UNIVERSITAS
GADJAH MADA
Order Statistic Tree: Struktur Data Domain Partition Search Tree Termodifikasi
Zaidurrohman, Prof. Dr. Suprapto, M.I.Kom
Universitas Gadjah Mada, 2026 | Diunduh dari http://etd.repository.ugm.ac.id/

DAFTAR ISI

Halaman Judul ii

Halaman Pengesahan iv

Halaman Pernyataan v

Halaman Persembahan v

PRAKATA vi

DAFTAR ISI vii

DAFTAR TABEL x

DAFTAR GAMBAR xi

DAFTAR ALGORITMA xiii

INTISARI 1

ABSTRACT 2

I PENDAHULUAN 1

1.1 Latar Belakang 1
1.2 Rumusan Masalah 2
1.3 Batasan Masalah 3
1.4 Tujuan Penelitian 3
1.5 Manfaat Penelitian 3

II TINJAUAN PUSTAKA 6

2.1 Struktur Data 6
2.2 Struktur Data *Tree* 6

III LANDASAN TEORI 10

3.1 Struktur Data 10
3.2 Graf 10

UNIVERSITAS
GADJAH MADA
Order Statistic Tree: Struktur Data Domain Partition Search Tree Termodifikasi
Zaidurrohman, Prof. Dr. Suprapto, M.I.Kom
Universitas Gadjah Mada, 2026 | Diunduh dari http://etd.repository.ugm.ac.id/

3.3 *Tree* . . . 13
3.3.1 *Binary Search Tree* . . . 14
3.3.2 *Domain Partition Search Tree* . . . 17
3.3.3 *Order Statistic Tree* . . . 21

IV ANALISIS DAN PERANCANGAN STRUKTUR DATA **23**
4.1 Analisis Kebutuhan Operasi . . . 23
4.2 Perancangan Struktur Data . . . 23
4.2.1 Modifikasi *Node Class* . . . 24
4.2.2 Modifikasi Prosedur *Insert* . . . 25
4.2.3 Modifikasi Prosedur *Delete* . . . 29
4.2.4 Rancangan Prosedur *Rank* . . . 31
4.2.5 Rancangan Prosedur *Select* . . . 35

V IMPLEMENTASI RANCANGAN STRUKTUR DATA **37**
5.1 *Domain Class* . . . 37
5.2 *Node Class* . . . 39
5.3 *Tree Class* . . . 40
5.4 Prosedur *Search* . . . 41
5.5 Prosedur *Insert* . . . 42
5.6 Prosedur *Delete* . . . 45
5.7 Prosedur *Rank* . . . 47
5.8 Prosedur *Select* . . . 47

VI HASIL DAN PEMBAHASAN **48**
6.1 Analisis Matematis . . . 48
6.1.1 Konsumsi Memori . . . 48
6.1.2 Konsumsi Waktu . . . 52
6.2 Pengujian Empiris . . . 54
6.2.1 Konsumsi Memori . . . 55
6.2.2 Konsumsi Waktu . . . 56

VIIKESIMPULAN DAN SARAN **60**
7.1 Kesimpulan . . . 60
7.2 Saran . . . 60

UNIVERSITAS
GADJAH MADA
Order Statistic Tree: Struktur Data Domain Partition Search Tree Termodifikasi
Zaidurrohman, Prof. Dr. Suprapto, M.I.Kom
Universitas Gadjah Mada, 2026 | Diunduh dari http://etd.repository.ugm.ac.id/

DAFTAR PUSTAKA **61**

UNIVERSITAS
GADJAH MADA
Order Statistic Tree: Struktur Data Domain Partition Search Tree Termodifikasi
Zaidurrohman, Prof. Dr. Suprapto, M.I.Kom
Universitas Gadjah Mada, 2026 | Diunduh dari http://etd.repository.ugm.ac.id/

DAFTAR TABEL

2.1 Perbandingan performa *search tree* 9

4.1 Perbandingan dukungan operasi pada *DPS Tree* dan *Order Statistic Tree* 23

6.1 Penghitungan konsumsi memori satu buah *node* 49
6.2 Kompleksitas memori dan waktu *Order Statistic Tree* 54

UNIVERSITAS
GADJAH MADA

Order Statistic Tree: Struktur Data Domain Partition Search Tree Termodifikasi
Zaidurrohman, Prof. Dr. Suprapto, M.I.Kom
Universitas Gadjah Mada, 2026 | Diunduh dari http://etd.repository.ugm.ac.id/

DAFTAR GAMBAR

3.1 Tiga contoh graf . . . 11
3.2 *Directed* dan *undirected graph* . . . 11
3.3 *Connected* dan *disconnected graph* . . . 12
3.4 *Strongly* dan *weakly connected graph* . . . 12
3.5 *Weighted graph* . . . 13
3.6 *Cyclic* dan *acyclic graph* . . . 13
3.7 *Rooted tree* dan *subtree* (Munir, 2005) . . . 14
3.8 *Perfect binary tree* . . . 15
3.9 *Unbalanced binary tree* dan *balanced binary tree* . . . 15
3.10 *Degenerate* dan *skewed binary tree* . . . 16
3.11 *Range* dan *value* pada *DPS Tree* . . . 18

4.1 Ilustrasi *node* pada *Order Statistic Tree* . . . 25
4.2 Diagram alur *insert* . . . 26
4.3 Ilustrasi menambahkan elemen 10 dengan prosedur *insert* . . . 28
4.4 Diagram alur *delete* . . . 30
4.5 Ilustrasi menghapus elemen 32 dengan prosedur *delete* . . . 33
4.6 Diagram alur perhitungan *rank* sebuah *node* . . . 34
4.7 Ilustrasi mencari urutan dari elemen 20 dengan prosedur *rank* . . . 34
4.8 Diagram alur proses *select* . . . 35
4.9 Ilustrasi mencari elemen terkecil ke-7 dengan prosedur *select* . . . 36

5.1 Kode sumber *domain class* pada *DPS Tree* dan *Order Statistic Tree* . . . 38
5.2 Kode sumber *node class* pada *DPS Tree* . . . 39
5.3 Kode sumber *node class* pada *Order Statistic Tree* . . . 40
5.4 Kode sumber *tree class* pada *Order Statistic Tree* . . . 41
5.5 Kode sumber prosedur *search* pada *DPS Tree* dan *Order Statistic Tree* 42
5.6 Kode sumber prosedur *insert* pada *DPS Tree* . . . 43
5.7 Kode sumber prosedur *insert* pada *Order Statistic Tree* . . . 44
5.8 Kode sumber prosedur *delete* pada *DPS Tree* . . . 45
5.9 Kode sumber prosedur *delete* pada *Order Statistic Tree* . . . 46
5.10 Kode sumber prosedur *rank* pada *Order Statistic Tree* . . . 47
5.11 Kode sumber prosedur *select* pada *Order Statistic Tree* . . . 47

UNIVERSITAS
GADJAH MADA
Order Statistic Tree: Struktur Data Domain Partition Search Tree Termodifikasi
Zaidurrohman, Prof. Dr. Suprapto, M.I.Kom
Universitas Gadjah Mada, 2026 | Diunduh dari http://etd.repository.ugm.ac.id/

6.1 Hasil pengujian empiris konsumsi memori 55
6.2 Hasil pengujian empiris konsumsi waktu 57
6.3 Perbandingan hasil pengujian empiris konsumsi waktu operasi *rank* dan *select* 58
6.4 Hasil pengujian empiris rata-rata kedalaman *node* 59

UNIVERSITAS
GADJAH MADA
Order Statistic Tree: Struktur Data Domain Partition Search Tree Termodifikasi
Zaidurrohman, Prof. Dr. Suprapto, M.I.Kom
Universitas Gadjah Mada, 2026 | Diunduh dari http://etd.repository.ugm.ac.id/

DAFTAR ALGORITMA

1 Prosedur *search* pada *DPS Tree* (Nurrokhman, 2023) 19
2 Prosedur *insert* pada *DPS Tree* (Nurrokhman, 2023) 20
3 Prosedur *delete* pada *DPS Tree* (Nurrokhman, 2023) 21

4 Prosedur *insert* pada *Order Statistic Tree* 27
5 Prosedur *delete* pada *Order Statistic Tree* 31
6 Prosedur *rank* pada *Order Statistic Tree* 32
7 Prosedur *select* pada *Order Statistic Tree* 36

UNIVERSITAS
GADJAH MADA

Order Statistic Tree: Struktur Data Domain Partition Search Tree Termodifikasi
Zaidurrohman, Prof. Dr. Suprapto, M.I.Kom
Universitas Gadjah Mada, 2026 | Diunduh dari http://etd.repository.ugm.ac.id/

INTISARI

Order Statistic Tree: Struktur Data *Domain Partition Search Tree* Termodifikasi

Oleh

ZAIDURROHMAN
21/477632/PA/20678

Pada tahun 2023, diusulkan sebuah struktur data bernama *Domain Partition Search Tree*, yang disingkat *DPS Tree*. *DPS Tree* merupakan salah satu bentuk *binary search tree*. Menggunakan skema partisi *domain*, *DPS Tree* dijamin selalu memiliki kedalaman logaritmik tanpa memerlukan skema *balancing* sebagaimana bentuk *binary search tree* lain. Dari segi pengimplementasian kode program pada penelitian terdahulu, *DPS Tree* dirancang untuk menyimpan nilai unik (*set*) sehingga belum dapat digunakan untuk menyelesaikan permasalahan yang memerlukan kemampuan menyimpan nilai duplikat (*multiset*).

Dalam penelitian ini dikembangkan struktur data *Order Statistic Tree*, yaitu sebuah struktur data hasil modifikasi terhadap struktur data *DPS Tree* agar dapat menyimpan nilai duplikat dan dapat melakukan operasi *rank* dan *select* secara efisien. Modifikasi utamanya adalah menambahkan dua atribut baru pada *node*, yaitu atribut *count* dan *size*. Atribut *count* pada suatu *node* menyimpan banyaknya nilai duplikat yang disimpan *node* tersebut, sedangkan atribut *size* pada suatu *node* menyimpan banyaknya data yang tersimpan pada *subtree* yang berakar di *node* tersebut.

Dengan modifikasi yang dilakukan, *Order Statistic Tree* dapat digunakan untuk menyimpan nilai duplikat dan dapat dikenai operasi *rank* dan *select*. *Order Statistic Tree* memiliki kompleksitas memori $O(n)$ pada kasus terburuk, $\Omega(1)$ pada kasus terbaik, dan $\Theta(\min(n, |R|))$ pada kasus rata-rata, sedangkan kompleksitas waktunya $O(\min(n, \log |R|))$ pada kasus terburuk, $\Omega(1)$ pada kasus terbaik, dan $\Theta(\min(\log n, \log |R|))$ pada kasus rata-rata, di mana $|R|$ adalah ukuran *domain* pada *root*.

Kata-kata kunci : struktur data, kompleksitas, pohon biner, *order statistic*.

UNIVERSITAS
GADJAH MADA
Order Statistic Tree: Struktur Data Domain Partition Search Tree Termodifikasi
Zaidurrohman, Prof. Dr. Suprapto, M.I.Kom
Universitas Gadjah Mada, 2026 | Diunduh dari http://etd.repository.ugm.ac.id/

ABSTRACT

Order Statistic Tree: A Modified Domain Partition Search Tree Data Structure

By

ZAIDURROHMAN
21/477632/PA/20678

In 2023, Nurrokhman proposed a data structure called the Domain Partition Search Tree, abbreviated as DPS Tree. The DPS Tree is a form of binary search tree that utilizes a domain partitioning scheme, ensuring logarithmic depth without the need for balancing mechanisms found in other binary search tree variants. Based on the implementation in previous research, the DPS Tree was unable to store duplicate values or perform *rank* and *select* operations.

This research develops the Order Statistic Tree, a modified version of the DPS Tree capable of storing duplicate values and supporting *rank* and *select* operations. The primary modification involves adding two new attributes to each node: *count* and *size*. The *count* attribute stores the number of duplicate values held by that specific node, while the *size* attribute represents the total number of data points stored within the subtree rooted at that node.

As a result of these modifications, the Order Statistic Tree successfully handles duplicates and executes *rank* and *select* operations. The data structure maintains a memory complexity of $O(n)$ in the worst case, $\Omega(1)$ in the best case, and $\Theta(\min(n, |R|))$ on average, where $|R|$ denotes the domain size at the root. Its time complexity is $O(\min(n, \log |R|))$ in the worst case, $\Omega(1)$ in the best case, and $\Theta(\min(\log n, \log |R|))$ on average.

Keywords : data structures, complexity, binary tree, order statistic.

UNIVERSITAS
GADJAH MADA
Order Statistic Tree: Struktur Data Domain Partition Search Tree Termodifikasi
Zaidurrohman, Prof. Dr. Suprapto, M.I.Kom
Universitas Gadjah Mada, 2026 | Diunduh dari http://etd.repository.ugm.ac.id/

BAB I

PENDAHULUAN

1.1 Latar Belakang

Kurangnya perhatian terhadap pemilihan struktur data yang tepat sering menyebabkan masalah skalabilitas pada aplikasi. Saat beban data dan jumlah pengguna meningkat, sistem yang tidak efisien akan melambat secara signifikan. Di samping menambah infrastruktur *server* yang mahal, solusi lain yang dapat dilakukan adalah dengan melakukan optimasi struktur data yang digunakan pada sistem.

Salah satu struktur data yang paling umum digunakan saat ini adalah pohon pencarian biner atau *binary search tree (BST)*. Dengan *BST*, pencarian data menjadi lebih cepat karena sistem tidak perlu mengecek semua data satu per satu, melainkan cukup mengikuti jalur tertentu. Semakin rendah atau seimbang tingkat kedalaman pohon tersebut, semakin singkat pula waktu yang dibutuhkan untuk menemukan, memasukkan, dan menghapus data.

Pada tahun 2023, Nurrokhman mengusulkan sebuah struktur data yang diberi nama *Domain Partition Search Tree*, yang disingkat *DPS Tree*. *DPS Tree* merupakan salah satu bentuk *BST* hasil modifikasi dari struktur data *Tree List*. *DPS Tree* mengadopsi keunggulan *Tree List* yaitu memiliki kedalaman logaritmik tanpa memerlukan proses *balancing*. Kedalaman logaritmik *DPS Tree* dicapai melalui skema partisi *domain*. Meskipun *DPS Tree* memiliki kedalaman logaritmik, akan tetapi tidak membuat *DPS Tree* termasuk *balanced BST*, karena terdapat kasus di mana bentuk pohon tidak *balanced*. Setiap *node* di *DPS Tree* memiliki atribut *value*, dua *children*, dan *domain*, yaitu sebuah rentang yang membatasi *value* yang dapat disimpan pada *node* tersebut. *Domain* tersebut kemudian dibagi secara seimbang untuk kemudian menjadi *domain* bagi dua *children*-nya. Kompleksitas kedalaman *DPS Tree* adalah $O(1)$ asimtot terhadap nilai konstan 65 pada kasus terburuk, $\Omega(\log n)$ pada kasus terbaik, dan $\Theta(\log n)$ pada kasus rata-rata, sedangkan kompleksitas memorinya $O(\log n)$. Implementasi *DPS Tree* relatif lebih mudah dan lebih singkat dari *WAVL Tree* karena pada *DPS Tree* tidak diperlukan proses *balancing*, yang mana proses *balancing* berupa rotasi *node* diperlukan pada *WAVL Tree*.

Di samping kelebihan-kelebihan yang dimiliki, *DPS Tree* memiliki beberapa keterbatasan. Keterbatasan tersebut dapat dibagi menjadi dua jenis, yaitu keterba-