
Tabel 3. 17 Solusi DFS $r = 8$ 34
Tabel 3. 18 Solusi DFS $r = 9$ 34
Tabel 3. 19 Solusi *Backtracking* $r = 1$ 36
Tabel 3. 20 Solusi *Backtracking* $r = 2$ 36
Tabel 3. 21 Solusi *Backtracking* $r = 3$ 37
Tabel 3. 22 Solusi *Backtracking* $r = 4$ 37
Tabel 3. 23 Solusi *Backtracking* $r = 5$ 38
Tabel 3. 24 Solusi *Backtracking* Baris ke-6 39
Tabel 3. 25 Solusi *Backtracking* $r = 5$ Dengan *Backtrack*..... 39
Tabel 3. 26 Solusi *Backtracking* $r = 4$ Dengan *Backtrack*..... 40

Tabel 3. 27 Solusi *Backtracking* $r = 5$ Setelah *Backtrack* 40
Tabel 3. 28 Solusi *Backtracking* $r = 2$ Dengan *Backtrack*..... 41
Tabel 3. 29 Solusi *Backtracking* $r = 3$ Dengan *Backtrack*..... 42
Tabel 3. 30 Solusi *Backtracking* $r = 4$ Setelah *Backtrack* 42
Tabel 3. 31 Solusi *Backtracking* $r = 5$ Setelah *Backtrack* 43
Tabel 3. 32 Solusi *Backtracking* $r = 6$ Dengan *Backtrack*..... 43
Tabel 3. 33 Solusi *Backtracking* $r = 7$ 44
Tabel 3. 34 Solusi *Backtracking* $r = 8$ 44
Tabel 3. 35 Solusi *Backtracking* $r = 9$ 45
Tabel 3. 36 Hasil Percobaan Sudoku Pada Tingkat Sulit..... 46
Tabel 3. 37 Hasil Percobaan Beberapa Permasalahan Sudoku 47

DAFTAR GAMBAR

Gambar 1. 1 Flowchart Algoritma DFS 6
Gambar 1. 2 Flowchart Algoritma *Backtracking* 7
Gambar 2. 1 Graf Sederhana 9
Gambar 2. 2 Graf Ganda Tak Sederhana 9
Gambar 2. 3 Graf Semu Tak Sederhana 9
Gambar 2. 4 Graf Berarah 10
Gambar 2. 5 Graf Tak Berarah 10
Gambar 2. 6 Graf Lengkap G dan Pohon Merentangnya 11
Gambar 2. 7 Pohon Berakar 11
Gambar 2. 8 Sudoku 6×6 12
Gambar 2. 9 Sudoku 9×9 12
Gambar 2. 10 Sudoku 16×16 12
Gambar 2. 11 Pohon Algoritma DFS 14
Gambar 2. 12 Pohon Algoritma *Backtracking* 15
Gambar 3. 1 Papan Sudoku 20
Gambar 3. 2 Permasalahan Sudoku Tingkat Sulit 22
Gambar 3. 3 Pohon DFS $r = 1$ 24
Gambar 3. 4 Pohon DFS $r = 2$ 25
Gambar 3. 5 Pohon DFS $r = 3$ 26
Gambar 3. 6 Pohon DFS $r = 4$ 26
Gambar 3. 7 Pohon DFS $r = 5$ 27
Gambar 3. 8 Pohon DFS $r = 6$ 28
Gambar 3. 9 Pohon DFS $r = 5$ Dengan *Backtrack* 28
Gambar 3. 10 Pohon DFS $r = 4$ Dengan *Backtrack* 29
Gambar 3. 11 Pohon DFS $r = 5$ Setelah *Backtrack* 29
Gambar 3. 12 Pohon DFS $r = 2$ Dengan *Backtrack* 30
Gambar 3. 13 Pohon DFS $r = 3$ Dengan *Backtrack* 31
Gambar 3. 14 Pohon DFS $r = 4$ Setelah *Backtrack* 31

Gambar 3. 15 Pohon DFS Lanjutan $r = 5$ 32
Gambar 3. 16 Pohon DFS Lanjutan $r = 6$ 32
Gambar 3. 17 Pohon DFS $r = 7$ 33
Gambar 3. 18 Pohon DFS $r = 8$ 34
Gambar 3. 19 Pohon DFS $r = 9$ 34
Gambar 3. 20 Penyelesaian Sudoku dengan DFS 35
Gambar 3. 21 Pohon *Backtracking* $r = 1$ 36
Gambar 3. 22 Pohon *Backtracking* $r = 2$ 36
Gambar 3. 23 Pohon *Backtracking* $r = 3$ 37
Gambar 3. 24 Pohon *Backtracking* $r = 4$ 37
Gambar 3. 25 Pohon *Backtracking* $r = 5$ 38
Gambar 3. 26 Pohon *Backtracking* $r = 6$ 39
Gambar 3. 27 Pohon *Backtracking* $r = 5$ Dengan *Backtrack*..... 39
Gambar 3. 28 Pohon *Backtracking* $r = 4$ Dengan *Backtrack*..... 40
Gambar 3. 29 Pohon *Backtracking* $r = 5$ Setelah *Backtrack* 40
Gambar 3. 30 Pohon *Backtracking* $r = 2$ Dengan *Backtrack*..... 41
Gambar 3. 31 Pohon *Backtracking* $r = 3$ Dengan *Backtrack*..... 42
Gambar 3. 32 Pohon *Backtracking* $r = 4$ Setelah *Backtrack* 42
Gambar 3. 33 Pohon *Backtracking* $r = 5$ Setelah *Backtrack* 43
Gambar 3. 34 Pohon *Backtracking* $r = 6$ Dengan *Backtrack*..... 43
Gambar 3. 35 Pohon *Backtracking* $r = 7$ 44
Gambar 3. 36 Pohon *Backtracking* $r = 8$ 44
Gambar 3. 37 Pohon *Backtracking* $r = 9$ 45
Gambar 3. 38 Penyelesaian Sudoku Menggunakan *Backtracking* 45

DAFTAR SIMBOL

$S_{i,j}$	:	Matriks Sudoku berordo $i \times j$, i = baris, j = kolom
$n \times n$	:	Ukuran papan sudoku
$G = (V, E)$	:	Graf G dengan pasangan himpunan simpul $V(G)$ dan himpunan sisi $E(G)$
$e = (u, v)$	:	Sisi yang menghubungkan simpul u dan v
S_i	:	Solusi baris ke-i, $i = 1,2, \dots, 9$

BAB I
PENDAHULUAN

1.1 Latar Belakang

Kemampuan untuk memecahkan masalah dan pemahaman algoritma sangat penting di era digital. Berbagai pendekatan seperti kecerdasan buatan (AI), optimasi, dan pemograman memanfaatkan algoritma pencarian untuk menyelesaikan dan menemukan solusi. Kecerdasan buatan (AI) mendukung komputer untuk membuat sistem yang lebih cerdas, misalnya dalam menyelesaikan permainan logika yang memiliki tantangan tertentu (Pratiwi, 2024).

Dalam ilmu komputer, algoritma pencarian adalah teknik dasar yang digunakan untuk menemukan solusi dari berbagai masalah kompleks yang melibatkan banyak kemungkinan. Algoritma pencarian yang sering digunakan adalah algoritma *Depth First Search* (DFS) dan algoritma *Backtracking*. Algoritma *Depth First Search* (DFS) melakukan pencarian secara mendalam dengan menjelajahi pohon berakar dari satu cabang ke cabang berikutnya. Algoritma *Depth First Search* (DFS) memiliki kemampuan untuk mengeksplorasi jalur secara menyeluruh, yang memungkinkannya menemukan solusi untuk masalah yang memiliki jalur pencarian yang kompleks (Andina, Noviani, & Yudhi, 2022). Pencarian algoritma *Backtracking* berbasis pada algoritma DFS, namun algoritma *Backtracking* tidak menjelajahi jalur secara menyeluruh, hanya jalur yang mengarah pada solusi yang akan dieksplorasi dan kembali ke langkah sebelumnya jika menemukan jalan buntu (Rismana, Nilamsari, & Fransiskus, 2024). Algoritma DFS dan *Backtracking* digunakan untuk menyelesaikan permasalahan kombinatorial pada permainan logika tertentu, salah satunya adalah sudoku.

Sudoku adalah permainan klasik dengan jenis teka-teki logika yang bertujuan untuk melatih kemampuan berpikir. Papan sudoku umumnya berukuran 9×9, namun seiring perkembangannya dibuat beberapa ukuran yang berbeda. Tujuan permainan sudoku adalah untuk mengisi angka dalam kotak tanpa ada angka yang berulang dalam satu *grid*, baris atau kolom. Permainan ini memiliki berbagai

tingkat kesulitan berdasarkan dengan jumlah petunjuk angka yang diberikan mulai dari tingkat sangat mudah, mudah, sedang, sulit dan sangat sulit. Pada tingkat sangat mudah diberikan ≥ 46 jumlah petunjuk angka, pada tingkat mudah diberikan $36 - 46$ jumlah petunjuk angka, pada tingkat sedang diberikan $32 - 35$ jumlah petunjuk angka, pada tingkat sulit diberikan $27 - 31$ jumlah petunjuk angka, pada tingkat sangat sulit diberikan $17 - 26$ jumlah petunjuk angka. Berdasarkan tingkat kesulitannya permainan sudoku membutuhkan waktu yang cukup lama untuk diselesaikan. Oleh karena itu diperlukan algoritma pencarian yang tepat untuk menyelesaikan permainan ini dengan waktu yang lebih sedikit.

Algoritma DFS dan *Backtracking* dapat menyelesaikan permasalahan sudoku dengan tepat, tetapi tetap memiliki perbedaan karakteristik. Kedua algoritma ini memiliki jumlah langkah, jumlah iterasi dan waktu eksekusi yang berbeda, sehingga dibutuhkan pengujian pada algoritma *Depth First Search* dan *Backtracking* untuk menentukan waktu eksekusi kedua algoritma ini dalam menyelesaikan permainan sudoku. Pengujian algoritma menggunakan bahasa pemograman yang mampu mengelola data dengan baik dan memiliki aturan penulisan yang mudah di pahami seperti *Python.*

Python memiliki banyak fitur yang membedakannya dari bahasa pemograman yang lain, yaitu pemograman tingkat tinggi yang mudah digunakan dan dipahami, mudah dalam pengembangannya, selain itu *python* memiliki menajemen memori yang dinamis, *open source* dan tentunya gratis (Sianipar & Wadi, 2015). Dengan demikian bahasa pemograman ini dapat digunakan untuk menentukan waktu ekskusi yang dibutuhkan algoritma *Depth First Search* (DFS) dan *Backtracking* dalam penyelesaian permainan sudoku.

1.2 Rumusan Masalah

Berdasarkan latar belakang tersebut rumusan masalah adalah bagaimana perbandingan algoritma *Depth First Search* dan algoritma *Backtracking* dalam menyelesaikan permasalahan permainan sudoku.

1.3 Tujuan Penelitian

Berdasarkan rumusan masalah, didapatkan tujuan penelitian sebagai berikut:

1. Menerapkan algoritma *Depth First Search* dan algoritma *Backtracking* untuk menyelesaikan permainan sudoku
2. Membandingkan algoritma *Depth First Search* dan algoritma *Backtracking* dalam menyelesaikan permainan sudoku

1.4 Batasan Masalah

Adapun batasan-batasan masalah penelitian sebagai berikut:

1. Fokus penelitian penyelesaian permainan sudoku dilakukan pada sudoku berukuran 9×9 pada tingkat sulit dari aplikasi sudoku versi 4.6.0.
2. Algoritma yang digunakan pada penelitian ini adalah algoritma *Depth First Search* dan algoritma *Backtracking.*
3. Bahasa pemograman *Python* untuk mengetahui jumlah langkah, jumlah iterasi dan waktu eksekusi yang dibutuhkan algoritma dalam menyelesaikan permainan sudoku.

1.5 Tinjauan Pustaka

Beberapa penelitian terkait algoritma *Depth First Search* (DFS) dan algoritma *Backtacking* dalam penyelesaian sudoku telah ditinjau dan diteliti. Andina, dkk (2022) menyatakan proses penelusuran pohon berakar pada algoritma DFS dimulai dari simpul akar yang mewakili keadaan awal permainan Sudoku. Kemudian, algoritma secara mendalam mengeksplorasi angka yang layak yang dapat ditempatkan pada kotak kosong dengan membangun cabang-cabang pohon berdasarkan kandidat angka yang layak, memperhatikan aturan tidak adanya angka yang sama di baris, kolom, dan *grid*. Setiap cabang mewakili pilihan angka tertentu, dan proses ini berlanjut ke tingkat berikutnya hingga mencapai kondisi solusi atau tidak ada lagi kandidat yang dapat dicoba. Jika jalur tersebut tidak menghasilkan solusi, algoritma melakukan *Backtrack* ke simpul sebelumnya untuk mencoba jalur lain. Proses ini berlanjut hingga tidak ada lagi kandidat yang dapat dicoba atau ketika sudah ditemukan solusi.

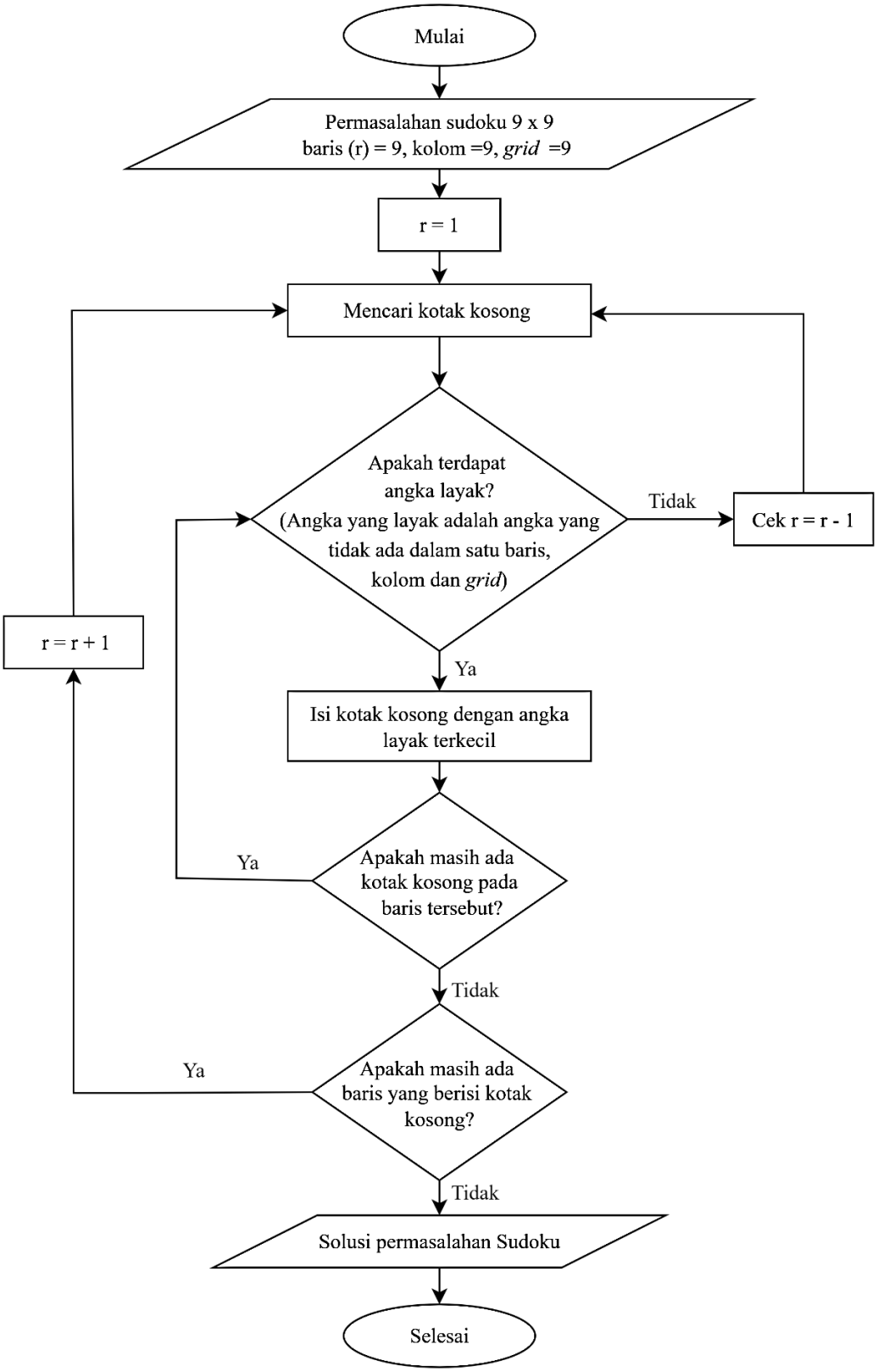
Pada tahun 2021 Hanafi, dkk melakukan penelitian mengenai Penyelesaian Permainan Sudoku Menggunakan Algoritma *Backtracking* Berbasis *Artificial Intelligence*. Dalam penelitiannya menyatakan pencarian algoritma *Backtracking* berbasis pada DFS. Proses pencarian dimulai dengan mencari posisi kotak kosong pada papan Sudoku. Setelah ditemukan algoritma mencoba mengisi kotak tersebut dengan angka dari 1 hingga 9 yang memenuhi aturan sudoku yaitu tidak terdapat angka yang berulang dalam satu baris, kolom dan *grid*. Jika angka valid, maka algoritma menempatkannya dan melanjutkan ke langkah berikutnya, tetapi jika tidak ada angka yang valid, algoritma akan kembali ke langkah sebelumnya *(Backtrack)* dan mencoba angka lain. Proses ini terus berulang hingga seluruh papan terisi dengan solusi yang valid. Pencarian solusinya secara sistematis mengeksplorasi Angka yang Layak di setiap kotak kosong, dan kembali jika menemukan jalan buntu, sehingga memastikan pencarian solusi dan terstruktur.

Selanjutnya pada tahun 2024 Rismana, dkk melakukan penelitian mengenai Penyelesaian Permainan Sudoku Menggunakan Algoritma *Backtracking*. Dalam penelitiannya menyatakan proses penelusuran dengan pohon penyelesaian dalam algoritma *Backtracking* dimulai dengan membentuk sebuah pohon berakar yang mewakili semua kemungkinan langkah dalam mengisi kotak kosong pada Sudoku, dimulai dari akar dan berkembang ke cabang-cabang yang mewakili pilihan angka yang mungkin untuk setiap kotak kosong. Setiap simpul pada pohon mewakili sebuah kondisi tertentu dari papan Sudoku, dan cabang-cabangnya menunjukkan pilihan angka yang dicoba. Jika pilihan tersebut memenuhi aturan, pohon akan berkembang ke tingkat berikutnya dengan mencoba angka lain pada kotak berikutnya, sedangkan jika pilihan tidak memenuhi syarat, cabang tersebut akan dipangkas dan algoritma kembali ke simpul sebelumnya untuk mencoba pilihan lain. Penelusuran dilakukan secara sistematis dari akar ke daun, membentuk lintasan yang berisi kemungkinan solusi, dan jika simpul terakhir pada simpul tanpa solusi, maka algoritma akan kembali ke simpul sebelumnya dan mencoba alternatif lain. Proses ini berlanjut hingga ditemukan simpul yang memenuhi semua kondisi solusi, atau semua kemungkinan telah dicoba.

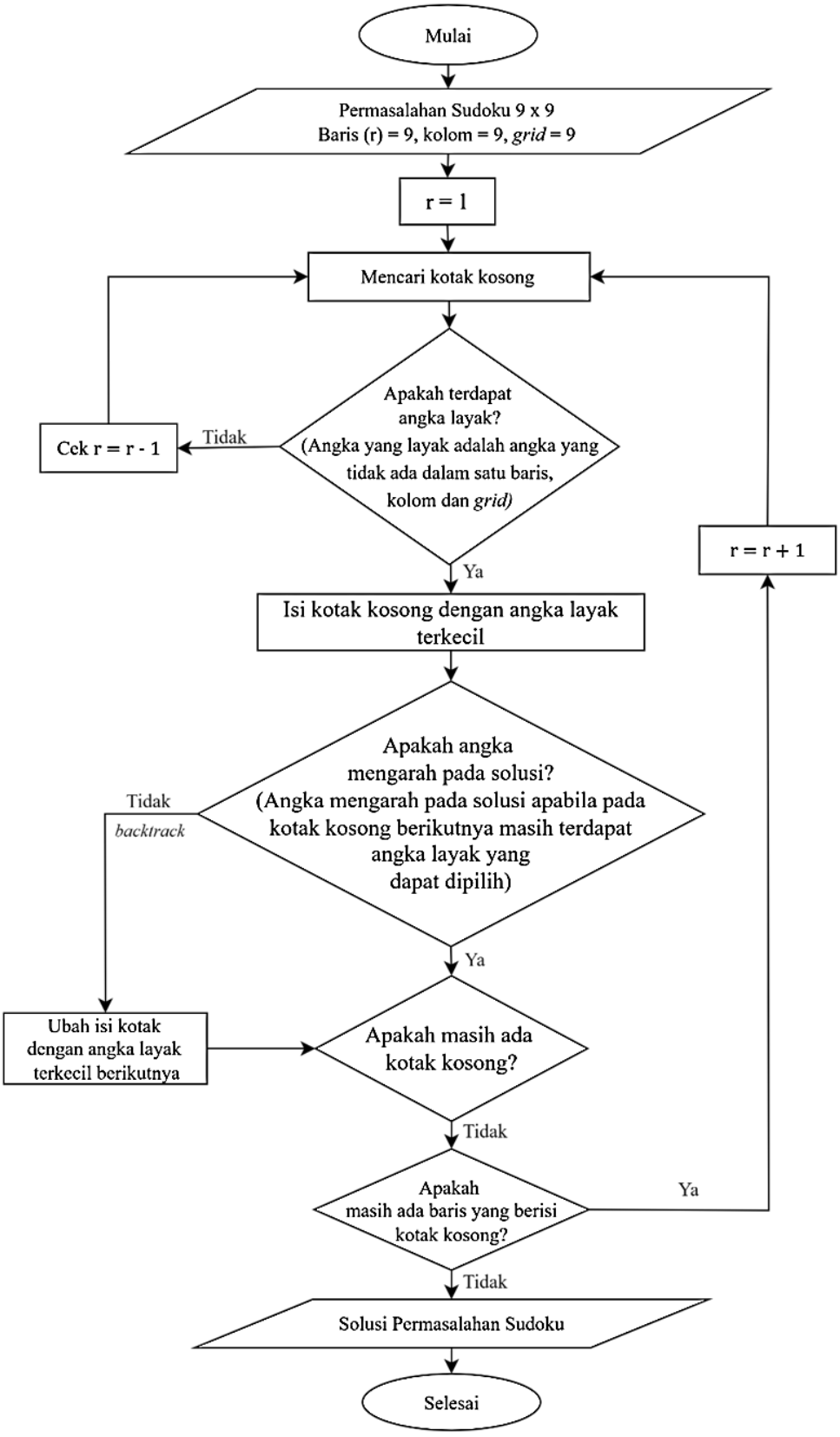
1.6 Metodologi Penelitian

Pemasalahan sudoku dalam penelitian ini berukuran 9×9 yang diambil dari aplikasi Sudoku versi 4.6.1 pada tingkatan sulit. Penelitian dimulai dengan tampilan sudoku yang sudah berisi beberapa angka pada papan sudoku berukuran 9×9 pada tingkat sulit. Selanjutnya dilakukan pencarian kotak kosong pada tiap baris, apabila tidak terdapat kotak kosong pada baris tersebut, maka pencarian kotak kosong dilakukan pada baris selanjutnya, tetapi jika masih terdapat kotak kosong pada baris tersebut maka, langkah selanjutnya adalah menentukan angka yang layak. Angka dikatakan layak apabila tidak terdapat angka yang sama pada baris, kolom, dan *grid* (Andina, Noviani, & Yudhi, 2022). Setelah menentukan angka yang layak selanjutnya adalah menyesuaikan angka yang layak dengan baris, kolom, dan *grid*.

Algoritma *Depth First Search* dan algoritma *Backtracking* digunakan untuk menemukan solusi dengan bantuan pohon berakar pada kotak kosong di setiap baris. Hasil akhirnya akan berupa angka yang berurutan sesuai dengan urutan kotak kosong pada setiap baris. Jika sudah ditemukan solusi pada setiap baris, maka semua kotak pada papan sudoku akan terisi secara keseluruhan. Urutan langkah pengerjaan dapat dilihat pada Gambar 1.1 dan Gambar 1.2



Gambar 1. 1 Flowchart Algoritma DFS



Gambar 1. 2 Flowchart Algoritma *Backtracking*