ISSN : 2407 - 3911

IMPLEMENTASI ALGORITMA A* DALAM PENJADWALAN MATAKULIAH

(STUDI KASUS PENJADWALAN SEMESTER GANJIL 2017-2018 INFORMATIKA ITENAS)

Youllia Indrawaty Nurhasanah[1], Sofia Umaroh[2], Asep Chaesar Trianto[2]
[1,2,3]Program Studi Informatika, Fakultas Teknologi Industri, Institut Teknologi Nasional, Bandung, Indonesia Jl. PKH. Mustopha No.23, Neglasari, Cibeunying Kaler, Kota Bandung, Jawa Barat 40124
Email: youllia@itenas.ac.id[1], sofia.umaroh@gmail.com[2], chaesartrianto@gmail.com[3]

Abstrak

Penjadwalan matakuliah merupakan proses atau kegiatan rutin yang dilakukan setiap pergantian semester atau tahun ajaran. Penjadwalan matakuliah secara manual atau tradisional merupakan suatu kegiatan yang memakan waktu lama dan memerlukan ketelitian yang tinggi, karena kesalahan kecil dapat mengakibatkan ketidaksesuain jadwal matakuliah atau dapat diartikan sebagai bentrok antar kelas matakuliah. Dalam melakukan penjadwalan matakuliah, dibutuhkan sebuah kemampuan untuk mencari slot waktu terbaik untuk sebuah kelas matakuliah. Kemampuan tersebut diisi oleh algoritma A*. Algoritma A* sendiri merupakan algoritma komputasi yang digunakan dalam pencarian dan perencanaan jalur yang bisa dilewati secara efisien disekitar titik-titik yang disebut node, dengan menggunakan nilai *heuristic* antar koordinat dan nilai sebenarnya dari node awal ke node tujuan. Berawal dari teori tersebut, algoritma A* diadopsi menjadi algoritma pencari nilai slot waktu dalam penjadwalan matakuliah. Algoritma A* diterapkan untuk mencari nilai kualitas slot waktu dengan cara menjumlahkan nilai jarak antar node dengan nilai beberapa faktor seperti, kelas, dosen dan tingkatan. Tujuan dari penelitian ini adalah membangun sebuah sistem perangkat lunak yang mampu menghasilkan jadwal matakuliah yang tidak memiliki bentrok antar kelas matakuliah. Penelitian ini berhasil mendapatkan sebuah jadwal matakuliah tanpa bentrok antar kelas matakuliah pada jadwal matakuliah semester ganjil tahun ajaran 2017 - 2018 di jurusan informatika Institut Teknologi Nasional.

Kata kunci : *A*, Penjadwalan, Matakuliah*

Abstract

*Arranging college subjects is a routine process or activity carried out every turn of the semester or school year. Scheduling subjects manually or traditionally is an activity that takes a long time and requires high accuracy, because small errors can lead to mismatch in the subject schedule or can be interpreted as clashes between classes. In scheduling college subjects, it takes an ability to find the best time slot for a course class. This ability is filled by A * algorithm. The A * algorithm itself is a computation algorithm used in search and planning paths that can be passed efficiently around the points called nodes, by using heuristic values between coordinates and the actual value from the initial node to the destination node. Starting from this theory, A * algorithm was adopted as a time slot value search algorithm in course scheduling. The A * algorithm is applied to find the quality value of the time slot by adding up the value of the distance between nodes with the value of several factors such as class, lecturer and level. The purpose of this study is to build a software system that is able to produce course schedules that do not have clashes between subject classes. This study succeeded in getting a course schedule without clashing between classes on the odd semester semester of 2017 - 2018 in the Informatics Department of the National Institute of Technology.*

Keywords : *A*, Scheduling, College Subject*

I. PENDAHULUAN

I.1 Latar Belakang

Dalam kegiatan penyusunan jadwal matakuliah yang dilakukan secara manual, dibutuhkan ketelitian dan waktu yang tidak sebentar, dan seringkali terdapat berbagai masalah seperti banyaknya jadwal matakuliah yang harus disusun, jumlah ruangan yang terbatas, dan kesanggupan dosen untuk mengajar pada jadwal tertentu, sehingga dapat menyebabkan bentroknya jadwal mata kuliah ataupun adanya ketidaksesuaian jadwal mata kuliah. Penyelesaian masalah penjadwalan mata kuliah dalam jumlah yang sangat besar hingga saat ini masih menjadi permasalahan yang rumit untuk diselesaikan secara manual (Puspaningrum, Djunaidy, & Vinarti, 2013)

Dalam melakukan penjadwalan mata kuliah, dibutuhkan sebuah kemampuan untuk mengalokasikan semua sumber daya kedalam slot waktu. Kemampuan tersebut diisi dengan pengetahuan terhadap batasan-batasan yang ada di *constraint* sehingga sistem dapat mengatasi masalah yang dihadapi ketika penjadwalan mata kuliah dilakukan. Setelah sumber daya masuk ke dalam slot waktu yang memiliki nilai masing – masing, selanjutnya akan dimasukkan kedalam sebuah *tree* untuk dicari slot waktu terbaik (biaya terendah) menggunakan algoritma A*. Algoritma A* akan mencari slot waktu yang terbaik (biaya terendah) untuk setiap mata kuliah yang akan dijadwalkan dengan menelusuri simpul – simpul pada *tree*. Setelah mendapatkan jadwal mata kuliah yang tidak memiliki bentrok, keluaran dari sistem akan digunakan sebagai data masukan ke *google calendar*, yang selanjutnya akan berperan sebagai penyedia informasi jadwal mata kuliah.

Pelaksanaan penelitian ini diharapkan dapat membantu efisiensi waktu dalam pembuatan jadwal mata kuliah sekaligus penjadwalan mata kuliah yang mampu mencakup seluruh mata kuliah untuk sebuah program studi.

I.2 Rumusan Masalah

Penjadwalan mata kuliah merupakan kegiatan administratif yang penting dan memakan waktu yang cukup lama. (Simamora, Purwanto. Tulus. Batubara R, 2007). Penelitian ini diharapkan mampu membuat perangkat lunak yang mampu menjadikan penjadwalan matakuliah lebih efisien. Berdasarkan dari masalah tersebut terdapat rumusan masalah yaitu :

a) Bagaimana cara penerapan algoritma A* untuk mendapatkan hasil penjadwalan perkuliahan yang tidak memiliki bentrok, kapasitas kelas dan jumlah mahasiswa dapat menyesuaikan.
b) Bagaimana cara penerapan *constraint* untuk data dosen, mata kuliah, ruangan, dan jam kuliah.

I.3 Batasan Masalah

Batasan masalah dalam penelitian ini adalah: (1) Sistem mengelola penjadwalan kuliah untuk Program Studi Teknik Informatika dan Program Studi Teknik Informatika Institut Teknologi Nasional. (2) Penjadwalan kuliah dilakukan untuk semester regular (gasal dan genap).

LANDASAN TEORI

II.1 Algoritma A*

Algoritma A* adalah algoritma komputer yang digunakan secara luas dalam *graph traversal* dan penemuan jalur serta proses perencanaan jalur yang bisa dilewati secara efisien disekitar titik-titik yang disebut node (Russell & Norvig, 2013). Algoritma ini memeriksa node dengan menggabungkan g(n), yaitu cost yang dibutuhkan untuk mencapai sebuah node dan h(n), yaitu cost yang didapat dari node awal ke node n. Sehingga didapatkan rumus dasar dari algoritma A* ini adalah:

$$f(n) = g(n) + h(n) \quad \ldots[1]$$

dimana: h(n) = nilai heuristik antar koordinat
g(n) = nilai sebenarnya dari node awal ke node n

II.2 Penjadwalan

Berdasarkan Kamus Besar Bahasa Indonesia, jadwal merupakan pembagian waktu berdasarkan rencana pengaturan urutan kerja. Jadwal juga didefinisikan sebagai daftar atau tabel kegiatan atau rencana kegiatan dengan pembagian waktu pelaksanaan yang terperinci. Sedangkan penjadwalan merupakan proses, cara, perbuatan menjadwalkan atau memasukkan dalam jadwal (Kartiko, Rifki. Indrawaty N, Youllia. & Utami, 2018). Penjadwalan merupakan suatu kegiatan alokasi sumber daya dengan memiliki kendala (batasan) yang diberikan kepada suatu objek seperti di ruang-waktu, sedemikian rupa untuk memenuhi sedekat mungkin set tujuan yang diinginkan. Definisi yang lebih umum adalah menugaskan satu kumpulan peristiwa (kuliah, kendaraan, acara-acara publik, dll) dengan kumpulan terbatas sumber dari waktu ke waktu sedemikian rupa untuk memenuhi kendala (batasan/constraint) yang

telah ditetapkan, kendala ini dapat dikategorikan sebagai hard constraint dan soft constraint , di mana hard constraint memiliki prioritas yang lebih tinggi dari pada soft constraint. Terdapat dua batasan dalam penyusunan penjadwalan kuliah yang dikemukakan oleh (Kartiko, Rifki. Indrawaty N, Youllia. & Utami, 2018) yaitu : hard constraint (harus terpenuhi) dan soft constraint (diupayakan untuk terpenuhi). Hard constraints merupakan batas-batas yang harus diterapkan pada penjadwalan mata kuliah dan harus dipenuhi. Sebuah solusi hanya dapat dikatakan sahih dan valid apabila dalam solusi tersebut sama sekali tidak ada hard constraint yang terlanggar. Hard constraints yang ditetapkan dalam pembangunan sistem penjadwalan matakuliah berbasis algoritma A* adalah sebagai berikut :

Satu (1) Dosen tidak dapat berada pada dua (2) kelas atau lebih pada waktu yang bersamaan,

- Satu (1) Ruangan tidak bisa digunakan untuk dua (2) atau lebih matakuliah pada waktu yang bersamaan,
- Satu (1) Kelas tidak bisa memiliki dua (2) jadwal matakuliah atau lebih pada waktu yang bersamaan,
- Matakuliah teori harus berada pada ruang kelas regular, dan matakuliah praktikum harus berada pada ruang kelas laboratorium,
- Hari Jum'at Pukul 12.00 – 13.00 tidak bisa dipakai untuk kegiatan belajar, Hari kuliah adalah hari Senin – Sabtu, dan jam kuliah dimulai dari jam 07.00 – 17.00 WIB,
- Satu (1) SKS matakuliah berdurasi selama 1 Jam (50 menit perkuliahan, 10 menit istirahat).
- Jumlah mahasiswa yang berada di satu ruangan tidak dapat melebihi jumlah kapasitas dari ruangan tersebut.

Berbeda dengan hard constraint, soft constraint merupakan kendala yang tidak selalu dapat terpenuhi dalam proses pembentukan jadwal, akan tetapi meskipun tidak harus terpenuhi, jadwal yang dihasilkan harus semaksimal mungkin berusaha memenuhi ketentuan soft constraint. Contoh soft constraints dalam penjadwalan mata kuliah antara lain:

- Dosen dapat meminta waktu jadwal mengajar tertentu yang diinginkan,

Berdasarkan penjelasan tersebut, terdapat sejumlah batasan-batasan tertentu, dimana pada batasan-batasan tersebut terdapat batasan yang tidak boleh dilanggar atau harus terpenuhin. Batasan tersebut merupakan ukuran kualitas dari penjadwalan matakuliah, sehingga suatu jadwal matakuliah yang optimal dapat terbentuk

ANALISIS DAN PERANCANGAN

III.1 Analisa Kebutuhan Sistem

Dalam pembangunan sistem penjadwalan matakuliah dengan algoritma A*, dibutuhkan spesifikasi software dan hardware tertentu agar proses pembangunan dapat berjalan dengan lancar, dan sistem juga memiliki kebutuhan yang harus dimiliki, berikut adalah identifikasi kebutuhan tersebut.

III.1.1 Identifikasi Kebutuhan Sistem

Pada tahap ini dilakukan identifikasi untuk mengetahui kebutuhan dan fungsi sistem yang diharapkan. Berikut ini merupakan kebutuhan utama yang diharapkan pada sistem penjadwalan matakuliah dengan algoritma A*, yaitu diantaranya adalah sebagai berikut :

1. Sistem mampu menghasilkan jadwal matakuliah tanpa adanya bentrok sama sekali didalam jadwal yang dihasilkan, meliputi tidak adanya bentrok dosen, bentrok ruangan dan bentrok waktu.
2. Sistem mampu menyesuaikan jumlah mahasiswa dengan kapasitas kelas.
3. Sistem mampu mengecualikan dosen pada jam dan hari tertentu.

III.1.2 Identifikasi Kebutuhan Software

Dalam membangun sistem penjadwalan matakuliah dengan algoritma A*, dibutuhkan berbagai macam perangkat lunak, perangkat lunak tersebut diantaranya adalah sebagai berikut:

1. Windows 10 64-bit, dibutuhkan sebagai operating system yang digunakan untuk membuat dan menjalankan aplikasi.
2. Google Calendar API, sebagai Framework PHP yang digunakan dalam perancangan sistem yang berbasis website.
3. XAMPP, sebagai server lokal untuk proses pengetesan sistem dan juga untuk menyediakan database yang berbasis MySQL.
4. HeidiSQL, sebagai aplikasi untuk mengakses dan membuat database.

5. Google Chrome, sebagai alat untuk membuka sistem yang dibangun.
6. Sublime Text, sebagai alat untuk penulisan dan pengkodean sistem yang dibangun.

III.1.3 Identifikasi Kebutuhan Hardware

Dalam membangun sistem penjadwalan matakuliah dengan algoritma A*, dibutuhkan perangkat keras, perangkat keras terpenting yang dibutuhkan dalam pembangunan sistem adalah laptop, dengan spesifikasi sebagai berikut :

- Sistem Operasi Windows 10 Pro,
- Intel® Core™ i5 6200U Processor,
- RAM 8GB,
- Harddisk Internal 500GB

III.2 Perancangan

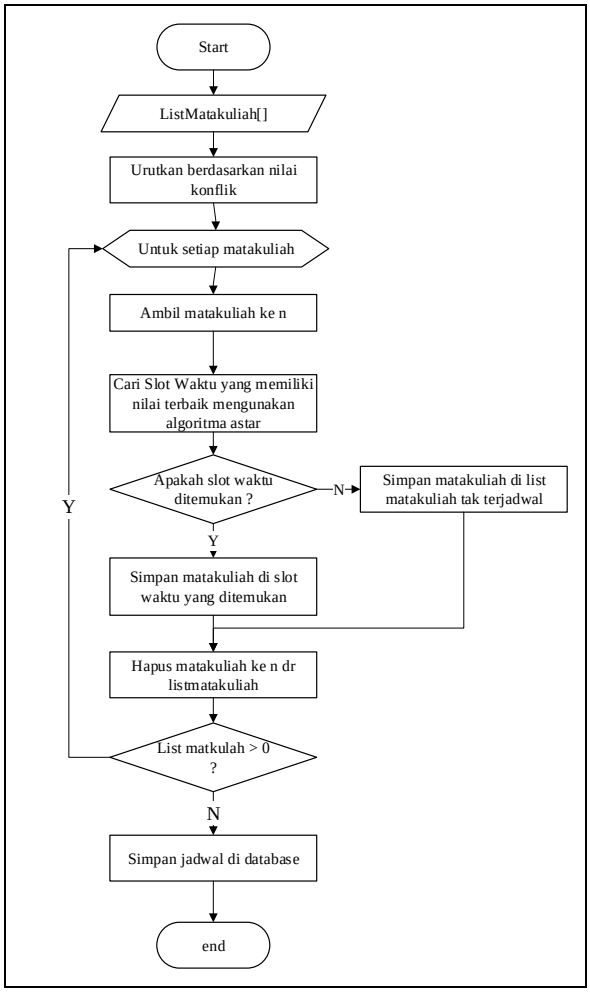
Perancangan dijelaskan menggunakan *flowchart* yang dapat dilihat pada *gambar 1*.

Seperti dapat dilihat pada gambar 1, tahap pertama dari sistem adalah memasukan list matakuliah dan mengurutkannya berdasarkan besar nilai conflict yang dapat dihitung dengan cara,

Nilai konflik = *N1+N2+N3* ……*(1)*

Keterangan:
N1: *constraint* ruangan : 0.1 / banyak *constraint* ruangan.
N : *constraint* waktu dosen dapat mengajar :0.45/cd
N3 : type kelas : jika kelas matakuliah dapat digabung nilainya 0.45 jika tidak bernilai 0
Cd : banyak *constraint* waktu/sks matakuliah

Setelah mendapatkan urutan matakuliah, sistem akan mencari slot waktu yang terbaik untuk setiap matakuliah, dimulai dengan matakuliah yang memiliki nilai konflik tertinggi. Hal ini dilakukan untuk mengurangi kemungkinan bentrok antar kelas matakuliah.Proses pencarian slot waktu dapat dilihat pada gambar 2. Matakuliah yang tidak mendapatkan slot waktu akan disimpan di list matakuliah tak terjadwal. Setelah semua matakuliah telah dicari slot waktunya. Semua matakuliah yang mendapatkan slot waktunya akan disimpan di *database* sistem.

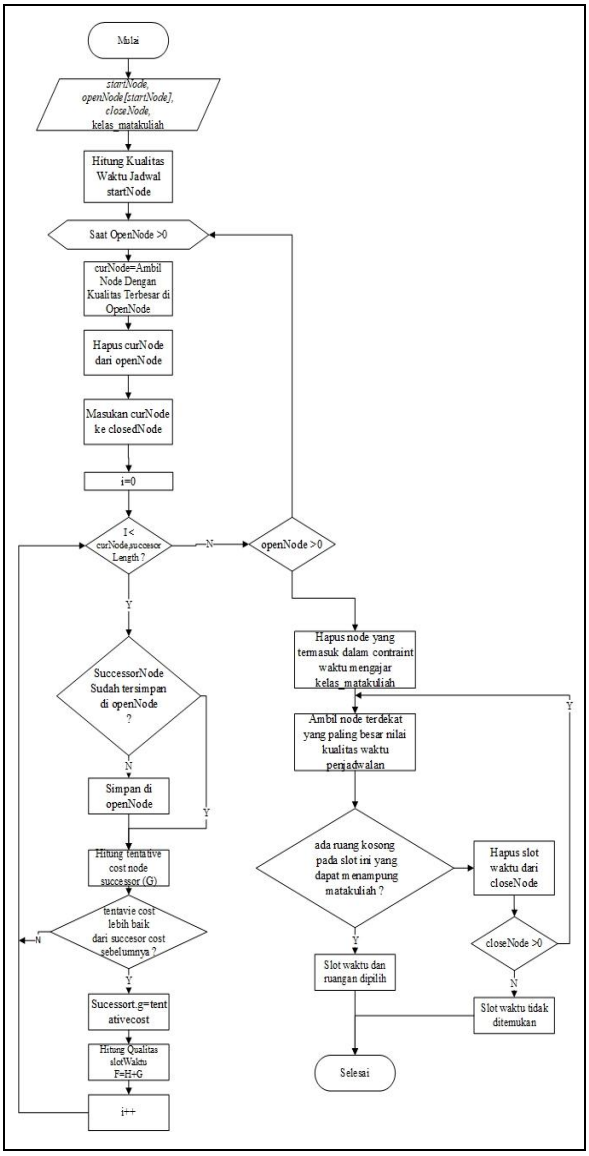


Gambar 1. Flowchart sistem

Agar lebih mudah dimengerti, penjelasan *flowchart* slot waktu pada gambar 2 akan dijelaskan menggunakan contoh studi kasus.

Contoh Studi Kasus Perhitungan A*:
Studi Kasus Penjadwalan
Diketahui :
Matakuliah :

- Matakuliah 1 Tingkat 1 = M1 (tipe kelas : Mandiri, Sks : 4, Jumlah mahasiswa : 40)
- Matakuliah 2 Tingkat 1 = M2 (tipe kelas : Mandiri, Sks : 2, Jumlah mahasiswa : 30)



Gambar 2. Flowchart slot waktu

- Matakuliah 3 Tingkat 1 = M3 (tipe kelas : Mandiri, Sks : 4, Jumlah mahasiswa : 30)
  Kelas : A
  Ruangan :
  - R1, Kapasitas 60
  - R2, Kapasitas 30

  Dosen :
  - D1, Constraint waktu : -.
  - D2, Constraint waktu : -.

Rumus perhitungan yang digunakan adalah :

Nilai Konflik (NC):

NC = N1+N2+N3

- A*: F = G+H

Dimana : G = Jarak node slot waktu dan H = Nilai kualitas slot waktu.

Kualitas Slot Waktu:

Kualitas slow waktu = P1+P2+P3…….(3)

Dimana :

P1 = Check apakah pada hari di slot ini kelas tersebut telah melaksanakan kuliah atau belum, jika telah melaksakan kuliah beri nilai 0.40

P2 = Check apakah pada hari di slot ini dosen yang mengajar telah mengajar suatu kuliah atau belum, jika telah melaksakan kuliah beri nilai 0.35

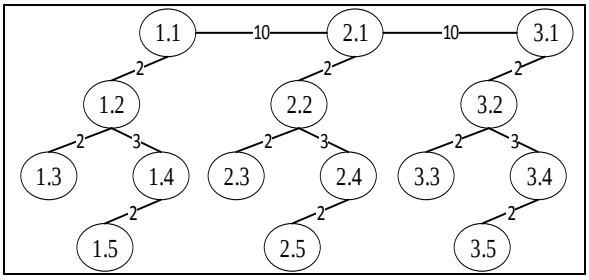
P3 = lihat apakah pada hari dan jam yang sama pada slot ini terdapat kelas yang sama dengan 1 angkatan di atas / di bawahnya. Jika iya beri nilai 0.25.

Catatan:

- Semakin sedikit ruangan yang dapat digunakan suatu kelas matakuliah maka semakin besar nilai constraint ruanganya. Hal ini dilakukan untuk mendahulukan sebuah matakuliah agar dapat terjadwal dengan menempati ruangan yang ditentukan.
- Semakin sedikit waktu yang dimiliki dosen untuk mengajar suatu kelas matakuliah maka semakin besar nilai constraint waktunya. Hal ini dilakukan untuk mendahulukan sebuah matakuliah agar dapat terjadwal sesuai dengan waktu yang dimiliki dosen tersebut agar tidak digunakan matakuliah lain.
- Ada beberapa kelas matakuliah yang dijadwalkan 1 waktu seperti lab fisika. Diasumsikan kelas itu digabung agar seluruh kelas dapat disatukan satu waktu maka dibuat parameter tipe kelas. sebuah kelas matakuliah bertipe gabung memiliki nilai 0.45 dengan harapan kelas yang digabung dijadwalkan terlebih dahulu sehinga semua kelas dapat disatukan dalam satu waktu tersebut.
- Sehingga jika parameter bernilai maksimal maka seluruh nilai conflict 0.1 + 0.45 + 0.45 = 1

Tree jadwal

Pada studi kasus ini tree yang digunakan dengan contoh kasus tiga hari dan lima slot waktu setiap harinya.



Gambar 3. Tree Studi Kasus

Digit pertama menggambarkan hari.
1 = senin s.d 3 = rabu
Digit kedua menggambarkan jam pelajaran
1 = 07.00-08.00 s.d 5 = 11.00-12.00

Penjadwalan

Hal pertama yang dilakukan adalah mengurutkan matakuliah berdasarkan nilai konflik terbesar, tingkatan, dan dosen.

- Pengurutan berdasarkan nilai konflik didahulukan agar jadwal yang memiliki constraint dapat menempati ruang/waktu yang sesuai.
- Pengurutan tingkat untuk mendahulukan matakuliah awal yang cenderung memiliki jadwal yang lebih padat daripada tingkat akhir.
- Pengurutan berdasarkan dosen yang paling banyak mengajarkan agar matakuliah – matakuliah tersebut dapat dijadwalkan terlebih dahulu pada saat slot – slot waktu masih cukup banyak memiliki opsi.

Nilai konflik matakuliah :
M1 Kelas A diajar oleh D1 = M1AD1. Jumlah mahasiswa = 60. Maka :

- N1 = 0.1/1 =0.1
- N2 = 0.45/(0/4) = 0
- N3 = 0
- NC = 0.1+0+0 = 0.1

M3 Kelas A diajar oleh D3 = M3AD3. Jumlah mahasiswa = 30.
Maka :

- N1 = 0.1/0 = 0
- N2 = 0.45/(4/4) = 0.45
- N3 = 0
- NC = 0+0.45+0 = 0.45

Dengan mengurutkan dari nilai konflik terbesar, maka didapatkan urutan penjadwalan sebagai berikut :

1. M3AD3
2. M1AD1

- M3AD3

Langkah Pertama

1. Tentukan Start Node 1.1
2. Jika matakuliah memiliki contstraint waktu ajar, pada setiap node yang tidak termasuk constraint waktu ajar beri label skip sehingga tidak akan di cek lagi kemungkinan untuk menjadwalkan di waktu tersebut. Kemudian ganti StartNode menjadi waktu awal yang terdapat di constraint waktu ajar (1.1). Jika tidak lanjutkan ke no 3.
3. Cari nilai f curNode [1.1]
   P1 = 0
   P2 = 0
   P3 = 0
   G = 0
   H = P1+P2+P3 = 0
   F = 0+0 = 0
4. Sisipkan curNode di list openNode=[1.1]
5. Ambil node yang memiliki nilai f terkecil (1.1)
6. Check apakah nilai f=0 Jika iya check apakah pada slot node 1.1 dapat dilakukan perkuliahan
7. Berdasarkan constraint waktu mengajar dosen matakuliah M1AD1 dapat diljadwalkan pada slot 1.1 (senin jam ke 2 hingga jam ke 5 (4Sks pelajaran. Setelah mendapatkan slot waktu, dilakukan pengecekan ruangan yang akan digunakan. Karena Kelas matakuliah tidak memiliki constraint ruangan maka dilakukan pengecekan ketersedian ruangan R1, R2 dan R3 pada slot tersebut dimana R1 pada slot 1.1 tersedia, sehingga matakuliah dapat dijadwalkan pada waktu tersebut.
8. Pencarian slot waktu dihentikan.
9. Simpan jadwal.

- M1AD1

Langkah Pertama

1. Tentukan StartNode = 1.1
2. Jika matakuliah memiliki contstraint waktu ajar, pada setiap node yang tidak termasuk constraint waktu ajar beri label skip sehingga tidak akan di cek lagi kemungkinan untuk menjadwalkan di waktu tersebut. Kemudian ganti StartNode

menjadi waktu awal yang terdapat di constraint waktu ajar. Jika tidak lanjutkan ke no 3.

3. Cari nilai f curNode [1.1] =
   P1 = 0.40
   P2 = 0
   P3 = 0
   G = 0;
   H = P1+P2+P3 = 0.40
   F = 0.40
4. Sisipkan curNode di list openNode = [1.1]
5. Ambil node yang memiliki nilai f terkecil di dalam list openNode (1.1)
6. Check apakah nilai f=0, bila tidak, berikan label skip pada node, jika benar check apakah pada slot node 1.1 dapat dilakukan perkuliahan. Jika tidak beri label skip pada node kemudian dilanjutkan dengan membuka node tetangga pada slot 1.1 yakni (1.2, 2.1).
7. Simpan node terpilih di closeNode [1.1] hapus node terpilih dari openNode.
8. Hitung f untuk 1.2
   P1 = 0.40
   P2 = 0
   P3 = 0
   G = 2
   H= P1+P2+P3 = 0.4
   F= 2.4
   Simpan di openNode=[1.2]
   9. Hitung f untuk 2.1
   P1 = 0
   P2 = 0
   P3 = 0
   G = 10
   H = 0
   F =10
   Simpan di openNode=[2.1,1.2]

Langkah Kedua

1. Ambil node yang memiliki nilai f terkecil [2.1, 1.2] (1.2)
2. Check apakah nilai f=0, bila tidak, berikan label skip pada node, jika benar check apakah pada slot node 1.2 dapat dilakukan perkuliahan. Jika tidak beri label skip pada node kemudian dilanjutkan dengan membuka node tetangga pada slot 1.2 yakni (1.1, 1.3 , 1.4).
3. Simpan node terpilih di closeNode [1.1,1.2] hapus node terpilih dari openNode.
4. Hitung f untuk 1.3
   P1 = 0.40
   P2 = 0
   P3 = 0
   G = 4
   H= P1+P2+P3 = 0.4
   F= 4.4
   5. Hitung f untuk 1.4
   P1 = 0.40
   P2 = 0
   P3 = 0
   G = 4
   H= P1+P2+P3 = 0.4
   F= 4.4

Langkah Ketiga

1. Ambil node yang memiliki nilai f terkecil di openNode [1.3, 1.4 , 2.1] (terpilih 1.3).
2. Check apakah nilai f=0, bila tidak, berikan label skip pada node, jika iya check apakah pada slot node 1.3 dapat dilakukan perkuliahan namun karena node 1.3 memiliki label skip maka dilanjutkan ke proses no. 3
3. Buka node tetangga pada slot 1.3 yakni (1.2).
4. Simpan node terpilih di closeNode [1.2 , 1.1, 1.3] hapus node terpilih dari openNode.
5. Karena 1.2 telah tersimpan di closedNode, perhitungan node ini dilewati.

Langkah Keempat

1. Ambil node yang memiliki nilai f terkecil di openNode [1.4 , 2.1] (terpilih 1.4).
2. Check apakah nilai f=0, bila tidak, berikan label skip pada node, jika benar check apakah pada slot node 1.4 dapat dilakukan perkuliahan namun karena node 1.4 memiliki label skip maka dilanjutkan ke proses no 3
3. Buka node tetangga pada slot 1.4 yakni (1.2 , 1.5).
4. Simpan node terpilih di closeNode [1.2 , 1.1, 13, 1.4] hapus node terpilih dari openNode.
5. Hitung f untuk 1.5
   P1 = 0.40
   P2 = 0
   P3 = 0
   G = 7
   H = Nilai Kualitas Slot Jadwal =0.40
   F=7.40
6. Simpan di openNode = [1.5 , 2.1]
7. Karena 1.2 telah tersimpan di closedNode, perhitungan node ini dilewati.

Langkah Kelima

1. Ambil node yang memiliki nilai f terkecil di openNode [1.5 , 2.1] (terpilih 1.5)
2. Check apakah nilai f=0, bila tidak, berikan label skip pada node. Jika benar, check apakah pada slot node 1.5 dapat dilakukan perkuliahan, bila tidak bisa dilakukan maka beri label skip pada node 1.5 dan lanjutkan ke proses no 3.
3. Buka node tetangga pada slot 1.5 yakni (1.4).
4. Simpan node terpilih di closeNode [1.2 , 1.1, 13, 1.4, 1.5] hapus node terpilih dari openNode.
5. Karena 1.4 telah tersimpan di closedNode, maka perhitungan node ini dilewati.

Langkah Keenam

1. Ambil node yang memiliki nilai f terkecil di openNode [2.1] (terpilih 2.1)
2. Check apakah nilai f=0, bila tidak, berikan label skip pada node [2.1]. Jika iya check apakah pada slot node 2.1 dapat dilakukan perkuliahan, bila tidak, maka berikan label skip pada node 2.1 dan lanjutkan ke proses no 3.
3. Buka node tetangga pada slot 2.1 yakni (2.2 , 3.1).
4. Simpan node terpilih di closeNode [1.2 , 1.1, 13, 1.4, 1.5, 2.1] hapus node terpilih dari openNode.
5. Hitung f untuk 3.1
   P1 = 0
   P2 = 0
   P3 = 0
   G = 20
   H = Nilai Kualitas Slot Jadwal =0
   F=20+0=20
   Simpan di openNode=[3.1]
6. Hitung f untuk 2.2
   P1 = 0
   P2 = 0
   P3 = 0
   G=12
   H = nilai Kualitas Slot Jadwal = 0
   F =12+0 = 12
   Simpan di openNode=[2.2, 3.1]

Langkah Ketujuh

1. Ambil node yang memiliki nilai f terkecil di openNode [2.2 , 3.1] (terpilih 2.2)
2. Check apakah nilai f=0, bila tidak, berikan label skip pada node, jika iya check apakah pada slot node 2.2 dapat dilakukan perkuliahan namun karena node memiliki label skip maka dilanjutkan ke proses no 3
3. Buka node tetangga pada slot 2.2 yakni (2.3 , 2.4).
4. Simpan node terpilih di closeNode [1.2 , 1.1, 13, 1.4, 1.5, 2.1, 2.2] hapus node terpilih dari openNode.
5. Hitung f untuk 2.3
   P1 = 0
   P2 = 0
   P3 = 0
   G = 14
   H= P1+P2+P3 = 0
   F = 0+14 = 14
   Simpan di openNode = [2.3, 3.1]
6. Hitung f untuk 2.4
   P1 = 0
   P2 = 0
   P3 = 0
   G = 15
   H= P1+P2+P3 = 0
   F = 0+15 = 15
   Simpan di openNode = [2.3, 2.4, 3.1]

Langkah Kedelapan

1. Ambil node yang memiliki nilai f terkecil di openNode [2.3, 2.4, 3.1] (terpilih 2.3)
2. Check apakah nilai f=0, bila tidak, berikan label skip pada node, jika iya check apakah pada slot node 2.3 dapat dilakukan perkuliahan namun karena node 2.3 memiliki label skip maka dilanjutkan ke proses no 3
3. Buka node tetangga pada slot 2.3 yakni (2.2).
4. Simpan node terpilih di closeNode [1.2 , 1.1, 13, 1.4, 1.5, 2.1, 2.2, 2.3] hapus node terpilih dari openNode.
5. Karena 2.2 telah tersimpan di closedNode, perhitungan node ini dilewati

Langkah Kesembilan

1. Ambil node yang memiliki nilai f terkecil di openNode [2.4, 3.1] (terpilih 2.4)
2. Check apakah nilai f=0, bila tidak, berikan label skip pada node, Jika benar check apakah pada slot node 2.4 dapat dilakukan perkuliahan, namun karena node 2.4 memiliki label skip maka dilanjutkan ke proses no 3
3. Buka node tetangga pada slot 2.4 yakni (2.2, 2.5).
4. Karena node 2.2 sudah ada pada closeNode, maka proses untuk node 2.2 tidak perlu dilakukan lagi.
5. Simpan node terpilih di closeNode [1.2 , 1.1, 13, 1.4, 1.5, 2.1, 2.2, 2.3, 2.4] hapus node terpilih dari openNode.
6. Hitung f untuk 2.5
   P1 = 0

P2 = 0
P3 = 0
G = 17
H= P1+P2+P3 = 0
F = 0+17 = 17
Simpan di openNode = [2.5, 3.1]

Langkah 10

1. Ambil node yang memiliki nilai f terkecil di openNode [2.5, 3.1] (terpilih 2.5)
2. Check apakah nilai f=0, bila tidak, berikan label skip pada node, Jika benar check apakah pada slot node 2.5 dapat dilakukan perkuliahan, namun karena node 2.5 memiliki label skip maka dilanjutkan ke proses no 3
3. Buka node tetangga pada slot 2.5 yakni (2.4).
4. Simpan node terpilih di closeNode [1.2 , 1.1, 13, 1.4, 1.5, 2.1, 2.2, 2.3, 2.4, 2.5] hapus node terpilih dari openNode.
5. Karena 2.4 telah tersimpan di closedNode, perhitungan node ini dilewati.

Langkah 11

1. Ambil node yang memiliki nilai f terkecil di openNode [3.1] (terpilih 3.1)
2. Check apakah nilai f=0, bila tidak, berikan label skip pada node, Jika benar check apakah pada slot node 3.1 dapat dilakukan perkuliahan, namun karena node 3.1 memiliki label skip maka dilanjutkan ke proses no 3
3. Buka node tetangga pada slot 3.1 yakni (3.2 , 2.1).
4. Simpan node terpilih di closeNode [1.2 , 1.1, 13, 1.4, 1.5, 2.1, 2.2, 2.3, 2.4, 2.5, 3.1] hapus node terpilih dari openNode.
5. Hitung f untuk 3.2

P1 = 0
P2 = 0
P3 = 0
G = 22
H= P1+P2+P3 = 0
F = 0+22 = 22
Simpan di openNode = [3.2]

6. Karena node 2.1 sudah berada pada closeNode, maka perhitungan node ini dilewati.

Langkah 12

1. Ambil node yang memiliki nilai f terkecil di openNode [3.2] (terpilih 3.2)
2. Check apakah nilai f=0, bila tidak, berikan label skip pada node, Jika benar check apakah pada slot node 3.2 dapat dilakukan perkuliahan, namun karena node 3.2 memiliki label skip maka dilanjutkan ke proses no 3
3. Buka node tetangga pada slot 3.2 yakni (3.3 , 3.4).
4. Simpan node terpilih di closeNode [1.2 , 1.1, 13, 1.4, 1.5, 2.1, 2.2, 2.3, 2.4, 2.5, 3.1, 3.2] hapus node terpilih dari openNode.
5. Hitung f untuk 3.3

P1 = 0
P2 = 0
P3 = 0
G = 24
H= P1+P2+P3 = 0
F = 0+24 = 24
Simpan di openNode = [3.3]

6. Hitung f untuk 3.4

P1 = 0
P2 = 0
P3 = 0
G = 25
H= P1+P2+P3 = 0
F = 0+25 = 25
Simpan di openNode = [3.3, 3.4]

Langkah 13

1. Ambil node yang memiliki nilai f terkecil di openNode [3.3, 3.4] (terpilih 3.3)
2. Check apakah nilai f=0, bila tidak, berikan label skip pada node, Jika benar check apakah pada slot node 3.3 dapat dilakukan perkuliahan, namun karena node 3.3 memiliki label skip maka dilanjutkan ke proses no 3.
3. Buka node tetangga pada slot 3.3 yakni (3.2).
4. Simpan node terpilih di closeNode [1.1 , 1.0, 1.2,1.3,1.4,2.0,2.1,2.2 2.3 , 2.4, 3.0 , 3.1 , 3.2, 3.3] hapus node terpilih dari openNode.
5. Karena 3.2 telah tersimpan di closedNode, perhitungan node ini dilewati.

Langkah 14

1. Ambil node yang memiliki nilai f terkecil di openNode [3.4] (terpilih 3.4)
2. Check apakah nilai f=0, bila tidak, berikan label skip pada node, Jika benar check apakah pada slot node 3.4 dapat dilakukan perkuliahan, namun karena node 3.4 memiliki label skip maka dilanjutkan ke proses no 3.
3. Buka node tetangga pada slot 3.4 yakni (3.5).
4. Simpan node terpilih di closeNode [1.1 , 1.0, 1.2,1.3,1.4,2.0,2.1,2.2 2.3 , 2.4, 3.0 , 3.1 , 3.2, 3.3, 3.4] hapus node terpilih dari openNode.
5. Hitung f untuk 3.5

P1 = 0
P2 = 0
P3 = 0
G = 29
H = P1+P2+P3 = 0
F = 29+0 = 29
Simpan di openNode=[3.5]

Langkah 15

1. Ambil node yang memiliki nilai f terkecil di openNode [3.5] (terpilih 3.5)
2. Check apakah nilai f=0, bila tidak, berikan label skip pada node, Jika benar check apakah pada slot node 3.5 dapat dilakukan perkuliahan, namun karena node 3.5 memiliki label skip maka dilanjutkan ke proses no 3.
3. Buka node tetangga pada slot 3.5 yakni (3.4).
4. Simpan node terpilih di closeNode [1.1 , 1.0, 1.2,1.3,1.4,2.0,2.1,2.2 2.3 , 2.4, 3.0 , 3.1 , 3.2, 3.4, 3.5] hapus node terpilih dari openNode.
5. Karena 3.4 telah tersimpan di closedNode, perhitungan node ini dilewati.

Langkah 16

1. Karena pada list openNode telah habis maka dilakukan penyortiran terhadap penemuan slot waktu dengan nilai kualitas terendah yang.
2. Didapatkan hasil bahwa nilai terendah didapatkan pada node 1.1.
3. Karena waktu yang dibutuhkan dalam penjadwalan adalah empat jam (empat sks) dan waktu yang tersedia di hari senin hanya node 1.3, 1.4 dan 1.5, maka diambil node dengan nilai terendah lainya yaitu 2.1. Setelah mendapatkan slot waktu, lakukan pengecekan ruang yang akan digunakan. Karena Kelas matakuliah memiliki constraint ruangan maka dilakukan pengecekan ketersedian ruangan sesuai constraint matakuliah yaitu ruangan R1. Dimana R1 pada slot 2.2 tersedia, sehingga matakuliah dapat dijadwalkan pada waktu dan ruangan tersebut.
4. Simpan jadwal.

KESIMPULAN

Kesimpulan yang didapat dari penelitian ini adalah sistem pendukung keputusan penjadwalan mata kuliah dengan model pemrograman *heuristic* menggunakan algoritma A* ini menghasilkan solusi jadwal mata kuliah yang dapat menjadi pertimbangan pengambil keputusan (decision maker) dalam membangun jadwal mata kuliah.

REFERENSI

Puspaningrum, W. A., Djunaidy, A., & Vinarti, R. A. (2013). Penjadwalan Mata Kuliah Menggunakan Algoritma Genetika di Jurusan Sistem Informasi ITS. *Jurnal Teknik Pomits*, *2*(1), 127–131. https://doi.org/10.12962/j23373539.v2i1.3234

Russell, S., & Norvig, P. (2013). *Artificial Intelligence A Modern Approach*. *Zhurnal Eksperimental'noi i Teoreticheskoi Fiziki*. https://doi.org/10.1017/S0269888900007724

Simamora, Purwanto. Tulus. Batubara R, F. (2007). Penjadwalan Kuliah Dengan Menggunakan Algoritma Genetika Studi Kasus Fakultas Teknik Universitas Sumatera Utara, (1), 1–4. https://doi.org/10.1007/s13398-014-0173-7.2