Resume Chapter 10

Kelompok 1

Muhammad Sabri S (D121181307)
Dea Nurhikma (D121181308)
Rahmadani (D121181328)
Ryan Terry Thahir (D121181330)
Muhammad Rezaldi Yanata Putra (D121191041)

Fakultas Teknik
Universitas Hasanuddin
2021

Pemodelan dan Simulasi untuk Manajemen proyek

Alat simulasi membantu kami meningkatkan perencanaan dan koordinasi dalam berbagai fase proyek sehingga kami selalu dapat mengontrolnya.

- Memperkenalkan manajemen proyek

Untuk menilai konsekuensi dari langkah strategis atau taktis sebelumnya, perusahaan membutuhkan sistem prediksi yang andal. Sistem analisis prediktif didasarkan pada pengumpulan data dan proyeksi skenario yang andal dalam jangka menengah dan panjang. Dengan cara ini, kami dapat memberikan indikasi dan pedoman untuk strategi yang kompleks, terutama yang harus mempertimbangkan banyak faktor dari entitas yang berbeda.

- Memahami analisis bagaimana-jika

Analisis bagaimana-jika adalah jenis analisis yang dapat berkontribusi secara signifikan untuk membuat keputusan manajerial lebih efektif, aman, dan terinformasi. Ini juga merupakan tingkat dasar analisis prediktif berdasarkan data. Analisis bagaimana-jika adalah alat yang mampu mengelaborasi skenario yang berbeda untuk menawarkan kemungkinan hasil yang berbeda. Tidak seperti analisis prediktif lanjutan, analisis bagaimana-jika memiliki keuntungan karena hanya membutuhkan data dasar untuk diproses.

- Mengelola masalah hutan kecil

suatu proses disebut Markovian ketika evolusi proses di masa depan hanya bergantung pada pengamatan instan terhadap sistem dan sama sekali tidak bergantung pada masa lalu. MDP dicirikan oleh lima elemen: zaman keputusan, status, tindakan, probabilitas transisi, dan penghargaan.

Dalam proses Markovian, pengambil keputusan memiliki opsi untuk memilih tindakan mana yang akan dilakukan di setiap status sistem. Tindakan yang dipilih membawa sistem ke keadaan berikutnya dan hadiah untuk pilihan itu dikembalikan. Transisi dari satu keadaan ke keadaan lain menikmati properti Markov: keadaan saat ini hanya bergantung pada yang sebelumnya. Sebuah proses Markov didefinisikan oleh empat elemen, sebagai berikut:

• S: Status sistem.

• SEBUAH: Tindakan yang tersedia untuk setiap negara bagian.

• P: Matriks transisi. Ini berisi probabilitas bahwa suatu tindakansebuah mengambil sistem dari S menyatakan untuk S' negara.

• R: Hadiah yang diperoleh dalam transisi dari S menyatakan untuk S' menyatakan dengan tindakan sebuah.

Kebijakan memetakan keadaan lingkungan dan tindakan yang akan dipilih ke keadaan tersebut, mewakili seperangkat aturan atau asosiasi yang merespons suatu stimulus. Tujuan kebijakan adalah untuk memaksimalkan total hadiah yang diterima melalui seluruh urutan tindakan yang dilakukan oleh sistem. Total hadiah yang diperoleh dengan mengadopsi kebijakan dihitung sebagai berikut: =+1 = ++1 + + =0 Pada persamaan sebelumnya, R adalah hadiah dari tindakan yang membawa lingkungan ke kondisi terminal S . Untuk mendapatkan total hadiah maksimum, kita dapat memilih tindakan yang memberikan hadiah tertinggi untuk setiap negara bagian. Ini mengarah pada pemilihan kebijakan optimal yang memaksimalkan total hadiah.

- Menjelajahi proses pengoptimalan

Proses Keputusan Markov Berbasis Simulasi, masalah MDP dapat diatasi dengan menggunakan pemrograman dinamis (DP). DP adalah teknik pemrograman yang bertujuan untuk menghitung kebijakan optimal berdasarkan model yang diketahui dari lingkungan. Inti dari DP adalah memanfaatkan state-value dan action-value untuk mengidentifikasi kebijakan yang baik. Dalam metode DP, dua proses yang disebut evaluasi kebijakan dan perbaikan kebijakan digunakan. Proses-proses tersebut saling berinteraksi, sebagai berikut:

- Evaluasi kebijakan dilakukan melalui proses iteratif yang berupaya memecahkan persamaan Bellman. Konvergensi proses untuk k → memberlakukan aturan aproksimasi, sehingga memperkenalkan kondisi berhenti.
- Perbaikan kebijakan meningkatkan kebijakan saat ini berdasarkan nilai-nilai saat ini.

Algoritme kemudian, melalui prosedur rekursif, memperbarui kedua vektor ini.

- Memperkenalkan MDPtoolbox

Itu kotak alat MDP Paket berisi beberapa fungsi yang terhubung dengan resolusi proses keputusan Markov waktu diskrit, yaitu iterasi nilai, cakrawala terbatas, iterasi kebijakan, algoritma pemrograman linier dengan beberapa varian, dan beberapa fungsi yang dapat kita gunakan untuk melakukan analisis pembelajaran penguatan.

- Mendefinisikan contoh kecil pengelolaan hutan

Untuk menganalisis secara rinci bagaimana menangani masalah manajemen menggunakan proses Markovian, kami akan menggunakan contoh yang sudah tersedia di: kotak alat MDP kemasan. Ini berkaitan dengan pengelolaan hutan kecil di mana ada dua jenis sumber daya: fauna liar dan pohon. Pohon-pohon di hutan bisa ditebang, dan kayu yang diperoleh bisa dijual. Pengambil keputusan memiliki dua tindakan: menunggu dan memotong. Tindakan pertama adalah menunggu pohon tumbuh sepenuhnya sebelum memotongnya untuk mendapatkan lebih banyak kayu. Tindakan kedua melibatkan menebang pohon untuk mendapatkan uang segera. Pengambil keputusan memiliki tugas membuat keputusan mereka setiap 20 tahun.

Mengatasi masalah manajemen menggunakan MDPtoolbox Tujuan kami adalah mengembangkan kebijakan yang memungkinkan kami mengelola hutan kecil untuk mendapatkan hadiah maksimal. Kami akan melakukan ini menggunakankotak alat MDP package, yang kami perkenalkan di bagian sebelumnya, dan menganalisis kode baris demi baris:

1. Mari kita mulai seperti biasa dengan mengimpor perpustakaan yang diperlukan: impor mdptoolbox.example Dengan melakukan ini, kami mengimpor kotak alat MDP modul, yang berisi data untuk contoh ini.

2. Untuk memulai, kita akan mengekstrak matriks transisi dan vektor hadiah:

P, R = mdptoolbox.example.forest()

Perintah ini mengambil data yang disimpan dalam contoh. Untuk memastikan data benar, kami mencetak konten variabel ini, mulai dari matriks transisi:

cetak(P[0]) Matriks berikut dicetak:

[[0.1 0.9 0.]

[0.1 0. 0.9]

[0.1 0. 0.9]]

Ini adalah matriks transisi untuk tindakan menunggu. Konsisten dengan apa yang ditunjukkan dalam Mendefinisikan contoh kecil pengelolaan hutan bagian, setelah mengatur p = 0,1, kita dapat mengkonfirmasi matriks transisi. Sekarang, kami

mencetak matriks transisi yang terkait dengan aksi potong: cetak(P[1]) Matriks berikut dicetak:

[[1. 0. 0.]

[1. 0. 0.]

[1. 0. 0.]]

Matriks ini juga konsisten dengan apa yang telah didefinisikan sebelumnya. Sekarang, mari kita periksa bentuk vektor hadiah, dimulai dengan tindakan menunggu: cetak(R[:,0]) Mari kita lihat isinya:

[0. 0. 4.]

Jika aksi potong dipilih, kami akan mendapatkan hadiah berikut: cetak(R[:,1]) Vektor berikut dicetak: [0. 1. 2.] Akhirnya, mari kita perbaiki faktor diskon: gamma = 0,9 Semua data masalah sekarang telah didefinisikan. Kita sekarang dapat melanjutkan dan melihat model secara lebih rinci.

3. Waktunya telah tiba untuk menerapkan algoritme iterasi kebijakan pada masalah yang baru saja kita definisikan: PolIterModel = mdptoolbox.mdp.PolicyIteration(P, R, gamma) PolIterModel.run()

Fungsi, mulai dari inisial P0 kebijakan, memperbarui nilai fungsi dan kebijakan melalui prosedur berulang, bergantian dua fase berikut:

• Evaluasi kebijakan: Mengingat kebijakan saat ini P, perkirakan fungsi nilai aksi.

• Perbaikan Kebijakan: Jika kita menghitung kebijakan yang lebih baik berdasarkan fungsi nilai tindakan, maka kebijakan ini dijadikan kebijakan baru dan kita kembali ke langkah sebelumnya. Ketika fungsi nilai dapat dihitung secara tepat untuk setiap pasangan aksi-negara, iterasi kebijakan yang kami lakukan dengan perbaikan kebijakan serakah mengarah ke konvergensi dengan mengembalikan kebijakan yang optimal. Pada dasarnya, berulang kali mengeksekusi kedua proses ini menyatukan proses umum menuju solusi optimal. Dalam mdptoolbox.mdp.PolicyIteration() fungsi, kami telah melewati argumen berikut: • P: Probabilitas transisi

• R: Penghargaan

• gamma: Faktor diskon Hasil berikut dikembalikan:

• V: Fungsi nilai optimal. V adalah S vektor panjang.

• aturan: Kebijakan yang optimal. Kebijakan tersebut merupakanS vektor panjang. Setiap elemen adalah bilangan bulat yang sesuai dengan tindakan yang

memaksimalkan fungsi nilai. Dalam contoh ini, hanya dua tindakan yang diperkirakan: 0 = tunggu, 1 = potong.

• iter: Jumlah iterasi.

• waktu: Waktu CPU yang digunakan untuk menjalankan program. Sekarang model sudah siap, kita harus mengevaluasi hasilnya dengan memeriksa kebijakan yang diperoleh. Untuk memulai, kami memeriksa pembaruan fungsi nilai: cetak (PolIterModel.V)

Hasil berikut dikembalikan: (26.2440000000000014, 29.484000000000016, 33.484000000000016) Sebuah fungsi nilai menentukan seberapa baik untuk sistem suatu negara. Nilai ini mewakili total hadiah yang diharapkan untuk sistem dari statusS. Fungsi nilai bergantung pada kebijakan yang dipilih agen untuk tindakan yang akan dilakukan.

Mari kita lanjutkan dan ekstrak kebijakannya:

cetak(PoliIterModel.policy)

Sebuah kebijakan menyarankan perilaku sistem pada waktu tertentu. Ini memetakan keadaan lingkungan yang terdeteksi dan tindakan yang harus diambil ketika mereka berada di keadaan tersebut. Ini sesuai dengan apa, dalam psikologi, yang disebut seperangkat aturan atau asosiasi respons stimulus. Kebijakan adalah elemen penting dari model MDP karena mendefinisikan perilaku. Hasil berikut dikembalikan:

(0, 0, 0)

Di sini, kebijakan optimal adalah tidak menebang hutan di ketiga negara bagian. Hal ini disebabkan oleh rendahnya kemungkinan terjadinya kebakaran, yang menyebabkan tindakan menunggu menjadi tindakan terbaik untuk dilakukan. Dengan cara ini, hutan memiliki waktu untuk tumbuh dan kita dapat mencapai kedua tujuan: memelihara hutan tua untuk satwa liar dan mendapatkan uang dengan menjual kayu yang dipotong.

Mari kita lihat berapa banyak iterasi yang telah dibuat:

cetak (PolIterModel.iter)

Hasil berikut dikembalikan: 2 Akhirnya, mari kita cetak waktu CPU: print(PolIterModel.time) Hasil berikut dikembalikan: 0.12830829620361328 Hanya 0,13 detik diperlukan untuk melakukan prosedur iterasi nilai.

- Mengubah kemungkinan kebakaran

pikirkan saja tempat-tempat yang hangat terutama yang terkena angin kencang. Untuk memodelkan kondisi baru ini, cukup ubah pengaturan masalah dengan mengubah nilai probabilitasP. Itumdptoolbox.contoh.hutan() modul memungkinkan kita untuk memodifikasi karakteristik dasar dari masalah. Mari kita mulai:

1. Mari kita mulai dengan mengimpor modul contoh: impor mdptoolbox.example P, R = mdptoolbox.example.forest(3,4,2,0.8) Berbeda dengan contoh yang dibahas pada bagian sebelumnya, Mengatasi masalah manajemen, di mana kami menggunakan kotak alat MDP, dalam hal ini, kami telah melewati beberapa parameter. Mari kita menganalisis mereka secara rinci. Dalammdptoolbox.contoh. hutan ()fungsi, kami melewati parameter berikut (3, 4, 2, 0.8). Mari kita menganalisis artinya: 3: Jumlah negara bagian. Ini harus berupa bilangan bulat yang lebih besar dari 0. 4: Hadiah ketika hutan berada dalam status tertua dan tindakan menunggu dilakukan. Ini harus berupa bilangan bulat yang lebih besar dari 0. 2: Hadiahnya saat hutan berada dalam kondisi tertua dan aksi tebang dilakukan. Ini harus berupa bilangan bulat yang lebih besar dari 0. 0.8: Kemungkinan terjadinya kebakaran. Ini harus dalam]0, 1[. Dengan menganalisis data masa lalu, kita dapat melihat bahwa kita mengkonfirmasi tiga parameter pertama, sementara kita hanya mengubah kemungkinan terjadinya kebakaran, sehingga meningkatkan kemungkinan ini dari 0,1 menjadi 0,8. Mari kita lihat perubahan ini pada data awal karena mengubah matriks transisi: cetak(P[0]) Matriks berikut dicetak: [[0.8 0.2 0.] [0.8 0. 0.2] [0.8 0. 0.2]

Mari kita lihat apa yang terjadi ketika matriks transisi ditautkan ke aksi potong: cetak(P[1]) Matriks berikut dicetak: [[1. 0. 0.] [1. 0. 0.] [1. 0. 0.]] Matriks ini tetap tidak berubah. Ini karena hasil dari tindakan potong, yang mengembalikan sistem ke keadaan awalnya. Demikian juga, vektor hadiah mungkin tidak berubah karena hadiah yang diberikan sama dengan yang disediakan oleh masalah default.

Mari kita cetak nilai-nilai ini: cetak(R[:,0]) Ini adalah vektor hadiah yang terhubung ke tindakan menunggu. Vektor berikut dicetak: [0. 0. 4.] Dalam kasus aksi potong, vektor berikut disetel ulang:

cetak(R[:,1]) Vektor berikut dicetak: [0. 1. 2.] Seperti yang diharapkan, tidak ada yang berubah.

Akhirnya, mari kita perbaiki faktor diskon: gamma = 0,9 Semua data masalah sekarang telah didefinisikan. Kita sekarang dapat melanjutkan dan melihat model secara lebih rinci. 2.

Sekarang kita akan menerapkan algoritma iterasi nilai: PolIterModel = mdptoolbox.mdp.PolicyIteration(P, R, gamma) PolIterModel.run()

Sekarang, kita dapat mengekstrak hasilnya, dimulai dengan fungsi nilai: cetak (PolIterModel.V) Hasil berikut dicetak: (1.5254237288135597, 2.3728813559322037, 6.217445225299711) Sekarang mari kita menganalisis bagian penting dari masalah: mari kita lihat kebijakan apa yang disarankan model simulasi kepada kita: cetak(PoliIterModel.policy) Hasil berikut dicetak: (0, 1, 0) Dalam hal ini, perubahan yang kami buat di sini, dibandingkan dengan contoh default, adalah substansial. Disarankan agar kita mengadopsi tindakan menunggu jika kita berada di negara bagian 1 dan 3, sedangkan jika kita berada di negara bagian 2, disarankan untuk mencoba menebang hutan. Karena kemungkinan kebakaran tinggi, lebih mudah untuk memotong kayu yang sudah tersedia dan menjualnya sebelum api menghancurkannya sepenuhnya. Kemudian, kami mencetak jumlah iterasi dari masalah: cetak (PolIterModel.iter) Hasil berikut dicetak: 1 Akhirnya, kami mencetak waktu CPU: print(PolIterModel.time) Hasil berikut dicetak: 0.14069104194641113 Contoh-contoh ini telah menyoroti betapa sederhananya prosedur pemodelan masalah manajemen melalui penggunaan MDP.

- Penjadwalan waktu proyek menggunakan simulasi Monte Carlo

Syaratmanajemen proyekmengacu pada penerapan pengetahuan, keterampilan, alat, dan teknik untuk tujuan perencanaan, pengelolaan, dan pengendalian proyek dan kegiatan yang terdiri.

Untuk proyek yang kompleks, perlu untuk membuat struktur yang teratur dengan menguraikan proyek menjadi tugas-tugas yang lebih sederhana. Untuk setiap tugas, perlu untuk menentukan aktivitas dan waktu pelaksanaan.

- Mendefinisikan grid penjadwalan

Bagian mendasar dari semua manajemen proyek adalah membangun grid penjadwalan. Ini adalah grafik berorientasi yang mewakili suksesi temporal dan ketergantungan logis antara kegiatan yang terlibat dalam realisasi proyek. Selain membangun grid, proses penjadwalan juga menentukan waktu mulai dan berakhirnya aktivitas berdasarkan faktor-faktor seperti durasi, sumber daya, dan sebagainya.

- Memperkirakan waktu tugas

Durasi tugas-tugas ini seringkali sulit diperkirakan karena sejumlah faktor yang dapat mempengaruhinya: ketersediaan dan/atau produktivitas sumber daya, kendala teknis dan fisik antara kegiatan, dan komitmen kontrak.

Mengembangkan algoritma untuk penjadwalan proyek Pada bagian ini, kami akan menganalisis algoritma untuk penjadwalan proyek berdasarkan simulasi Monte Carlo. Kami akan melihat semua perintah secara rinci, baris demi baris:

1. Mari kita mulai dengan mengimpor perpustakaan yang akan kita gunakan dalam algoritme:

impor numpy sebagai np

impor acak

impor panda sebagai pd

numpy (np) adalah pustaka Python yang berisi banyak fungsi yang membantu kami mengelola matriks multidimensi. Selain itu, ini berisi banyak koleksi fungsi matematika tingkat tinggi yang dapat kita gunakan pada matriks ini. Itu acak library mengimplementasikan generator nomor pseudo-random untuk berbagai distribusi. Ituacak modul didasarkan pada algoritma Mersenne Twister. Itu panda library adalah library berlisensi BSD open source yang berisi struktur data dan operasi yang dapat digunakan untuk memanipulasi nilai numerik kinerja tinggi untuk bahasa pemrograman Python.

2. Mari kita lanjutkan dan inisialisasi parameter dan variabel:

N = 10000

TotalWaktu=[]

T = np.kosong(bentuk=(N,6))

n mewakili jumlah poin yang kami hasilkan. Ini adalah angka acak yang akan membantu kami menentukan waktu tugas itu. ItuTotalWaktu variabel adalah daftar yang akan berisi n penilaian dari keseluruhan waktu yang dibutuhkan untuk menyelesaikan proyek. Akhirnya, T variabel adalah matriks dengan n baris dan enam kolom dan akan berisi n penilaian waktu yang dibutuhkan untuk menyelesaikan setiap tugas individu.

3. Sekarang, mari kita atur

matriks estimasi tiga titik, seperti yang didefinisikan dalam tabel di Memperkirakan waktu tugas bagian: Waktu Tugas=

[[3,5,8],

[2,4,7],

[3,5,9],

[4,6,10],

[3,5,9],

[2,6,8]]

Matriks ini berisi tiga kali untuk setiap baris perwakilan dari enam tugas: optimis, lebih mungkin, dan pesimis. Pada titik ini, kita harus menetapkan bentuk distribusi waktu yang ingin kita adopsi.

- Menjelajahi distribusi segitiga

Saat mengembangkan model simulasi, perlu untuk memperkenalkan peristiwa probabilistik. Seringkali, proses simulasi dimulai sebelum Anda memiliki cukup informasi tentang perilaku data input. Ini memaksa kita untuk memutuskan distribusi. Di antara yang berlaku untuk data yang tidak lengkap adalah distribusi segitiga. Distribusi segitiga digunakan ketika asumsi dapat dibuat pada nilai minimum dan maksimum dan pada nilai modal.

Tujuan kami adalah untuk memodelkan waktu setiap tugas individu menggunakan variabel acak dengan distribusi seragam dalam interval (0, 1). Jika kita menunjukkan variabel acak ini dengan trand, maka distribusi segitiga memungkinkan kita untuk mengevaluasi probabilitas bahwa tugas berakhir pada waktu itu didistribusikan. Dalam distribusi segitiga, kami telah mengidentifikasi dua segitiga yang memiliki absis yang sama untuk x = c. Nilai ini bertindak sebagai pemisah antara dua nilai yang diasumsikan oleh distribusi.

Dengan menganalisis statistik ini, kami telah mengkonfirmasi bahwa perkiraan waktu berada di antara nilai batas yang ditentukan oleh masalah: optimis dan pesimis. Faktanya, waktu minimum dan maksimum sangat dekat dengan nilai-nilai ini. Selanjutnya, kita dapat melihat bahwa standar deviasi sangat dekat dengan unit, sehingga menghasilkan nilai yang lebih banyak dan rinci.

- Ringkasan

Dalam bab ini, kami membahas beberapa aplikasi simulasi model praktis berdasarkan model terkait manajemen proyek. Untuk memulai, kami melihat elemen penting dari manajemen proyek dan bagaimana faktor-faktor ini dapat disimulasikan untuk diambil informasi berguna. Selanjutnya, kami menangani masalah mengelola hutan kecil untuk perdagangan kayu. Kami memperlakukan masalah sebagai MDP, meringkas karakteristik dasar dari proses ini dan kemudian beralih ke diskusi praktis tentang mereka. Kami mendefinisikan elemen masalah dan kemudian kami melihat bagaimana menggunakan evaluasi kebijakan dan algoritma perbaikan kebijakan untuk mendapatkan kebijakan pengelolaan hutan yang optimal. Masalah ini telah diatasi menggunakan kotak alat MDP paket, yang tersedia dari Python. Selanjutnya, kami membahas masalah mengevaluasi waktu pelaksanaan proyek menggunakan simulasi Monte Carlo. Untuk memulai, kami mendefinisikan diagram eksekusi tugas dengan menentukan tugas mana yang dilakukan secara seri dan mana yang dilakukan secara paralel. Jadi, kami memperkenalkan waktu setiap tugas melalui estimasi tiga poin. Setelah ini, kami melihat bagaimana memodelkan waktu pelaksanaan proyek dengan distribusi segitiga menggunakan evaluasi acak dari setiap fase. Akhirnya, kami melakukan 10.000 penilaian dari keseluruhan waktu proyek. Dalam bab berikutnya, kami akan merangkum proses pemodelan simulasi yang telah kami lihat di bab sebelumnya. Kemudian, kita akan mengeksplorasi aplikasi pemodelan simulasi utama yang digunakan dalam kehidupan nyata. Akhirnya, kami akan menemukan tantangan masa depan mengenai pemodelan simulasi