Penerapan Algoritma BFS dalam Penyelesaian *Rubik's Cube* dengan Langkah Minimal

Hafizh Dary Faridhan Hudoyo - 13514072
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13514072@std.stei.itb.ac.id

***Abstrak*—*Rubik's Cube* adalah *puzzle* tiga dimensi berbentuk kubus yang dapat diputar (*twisty puzzle*) untuk menyesuaikan warna di setiap sisinya. Dalam penyelesaiannya, perlu dilakukan algoritma agar mencapai keadaan semula. Dalam makalah ini akan dibahas tentang penyelesaian *Rubik's Cube* dengan metode yang dipakai banyak orang dan dengan menggunakan salah satu algoritma *first search*, yaitu BFS.**

***Kata Kunci*—langkah minimal, warna, metode, algoritma**

I. PENDAHULUAN

Rubik's Cube atau Kubus Rubik (selanjutnya disebut rubik) adalah salah satu jenis *puzzle* yang populer di dunia. Sampai sekarang, Rubik masih digemari banyak orang, termasuk penulis sendiri. Permainan ini dikabarkan dapat mengasah kemampuan otak manusia.

Apakah permainan ini dapat membuat manusia menjadi lebih pintar? Jawabannya tidak. Namun, permainan ini dapat membantu untuk mengasah kemampuan otak berkonsentrasi dalam mengeksplorasi solusi yang mungkin untuk memecahkan sebuah masalah. Semakin sering dimainkan, maka kemampuan dalam memecahkan masalah akan semakin cepat dan efektif serta efisien sehingga mengasah kemampuan berpikir manusia. Rubik juga dapat melatih koordinasi mata dengan tangan atau kaki (jika dimainkan dengan kaki). Di segi memori otak, Rubik juga dapat mengasah *short-term memory* dan *long-term memory* (berlaku untuk *Blindfolded Solving* atau menyelesaikan Rubik dengan mata tertutup).

Untuk seorang *speedcuber* (orang yang dapat menyelesaikan Rubik dengan cepat), bermain rubik juga dapat mengasah kecepatan dan kelenturan jari tangan sehingga, berdasarkan pengalaman pribadi penulis, tidak sedikit orang yang mengatakan bahwa seorang *speedcuber* dapat bermain alat musik yang menggunakan *key* atau *tuts* seperti piano dan sejenisnya, dan kenyataannya tidak selalu.

Adapun metode yang diciptakan untuk menyelesaikan rubik secara cepat seperti metode Fridrich dan metode Roux yang akan diulas sedikit pada bab selanjutnya. Metode yang dipakai untuk menyelesaikan rubik tidak hanya metode untuk menyelesaikan rubik secara cepat, ada juga metode yang diciptakan untuk menyelesaikan rubik dengan langkah seminim mungkin seperti metode Petrus dan metode Heise. Di dalam metode-metode tersebut terdapat algoritma yang dilakukan pada setiap langkahnya. Karena penyelesaian rubik dapat memakai algoritma, maka penulis tertarik untuk menerapkan salah satu algoritma yang telah diajarkan pada kuliah "Strategi Algoritma" untuk menyelesaikan rubik, yaitu algoritma *Breadth-First Search* (selanjutnya disebut BFS) yang tentunya dapat menghasilkan solusi dengan langkah minimal.

II. DASAR TEORI

A. Rubik's Cube

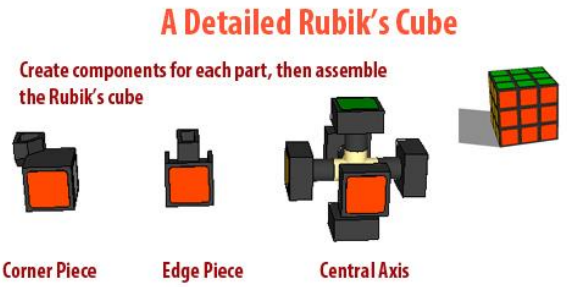
Rubik's Cube atau yang sering disebut rubik adalah salah satu jenis *puzzle* yang dapat diputar (*twisty puzzle*) yang berbentuk kubus dan memiliki enam warna yang mewakili masing-masing sisi pada rubik. Di Indonesia, sebutan "Rubik" biasanya mewakili semua jenis *twisty puzzle* yang beredar di pasaran, namun sebenarnya *Rubik's Cube* adalah *twisty puzzle* 3x3x3 karena berdasarkan namanya, *puzzle* kubus yang diciptakan oleh penemunya, Ernő Rubik, adalah *puzzle* yang memiliki dimensi 3x3x3.

Pada tahun 1974, Ernő Rubik, yang bekerja sebagai arsitek pada saat itu, membuat suatu objek yang sulit dipercaya bahwa benda tersebut dapat dibuat, yaitu sebuah kubus berlapis tiga pada setiap sisinya dan dapat diputar tanpa terdapat kerusakan. Setelah diberi stiker berwarna yang berbeda pada setiap sisinya, jadilah *puzzle* yang sangat sulit untuk dipecahkan.

Gambar 1: Ernő Rubik dan *Rubik's Cube* [2]

Struktur rubik terbagi menjadi beberapa bagian, yaitu *center*, *corner*, dan *edge* untuk bagian yang berwarna dan *core* serta komponen lain yang membuat bagian berwarna dari sebuah rubik dapat berputar dan memiliki konstruksi yang kokoh. *Center* adalah bagian tengah yang hanya memiliki satu warna pada rubik. *Corner* adalah bagian ujung yang memiliki

tiga warna pada satu *piece*-nya. *Edge* adalah bagian pinggir yang memiliki dua warna pada satu *piece*-nya.



Gambar 2: *Corner*, *edge*, dan enam *center* yang menempel pada *core* dari sebuah rubik

Sumber: http://www.sketchup-ur-space.com/2013/feb/img/Article1e-Level-of-Detail-Case-Study.jpg diakses tanggal 8 Mei 2016 pukul 21:22 WIB

Christopher Scott Vaughen, seorang profesor matematika dari Montgomery County Community College, membuat perhitungan untuk menemukan banyak permutasi yang mungkin pada sebuah rubik. Sebelum ia menemukannya, dikatakan pada tahun 1980 bahwa terdapat 3 milyar permutasi yang mungkin ditemukan pada sebuah rubik. Namun sebenarnya angka tersebut masih jauh dari angka sebenarnya, yaitu 43,252,003,274,489,856,000 (43 *quintillion*, 252 *quadrillion*, 3 trilyun, 274 milyar, 489 juta, 856 ribu) permutasi yang mungkin dan 1 permutasi yang disebut sebagai *solved state*.

Angka tersebut dicapai dengan menghitung kemungkinan yang dapat dicapai oleh setiap *piece* pada rubik. Diketahui bahwa jumlah *corner* dan *edge* pada sebuah rubik adalah delapan dan dua belas buah, sehingga didapat angka

$$8!12!$$

dari kemungkinan penempatan atau permutasi setiap *corner* dan *edge*. Namun, jika dieksplorasi lebih lanjut, terdapat kemungkinan permutasi yang tidak mungkin, yaitu hanya dua buah *corner* atau dua buah *edge* yang tertukar (tidak berlaku untuk keduanya sekaligus) sehingga angka tadi disederhanakan menjadi

$$\frac{8!\cdot 12!}{2}.$$

Lalu untuk orientasi, pada *edge*, terdapat dua buah warna sehingga didapat angka

$$2^{12}.$$

Namun, sebuah *edge* tidak dapat diubah orientasinya tanpa mengubah orientasi satu *edge* lain. Dengan kata lain, orientasi *edge* dapat berubah secara berpasangan sehingga angka tadi direduksi menjadi

$$2^{11}.$$

Untuk *corner*, terdapat tiga buah warna namun seperti halnya *edge*, orientasi dapat diubah secara berpasangan dan khusus untuk *corner*, orientasi yang dilakukan harus berlawanan satu sama lain, sehingga didapat angka

$$3^{7}.$$

Dari semua perhitungan di atas, didapat bahwa jumlah kemungkinan permutasi pada rubik adalah sekitar

$$\frac{12!\cdot 8!}{2}\cdot 2^{11}\cdot 3^{7} = 4.3\times 10^{19}.$$

B. Notasi dan Metode untuk Menyelesaikan Rubik

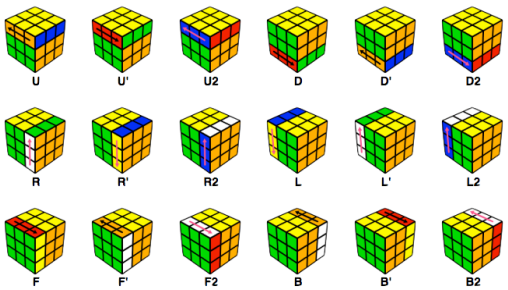
Untuk menyelesaikan rubik, bisa dilakukan secara eksplorasi. Namun, untuk kemudahan menyelesaikannya, dibuatlah metode penyelesaian rubik. Sebelum metode dibuat, terdapat notasi yang dipakai pada algoritma di dalamnya.

1. Notasi

   Notasi pada rubik adalah simbol-simbol yang mewakili setiap gerakan pada rubik. Notasi rubik berdasarkan regulasi *World Cube Association* (WCA) terbagi menjadi tiga bagian.

   a) Face Moves

   Terdiri atas enam gerakan dasar yang dilakukan searah jarum jam, yaitu F (*front face*), B (*back face*), R (*right face*), L (*left face*), U (*upper face*), dan D (*down face*). Jika notasi tersebut ditambahkan aksen (misalnya R') maka gerakan tersebut dilakukan berlawanan arah jarum jam. Jika notasi tersebut ditambahkan angka "2" (misalnya R2) maka gerakan tersebut dilakukan searah jarum jam atau berlawanan arah jarum jam sebanyak dua kali (tidak dipermasalahkan gerakan dilakukan searah atau berlawanan arah jarum jam karena *puzzle* berbentuk kubus). Berdasarkan *Outer Block Turn Metrics* (OBTM), gerakan ini memiliki nilai 1 *move*.



Gambar 3: Notasi *face moves* pada rubik

Sumber: http://i58.servimg.com/u/f58/18/49/85/51/notati10.png diakses tanggal 8 Mei 2016 pukul 20:40 WIB

   b) Rotation (Rotasi)

Terdiri atas tiga gerakan dasar, yaitu x atau [r] (searah R), y atau [u] (searah U), dan z atau [f] (searah F). Aturan aksen dan angka "2" juga berlaku untuk rotasi. Berdasarkan OBTM, gerakan ini memiliki nilai 0 *move*.

c) Outer Block Moves

Outer block moves adalah gerakan yang dilakukan seperti *face moves* namun mengikutsertakan lapis tengah. Gerakan ini ditulis dengan notasi seperti *face moves* dengan menambahkan "w" setelahnya (contoh: Fw, Rw', Uw2). Berdasarkan OBTM, gerakan ini memiliki nilai yang sama dengan *face moves*, yaitu 1 *move*. Namun pada 3x3x3, gerakan ini setara dengan melakukan *face moves* dan rotation secara bersamaan sehingga tidak akan dipakai pada makalah ini. Sebagai contoh, Uw setara dengan melakukan D y.

2. Metode yang Digunakan

Terdapat berbagai metode untuk menyelesaikan rubik. Pemegang rekor dunia pertama, Minh Thai asal Amerika Serikat dengan catatan waktu 22,95 detik, memakai metode yang tidak cukup terkenal, yaitu dengan cara menyelesaikan lapis atas, kemudian menyelesaikan empat *corner* pada lapis bawah, dilanjutkan dengan menyelesaikan *edge* pada lapis bawah, dan langkah terakhir adalah menyelesaikan *center* dan empat *edge* terakhir pada lapis tengah.

Metode yang sekarang paling banyak digunakan untuk *speedcubing* (menyelesaikan rubik secepat-cepatnya) adalah metode Fridrich (CFOP) yang diciptakan oleh Jessica Fridrich beserta variasinya seperti metode ZB (Zborowski-Bruchem) yang diciptakan oleh Zbigniew Zborowski dan Ron van Bruchem, dan metode Roux yang diciptakan oleh Gilles Roux (tidak dibahas pada makalah ini). Lalu, metode yang terkenal untuk menyelesaikan rubik dengan langkah minimal adalah metode Petrus yang diciptakan oleh Lars Petrus dan metode Heise yang diciptakan oleh Ryan Heise.

a) Metode Fridrich

Metode Fridrich atau CFOP (*cross-first two-layer-orientation of last layer-permutation of last layer*) adalah metode yang paling umum digunakan oleh seluruh *speedcuber* di dunia. Metode ini dikemukakan oleh Jessica Fridrich asal Republik Ceko. Rekor dunia rubik saat ini dipegang oleh Lucas Etter dengan catatan waktu 4,90 detik. Saat memecahkan rekor tersebut, Lucas memakai metode ini dengan variasinya yaitu metode ZB. Rekor nasional rubik saat ini dipegang oleh Vincent Hartanto Utomo dengan catatan waktu 6,61 detik pada kompetisi Bandung Open 2015 pada bulan September 2015 yang lalu. Ia juga memakai metode ini untuk memecahkan rekor nasional.

Gambar 4: *World Record Attempt* oleh Lucas Etter

Sumber: https://www.youtube.com/watch?v=vh0W8E4cNkQ

Gambar 5: Catatan waktu *World Record* yang dicetak Lucas Etter

Sumber: https://www.youtube.com/watch?v=vh0W8E4cNkQ

Metode ini dibagi menjadi empat tahap (tiga tahap jika memakai metode ZB, tidak dibahas pada makalah ini), yaitu

- *Cross*, yaitu menyelesaikan *edge* pada lapis bawah,
- *First Two Layer* (F2L), yaitu menyelesaikan dua lapis pertama pada rubik,
- *Orientation of Last Layer* (OLL), yaitu mengorientasikan semua *piece* pada lapis atas, dan
- *Permutation of Last Layer* (PLL), yaitu melakukan permutasi pada lapis atas agar semua *piece* tersusun dengan benar.

Metode ini memang umumnya digunakan untuk *speedcubing*. Namun, tidak sedikit *cuber* yang memakai metode ini untuk menyelesaikan rubik dengan langkah yang seminim mungkin. Terkadang ada kasus yang mengharuskan kita untuk memakai metode ini agar terselesaikan dengan langkah minimal.

b) Metode Petrus

Metode Petrus adalah pionir dari metode yang dikhususkan untuk menyelesaikan rubik dengan langkah minimal atau FMC. Salah satu *cuber* Indonesia yang mendalami FMC, Ardianto Satriawan (salah satu alumnus S2 Teknik Elektro ITB), memakai metode ini sebagai dasar melakukan FMC.

Langkah-langkah yang perlu dilakukan pada metode Petrus ini adalah

- Membuat blok 2x2x2,
- Perluas blok tersebut menjadi 2x2x3,
- *Flip bad edges*, yaitu memutar *edge* yang tidak menghadap dengan benar,
- Selesaikan dua lapis,
- Menempatkan *corner* sisa,
- Mengorientasi *corner* sisa,
- Menempatkan *edge* sisa.

c) Metode Heise

Metode Heise adalah metode lanjutan untuk melakukan FMC. Banyak *cuber* yang memakai metode ini untuk menyelesaikan rubik. Langkah-langkah pada metode ini adalah

- Membuat empat blok 1x2x2 (tiga blok 1x2x2 dibangun pada lapis luar, satu blok 1x2x2 dibangun pada lapis tengah),
- Memasangkan empat blok 1x2x2 dan mengorientasi *edge*,
- Selesaikan semua *edge* dan dua *corner* manapun,
- Selesaikan tiga *corner* terakhir dengan *commutator*.

Commutator adalah suatu gerakan dengan pola A B A' B' untuk, pada umumnya, menukar tiga buah *edge* atau tiga buah *corner*.

Dua metode terakhir umumnya tidak memiliki algoritma yang tertulis sehingga harus dilakukan secara intuitif. Rekor dunia FMC saat ini dipegang oleh Marcel Peters dan Tim Wong dengan catatan sembilan belas gerakan yang dilakukan.

Pada kompetisi resmi WCA, rubik diacak dengan 20-23 gerakan. Dengan jumlah gerakan pada acakan ini, dimungkinkan bahwa rubik dapat diselesaikan dengan dua puluh gerakan, bahkan kurang dari itu. Penjelasan ini akan dibahas pada poin selanjutnya.

C. God's Number

Dimulai pada tahun 1980, penelitian tentang jumlah gerakan minimal yang harus dilakukan untuk menyelesaikan rubik dilakukan. Dengan memakai notasi standar WCA, dibuktikan oleh Tomas Rokicki, Herbert Kociemba, Morley Davidson, dan John Dethridge bahwa rubik dapat diselesaikan dengan jumlah dua puluh gerakan atau kurang pada tahun 2010.

Mereka menggunakan komputer yang hidup selama 35 tahun tanpa henti. Cara mereka untuk mendapatkan hasil ini adalah

- Membagi posisi menjadi 2,217,093,120 set dengan 19,508,428,800 posisi dari setiap set,
- Mengurangi jumlah set dengan simetri (misal: dengan gerakan acakan yang sama, sisi acakan yang diawali dengan warna merah di depan akan berbeda hasilnya dengan warna hijau di depan, namun solusi keduanya sama, hanya saja orientasinya yang berbeda),
- Mendapatkan solusi yang tidak lebih dari dua puluh gerakan.

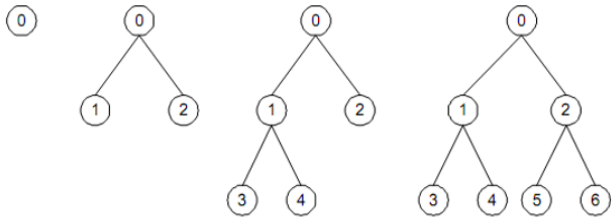
Distance	Count of Positions
0	1
1	18
2	243
3	3,240
4	43,239
5	574,908
6	7,618,438
7	100,803,036
8	1,332,343,288
9	17,596,479,795
10	232,248,063,316
11	3,063,288,809,012
12	40,374,425,656,248
13	531,653,418,284,628
14	6,989,320,578,825,358
15	91,365,146,187,124,313
16	about 1,100,000,000,000,000,000
17	about 12,000,000,000,000,000,000
18	about 29,000,000,000,000,000,000
19	about 1,500,000,000,000,000,000
20	about 490,000,000

Tabel 1: Jumlah gerakan yang dilakukan (*Distance*) dan jumlah posisi yang didapat (*Count of Positions*) [9]

D. *Algoritma* Breadth-First Search (BFS)

Algoritma BFS adalah algoritma yang memakai pohon sebagai model penyajian data dengan pengaksesan data yang memprioritaskan simpul tetangga terlebih dahulu

(menghabiskan satu *level* terlebih dahulu), lalu mengakses *level* selanjutnya jika tidak ada solusi dengan menghidupkan simpul anak dari *level* sebelumnya. Algoritma ini cocok dipakai untuk kasus penyelesaian suatu *puzzle* dengan langkah minimal.



Gambar 6: Pembentukan Pohon Status dengan BFS

Sumber: http://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2014-2015/BFS%20dan%20DFS%20(2015).pptx

III. Penerapan Algoritma BFS pada Rubik

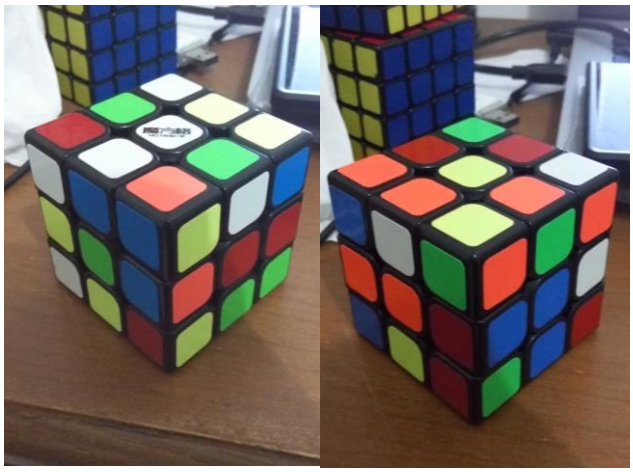
A. *Mengacak Rubik*

Langkah pertama yang dilakukan untuk bermain rubik adalah mengacaknya. Acakan dari WCA umumnya memiliki jumlah 20-23 gerakan. Sebagai contoh, penulis ambil acakan dari rekor dunia yang dicetak pada tahun 2015 oleh Tim Wong.

Acakan, menurut regulasi WCA, dilakukan dengan sisi hijau menghadap ke depan (F) dan sisi putih menghadap ke atas (U).

Acakan/*Scramble*:

R2 B' U2 L2 B2 D2 R2 B R' D2 B' D F D L B2 D' U2



Gambar 7: Tampilan rubik setelah diacak

B. *Solusi dari Tim Wong*

Langkah yang ia lakukan untuk menyelesaikannya adalah sebagai berikut.

- *Inverse* acakan di atas menjadi

  U2 D B2 L' D' F' D' B D2 R B' R2 D2 B2 R2 B' L2 U2 B R2

Gambar 8: Tampilan rubik setelah diacak dengan *inverse scramble*

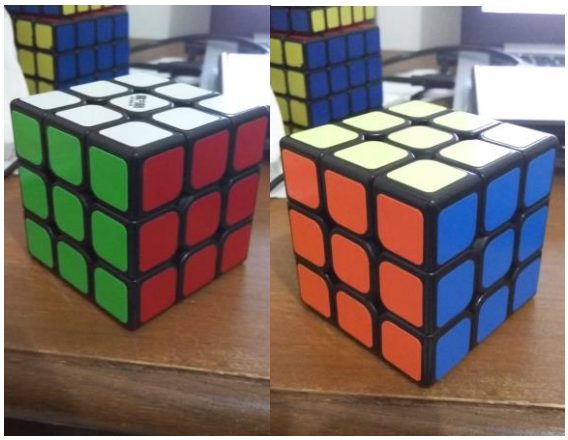
- Menyelesaikan F2L-1 (melakukan langkah *First Two Layer* dengan meninggalkan satu pair F2L yang tersisa)

  F2L-1 (i) : R2 F L2 B' R' F2 B L B2 (9/9)

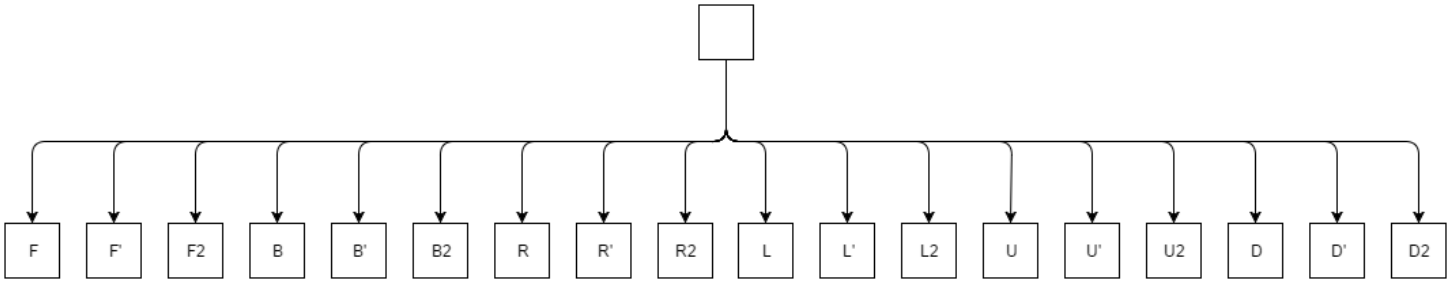
Gambar 9: Tampilan rubik setelah dilakukan F2L-1 (i)

- Menyelesaikan *Last Layer* dan rubik selesai.

  LL (i) : R D' F D2 F' L' D R' D' L (10/19)



Gambar 10: Tampilan rubik yang selesai setelah dilakukan langkah LL (i)



Gambar 11: Inisialisasi Pohon BFS (di dalam kotak merupakan string solusi)

- Karena solusi dari *inverse scramble* telah ditemukan, maka menyelesaikan acakan aslinya dilakukan dengan meng-*inverse* solusi dari *inverse scramble*, yaitu

  L' D R D' L F D2 F' D R' B2 L' B' F2 R B L2 F' R2 (19 *Moves*)

C. Solusi dengan Algoritma BFS

Acakan di atas dapat diselesaikan dengan algoritma BFS. Namun dibutuhkan banyak komputer untuk menemukan solusi optimalnya. Di sini, penulis mengasumsi bahwa solusi Tim Wong adalah solusi yang paling optimal untuk mempersingkat waktu pengerjaan. Adapun algoritma dengan pendekatan pemrograman berorientasi objek yang penulis buat sebagai berikut.

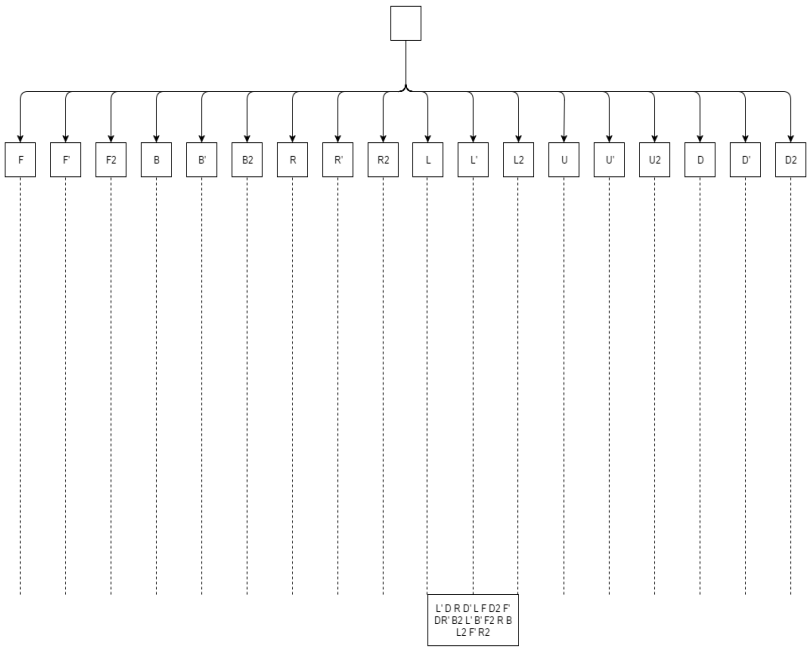
- Buat obyek bernama Cube dengan atribut:
  - F, B, R, L, U, D bertipe "sisi",
  - isF, isB, isR, isL, isU, isD bertipe boolean yang mengecek apakah langkah sebelumnya menggunakan gerakan yang dimaksud,
  - isParFB, isParRL, isParUD bertipe boolean yang mengecek apakah dua langkah sebelumnya adalah salah satu dari notasi yang dimaksud pada "isPar..." dan langkah sebelumnya adalah notasi lainnya yang dimaksud.
  - Move bertipe integer yang mencatat banyak langkah yang dilakukan.
  - Solution bertipe string yang mencatat string solusi.
- Buat prosedur bernama SolveCube dengan parameter "acakan" bertipe string.

  Pada prosedur ini, dilakukan pengacakan pada Cube. Setelah selesai, program akan membangkitkan anak berjumlah delapan belas yang mewakili F, F', F2, B, B' B2, R, R', R2, L, L', L2, U, U', U2, D, D', dan D2 seperti pada gambar 11.

  Pengecekan dimulai dari F sampai D2. Misal telah sampai pada *state* L', maka isL akan bernilai true sehingga langkah selanjutnya tidak bisa melakukan gerakan L dan variasinya.

  Sampai ke gerakan kedua, misal sampai pada *state* R, maka isL bernilai false, isR bernilai true, dan isParRL bernilai true sehingga tidak dapat melakukan gerakan L dan R lagi.

  Program terus melakukan algoritma tersebut sampai ditemukannya solusi seperti pada gambar 12. Lalu program akan mencetak string solusi dan jumlah gerakan pada layar.



Gambar 12: Solusi ditemukan (garis putus-putus menandakan ada langkah yang tidak diikutsertakan pada gambar)

IV. KESIMPULAN

Dengan mengimplementasikan algoritma BFS pada penyelesaian rubik, maka solusi yang optimal dapat dicapai. Walaupun begitu, diperlukan sangat banyak memori untuk dipakai dalam membuat pohon BFS sehingga perlu banyak komputer yang harus dipakai.

V. Ucapan Terima Kasih

Pertama-tama penulis mengucapkan terima kasih kepada Allah SWT karena telah memberikan rahmat dan berkat sehingga penulis dapat menyelesaikan makalah ini. Terima kasih juga kepada dosen pengajar Mata Kuliah IF2211 Strategi Algoritma yaitu Dr. Ir. Rinaldi Munir, M.T. dan Dr. Nur Ulfa Maulidevi, S.T., M.Sc atas segala bimbingannya selama perkuliahan di semester IV ini. Terima kasih juga kepada teman-teman Komunitas Rubik's Indonesia khususnya Vincent Hartanto Utomo dan Feby Kristianto yang telah mendukung dan membantu dalam pencarian data untuk bahan makalah ini. Terima kasih juga kepada teman-teman mahasiswa Teknik Informatika yang selalu mendukung dalam kuliah maupun luar kuliah.

VI. Referensi

[1] http://www.chalkstreet.com/blog/how-solving-the-rubiks-cube-can-make-your-brain-sharper/ diakses tanggal 8 Mei 2016 pukul 19:52 WIB

[2] https://www.rubiks.com/about/the-history-of-the-rubiks-cube diakses tanggal 8 Mei 2016 pukul 20:48 WIB

[3] http://faculty.mc3.edu/cvaughen/rubikscube/cube_counting.ppt diakses tanggal 7 Mei 2016 pukul 19:44 WIB

[4] https://www.worldcubeassociation.org diakses tanggal 8 Mei 2016 pukul 22:39 WIB

[5] http://www.alchemistmatt.com/cube/rubik.html diakses tanggal 8 Mei 2016 pukul 22:52 WIB

[6] https://www.speedsolving.com diakses tanggal 7 Mei 2016 pukul 09:14 WIB

[7] Munir, Rinaldi. 2009. Diktat Kuliah IF2211 Strategi Algoritma. Bandung: Program Studi Teknik Informatika

[8] http://lar5.com/cube/ diakses tanggal 8 Mei 2016 pukul 23:44 WIB

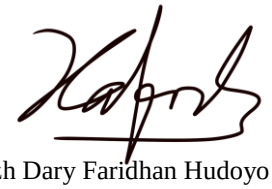
[9] http://www.cube20.org/ diakses tanggal 8 Mei 2016 pukul 22:19 WIB

[10] http://www.ryanheise.com/cube/heise_method.html diakses tanggal 8 Mei 2016 pukul 23:47 WIB

VII. Pernyataan

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 8 Mei 2016



Hafizh Dary Faridhan Hudoyo
13514072