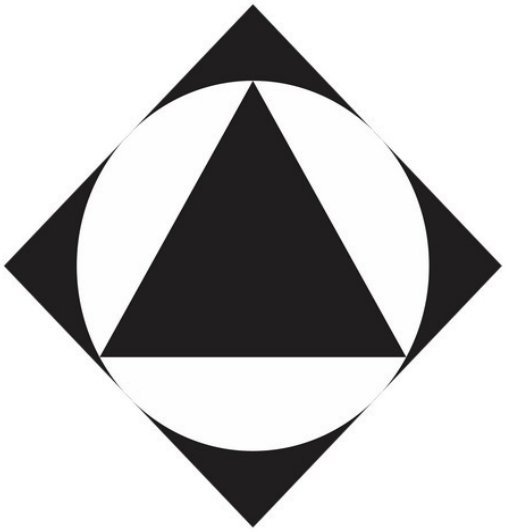
LAPORAN

Implementasi Metode Goal Stack Planning dalam Menyelesaikan Block World Problem Berbasis Web

Disusun untuk memenuhi salah satu tugas mata kuliah IFB-305 Kecerdasan Buatan yang diberikan oleh: Youllia Indrawaty Nurhasanah, S.T., M.T.



Disusun oleh:

Kelompok 10

Cikal Gemintang Seya	15-2022-099
Muhammad Ferran Hilmadiansyah	15-2022-102

Kelas AA

INSTITUT TEKNOLOGI NASIONAL BANDUNG

FAKULTAS TEKNOLOGI INDUSTRI

INFORMATIKA

2022/2023

KATA PENGANTAR

Puji syukur kehadirat Tuhan Yang Maha Kuasa atas segala limpahan rahmat, taufik, dan hidayah-Nya sehingga kami dapat menyelesaikan penyusunan laporan ini guna memenuhi tugas akhir untuk mata kuliah IFB-305 Kecerdasan Buatan, dengan judul "Implementasi Metode Goal Stack Planning dalam Menyelesaikan Block World Problem Berbasis Web".

Laporan ini bertujuan untuk menerapkan kepemahaman kami dengan salah satu materi pembelajaran mata kuliah IFB-305 Kecerdasan Buatan. Kami menyampaikan ucapan terima kasih kepada Ibu Youllia Indrawaty Nurhasanah, S.T., M.T. selaku dosen mata kuliah yang telah memberikan pelajaran, ilmu, wawasan, serta arahan dalam topik ini. Tidak lupa pula kami untuk berterima kasih kepada semua pihak yang telah membantu dalam penyusunan laporan ini.

Tentu laporan ini tidak lepas dari kesalahan dan kekurangan. Oleh karena itu, kami memohon maaf atas segala kesalahan yang tertulis dalam laporan ini, serta mengharapkan masukan dan saran dari pembaca untuk perbaikan di masa depan.

Bandung, 16 Agustus 2023

Kelompok 10

DAFTAR ISI

DAFTAR GAMBAR

BAB 1
PENDAHULUAN

1.1. Latar Belakang

Block World Problem merupakan salah satu masalah klasik dalam bidang kecerdasan buatan. Masalah ini merupakan masalah perencanaan untuk memanipulasi blok-blok agar mencapai keadaan akhir yang diinginkan. Salah satu metode yang dapat digunakan untuk menyelesaikan Block World Problem adalah metode Goal Stack Planning.

Metode Goal Stack Planning awalnya dikembangkan dalam konteks kecerdasan buatan untuk merencanakan tindakan yang terdiri dari serangkaian langkah yang lebih kecil. Pendekatan ini didasarkan pada ide hierarki tujuan, di mana tujuan yang lebih tinggi dapat dipecah menjadi tujuan-tujuan yang lebih spesifik dan tindakan-tindakan yang konkret. Konsep ini berakar dalam ilmu kognitif manusia, di mana pemikiran dan perencanaan manusia sering kali melibatkan hierarki tujuan.

Oleh karena itu, latar belakang ini merangkum sebuah dorongan untuk mengimplementasikan metode Goal Stack Planning dalam lingkup Block World Problem dengan pendekatan berbasis web, dengan harapan dapat menghasilkan kontribusi signifikan dalam bidang pengembangan teknologi dan penelitian perencanaan yang lebih lanjut terutama dalam bidang pengembangan web.

1.2. Rumusan Masalah

Dari latar belakang di atas maka dapat ditarik beberapa rumusan masalah, yaitu:

1. Bagaimana cara menyelesaikan suatu keadaan tumpukan balok dengan keadaan yang diinginkan?
2. Bagaimana cara mengimplementasikan Goal Stack Planning untuk menyelesaikan Block World Problem?
3. Bagaimana cara memvisualisasikan cara penyelesaian Block World Problem dengan metode Goal Stack Planning dalam suatu aplikasi web?

1.3. Batasan Masalah

Dikarenakan lingkup perencanaan kecerdasan buatan dan pengembangan web yang sangat luas, batasan masalah yang dibuatkan untuk laporan ini ialah:

1. Metode Goal Stack Planning adalah sebagaimana yang telah dipelajari dalam mata kuliah IFB-305 Kecerdasan Buatan.
2. Perancangan antarmuka web digunakan untuk memfasilitasi interaksi pengguna, namun tidak melibatkan visualisasi yang rumit.
3. Alternatif penyelesaian dan optimisasi algoritma tidak akan menjadi fokus utama dalam laporan ini.

1.4. Tujuan

Ada pula tujuan dari laporan ini, yaitu:

1. Implementasi metode Goal Stack Planning untuk menyelesaikan Block World Problem.
2. Integrasi antarmuka dengan algoritma perancanaan metode Goal Stack Planning dalam lingkungan web.

1.5. Manfaat

Dari laporan ini bisa didapatkan manfaat untuk umum, khususnya bagi mahasiswa yang akan mempelajari mata kuliah IFB-305 Kecerdasan Buatan, yakni:

1. Pembelajaran interaktif untuk memahami lebih lanjut tentang cara kerja metode Goal Stack Planning dalam menyelesaikan Block World Problem.
2. Penghematan waktu dan upaya dalam penyelesaian Block World Problem dengan menggunakan algoritma pemecah masalah otomatis.

BAB 2
LANDASAN TEORI

2.1. Pengertian Goal Stack Planning

Goal Stack Planning (GSP) adalah sebuah pendekatan dalam bidang kecerdasan buatan (*artificial intelligence*) yang digunakan untuk perencanaan tindakan. Tujuan utama dari GSP adalah merencanakan urutan tindakan yang diperlukan untuk mencapai tujuan tertentu. Pendekatan ini fokus pada representasi hierarkis dari tujuan dan tindakan yang diperlukan untuk mencapai tujuan tersebut.

Dalam GSP, tujuan dan tindakan diwakili dalam bentuk tumpukan (*stack*). Pada awalnya, tumpukan hanya berisi tujuan akhir yang ingin dicapai. Kemudian, tindakan-tindakan yang diperlukan untuk mencapai tujuan tersebut secara bertahap ditambahkan ke tumpukan. Setiap tindakan bisa memiliki sub-tindakan atau subtujuan sendiri, dan tumpukan secara perlahan terisi dengan hierarki tindakan yang semakin rinci.

GSP membantu dalam merencanakan urutan tindakan dengan cara yang sistematis dan terstruktur. Pendekatan ini memungkinkan sistem untuk menguraikan tujuan kompleks menjadi serangkaian tindakan yang lebih sederhana, memperhitungkan ketergantungan antara tindakan-tindakan tersebut. Selain itu, GSP juga dapat menangani situasi di mana beberapa tujuan harus dicapai secara bersamaan atau berurutan.

Namun, perlu dicatat bahwa GSP adalah salah satu dari banyak pendekatan yang ada dalam bidang perencanaan kecerdasan buatan, dan ada pendekatan lain seperti Planning Graphs, STRIPS, dan lain-lain yang digunakan untuk mengatasi masalah perencanaan.

2.2. Cara Mengimplementasikan Goal Stack Planning

Mengimplementasikan Goal Stack Planning (GSP) melibatkan pembuatan program komputer yang dapat merencanakan tindakan (*operation*) untuk mencapai tujuan tertentu menggunakan pendekatan tumpukan tujuan (*goal stack*) dan *operation*.

Di bawah ini adalah hal yang perlu disiapkan untuk mengimplementasikan GSP:

1. Struktur data yang akan digunakan untuk merepresentasikan kondisi (*state*), tumpukan (*stack*), dan baris (*queue*).

2. Mendefinisikan kondisi awal (*initial state*) dan kondisi tujuan (*goal state*) menggunakan struktur data yang telah dibuat secara hierarkis sehingga tidak terjadi konflik di antara kedua kondisi.
3. Membuat fungsi dan prosedur yang akan dibutuhkan untuk memanipulasi kondisi dan tumpukan (*stack*) sehingga tidak memenuhi kode sumber dengan baris yang terulang-ulang.

Setelah itu, mulailah dengan menerapkan logika GSP. Berikut adalah langkah umum yang dapat diikuti dalam implementasi:

a) Inisialisasi tumpukan dengan tujuan akhir.
b) Selama tumpukan tujuan tidak kosong:
    - Ambil tujuan teratas dari tumpukan.
    - Jika tujuan tersebut sudah tercapai atau merupakan tindakan, lanjutkan ke tujuan berikutnya.
    - Jika tujuan tersebut adalah tujuan yang kompleks, tambahkan subtujuan atau tindakan yang diperlukan ke tumpukan.

Mengimplementasikan GSP memerlukan pemahaman yang kuat tentang konsep perencanaan kecerdasan buatan dan kemampuan pemrograman. Tidak disarankan untuk yang baru mengenal konsep ini karena memerlukan waktu untuk memahami prinsip-prinsip dasar sebelum memulai implementasi.

2.3. Teknologi yang Digunakan

2.3.1. HTML

Menurut Hidayatullah dan Kawistara (2015) dalam jurnal Fitri Ayu dan Nia Permata Sari (ISSN:2549-0222) “Hypertext Merkup Languange (HTML) adalah bahasa standar yang digunakan untuk menampilkan halaman web”.

HTML5 (Hypertext Markup Language version 5) adalah sebuah bahasa markah yang menstrukturkan isi dari World Wide Web, sebuah teknologi utama pada internet. Standar HTML5 menyempurnakan elemen-elemen lama yang terdapat pada standar sebelumnya, menambahkan elemen-elemen yang lebih semantik dan menambahkan fitur-fitur baru untuk mendukung pembuatan aplikasi web yang lebih kompleks.

2.3.2. JavaScript

JavaScript adalah bahasa pemrograman tingkat tinggi yang paling umum digunakan dalam pengembangan web. JavaScript digunakan untuk mengontrol perilaku interaktif pada sisi klien (browser) dalam pengembangan web. JavaScript umumnya dieksekusi pada browser klien dan digunakan untuk membuat tampilan dinamis, mengelola interaksi pengguna, dan melakukan permintaan ke server.

2.3.3. Node.js

Node.js adalah sistem perangkat lunak dan lingkungan *runtime* yang dibangun di atas mesin JavaScript V8 milik Google sehingga memungkinkan eksekusi JavaScript di sisi server. Node.js memungkinkan pengembang untuk membuat aplikasi baik *server-side* ataupun *client-side* menggunakan JavaScript, mengatasi banyak permintaan secara efisien, dan mengelola operasi asynchronous melalui event loop.

2.3.4. TypeScript

TypeScript adalah sebuah bahasa pemrograman sumber terbuka (*open source*) yang dikembangkan oleh Microsoft. TypeScript adalah superset dari Javascript, artinya TypeScript memiliki semua fitur yang ada pada JavaScript dengan tambahan beberapa fitur. Beberapa fitur tambahan pada TypeScript adalah dukungan pemrograman berorientasi objek dan tipe statis secara opsional. TypeScript ini dikembangkan dengan tujuan untuk membuat aplikasi dalam skala besar dan akan dikompilasi ke bahasa JavaScript.

TypeScript dan Node.js saling terkait karena TypeScript menghasilkan kode JavaScript. Dalam pengembangan aplikasi Node.js menggunakan TypeScript, kode yang ditulis dalam bahasa TypeScript kemudian dikompilasi menjadi JavaScript agar bisa dieksekusi oleh Node.js. Penggunaan TypeScript dalam pengembangan Node.js membantu dalam mengurangi kesalahan dan meningkatkan kualitas kode.

2.4. Pengertian Block World Terhadap Goal Stack Planning

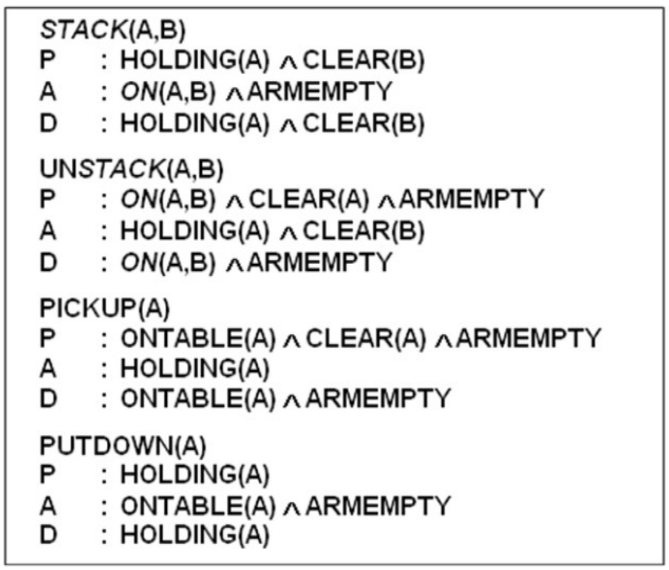
Dalam block world memiliki beberapa balok yang ditempatkan di atas meja. Tujuan Anda adalah mengatur balok-blok ini dalam urutan tertentu. Konsep Goal Stack Planning digunakan untuk merencanakan langkah-langkah yang diperlukan untuk mencapai tujuan ini. Di dalam dunia balok terdapat nama istilah yaitu:

1. Initial State: Keadaan awal balok.
2. Goal State: Tujuan balok yang diinginkan
3. Operator: Langkah-langkah konkret yang dapat diambil untuk mencapai tujuan dalam lingkungan. Misalnya, dalam "dunia balok," beberapa operator mungkin melibatkan:
   a) PICKUP: Mengambil balok dari posisi di atas meja.
   b) PUTDOWN: Meletakkan balok pada posisi yang diinginkan di atas meja.
   c) STACK: Menumpuk balok satu di atas balok lainnya.
   d) UNSTACK: Mengambil balok dari posisi balok di atas balok.
4. Arm: Lengan robot untuk memanipulasi balok.

Di dalam goal stack planning terdapat nama istilah juga, yaitu:

1. Stack: Tumpukan untuk menampung States.
2. Current State: Kondisi saat ini.
3. Queue: Antrian untuk menampung solusi.

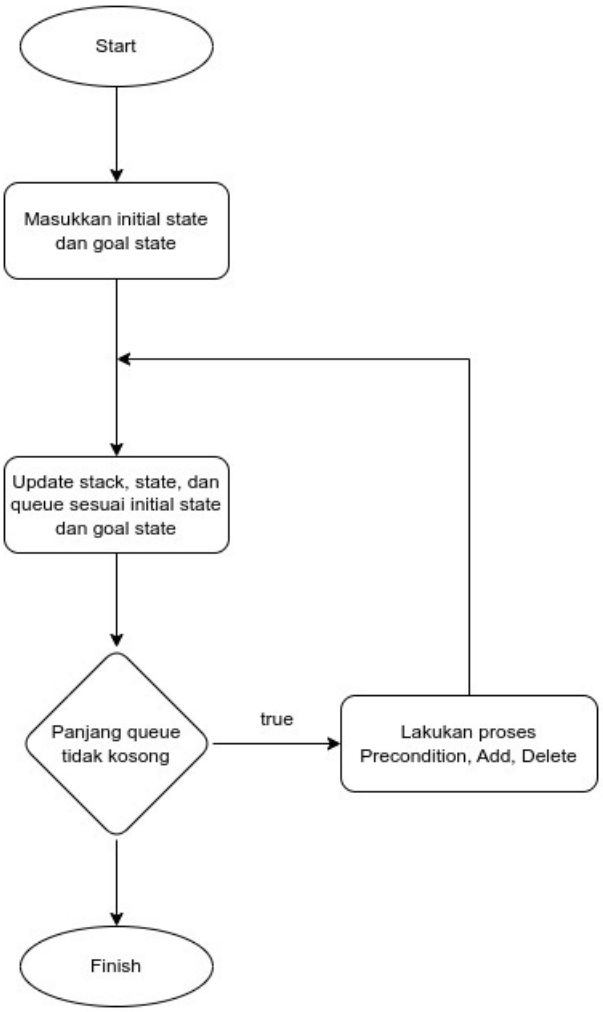
Singkatan PAD ini berasal dari P (Pre-condition), A (Add), dan D (Delete).



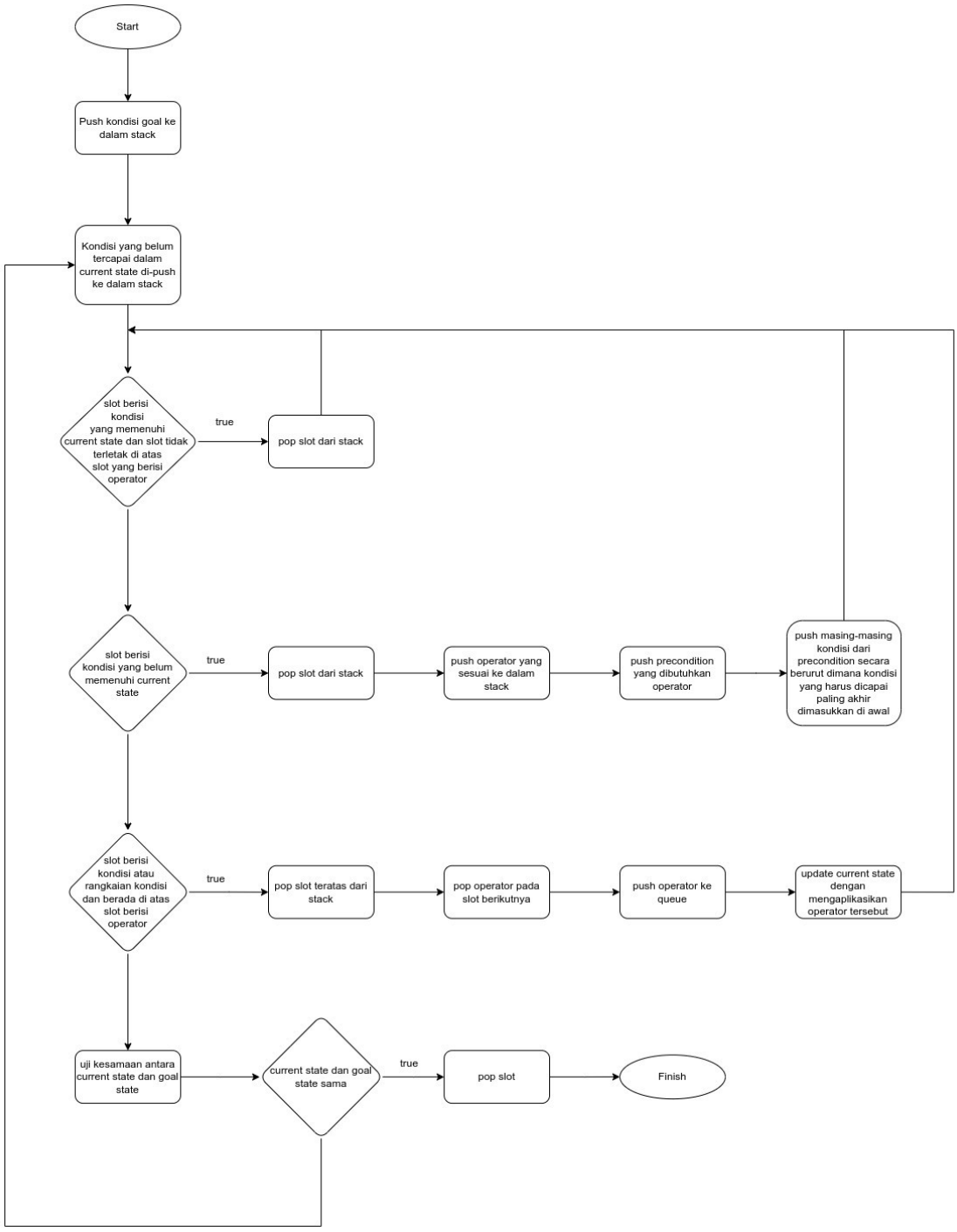
Gambar 1: Tabel PAD (Precondition, Add, Delete)

BAB 3
PERANCANGAN

3.1. Diagram Blok

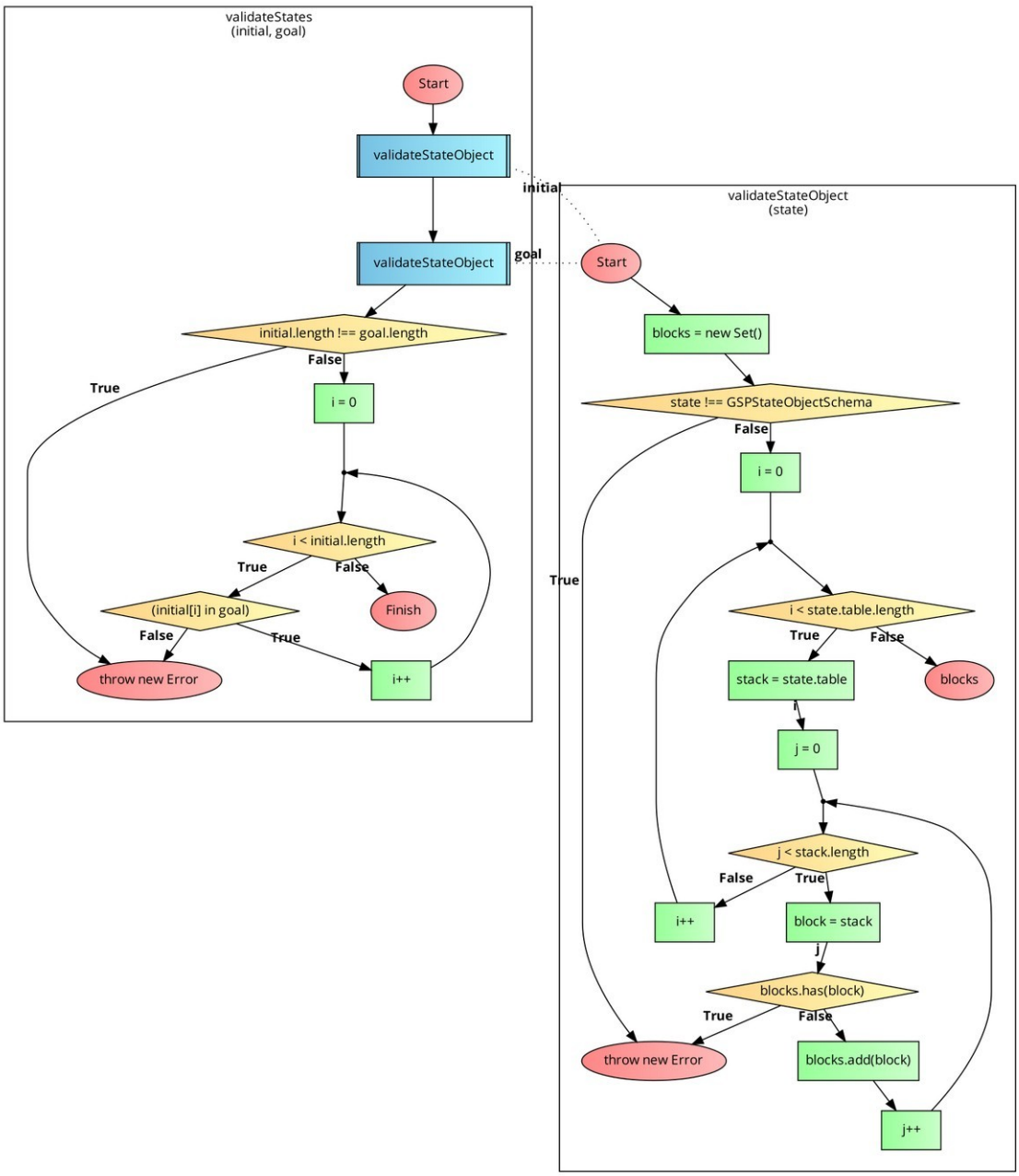


Gambar 2: Diagram blok proses aplikasi

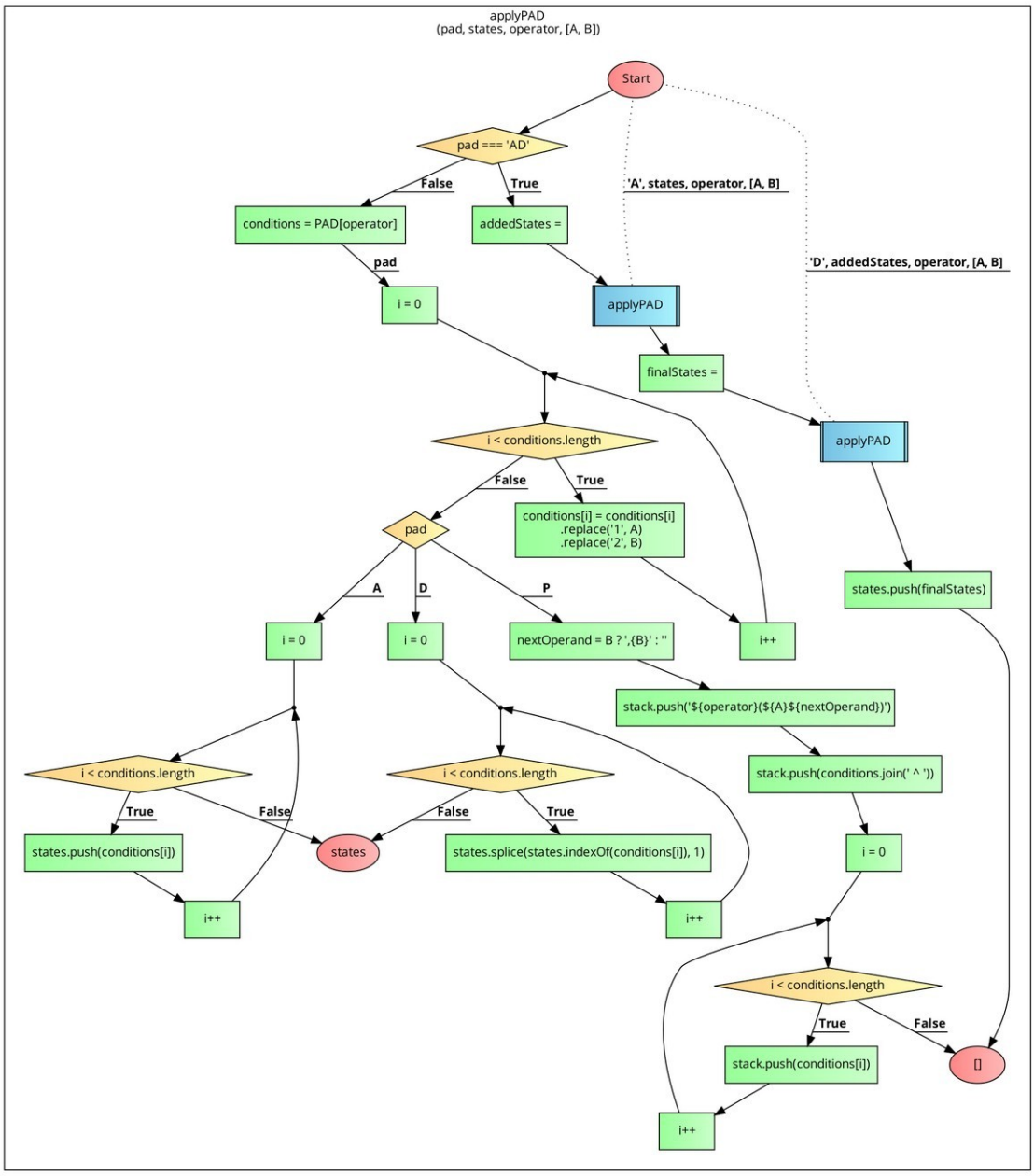


Gambar 3: Diagram blok proses Precondition, Add, Delete (PAD)

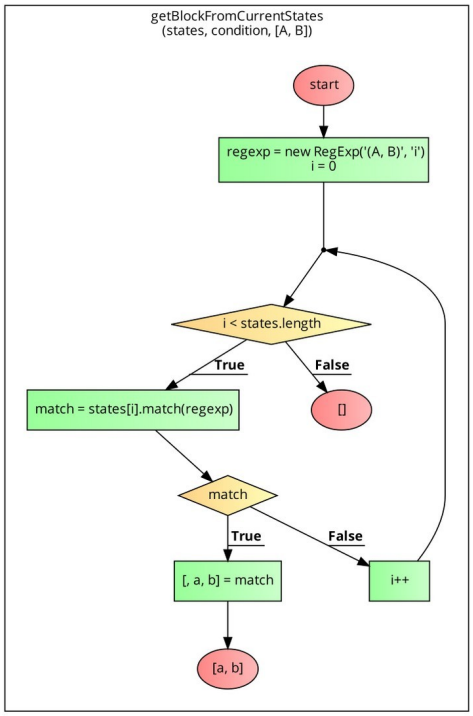
3.2. Flowchart



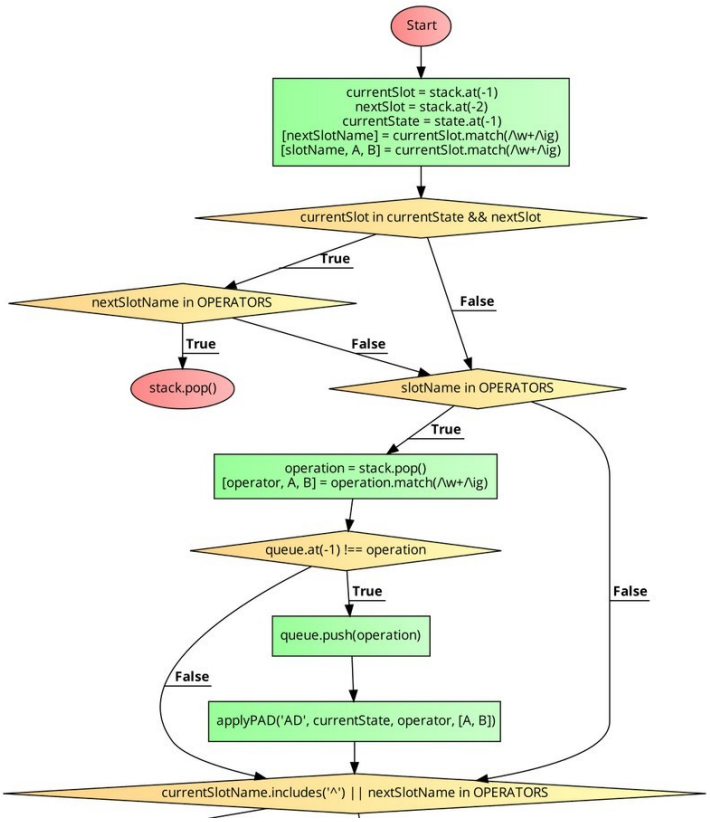
Gambar 4: Diagram alir proses validasi state



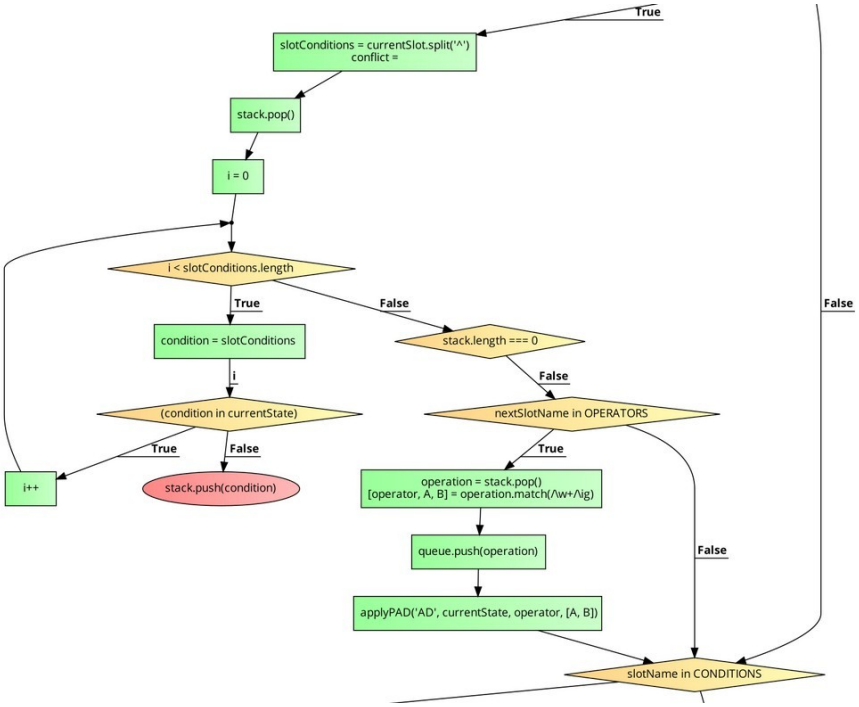
Gambar 5: Diagram alir fungsi applyPAD



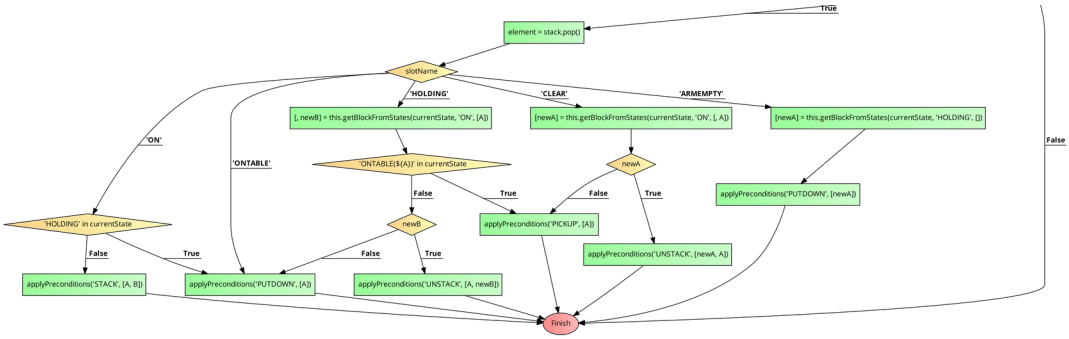
Gambar 6: Diagram alir fungsi getBlockFromCurrentStates



Gambar 7: Diagram alir fungsi nextIteration (1)



Gambar 8: Diagram alir fungsi nextIteration (2)



Gambar 9: Diagram alir fungsi nextIteration (3)

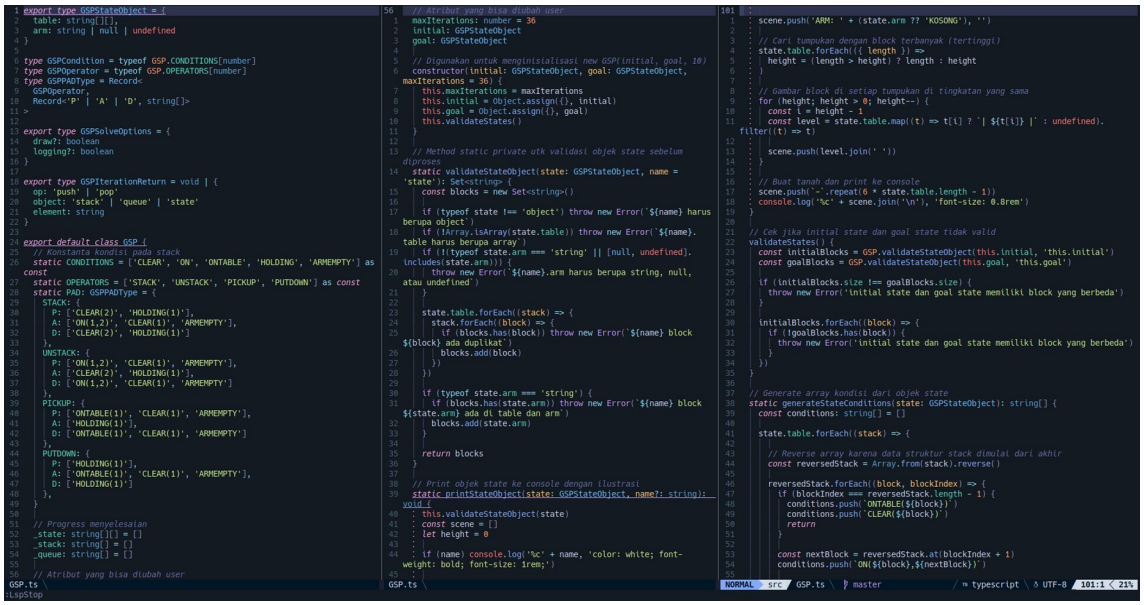
3.3. PAGE (Percept, Action, Goal, Environment)

- Agent Type: Block Puzzle Solver
- Percept: Clear, On, Ontable, Holding, Armempty
- Action: Stack, Unstack, Pickup, Putdown
- Goal: Menyesuaikan initial state dengan goal state
- Environment: Dasar meja, block, arm

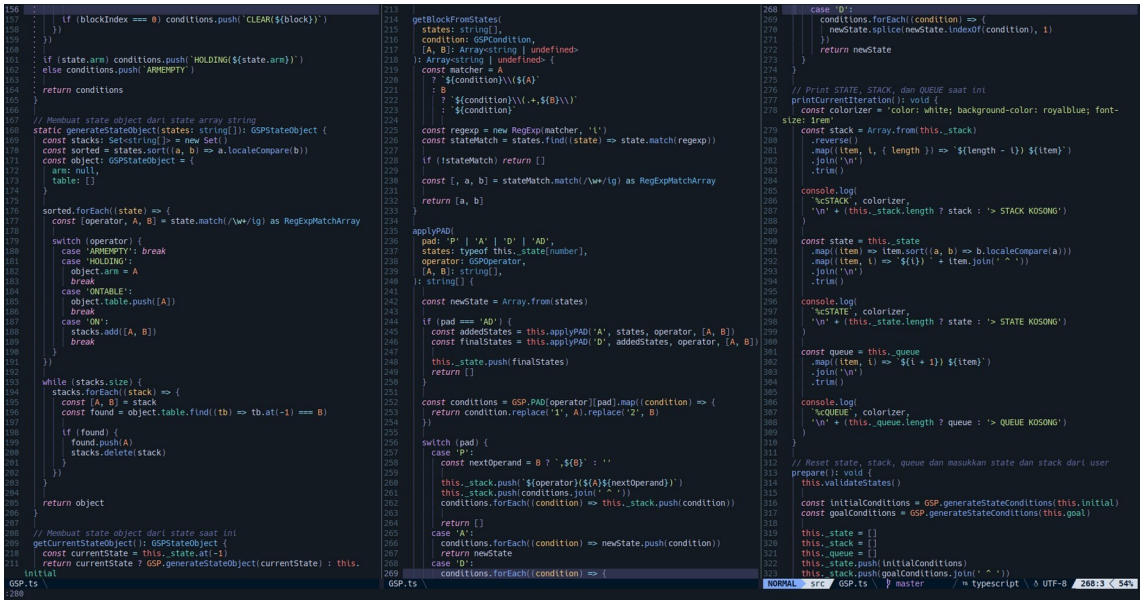
BAB 4
IMPLEMENTASI

4.1. Kode Sumber

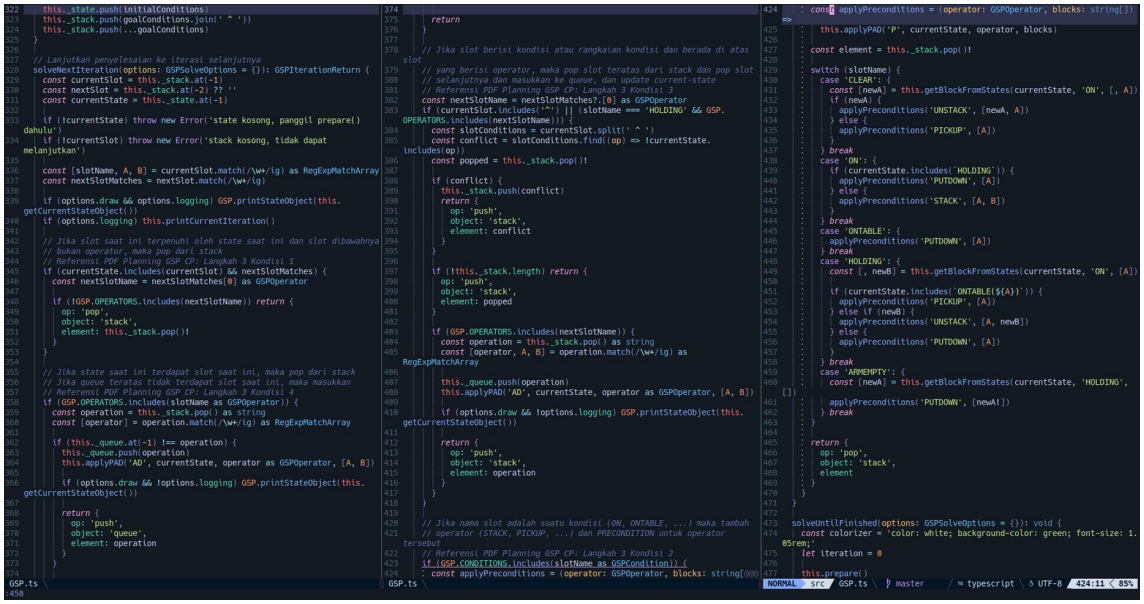
Semua kode sumber tercantum dan tak tercantum di laporan ini dapat diakses dalam repositori GitHub dengan link https://github.com/sglkc/goal-stack-planning.



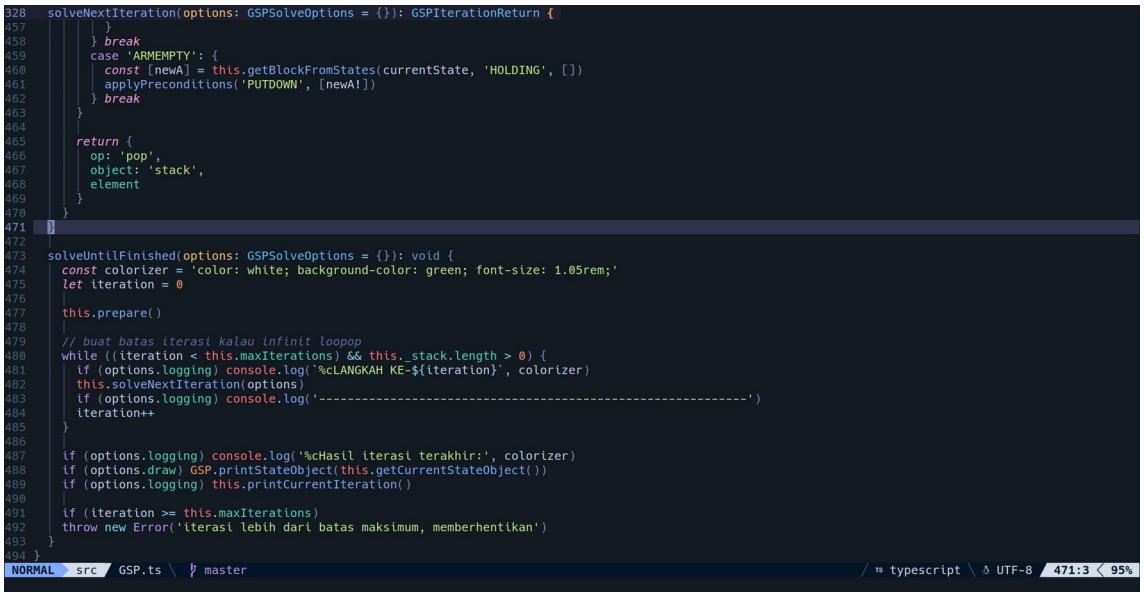
Gambar 10: Kode sumber GSP.ts *(1)*



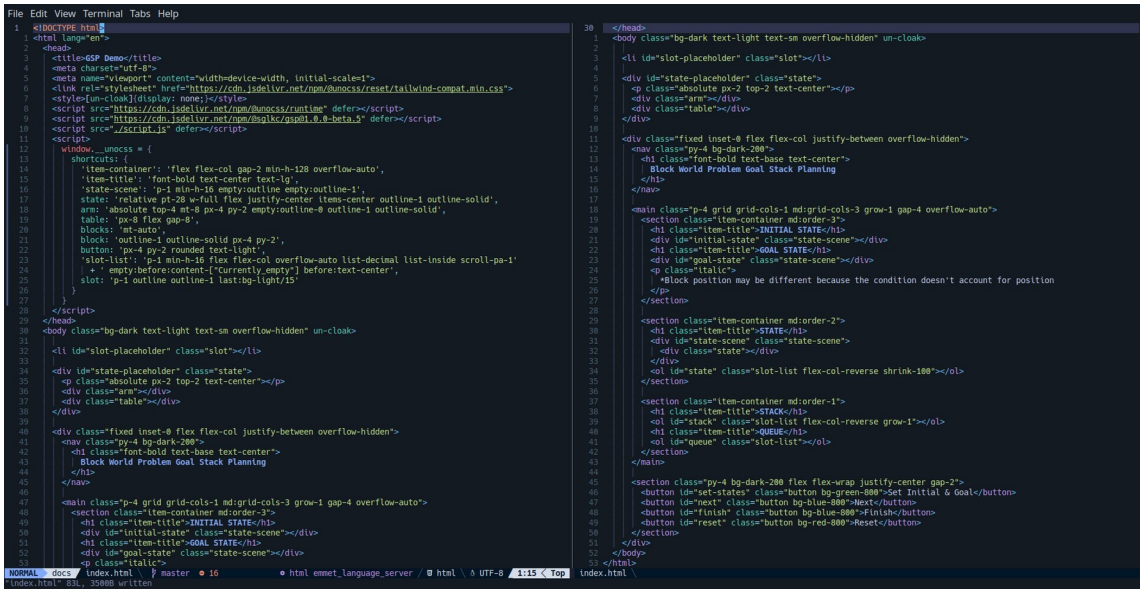
Gambar 11: Kode sumber GSP.ts *(2)*



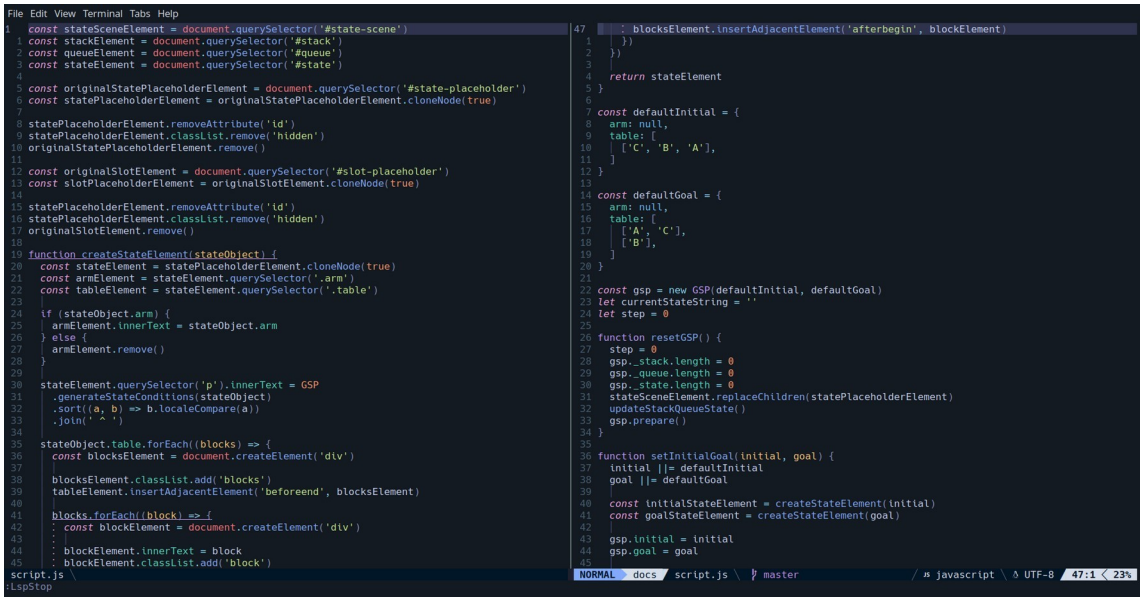
Gambar 12: Kode sumber GSP.ts *(3)*



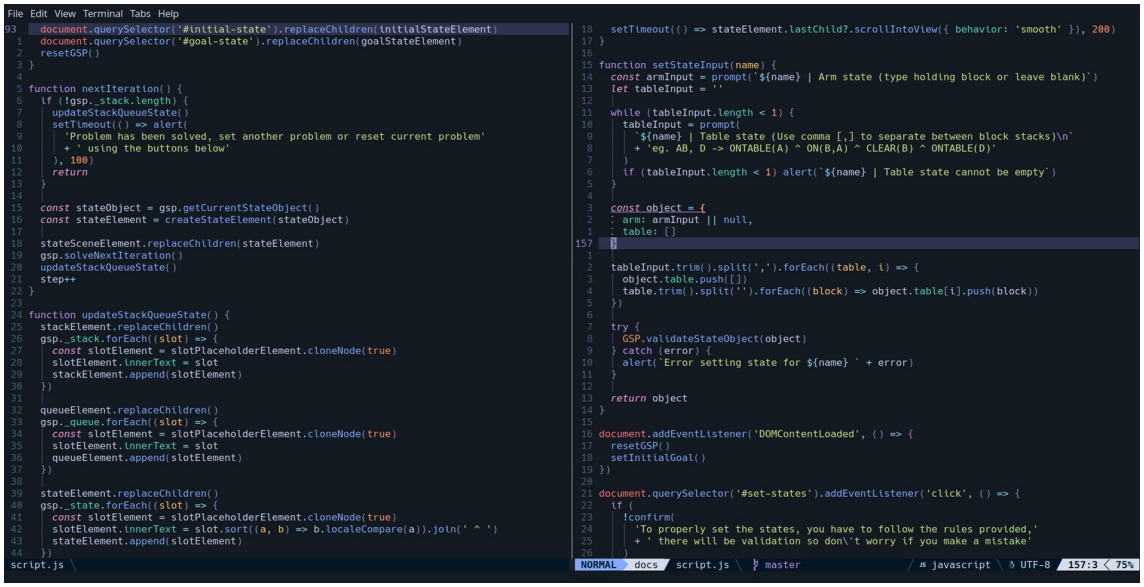
Gambar 13: Kode sumber GSP.ts *(4)*



Gambar 14: Kode sumber index.html



Gambar 15: Kode sumber script.js *(1)*



```
File  Edit  View  Terminal  Tabs  Help
93    document.querySelector('#initial-state').replaceChildren(initialStateElement)
 1    document.querySelector('#goal-state').replaceChildren(goalStateElement)
 2    resetGSP()
 3  }
 4
 5  function nextIteration() {
 6    if (!gsp._stack.length) {
 7      updateStackQueueState()
 8      setTimeout(() => alert(
 9        'Problem has been solved, set another problem or reset current problem'
10        + ' using the buttons below'
11      ), 100)
12      return
13    }
14
15    const stateObject = gsp.getCurrentStateObject()
16    const stateElement = createStateElement(stateObject)
17
18    stateSceneElement.replaceChildren(stateElement)
19    gsp.solveNextIteration()
20    updateStackQueueState()
21    step++
22  }
23
24  function updateStackQueueState() {
25    stackElement.replaceChildren()
26    gsp._stack.forEach((slot) => {
27      const slotElement = slotPlaceholderElement.cloneNode(true)
28      slotElement.innerText = slot
29      stackElement.append(slotElement)
30    })
31
32    queueElement.replaceChildren()
33    gsp._queue.forEach((slot) => {
34      const slotElement = slotPlaceholderElement.cloneNode(true)
35      slotElement.innerText = slot
36      queueElement.append(slotElement)
37    })
38
39    stateElement.replaceChildren()
40    gsp._state.forEach((slot) => {
41      const slotElement = slotPlaceholderElement.cloneNode(true)
42      slotElement.innerText = slot.sort((a, b) => b.localeCompare(a)).join(' ^ ')
43      stateElement.append(slotElement)
44    })
script.js \
```

```
18    setTimeout(() => stateElement.lastChild?.scrollIntoView({ behavior: 'smooth' }), 200)
17  }
16
15  function setStateInput(name) {
14    const armInput = prompt(`${name} | Arm state (type holding block or leave blank)`)
13    let tableInput = ''
12
11    while (tableInput.length < 1) {
10      tableInput = prompt(
 9        `${name} | Table state (Use comma [,] to separate between block stacks)\n`
 8        + 'eg. AB, D -> ONTABLE(A) ^ ON(B,A) ^ CLEAR(B) ^ ONTABLE(D)'
 7      )
 6      if (tableInput.length < 1) alert(`${name} | Table state cannot be empty`)
 5    }
 4
 3    const object = {
 2      arm: armInput || null,
 1      table: []
157    }
 1
 2    tableInput.trim().split(',').forEach((table, i) => {
 3      object.table.push([])
 4      table.trim().split('').forEach((block) => object.table[i].push(block))
 5    })
 6
 7    try {
 8      GSP.validateStateObject(object)
 9    } catch (error) {
10      alert(`Error setting state for ${name} ` + error)
11    }
12
13    return object
14  }
15
16  document.addEventListener('DOMContentLoaded', () => {
17    resetGSP()
18    setInitialGoal()
19  })
20
21  document.querySelector('#set-states').addEventListener('click', () => {
22    if (
23      !confirm(
24        'To properly set the states, you have to follow the rules provided,'
25        + ' there will be validation so don\'t worry if you make a mistake'
26      )
NORMAL  docs  script.js \  master   javascript \  UTF-8  157:3  75%
```

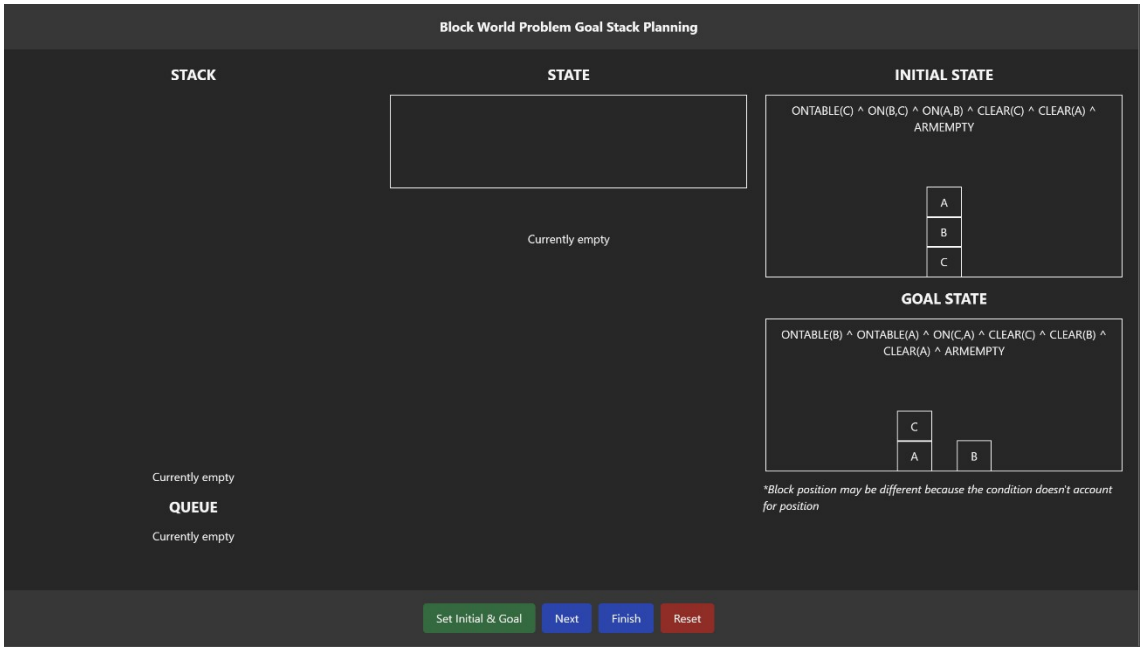
Gambar 16: Kode sumber script.js *(2)*

```
File  Edit  View  Terminal  Tabs  Help
26      !confirm(
25        'To properly set the states, you have to follow the rules provided,'
24        + ' there will be validation so don\'t worry if you make a mistake'
23      )
22    ) return
21
20    let isError = false
19
18    do {
17      isError = false
16
15      try {
14        const initial = setStateInput('INITIAL')
13        const goal = setStateInput('GOAL')
12
11        setInitialGoal(initial, goal)
10      } catch (error) {
 9        isError = true
 8        if (!confirm(`Error setting initial and goal state: ${error}\nRetry?`)) return
 7      }
 6    } while (isError)
 5  })
 4
 3  document.querySelector('#reset').addEventListener('click', resetGSP)
 2  document.querySelector('#next').addEventListener('click', nextIteration)
 1  document.querySelector('#finish').addEventListener('click', () => {
206    if (!gsp._state.length) resetGSP()
 1    while (gsp._stack.length > 0) nextIteration()
 2    nextIteration()
 3  })
NORMAL  docs  script.js \  master   javascript \  UTF-8  206:36  98%
```

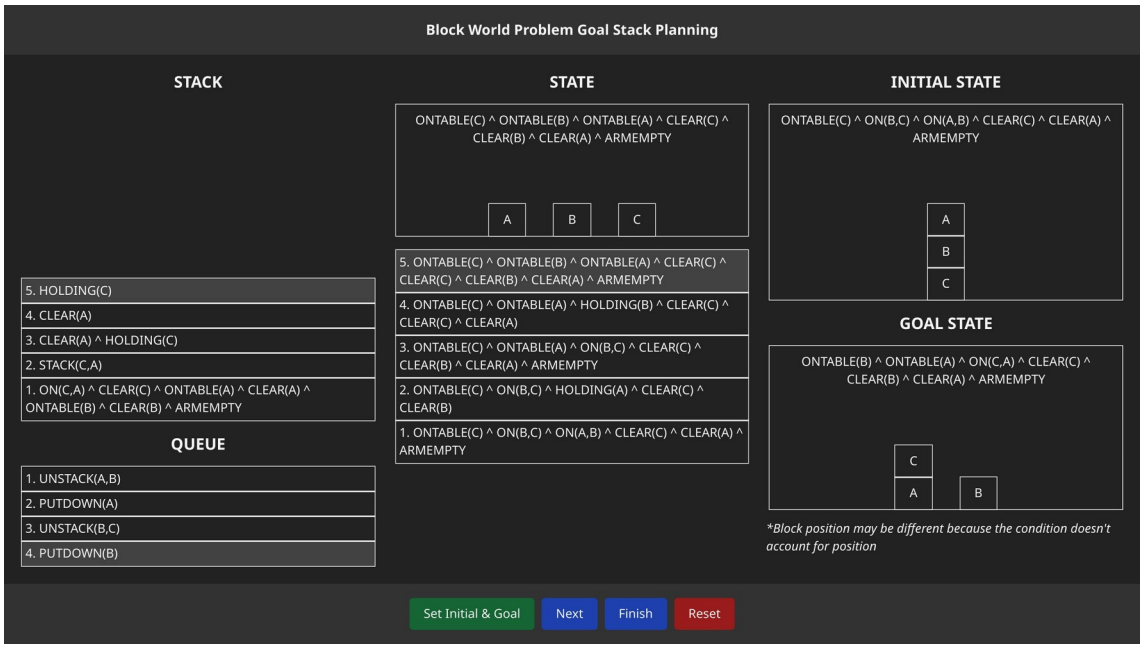
Gambar 17: Kode sumber script.js *(3)*

4.2. Aplikasi

Implementasi aplikasi untuk proyek laporan ini dapat diakses dengan link https://sglkc.github.io/goal-stack-planning/.



Gambar 18: Tangkapan layar implementasi aplikasi



Gambar 19: Tangkapan layar penggunaan aplikasi

BAB 5
PENUTUP

5.1. Kesimpulan

Goal Stack Planning adalah metode perencanaan yang berguna dalam kecerdasan buatan karena memungkinkan sistem untuk merencanakan tindakan-tindakan yang diperlukan secara hierarkis dan terstruktur untuk mencapai tujuan yang kompleks. Dengan memecah tujuan menjadi langkah-langkah yang lebih kecil dan terkelola, sistem dapat menciptakan rencana yang lebih terorganisasi dan dapat dijalankan dengan lebih efisien.

5.2. Saran

Aplikasi masih jauh dari kata sempurna dan aplikasi masih mempunyai beberapa kekurangan, dalam implementasi Goal Stack Planning yang diterapkan dalam proyek ini, terdapat beberapa saran yaitu:

1. Kondisi sebuah state sebaiknya mencakup semua posisi block pada meja.
2. Aplikasi membutuhkan perangkat yang memadai untuk memproses setiap langkah dalam waktu yang singkat.

Maka dari itu kami bersedia mendapatkan kritik dan saran dari pembaca untuk lebih membangun hasil yang lebih baik lagi.

DAFTAR PUSTAKA

Saraswati, N. W. S. (2015). Optimasi Perencanaan Pengambilan Mata Kuliah Dengan Metode Goal Stack Planning. S@ CIES, 5(2), 86-90.

Nugraheni, C. E., & Abednego, L. (2013). Pemodelan Sudoku sebagai Block World Problem. Research Report-Engineering Science, 1.

Rizaldy, R. E. P., Ahmad, U. A., & Dirgantoro, B. (2023). Implementasi REST API Pada Pengembangan Aplikasi Backend Untuk Platform Kursus Online (Growup). eProceedings of Engineering, 10(1).

Iqbal, M., Husni, M., & Studiawan, H. (2012). Implementasi klien sip berbasis web menggunakan Html5 dan Node. Js. Jurnal Teknik ITS, 1(1), A242-A245.

Siahaan, V., & Sianipar, R. H. (2020). *Buku Pintar JavaScript*. Balige Publishing.

Novendri, M. S., Saputra, A., & Firman, C. E. (2019). Aplikasi Inventaris Barang Pada Mts Nurul Islam Dumai Menggunakan Php Dan Mysql. *lentera dumai*, *10*(2).