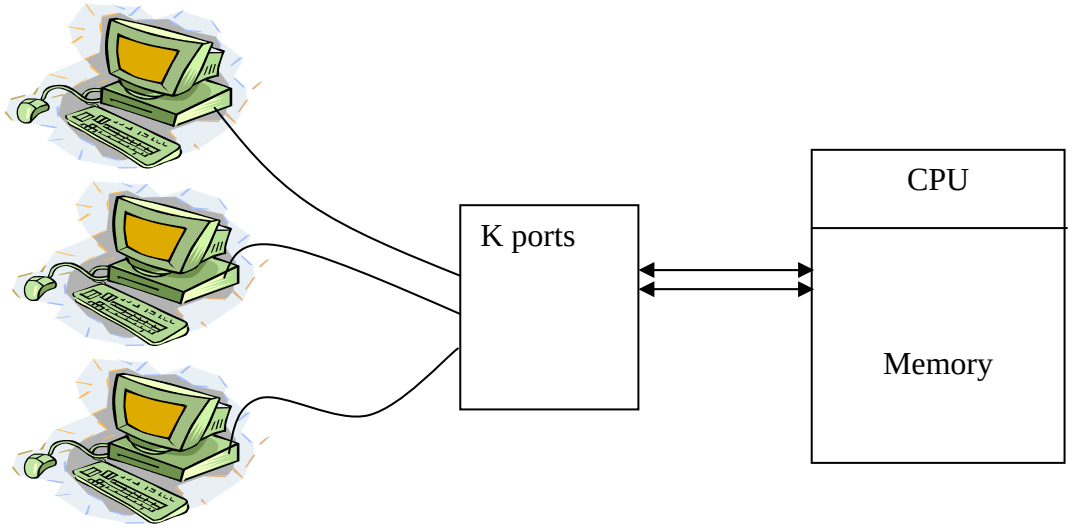
Sistem Komputer Time-Shared

Perhatikan sistem komputer time-shared, dimana pemakai dihubungkan ke sistem melalui jaringan telepon. Hanya ada sedikit jumlah port untuk koneksi seperti ini, dan ketika semua port digunakan saat panggilan masuk, maka pengguna akan menerima signal sibuk dan harus mencoba membuat koneksi di lain waktu. Sekali koneksi tersambung, port tidak akan dapat digunakan lagi oleh pemakai lain sampai pemakai saat itu memutus hubungan dengan menutup telepon. Skematik sistem ditunjukkan oleh Gambar 1.



Gambar 5. Sistem komputer time-shared

Dengan maksud menyederhanakan permasalahan, kita akan mengasumsikan bahwa pemakai berusaha menghubungi komputer pada waktu acak sepanjang hari dengan laju rata-rata panggilan 35 per jam. Data historis rata-rata lamanya waktu, termasuk perhitungan dan bukan perhitungan, 25 menit. Meskipun saat ini ada 14 port, tidak jarang pemakai menemukan bahwa semua port sibuk saat mereka melakukan pemanggilan. Permintaan memperbesar memori CPU dan menambah jumlah port dan kecepatan transmisi sudah sering diajukan untuk meningkatkan level pelayanan.

Port relatif mahal karena biaya perangkat keras dan biaya pelayanan telepon per bulan. Harga perangkat keras untuk setiap port sekitar Rp10,000,000,- dan biaya pelayanan telepon serta perawatan setiap port per bulan sebesar Rp250,000,-. Sistem komputer hanya dapat mendukung 32 port dan kapasitas perhitungan sistem terbatas.

Dipercaya bahwa dengan meningkatkan laju transmisi antara sistem komputer dengan pemakai yang memanggil dari 120 kpd (karakter per detik) menjadi 960 kpd, lama sesi dapat dikurangi. Diyakini bahwa lama sesi rata-rata dapat dikurangi tiga menit menggunakan laju transmisi lebih tinggi. Peningkatan dari 120 ke 960 kpd membutuhkan biaya Rp4,000,000,- untuk setiap 100 terminal yang dapat dikoneksikan ke sistem.

Studi pendahuluan menunjukkan bahwa kinerja sistem akan diperbaiki jika memori CPU diperbesar. Pengaruh peningkatan seperti itu akan menjadi sesi paling sensitif dimana pemakai akan terhubung ke sistem dalam jangka waktu singkat. Memori CPU saat ini 1MB dan dapat diperbesar ke 2 MB atau 3 MB. Tabel 1 menunjukkan biaya dan pengurangan waktu koneksi untuk memori 1MB, 2 MB dan 3 MB.

Dengan maksud mengevaluasi kegunaan penambahan port, peningkatan kecepatan transmisi, atau menambah memori, disarankan untuk mempelajari kinerja sistem dan biaya yang berhubungan dengan bantuan model analitik, model simulasi dan keduanya.

Tabel 1. Biaya dan pengurangan waktu koneksi dengan beberapa alternatif memori

Waktu perhitungan rata-rata	Biaya (dolar)	peningkatan
10	0	Konfigurasi saat ini
8	20,000	Memori 2M-Byte
7	30,000	Memori 3M-byte

Formulasi Masalah

Untuk memformulasikan masalah, kita perlu menjawab pertanyaan :

1. Apa yang kita harapkan untuk dipelajari dengan membangun model simulasi kasus ini?
2. Informasi apa yang kita inginkan disediakan simulasi?

Pertanyaan-pertanyaan ini bisa dikembangkan lagi dan tidak selalu diungkapkan secara eksplisit.

Kita dapat menggunakan model simulasi untuk memprediksi kinerja sistem sebagai parameter perubahan sistem atau lebih disukai, model simulasi dapat digunakan untuk mengarahkan pengoptimuman beberapa tujuan yang dibatasi oleh sumber daya terbatas.

Untuk kasus di atas, dengan mencoba konfigurasi sistem berbeda, kita dapat mengamati ukuran kinerja sistem. Pertanyaan yang akan dijawab bisa dalam bentuk:

1. Berapa probabilitas keterhubungan ke sistem sebagai fungsi jumlah koneksi terminal (port)? atau
2. Berapa jumlah rata-rata port sibuk, sebagai fungsi memori, koneksi terminal dan kecepatan transmisi? atau
3. berapa level kepuasan pemakai sebagai fungsi peningkatan sumber daya?

Berbagai pertanyaan lain dapat dibentuk, setiap pertanyaan membantu analis untuk fokus pada tujuan pemodelan simulasi.

Alternatifnya, kita dapat menyatakan fungsi objektif yang akan dioptimalkan bersamaan dengan pembatas yang harus dipenuhi untuk mendapatkan solusi layak. Sebagai contoh, kita mungkin memilih dari salah bentuk di bawah ini untuk tujuan dan pembatas :

- maksimumkan (kepuasan pengguna)
  terhadap : biaya total pengeluaran < C_0

atau

- minimumkan (total pengeluaran)
  terhadap : kepuasan pemakai > S_0

atau

- minimumkan (lama rat-rata sesi per pemakai)
  terhadap : biaya total < C_0

atau

- minimumkan (total biaya)
  termasuk biaya pemakai dan biaya sumber daya

Beberapa dari tujuan dan pembatas ini perlu diklarifikasi, yang merupakan langkah penting dalam formulasi masalah. Tanpa mengidentifikasi dengan jelas dan lebih dini pertanyaan yang harus dijawab dari model simulasi, proses pemodelan simulasi akan berakhir dengan sendirinya dan analisis akan sangat mudah kehilangan wawasan tujuan akhir dari pemodelan. Detil dan level kompleksitas model harus merefleksikan penggunaan akhir

model. Model tidak perlu lebih kompleks atau lebih detil dari pertanyaan yang harus dijawab yang dibuat di awal analisis.

Kita akan definisikan variabel dan paramater sistem.

Variabel Eksogenus – tidak dapat dikontrol

Simbol	penjelasan
k_0	Jumlah port saat ini
λ (t)	Rata-rata laju kedatangan pada waktu t
C_T	Biaya per bulan setiap penambahan satu line telepon
C_H	Biaya perangkat keras untuk setiap penambahan port
C_U	Biaya upgrading semua pemakai ke 960 kpd
L	Umur ekspekstasi perangkat keras

Variabel Eksogenus –dapat dikontrol (variabel keputusan)

Simbol	penjelasan
K_l	Jumlah port tambahan
C_R	Investasi sumberdaya komputer tambahan
K	Jumlah total port = $k_0 + k_1$
U	Biaya upgrading semua pemakai sampai 960 kpd,C_U [0, jk pemakai tdk diupgrade]
E(T)	Ekspektasi jumlah pemakai terhubung per sesi

Variabel Endogenus – variabel status

N(t)	Jumlah pemakai terkoneksi ke sistem pada waktu t

Variabel Endogenus – ukuran kinerja

Simbol	penjelasan
TC	Biaya total per tahun = $C_R + k_1C_H + U/L+12k_1C_T$
P_K	Peluang pemakai berusaha koneksi dan menemukan semua port terpakai
P_C	Peluang pemakai berusaha koneksi dan menemukan tidak semua port terpakai (juga disebut level pelayanan) = $1-P_K$

Karena tujuan pengembangan model adalah untuk mendukung pengambilan keputusan, maka selanjutnya kita harus mempertimbangkan kriteria pengambilan keputusan termasuk

tujuan dan kendala yang dihadapi. Berbagai kriteria keputusan ada, tapi jika kita membatasi pilihan pada pengukuran biaya dan level pelayanan, kita dapat mempertimbangkan kriteria seperti berikut:

1. minimumkan TC
2. minimumkan TC dengan kendala $P_K < P_0$
3. minimumkan P_K dengan kendala $TC < TC_0$
4. minimumkan biaya total sistem (termasuk nilai waktu pemakai, biaya perangkat keras dan telepon).

Kriteria terakhir ini adalah yang umum, tapi kita akan menghadapi model yang lebih kompleks. Kompleksitasnya ada karena kita harus mengukur nilai waktu pemakai. Untuk selanjutnya kita akan menggunakan kriteria keputusan no. 2. Nilai P_0 dibuat 0.02 dan level pelayanan paling tidak 0.98.

Model Analitik

Sebelum mengembangkan model simulasi, pertama-tama harus selalu dipertimbangkan apakah model analitik dapat digunakan. Pengembangan dan penggunaan model analitik lebih murah dibandingkan model simulasi, dan bahkan jika model analitik sempurna tidak dapat dikembangkan, model analitik pendekatan akan sangat berguna untuk menganalisis sistem.

Model analitik untuk kasus komputer time-shared di atas dapat dikembangkan dengan membuat beberapa asumsi terlebih dahulu :

- Waktu koneksi berdistribusi secara eksponensial dengan rata-rata konstan.
- Waktu antara dua panggilan berdistribusi eksponensial dengan rata-rata konstan.
- Pengaruh jumlah awal pemakai yang terhubung ke sistem pada permulaan hari menghilang dengan cepat
- Distribusi waktu di antara panggilan tidak berubah (paling tidak secara mendasar) ketika semua port terpakai.

Misalkan : P_i = peluang secara tepat sejumlah i pemakai terhubung, i = 0, 1, ..., K

λ = 1/waktu rata-rata antara kedatangan

μ = 1/rata-rata waktu terhubung

maka,

$$P_0 = \left[\frac{\sum_{i=0}^{K} [\lambda/\mu]^i}{i!} \right]^{-1} \quad (1) \qquad \text{dan} \quad P_1 = \frac{P_0 \left[\lambda/\mu \right]^i}{i} \quad (2) \qquad 1 \le i \le K$$

dalam model ini, P_K adalah probabilitas bahwa semua port digunakan atau probabilitas bahwa pengguna yang mau masuk ke sistem tidak dapat terhubung. Menggabungkan kedua persamaan di atas akan diperoleh:

$$P_K = \frac{[\lambda/\mu]^K / K!}{\sum_{i=0}^{K} [\lambda/\mu]^i} \quad (3)$$

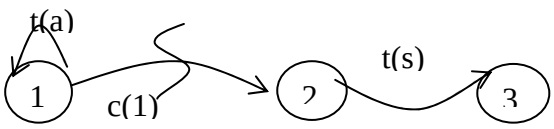
ada 108 pilihan alternatif berbeda (2 kecepatan transmisi, 3 ukuran memori dan sampai 18 ports). Dari sudut pandang model antrian, setiap alternatif pilihan dapat direpresentasikan dengan waktu koneksi rata-rata ($\frac{1}{\mu}$) dan jumlah port K. menggunakan persamaan (3), level pelayanan P_C dapat dihitung untuk setiap alternatif. Tabel 1 menunjukkan waktu terhubung rata-rata untuk himpunan bagian pilihan kecepatan transmisi dan 3 memori.

Tabel 1. Waktu terhubung rata-rata

Kecepatan transmisi (kpd)	memori		
	1M byte	2M byte	3M byte
30	25	22	20
120	22	19	17

Model Simulasi

Diagram (graf) kejadian untuk sistem di atas adalah:



Gambar 7. Graf kejadian untuk sistem komputer time-shared

Variabel status :

N(t) = jumlah port sibuk

Kejadian :

1. pemakai berusaha koneksi ke sistem
2. pemakai terhubung dan sesi mulai
3. pemakai menyudahi sesi

Kondisi :

C(1) : n(t) < K

Penundaan :

t(a) = waktu sampai pemakai berikutnya berusaha masuk

t(s) = jangka waktu pemakai terhubung dengan sistem

Variabel model :

Variabel Eksogenus – tidak dapat dikontrol

Simbol	penjelasan
MEAN_CALL_TIME	Ekspektasi waktu antara dua panggilan

Variabel Eksogenus –dapat dikontrol (variabel keputusan)

Simbol	penjelasan

K	Jumlah port
MEAN_CONNECT_TIME	Ekspektasi lama sesi pemakai

Variabel Endogenus – variabel status

N	Jumlah port yang sedang digunakan
T_NEXT_CALL	Waktu pemanggilan berikutnya
T_CALL_END (i)	Waktu akhir koneksi port ke-i
PORT_STATUS (i)	Mengindikasikan apakah port sibuk atau menganggur

Variabel Endogenus – ukuran kinerja

Simbol	penjelasan
CUM_CONNECT_TIME	Waktu kumulatif pemakai terhubung ke sistem
N_CALLS	Jumlah total pemakai memanggil sistem
N_CONNECT	Jumlah total panggilan yang terhubung
N_FAIL_CONNECT	Jumlah total panggilan yang gagal terhubung
PROB_CONNECT	N_CONNECT/N_CALLS
PROB_FAIL_CONNECT	N_FAIL_CONNECT/N_CALLS
AVE_NUM_USER	CUM_CONNECT_TIME/T_FINAL

Parameter simulasi

T	Waktu sekarang
SEED1	Pembangkitan bilangan acak pemanggilan ke sistem
SEED2	Pembangkitan bilangan acak untuk waktu koneksi
T_FINAL	Lama simulasi

Function dan Subroutines

F_NEXT_CALL	Hitung waktu sampai ada panggilan masuk berikutnya
F_CONNECT_TIME	Hitung lamanya sesi pemakai
FREE_PORT	Tempatkan port yang tersedia
NEXT_CALL_END	Tempatkan port yang sesinya selesai terlebih dahulu
RANDOM	Kembalikan distribusi uniform bilangan acak [0,1]

Logika pemrograman menggunakan metode penjadwalan kejadian waktu berlanjut ditunjukkan Gambar 8.

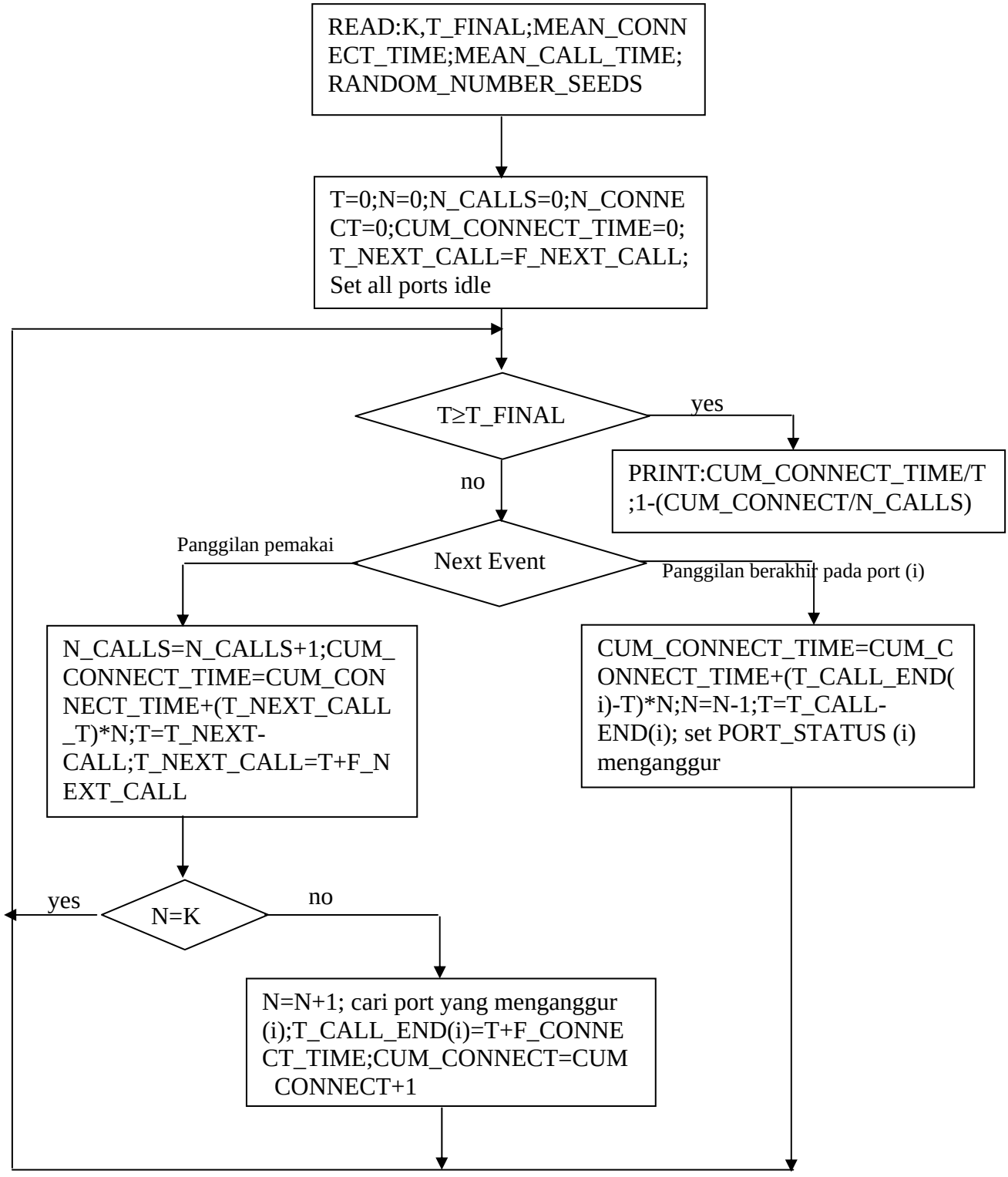
Pertimbangan Pemrograman dan Struktur Data

Usaha perhitungan dalam simulasi pada umumnya melibatkan banyak kejadian, seperti identifikasi kejadian paling dekat berikutnya dan penjadwalan kejadian berikutnya. Program simulasi akan jauh lebih efisien jika kejadian yang dieksekusi diurutkan berdasarkan waktu kejadian berikutnya, daripada disimpan tidak terurut. Daripada mencari sekumpulan kejadian berikutnya secara fisik di dalam komputer, akan lebih baik melakukan pencarian secara logika. Menggunakan metode penstruktur data, efisiensi simulasi komputer dapat ditingkatkan secara signifikan, khususnya saat simulasi mempunyai banyak kejadian terjadwal berikutnya.

Penambahan Waktu dalam Model Simulasi

Penambahan waktu dalam model simulasi harus diperhatikan. Dua sisi sering harus dihadapi dalam penambahan waktu model simulasi. Pertama, dalam banyak kasus, waktu (T) ditambah dengan selisih waktu (T+delta) tanpa ada kejadian selama interval waktu (T, T+delta), menyebabkan pencarian kejadian selama interval waktu menjadi sia-sia. Kedua, akan terjadi kehilangan akurasi jika kejadian dimungkinkan terjadi hanya pada waktu delta dan kelipatannya. Jika delta dinaikkan, permasalahan yang pertama akan dikurangi tetapi akan menyebabkan penurunan akurasi. Sebaliknya, ukuran delta dikurangi. Keakuratan akan lebih baik, tetapi frekuensi pembaharuan (dan pencarian berurutan kejadian) meningkat, menyebabkan komputasi model tidak efisien.

Metode alternatif untuk memperbaharui waktu (T) adalah dengan memeriksa semua kejadian di masa mendatang secara berurut dan memperharui T dengan waktu kejadian paling dulu terjadi. Dengan melangkah ke kejadian terjadwal terdekat, baik permasalahan pembaharuan yang tidak penting dan akurasi dapat dihilangkan, tapi logikanya akan lebih kompleks.pada kebanyakan simulasi kejadian-diskrit. Pelacakan kejadian adalah metode yang paling disukai dalam penambahan waktu dan merupakan metode paling banyak dilakukand alam bahasa simulasi.



Gambar 8. Logika pemrogaman time-shared computer

Manajemen Kejadian : Daftar Terhubung

Kejadian dalam sistem perlu disimpan sehingga mudah dicari, karena itu manajemen kejadian harus dilakukan dengan baik. Struktur data yang diperlukan untuk tujuan ini dinamakan daftar terhubung (Linked list). Dalam model simulasi, kita harus memiliki metode penambahan dan penghapusan elemen ke/dari daftar. Jika elemen tersortir pertama dihilangkan, kepala (***head***) elemen baru akan menjadi elemen yang akan ditunjukkan oleh kepala saat ini. Jika elemen baru ditambahkan ke daftar, penunjuk daftar harus dirubah untuk memasukkan elemen baru ke dalam urutan terhubung. Penyisipan elemen baru ke dalam daftar memerlukan tiga operasi:

1. Tempatkan elemen baru di lokasi yang sesuai.
2. Atur ulang penunjuk elemen sebelumnya sehingga penunjuk sekarang menunjuk elemen baru.
3. atur penunjuk ke elemen baru sehingga penunjuk mengarah ke elemen yang pendahulunya tadi ditunjuk.

Logika simulasi perlu dirubah untuk memungkinkan metode baru dalam penempatan kejadian yang paling dekat (pemilihan elemen pada kepala daftar) dan menyisipkan kejadian mendatang ke dalam daftar terhubung.

Implementasi Antrian dalam Model Simulasi

Jika kita mensimulasikan sistem dimana kedatangan dapat mengantri, menunggu ketersediaan fasilitas, mungkin akan diinginkan mengumpulkan informasi tentang pengalaman kedatangan, seperti rata-rata, ragam dan kisaran waktu kedatangan. Ada beberapa metode untuk menangani informasi kedatangan:

1. array tidak tersortir yang memuat prioritas pada setiap kedatangan.
2. daftar terhubung dengan kedatangan tersortir sesuai dengan prioritas.
3. array kedatangan tersortir sesuai dengan prioritas.

Metode pertama mudah diimplementasikan tapi tidak efisien jika arraynya panjang, karena pencarian harus dilakukan terhadap semua array dengan tujuan untuk mencari kedatangan

berikutnya yang akan dilayani. Pendekatan yang kedua sama, menyimpan daftar untuk kejadian mendatang dan disortir sesuai dengan daftar terhubung, tetapi informasi kedatangan dihubungkan sesuai dengan prioritas setiap kedatangan. Kebanyakan simulasi menggunakan metode ini untuk antrian.

Manajemen Memori dalam model Simulasi

Dalam kebanyakan model simulasi adalah sulit memprediksi jumlah maksimum kejadian yang akan datang yang akan dijadwalkan, panjang maksimum setiap antrian atau jumlah maksimum entitas (seperti pelanggan) yang akan hadir. Salah satu solusi untuk permasalahan ini adalah mengalokasikan memori besar untuk daftar, antrian dan entitas dengan kemungkinan kecil memori kurang dari yang dibutuhkan dalam penjalanan simulasi. Tetapi tentu saja hal ini tidak diinginkan. Alternatif lain adalah mengalokasikan memori sesuai dengan kebutuhan sewaktu simulasi dijalankan. Ketika satu elemen ditambahkan ke daftar, memori untuk itu dialokasikan dari memori penyimpanan umum yang tidak digunakan. Dengan cara sama, ketika memori tidak dibutuhkan lagi, akan dikembalikan ke memori penyimpanan umum untuk tujuan penggunaan di masa mendatang. Dengan cara seperti ini, memori tidak dikhususkan untuk penggunaan tertentu, tetapi dapat dialokasikan sesuai dengan kebutuhan.

Struktur data yang memungkinkan pengalokasian dan penarikan kembali adalah stack. Stack adalah lokasi memori terhubung bersama dimana urutan yang terakhir masuk akan menjadi yang pertama keluar. Awalnya, stack memuat sejumlah L elemen dimana elemen 2 menunjuk ke elemen 1, elemen 3 menunjuk ke elemen 2, dst. Variabel yang disebut penunjuk stack diatur ke elemen terakhir dalam daftar. Awalnya, penunjuk stack diatur ke L, atau menunjuk ke elemen terakhir dalam daftar yang pada gilirannya akan menunjuk ke elemen L-1, dst.

Menghilangkan elemen dari stack biasanya disebut dengan ***popping stack*** dan penambahan elemen ke stack umumnya disebut dengan ***pushing stack***. Ketika sebuah elemen dibutuhkan untuk penjadwalan kejadian masa mendatang, elemen L dipilih dan penunjuk stack diatur ke nilai elemen itu. Untuk mengembalikan elemen ke stack setelah kejadian dijalankan, kita hanya membutuhkan himpunan nilai elemen dikembalikan ke nilai penunjuk stack dan kemudian atur penunjuk stack ke indeks elemen yang sedang dikembalikan.