ANALISIS OPTIMASI PENJADWALAN JAGA DOKTER RESIDEN PENYAKIT DALAM PADA RUMAH SAKIT PENDIDIKAN

Erlanie Sufarnap[1], Sudarto[2]

STMIK Mikroskil
Jl. Thamrin No. 112, 124, 140 Medan 20212
airlanee@yahoo.com[1], Sudarto@mikroskil.ac.id[2]

Abstrak

Penjadwalan kegiatan jaga residen pada sebuah rumah sakit adalah hal yang rumit. Hampir sama rumitnya dengan penjadwalan belajar mengajar pada sebuah institusi pendidikan. Seorang dokter residen biasanya mendapat giliran jaga pada dua rumah sakit yang berbeda. Selain dilihat dari sisi dokter supervisornya juga harus dilihat dari sisi dokter residen, yaitu kemungkinan-kemungkinan dokter residen akan menangani beberapa kasus yang berbeda sesuai dengan pasien yang masuk rumah sakit saat itu. Hal ini bisa dilihat dari sudah berapa *stase* yang dilewati oleh seorang dokter residen selama pendidikan kespesialisannya sebab ada kemungkinan jumlah kasus pasien yang banyak dan berbeda-beda serta jumlah dokter residen tidak sebanding pada saat jaga, sehingga harus dipikirkan juga solusi agar dokter residen bekerja dengan tim jaganya pada hari , tanggal dan jam jaga yang sama. Selain itu, harus dipertimbangkan juga tingkat atau semester seorang dokter residen sehingga levelisasi tim jaga dapat dilaksanakan. Salah satu metode yang dapat digunakan untuk menyelesaikan permasalahan tersebut adalah dengan menggunakan pendekatan algoritma genetik. Algoritma genetik merupakan pendekatan komputasional untuk menyelesaikan masalah yang dimodelkan dengan proses biologi dari evolusi. Diharapkan dengan digunakannya algoritma genetik akan diperoleh optimasi penjadwalan yaitu kondisi dimana terjadi kombinasi terbaik untuk pasangan dokter residen dengan tim jaga beserta keahlian dan kemampuan lainnya secara keseluruhan, tidak ada permasalahan bentrokan jadwal jaga pada stase lain dan pada rumah sakit lainnya.

Kata kunci: *penjadwalan, optimasi, algoritma genetik.*

1. Pendahuluan

Penjadwalan jaga dokter residen pada sebuah rumah sakit merupakan pekerjaan yang tidak mudah. Terdapat berbagai aspek yang berkaitan dalam penjadwalan tersebut yang harus dilibatkan dalam pertimbangan di antaranya :

1. Terdapat di mana dokter residen yang bersangkutan tidak melewati minimal 3 stase untuk berada di level jaga 2nd atau level 3rd call.
2. Tidak boleh adanya jadwal jaga yang beririsan dengan jadwal jaga pada stase tertentu sehingga dokter residen dapat hadir pada hari dan jam yang ditentukan.
3. Distribusi jadwal jaga diharapkan merata keseluruh dokter residen penyakit dalam dari semester 0 paling awal hingga maksimal semester 7.
4. Pekerjaan penjadwalan jaga ini akan semakin berat jika melibatkan semakin banyak jumlah dokter residen per angkatannya.

Di samping aspek-aspek di atas, dalam penyusunan jadwal jaga ini pun terdapat sangat banyak kemungkinan yang selayaknya dicoba untuk menemukan penjadwalan yang terbaik. Karena itu dibutuhkan metode optimasi yang dapat diterapkan untuk mengerjakan penjadwalan jaga residen ini.

2. Kajian Pustaka

2.1 Algoritma Genetik

Algoritma genetik ditemukan di Universitas Michigan,Amerika Serikat oleh John Holland (1975) melalui sebuah penelitian dan dipopulerkan oleh salah satu muridnya, David Goldberg. [1] Algoritma genetik adalah algoritma yang berusaha menerapkan pemahaman mengenai evolusi alamiah pada tugas-tugas pemecahan-masalah (*problem solving*).
Pendekatan yang diambil oleh algoritma ini adalah dengan menggabungkan secara acak berbagai pilihan solusi terbaik di dalam suatu kumpulan untuk mendapatkan generasi solusi terbaik berikutnya yaitu pada suatu kondisi yang memaksimalkan kecocokannya atau lazim disebut *fitness*. Generasi ini akan merepresentasikan perbaikan-perbaikan pada populasi awalnya. Dengan melakukan proses ini secara berulang, algoritma ini diharapkan dapat mensimulasikan proses evolusioner. Pada akhirnya, akan didapatkan solusi-solusi yang paling tepat bagi permasalahan yang dihadapi. Untuk menggunakan algoritma genetik, solusi permasalahan direpresentasikan sebagai *khromosom*. Tiga aspek yang penting untuk penggunaan algoritma genetik:
1. Definisi *fitness function*
2. Definisi dan implementasi representasi genetik
3. Definisi dan implementasi operasi genetik

Jika ketiga aspek di atas telah didefinisikan, algoritma genetik generik akan bekerja dengan baik. Tentu saja, algoritma genetik bukanlah solusi terbaik untuk memecahkan segala masalah. Sebagai contoh, metode tradisional telah diatur untuk untuk mencari penyelesaian dari fungsi analitis *convex* yang "berperilaku baik" yang variabelnya sedikit. Pada kasus-kasus ini, metode berbasis kalkulus lebih unggul dari algoritma genetik karena metode ini dengan cepat menemukan solusi minimum ketika algoritma genetik masih menganalisa bobot dari populasi awal. Untuk problem – problem ini pengguna harus mengakui fakta dari pengalaman ini dan memakai metode tradisional yang lebih cepat tersebut. Akan tetapi, banyak persoalan realistis yang berada di luar golongan ini. Selain itu, untuk persoalan yang tidak terlalu rumit, banyak cara yang lebih cepat dari algoritma genetik. Jumlah besar dari populasi solusi, yang merupakan keunggulan dari algoritma genetik, juga harus mengakui kekurangannya dalam dalam kecepatan pada sekumpulan komputer yang dipasang secara seri –*fitness function* dari tiap solusi harus dievaluasi. Namun, bila tersedia komputer-komputer yang paralel, tiap prosesor dapat mengevaluasi fungsi yang terpisah pada saat yang bersamaan. Karena itulah, algoritma genetik sangat cocok untuk perhitungan yang paralel.

2.2 Teknik Penggunaan Algoritma Genetik

Algoritma genetik dimulai dengan sekumpulan set status yang dipilih secara random,yang disebut populasi. Algoritma ini mengkombinasikan dua populasi induk. Setiap status atau individual direpresentasikan sebagai sebuah string seperti sbb :

1. Fitness function

Setiap individual dievaluasi dengan *fitness function*. Sebuah *fitness function* mengembalikan nilai tertinggi untuk individual yang terbaik. Individu akan diurutkan berdasarkan nilai atau disebut dengan selection.

2. Crossover

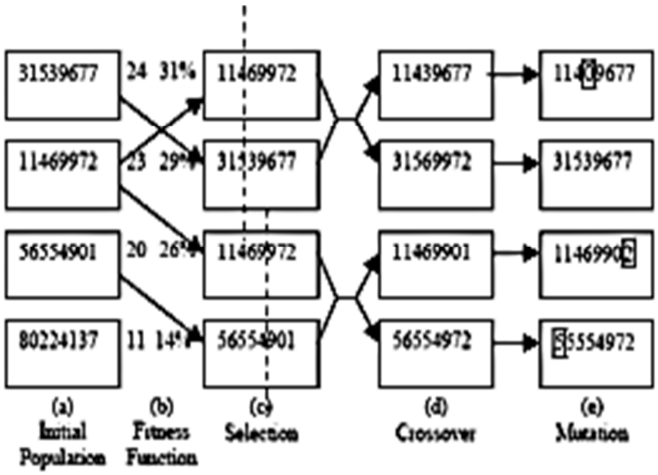
Untuk setiap pasang induk, sebuah titik *crossover* akan dipilih secara random dari posisi dalam string. Pada gambar titik *crossover* terletak pada indeks ketiga dalam pasangan pertama dan setelah indeks kelima pada pasangan kedua.

3. Mutasi

Pada mutasi, tiap lokasi menjadi sasaran mutasi acak, dengan probabilitas independen yang kecil. Sebuah digit dimutasikan pada anak pertama, ketiga, dan keempat. Algoritma genetik mengkombinasikan suatu kecenderungan menaik dengan pengeksplorasian acak di antara thread pencarian paralel. Keuntungan utamanya, bila ada datang dari operasi *crossover*. Namun, secara matematis dapat tunjukkan bahwa bila posisi dari kode genetik di permutasikan di awal dengan urutan acak, *crossover* tidak memberikan keunggulan. Secara intuisi, keuntungannya didapat dari kemampuan *crossover* untuk menggabungkan blok-blok huruf berukuran besar yang telah berevolusi secara independen untuk melakukan fungsi yang bermanfaat sehingga dapat menaikkan tingkat granularity di mana pencarian dilakukan.

4. Schema

Teori dari algoritma genetik menjelaskan cara kerjanya menggunakan ide dari suatu *schema*, suatu substring di mana beberapa posisi tidak disebutkan. Dapat ditunjukkan bahwa, bila *fitness* rata-rata dari *schema* berada di bawah mean maka jumlah instansiasi dari *schema* di dalam populasi akan bertambah seiring bertambaahnya waktu. Jelas sekali bahwa efek ini tidak akan signifikan bila bit-bit yang bersebelahan sama sekali tidak berhubungan satu sama sekali, karena akan ada beberapa blok kontinu yang memberikan keuntungan yang konsisten. Algoritma genetik paling efektif dipakai bila *schema-schema* berkorespondensi menjadi komponen berati dari sebuah solusi. Sebagai contoh, bila string adalah representasi dari sebuah antena, maka *schema* merepresentasikan komponen-komponen dari antena, seperti reflector dan deflector. Sebuah komponen yang baik cenderung akan berkerja baik pada rancangan yang berbeda. Ini menunjukkan bahwa penggunaan algoritma genetik yang benar memerlukan rekayasa yang baik pada representasinya.



Gambar 1. Contoh Penggunaan Algoritma Genetik

2.3 Pseudocode Algoritma Genetika

```
function
GenetikAlgorithm(population,Fitness-FN)
-> an individual
{input berupa population, sebuah
kumpulan individual dan Fitness-FN,
sebuah fungsi yang mengukur fitness
suatu individual}
deklarasi
i,x,y : integer

algoritma
repeat
   new_population<-empty set
   for i=1 to size(population) do
      x<-RandomSelection(population,
      Fitness-FN)
      y<-RandomSelection(population,
      Fitness-FN)
      child<-Reproduce(x,y)
      if(smallRandomProbability) then
      child<-mutate(child)
      add child to new_population
   population<-new_population
   until some individual is fit enough
or the time has elapsed
return the best individual in
population(based on Fitness-FN)

function Reproduce(x,y : parent
individuals)->individual
deklarasi
algoritma
n<-length(x)
c<-random number from 1 to n
return Append(substring(x,1,c),
substring(y,c +1,n))
```

Gambar 2. Pseudocode Algoritma Genetika

3. Hasil dan Pembahasan

3.1. Flowchart Program

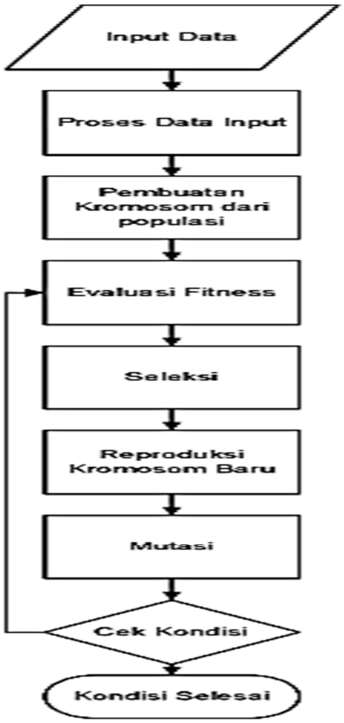
Flowchart program dapat dilihat pada Gambar 2. Flowchart ini terdiri dari delapan sub program yaitu input data, proses data input, pembuatan kromosom dan populasi, evaluasi fitness, seleksi, reproduksi kromosom baru, mutasi, serta kondisi selesai. Input, output dan proses dari setiap subprogram akan dibahas berikut ini.

a. Input Data

Terdapat enam masukan kelompok data yang perlu diberikan, yaitu :

1. Tabel Levelisasi Jaga
2. Tabel Dokter residen
3. Tabel Posisi Jaga
4. Tabel Ruang
5. Bobot Fitness
6. Kondisi Selesai Tabel Jaga berisikan daftar jaga seluruh dokter residen akan dilaksanakan pada bulan berikutnya pada rumah sakit pendidikan. Data yang perlu disertakan untuk setiap Levelisasi yang ada adalah semester residen, jumlah stase yang sudah dilalui, dokter residen serta ruangan.

Tabel Dokter Residen, Tabel Posisi Jaga, serta Tabel Ruang adalah tabel waktu yang menginformasikan waktu-waktu dari dokter residen, posisi dan ruangan yang dapat digunakan untuk jaga yang bersangkutan.



Gambar 3. Flowchart Program

2. Proses Data Input

Agar dapat diproses dalam algoritma ini Tabel Levelisasi Jaga, Tabel Dokter Residen, Tabel Posisi Jaga dan Tabel Ruang harus digabungkan terlebih dahulu menjadi Tabel Prioritas Levelisasi Jaga. Untuk menjadwalkan suatu Levelisasi Jaga, perlu mempertimbangkan jadwal waktu dokter residen, posisi jaga dan ruangan yang ada. Maka setiap levelisasi jaga akan memiliki banyaknya pilihan penjadwalan yang berbeda. Bisa jadi ada levelisasi jaga yang memiliki tiga pilihan hari dalam satu minggu dan bisa jadi ada levelisasi jaga yang hanya memiliki satu pilihan hari saja. Tabel Prioritas levelisasi Jaga berisikan banyaknya tingkat pilihan penjadwalan dari setiap level jaga yang ada serta telah diurutkan dari level tertinggi yang paling sedikit pilihan penjadwalannya hingga levelisasi jaga yang terbanyak pilihan penjadwalannya. Dari proses ini diharapkan tidak ada levelisasi jaga yang tidak dapat teralokasikan penjadwalannya dikarenakan pada jadwal-jadwal yang memungkinkan bagi level jaga tersebut telah digunakan oleh level jaga lainnya.

3. Pembuatan Kromosom dan Populasi

Berdasarkan urutan dari Tabel Prioritas Levelisasi Jaga, setiap level jaga akan dijadwalkan ke dalam Tabel Jadwal Level Jaga secara acak.

Agar diketahui apakah pada waktu tersebut dokter residen, maupun ruangan dapat dibagi untuk memposisikan jaga, maka Tabel Dokter residen, Tabel Ruang, dan Tabel Posisi Jaga untuk setiap Level jaga serta Tabel Levelisasi Jaga harus dipetakan terlebih dahulu dalam Tabel Jadwal Level Jaga Sebelum menjadwalkan suatu level jaga pada Tabel Jadwal Level Jaga, algoritma akan mengecek terlebih dahulu kepada Tabel Jadwal Level Jaga Bayangan untuk mengetahui apakah pada waktu tersebut dapat digunakan untuk jaga atau tidak. Jika tidak maka algoritma didesain untuk mencari alokasi waktu lainnya.

4. Evaluasi Fitness

Faktor-faktor yang mempengaruhi evaluasi fitness terhadap alternatif solusi adalah sebagai berikut :

1. *Pemecahan levelisasi jaga;* Terhadap level jaga dengan minimal 3 stase terlewati, program dapat memecah level jaga tersebut menjadi 2 atau 3 kelompok jaga jika waktu penjadwalan yang ada tidak memungkinkan untuk dilaksanakannya jaga oleh dokter residen tersebut dalam satu waktu. Hal ini dibuat dengan tujuan memperluas kemungkinan alternatif penjadwalan yang ada terutama pada level jaga yang hanya memiliki sedikit alternatif penjadwalan. Tetapi pemecahan levelisasi jaga ini akan memperkecil nilai *fitness*, sehingga kelak program akan cenderung menyeleksi solusi penjadwalan yang memiliki pemecahan levelisasi jaga yang terlalu banyak.
2. *Pemadatan di suatu waktu;* Untuk meningkatkan produktivitas pemakaian ruangan, maka dikehendaki agar ruangan dapat segera digunakan semenjak pagi hari. Jika program menawarkan solusi penjadwalan di mana terdapat waktu pagi yang tidak digunakan, maka hal ini akan memperkecil nilai fitness solusi.
3. *Frekuensi jaga dokter residen;* Diinginkan agar tugas jaga dapat terdistribusi merata di tiap hari kerjanya dengan tujuan agar performansi dokter residen sewaktu jaga dapat tetap dijaga optimal. Nilai fitness solusi akan berkurang jika dalam solusi tersebut terdapat dokter residen yang memiliki tingkat jaga terlalu tinggi dalam satu minggunya.
4. *Frekuensi dokter Supervisor;* Seperti halnya pada Dokter residen, untuk menjaga performansi jaga maka diharapkan tidak ada jadwal jaga yang terlalu berdekatan dalam satu minggunya. Jika solusi menawarkan jadwal level jaga yang terlalu padat dalam satu minggu , maka nilai fitness solusi yang berkurang. Posisi Jaga yang memiliki level kurang dari 3 semester pada satu hari jaga didefinisikan sebagai posisi yang memiliki frekuensi jaga yang tinggi (istilah garsa).
5. *Kedekatan antar level jaga;* Idealnya dokter residen memiliki waktu istirahat antar dua hari jaga yang ada dalam satu minggu sehingga kelelahan dokter residen dalam melakukan jaga pertama tidak memperngaruhi proses jaga selanjutnya. Didefinisikan bahwa dua level jaga yang berjarak kurang dari dua semester untuk satu posisi jaganya digolongkan sebagai levelisasi jaga yang berdekatan dan dapat memperkecil nilai fitness dari solusi yang ditawarkan program.

Rumus fitness yang digunakan adalah sebagai berikut:

$$Fitness = \frac{1}{B1xF1 + B2xF2 + B3xF3 + B4xF4 + B5xF5 + B6xF6}$$

dengan :

F1 = Banyaknya Levelisasi Jaga yang dipecah
F2 = Banyaknya waktu jaga yang kosong
F3 = Banyaknya frekuensi jaga yang tinggi dari seorang dokter residen
F4 = Banyaknya frekuensi posisi jaga yang tinggi dari satu posisi jaga
F5 = Banyaknya level jaga yang berdekatan
F6 = Banyaknya level jaga yang berjauhan.
B1 = Bobot pemecahan levelisasi jaga
B2 = Bobot waktu jaga yang kosong
B3 = Bobot frekuensi jaga dokter residen
B4 = Bobot frekuensi level posisi
B5 = Bobot level jaga yang berdekatan

B6 = Bobot level jaga yang berjauhan

Setiap faktor yang mempengaruhi nilai fitness di atas memiliki tingkat pengaruh yang berbeda terhadap nilai fitness. Tingkat pengaruh ini disebut sebagai bobot. Jika suatu faktor pengaruh memiliki harga bobot yang tinggi maka setiap kali faktor tersebut terjadi dalam suatu solusi maka akan sangat mengurangi nilai fitness dari solusi tersebut. Dan sebaliknya jika suatu faktor memiliki harga bobot yang kecil, maka tidak akan terlalu mengurangi nilai fitness dari solusi meskipun faktor tersebut banyak terjadi dalam solusi yang ditawarkan. Dari rumus nilai fitness di atas dapat terlihat bahwa yang mempengaruhi besar nilai fitness adalah harga FN karena harga BN akan tetap selama proses. Jika harga FN semakin besar maka nilai Fitness akan semakin kecil. Karena diinginkan solusi yang memiliki nilai Fitness yang besar, maka program ini diharapkan tidak terlalu banyak memunculkan faktorfaktor pengaruh ini dalam solusi yang ditawarkan.

5. Seleksi

Untuk mendapatkan solusi yang terbaik, maka program harus menyeleksi solusi yang memiliki nilai fitness yang tergolong rendah. Seleksi menggunakan metode good fitness yaitu setengah dari jumlah populasi yang memiliki harga fitness yang terendah akan dihilangkan sehingga akan hanya selalu tersisa sekelompok solusi yang terbaik yang pernah diperoleh oleh program. Solusi yang tersisa hasil seleksi ini dikenal dengan nama populasi induk. Agar jumlah populasi tetap, maka perlu dibangkitkan solusi baru sebanyak setengah dari jumlah populasi yang ada. Dalam program ini, cara yang digunakan untuk membangkitkan solusi baru menggunakan dua cara yaitu reproduksi kromosom baru dan cara mutasi dari solusi induk. Tujuan pembangkitan solusi baru ini untuk menemukan alternatif solusi yang lebih baik dari solusi-solusi yang sudah diperoleh.

6. Reproduksi Kromosom Baru

Setengah dari jumlah populasi baru akan dibangkitkan dengan cara reproduksi kromosom baru. Yaitu penyusunan alternatif solusi penjadwalan secara acak kembali untuk setiap levelisasi jadwal. Proses ini sama dengan langkah ketiga yang telah dibahas. Dengan proses ini maka akan dihasilkan sekelompok populasi baru yang benar-benar berbeda dengan populasi induknya.

7. Mutasi

Adapun setengah populasi baru lainnya akan dibangkitkan dengan cara mutasi. Yaitu setengah dari populasi induk akan dipilih untuk diduplikasi. Pemilihan dapat dilakukan dengan metode good fitness, random maupun roulette whell. Pada hasil duplikasi ini akan dilakukan sedikit percobaan terhadap posisi penjadwalan beberapa level jaga. Proses mutasi ini adalah suatu proses eksploitasi terhadap kemungkinan-kemungkinan modifikasi pada jadwal yang telah ada. Perubahan posisi beberapa level jaga ini (mutasi) dapat membuat solusi duplikasi ini menjadi memiliki nilai fitness yang lebih rendah maupun lebih tinggi daripada solusi induknya. Jika ternyata diperoleh solusi yang memiliki fitness yang lebih tinggi maka hal itulah yang diharapkan. Tetapi jika diperoleh solusi dengan nilai fitness yang lebih rendah maka bisa jadi pada iterasi berikutnya diperoleh solusi hasil mutasi yang lebih baik nilai fitnessnya daripada solusi induknya. Maksudnya adalah tidak menjadi masalah jika solusi hasil mutasi ini mengalami penurunan nilai fitness daripada solusi induknya. Kita hanya perlu untuk mengacuhkannya saja dan tetap memberikan perhatian pada solusi terbaik yang telah

dicapai. Pada proses mutasi ini, program akan memilih empat levelisasi jaga secara acak untuk dikeluarkan dari solusi hasil penggandaan tersebut untuk dijadwalkan kembali pada posisi yang berbeda. Sebelum level jaga yang dikeluarkan tersebut dijadwalkan kembali, maka program harus memproses data input terlebih dahulu dari keempat levelisasi jaga tersebut. Proses ini sama dengan proses pada langkah kedua dari algoritma pemrograman ini. Hal ini dilakukan untuk menghindari level jaga yang memiliki pilihan penjadwalan yang sedikit menjadi tidak bisa diposisikan kembali dalam penjadwalan karena pilihan-pilihan waktu penjadwalannya telah terisi oleh mata kuliah yang dikeluarkan lainnya.

8. Kondisi Selesai

Terdapat tiga kondisi selesai yang dapat menghentikan proses algoritma pemrograman ini, yaitu:

1. Jika setelah beberapa generasi berturut-turut nilai fitness terbaik dari populasi tidak mengalami perubahan kembali
2. Jika jumlah generasi atau iterasi maksimum telah tercapai.
3. Jika nilai fitness terbaik minimal telah tercapai.

Jika salah satu kondisi di atas telah diperoleh maka iterasi akan dihentikan.dan jika salah satu kondisi selesai ini belum tercapai maka program akan mengulang kembali proses ini / iterasi dari langkah keempat yaitu evaluasi fitness terhadap populasi baru tadi.

4. Kesimpulan

Dengan bantuan Algoritma Genetik penyusunan penjadwalan jaga dokter residen dapat dioptimalkan. Program dapat mencari solusi penjadwalan pada waktu yang dapat digunakan baik oleh dokter residen, posisi jaga maupun ruangan yang terlibat dalam suatu levelisasi jaga. Di samping itu, program dapat meminimalkan tingginya frekuensi jaga seorang dokter residen, frekuensi posisi jaga dan faktor pengaruh lainnya. Proses penjadwalan levelisasi jaga menggunakan Algoritma Genetik ini dapat diterapkan pada kasus-kasus penjadwalan dengan multi angkatan, multti posisi dan multi ruangan. Dengan menggunakan metode best fitness, maka Algoritma Genetik akan selalu menunjukkan kenaikan fitness atau dengan kata lain generasi selanjutnya lebih baik atau minimal sama dengan generasi sebelumnya.

Referensi

[1] Randy L. Haupt . 2004. “Practical Genetic Algorithms”. A John Wiley & Sons, Inc.
[2] Muhammad Aria. 2006. "Aplikasi Algoritma Genetik Untuk Optimasi Penjadwalan Mata Kuliah". Universitas Komputer Indonesia.
[3] Wilhelm Erben 2005 "A Hybrid Grouping Genetic Algorithm for Examination Timetabling". University of Applied Sciences.
[4] http://www.esat.kuleuven.ac.be/sista/mai/2006- 2007/cs/gafortimetabling.pdf
[5] http://ilmukomputer.com/2007/03/29/algoritma-genetika-dan-contoh-aplikasinya/.
[6] http://lancet.mit.edu/~mbwall/presentations/IntroToGAs/