Aplikasi Pohon Keputusan dalam Alogritma Mobil Otomatis

Ayrton Cyril 13516019
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13516019@std.stei.itb.ac.id

***Abstrak*—Seiring dengan perkembangan zaman, semakin banyak teknologi modern yang ditemukan. Teknologi terbarukan itu didukung dengan perkembangan pada dunia robotika yang membuat segala hal dapat berjalan secara otomatis. Mulai dari pembersih lantai, penyapu jalan, hingga mobil otomatis. Mobil otomatis yang sedang dikembangkan saat ini diharapkan dapat membantu manusia dalam berkendara, selain membantu mobil ini diharapkan dapat berkendara secara aman dan tertib. Dalam pembuat mobil otomatis ini diperlukan suatu algoritma untuk memutuskan kondisi tertentu, seperti saat mobil berada pada lampu merah, mobil melihat orang menyebrang dan sebagai. Seorang *programmer* mobil otomatis ini dalam penyusunan algoritma mobil otomatisnya memerlukan bantuan dari pohon keputusan. Dengan membuat pohon keputusan maka pembuatan algoritma lebih mudah dan terstruktur. Pembuatan pohon keputusan juga diggunakan saat *programmer* menghadapi dilema akibat dari suatu kondisi, dan diperlukan analisis terkait kode etik maka dari itu dengan pohon keputusan. Kenyamanan dalam berkendaraan juga diperhitungkan dalam pembuatan mobil otomatis maka dari itu dibuthkan pohon keputusan yang mengecek setiap kondisi secara teratur.**

***Kata kunci*—Pohon, pohon keputusan, mobil otomatis, algoritma**

I. PENDAHULUAN

Kendaraan merupakan suatu kebutuhan primer saat ini. Beragam mobil sudah keluar mulai dari yang untuk kalangan menegah hingga mobil mewah yang harganya milyaran rupiah. Saat ini mobil terbaru yang sedang dikembangkan adalah mobil otomatis. Perkembangan teknologi yang maju saat ini mengakibatkan munculnya mobil otomatis. Mobil otomatis saat ini merupakan terobosan baru yang diharapkan dapat membantu manusia dalam berkendara. Saat ini mobil otomatis yang sedang berkembang salah satunya buatan perusahaan internet terbesar yaitu Google.

Mobil otomatis saat ini dituntut dapat berkendara dengan aman dan taat pada peraturan yang berlaku. Mobil ini dapat memberikan rasa aman saat berkendaraan karena diharapkan dapat mengurangi *human error* yang sering menyebabkan terjadinya kecelakaan. Kecelakaan akibat mengantuk di jalan, maupun akibat menggunakan *handphone* dapat teratasi dengan menggunakan mobil otomatis ini. Penyebab kecelakaan lainnya adalah akibat dari pelanggaran lalu lintas, seperti menerobos lampu maupun mengebut melebihi batas kecepatan. Dengan adanya penerapan mobil otomatis disetiap kendaraan diharapkan dapat mewujudkan ketertiban dalam berlalu-lintas.

Penerapan mobil otomatis saat ini membutuhkan suatu algoritma yang dapat memutuskan pergerakan dari mobil tersebut, seperti sekarang mobil itu harus berbelok ataupun berenti semua diatur oleh suatu algoritma. Algoritma lain seperti saat mobil ingin berbelok kiri , kita harus melihat dulu lokasi mobil kita berada di lajur paling kiri atau tidak. Apabila belum di paling kiri kita harus berpindah jalur terlebih dahulu setelah itu kita melihat lampu lalu lintas bila bewarna hijau maka kita akan berbelok kiri apabila belum bewarna hijau maka kita akan menunggu lampu berubah warna menjadi hijau.

Ada berbagai cara untuk menyusun alogirtma yang ada salah satunya dengan pohon keputusan. Penerapan pohon keputusan dapat menjadi sarana dalam memutuskan kendaraan harus melakukan apak tergantung dengan kondisi yang terjadi saat itu. Pada makalah ini penulis akan menjelaskan cara menggunakan pohon keputusan dalam penyusan algoritma mobil otomatis

II. LANDASAN TEORI

2.1. Graf

Graf G = (V, E), yang dalam hal ini: V = himpunan tidak-kosong dari simpul-simpul (vertices) = { v1 , v2 , ... , vn } E = himpunan sisi (edges) yang menghubungkan sepasang simpul = {e1 , e2 , ... , en }[1]

2.2. Jenis-jenis Graf

Berdasarkan ada tidaknya gelang atau sisi ganda pada suatu graf, maka graf digolongkan menjadi dua jenis:

1. Graf sederhana (simple graph).
   Graf yang tidak mengandung gelang maupun sisi-ganda dinamakan graf sederhana.
2. Graf tak-sederhana (unsimple-graph).
   Graf yang mengandung sisi ganda atau gelang dinamakan graf tak-sederhana (unsimple graph).

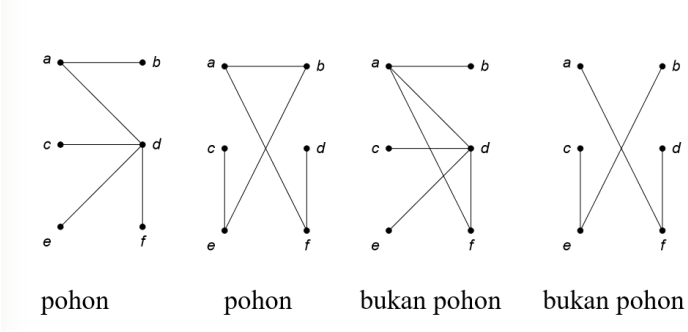
Berdasarkan orientasi arah pada sisi, maka secara umum graf dibedakan atas 2 jenis:

1. Graf tak-berarah (undirected graph)
   Graf yang sisinya tidak mempunyai orientasi arah disebut graf tak-berarah.

2. Graf berarah (directed graph atau digraph)
   Graf yang setiap sisinya diberikan orientasi arah disebut sebagai graf berarah.

2.3. Pohon

Pohon adalah graf yang khusus. Pohon memiliki definisi sebagai berikut, pohon adalah graf tak-berarah terhubung yang tidak mengandung sirkuit.[2]



Gambar 1. Jenis pohon
(Sumber: http://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2013-2014/Pohon%20(2013).pdf)

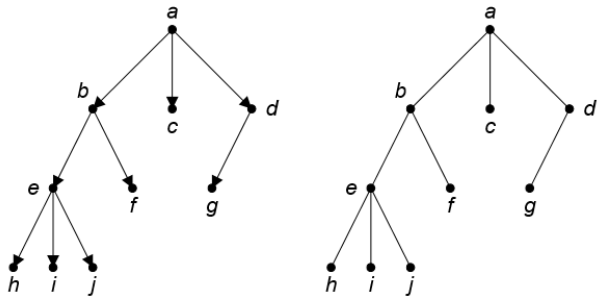
Karena definisi pohon diacu dari teori graf, maka sebuah pohon dapat mempunyai hanya seuah simpul tanpa sebuah sisipan. Dengan kata lai, jika G = (V,E) adalah pohon, maka V tidak boleh berupa himpunan kosong, namun E boloh kosong. Pada sebagian litelatur, pohon yang dimaksudkan pada gambar 1, sering disebut juga pohon bebas(*free tree*) untuk membedakannya dengan pohon berakar(*rooted tree).* Pohon juga sering kita definisikan sebagai graf tak-berarah dengan sifat bahwa hanya terdapat sebuah lintasan unik antara setiap pasang simpul. Pada gambar 1 Setiap simpul pada pohon terhubung dengan lintasan tunggal. Selain itu, kita juga melihat bahwa di dalam pohon, jumlah sisinya adalah jumlah simpul dikurangi satu.

Jadi dapat kita simpulkan bahwa sifat-sifat pohon sebagai berikut:

1. Setiap pasang simpul di dalam pohon terhubung dengan lintasan tunggal
2. Pohon terhubung dan memiliki sisi sebanyak simpul dikurang satu.
3. Pohon tidak mengandung sirkuit.
4. Penambahan satu sisi pada pohon akan membuat hanya satu sirkuit.
5. Pohon terhubung dan semua sisinya adalah jembatan(jembatan adalah sisi yang bila dihapus akan menyebabkan graf terpecah menjadi dua komponen).

2.2. Pohon berakar

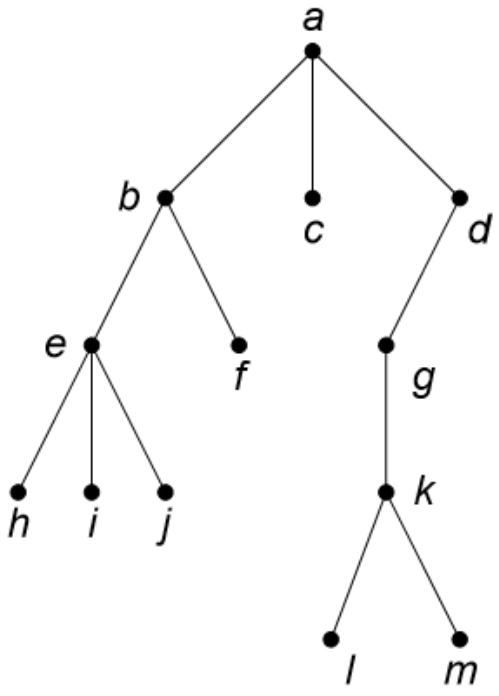
Pada kebanyakan aplikasi pohon, simpul tertentu diperlakukan sebagai akar(*root).* Sekali sebuah simpul ditetapkan sebagai akar, makan simpul-simpul lainnya dapat dicapai dari akar dengan memberi arah pada sisi-sisi pohon yang mengikutinya. Definisi pohon berakar adalah sebagai berikut, pohon yang sebuah simpulnya diberlakukan sebagai akar dan sisi-sisinya diberi arah sehingga menjadi graf berarah dinamakan pohon berakar(*rooted tree).*



Gambar 2. Jenis pohon berakar
(Sumber: http://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2013-2014/Pohon%20(2013).pdf)

Akar mempunyai derajat-masuk sama dengan nol dan simpul-simpul lainnnya berderajat masuk sama dengan satu. Simpul yang memiliki derajat-keluar sama dengan nol disebut daun atau simpul terminal. Simpul yang mempunyai derjat-keluar tidak sama dengan nol disebut simpul dalam atau simpul cabang. Setiap simpul di pohon dapat dicapai dari akar dengan sebuah lintasan tunggal(unik). Kita dapat melihat bahwa pada gambar 2 pohon berakar yang sebelah kiri meiliki arah sedangkan yang kanan tidak. Seluruh arah pada pohon berakar berasal dari akar hingga keseluruh daunnya tidak ada yang kearah akar, maka dari itu sesuai perjanjian kita dapat menghilangkan arah dari pohon berakar dan menggambanya seperti pada pohon di gambar 2 yang sebelah kanan.

Pohon memiliki beberapa terminologi sebagai berikut:

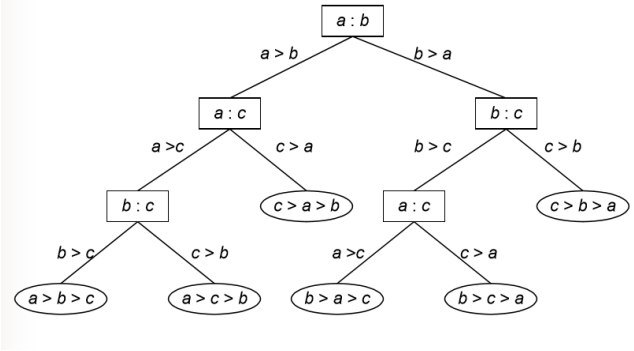


Gambar 3. Terminologi pada pohon
(Sumber: http://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2013-2014/Pohon%20(2013).pdf)

1. Anak(*child* atau *children*) dan orangutan(*parent*)
   Seperti dapat dilihat pada gambar 3,b,c dan d adalah anak-anak simpul dari a, dan a adalah orangutan dari anak-anak itu. Simpul h, i, j, l, dan m tidak mempunyai anak.
2. Lintasan (*path*)
   Lintasan dari simpul v_1 ke simpul v_k adalah runtunan dari simpul-simpul $v_1,v_2,v_3,\ldots,v_k$ sedemikian sehingga v_1 adalah orangtua dari v_{1+1} untuk I lebih besar sama dengan 1 dan lebih kecil sama dengan k. dari gambar 3 dapat dilihat bahwa lintasan dari a ke j adalah a, b, e , j. Panjang lintasan adalah jumlah sisi yang dilalui dalam suatu lintasan. Maka Panjang lintasan dari a ke j adalah tiga.
3. Keturunan (*descendant*) dan leluhur (*ancestor*)
   Jika terdapat lintasa dari simpul x ke simpul y didalam pohon, maka x adalah leluhur dari simpul y, dan y adalah keturunan simpul x. pada gambar 3, b adalah leluhur simpul h, dan dengan deimikian h adalah keturunan b.
4. Saudara kandung (*sibling)*
   Simpul yang berorangtuan sama adalah saudara kandung satu sama lain. Pada gambar 3, f adalah saudara kandung e, tetapi g bukan saudara kandung e, karena orangtua mereka berbeda.
5. Tinggi (*height*) dan kedalaman (*depth*)
   Aras maksimum dari suatu pohon disebut tinggi atau kedalaman pohon tersebut. Atau, dapat dikatakan, tinggi pohon adalah Panjang maksimum lintasa dari akar ke daun. Pohon di gambar 3 mempunyai tinggi 4.
6. Upapohon(*subtree*)
   Misalkan x adalah simpul di dalam pohon T. yang dimaksud dengan uppohon dengan x sebagai akarnya ialah upgraf T' = (V',E') sedemikian sehingga V' mengandung x dan semua keturunannya dan E' mengandung sisi-sisi dalam semua lintasan yang berasal dari x. sebagai contoh, T' = (V',E') adalah uppohon dari gambar 3 dengan V' = {e, h, i ,j} dan E' = { (e, h), (e, i), (e, j).
7. Derajat(*degree*)
   Derajat sebuah simpul adalah jumlah upapohon (atau jumlah anak) pada simpul tersebut. Derajat a adalah 3, derajat b adalah 2, Derajat d adalah satu dan derajat c adalah 0. Jadi, derajat yang dimaksudkan di sini adalah derajat-keluar. Derajat maksimum dari semua simpul merupakan derajat pohon itu sendiri. Pohon di gambar 3 berderajat 3. Karena derajat tertinggi dari seluruh simpulnya adalah 3.
8. Daun (*leaf*)
   Simpul yang berderajat nol (atau tidak mempunyai anak) disebut daun. Pada gambar 3 simpul h, i, j, f, c, l, dan m adalah daun.
9. Simpul dalam (*internal nodes*)
   Simpul yang mempunyai anak disebut simpul dalam. Pada gambar 3 simpul b, d, e, g, dan k adalah simpul dalam.
10. Aras (*level*) atau tingkat
    Akar mempunyai aras = 0, sedangkan aras simpul lainnya = panjang lintasan dari akar ke simpul tersebut. Pada gambar 3 simpul b memiliki aras = 1 sedangkan simpul k memiliki aras 3.

2.3. Pohon keputusan

Pohon keputusan digunakan untuk memodelkan persoalan yang teriri dari serangkaian keputusan yang mengarah ke solusi. Tiap simpul menyatakan keputusan, sedangkan daun menyatakn solusi. Sebagai contoh, kita ingin mengurutkan tiga buah bilangan, a, b, dan c.



Gambar 4. Pohon keputusan dalam mengurutkan tiga bilangan
(Sumber: http://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2013-2014/Pohon%20(2013).pdf)

Pohon keputusan diatas memuncul semua kemungkinan yang dapat terjadi dari pengurutan tiga bilangan. Setiap kemungkinan dimunculkan dengan cara membandingkan nilai dari setia bilangan dan kita memutuskan langkah selanjutnya berdasarkan hasil perbandingan dua bilangan tersebut

III. APLIKASI POHON KEPUTUSAN DALAM ALGORITMA MOBIL OTOMATIS

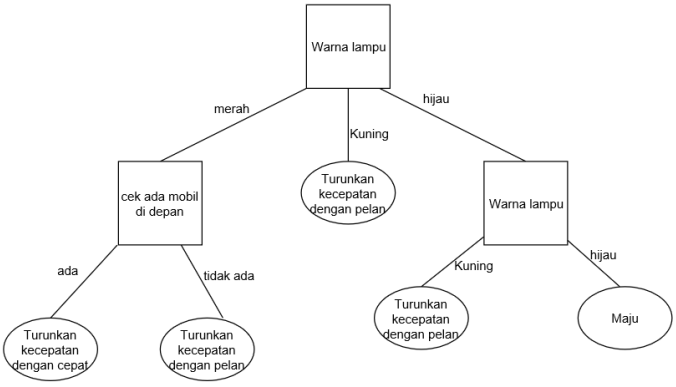
Setelah melihat kegunaan pohon keputusan, kita dapat melihat banyak manfaat dalan penggunaan pohon keputusan. Penganalisaan suatu kondisi untuk memutuskan tidakan yang paling tepat merupakan salah satu pengaplikasiaan pohon keputusan. Maka dari itu seiring dengan perkembangan mobil otomatis di dunia saat ini para *programmer* membutuhkan bantuan pohon keputusan untuk menyusun algoritma yang ingin mereka ciptakan. Seluruh algoritma itu harus dapat mencakup segala kondisi yang akan terjadi pada situasi di jalan. Demi menjaga mobil tetap aman dan tetap mematuihi aturan yang berlaku setiap mobil harus memutuskan sesuatu keputusan yang mangkus dan sangkil. Keputusan seorang *programmer* dalam memutuskan tindakan mobil akibat dari suatu kondisi sangat berpengaruh dalam keselamatan penumpang dari mobil mereka, seperti apakah yang harus dilakukan setiap melalui lampu merah? Keputusan yang salah dapat berakibat fatal pada penumpang mobil otomatis maupun masyarakt sekitar, maka dari itu dengan bantuan pohon keputusan *programmer* diharapkan dapat menganalisis dengan baik segala kondisi yang akan terjadi dan memutuskan mobil harus melakukan apa dengan sebaik mungkin.

Selain membantu *programmer* memutuskan suatu tindakan akibat dari suatu kondisi, pohon keputusan dapat diggunakan sebagai bahan pertimbangan saat *programmer* dihadapkan pada keputusan-keputusan yang membutuhkan berbagai pertimbangan dari kode etik dan peraturan. Sebagai contoh saat urusan yang menyangkut nyawa seseorang. *Programmer* harus meminta berbagai pendapat dari orang-orang dengan menunjukan berbagai pohon keputusan yang telah mereka buat

dan Bersama memutuskan pohon keputusan mana yang terbaik bagi semua orang.

IV. STUDI KASUS APLIKASI POHON KEPUTUSAN PADA ALGORITMA MOBIL OTOMATIS

Seperti sudah dijelaskan diatas penerapan pohon keputusan dapat digunakan pada algoritma mobil otomatis. Pohon keputusan digunakan saat mobil kita dihadapkan suatu kondisi dan mobil harus dapat memutuskan tindakan apa yang harus dilakukan agar pergerakan mobil selalu menaati peraturan dan mobil dapat berkendara secara aman. Penerapan pohon keputusan ini pada algoritma diterapkan pada kondisi *if..else state*. Setiap kali kondisi terjadi maka diterapkan suatu kondisi. Dalam penerapannya mobil saat bergerak akan dihadapkan oleh berbagai kondisi. Kondisi yang kita bicarakan saat ini adalah kondisi yang membutuhkan tindakan dari mobil otomatis kita kibat dari kondisi yang terjadi. Kondisi seperti ada berbagai jenis, karena kita mementingkan banyak aspek dalam pembuatan mobil otomatis ini, seperti aspek keamanan, kenyamanan dan ketertiban mobil dalam berkendara. Kita tidak ingin mobil kita berhenti mendadak saat melihat polisi tidur bukan? Atau kita tidak ingin mobil kita menabrak kendaraan yang berhenti mendadak. Tentu saja yang terakhir kita tidak ingin melanggar lalu-lintas akibat dari program mobil otomatis kita tidak dapat memproses kondisi tertentu, seperti pada kondisi lampu lalu-lintas.



Gambar 5. Pohon keputusan saat di lampu lalu-lintas
(sumber:koleksi pribadi)

Pohon keputusan pada gambar 5 menjelaskan cara kerja algoritma mobil otomatis. Pertama mobil mengecek warna dari lampu lalu-lintas, bila warna merah maka mobil akan mengcek lagi mobil didepannya, bila tidak ditemukan mobil di depannya maka mobil akan menurunkan kecepatans ecara perlahan, tapi apabila ada mobil di depannya maka mobil akan menurunkan kecepetan mobilnya lebih cepat. Saat lampu bewarna kuning mobil akan melambat secara perlahan. Saat lampu bewarna hijau mobil akan tetap berjalan maju apabila warna lampu berubah menjadi warna kuning maka mobil akan melambat apabila tetap bewarna hijau mobil akan tetap maju.

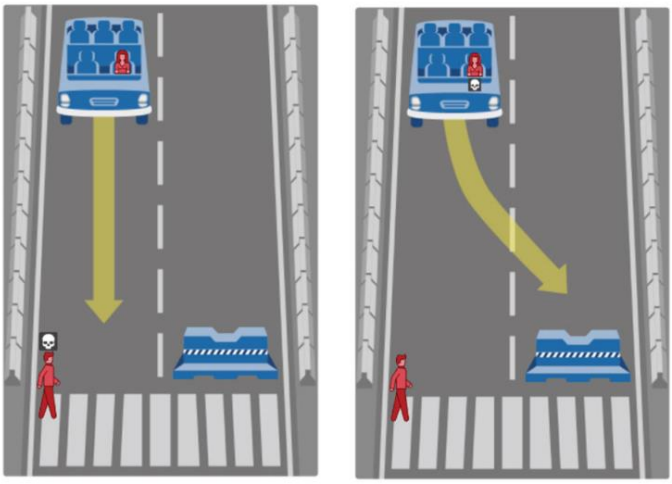
Penerapan keputusan seperti diatas tidak akan terlalu sulit apabila kita sudah daapt meprediksikan kondisi yang akan terjadi. Pada kondisi gambar 5 kita sudah tau bahwa dalam perjalan pasti akan ditemukan sebuah persimpangan dan disana akan terdapat sebuah lampu lalu-lintas. Kondisi yang sulit adalah apabila kondisi adalah kondisi secara tiba-tiba. Seperti saat ada orang menyebrang secara tiba-tiba maupun mobil di depan kita berhenti secara mendadak. Kondisi diatas bila diterapkan pada pohon keputusan sebagai berikut:

Gambar 6. Pohon keputusan saat ada objek(mobil, orang, dll) muncul secara tiba tiba
(sumber:koleksi pribadi)

Bila saat berkendara mobil melihat objek muncul didepanya secara tiba-tiba mobil akan langsung mengecek jarak mobil kita dengan objek didepan kita. Apabila jaraknya masih memungkinkan untuk mengerem secara perlahan maka mobil kita akan mengerem secara perlahan. Apabila tidak memungkinkan maka mobil akan mengecek kondisi jalur disebelah kanannya. Bila jalur kanan dari mobil kosong maka mobil akan berpindah jalur kekanan. Pengecekan pertama jalur kanan karena jalur kanan adalah jalur untuk menyusul kendaraan lain jadi apabila objek didepan kita adalah mobil maka kita tidak melanggaran peraturan dengan menyusul mobil di depan dari arah kanan. Selanjutkan apabila jalur kanan terisi, maka mobil harus mencari jalan lain untuk menghindar yaitu jalur disebelah kirinya. Apabila jalur disebelah kirinya kosong maka mobil dapat berpindah ke jalur kiri tapi apabila tidak maka mobil harus mengecek mobil dibelakangnya. Saat mobil mengecek kondisi mobil belakangnya berarti mobil berada pada kondisi tidak bias menghindar maka dari itu mobil harus mengerem. Apabila jarak mobil kita dengan mobil dibelakang kita cukup jauh kita dapat mengerem secara mendadak, tapi apabila jaraknya tidak terlalu jauh kita tidak bisa mengerem mendadak karena akan menimbulkan tabrakan beruntun, maka dari itu kita akan memperlambat mobil secara cepat dan sambal menyalakan lampu peringatan agar mobil dibelakang waspada. Perlakukan ini dilakukan agar apabila terjadi tabrakan sekalipun

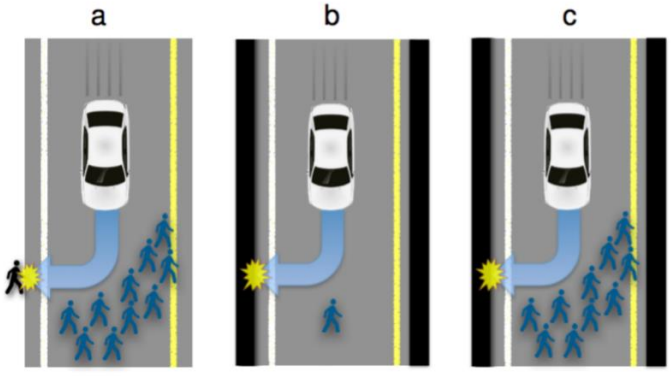
dampak yang terjadi dibuat seminimal mungkin.

Penerapan pohon keputusan dalam algoritma mobil otomatis tidak semudah yang kita bayangkan. Dalam litelatur yang penulis baca ada yang disebut dilema *programmer*. Dilema ini terjadi saat *programmer* diberikan suatu kondisi tertentu yang sangat sulit untuk dibuat pohon keputusannya seperti dapat dilihat pada gambar berikut.



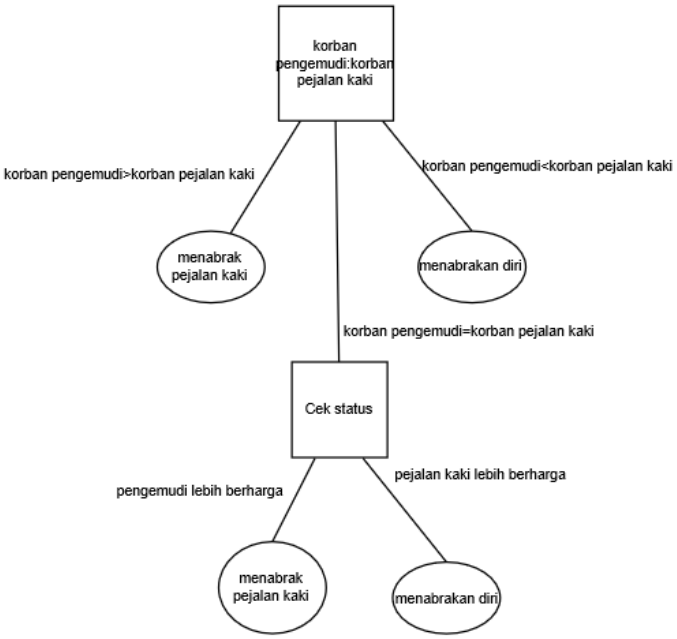
Gambar 7. *Programmer dilemma*
(Sumber: http://www.businessinsider.sg/self-driving-cars-already-deciding-who-to-kill-2016-12/?r=US&IR=T)[4]

Gambar 7 diatas menaruh mobil pada suatu kondisi tertentu yaitu mobil dalam kondisi tidak dapat menghindar. Apakah mobil akan menabrak pejalan kaki atau ia akan menabrakan dirinya sendiri untuk menyelamatkan pejalan kaki? Pohon keputusan pada kondisi ini akan sulit karena membutuhkan kode etik dan moral. Apabila kita sedang berkendara secara manual apapun yang kita lakukan merupakan reaksi refleks dari kondisi tersebut, baik saat kita menabarak pejalan kaki maupun menabrak diri tergantung dari refleks kita. Tapi saat ini kita berbicara soal mobil otomatis. Mobil otomatis tidak memiliki refleks tentunya, semua yang mobil otomatis lakukan sudah direncanakan saat penyusunan algoritma. Jadi saat kondisi diatas apabila kita menabrak pejalan kaki maka kondisi itu sudah direncanakan jauh sebelum peristiwa itu terjadi. Sebagai *programmer* mobil otomatis kita dituntut untuk membuat algoritma yang paling baik. Dalam gambar 6 pohon yang dibuat tidak dapat digunakan pada kondisi di gambar 7. Pohon yang harus dibuat untuk menerapkan kondisi pada gambar 7 harus memuat simpul lain. Setelah dalam pertimbangan diputuskan mobil harus menyelamatkan nyawa sebanyak mungkin seperti pada contoh berikut.



Gambar 8. *Programmer dilemma ethics*
(Sumber: https://www.technologyreview.com/s/542626/why-self-driving-cars-must-be-programmed-to-kill/)

Gambar 8 menunjukan pada kita bagaimana mobil seharusnya bergerak. Maka dari itu kita dapat memuat pohon keputusan dengan lebih mudah setelah mengetahui *ethic rule* dari kondisi tabrakan yang tidak bisa dihindari. Sekarang kita akan membuat pohon keputusan terhadap kondisi gambar 7

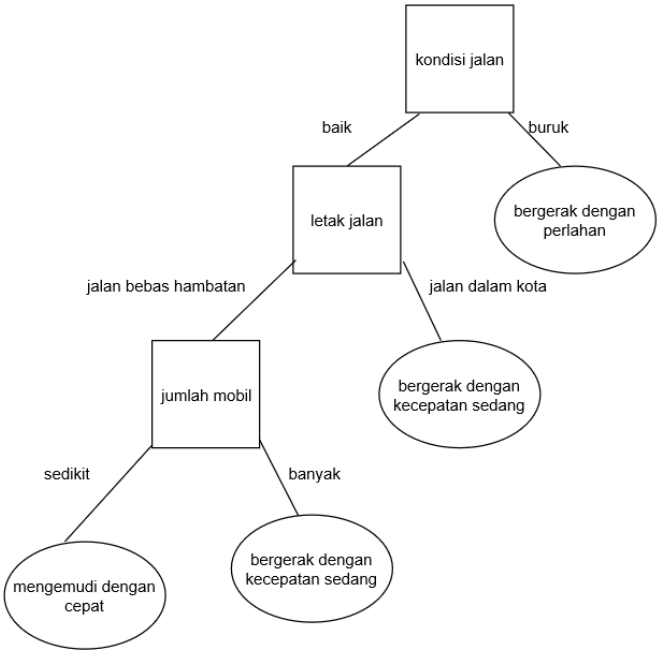


Gambar 8. pohon keputusan
(sumber:koleksi pribadi)

Dalam pohon keputusan yang penulis buat berdasarkan literatur yang penulis baca. Mobil harus dapat memilih korban seminimal mungkin. Pertama-tama tentu mobil harus mendahulukan penjalan kaki yang berjalan tanpa pengaman, tapi mobil harus dapat memperhitungkan jumlah korban akibat dari tindakannya. Misalakan mobil bergerak dengan kecepatan tinggi yang dapat menyebabkan kematian apabila mobil menabrakan dirinya sendiri. Mobil harus dapat memprediksi berapa jumlah korban akibat menbrakan diri ke tembok, setelah itu mobil juga harus dapat menghitung jumlah pejalan kaki yang akan tertabrak. Setelah mengetahui jumlahnya mobil akan memilih yang jumlah kematian korban lebih sedikit. Apabila setelah perhitungan dengan memperhitungkan peralatan keselamatan dalam mobil didapat jumlah kematian lebih banyak daripada jumlah pejalan kaki maka mobil akan menabrak pejalan kaki, sedangkan bila lebih sedikit maka mobil akan menabrakan dirinya sendiri. Sekarang bagaimana bila jumlah sama banyak? Litelatur lain mengatakan kita harus memperhitungkan status dari korban kita. Apabila di dalam mobil kita terdapat bayi dan pejalan kaki yang akan kita tabrak adalah orang dewasa maka kita harus mendahulukan anak kecil karena umur mereka lebih panjang[3]. Status lain yang dapat dijadikan pertimbangan adalah seperti pekerjaan, keamanan mereka saat ini, dan masih banyak lagi. Jadi saat mobil melihat di depannya ada kereta bayi dengan kondisi jumlah korban sama banyak, maka mobil akan lebih memilih untuk menabrakan dirinya sendiri daripada menabrak kereta bayi tersebut karena bayi masih memiliki umur panjang dan tingkat selamat sehabis tabrakan untuk bayi sangatlah kecil, tapi apabila di dalam mobil

ada anak kecil dan yang akan ditabrak hanyak orang dewasa maka mobil akan menabrak pejalan kaki itu demin menyelamatkan anak kecil yang berada di dalam mobil tersebut.

Pengaplikasian pada mobol otomatis sebenarnya masih banyak sekali karena dalam kehidupan nyata banyak sekali kondisi-kondisi yang akan terjadi dan meminta aksi yang paling tepat dari mobil otomatis kita. Selain dari sisi keamanan dan ketertiban, mobil otomatis ini diharapkan dapat memberikan kenyaman dalam berkendara. Kenyamanan yang dapat diberikan oleh mobil otomatis ini adalah berkaitan dengan pohon keputusan juga. Seperti saat kita mengemudi dengan normal kita harus mengatur kecepatn mobil kita maka dari itu suatu pohon keputusan diperlukan untuk mengatur kecepatan mobil, pohon keputusan dapat dilihat sebagai berikut:



Gambar 9. Pohon keputusan
(Sumber: koleksi pribadi)

Dapat dilihat kecepatan mobil disesuaikan dengan kondisi jalan yang sedang dilalui mobil otomatis. Apabila kita sebagai pengendara mobil otomatis apakah kita merasa nyaman apabila mobil sering mengerem? Tentu saja tidak, maka dari itu kecepatan mobil harus tepat. Kecepatan mobil dalam kondisi jalan yang berlubang harusnya perlahan agar tidak menimbulkan guncangan, lain halnya ketika kita sedang berkendara di jalan yang baik dan berada pada tengah kota. Kecepatan mobil dapat berada kecepetan sedang agar mobil tidak terlalu mengerem saat menghadapi kondisi di jalanan. Saat kita berada di jalan bebas hambatan kita dapat bergerak dengan cepat, tapi sebelum itu kita harus mengecek terlebih dahulu jumlah mobil yang berada disekitar kita. Apabila banyak maka kita tidak bisa bergerak dengan cepat karena dapat membahayakan keselamatan pengemudi mobil otomatis dan dapat menimbulkan ketidakyamanan karena mobil harus sering mengerem. Apabila jalan bebas hambatan kosong maka mobil otomatis dapat bergerak dengan cepat dan nyaman. Dengan menerapkan pohon keputusan pada gambar 9 untuk pengaturan kecepatan mobil otomatis maka mobil akan memberika rasa nyaman selama berkendara dan mobil juga dapat berkendara dengan aman karena berada pada batas kecepatan yang sesuai dengan keadaan kondisi sekitarnya.

V. KESIMPULAN

Pohon keputusan dapat diterapkan pada penyusunan algoritma mobil otomatis. Pohon keputusan dapat membantu *programmer* dalam memutuskan tindakan mobil yang paling mangkus dan sangkil dan setiap tindakan itu haruslah aman dan menaati peraturan yang ada. Dengan bantuan pohon keputusan segala tindakan dapat dianalisis dengan mudah dan menjadi refresi bagi *programmer* saat menghadapi suatu kondisi tertentu. Selain itu algoritma yang dibuat harus memberikan rasa nyaman pada pengemudi mobil otomatis dengan cara memperhatikan kondisi sekitar. Terakhir penulis menyimpulkan bahwa saat ini

VI. UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada Tuhan Yang Maha Esa untuk setiap berkatNya dalam hidup saya sehingga saya dapat menyelesaikan makalah saya ini yang berjudul aplikasi pohon keputusan dalam algoritma mobil otomatis. Penulis juga tidak lupa untuk berterima kasih pada orangtua saya yang selalu memberikan dukungan bagi saya untuk menyelesaikan makalah ini. Tak lupa penulis juga mengucapkan terima kasih kepada Dr. Ir. Rinaldi Munir, MT. , Dra. Harlili S., M.Sc. dan Dr. Judhi Santoso, M.Sc. selaku dosen IF2120 Matematika Diskrit. Semoga dengan adanya makalah ini pembaca mendapatkan manfaat yang berarti.

REFERENSI

[1] Rinaldi Munir, *Diktat Kuliah Matematika Diskrit*. Bandung:Departemen Teknik Informatika,2003,bab 8.
[2] Rinaldi Munir, *Diktat Kuliah Matematika Diskrit*. Bandung:Departemen Teknik Informatika,2003,bab 9.
[3] www.techologyreview.com/s/542626/why-self-driving-cars-must-be-programmed-to-kill/ ,diakses pada 2 Desember 2017
[4] www.businessinsider.sg/self-driving-cars-already-deciding-who-to-kil-2016-12//?r=US&IR=T, diakses pada 2 Desember 2017.

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, 3 Desember 2017

Ayrton Cyril-13516019