

Aplikasi Tree Pada Pembuatan Behavior Tree

Putu Gde Aditya Taguh Widiana 13517032

Program Studi Teknik Informatika

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia

13517032@std.stei.itb.ac.id

Abstrak— Teknologi pada masa kini berkembang dengan sangat pesat, khususnya teknologi informasi. Perkembangan teknologi ini diiringi dengan munculnya berbagai perangkat lunak untuk berbagai keperluan, salah satunya perangkat lunak berupa *game*. Pada *shooter game* dan *role playing game*, tentunya ada NPC (*Non Player Character*) baik sebagai kawan maupun lawan. Perilaku NPC ini dapat dimodelkan dan ditentukan melalui *Behavior Tree*.

Kata Kunci— tree, behavior tree, game, npc

I. PENDAHULUAN

Pada masa kini, teknologi sudah berkembang sangat pesat, khususnya pada bidang teknologi informasi. Perkembangan ini diiringi dengan munculnya berbagai perangkat lunak yang dapat digunakan untuk berbagai keperluan, baik untuk bekerja, membantu dalam mengerjakan suatu hal, sampai untuk hiburan. Perangkat lunak untuk hiburan ini salah satunya adalah perangkat lunak berupa *game*.

Game sendiri terdiri dari berbagai macam genre, mulai dari action, adventure, strategy, shooter game, simulation, role playing game, dll. Pada umumnya, shooter game, role playing game, dan beberapa genre game lain pada umumnya, terdapat NPC (*Non Player Character*) yang dapat muncul sebagai lawan maupun kawan. NPC ini sering juga disebut sebagai AI (*Artificial Intelligence*), terkadang juga disebut sebagai bot.

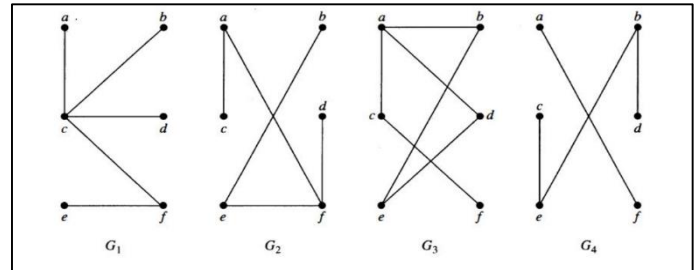
Seluruh perilaku NPC diatur oleh program yang telah dibuat oleh programmer. Pengaturan dari NPC ini dapat dibuat dan dimodelkan menggunakan tree, yang dinamakan behavior tree. Behavior tree ini yang akan menentukan alur perilaku dan aksi dari NPC.

II. LANDASAN TEORI

A. Tree (Pohon)

Tree (Pohon) adalah suatu cara untuk merepresentasikan suatu struktur dari suatu susunan yang dapat berisi berbagai macam informasi. Dinamakan pohon karena bentuknya yang menyerupai pohon, walaupun digambarkan secara terbalik. Konsep ini terapanannya sangat banyak, baik pada ilmu komputer, maupun pada disiplin ilmu lainnya. Dalam kehidupan sehari-hari, manusia telah lama menggunakan pohon untuk menggambarkan hirarkhi. Contohnya adalah pohon silsilah keluarga, struktur organisasi, organisasi pertandingan, dll. Para ahli Bahasa juga telah menggunakan pohon untuk menguraikan kalimat, yang disebut pohon parsing (parse tree).

Dalam ilmu matematika, tree (pohon) adalah suatu graf terhubung yang tidak mengandung sirkuit.



Gambar 1. Graf yang berupa tree dan graf yang bukan merupakan tree
Sumber : Matematika Diskrit Edisi ke-3

Pada gambar diatas, G1 dan G2 merupakan tree karena G1 dan G2 adalah graf tertutup yang tidak mengandung sirkuit. G3 bukan merupakan tree karena G3 mengandung sirkuit. G4 bukan merupakan tree karena G4 bukan graf terhubung.

Sifat-sifat dan properti pohon dapat dinyatakan dengan teorema di bawah ini :

Teorema. Misalkan $G = (V, E)$ adalah graf tak-berarah sederhana dan jumlah simpulnya n . Maka, semua pernyataan di bawah ini adalah ekuivalen:

1. G adalah pohon.
2. Setiap pasang simpul di dalam G terhubung dengan lintasan tunggal.
3. G terhubung dan memiliki $m = n - 1$ buah sisi.
4. G tidak mengandung sirkuit dan memiliki $m = n - 1$ buah sisi.
5. G tidak mengandung sirkuit dan penambahan satu sisi pada graf akan membuat hanya satu sirkuit.
6. G terhubung dan semua sisinya adalah jembatan.

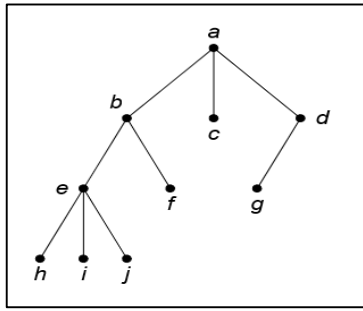
Semua poin-poin pernyataan di atas dapat dianggap sebagai definisi lain dari tree (pohon).

B. Spanning Tree (Pohon Merentang)

Spanning tree (pohon merentang) adalah pohon yang didapat dengan cara memutus sirkuit yang terdapat pada suatu graf terhubung. Graf yang terhubung sedikitnya memiliki hanya satu buah pohon merentang. Graf tidak terhubung yang mempunyai n bagian juga memiliki n pohon merentang, atau bisa disebut juga spanning forest (hutan merentang).

C. Rooted Tree (Pohon Berakar)

Tree ini merupakan salah satu jenis tree yang salah satu simpulnya diperlakukan sebagai akar dan sisi sisi lainnya diberikan arah sehingga menjadi graf berarah.



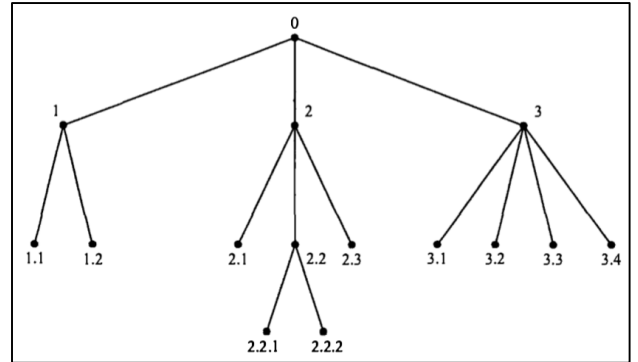
Gambar 2. Pohon Berakar
Sumber : Matematika Diskrit Edisi ke-3

Berikut ini merupakan beberapa istilah yang perlu dipahami dalam menggunakan rooted tree (gambar yang digunakan adalah gambar 2) :

1. Anak (child) dan orang tua (parent)
Simpul b,c, dan d merupakan anak dari simpul a, dan simpul a merupakan orang tua dari simpul b, c, dan d.
2. Lintasan (path)
Lintasan dari a ke h adalah a,b,e,h. Panjang lintasan dari a ke h adalah 4 (banyaknya simpul yang dilewati dari simpul awal sampai simpul tujuan).
3. Saudara kandung (sibling)
Simpul b adalah saudara kandung dari simpul c dan simpul d, tetapi simpul e bukan merupakan saudara kandung dari simpul g, karena orang tua dari kedua simpul tersebut berbeda.
4. Upapohon (subtree)
Pohon dengan simpul b sebagai akar merupakan upapohon dari pohon dengan simpul a sebagai akar.
5. Derajat (degree)
Derajat adalah jumlah anak dari simpul tersebut. Simpul a memiliki derajat 3, dan simpul b memiliki derajat 2.
6. Daun (leaf)
Daun adalah simpul dalam tree yang berderajat nol. Simpul h, i, j, f, c, dan g adalah daun.
7. Simpul dalam (internal nodes)
Simpul yang memiliki anak disebut simpul dalam. Simpul b dan d merupakan contoh simpul dalam.
8. Tingkat (level)
Tingkat adalah hitungan mengenai seberapa dalam suatu simpul dari akarnya, dengan akar memiliki tingkat 0. Simpul a adalah tingkat 0, simpul b, c, dan d adalah tingkat 1, simpul e adalah tingkat 2, dst.
9. Tinggi (height) atau kedalaman (depth)
Tinggi adalah tingkat tertinggi (terdalam) dari suatu pohon. Gambar 2 adalah pohon dengan tinggi 3.

D. Ordered Tree (Pohon berakar terurut)

Pohon berakar terurut adalah suatu jenis pohon yang memperhatikan keterurutan dari anak-anaknya. Pohon berakar terurut ini bertujuan agar data data yang terdapat dalam pohon tersebut terurut dari kiri ke kanan. Contoh dari penggunaan pohon berakar terurut ini adalah sistem pengalaman yang universal dari isi suatu buku dalam bab-bab dan subbab-subbab. Simpul akarnya diberi nomor 0, simpul lain yang merupakan anaknya diberi nomor 1,2,3,dst. Anak anak dari simpul 1 diberi nomor 1.1 , 1.2 , dst, anak-anak dari simpul 2 diberi nomor 2.1, 2.2, dst.

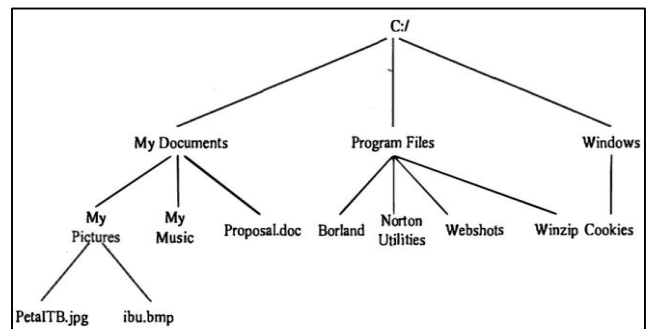


Gambar 3. Contoh dari penggunaan pohon berakar terurut untuk sistem pengalaman dari bab-bab suatu buku
Sumber : Matematika Diskrit Edisi ke-3

E. Pohon n -ary

Pohon berakar yang setiap simpulnya mempunyai paling banyak n buah anak disebut pohon n -ary. Jika $n=2$, maka disebut pohon biner, jika $n=3$, maka disebut pohon 3-ary.

Pohon n -ary ini sangat banyak aplikasi dan kegunaannya, beberapa diantaranya adalah untuk struktur organisasi, silsilah keluarga, bagan suatu pertandingan, untuk penurunan kalimat, untuk direktori komputer, dll.



Gambar 4. Pohon n -ary untuk pemetaan direktori komputer
Sumber : Matematika Diskrit Edisi ke-3

Pohon n -ary dapat disebut penuh ketika semua simpulnya memiliki tepat n buah anak. Pada pohon n -ary dengan tinggi h , jumlah daunnya adalah n^h . Tentunya,

ketika pohon n -ary tersebut tidak penuh, jumlah daunnya akan lebih sedikit dari n^h .

Pada pohon n -ary yang penuh dengan tinggi h ,

- Tingkat 0 → jumlah simpul = $n^0 = 1$
- Tingkat 1 → jumlah simpul = n^1
- Tingkat 2 → jumlah simpul = n^2
- ...
- Tingkat h → jumlah simpul = n^h

maka, jumlah seluruh simpul adalah

$$S = n^0 + n^1 + n^2 + \dots + n^h = \frac{n^{h+1} - 1}{n - 1}$$

F. Games

Game (Video Game) adalah game elektronik yang melibatkan interaksi dengan antar muka pengguna untuk menghasilkan umpan balik visual pada perangkat video seperti layar TV atau monitor komputer. Pada dasarnya, game bertujuan untuk menjadi hiburan (entertaining) bagi pemainnya. Perangkat elektronik yang biasanya digunakan untuk bermain video games biasanya disebut *platform*. Platform yang ada sekarang bermacam macam, mulai dari Playstation, Xbox, Nitendo, game PC, android, dll.

Video game dapat menggunakan beberapa jenis alat untuk mengubah masukan dari pemain menjadi suatu aksi dalam game. Beberapa alat yang paling umum digunakan adalah mouse dan keyboard untuk game berbasis PC, *gamepad* untuk game berbasis konsol.

Video game secara garis besar dapat diklasifikasikan menjadi 3, yaitu :

a. Casual games

Casual games dapat diartikan sebagai game yang sederhana. Seperti namanya, game semacam ini menghadirkan konsep yang sederhana dan mudah dimainkan di mana saja, dan kapan saja, tanpa perlu berpikir keras untuk memainkannya. Contoh game yang termasuk dalam kategori game ini adalah solitaire, minesweeper, game puzzle, mahjong, dsb.

b. Serious games

Serious games adalah jenis game yang bertujuan untuk menyampaikan informasi ataupun memberikan pengalaman belajar yang baru bagi pemainnya. Bahkan, pada beberapa serious game, bisa dibayangkan tidak bisa disebut sebagai game lagi dalam arti game yang sederhana. *Serious game* pada umumnya dibuat dengan maksud lebih dari sekedar untuk hiburan. Game semacam ini biasanya dimainkan oleh para profesional untuk meningkatkan kemampuan mereka dalam suatu bidang. Contoh dari game jenis ini adalah Microsoft Flight Simulator, Darfur is Dying, dan Finding Zoe.

c. Educational games

Seperti namanya, *educational games* adalah jenis game yang bertujuan untuk meningkatkan kemampuan dari pemainnya pada bidang tertentu yang dibawakan oleh game sebagai temanya. Educational games ini mulai tenar saat presiden Barack Obama menyampaikan kampanyenya yang bertajuk “*Educate to Innovate*” yang bertujuan untuk meningkatkan kemampuan pelajar Amerika dalam bidang teknologi, matematika, dan sains. Kampanye ini berisi tentang penggunaan game yang interaktif untuk meningkatkan kemampuan pelajar dalam bidang-bidang tersebut.

Video game juga terdiri dari banyak genre, beberapa genre video game antara lain :

- Action
- Adventure
- Fighting
- Platform
- Puzzle
- Racing
- Role-playing
- Shooter
- Simulation
- Sports
- Strategy



Gambar 5. Halo , salah satu game yang bergenre shooter
 Sumber : <https://www.digitaltrends.com/wp-content/uploads/2011/09/halo-combat-evolved-anniversary.jpg>
 Waktu akses : 8 Desember 2018 13.52 WIB

Dalam video game, tentunya ada tantangan yang harus dilewati oleh player untuk mendapatkan suatu *reward*. Tantangan-tantangan itu bisa bermacam-macam, contohnya pada shooter game ada musuh yang harus di kalahkan, pada role playing game ada boss yang harus dibunuh, pada game bergenre racing ada lawan yang harus di balap sehingga player bisa menjadi juara pertama dan memenangkan pertandingan. Lawan-lawan tersebut tentunya adalah NPC (*Non Player Character*). Istilah NPC ini kebanyakan berasal dari game bergenre role playing game, tetapi istilah ini sekarang sudah biasa digunakan untuk menggambarkan suatu entitas dalam game yang dikendalikan oleh program. NPC juga

terkadang disebut sebagai AI (Artificial Intelligence) dan juga terkadang disebut sebagai bot.



Gambar 6. Salah satu boss dalam game Darksouls 3 yang bergenre role playing, yang tentunya dikendalikan oleh program dari game tersebut.

Sumber : <https://www.technobuffalo.com/wp-content/uploads/2016/04/dark-souls-3-screenshot-saturday-01.jpg>
 Waktu akses : 8 Desember 2018 14.16 WIB

Ada banyak cara dalam memrogram dan mengendalikan NPC atau AI ini, salah satunya adalah dengan menggunakan behavior tree.

III. BEHAVIOR TREE

Behavior tree adalah suatu model matematis berbentuk tree yang dibuat untuk merepresentasikan perilaku dari suatu entitas yang diprogram. Behavior tree adalah sesuatu yang sangat fundamental dalam pengembangan game. Behavior tree mirip dengan finite state machine, hanya saja tidak seperti finite state machine yang berisi keadaan-keadaan, behavior tree berisi kumpulan instruksi dan perintah yang harus dijalankan oleh suatu entitas AI. Behavior tree banyak digunakan oleh pengembang game karena behavior tree mudah dimengerti oleh manusia sehingga tingkat kesalahan yang dibuat menjadi minimum. Behavior tree juga biasa digunakan dalam arsitektur kontrol (*control architecture*).

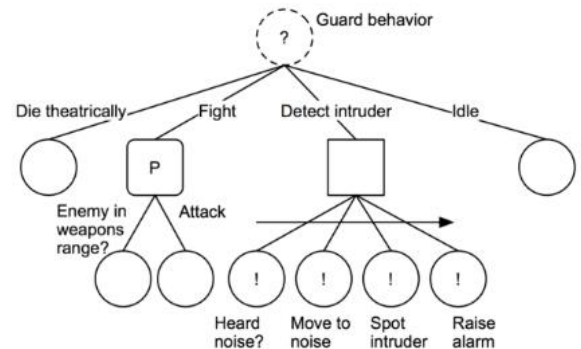
Behavior tree bisa menjadi tree yang sangat dalam, dikarenakan ada sangat banyak simpul-simpul yang terdapat dalam behavior tree, yang bertujuan untuk membuat suatu AI yang memiliki perilaku yang baik dan meyakinkan. Pengembangannya sangat mudah untuk dilakukan, dengan maksud bahwa pengembang dapat membuat perilaku yang sederhana terlebih dahulu, lalu mengembangkan alternatif-alternatif lain untuk mencapai kondisi keadaan yang diinginkan. Jika suatu perilaku gagal untuk dilakukan, AI akan mencari alternatif lain yang telah dibuat oleh programmer.

Berdasarkan cara penyimpanan dari behavior tree tersebut, secara garis besar behavior tree dapat dibedakan menjadi 2, yaitu :

- **Data Driven**
 Pada behavior tree jenis ini, tree disimpan secara eksternal dari *codebase*-nya, misalkan pada file XML

ataupun pada tipe file lainnya yang dapat diubah isinya dengan menggunakan editor eksternal.

- **Code Driven**
 Pada behavior tree jenis ini, tree didefinisikan secara langsung pada *codebase*-nya.



Gambar 7. Contoh Behavior Tree

Sumber : <https://answers.unity.com/storage/temp/28643-dungeon+guardian.png>
 Waktu Akses : 8 Desember 2018 21.49 WIB

Sebelum melangkah lebih jauh mengenai behavior tree, ada suatu istilah yang perlu diketahui, yaitu '*tick*'. *Tick* adalah suatu istilah dalam pembuatan game yang biasanya diartikan sebagai suatu satuan waktu. Istilah ini muncul dari suara jam berdetik ('tik' dan 'tok'). *Tick* secara khusus merupakan suatu perulangan (*loop*) dari suatu logic loop. Biasanya, satu *tick* melambangkan satu kali *loop* pada *main game logic loop*, dan pada setiap *loop* ini posisi karakter dan semua keadaan dalam game di update.

Pada penggunaan behavior tree secara sederhana, sistem akan menelusuri dari akar untuk mencari simpul manakah yang sedang aktif, mengecek kembali simpul yang telah aktif, sampai akhirnya mencapai simpul yang sedang aktif. Proses ini berjalan dalam satu *tick*. Ini adalah cara yang kurang efisien dalam melakukan perulangan. Dari pada melakukan penelusuran untuk setiap *tick*, akan lebih baik jika posisi simpul yang sedang aktif disimpan untuk digunakan kembali pada *tick* berikutnya, sehingga proses penelusuran dalam tree akan lebih efisien.

Pada behavior tree ada tiga status yang bisa dikembalikan (*return*) pada setiap *tick* dari suatu simpul menuju ke orang tua dari simpul tersebut, antara lain :

- Success**
 Memberikan informasi kepada orang tua dari simpul tersebut bahwa operasi yang dijalankan dalam simpul tersebut berhasil.
- Failure**
 Memberikan informasi kepada orang tua dari simpul tersebut bahwa operasi yang dijalankan dalam simpul tersebut gagal untuk dilakukan.

- c. Running
Memberikan informasi kepada orang tua dari simpul tersebut bahwa operasi yang dijalankan dalam simpul belum selesai, dan belum dapat ditentukan statusnya.

Pada setiap tick yang berjalan, simpul bisa di tick kembali dan pada saat itu akan dicek apakah statusnya sekarang adalah success, failure, atau running. Ini adalah inti dari behavior tree, karena memungkinkan pemrosesan simpul untuk tetap berjalan selama banyak tick permainan. Contoh : Sebuah simpul untuk melakukan aksi “berjalan” sedang dalam status *running* . Jika komputer gagal untuk mencari jalan untuk berjalan, maka status akan menjadi *failure*. Sebaliknya, jika komputer berhasil mencari jalan dan posisi target sama dengan posisi karakter sekarang, maka status yang di return akan menjadi *success*.

Simpul-simpul dalam behavior tree juga dapat dibagi menjadi 3 jenis, yaitu :

- a. Composite

Simpul jenis ini adalah simpul yang bisa memiliki lebih dari satu anak. Simpul ini akan memproses anak-anaknya baik secara terurut, maupun secara acak, tergantung kondisi yang sedang terjadi saat itu, dan saat prosesnya selesai, simpul ini akan mengembalikan nilai success ke orangtuanya jika prosesnya berhasil, dan dapat juga mengembalikan failure ke orangtuanya jika prosesnya gagal. Selama simpul ini memproses anak-anaknya, dia akan mengembalikan nilai running ke orangtuanya. Jenis-jenis dari simpul ini akan dijelaskan pada bagian berikutnya.

- b. Decorator

Simpul ini juga dapat memiliki anak, hanya saja anak yang dimiliki oleh simpul ini terbatas, yaitu hanya satu. Fungsinya bermacam-macam, bisa untuk mengubah status yang diterima dari anaknya, bisa untuk menghentikan proses dari anaknya, ataupun mengulang proses yang terjadi di anaknya. Itu semua bergantung pada jenis dari decoratornya. Jenis decorator yang digunakan pada umumnya adalah Inverter, yang berfungsi untuk mengubah hasil (result) yang diberikan oleh anak dari simpul tersebut. Jika anak berstatus fail, maka akan mengirimkan success ke orang tua dari simpul tersebut, begitu juga sebaliknya, jika anak berstatus success, maka simpul tersebut akan mengirimkan status failure ke orang tuanya.

- c. Leaf

Ini adalah simpul yang tidak memiliki anak. Akan tetapi, simpul ini adalah simpul yang paling berperan pada behavior tree, karena simpul jenis ini yang akan menentukan dan menetapkan jalannya game. Contohnya adalah leaf yang digunakan untuk berjalan. Simpul inilah yang akan benar-benar menjalankan aksinya untuk berjalan menuju poin tertentu pada peta yang telah didefinisikan

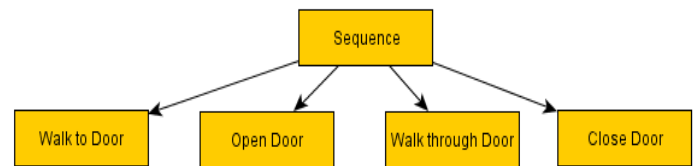
sebelumnya. Simpul ini dapat memiliki prosedur yang memiliki parameter, misalnya leaf untuk berjalan akan memiliki parameter berupa targetnya. Ada juga leaf yang fungsinya untuk memanggil dan menjalankan behavior tree yang lainnya, dan datanya diberikan pada behavior tree tersebut untuk diproses.

Composite Nodes (Simpul Gabungan)

Berdasarkan kegunaannya, simpul gabungan terdiri dari 2 jenis, yaitu :

1. Sequences

Simpul gabungan jenis ini menyatakan rangkaian dari suatu *task* yang perlu untuk dilakukan. Simpul ini akan mengunjungi semua anak anaknya satu per satu untuk melakukan operasi-operasi yang ada pada anak anak tersebut. Operasi-operasi yang ada pada anak anak tersebut harus dijalankan seluruhnya. Jika anak terakhir berhasil untuk dijalankan, maka simpul tersebut akan mengembalikan nilai success ke pada orangtuanya.



Gambar 8. Sequences Node

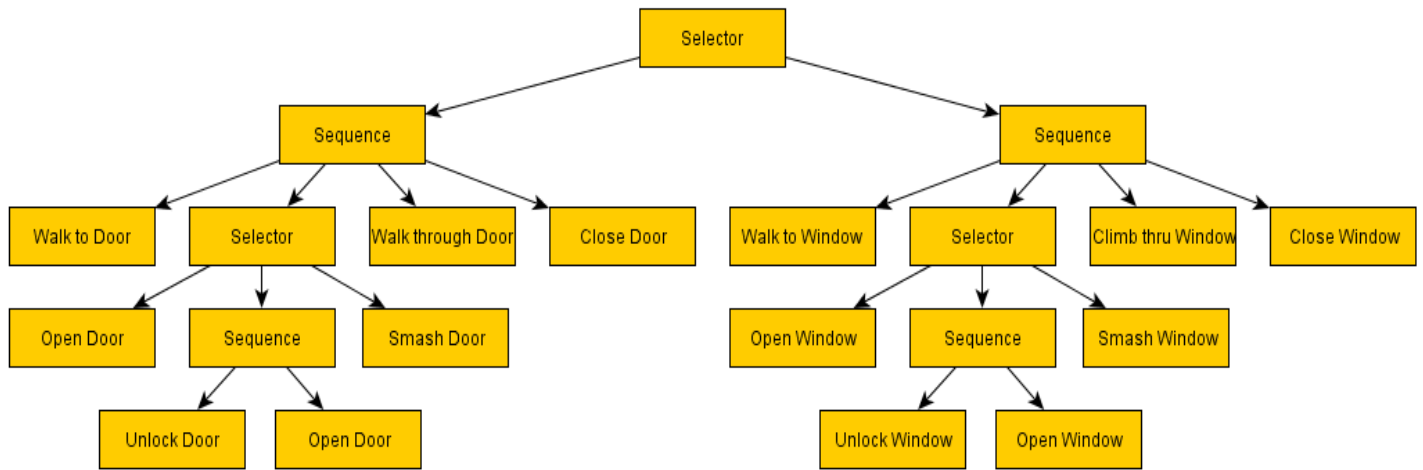
Sumber : <http://outforafight.files.wordpress.com/2014/07/image00.png>
Waktu Akses : 8 Desember 2018 21.59 WIB

Pada gambar di atas, terlihat bahwa ada empat anak dari simpul sequence tersebut, yaitu “Walk to Door”, “Open Door”, “Walk through door”, dan “Close door”. Semua anak-anak dari simpul ini akan dijalankan dengan urutan : Walk to Door → Open Door → Walk Through Door → Close Door . Jika salah satu anak menghasilkan failure, maka proses akan dihentikan. Misalnya proses buka pintu tidak dapat dijalankan, mungkin pintu tersebut dikunci sehingga tidak bisa dibuka dan tidak mungkin melangkah melewati pintu tersebut jika pintu tersebut tidak bisa dibuka.

Berikut adalah contoh pseudocode dari simpul sequence :

```

for i from 1 to n do
  childstatus ← Tick(child(i))
  if childstatus = running
    return running
  else if childstatus = failure
    return failure
end
return success
  
```



Gambar 9. Gabungan antara sequence dan selector
 Sumber :
<https://outforafight.files.wordpress.com/2014/07/selector3.png>
 Waktu Akses : 8 Desember 2018 22.22 WIB

2. Selector

Pada intinya, selector berfungsi untuk memilih atau menjalankan salah satu proses pada anak dari simpul tersebut. Selector seperti kebalikan dari sequences. Jika pada sequence akan menghasilkan failure jika salah satu anaknya menghasilkan failure, selector akan menghasilkan success jika hanya minimal satu dari anaknya menghasilkan success. Program akan mengecek dari anak pertama. Jika anak pertama mengembalikan nilai failure, maka akan lanjut ke anak kedua. Jika anak kedua menghasilkan success, maka simpul ini akan mengirimkan success ke orantuanya.

Berikut merupakan pseudocode dari selector :

```

for i from 1 to n do
  childstatus ← Tick(child(i))
  if childstatus = running
    return running
  else if childstatus = success
    return success
end
return failure

```

Gambar 9 merupakan contoh penggunaan behavior tree sederhana, yaitu terdapat simpul yang merupakan sequence dan selector. Behavior tree yang sesungguhnya tentunya jauh lebih kompleks dari gambar di atas, tetapi gambar ini dapat menjelaskan secara sederhana bagaimana suatu NPC atau entitas AI berperilaku.

IV. KESIMPULAN

Behavior tree merupakan salah satu aplikasi dari penggunaan tree di bidang informatika. Penggunaan tree untuk menentukan perilaku dari entitas AI cukup efektif karena mudah dimengerti dan mudah untuk dikembangkan. Semakin dalam suatu behavior tree, maka akan semakin menghasilkan AI yang lebih kompleks dan lebih baik. Tentunya aplikasi tree tidak hanya pada behavior tree, tetapi masih banyak aplikasi tree yang lainnya.

V. PENUTUP

Saya mengucapkan terima kasih kepada semua pihak yang secara langsung maupun tidak langsung telah membantu kelancara pembuatan makalah ini. Saya juga berharap bahwa makalah ini dapat berguna bagi pembaca. Mohon maaf jika ada informasi yang dirasa kurang tepat.

VI. REFERENSI

- [1]. Munir, Rinaldi. *Matematika Diskrit Edisi ke-3*
- [2]. Crawford, Chris (1982). *The Art of Computer Game Design*.
- [3]. Salen, Katie; Eric Zimmerman (2005). *The Game Design Reader: A Rules of Play Anthology*. The MIT Press. ISBN 978-0-262-19536-2.
- [4]. Williams, J. P., & Smith, J. H., eds. (2007). *The players' realm: studies on the culture of video games and gaming*. Jefferson, N.C.: McFarland & Co.
- [5]. Colledanchise Michele, and Ögren Petter 2017. *Behavior Trees in Robotics and AI: An Introduction*.
- [6]. Colledanchise, Michele and Ögren Petter. "How Behavior Trees Generalize the Teleo-Reactive Paradigm and And-Or-Trees" In *Intelligent Robots and Systems (IROS 2016)*, IEEE/RSJ International Conference on, 2016.

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 Desember 2018

Putu Gde Aditya Taguh Widiana
 13517032