IMPLEMENTASI *BACKTRACKING ALGORITHM* UNTUK PENYELESAIAN PERMAINAN *SU DOKU* POLA 9X9

Febri Utama[1)], Awang Harsa Kridalaksana[2)], Indah Fitri Astuti[3)]

[1,2,3)]Jurusan Ilmu Komputer, FMIPA Universitas Mulawarman
Jalan Barong Tongkok No. 4 Kampus Gunung Kelua Samarinda, Kalimantan Timur
Email : febri_shadow@yahoo.co.id[1)] , awangkid@gmail.com[2)] , indahfitriastuti@fmipa.unmul.ac.id[3)]

ABSTRAK

Permainan *Su Doku* pertama kali muncul pada tahun 1979 di majalah Dell Magazines dengan nama "*Number Places*", kemudian menjadi populer dengan nama "*Su Doku*" di Jepang pada tahun 1984. Pada umumnya permainan ini terdiri dari 81 kotak kecil (sel), yang disebut juga dengan *Su Doku* pola 9x9. *Su Doku* ini dibagi menjadi 9 grid dengan pola 3x3. Diantara sel-sel tersebut terdapat angka 1 sampai dengan 9 sebagai angka awal. Angka-angka awal ini digunakan sebagai pembatas, sehingga kita hanya melanjutkan dengan mengisi angka hingga seluruh sel-sel terisi penuh masing-masing dengan angka 1 sampai dengan 9. Metode umum pencarian solusi *Su Doku* adalah dengan kombinasi teknik pemindaian (*scanning*), penandaan (*marking*), dan analisa (*analyzing*). Tujuan penelitian ini adalah untuk membuat suatu aplikasi yang merupakan implementasi dari metode backtracking algorithm untuk menampilkan solusi *Su Doku* pola 9x9 yang unik dan membuktikan teori ketepatan metode backtracking algorithm dalam penggunaan pencarian solusi *Su Doku* pola 9x9. Implementasi dari metode *backtracking algorithm* dibuat dengan bahasa pemrograman *Delphi*. Hasil penelitian ini adalah sebuah aplikasi pencarian solusi *Su Doku* dengan pola 9x9, yang diberi nama "Program Aplikasi Sudoku Solver Backtracking Algorithm". Didalam aplikasi ini user menginputkan angka awal sesuai dengan level *Su Doku*, dimana hasil outputnya adalah tampilan dari solusi unik *Su Doku*.

Kata kunci : *Su Doku* pola 9x9, *backtracking algorithm*, sel, angka awal, solusi unik.

PENDAHULUAN

Permainan *Su Doku* pertama kali muncul pada tahun 1979 di majalah *Dell Magazines* dengan nama "*Number Places*", kemudian menjadi populer dengan nama *Su Doku* di Jepang pada tahun 1984. Pada umumnya permainan ini terdiri dari 81 kotak kecil (sel), yang disebut juga dengan *Su Doku* pola 9x9. *Su Doku* ini dibagi menjadi 9 *grid* dengan pola 3x3. Di antara sel-sel tersebut terdapat angka 1 sampai dengan 9 sebagai angka awal. Angka-angka awal ini digunakan sebagai pembatas, sehingga pengguna hanya melanjutkan dengan mengisi angka hingga seluruh sel-sel terisi penuh dengan angka 1 sampai dengan 9.

Tidak seperti permainan teka-teki silang, yang membutuhkan pengetahuan umum untuk menjawab setiap soal yang ada dan menguasai bahasa tertentu (karena tentunya terdapat istilah asing yang merupakan jawaban dari pertanyaan teka-teki silang tersebut), pengguna tidak perlu memiliki kemampuan pengetahuan umum dan bahasa tertentu untuk dapat menikmati permainan *Su Doku*. Bahkan secara teknis, pengguna tidak perlu tahu cara berhitung. Pengguna hanya harus menempatkan angka 1 sampai dengan 9, tidak perlu urut kedalam setiap baris (kiri ke kanan), setiap kolom (atas ke bawah), dan setiap *grid* (yang masing-masing berisi sembilan kotak kecil atau sel). Strategi yang baik adalah dengan mula-mula berpikir dalam *grid-grid*, atau lebih baik sekumpulan *grid*, kemudian mencari pasangan angka, yang dari keduanya pengguna dapat menemukan yang ketiga.

Namun, walaupun tidak perlu pengetahuan umum, kemampuan bahasa tertentu dan kemampuan berhitung, tidak mudah menyelesaikan *Su Doku*. Tantangannya adalah, bagaimana caranya mengisi seluruh sel yang tadinya hanya terdiri dari beberapa angka saja menjadi penuh terisi semua, dengan syarat dari kesembilan angka tersebut tidak boleh terulang disetiap 1 baris, kolom dan *gridnya*. Boleh dikatakan bila pengguna salah mengisi angka di awal-awal permainan, dapat dengan mudah ditebak pengguna akan mengacaukan aturan permainan dengan kebingungan sendiri, mengapa ada angka yang sama terulang pada baris, kolom atau *grid* yang sama? (misalnya dalam 1 kolom terdapat dua angka 6 atau dalam satu *grid* terdapat dua angka 9) yang berakibat pada tidak akan tercapainya solusi akhir dan harus mengulang dari awal.

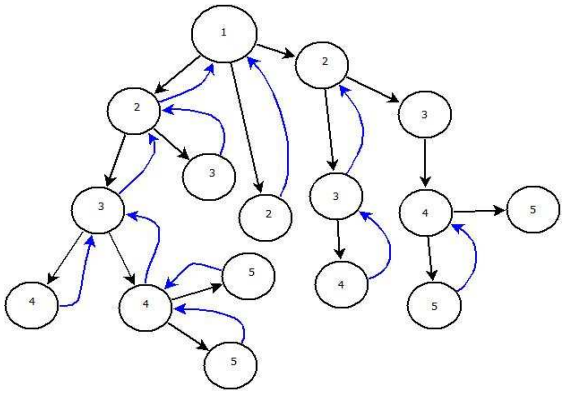
Untuk mencapai solusi akhir, dibutuhkan sebuah metode yang harus memenuhi persyaratan unik *Su Doku*, dengan menebak sebuah angka dan mencocokkan angka tersebut dengan keadaan sekelilingnya (dalam hal ini adalah baris, kolom dan *grid*). Jika terdapat angka yang sama, metode tersebut harus bisa mengganti angka yang sama tersebut dengan angka yang lain, sehingga akan mencapai solusi akhir, yakni sel *Su Doku* telah terisi semua dengan angka 1 sampai dengan 9, dengan tidak ada kesamaan angka pada satu baris,

kolom, dan *grid*nya. Metode yang cukup efektif dengan persyaratan *Su Doku* ini adalah dengan menggunakan metode *backtracking algorithm* (algoritma runut-balik). Mengapa menggunakan algoritma runut-balik, algoritma runut-balik berbasis pencarian *depth first search* (pencarian pertama terdalam) merupakan algoritma yang digunakan karena sifatnya yang hanya memberikan satu solusi saja, cukup efektif dengan solusi unik *Su Doku* yaitu hanya memiliki satu solusi (Arifiyanto, 2007).

LANDASAN TEORI

Backtracking Algorithm

Backtracking Algorithm (algoritma runut-balik) adalah sebuah algoritma pencarian yang berdasarkan pada proses pencarian DFS (*Depth First Search*). Pencarian dengan metode ini dilakukan dari node awal secara mendalam hingga yang paling akhir (*dead-end*) atau sampai ditemukan. Dengan kata lain, simpul cabang atau anak yang terlebih dahulu dikunjungi (Desiani, 2006).

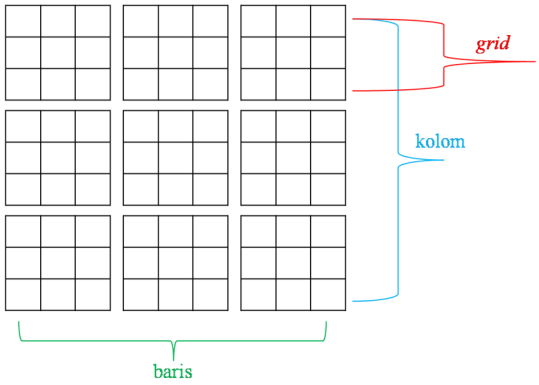


Gambar 1. Teknik pencarian dengan metode *backtracking algorithm*

Berdasarkan gambar 1, proses pencarian dilakukan dengan mengunjungi cabang terlebih dahulu hingga tiba di simpul terakhir. Jika tujuan yang diinginkan belum tercapai maka pencarian dilanjutkan ke cabang sebelumnya, turun ke bawah jika memang masih ada cabangnya. Begitu seterusnya hingga diperoleh tujuan akhirnya (*Goal*). Dengan metode *backtracking* ini, dalam proses pencariannya tidak mengarah kepada semua kemungkinan solusi yang ada, hanya mengarah kepada satu solusi saja. Akibatnya rentang waktu dalam pencarian solusi lebih sedikit (proses pencarian menjadi lebih efektif dan efisien). *Bactracking algorithm* lebih alami dinyatakan dalam *recursif algorithm* (algoritma rekursif). Kadang-kadang algoritma ini (*backtracking algorithm*) disebut pula sebagai bentuk tipikal dari algoritma rekursif (*recursif algorithm*).

Su Doku

Nikoli, sebuah penerbit di Jepang, menyatakan *Su Doku* sebenarnya penyederhanaan kalimat bahasa Jepang "Suuji wa dokushin ni kagiru," yang berarti "terbatas hanya angka tunggal", sedangkan Su berarti angka dan Doku berarti tunggal. Nikoli pertama kali melihatnya di majalah Amerika, Dell Puzzle Magazines, yang disebut dengan "Number Place." Puzzle Number Place ini adalah hasil karya seorang arsitek, Howard Garns di tahun 1979, terinspirasi oleh Latin Squares, sebuah konsep matematika temuan Leonard Eulers, warga Swiss pada tahun 1783. Sejak tahun 1984 Nikoli menerbitkan *Su Doku* (Musadik, 2010).



Gambar 2. Ilustrasi pembagian sel

6				5	8	4	9	
	2	4						
				2			3	6
						9		7
7			3			8		
1	5			8	9			
	3	1						
				3		5		
		8		9	5			

Gambar 3. *Su Doku* terdiri dari 9 baris dan kolom dan dibagi lagi menjadi 9 *grid*

Pada umumnya, permainan *Su Doku* ini terdiri dari 9 buah kolom dan baris, dan dipisahkan lagi menjadi 9 kotak berukuran 3x3 yang selanjutnya disebut dengan *grid*. Dari gambar 2.2 terlihat bahwa hasil solusi dari permainan *Su Doku* ini adalah untuk setiap kotak kecil (sel) dan setiap *grid*-nya terdiri dari angka 1 sampai dengan angka 9, dimana tidak boleh ada angka yang terulang disetiap 1 baris, kolom, dan *grid*-nya. Sehingga dapat disimpulkan pada permainan *Su Doku* ini penyelesaiannya terdiri dari pola yang unik dan hanya terdapat satu solusi saja.

6	7	3	1	5	8	4	9	2
9	2	4	6	7	3	1	5	8
8	1	5	9	2	4	7	3	6
3	8	6	5	4	2	9	1	7
7	4	9	3	1	6	8	2	5
1	5	2	7	8	9	6	4	3
5	3	1	4	6	7	2	8	9
2	9	7	8	3	1	5	6	4
4	6	8	2	9	5	3	7	1

Gambar 4. Solusi akhir *Su Doku*

Strategi Umum Penyelesaian *Su Doku*

Secara umum, *Su Doku* dapat diselesaikan dengan kombinasi teknik pemindaian (*scanning*), penandaan (*marking*), dan analisa (*analyzing*) (Morenvino, 2006).

1. Pemindaian, berupa proses memindai baris atau kolom untuk mengindentifikasi baris mana dalam suatu *grid* yang terdapat angka-angka tertentu. Proses ini kemudian diulang pada setiap kolom (atau baris) secara sistematis. Kemudian menentukan nilai dari suatu sel dengan membuang nilai-nilai yang tidak mungkin.
2. Penandaan, berupa analisa logika dengan menandai kandidat angka yang dapat dimasukkan dalam sebuah sel.
3. Analisa, berupa eliminasi kandidat, dimana kemajuan dicapai dengan mengeliminasi kandidat angka secara berturut-turut hingga sebuah sel hanya punya 1 kandidat.

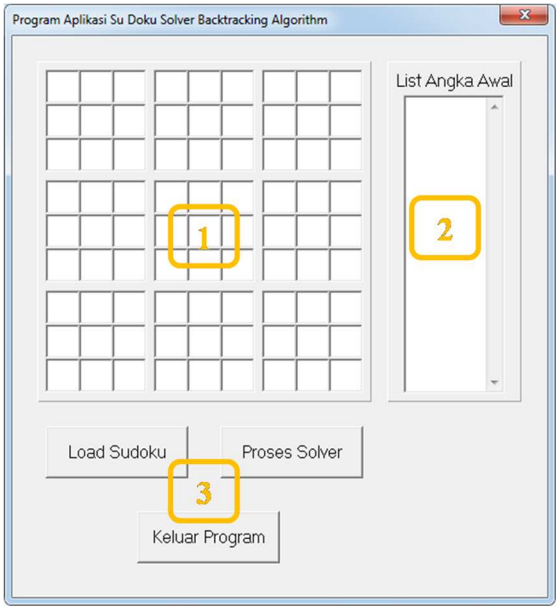
METODE PENELITIAN

Penelitian ini pada intinya adalah untuk menghasilkan aplikasi yang mampu menyelesaikan masalah penyelesaian permainan *Su Doku* pola 9x9, dengan tahap :

1. Solusi dicari dengan membentuk lintasan dari akar ke daun. Aturan pembentukan yang dipakai adalah mengikuti aturan pencarian mendalam (DFS). Simpul-simpul yang sudah dilahirkan dinamakan simpul hidup (*live node*). Simpul hidup yang sedang diperluas dinamakan simpul-E (*Expand-node*).
2. Tiap kali simpul-E diperluas, lintasan yang dibangun olehnya bertambah panjang. Jika lintasan yang sedang dibentuk tidak mengarah ke solusi, maka simpul-E tersebut "dibunuh" sehingga menjadi simpul mati (*dead node*). Fungsi yang digunakan untuk membunuh simpul-E adalah dengan menerapkan fungsi pembatas (*bounding function*). Simpul yang sudah mati tidak akan pernah diperluas lagi.
3. Jika pembentukan lintasan berakhir dengan simpul mati, maka proses pencarian diteruskan dengan membangkitkan simpul anak yang lainnya. Bila tidak ada lagi simpul anak yang dapat dibangkitkan, maka pencarian solusi dilanjutkan dengan melakukan runut-balik ke simpul hidup terdekat (simpul orangtua). Selanjutnya simpul ini menjadi simpul-E yang baru.
4. Pencarian dihentikan bila kita telah menemukan solusi atau tidak ada lagi simpul hidup untuk runut-balik.

HASIL DAN PEMBAHASAN

Implementasi *User Interface*



Gambar 5. *User interface* aplikasi

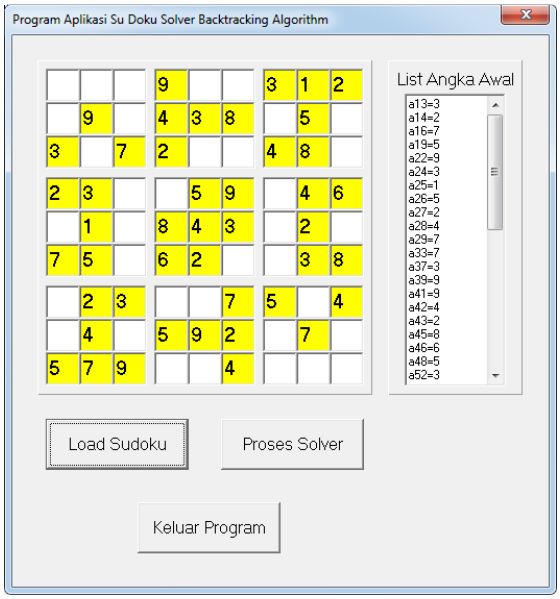
User interface aplikasi terdiri dari tiga bagian, yaitu :

1. Matriks *Su Doku* (berisi angka awal dan angka solusi).
2. List angka awal yang berfungsi sebagai inisiasi angka awal dari file berekstensi*.txt (file notepad) kedalam sistem matriks [9,9] dalam program.
3. Tombol eksekusi perintah (memilih level *Su Doku*, proses pencarian solusi *Su Doku*, dan keluar dari aplikasi *Su Doku* Solver).

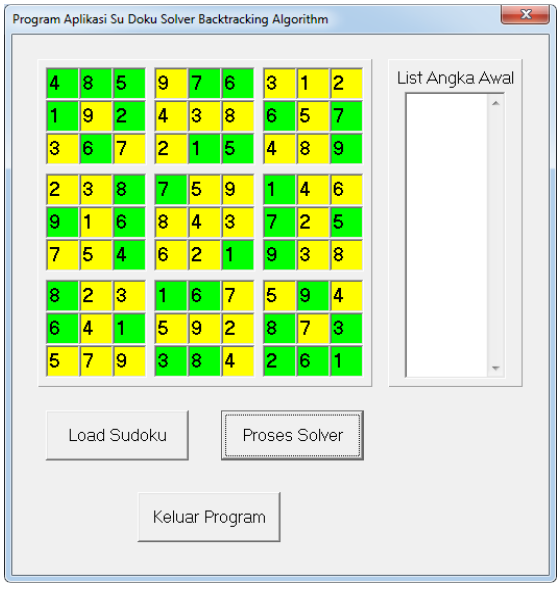
Implementasi *Backtracking Algorithm*

Penerapan *backtracking algorithm* dalam solusi *Su Doku* adalah :

1. Sistem memulai algoritma dari sel kosong yang pertama pada matriks.
2. Proses pencarian solusi secara umum dimulai berurutan dari sel kiri atas (a11) menuju ke sel kanan bawah (a99).



Gambar 6. Tampilan angka awal pada matriks *Su Doku* dan list angka awal

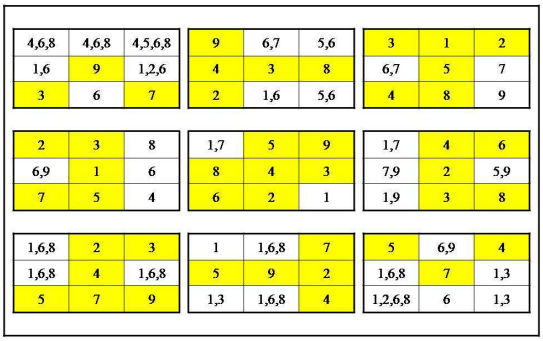


Gambar 7. Tampilan solusi akhir *Su Doku*

3. Proses pencarian pada baris dimulai berurutan dari sel kiri menuju ke sel kanan, pencarian pada kolom dimulai berurutan dari sel atas ke sel bawah, dan pencarian pada *grid* dimulai berurutan dari sebelah kiri atas ke sebelah kanan bawah.
4. Kumpulkan seluruh angka-angka yang menjadi kemungkinan solusi yaitu angka kemungkinan solusi, yang terdiri dari angka 1 sampai dengan 9, dan menempatkan kemungkinan solusi tersebut kedalam sistem data array.
5. Jika pada satu sel kosong terdapat lebih dari satu kemungkinan solusi, aplikasi menggunakan kemungkinan solusi dengan nilai terkecil untuk mengisi sel kosong tersebut.
6. Lakukan proses pengecekan apakah angka kemungkinan solusi memenuhi fungsi pembatas atau tidak. Fungsi pembatas yaitu tidak boleh ada angka yang sama pada satu baris, satu kolom, dan satu *grid*, serta tidak mengubah angka awal.
   a. Jika memenuhi fungsi pembatas, maka sel kosong tersebut diisi dengan angka kemungkinan solusi, kemudian proses pengecekan dilanjutkan ke sel kosong berikutnya.
   b. Jika tidak memenuhi fungsi pembatas, maka angka kemungkinan solusi diganti dengan angka kemungkinan solusi berikutnya yang belum dipakai.
7. Apabila seluruh angka kemungkinan solusi telah diganti dan masih belum memenuhi fungsi pembatas, maka lakukan proses *backtracking* ke sel sebelumnya. Sel sebelumnya ini merupakan sel yang telah terisi dengan kemungkinan solusi, bukan yang terisi dengan angka awal.
8. Pada sel dari hasil proses *backtracking* ini, jika terdapat lebih dari satu angka kemungkinan solusi, angka solusi yang lama diganti dengan angka kemungkinan solusi yang berkutnya, kemudian dilakukan pengecekan kembali apakah memenuhi fungsi pembatas atau tidak.
   a. Jika memenuhi fungsi pembatas, maka sel akan diisi dengan angka kemungkinan solusi yang baru dan menjadi angka solusi yang baru, serta menghapus angka kemungkinan solusi yang lama. Kemudian proses pengisian sel kosong dengan kemungkinan solusi dilanjutkan ke sel berikutnya.
   b. Jika seluruh angka kemungkinan solusi yang baru tidak memenuhi fungsi pembatas, maka lakukan proses *backtracking* ke sel sebelumnya.
9. Aplikasi akan melakukan semua proses terus-menerus (proses 1 sampai proses 8) hingga ditemukan solusi yang valid atau tidak ditemukan solusi yang valid.
   a. Solusi dinyatakan valid jika aplikasi berhasil mengisi semua sel kosong yang tersedia dengan angka solusi yang memenuhi fungsi pembatas.
   b. Solusi dinyatakan tidak valid jika dalam proses pencarian angka solusi, aplikasi melakukan *backtracking* hingga kembali ke sel kosong yang pertama (a11) dan tidak menemukan angka solusi untuk mengisi sel kosong.

Implementasi Pengisian Angka Pada Aplikasi

Dari gambar 6, jika seluruh sel kosong diisi dengan kemungkinan solusi dimana tidak memiliki kesamaan angka pada satu baris, kolom, dan *grid*, dengan angka awal yang telah diberi warna kuning pada selnya, maka hasilnya dapat dilihat pada gambar 8.



Gambar 8. Tampilan seluruh sel kosong terisi dengan kemungkinan solusi

a11	a21	a31	a41	a51	a61	a71	a81	a91
a12	a22	a32	a42	a52	a62	a72	a82	a92
a13	a23	a33	a43	a53	a63	a73	a83	a93
a14	a24	a34	a44	a54	a64	a74	a84	a94
a15	a25	a35	a45	a55	a65	a75	a85	a95
a16	a26	a36	a46	a56	a66	a76	a86	a96
a17	a27	a37	a47	a57	a67	a77	a87	a97
a18	a28	a38	a48	a58	a68	a78	a88	a98
a19	a29	a39	a49	a59	a69	a79	a89	a99

Gambar 9. Ilustrasi permodelan sel kedalam matriks pada *Su Doku*

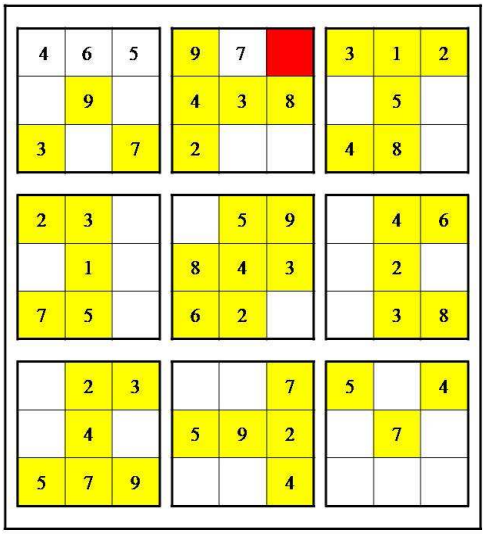
Proses pertama yang dieksekusi oleh aplikasi *Su Doku solver* adalah memeriksa apakah terdapat sel kosong, jika terdapat sel kosong maka sel kosong tersebut akan menjalani proses kedua yaitu mencari seluruh kemungkinan solusi yang akan mengisi sel kosong tersebut. Pada gambar 7 (a) sel pertama a11 merupakan sel kosong, maka proses pencarian dimulai dari sel a11. Setelah mencari seluruh kemungkinan solusi dengan tidak ada kesamaan angka pada baris, kolom dan *grid*, maka didapat 3 nilai kemungkinan solusi yaitu 4, 6, dan 8. Kemudian aplikasi akan menggunakan kemungkinan solusi dengan nilai terendah yaitu 4, untuk mengisi sel a11.

Setelah mengisi sel a11 dengan angka 4, kemudian aplikasi akan melanjutkan pemeriksaan sel kosong ke sel berikutnya yaitu sel a21. Karena sel a21 merupakan sel kosong, maka proses pencarian kemungkinan solusi dieksekusi pada sel a21. Pada sel a21 didapat 2 kemungkinan solusi yaitu 6 dan 8. Angka 4 tidak termasuk kedalam kemungkinan solusi karena terdapat angka 4 pada sel sebelumnya yaitu a11. Aplikasi menggunakan kemungkinan solusi dengan nilai terendah yaitu 6, dan menghapus kemungkinan solusi terendah yaitu 4, kemudian mengisi sel a21 dengan angka 6. Kemudian aplikasi memeriksa sel berikutnya yaitu sel a31. Sel a31 merupakan sel kosong maka proses pencarian kemungkinan solusi dieksekusi pada sel a31. Setelah dilakukan pencarian maka terdapat 2 kemungkinan solusi yaitu 5 dan 8, aplikasi mengisi sel a31 menggunakan kemungkinan solusi dengan nilai terendah yaitu 5.

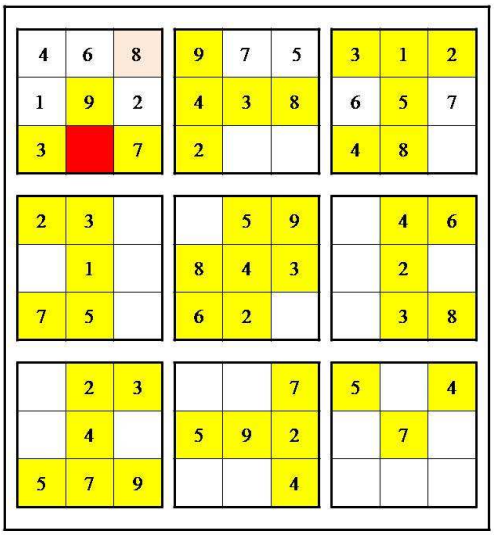
Kemudian aplikasi akan memeriksa sel berikutnya yaitu sel a41. Sel a41 bukan merupakan sel kosong, maka proses dilanjutkan ke sel berikutnya yaitu sel a51. Sel a51 merupakan sel kosong, maka proses pencarian kemungkinan solusi dieksekusi pada sel a51. Setelah dilakukan pencarian maka terdapat 1 kemungkinan solusi yaitu 7, aplikasi mengisi sel a51 dengan angka 7. Kemudian aplikasi memeriksa sel berikutnya yaitu sel a61. Sel a61 merupakan sel kosong maka proses pencarian kemungkinan solusi dieksekusi pada sel a61. Setelah dilakukan ternyata tidak terdapat kemungkinan solusi, maka dilakukan *backtracking* ke sel sebelumnya yaitu sel a51. Pada sel a51 hanya terdapat 1 kemungkinan solusi, yang hal ini tidak mungkin mengubah kemungkinan solusi tersebut maka isi sel a51 dihapus dan dilakukan kembali proses *backtracking* kesel sebelumnya yaitu sel a41. Karena sel a41 merupakan angka awal yang tidak boleh diubah nilainya maka, dilakukan kembali *backtracking* ke sel sebelumnya yaitu a31 dengan tidak menghapus isi dari sel a41.

Pada sel a31 masih tersisa kemungkinan solusi yaitu angka 8, maka aplikasi mengisi sel a31 dengan angka 8. Kemudian aplikasi melanjutkan pemeriksaan kembali pada sel berikutnya yaitu sel a41. Proses pemeriksaan sel kosong, pencarian kemungkinan solusi, proses pengecekan kesamaan angka pada baris, kolom, *grid*, dan proses *backtracking* akan dilakukan terus-menerus hingga seluruh sel kosong terisi dengan angka yang unik, yaitu terdiri dari angka 1 sampai dengan angka 9 yang tidak memiliki kesamaan pada satu baris, kolom, dan *grid*nya.

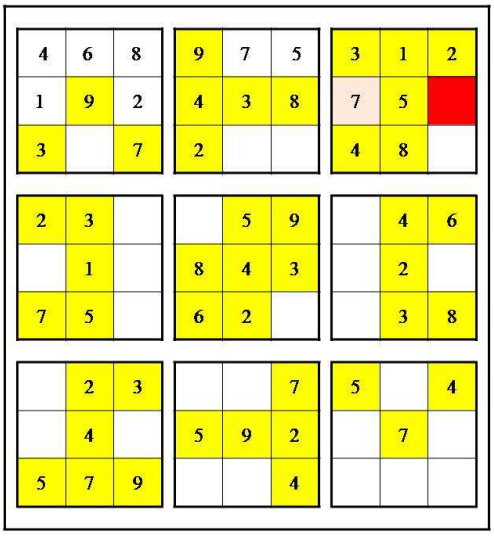
Pada gambar 10(a) - (i), proses pencarian solusi dimulai dari sel kiri atas yaitu sel a11 menuju sel kanan bawah yaitu sel a99. Kemungkinan solusi pada sel kosong disimpan didalam sistem data array, kemudian setelah semua sel kosong tersebut telah terisi dengan angka solusi dan tidak ada kesamaan angka pada baris, kolom, dan *grid*, solusi akhir baru ditampilkan, hal ini untuk mengoptimalkan kecepatan proses pencarian solusi dengan *backtracking algorithm*. Untuk sel yang terjadi proses *backtracking*, digunakan warna merah sebagai penanda. Untuk sel yang mengalami perubahan angka kemungkinan solusi, yaitu angka kemungkinan solusi yang lama diganti dengan angka kemungkinan solusi yang baru, digunakan warna biru sebagai penanda. Sedangkan warna merah muda digunakan sebagai sel awal pencarian hasil proses *backtracking*. Pada gambar 10 (a) - (h), terjadi proses *backtracking* sebanyak 8 kali. Setelah semua sel kosong telah terisi dengan angka kemungkinan solusi, tidak memiliki kesamaan angka pada baris, kolom, dan *grid* (gambar 10 (i)), maka angka kemungkinan solusi menjadi solusi akhir yang kemudian akan ditampilkan dengan sel berwarna hijau (gambar 10 (j)).



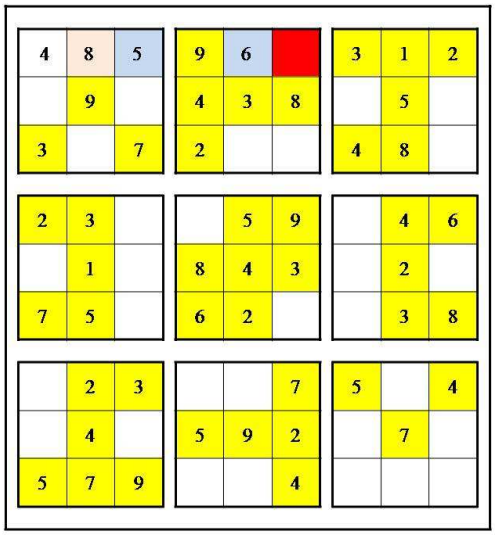
(a)



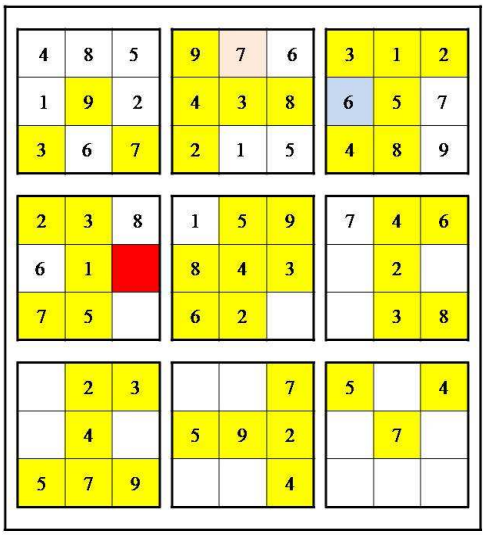
(b)



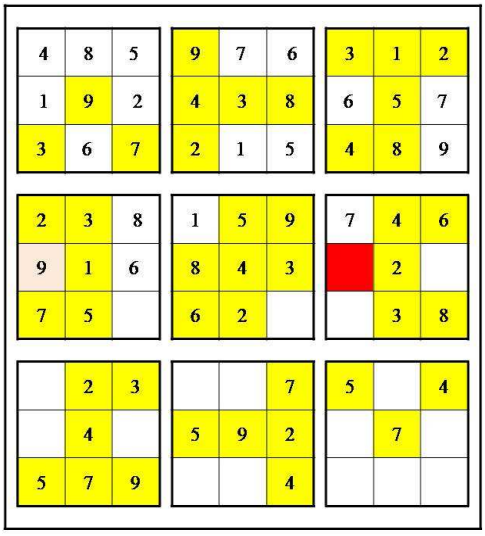
(c)



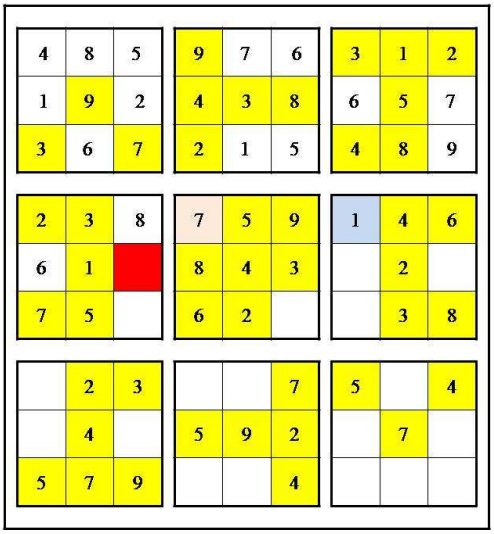
(d)



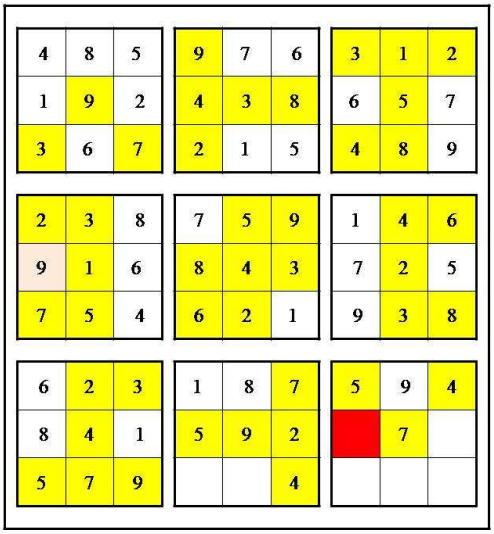
(e)



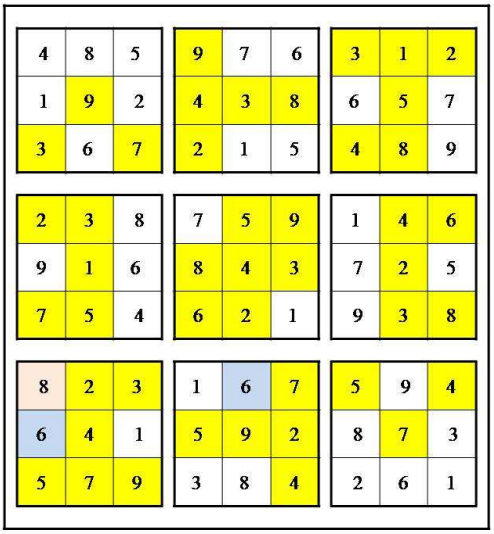
(f)



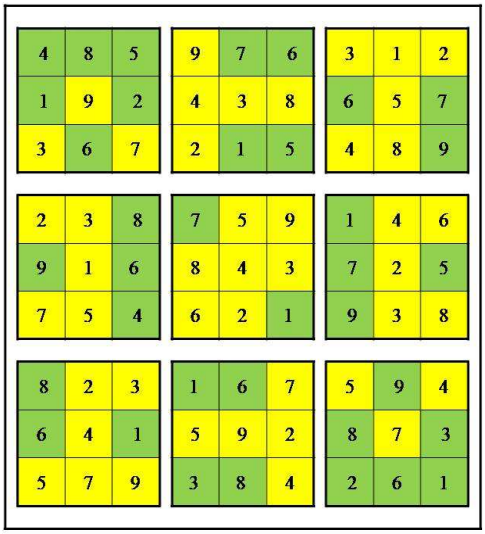
(g)



(h)



(i)



(j)

Gambar 10(a) - (j). Proses pencarian solusi *Su Doku*

KESIMPULAN

Berdasarkan penelitian yang telah dilakukan oleh penulis, dsapat diperoleh beberapa kesimpulan, diantaranya :

1. Telah dihasilkan sebuah aplikasi *Su Doku Solver*, implementasi dari metode *backtracking algorithm* untuk menampilkan solusi *Su Doku* pola 9x9 yang unik dan terbukti bahwa teori metode *backtracking algorithm* dalam penggunaan pencarian solusi *Su Doku* pola 9x9 cukup tepat.
2. Aplikasi *Su Doku Solver* ini memiliki fungsi sebagai penyedia solusi bagi permasalahan pencarian angka-angka yang valid didalam pengisian sebuah matriks *Su Doku*. Setelah memilih level *Su Doku*, *user* bisa langsung melihat tampilan solusinya ketika mengeksekusi perintah pencarian solusi *Su Doku*.
3. Metode umum pencarian solusi *Su Doku* adalah dengan kombinasi teknik pemindaian (*scanning*), penandaan (*marking*), dan analisis (*analyzing*).
   a. Pemindaian, berupa proses memindai baris atau kolom untuk mengindentifikasi baris mana dalam suatu *grid* yang terdapat angka-angka tertentu yang selanjutnya disebut angka awal. Proses ini kemudian diulang pada setiap kolom (atau baris) secara sistematis. Kemudian menentukan nilai dari suatu sel dengan membuang nilai-nilai yang tidak mungkin.
   b. Penandaan, berupa analisa logika dengan menandai kandidat angka yang dapat dimasukkan dalam sebuah sel.
   c. Analisis, berupa eliminasi kandidat, dimana kemajuan dicapai dengan mengeliminasi kandidat angka secara berturut-turut hingga sebuah sel hanya punya 1 kandidat.
4. Pencarian solusi *Su Doku* pada aplikasi *Su Doku Solver* dilakukan menggunakan *backtracking algorithm*/algoritma runut balik,

dimana pencarian dilakukan dari sel sisi kiri atas menuju sel sisi kanan bawah matriks, dengan rincian pencarian pada baris dimulai dari kiri ke kanan dan pada kolom dimulai dari atas kebawah. Pada implementasinya algoritma akan mengumpulkan semua angka kemungkinan yang akan menjadi solusi, kemudian menandai angka tersebut dan meletakkannya di dalam matriks *Su Doku*. Setelah itu algoritma akan memulai proses pencarian solusi pada sel pertama (sel kiri atas), jika sel pertama merupakan angka awal, maka pencarian dilakukan pada sel setelahnya, begitu seterusnya. Pada proses pencarian algoritma mengecek apakah angka yang ditandai tersebut memiliki kesamaan pada satu baris, kolom, dan *grid*. Apabila terdapat angka yang sama pada sel kedua, ketiga, dan seterusnya, algoritma akan melakukan *backtracking*/runut balik ke sel sebelumnya dan angka yang sama tersebut diganti dengan angka yang baru. Pengecekan terus dilakukan sehingga semua sel pada matriks *Su Doku* telah terisi semua dengan tidak ada kesamaan angka pada satu baris, kolom, dan *grid*.

5. Metode pencarian menggunakan *backtracking algorithm* mampu memberikan solusi *Su Doku* yang unik, yakni semua sel terisi masing-masing dengan angka 1 sampai dengan 9, dengan syarat tidak boleh ada angka yang sama pada satu baris, kolom, dan *grid.*

DAFTAR PUSTAKA

[1] Arifiyanto, W. A.. 2007. *Penggunaan Algoritma Backtracking Dalam Penyelesaian Permainan S Sudoku. Makalah STMIK 2007-096. Bandung : Teknik Informatika Institut Teknologi Bandung.*

[2] Desiani, A. dan Arhami, M.. 2006. *Konsep Kecerdasan Buatan*. Yogyakarta : Penerbit Andi.

[3] Morenvino, M., Ray, A. I., Anton, R. S.. 2006. *Penerapan Algoritma Runut-Balik Untuk Penyelesaian Teka-Teki Sudoku*. Makalah STMIK 2006-046. Bandung : Teknik Informatika Institut Teknologi Bandung.

[4] Musadik, R. 2010. *Pengantar dan Sejarah Sudoku***.** Diakses 22 Maret 2011. http://universologi.blogspot.com/2010/03/permainan-seperti-sudoku-sudah-dikenal.html.