APLIKASI PENYELESAIAN *PUZZLE* ANGKA MENGGUNAKAN METODE *BI-DIRECTIONAL SEARCH* (BDS) DAN *BI-DIRECTIONAL* A* (BDA*)

MARLA SHEILAMITA SHALIN PIETER[*)]
wawavdp@gmail.com

EVANITA VERONICA MANULLANG, MT[*)]
eva.manullang@gmail.com

MARIANA

[*)]**Staf Pengajar pada Program Studi Teknik Informatika – S1**
Fakultas Ilmu Komputer dan Manajemen (FIKOM)
Universitas Sains dan Teknologi Jayapura (USTJ)

Abstraksi - Permasalahan *Puzzle* Angka merupakan salah satu persoalan klasik dalam bidang studi *Artificial Intelligence* (AI). Problema penyelesaian dalam masalah ini adalah bagian yang paling penting dalam membuat suatu aplikasi. Terutama dalam sebuah aplikasi yang diharapkan untuk memberikan solusi. Dalam mencari cara penyelasaian masalah, terdapat beberapa algoritma yang dapat digunakan. Diantaranya adalah algoritma *Bi-directional search (BDS)* serta *Bi-directional A* (BDA*)*.

Tujuan dari penelitian ini adalah untuk merancang dan membangun suatu perangkat lunak yang mampu untuk menyelesaikan permasalahan dalam permainan Puzzle Angka dengan menggunakan algoritma pencarian *BDS* yang dilakukan dari 2 arah sekaligus, yaitu pencarian maju (*start to goal*), dan pencarian mundur (*goal to start*). Aplikasi dirancang dengan matriks 3x3, 4x4, 5x5 dan dibangun menggunakan bahasa pemograman *Visual Basic* dan *Microsoft Access.*

Hasil dari aplikasi yang dibangun berupa simulasi penyelesaian dari permasalahan permainan *Puzzle* Angka dengan dua algoritma, yaitu *BDS* dan *BDA*.* Dimana dapat digunakan untuk membantu pembelajaran terhadap cara kerja kedua metode dalam pencarian solusi permainan *Puzzle* Angka serta sebagai fasilitas pendukung dalam proses belajar mengajar dalam mata kuliah *Artificial Intelligence*.

Kata Kunci : *Artificial Intelligence, Puzzle Angka, Bi-Directional Search, Bi-Directional A*.*

1. PENDAHULUAN

1.1. Latar Belakang

Kecerdasan Buatan / *Artificial Intelligence* (AI) merupakan cabang dari ilmu komputer yang *concern* dengan pengotomatisasi tingkah laku cerdas. Ini menunjukkan bahwa AI adalah bagian dari komputer sehingga harus didasarkan pada teori suara dan prinsip-prinsip aplikasi dibidangnya. Prinsip-prinsip ini meliputi struktur data yang digunakan dalam representasi pengetahuan, algoritma yang diperlukan untuk mengaplikasikan pengetahuan tersebut,serta bahasa dan teknik pemograman yang yang digunakan dalam mengimplementasikannya. Pada dasarnya, ada 2 teknik pencarian dan pelacakan yang digunakan, yaitu pencarian buta (*blind search*) dan pencarian terbimbing (*heuristic search*). Contoh metode pencarian buta adalah metode *Bi-Directional Search* (BDS) dan metode *Bi-Directional A** (BDA*). Pada setiap iterasi, pencarian BDS dilakukan dari 2 arah sekaligus, yaitu pencarian maju (dari *start* ke *goal*) dan pencarian mundur (dari *goal* ke *start*). Ketika dua arah pencarian telah membangkitkan simpul atau keadaan yang sama, maka solusi telah ditemukan, yaitu dengan cara menggabungkan kedua jalur yang bertemu. Dengan teknik ini, BDS jauh lebih cepat dibandingkan dengan metode pencarian buta lainnya. Metode pencarian dua arah lain di dalam kecerdasan buatan adalah (BDA*). Metode ini juga melakukan pencarian maju dan sekaligus

pencarian mundur, namun hanya keadaan-keadaan terbaik yang dianggap terbaik, atau lebih dekat ke solusi yang akan ditelusuri terlebih dahulu. Untuk membandingkan keadaan mana yang terbaik dan paling dekat ke solusi, BDA* menggunakan fungsi heuristik. Dengan teknik iterasi seperti ini, baik BDS ataupun BDA* akan menemukan langkah-langkah solusi yang terpendek.

Puzzle angka adalah salah satu permasalahan klasik di dalam proses komputasi. *Puzzle* ini ditemukan oleh Sam Lloyad. Ukuran *puzzle* adalah *n* x *n* kotak dengan setiap kotak diberi nomor dari 1 sampai (n^3 – 1), dengan salah satu kotak dibiarkan kosong. Setiap kotak yang bernomor bisa saling berpindah tempat dengan kotak yang kosong, dengan syarat kotak yang kosong berada di sebelah kotak yang bernomor. Pertama, posisi kotak angka teracak sedemikian rupa, dan sasaran dari permainan ini adalah bagaimana mengatur kotak angka sehingga semua kotak berada pada tempatnya yang sesuai dan terurut. Permasalahan *puzzle* angka ini dapat diselesaikan secara cepat dengan menggunakan kecerdasan buatan (*Artificial Intelligence* / AI).

Berdasarkan uraian di atas, penulis bermaksud membangun suatu aplikasi Puzzle Angka dengan besar kotak 3x3, 4x4 dan 5x5 dengan menggunakan metode BDS dan metode BDA* serta membandingkan kecepatan dan efisiensi dari metode yang digunakan. Ini diharapkan dapat memberikan gambaran dan pemahaman terhadap cara kerja kedua metode dalam menyelesaikan masalah *puzzle* angka. Perangkat lunak yang digunakan dalam membangun aplikasi ini adalah Bahasa Pemrograman Visual Basic dan Microsoft Access

1.2. Rumusan masalah

Berdasarkan latar belakang di atas, maka rumusan masalah dalam penelitian ini adalah "Bagaimana menerapkan metode BDS dan BDA* dalam menyelesaikan permaian Puzzle Angka ?"

1.3. Tujuan

Tujuan penelitian ini adalah menghasilkan suatu aplikasi yang dapat menyelesaikan permainan *Puzzle* Angka dengan menggunakan metode BDS dan metode BDA*, dan melakukan perbandingan hasil antara kedua metode.

2. TINJAUAN PUSTAKA

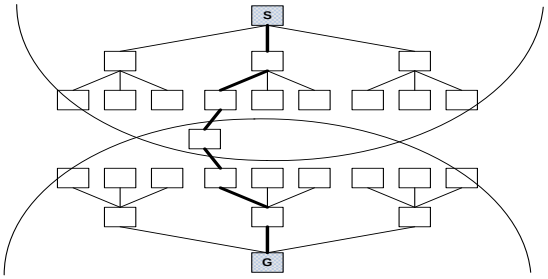
a. Kecerdasan Buatan (*Artificial Intelligence* - AI)

Kecerdasan Buatan sering disebut juga dengan *Artificial Intelligence* (AI). AI merupakan salah satu bagian ilmu komputer yang mempelajari tentang bagaimana caranya agar komputer dapat melakukan pekerjaan seperti yang dapat dilakukan oleh manusia. Pada awal diciptakannya, komputer hanya difungsikan sebagai alat hitung saja. Namun seiring dengan perkembangan zaman, teknologi komputer semakin ditingkatkan dan peran komputer semakin mendominasi kehidupan umat manusia. Komputer tidak lagi hanya digunakan sebagai alat hitung, melainkan komputer diharapkan dapat diberdayakan untuk mengerjakan hal-hal yang dapat memudahkan pekerjaan manusia.

b. Metode Pencarian

a. *Bi-Directional Search* (BDS)

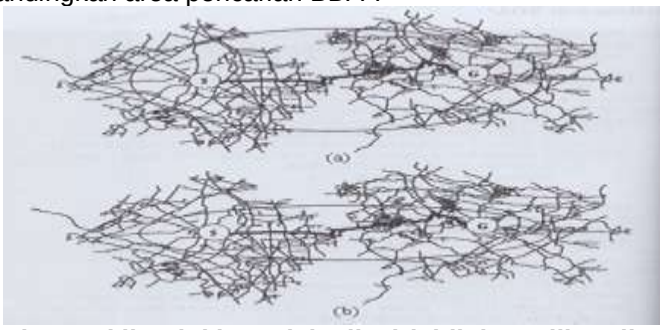
Pada setiap iterasi, pencarian BDS dilakukan dari dua arah, yaitu pencarian maju (*start to goal*) dan pencarian mundur (*goal to start*). Arah pencarian BDS sama dengan arah pencarian BFS. Ketika dua arah pencarian telah membangkitkan simpul yang sama, maka solusi telah ditemukan, yaitu dengan cara menggabungkan kedua jalur yang bertemu. Secara teori, BDS mempunyai harapan yang bagus untuk digunakan, karena hemat waktu maupun memori dan selalu menemukan serta memberikan solusi yang optimal, jika solusinya memang ada. Arah penelusuran metode BDS dapat dilihat pada Gambar 2. Jalur yang ditunjukkan dengan garis tebal adalah jalur solusi dari keadaan awal (*start*) ke keadaan tujuan (*goal*).



Gambar 2.Penelusuran Metode BDS

b. Bi-Directional A* (BDA*)

Jika terdapat hanya satu simpul tujuan, maka algoritma BDA* dapat digunakan, yaitu pencarian heuristik dari dua arah sekaligus, yakni dari simpul asal dan dari simpul tujuan. Implementasi BDA* menggunakan algoritma A*, dan BDA* juga menggunakan fungsi heuristik yang sama dengan A*. Tetapi, algoritma A* harus dimodifikasi sedikit dengan menghilangkan *loop* utama, sehingga algoritma tersebut hanya dikerjakan satu iterasi saja. Setelah itu, iterasi mundur (dari *goal state*) dikerjakan, dan barulah berulang kembali ke iterasi maju. Sama dengan A*, algoritma BDA* adalah *complete* dan optimal. Dibandingkan dengan A* untuk kasus ruang masalah yang besar, BDA* bisa lebih cepat dalam waktu proses dan lebih hemat dalam pemakaian memori karena jumlah simpul yang dibangkitkan lebih sedikit. Jumlah simpul yang dibangkitkan oleh BDA* sekitar setengah dari jumlah simpul yang dibangkitkan oleh A*. Gambar 2.2 berikut mengilustrasikan bagaimana area pencarian A* lebih luas dibandingkan area pencarian BDA*.



Gambar 2. (a) Area Pencarian A* (ditunjukkan oleh elips) lebih luas dibandingkan dengan area pencarian yang dilakukan oleh BDA* (b)

2.2. Permasalahan Puzzle Angka

Permasalahan *Puzzle* Angka, sebenarnya diangkat dari permainan *Puzzle* anak-anak. Contohnya adalah *8-puzzle*. Dalam permainan ini, terdapat 8 buah ubin yang memiliki angka dari 1 sampai 8. Pada tempat yang disediakan, tersedia sebuah tempat kosong untuk menggerakkan ubin. Ubin kemudian diacak dan pemain dituntut untuk mengurutkan kembali angka yang telah teracak. Permasalahan ini termasuk permasalahan ruang keadaan (*state space problem*), karena memiliki keadaan awal (*start state*), aturan main (*rule*), dan keadaan tujuan (*initial state*).

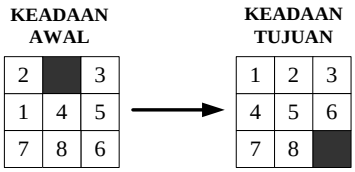
1	2	3
4	5	6
7	8	

Gambar 3. Bentuk 8-Puzzle

3. HASIL DAN PEMBAHASAN

3.1. PENYELESAIAN PUZZLE 8 ANGKA DENGAN METODE BDS dan BDA*

Sebagai contoh, misalkan soal puzzle angka yang ingin diselesaikan, seperti terlihat pada gambar 4 berikut.



Gambar 4. Contoh Keadaan Awal dan Tujuan dari *Puzzle* Angka 3x3

a. Penyelesaian dengan Metode Bi-directional search

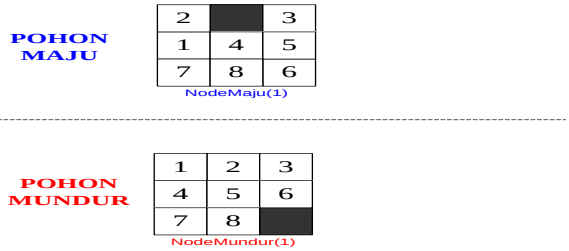
Cara kerja metode metode BDS hampir sama seperti cara kerja metode BFS, dimana semua *node* pada *level* n akan dikunjungi terlebih dahulu sebelum mengunjungi *node-node* pada *level* n+1. Pencarian dimulai dari *node* akar terus ke *level* 1 dari kiri ke kanan, kemudian berpindah ke *level* berikutnya dari kiri ke kanan hingga solusi ditemukan. Hanya saja untuk pencarian BDS, arah pencarian dilakukan dari dua arah sekaligus, yaitu arah maju (dari keadaan awal ke keadaan tujuan) dan arah mundur (dari keadaan tujuan ke keadaan awal). Bila dalam pengembangannya, terdapat keadaan / *node* yang sama di antara kedua pohon yang menelusuri secara maju dan mundur, maka jawaban *puzzle* angka ditemukan dengan menyatukan alur dari kedua pohon tersebut.

Sebagai contoh, misalkan contoh kasus dari keadaan awal dan keadaan tujuan seperti terlihat pada gambar 4.

Maka penyelesaian dengan menggunakan metode BDS adalah sebagai berikut:

1) *Node* pertama dari masing-masing pohon.
   a) Penelusuran Maju:
      NodeMaju-1 = keadaan awal *puzzle* = “[2|0|3][1|4|5][7|8|6]”
   b) Penelusuran Mundur:
      NodeMundur-1 = keadaan awal *puzzle* = “[1|2|3][4|5|6][7|8|0]”

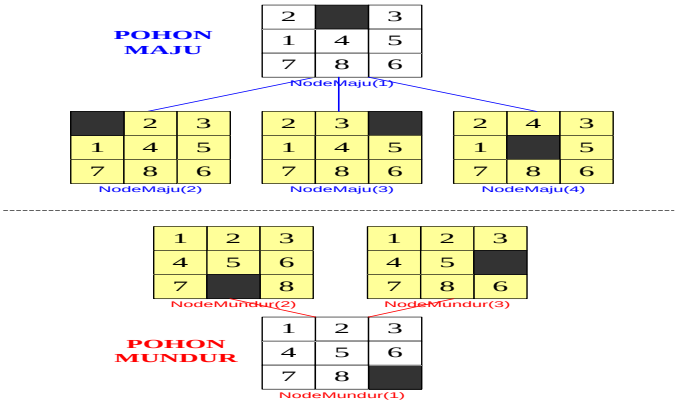
   Illustrasi:



Gambar 5. Illustrasi-1 dari Metode BDS

2) Periksa Node-1
   a) Penelusuran Maju, Keadaan NodeMaju-1 : “[2|0|3][1|4|5][7|8|6]”:
      (a) Geser angka 2 ke kanan, menghasilkan NodeMaju-2 : “[0|2|3][1|4|5][7|8|6]”
      (b) Geser angka 3 ke kiri, menghasilkan NodeMaju-3 : “[2|3|0][1|4|5][7|8|6]”
      (c) Geser angka 4 ke atas, menghasilkan NodeMaju-4 : “[2|4|3][1|0|5][7|8|6]”
   b) Penelusuran Mundur, Keadaan NodeMundur-1 : “[1|2|3][4|5|6][7|8|0]”:
      (a) Geser angka 8 ke kanan, menghasilkan NodeMundur-2 :
      “[1|2|3][4|5|6][7|0|8]”
      (b) Geser angka 6 ke bawah, menghasilkan NodeMundur-3:
      “[1|2|3][4|5|0][7|8|6]”

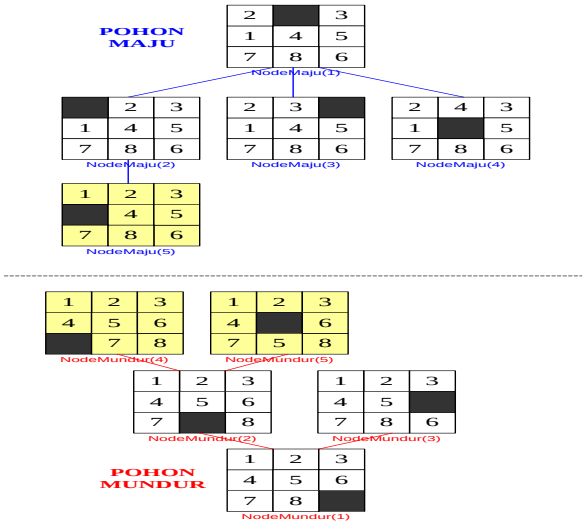
Illustrasi:



Gambar 6. Illustrasi-2 dari Metode BDS

3) Periksa Node-2
   a) Penelusuran Maju, Keadaan NodeMaju-2 : "[0|2|3][1|4|5][7|8|6]":
      Geser angka 1 ke atas, menghasilkan NodeMaju-5 : "[1|2|3][0|4|5][7|8|6]"
   b) Penelusuran Mundur, Keadaan NodeMundur-2 : "[1|2|3][4|5|6][7|0|8]":
      (a) Geser angka 7 ke kanan, menghasilkan NodeMundur-4 :
          "[1|2|3][4|5|6][0|7|8]"
      (b) Geser angka 5 ke bawah, menghasilkan NodeMundur-5:
          "[1|2|3][4|0|6][7|5|8]"

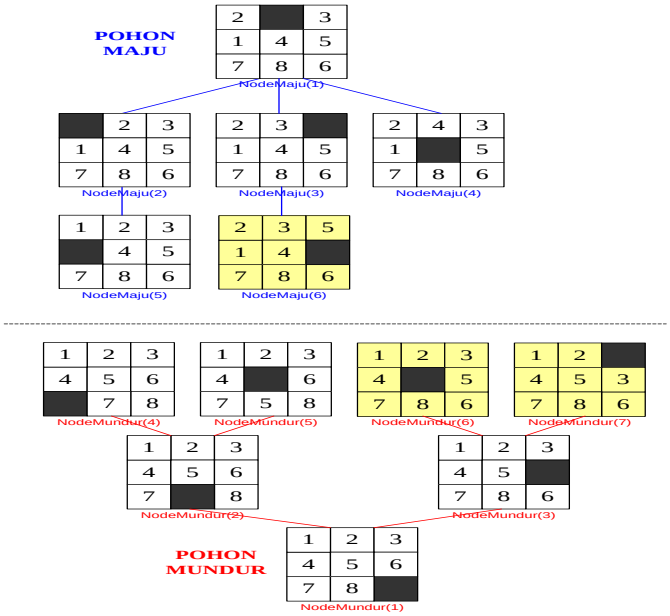
Illustrasi:



Gambar 7. Illustrasi-3 dari Metode BDS

4) Periksa Node-3
   a) Penelusuran Maju, Keadaan NodeMaju-3 : "[2|3|0][1|4|5][7|8|6]":
      Geser angka 5 ke atas, menghasilkan NodeMaju-6 : "[2|3|5][1|4|0][7|8|6]"
   b) Penelusuran Mundur, Keadaan NodeMundur-3 : "[1|2|3][4|5|0][7|8|6]":
      (a) Geser angka 5 ke kanan, menghasilkan NodeMundur-6 :
          "[1|2|3][4|0|5][7|8|6]"
      (b) Geser angka 3 ke bawah, menghasilkan NodeMundur-7:
          "[1|2|0][4|5|3][7|8|6]"

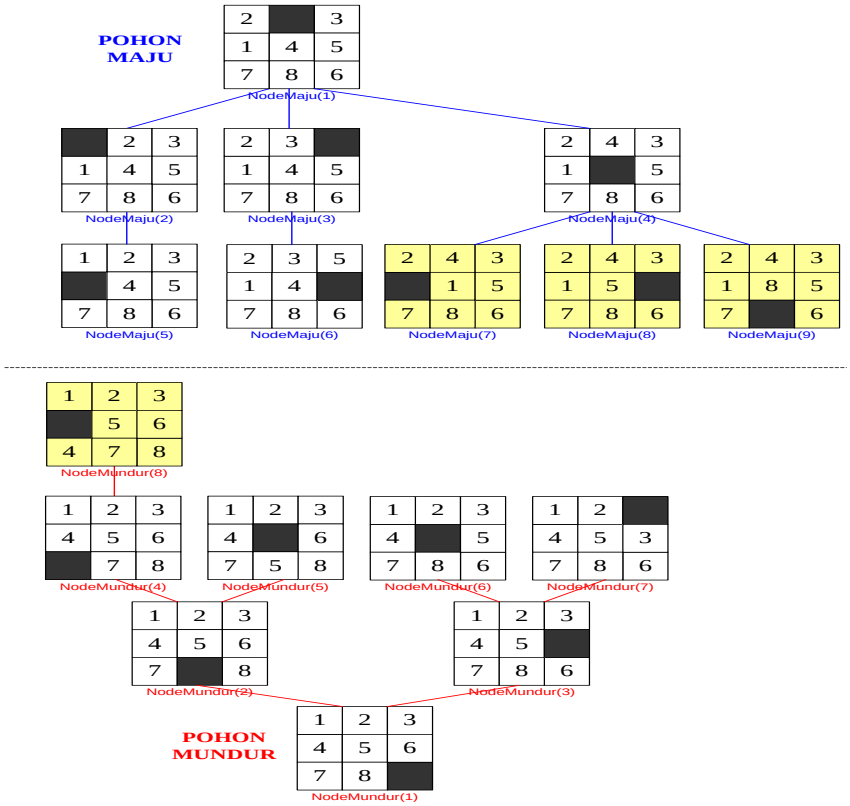
Illustrasi:



Gambar 8. Illustrasi-4 dari Metode BDS

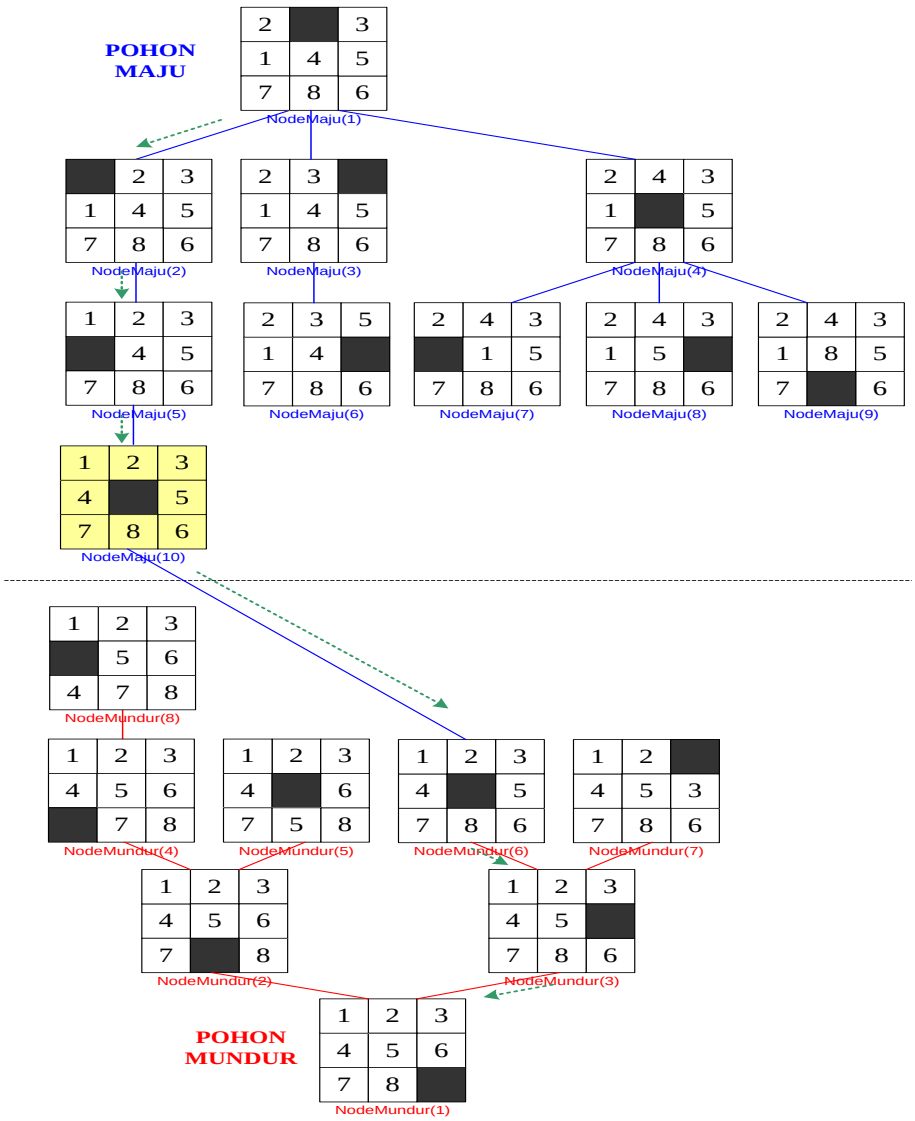
5) Periksa Node-4
   a) Penelusuran Maju, Keadaan NodeMaju-4 : “[2|4|3][1|0|5][7|8|6]”:
      (a) Geser angka 1 ke kanan, menghasilkan NodeMaju-7 : ”[2|4|3][0|1|5][7|8|6]”
      (b) Geser angka 5 ke kiri, menghasilkan NodeMaju-8 : “[2|4|3][1|5|0][7|8|6]”
      (c) Geser angka 8 ke atas, menghasilkan Node-9 : “[2|4|3][1|8|5][7|0|6]”
   b) Penelusuran Mundur, Keadaan NodeMundur-4 : “[1|2|3][4|5|6][0|7|8]”:
      Geser angka 4 ke bawah, menghasilkan NodeMundur-8:
      “[1|2|3][0|5|6][4|7|8]”

Illustrasi:



Gambar 9. Illustrasi-5 dari Metode BDS

6) Periksa Node-5
   Penelusuran Maju, Keadaan NodeMaju-5 : "[1|2|3][0|4|5][7|8|6]":
   Geser angka 4 ke kiri, menghasilkan NodeMaju-10 : "[1|2|3][4|0|5][7|8|6]"
7) Ternyata keadaan pada NodeMaju-10 sama dengan keadaan pada NodeMundur-6. Jawaban ditemukan dan pencarian dihentikan. Alur solusi dari keadaan awal ke keadaan tujuan dapat dilihat pada Gambar 10.



Gambar 10. Illustrasi-6 dari Metode BDS

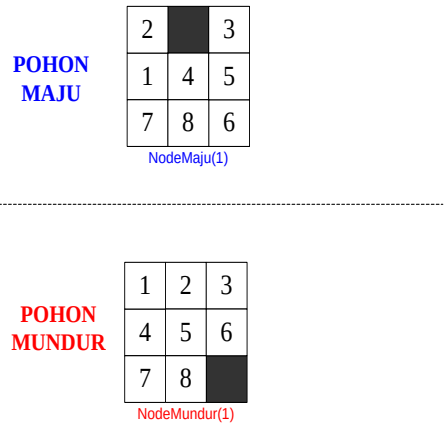
b. Metode Bi-Direction A* Untuk Menyelesaikan *Puzzle Angka*

Arah penelusuran BDA* melakukan penelusuran dari dua arah sekaligus, yaitu penelusuran maju dari keadaan awal ke keadaan tujuan, dan penelusuran mundur dari keadaan tujuan ke keadaan awal. Proses ini diyakini akan mempercepat pencarian, karena akan mengurangi tinggi pohon.

Sebagai contoh, misalkan contoh kasus dari keadaan awal dan keadaan tujuan seperti terlihat pada gambar 4 sebelumnya, maka penyelesaian dengan menggunakan metode BDA* adalah sebagai berikut:

1) *Node* pertama dari masing-masing pohon.
   a) Penelusuran Maju:
      NodeMaju-1 = Keadaan awal *puzzle* = “[2|0|3][1|4|5][7|8|6]”, keadaan tujuan = “[1|2|3][4|5|6][7|8|0]”
   b) Penelusuran Mundur:
      NodeMundur-1,= Keadaan awal *puzzle* = “|1|2|3|4|5|6|7|8|0”, keadaan tujuan = “[2|0|3][1|4|5][7|8|6]”

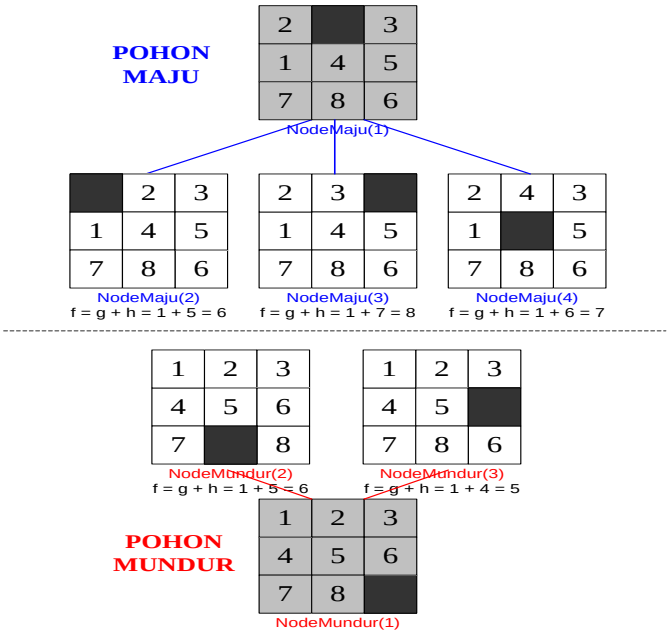
Illustrasi:



Gambar 11. Illustrasi-1 dari Metode BDA*

2) Periksa Node-1
   a) Penelusuran Maju, Keadaan NodeMaju-1 : "[2|0|3][1|4|5][7|8|6]":
      (a) Geser angka 2 ke kanan,
      menghasilkan NodeMaju-2: "[0|2|3][1|4|5][7|8|6]".
      (b) Geser angka 3 ke kiri, menghasilkan NodeMaju-3 : "[2|3|0][1|4|5][7|8|6]".
      (c) Geser angka 4 ke atas, menghasilkan NodeMaju-4 : "[2|4|3][1|0|5][7|8|6]".
   b) Penelusuran Mundur, Keadaan NodeMundur-1 : "[1|2|3][4|5|6][7|8|0]":
      (a) Geser angka 8 ke kanan,
      menghasilkan NodeMundur-2 : "[1|2|3][4|5|6][7|0|8]".
      (b) Geser angka 6 ke bawah, menghasilkan NodeMundur-3: "[1|2|3][4|5|0][7|8|6]".

Illustrasi:



Gambar 12. Illustrasi-2 dari Metode BDA*

3) Periksa *node* yang masih berstatus OPEN (belum dikembangkan keadaan anaknya) dan memiliki fungsi heuristik *f* terkecil.
   a) Penelusuran Maju, Keadaan NodeMaju-2 : "[0|2|3][1|4|5][7|8|6]":

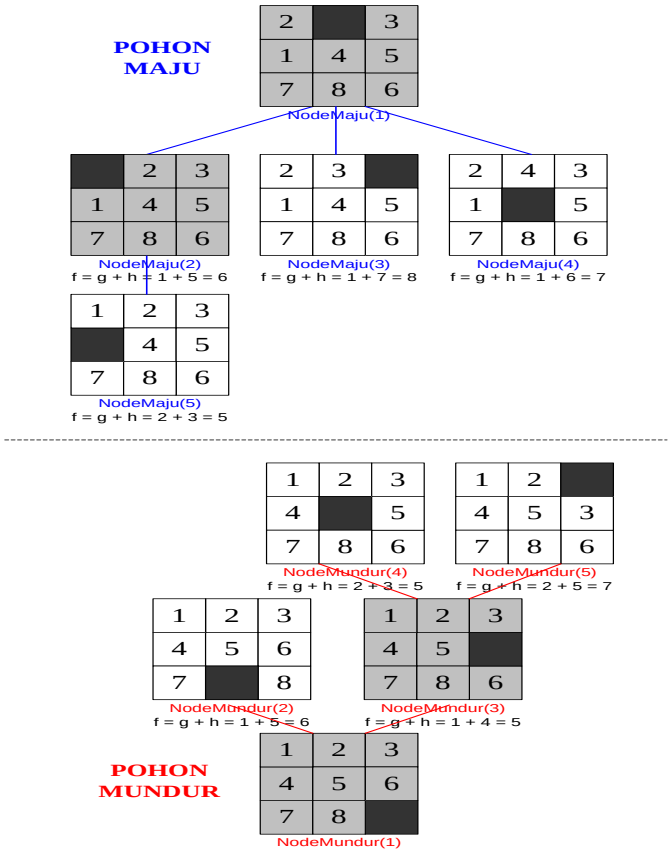
Geser angka 1 ke atas, menghasilkan NodeMaju-5 : “[1|2|3][0|4|5][7|8|6]”.

b) Penelusuran Mundur, Keadaan NodeMundur-3 : “[1|2|3][4|5|0][7|8|6]”:

(a) Geser angka 5 ke kanan,
menghasilkan NodeMundur-4 : “[1|2|3][4|0|5][7|8|6]”.

(b) Geser angka 3 ke bawah,
menghasilkan NodeMundur-5: “[1|2|0][4|5|3][7|8|6]”.

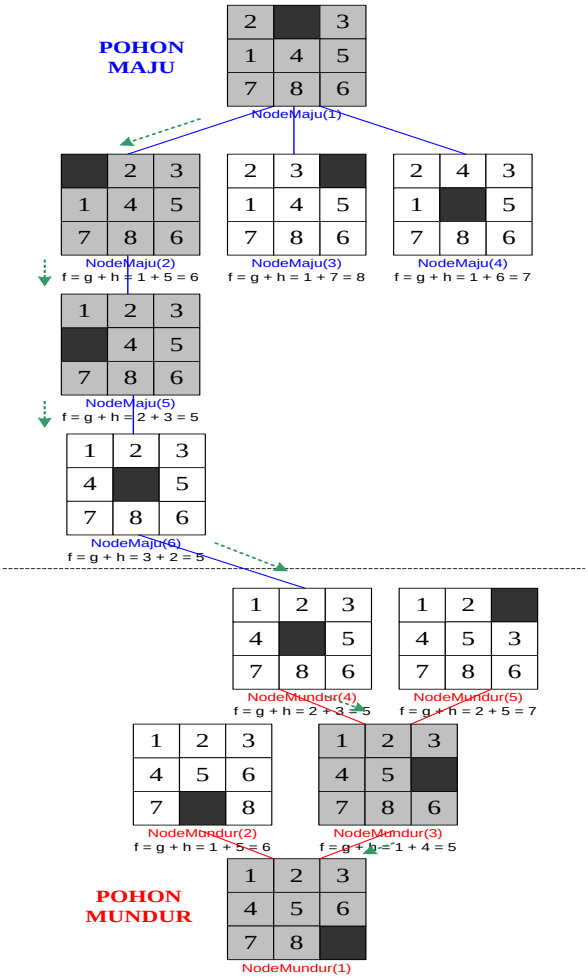
Illustrasi:



Gambar 13. Illustrasi-3 dari Metode BDA*

4) Periksa *node* yang masih berstatus OPEN (belum dikembangkan keadaan anaknya) dan memiliki fungsi heuristik *f* terkecil.
Penelusuran Maju, Keadaan NodeMaju-5 : “[1|2|3][0|4|5][7|8|6]”:
Geser angka 4 ke kiri, menghasilkan NodeMaju-6 : “[1|2|3][4|0|5][7|8|6]”.

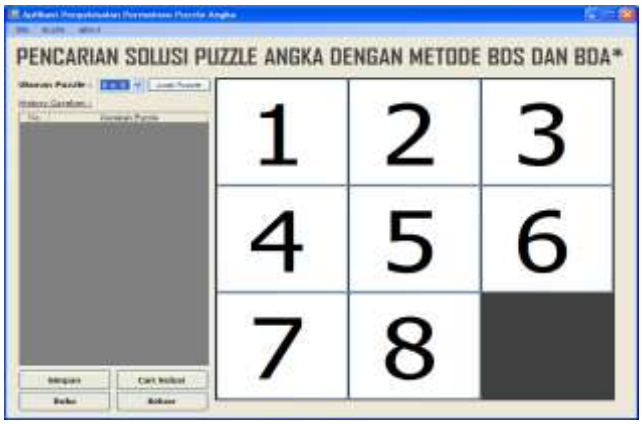
5) Ternyata keadaan pada NodeMaju-6 sama dengan keadaan pada NodeMundur-4. Jawaban ditemukan dan pencarian dihentikan. Alur solusi dari keadaan awal ke keadaan tujuan dapat dilihat pada gambar 14.



Gambar 14. Proses Kerja Pohon BDA*

3.2. **Implementasi Sistem**

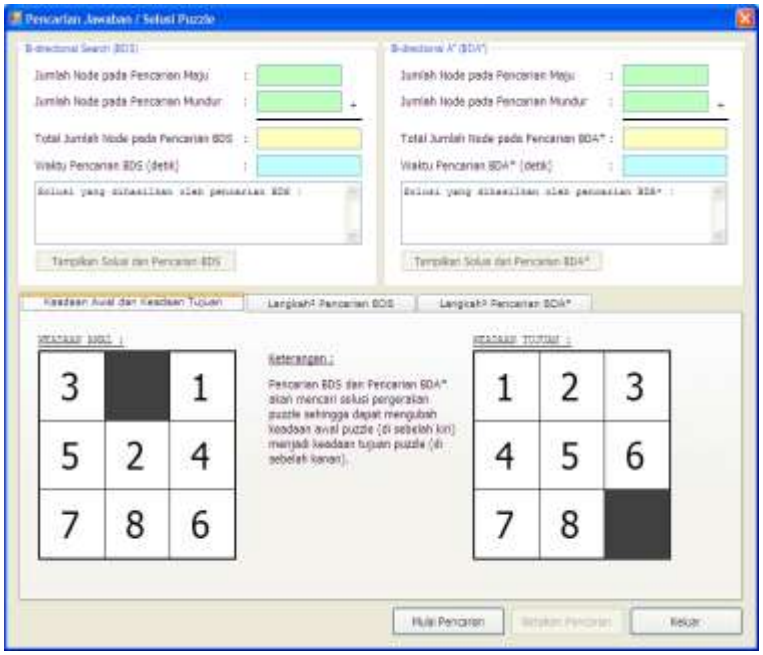
Bila aplikasi dijalankan, maka *form* utama akan tampil seperti yang terlihat pada Gambar 15. *Form* Utama berisi kotak *puzzle* dan tombol-tombol yang memiliki fungsinya masing-masing. Tampilan form menu utama ini menampilkan menu dan toolbar yang berisi sejumlah fungsi utama dari aplikasi seperti menu file yang berisi buka, simpan, keluar. Menu *puzzle* yang berisi cari solusi puzzle, langka-langka pencarian metode BDS dan BDA* dan kesimpulan perbandingan.



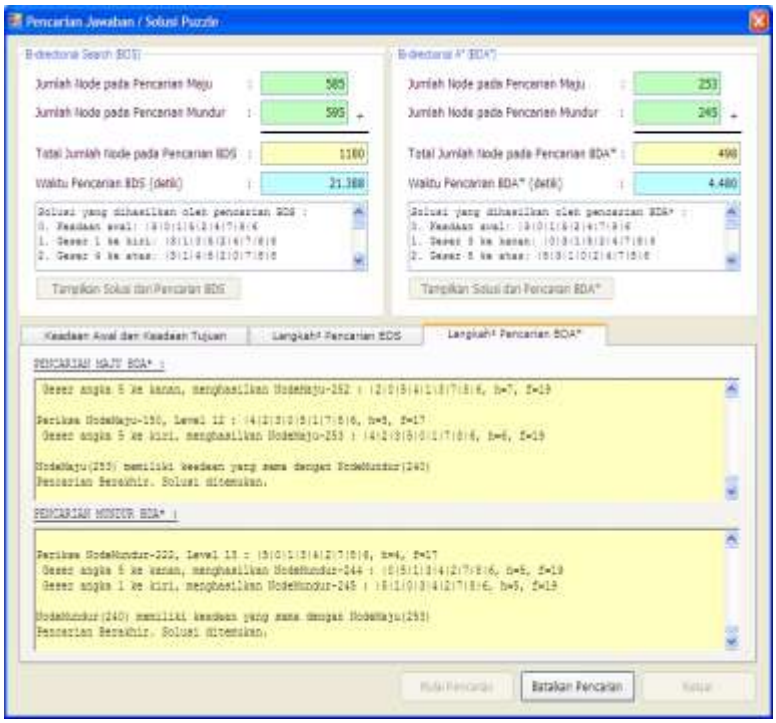
Gambar 15. Tampilan Form Utama

a. Tampilan Form pencarian jawaban/solusi puzzle

Apabila user ingin mencari solusi *puzzle* dengan menggunakan metode BDS dan BDA*, terdapat tombol 'Cari Solusi' untuk memulai pencarian solusi dengan Metode BDS maupun Metode BDA*. Apabila dieksekusi maka *Form* Pencarian Solusi akan tampil seperti terlihat pada Gambar 16. Dan secara otomatis dapat memperlihatkan hasil pencarian seperti terlihat pada Gambar 17.



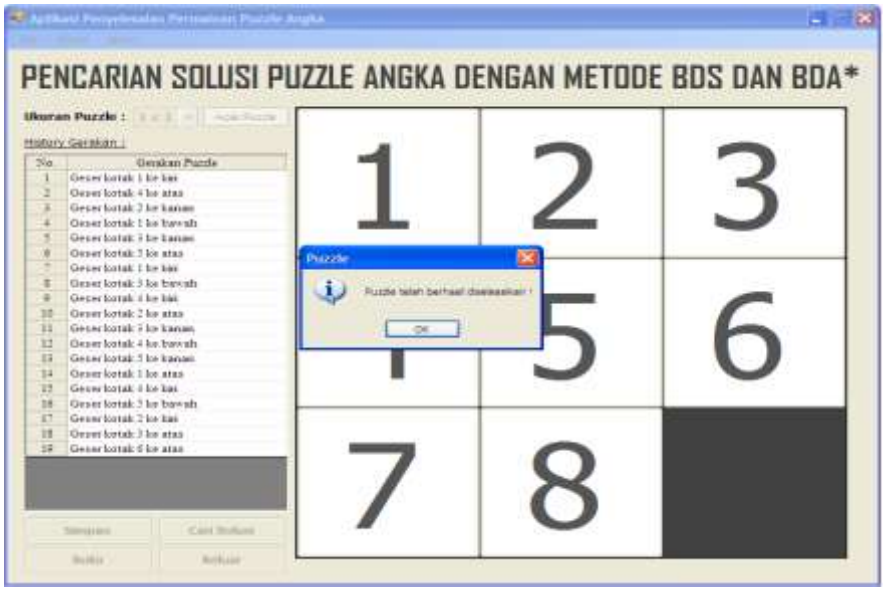
Gambar 16. Tampilan Form Pencarian Jawaban/Solusi Puzzle



Gambar 17. Tampilan Proses Pencarian Jawaban Selesai

b. Tampilan Simulasi Pergerakan Solusi pada Form Utama

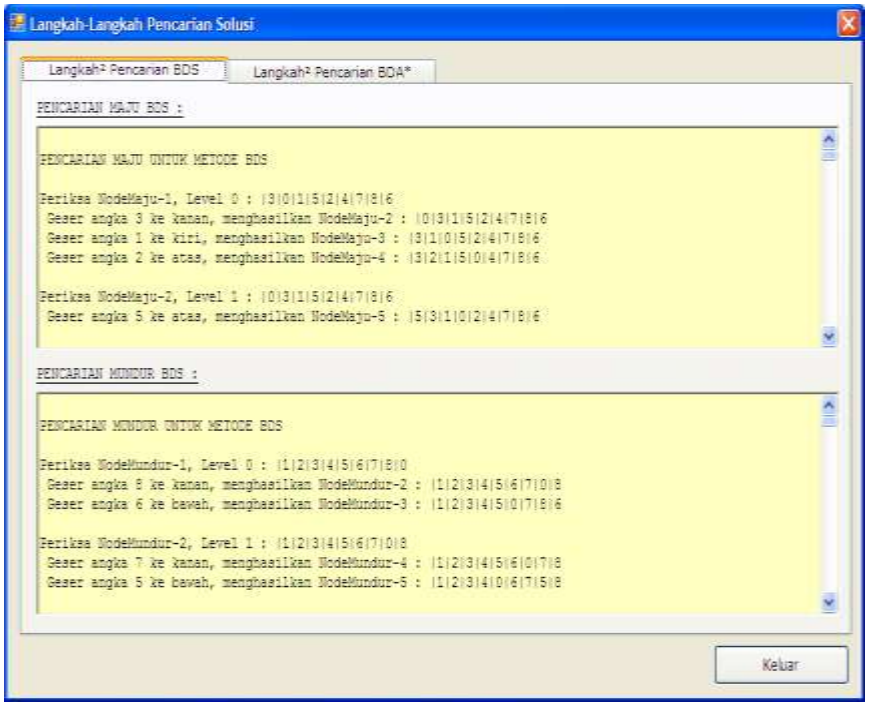
Aplikasi akan menampilkan pergerakan solusi *puzzle* pada *form* Utama, seperti terlihat pada Gambar 18 sebagai berikut:



Gambar 18. Tampilan Simulasi Pergerakan Solusi pada Form Utama

c. Tampilan Form Langkah-Langkah Pencarian

Tampilan form untuk melihat kembali langkah-langkah pencarian solusi berfungsi untuk menampilkan langka pencarian BDS dan BDA* yang diambil dari menu puzzle pada form utama dan dapat diliat pada Gambar 15 dan dibawah ini adalah langkah-langka pencarian solusi yang terlihat pada gambar 19 sebagai berikut:



Gambar 19. Tampilan Form Langkah-Langkah Pencarian BDS

4. PENUTUP

Kesimpulan yang dapat diambil adalah sebagai berikut :

a. Metode BDS dan BDA* dapat dideterapkan dalam menyelesaikan *Puzzle* Angka.
b. Langkah metode BDS yang mencari dan mengembangkan semua keadaan *puzzle* per level, menjamin didapatkannya solusi terpendek (*shortest path*), sedangkan metode BDA* yang menggunakan fungsi heuristik untuk menghitung kedekatan suatu keadaan dengan keadaan tujuan.
c. Metode BDA* lebih cepat dan lebih efisien dalam menemukan solusi dari pada metode BDS.
d. Aplikasi dapat menampilkan langkah-langkah pencarian dari metode BDS dan metode BDA*, sehingga dapat membantu pembelajaran terhadap kedua metode tersebut.

5. DAFTAR PUSTAKA

Desiani dan Muhammad Arhami, 2005, *Konsep Kecerdasan Buatan*, Penerbit Graha Ilmu, Yogyakarta

Hariyanto, B., *Struktur Data,* Edisi Kedua, Informatika, Bandung, 2003.

Kusumadewi,S,2003, *Artifial Intelligence (Teknik Dan Aplikasi),*penerbit graha ilmu

Suyanto, 2007, *Artificial Intelligence*, Informatika, Bandung.