Aplikasi Permainan *Battleship* Menggunakan Algoritma Runut-Balik Dengan *Breadth First Search*

Arif Aliyanto[1], Felix Novendo Ishak[2]
[1]*Sistem Informasi, Sekolah Tinggi Teknik Musi*
Jl. Bangau No.60, Palembang, 30113
HP: +62 81532894444
E-mail : meymey_plb@yahoo.com[1)]

ABSTRAK

Battleship merupakan permainan berjenis board game. Cara memainkannya dengan menembakan kotak-kotak yang kosong pada papan permainan sampai salah satu menang antara pemain atau komputer (AI). Permainan battleship ini berbasis komputer memiliki kecerdasan buatan atau Artificial Intelligence (AI) sehingga permainan ini tidak membutuhkan dua orang untuk bermain. Selain juga tidak memerlukan dua orang untuk bermain, pemain juga dapat menimati permainan battleship ini dengan kecerdasan buatan (AI) yang dilengkapi dengan algoritma atau metode. Metode yang diambil sebagai pembuatan permainan battleship ini adalah algoritma backtracking dengan menggunakan metode BFS. Tujuan dari pembuatan aplikasi ini adalah agar pemain yang ingin memainkan permainan battleship ini dapat menjalankan aplikasi ini dengan algoritma yang sudah dilengkapi dengan pencarian solusi yang dibuat. Gambaran permainan battleship ini seperti permainan yang saling menghancurkan, tetapi yang tema yang diambil dalam pembuatan aplikasi ini adalah apakah pencarian solusi dengan menggunakan algoritma backtracking metode BFS merupakan metode atau algoritma yang sangat efisien dalam aplikasi permainan battleship ini. Aplikasi permainan battleship ini dibuat dengan menggunakan metode waterfall dengan pemodelan Unified Modeling Language (UML). Serta menerapkan algoritma backtrackingmetode BFS. Aplikasi menggunakan penerapan pencarian solusi yang digunakan dengan menggunakan algoritma backtracking. Aplikasi ini dapat digunakan sebagai media hiburan. Selain sebagai media hiburan, aplikasi ini pun dapat dikembangkan dengan fitur yang lebih menarik lagi.

Kata Kunci : Battleship, Artificial Intelligence, Backtracking, BFS

1. LATAR BELAKANG

Bermain *game* merupakan salah satu aktifitas yang sangat disukai oleh sebagian besar masyarakat di dunia ini. Alasan mereka bermain *game* tentunya berbeda-beda, ada yang untuk melepas lelah, ada juga yang memang suka atau hobi bermain *game*. Dengan berkembangnya teknologi sekarang ini, *game-game* ini tidak hanya dapat kita jumpai pada kehidupan nyata, tapi juga dapat kita jumpai di dalam dunia maya. Jenisnya pun semakin banyak dan bervariasi. Salah satu yang cukup menarik perhatian adalah permainan komputer.

Permainan-permainan berbasis komputer ini juga bermacam-macam. Salah satu kelebihannya adalah kita tidak harus mencari orang untuk menjadi lawan tanding jika ingin bermain karena permainan berbasis komputer ini sudah mendukung *single-player mode* dimana kita dapat bermain sendiri melawan komputer yang dirancang untuk dapat berlaku seperti pemain manusia atau yang sering dikenal dengan *Artificial Inteligince* (AI) [1].

Runut-Balik merupakan salah satu contoh dari Kecerdasan Buatan. Prinsip kerja dari algoritma ini pemangkasan simpul- simpul yang tidak mengarah ke solusi. Sehingga, setiap simpul yang tidak memenuhi suatu fungsi pembatas, tidak akan diproses di algoritma ini. Adapun media yang cocok untuk penerapan algoritma Runut-Balik adalah permainan *Battleship*, beberapa alasan mengapa *Battleship* digunakan sebagai media penerapan kecerdasan buatan antara lain fungsi pembangkit pada kasus ini adalah semua kemungkinan bagian kapal baik secara vertikal maupun horisontal serta bagian air. Fungsi pembatas yang digunakan adalah perbandingan jumlah bagian kapal pada suatu kolom dan baris dengan angka pembanding yang berada dipinggir kanan (untuk baris) dan bawah (untuk kolom). Selain itu, fungsi pembatas lainnya adalah pengecekan ada atau tidaknya bagian kapal disekitar lokasi penempatan bagian kapal. Apabila ada, maka lokasi tersebut tidak dapat ditempatkan bagian kapal, sehingga dapat ditempatkan bagian air [2].

2. TINJUAN PUSTAKA

2.1. *Game*

Permainan atau *game* merupakan media pembelajaran langsung dengan pola *learning by doing*. Pembelajaran yang dilakukan dalam sebuah *game* merupakan suatu konsekuensi dari sang pemain *game* untuk dapat melalui

tantangan yang ada dalam suatu permainan *game* yang dijalankan. Dengan demikian, *game* menawarkan suatu bentuk media dan metode yang menakjubkan. *Game* mempunyai potensi yang sangat besar dalam membangun motivasi pada proses pembelajaran. Pada penerapan metode konvensional untuk menciptakan motivasi belajar sebesar motivasi dalam *game*, dibutuhkan seorang guru/instruktur yang berkompeten dalam pengelolaan proses pembelajaran [3].

2.2. *Board Game*

Board game merupakan sebagai sebuah media memiliki begitu banyak potensi, salah satunya sebagai sarana penyampaian informasi yang efektif. Dengan format yang mendorong pemainnya untuk berinteraksi, penyampaian informasi melalui *board game* menjadi lebih dinamis dan tidak membosankan. Selain itu, *board game* umumnya mengangkat tema yang spesifik serta menarik, dilengkapi dengan peraturan yang mendetail tapi tetap mudah diikuti, mekanisme permainan yang menantang logika, dan menuntut adanya interaksi yang cukup sering antara para pemain, jauh berbeda dari berbagai permainan papan yang umumnya telah lama kita kenal (ular tangga, ludo dan halma [4].

2.3. Permainan *Battleship*

Permainan ini menyediakan beberapa buah tempat yang disusun oleh kotak-kotak sebagai medan perang. Kapal-kapal perang dengan ukuran yang bervariasi disusun dalam medan perang masing-masing. Letak dari kapal-kapal perang ini tidak terlihat dalam di pemain lawan. Pemain berusaha untuk menghancurkan kapal-kapal perang pemain lawan yaitu AI dengan cara meng-klik kotak-kotak yang dianggap sebagai letak dari kapal-kapal perang lawan. Pemain hanya boleh menebak satu kali saja setiap giliran. Pemain yang masih menyisakan kapal perang yang memenangkan permainan (game).

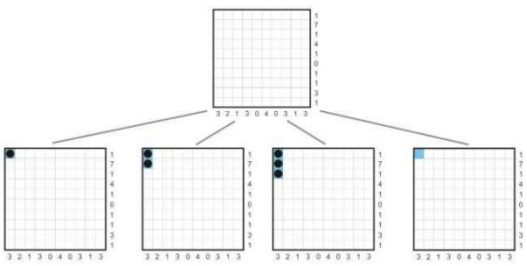
2.4. Algoritma Runut-Balik (*Backtracking*)

Di dalam algoritma ini, terdapat beberapa fungsi dan variabel yang digunakan di dalam pemrosesan solusi, yaitu ruang solusi, fungsi pembangkit, serta fungsi pembatas. Ruang solusi persoalan dinyatakan sebagai vektor dengan n-tuple. Fungsi pembangkit untuk membangkitkan nilai untuk nilai elemen dengan indeks tertentu yang merupakan komponen vektor solusi. Fungsi pembatas digunakan untuk menentukan apakah suatu simpul dapat mengarahkan ke solusi atau tidak. Ruang solusi diatur sehingga menjadi sebuah struktur pohon seperti pada algoritma DFS. Setiap simpul dari pohon tersebut menyatakan status persoalan, sedangkan sisi/cabang dilabeli dengan nilai-nilai komponen persoalan. Lintasan dari akar pohon hingga daun menyatakan solusi yang mungkin. Pengaturan pohon ruang solusi ini disebut juga sebagai pohon ruang status [2].

2.5. Pohon Permainan

Solusi dari teka-teki *Battleship* dapat ditemukan dengan berbasiskan algoritma Runut-Balik. Fungsi pembangkit pada kasus ini adalah semua kemungkinan bagian kapal baik secara vertikal maupun horizontal serta bagian air. Fungsi pembatas yang digunakan adalah perbandingan jumlah bagian kapal pada suatu kolom dan baris dengan angka pembanding yang berada di pinggir kanan (untuk baris) dan bawah (untuk kolom). Selain itu, fungsi pembatas lainnya adalah pengecekan ada atau tidaknya bagian kapal di sekitar lokasi penempatan bagian kapal. Apabila ada, maka lokasi tersebut tidak dapat ditempatkan bagian kapal, sehingga dapat ditempatkan bagian air.

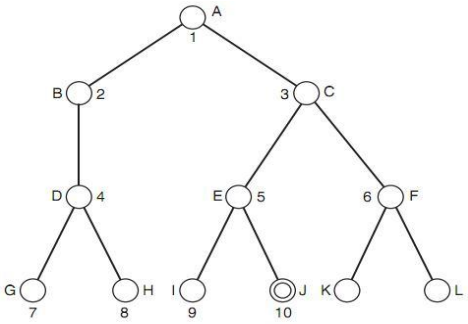
Algoritma pencarian solusi teka-teki ini dapat dimulai dengan mengisi kotak pada baris pertama dan kolom pertama dengan bagian kapal yang memenuhi fungsi pembatas. Misalnya, pada baris pertama serta kolom pertama, komponen yang dapat dibangkitkan adalah komponen yang memenuhi angka pinggir kanan baris tersebut yaitu satu dan pinggir bawah kolom tersebut yaitu tiga. Komponen yang memenuhi kriteria tersebut adalah bagian air, kapal *submarine* (2 bagian), kapal *destroyer* (1 bagian) secara, kapal *battleships* (4 bagian) dan kapal *cruiser* (3 bagian) secara. Sehingga kita dapat memperoleh pohon ruang status awal seperti pada gambar 1[2].



Gambar 1. Pohon Ruang Status Awal

2.6. Metode BFS (*Breadth First Search*)

Algoritma BFS ini lebih mendahulukan pencarian cabang daripada pencarian kedalaman. Jadi, pencarian akan dilakukan dari tingkat n dan akan lanjut ke tingkat n+1 jika dan hanya jika tingkat n telah ditelusuri seluruhnya



Gambar 2. Penelusuran pohon permainan dengan BFS

Penelusuran pada Gambar 2 dimulai dari A-B-C-D-E-F-G-H-I-J di mana A merupakan kedaaan awal dan mencapai J yang merupakan keadaan tujuan. Dapat diketahui dengan jelas bahwa penelusuran BFS dilakukan secara tingkat ke tingkat pada pohon permainan [5].

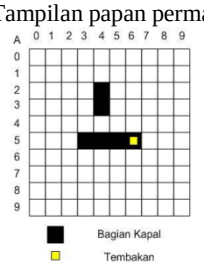
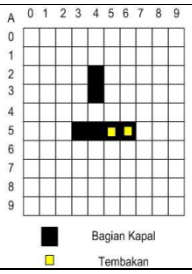
3. METODE PENGEMBANGAN PERANGKAT LUNAK

Metode pengembangan perangkat lunak yang dibuat dengan tahapan, *analysis* (analisis), *design* (perancangan), *development* (pembangunan), dan *testing* (pengujian) [6].

3.1. *Analysis* (Analisis)

Pada analisis ini akan diuraikan mengenai analisis permainan battleship itu sendiri. Analisis yang dilakukan yaitu berupa analisis aturan permainan dan analisis algoritma *backtracking* pada *game battleship*. Serta analisis kebutuhan sistem yang diperlukan adalah peletakan bidak-bidak kapal yang akan diletakan dan papan permainan yang digunakan untuk bermaik *game battleship*.

Tabel 1. Analisis Permainan *Battleship* dengan algoritma *Backtracking*

No	Tampilan papan permainan	Keterangan
1		Diawal permainan awan, player dan AI akan meletakan masing-masing bidak secara acak. Tampilan yang digunakan penulis sebagai contoh hanya diambil 2 bidak yang terdiri atas 1 kapal *submarine*A[(2,4),(3,4)] dan 1 kapal *battleship* A[(5,3),(5,4),(5,5),(5,6)] dan dilakukan tembakan pada koordinat (5,6)
2		Lalu pencarian solusipun dimulai dari koordinat (5,5), dan seterusnya sampai keberadaan posisi kapal berada.

3.2. *Design* (Perancangan)

Dalam desain ini yang perlu di desain yaitu desain dari proses permainan *battleship* berupa *use case* dari sistem tersebut. Desain skenario permaian *battleship* dan desain tampilan *game battleship*.

3.3. *Development* (Pembangunan)

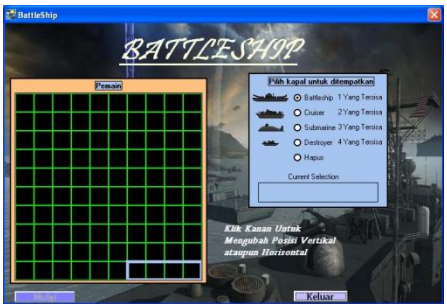
Pada tahapan pembangunan atau pengembangan ini digunakan tahapan pengembangan rekayasa perangkat lunak, dalam tahapan pembangunan ini meliputi : Rancangan Aplikasi Permainan *Battleship*, rancangan yang telah dibuat akan dikembangkan dengan menggunakan *tools developer* untuk mendapatkan implementasi sistem yang sesuai dengan rancangan sebelumnya.

3.4. *Testing* (Pengujian)

Untuk menghasilkan suatu perangkat lunak yang bermutu, maka perlu dilakukan pengujian. Aplikasi permainan *Battleship* dengan menggunakan algoritma *backtracking* ini diuji dengan pengujian *Black-Box Testing* dan *White- Box Testing*.

4. HASIL DAN PEMBAHASAN

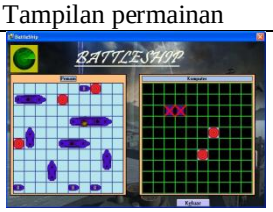
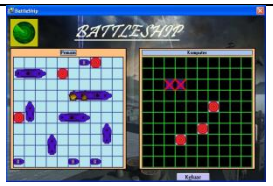
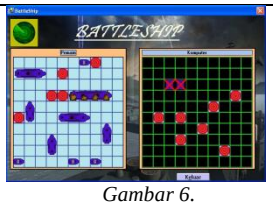
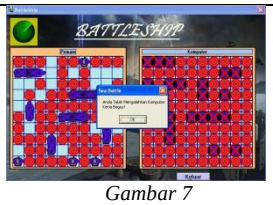
Berdasarkan hasil analisis dan perancangan sistem yang telah dilakukan, maka dilakukan implementasi ke dalam bentuk program komputer. Implementasi merupakan tahap menerjemahkan hasil perancangan perangkat lunak secara rinci kedalam bahasa pemograman. Implementasi dilakukan dengan menggunakan bahasa pemograman Microsoft Visual Basic 6.0, sehingga hanya dapat dijalankan pada komputer dengan sistem operasi *Windows*. Berikut tampilan papan permainan battleship terlihat pada gambar 3



Gambar 3. Tampilan papan permainan

Pada *form* ini digunakan untuk menaruh bidak pemain yang sudah tersedia. Fungsi yang digunakan adalah klik kanan pada *mouse* digunakan untuk merubah posisi kapal dari horisontal menjadi vertikal. Klik kiri digunakan untuk meletakkan bidak.

Tabel 2. Hasil dan pembahasan aplikasi permainan *battleship*

No	Tampilan permainan	Keterangan
1	 *Gambar 4.*	Pada tampilan gambar 4 ini, komputer berhasil menembakan salah satu badan kapal milik pemain.
2	 *Gambar 5.*	Pada gambar 5 ini, metode BFS mulai jalan secara otomatis, karena ada bagian badan kapal pemain yang tertembak. Logika pencarian solusi-nya dengan menggunakan algoritma *backtracking*. Dengan menggunakan logika x-1. Setelah logika x-1 di jalankan, maka logika x-2 berjalan secara otomatis, pencarian solusi dilakukan pada arah kiri. Setelah logika x-2 di jalankan, maka logika x-3 berjalan secara otomatis, pencarian solusi dilakukan pada arah kiri. Pencarian solusi dijalankan pada logika x-4. Setelah pada posisi ini, maka pencarian solusi akan dijalankan pada posisi ke kanan.
3	 *Gambar 6.*	Setelah pencarian solusi pada sisi kiri telah selesai, maka pencarian solusi akan dilanjutkan pada logika x+1, atau pada sisi kanan. Pencarian solusi dilanjutkan pada koordinat x+2. Pencarian solusi hanya berhenti sampai x+2, karena telah mencapai sisi papan permainan. Jika tidak terkena sisi papan, maka pencarian akan dilanjutkan sampai koordinat x+4. Pencarian solusi dilanjutkan kearah bawah dengan koordinat y-1, dilanjutkan pada logika y-2. Pencarian solusi dilanjutkan pada logika y-3. Walaupun ada terkena tembakan badan kapal yang lain, pencarian solusi tetap dijalankan dengan urutan logika, sehingga pencarian solusi tidak menjadi acak.
4	 *Gambar 7*	Setelah antara salah satu sisi menang, akan tampil seperti pada gambar 7, baik player yang menang maupun komputer yang menang atas permainan *battleship*.

4.1. Pengujian

Untuk menghasilkan suatu perangkat lunak yang baik, maka perlu dilakukan pengujian. Aplikasi permainan *Battleship* dengan menggunakan algoritma *backtracking* ini diuji dengan pengujian *Black-Box Testing* dan *White- Box Testing*.

4.1.1. Pengujian *Black-Box Testing*

Pengujian ini merupakan pengujian yang digunakan untuk menguji aplikasi yang telah dibuat. Bagian-bagian yang akan diuji adalah *form* menu utama dan *form* permainan. Dari hasil pengujian yang dilakukan dari item pengujian yang ada di *form* menu utama dan *form* permainan didapatkan hasil sesuai yang diharapkan dengan yang dimplementasikan.

4.1.2. Pengujian *White-Box Testing*

Pengujian *white box* atau dapat disebut juga pengujian *glass box* dilakukan untuk memastikan bahwa semua jalur independen dari suatu modul telah dilalui paling sedikit satu kali. Pada aplikasi permainan *battleship* ini, pengujian *white box* yang dilakukan adalah membuat *Flowgraph* yang diambil dari *Flowchart*. Pengujian ini dilakukan pada *flowgraph* pemain dan *flowgraph* AI, selanjutnya dari hasil *flowgraph* baik untuk pemain maupun untuk AI akan dilakukan perhitungan nilai *Cyclomatic Complexity*. Setelah dihitung nilai *Cyclomatic Complexitnya*, ternyata didapat bahwa V(G)= 2. Maka dapat ditarik kesimpulan bahwa aplikasi permainan ini memiliki prosedur yang sederhana.

5. SIMPULAN DAN SARAN

5.1. Simpulan

Setelah menyelesaikan perancangan perangkat lunak permainan strategi *BattleShip* yang dapat dimainkan dengan AI, penulis menarik kesimpulan sebagai berikut :

1. Perangkat lunak ini memungkinkan pemakai (*user*) komputer untuk memainkan permainan *Battle Ship* tanpa harus berhadapan secara langsung.
2. Perangkat lunak hanya dapat dimainkan oleh satu orang saja dengan melawan AI dengan algoritma yang telah diteliti oleh penulis.

5.2. Saran

1. Perangkat lunak dapat dikembangkan untuk user dengan algoritma yang lebih baik.
2. Perangkat lunak dapat ditambahkan konsep permainan yang memiliki tingkat kesulitan permainan atau Level.

DAFTAR PUSTAKA

[1] Munir, Rinaldi. *Diktat Kuliah IF 2251 Strategi Algoritmik*. Program Studi Teknik Informatika, Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung. 2009.

[2] Sumarsono, Abraham Ranardo. 2009. Penerapan Algortima Runut Balik dalam Pencarian Solusi Teka Teki Battleship. Institut Teknologi Bandung.

[3] Donald Clark. 2006. Game and e-learning. Sunderland: Caspian Learning. (diakses:http://repository.usu.ac.id/bitstream/123456789/20871/5/Chapter%20I.pdf, 15-05-2012).

[4] Fitriani, Wanti. 2011. *Efektivitas Media Board Game Untuk Meningkatkan Hasil Belajar Siswa pada Mata Pelajaran Ilmu Pengetahuan Sosial di Sekolah Dasar*. Jakarta: Teknologi Pendidikan, Universitas Pendidikan Indonesia.

[5] Wijaya, Surya. 2010. *Penerapan Konsep Algoritma Minimax dengan Menggunakan Breadth First Search (BFS) pada Permainan Reversi* . Medan: Universitas Sumatra Utara.

[6] Roger.S, 2001. *Software Engineering : A Practioner's Approach*. 5th .McGrawHill.