PENCARIAN SOLUSI TTS ANGKA DENGAN ALGORITMA RUNUT BALIK BESERTA PENGEMBANGANNYA

Wahyu Fahmy Wisudawan

Program Studi Teknik Informatika Institut Teknologi Bandung, NIM: 13506113
Jl. Dago Asri 4 No. 14, Bandung
e-mail: al_izzatusysyifa@students.itb.ac.id

ABSTRAK

TTS angka merupakan permainan sederhana yang terdiri dari matriks persegi yang masing-masing persegi kecilnya harus diisi dengan satu digit bilangan berdasarkan daftar angka yang disediakan. Masalah ini dapat diselesaikan secara sederhana dengan algoritma runut balik. Algoritma ini diimplementasikan secara rekursif dan akan terus berproses sampai semua elemen matriks yang menjadi persoalan terisi. Untuk mengisi elemen matriks ini, digunakan fungsi kriteria yang mencari bilangan yang cocok untuk ditempatkan pada posisi yang sedang menjadi fokus pemrosesan. Jika tidak ada bilangan cocok yang ditemukan, maka terjadi runut balik. Untuk mempermangkus algoritma ini, salah satu cara yang dapat digunakan adalah dengan mengefektifkan perolehan fungsi kriteria. Dengan demikian, proses-proses komputasi yang tidak diperlukan dapat dilewati.

Kata kunci: Teka Teki Silang Angka, Runut Balik, *Backtracking*, Algoritma.

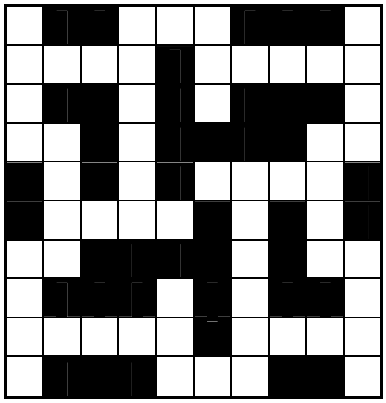
1. PENDAHULUAN

Teka teki silang (TTS) angka merupakan sebuah permainan yang populer di berbagai kalangan. Teka teki silang angka, atau selanjutnya kita sebut sebagai TTS angka, mirip dengan teka teki silang biasa yang umumnya merupakan teka-teki kata-kata. TTS angka terdiri dari beberapa deret tabel yang saling menyilang satu sama lain. Deret-deret tabel tersebut ada yang horizontal dan ada yang vertikal, dan dapat diisi dengan sederetan angka-angka tertentu yang telah ditentukan. Tantangan dari permainan ini adalah kita harus tepat mengisi TTS dengan bilangan-bilangan positif yang disediakan, yang umumnya telah ditentukan jenisnya, apakah termasuk angka yang mengisi matriks teka-teki secara mendatar atau mengisi matriks teka-teki secara menurun. Dalam bermain TTS angka, tidak diperlukan IQ tinggi dan pengetahuan yang luas. Permainan ini hanya membutuhkan keuletan, keterampilan, dan kecermatan saja.

Berikut ini salah satu contoh TTS angka dengan ukuran 10x10:

Tabel 1 Daftar Angka yang Disediakan

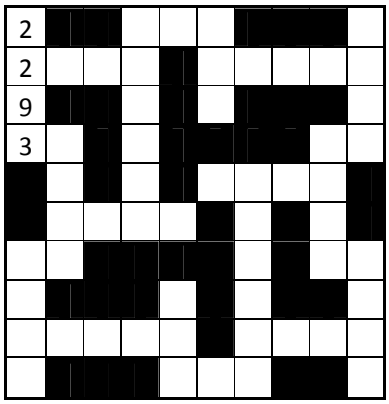
Jumlah Digit	Mendatar	Menurun
2 (dua)	24	-
	26	-
	34	-
	36	-
3 (tiga)	578	467
	794	859
4 (empat)	1282	2293
	5372	2532
	8416	3912
	9128	4384
	-	4936
	-	6484
5 (lima)	12036	-
	54789	-
6 (enam)	-	263794
	-	523571



Gambar 1. Contoh TTS angka

Matriks teka-teki di atas harus diisi dengan angka-angka yang telah disediakan di Tabel 1. Contoh, misalnya kita

mengisi secara vertikal 4 *grid* di sudut kiri atas dengan angka 2293. Hasilnya adalah sebagai berikut:



Gambar 2. Contoh TTS angka setelah satu pengisian

Permainan berakhir jika kita berhasil mengisi semua matriks TTS dengan angka-angka yang telah disediakan.

2. METODE

Metode yang digunakan di dalam makalah ini untuk menyelesaikan persoalan TTS angka di atas adalah algoritma runut balik (*backtracking*). Namun, algoritma ini dapat lebih dimangkuskan dengan beberapa metode yang nanti akan dijelaskan. Sebelum memasuki penjelasan algoritma tersebut, kita akan terlebih dahulu memasuki pembahasan struktur data untuk menyimpan soal TTS dan daftar bilanga yang tersedia di dalam memori komputer.

2.1 Struktur Data

Dalam bagian ini, akan dijelaskan rincian tentang struktur data yang akan digunakan untuk menyelesaikan masalah TTS angka ini. Struktur data yang pertama kali dibutuhkan adalah matriks penyimpan soal TTS dan matriks daftar angka. Sementara selama proses penyelesaian persoalan, dibutuhkan pohon dinamis yang mempresentasikan status pencarian solusi.

2.1.1 Matriks Soal TTS

Matriks ini berukuran *m* x *n*, dimana *m* adalah banyak baris TTS angka, dan *n* adalah banyak kolomnya. Matriks ini hanya berisi bilangan satu *digit*, sehingga untuk efisiensi memori, dalam implementasinya, lebih baik menggunakan tipe data primitif yang berukuran kecil yang didukung oleh bahasa tersebut. Misalnya *small byte* (yang hanya berukuran 8 bit) di bahasa pemrograman C#. Untuk mudahnya, selanjutnya kita akan menganggap bahwa elemen matriks ini adalah *integer*. Di dalam makalah ini, matriks ini kita beri nama `MatriksSoal`. Sementara untuk area yang diwarnai hitam di atas (tidak valid) direpresentasikan oleh matriks ini dengan bilangan negatif, misalnya -1.

2.1.2 Pasangan Matriks Daftar Angka

Matriks ini berisi daftar angka-angka yang disediakan untuk diisikan ke TTS dan tanda kalau elemen matriks yang berkesesuaian telah dipakai. Ada 4 matriks yang dibutuhkan. Yaitu matriks untuk bilangan yang harus diisi secara menurun dan matriks untuk bilangan yang harus diisi secara mendatar. Matriks untuk bilangan yang harus diisi secara mendatar kita beri nama `MatriksDaftarMendatar`, sementara matriks untuk bilangan yang harus diisi secara menurun kita beri nama `MatriksDaftarMenurun`.

Dua Matriks sisanya berkesesuaian dengan 2 matriks sebelumnya, dimana ukurannya sama, namun tipe datanya boolean. Matriks ini berfungsi sebagai penanda bahwa Matriks pasangannya dengan elemen yang berkesesuaian telah dipakai. Pasangan `MatriksDaftarMendatar` kita beri nama `MatriksDaftarMendatarIsDipakai`, dan pasangan `MatriksDaftarMenurun` kita beri nama `MatriksDaftarMenurunIsDipakai`.

Masing-masing matriks ini berukuran *p* x *q* dengan *p* adalah jumlah jenis bilangan (dilihat dari jumlah *digit* bilangannya) dan *q* adalah jumlah maksimum bilangan masing-masing jenis tersebut. Misalnya untuk contoh di atas kita membutuhkan `MatriksDaftarMendatar` berukuran 4 x 4 dan `MatriksDaftarMenurun` berukuran 3 x 5. Bilangan yang disediakan diasumsikan tidak dimulai dari 0.

2.2 Algoritma Runut Balik

Algoritma yang digunakan untuk pencarian solusi teka-teki ini adalah algoritma runut balik. Pertama-tama, prosedur menerima masukan i dan j yang merupakan penunjuk (*pointer*) yang menunjukkan posisi program saat itu memulai pencarian solusinya. Kemudian program memasuki pengulangan `while` yang akan terus terjadi jika `MatriksSoal` yang harus diisi belum diisi semua.

Selanjutnya dicek apakah `MatriksSoal[i,j]` merupakan area yang tidak valid (bukan merupakan area yang diperbolehkan untuk diisi) atau sudah diisi. Jika benar, program akan mencari posisi selanjutnya yang valid dan belum diisi. Jika salah, program akan memulai proses pencarian solusi.

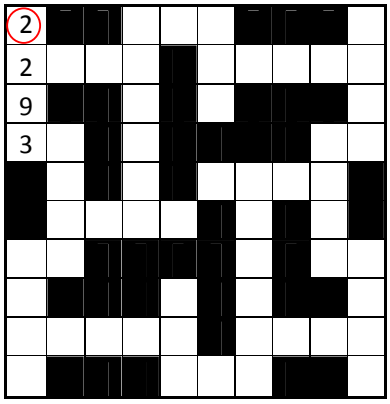
Solusi dicari dengan terlebih dahulu mengecek arah manakah yang memungkinkan untuk proses pengisian, apakah mendatar, ataukah menurun. Kemudian dicari manakah pangkal dan berapa panjang dari deret kotak

kosong tersebut. Setelah itu ada fungsi yang memperoleh bilangan yang sesuai panjang dan arahnya dari matriks penyimpan daftar bilangan yang disediakan (`MatriksDaftarMendatar` atau `MatriksDaftarMenurun` tergantung dari deret kotak kosong tersebut), dan angka pada bilangan tersebut berkesesuaian dengan angka-angka yang telah dimasukkan sebelumnya. Fungsi ini disebut fungsi kriteria. Setelah bilangan yang sesuai didapatkan, maka bilangan tersebut diisikan ke deret kotak kosong yang didapat sebelumnya. Setelah itu terjadi pemanggilan prosedur itu sendiri secara rekursif.

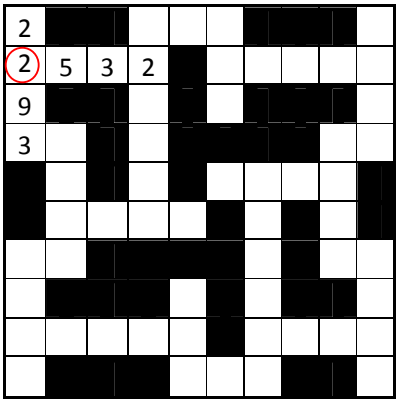
Namun, jika bilangan ini tidak didapatkan, maka pengisian solusi selangkah sebelumnya dihapus dan disinilah terjadi runut balik. Perhatikan algoritma di bawah ini:

```
procedure SolvingInBackTracking
(input: i: integer, j: integer)

Kamus:
isMendatar: boolean
p, q: integer
k : integer
temp : integer

Algoritma:
while (MatriksSoal belum diisi semua) do
    if  ((MatriksSoal[i,j]  tidak  valid)  or
(MatriksSoal[i,j] sudah diisi) ) then
        {cari i dan j yang mana
        MatriksSoal[i,j] belum diisi
        dan bukan tembok}
    else
        isMendatar := getposisiBil()
        setPosisiAwalBilangan(p,q)
        k := getPanjangBil(isMendatar, p, q)
        if  (mendapatkan  bilangan  yang  sesuai
panjang  dan  arahnya  dari  matriks  penyimpan
daftar bilangan yang disediakan) then
            temp := getBilCocok(isMendatar, k)
            {isi MatriksSoal[i,j] dengan temp
            digit per digit}
            {set Matriks penanda telah
            dipakai yang berkesesuaian}
            SolvingInBackTracking
(getPosisiX(), getPosisiY())
        else
            {tdk didapat bilangan yang sesuai}
            {hapus pengisian solusi
            sebelumnya dari MatriksSoal}
            {set Matriks penanda telah
            dipakai yang berkesesuaian}
        endif
    endif
endwhile
```

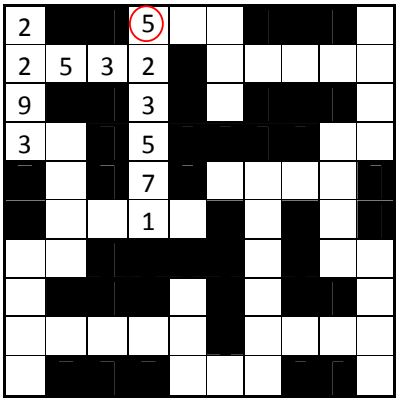
Contoh penggunaan algoritma ini untuk contoh soal pada tabel 1 dan gambar 2 digambarkan sebagai berikut:



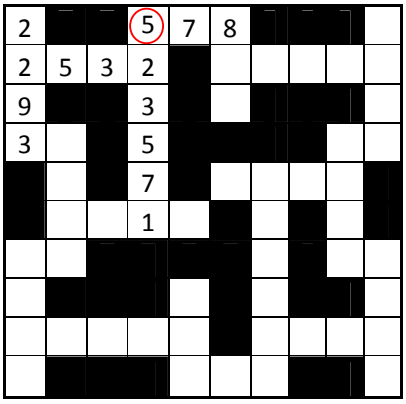
Proses pertama, pengisian pertama



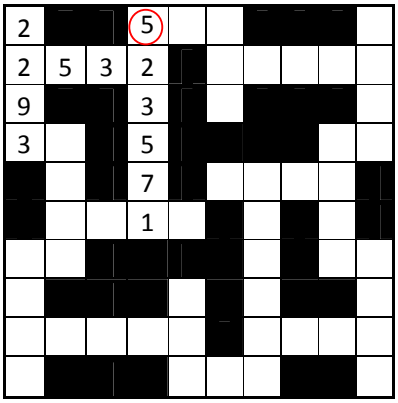
Proses kedua, pengisian kedua



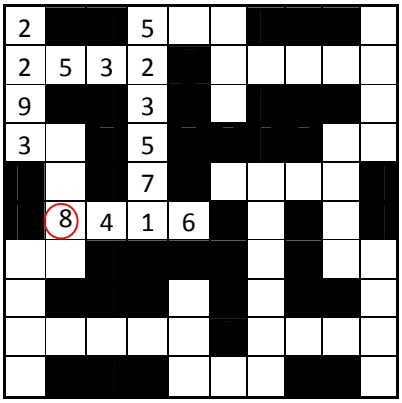
Proses ketiga, pengisian ketiga



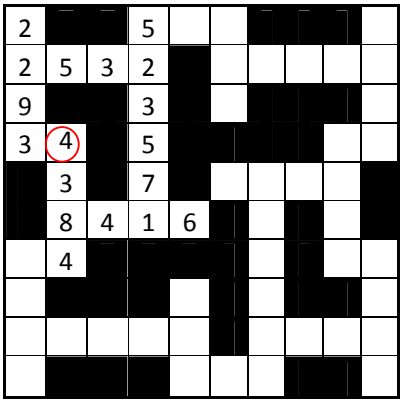
Proses keempat, pengisian keempat



Proses kelima, terjadi runut balik



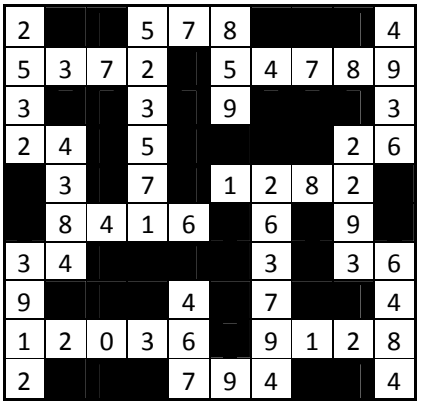
Proses keenam



Proses ketujuh

Gambar 3. Tujuh proses pertama pemecahan solusi persoalan TTS angka dengan algoritma runut balik

Pada gambar di atas, lingkaran merah menunjukkan posisi penunjuk (*pointer*) berada. Proses di atas akan terus berlangsung hingga didapatkan solusi berikut:



Gambar 4. Solusi akhir

2.3 Pengembangan Fungsi Kriteria

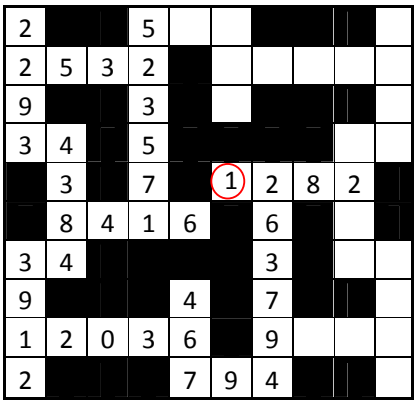
Algoritma di atas masih dapat dikembangkan dengan dengan pengembangan fungsi kriteria. Perhatikan proses ke-empat di atas. Sebenarnya penambahan bilangan "578" dapat ditolak karena dengan penambahan bilangan ini, maka tidak ada bilangan yang sesuai untuk proses selanjutnya, yaitu pada posisi (1,6). Sehingga penolakan sejak dini ini dapat memperingkas banyaknya proses yang diperlukan supaya solusi dapat dicapai.

Pengembangan baru spesifikasi fungsi pembatas dapat dijabarkan sebagai berikut:

1. Sesuai arahnya.
2. Sesuai panjangnya dengan panjang deret kotak kosong.

3. Angka pada bilangan tersebut berkesesuaian dengan angka-angka yang telah dimasukkan sebelumnya.
4. Jika ketiga kriteria di atas terpenuhi, maka dicek, apakah penambahan tersebut memungkinkan untuk menuju solusi.

Pengembangan fungsi kriteria di atas mempersingkat proses pencarian solusi dengan salah satunya melompati proses keempat yang digambarkan di atas. Contoh lainnya adalah proses ke-17 (yang tidak digambarkan di makalah ini). Perhatikan gambar di bawah ini:



Gambar 5. Proses ke-17

Pada gambar di atas, proses ke-17 menambahkan bilangan "1282" mulai dari posisi (5,6). Namun, penambahan ini tidak menuju solusi karena tidak ada bilangan pada daftar yang tersedia yang memenuhi kriteria untuk mengisi deret kotak kosong selanjutnya, yang awal posisinya dimulai dari (4,9).

3. KESIMPULAN

Teka teki silang (TTS) angka dapat dicari solusinya dengan algoritma runut balik. Algoritma ini sangat mudah dipahami dan diimplementasikan. Algoritma ini juga cukup fleksibel untuk dikembangkan, terutama pada fungsi kriterianya. Pengembangan fungsi kriterianya meliputi pengecekan sejak dini terhadap solusi yang diperoleh. Sehingga perolehan yang diketahui tidak menuju solusi tidak perlu diproses lebih lanjut.

REFERENSI

[1] Rinaldi Munir, "Strategi Algoritmik", ITB, 2007.