Prinsip Umum Simulasi

Muhammad Luthfi Shahab

Departemen Matematika

Institut Teknologi Sepuluh Nopember

Apa yang Dipelajari pada Bab Ini?

- Bab ini membahas kerangka umum pemodelan sistem kompleks menggunakan **simulasi peristiwa diskrit (discrete-event simulation)**.
- Simulasi peristiwa diskrit dibangun atas elemen dasar: **entitas** dan **atribut**, **aktivitas**, serta **peristiwa**.
- Sistem dimodelkan berdasarkan:
  - **Keadaan sistem (state)** pada setiap waktu,
  - **Entitas** yang melalui sistem dan entitas yang mewakili sumber daya,
  - **Aktivitas** dan **peristiwa** yang mengubah keadaan sistem.
- Fokus bab ini adalah **konsep dan metodologi simulasi** yang mendasari berbagai bahasa dan perangkat lunak simulasi (atau **paket simulasi**).
- Konsep-konsep yang dibahas bersifat **fundamental dan umum**, tidak terkait dengan satu paket simulasi tertentu.
- Setiap paket simulasi mungkin menggunakan istilah berbeda (misalnya mesin, conveyor, kendaraan) untuk memudahkan pemodelan sesuai bidang aplikasi.
- **Bagian 1** membahas prinsip umum simulasi peristiwa diskrit:
  - Konsep dasar,
  - Algoritma penjadwalan peristiwa/kemajuan waktu,
  - Tiga sudut pandang utama: **event scheduling**, **process interaction**, **activity scanning**.
- **Bagian 2** memperkenalkan **list processing**, salah satu metodologi penting yang digunakan dalam perangkat lunak simulasi peristiwa diskrit (buku Banks).

Konsep dalam Simulasi Kejadian Diskrit

Konsep dalam Simulasi Kejadian Diskrit

System A collection of entities (e.g., people and machines) that interact together over time to accomplish one or more goals.

Model An abstract representation of a system, usually containing structural, logical, or mathematical relationships that describe a system in terms of state, entities and their attributes, sets, processes, events, activities, and delays.

System state A collection of variables that contain all the information necessary to describe the system at any time.

Entity Any object or component in the system that requires explicit representation in the model (e.g., a server, a customer, a machine).

Attributes The properties of a given entity (e.g., the priority of a waiting customer, the routing of a job through a job shop).

List A collection of (permanently or temporarily) associated entities, ordered in some logical fashion (such as all customers currently in a waiting line, ordered by "first come, first served," or by priority).

Konsep dalam Simulasi Kejadian Diskrit

Event An instantaneous occurrence that changes the state of a system (such as an arrival of a new customer).

Event notice A record of an event to occur at the current or some future time, along with any associated data necessary to execute the event; at a minimum, the record includes the event type and the event time.

Event list A list of event notices for future events, ordered by time of occurrence; also known as the future event list (FEL).

Activity A duration of time of specified length (e.g., a service time or interarrival time), which is known when it begins (although it may be defined in terms of a statistical distribution).

Delay A duration of time of unspecified indefinite length, which is not known until it ends (e.g., a customer's delay in a last-in–first-out waiting line which, when it begins, depends on future arrivals).

Clock A variable representing simulated time, called CLOCK in the examples to follow.

Konsep dalam Simulasi Kejadian Diskrit

- Terminologi paket simulasi berbeda-beda:
  - Daftar (**lists**) kadang disebut **sets**, **queues**, atau **chains**.
  - Daftar digunakan untuk menyimpan **entitas** dan **event notices**.
  - Entitas dalam daftar diurutkan dengan aturan tertentu:
    - **FIFO (first-in–first-out)**
    - **LIFO (last-in–first-out)**
    - Berdasarkan atribut entitas, seperti **prioritas** atau **due date**.
  - **Future Event List (FEL)** selalu diurutkan berdasarkan **waktu kejadian** pada event notice.
- **Aktivitas (activity)** mewakili **durasi waktu proses**, misalnya waktu layanan, waktu antar kedatangan, dll.
  - Durasi aktivitas dapat ditentukan dengan:
    - **Deterministik** → contoh: selalu 5 menit.
    - **Statistik** → contoh: random dari {2, 5, 7} dengan peluang sama.
    - **Fungsi sistem/atribut entitas** → contoh: waktu muat kapal = f(kapasitas kargo, laju muat).
  - Durasi aktivitas dapat dihitung saat aktivitas dimulai dan **tidak dipengaruhi oleh kejadian lain**, kecuali jika ada logika khusus untuk membatalkan/menunda.
  - Saat aktivitas dimulai → dibuat **event notice** dengan waktu = **waktu mulai + durasi aktivitas**.

Konsep dalam Simulasi Kejadian Diskrit

- **Delay (penundaan)** berbeda dengan aktivitas:
  - Durasi **tidak ditentukan sebelumnya oleh modeler**, melainkan bergantung pada **kondisi sistem**.
  - Durasi delay biasanya **diukur** dan menjadi salah satu output simulasi.
  - Delay berakhir ketika **syarat logis terpenuhi** atau **kejadian tertentu terjadi**.
  - Contoh: waktu tunggu pelanggan dalam antrean bergantung pada panjang antrean dan ketersediaan server.
  - Delay = **conditional wait**, sedangkan Activity = **unconditional wait**.
- **Perbedaan pengelolaan**:
  - **Aktivitas selesai** → menghasilkan **event (primary event)** yang dimasukkan ke FEL.
  - **Delay selesai** → entitas ditempatkan pada daftar (misalnya antrean) sampai kondisi sistem memungkinkan diproses.
  - Delay completion = **conditional/secondary event**, tidak direpresentasikan sebagai event notice dan tidak muncul di FEL.
- **Sistem bersifat dinamis**:
  - State sistem, atribut entitas, jumlah entitas aktif, isi daftar, aktivitas dan delay → semuanya **berubah seiring waktu**.
  - Waktu simulasi dilacak dengan variabel **CLOCK**.

Example 1: Call Center, Revisited

Consider the Able–Baker call center system of Example 6 from the Simulation Examples in a Spreadsheet chapter. A discrete-event model has the following components:

System state

$L_Q(t)$, the number of callers waiting to be served at time t;
$L_A(t)$, 0 or 1 to indicate Able as being idle or busy at time t;
$L_B(t)$, 0 or 1 to indicate Baker as being idle or busy at time t.

Entities Neither the callers nor the servers need to be explicitly represented, except in terms of the state variables, unless certain caller averages are desired (compare Examples 4 and 5).

Events Kejadian

Arrival event;
Service completion by Able;
Service completion by Baker.

Activities Aktifitas

Interarrival time, defined in Table 12 from the Simulation Examples in a Spreadsheet chapter;
Service time by Able, defined in Table 13 from the Simulation Examples in a Spreadsheet chapter;
Service time by Baker, defined in Table 14 from the Simulation Examples in a Spreadsheet chapter.

Delay

A caller's wait in queue until Able or Baker becomes free.

Event Scheduling / Time Advance Algorithm

Contoh Simulasi dengan Tipe Kejadian

CLOCK	System state	Entities and attributes	Set 1	Set 2	. . .	Future event list (FEL)	Cumulative statistics and counters
t	$(x, y, z, \ldots)$					$(3, t_1)$ – Type 3 event to occur at time t_1 $(1, t_2)$ – Type 1 event to occur at time t_2	

Figure 1 Prototype system snapshot at simulation time t.

Jika kejadian di t_1 selesai dan ada kejadian t^*

If $t^* < t_2$, place event 4 at the top of the FEL.

If $t_2 \leq t^* < t_3$, place event 4 second on the list.

If $t_3 \leq t^* < t_4$, place event 4 third on the list.

$\vdots$

If $t_n \leq t^*$, place event 4 last on the list.

Old system snapshot at time t

CLOCK	System state	...	Future event list	...
t	(5, 1, 6)		$(3, t_1)$ – Type 3 event to occur at time t_1 $(1, t_2)$ – Type 1 event to occur at time t_2 $(1, t_3)$ – Type 1 event to occur at time t_3 . . . . $(2, t_n)$ – Type 2 event to occur at time t_n	

Event-scheduling/time-advance algorithm

Step 1. Remove the event notice for the imminent event (event 3, time t_1) from FEL.

Step 2. Advance CLOCK to imminent event time (i.e., advance CLOCK from t to t_1).

Step 3. Execute imminent event: update system state, change entity attributes, and set membership as needed.

Step 4. Generate future events (if necessary) and place their event notices on FEL, ranked by event time. (Example: Event 4 to occur at time t^*, where $t_2 < t^* < t_3$.)

Step 5. Update cumulative statistics and counters.

New system snapshot at time t_1

CLOCK	System state	...	Future event list	...
t_1	(5, 1, 5)		$(1, t_2)$– Type 1 event to occur at time t_2 $(4, t^*)$– Type 4 event to occur at time t^* $(1, t_3)$– Type 1 event to occur at time t_3 . . . . $(2, t_n)$– Type 2 event to occur at time t_n	

Figure 2 Advancing simulation time and updating system image.

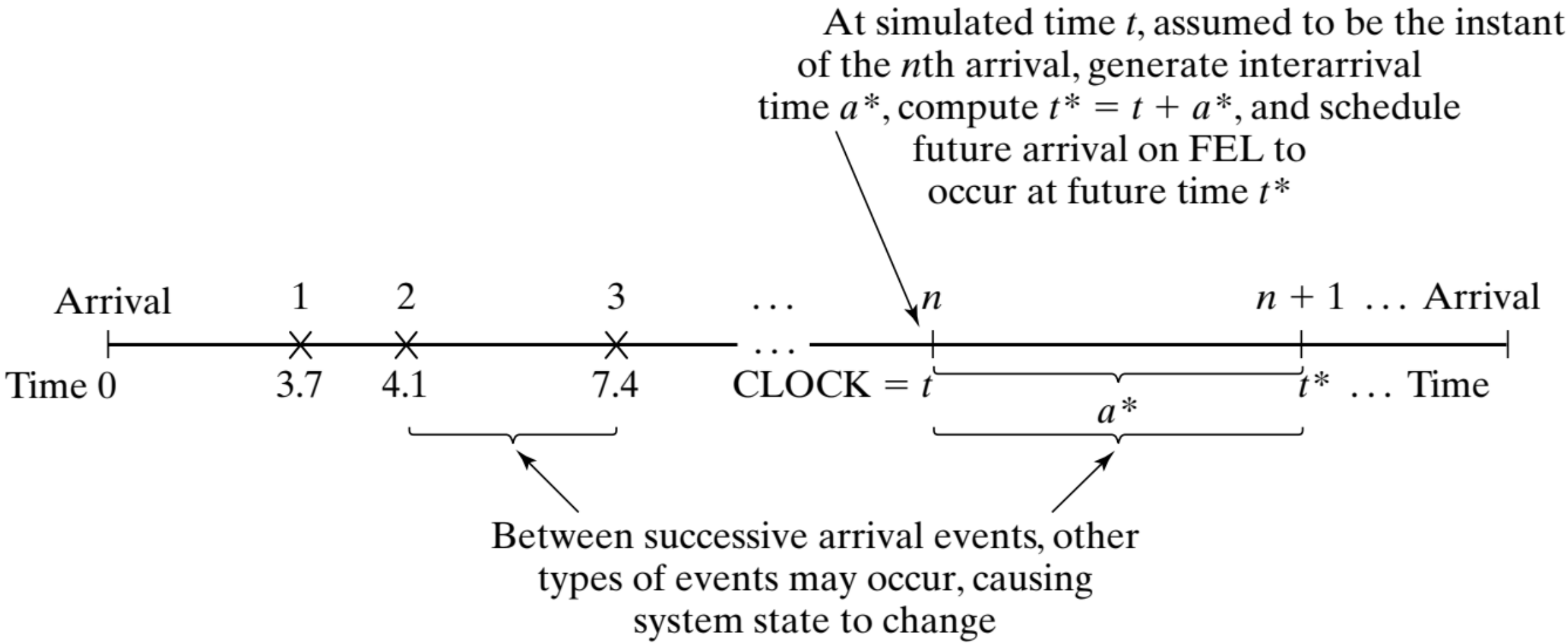

Figure 3 Generation of an external arrival stream by bootstrapping.

World Views

World Views dalam Simulasi

- **World view** adalah orientasi atau cara pandang modeler dalam membangun model simulasi.
- Tiga world view utama:
  - **Event-scheduling**
  - **Process-interaction**
  - **Activity-scanning**
- Penting dipahami meskipun paket simulasi tertentu tidak mendukung semuanya, karena memberi alternatif dalam pemodelan.

Event-Scheduling World View

- Fokus pada **event** dan pengaruhnya terhadap **state** sistem.
- Implementasi: simulasi manual (contoh dibahas di Section 1.3).
- Menggunakan **variable time advance** → waktu simulasi maju ke event berikutnya pada **Future Event List (FEL)**.

Process-Interaction World View

- Fokus pada **proses**, yaitu **daur hidup entitas** saat melewati sistem.
- Proses terdiri dari event, aktivitas, delay, serta permintaan resource.
- Karakteristik:
  - Entitas dapat **menunggu dalam antrean** jika resource terbatas.
  - Banyak proses aktif sekaligus → interaksi antar proses bisa kompleks.
- Kelebihan: intuitif dan didukung paket simulasi dengan **block atau network constructs** tingkat tinggi.
- Secara internal: tetap menggunakan **event scheduling & list processing**, namun tersembunyi dari modeler.

Activity-Scanning World View

- Fokus pada **aktivitas** dan **kondisi** yang memungkinkan aktivitas dimulai.
- Menggunakan **fixed time increment** → pada setiap langkah waktu, semua kondisi aktivitas dicek.
- Kelebihan: sederhana, modular, mudah dipelihara & dipahami.
- Kekurangan: runtime lambat karena **scanning berulang**.

Three-Phase Approach (variasi Activity-Scanning)

- Menggabungkan **event scheduling** dengan **activity scanning**.
- Memungkinkan **variable time advance** dan menghindari scanning yang tidak perlu.
- Konsep:
  - Event dianggap sebagai aktivitas dengan durasi nol.
  - Aktivitas dibagi dua jenis:
    - **B-type**: aktivitas pasti terjadi (event primer, unconditional).
    - **C-type**: aktivitas/event bersyarat (conditional).
- **FEL hanya berisi B-type events**.
- Simulasi berjalan dalam **3 fase** berulang:
  - **Phase A**: Hapus event terdekat dari FEL, majukan CLOCK ke waktu event.
  - **Phase B**: Eksekusi semua B-type event (misalnya membebaskan resource).
  - **Phase C**: Scan kondisi untuk C-type activities; jalankan jika syarat terpenuhi; ulangi sampai tidak ada aktivitas baru yang bisa mulai.
- Kelebihan Three-Phase Approach:
  - Lebih efisien dibanding pure activity-scanning.
  - Baik untuk masalah resource kompleks → menjamin semua resource yang bebas pada satu waktu simulasi dilepas lebih dulu sebelum dialokasikan kembali.

Example 2: Call Center, Back Again

For the Call Center in Example 1, if we were to adopt the three-phase approach to the activity-scanning world view, the conditions for beginning each activity in Phase C would be as follows:

Activity	Condition
Service time by Able	A caller is in queue and Able is idle
Service time by Baker	A caller is in queue, Baker is idle and Able is busy

If it were to take the process-interaction world view, we would view the model from the viewpoint of a caller and its "life cycle." Considering a life cycle as beginning upon arrival, a customer process is pictured in Figure 4.

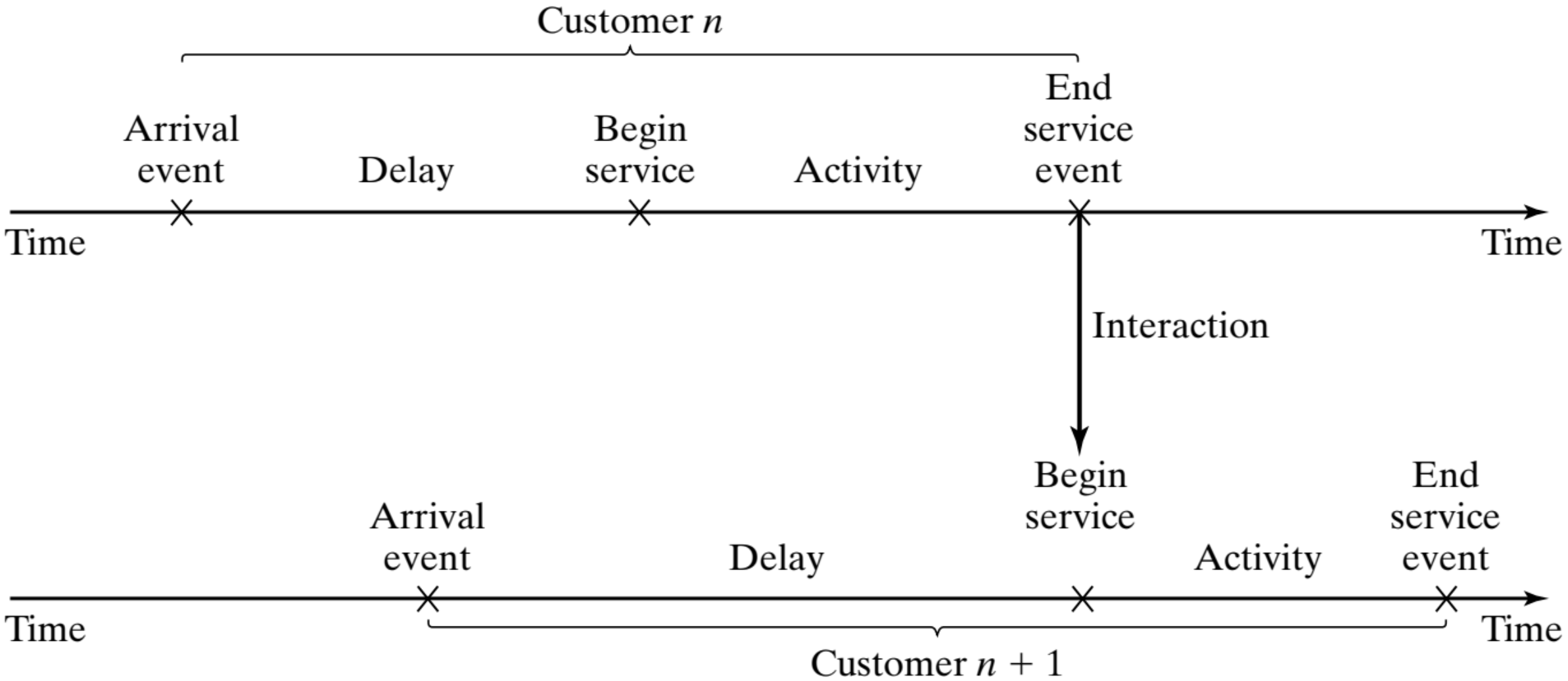

Figure 4 Two interacting customer processes in a single-server queue.

Simulasi dengan Event Scheduling

Example 3: Single-Channel Queue

Consider the grocery store with one checkout counter that is simulated in Example 5 from the Simulation Examples in a Spreadsheet chapter by an ad hoc method. The system consists of those customers in the waiting line plus the one (if any) checking out. A stopping time of 60 minutes is set for this example. The model has the following components:

System state $(LQ(t), LS(t))$, where $LQ(t)$ is the number of customers in the waiting line, and $LS(t)$ is the number being served (0 or 1) at time t.

Entities The server and customers are not explicitly modeled, except in terms of the state variables.

Events

Arrival (A);

Departure (D);

Stopping event (E), scheduled to occur at time 60.

Example 3: Single-Channel Queue

Event notices

(A, t), representing an arrival event to occur at future time t;
(D, t), representing a customer departure at future time t;
(E, 60), representing the simulation stop event at future time 60.

Activities

Interarrival time, defined in Table 9 from the Simulation Examples in a Spreadsheet chapter;
Service time, defined in Table 10 from the Simulation Examples in a Spreadsheet chapter.

Delay Customer time spent in waiting line.

The event notices are written as (event type, event time). In this model, the FEL will always contain either two or three event notices. The effect of the arrival and departure events is shown in detail in Figures 5 and 6.

Example 3: Single-Channel Queue

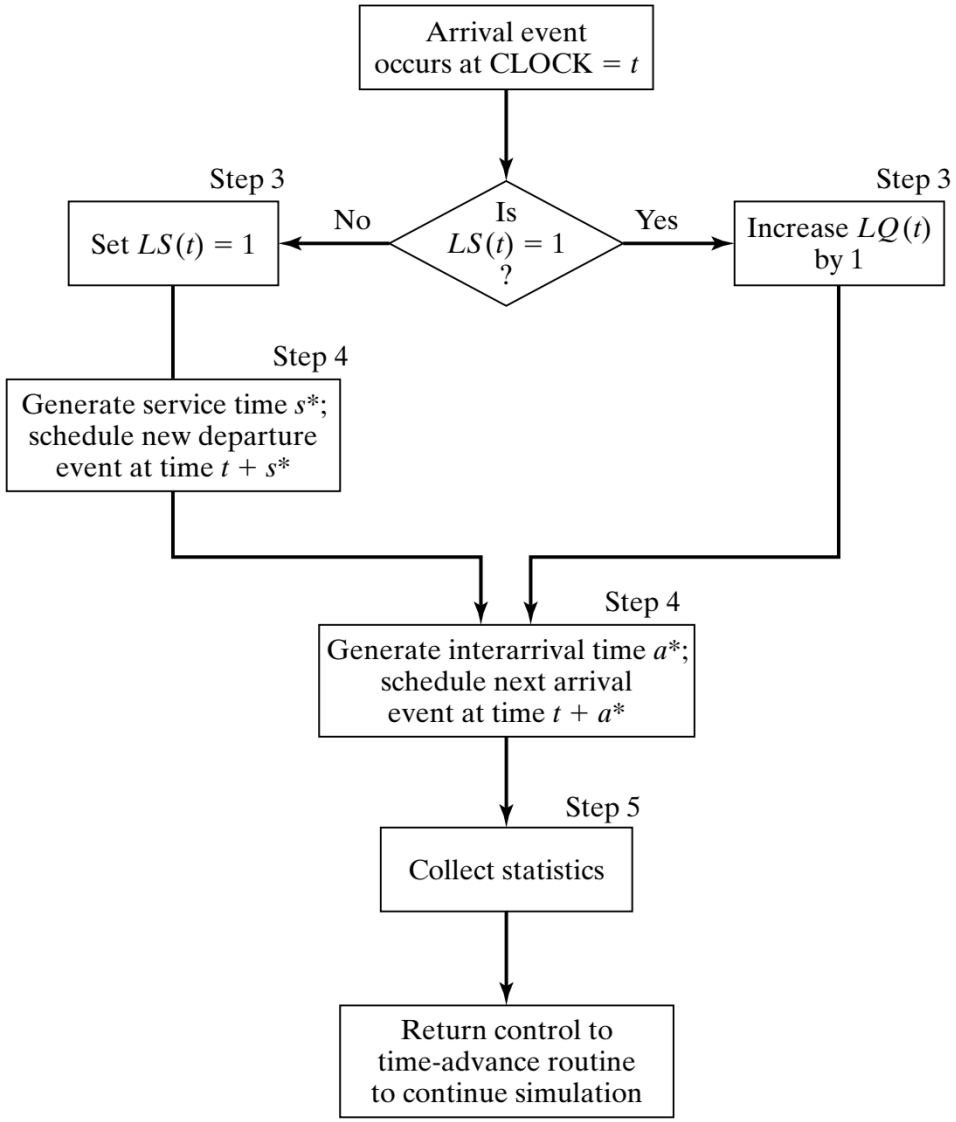

Figure 5 Execution of the arrival event.

Example 3: Single-Channel Queue

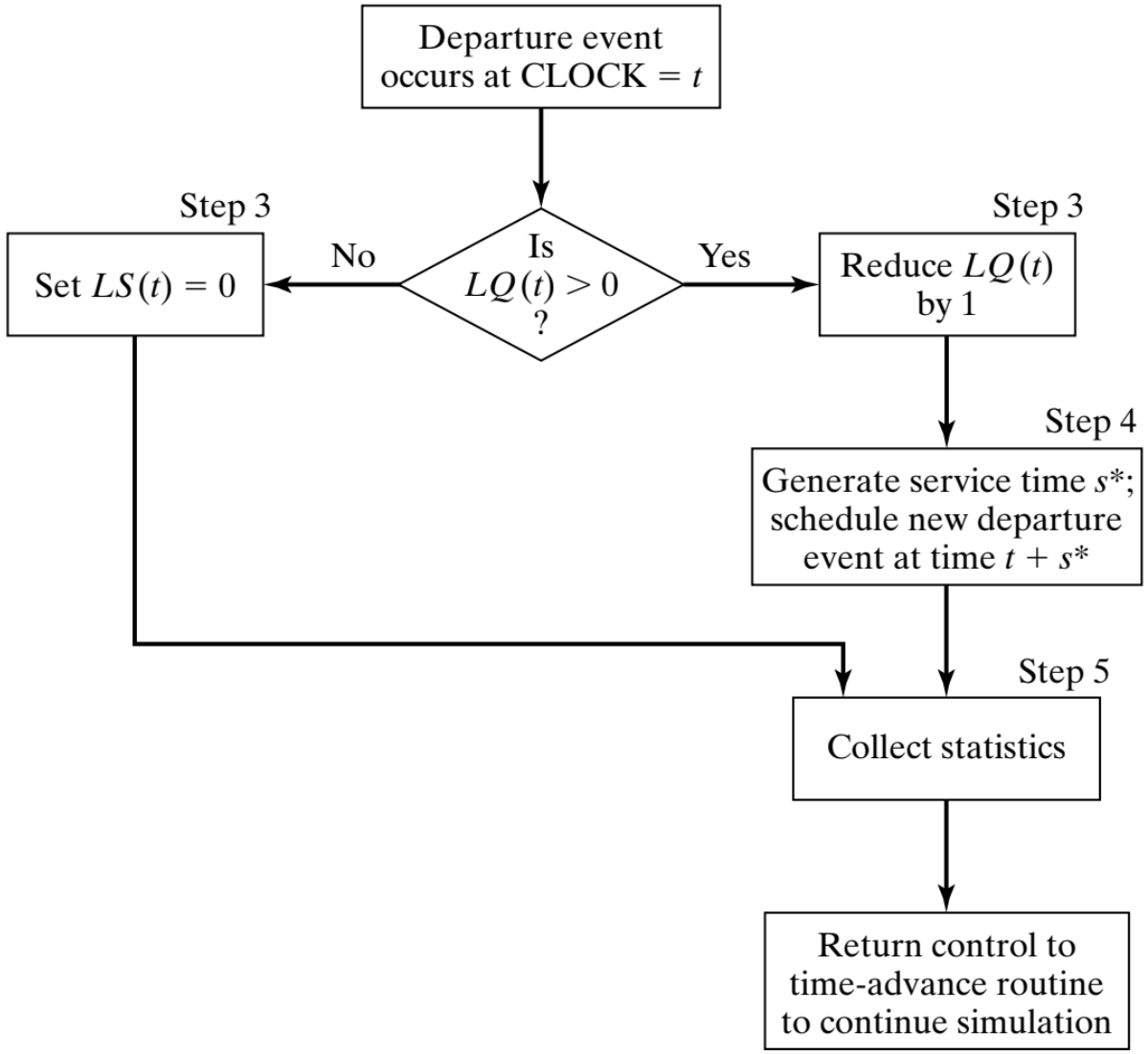

Figure 6 Execution of the departure event.

Example 3: Single-Channel Queue

Interarrival Times	1	1	6	3	7	5	2	4	$1\cdots$
Service Times	4	2	5	4	1	5	4	1	$4\cdots$

Example 3: Single-Channel Queue

Initial conditions are that the first customer arrive at time 0 and begin service. This is reflected in Table 1 by the system snapshot at time zero (CLOCK $= 0$), with $LQ(0) = 0, LS(0) = 1$, and both a departure event and arrival event on the FEL. Also, the simulation is scheduled to stop at time 60. Only two statistics, server utilization and maximum queue length, will be collected. Server utilization is defined by total server busy time (B) divided by total time (T_E). Total busy time, B, and maximum queue length, MQ, will be accumulated as the simulation progresses. A column headed "Comments" is included to aid the reader (a^* and s^* are the generated interarrival and service times, respectively).

As soon as the system snapshot at time CLOCK $= 0$ is complete, the simulation begins. At time 0, the imminent event is (A,1). The CLOCK is advanced to time 1, and (A,1) is removed from the FEL. Because $LS(t) = 1$ for $0 \leq t \leq 1$ (i.e., the server was busy for 1 minute), the cumulative busy time is increased from $B = 0$ to $B = 1$. By the event logic in Figure 6, set $LS(1) = 1$ (the server becomes busy). The FEL is left with three future events, (A, 2), (D, 4), and (E, 60). The simulation CLOCK is next advanced to time 2, and an arrival event is executed. The interpretation of the remainder of Table 1 is left to the reader.

Example 3: Single-Channel Queue

Table 1 Simulation Table for Checkout Counter

CLOCK	System state $LQ(t)$	System state $LS(t)$	Future event list	Comment	Cumulative statistics B	Cumulative statistics MQ
0	0	1	(A, 1) (D, 4) (E, 60)	First A occurs $(a^* = 1)$ Schedule next A $(s^* = 4)$ Schedule first D	0	0
1	1	1	(A, 2) (D, 4) (E, 60)	Second A occurs: (A, 1) $(a^* = 1)$ Schedule next A (Customer delayed)	1	1
2	2	1	(D, 4) (A, 8) (E, 60)	Third A occurs: (A, 2) $(a^* = 6)$ Schedule next A (Two customers delayed)	2	2

Example 3: Single-Channel Queue

Table 1 Simulation Table for Checkout Counter

CLOCK	System state $LQ(t)$	System state $LS(t)$	Future event list	Comment	Cumulative statistics B	Cumulative statistics MQ
0	0	1	(A, 1) (D, 4) (E, 60)	First A occurs ($a^* = 1$) Schedule next A ($s^* = 4$) Schedule first D	0	0
1	1	1	(A, 2) (D, 4) (E, 60)	Second A occurs: (A, 1) ($a^* = 1$) Schedule next A (Customer delayed)	1	1
2	2	1	(D, 4) (A, 8) (E, 60)	Third A occurs: (A, 2) ($a^* = 6$) Schedule next A (Two customers delayed)	2	2
4	1	1	(D, 6) (A, 8) (E, 60)	First D occurs: (D, 4) ($s^* = 2$) Schedule next D (Customer delayed)	4	2
6	0	1	(A, 8) (D, 11) (E, 60)	Second D occurs: (D, 6) ($s^* = 5$) Schedule next D	6	2
8	1	1	(D, 11) (A, 11) (E, 60)	Fourth A occurs: (A, 8) ($a^* = 3$) Schedule next A (Customer delayed)	8	2
11	1	1	(D, 15) (A, 18) (E, 60)	Fifth A occurs: (A, 11) ($a^* = 7$) Schedule next A Third D occurs: (D, 11) ($s^* = 4$) Schedule next D (Customer delayed)	11	2
15	0	1	(D, 16) (A, 18) (E, 60)	Fourth D occurs: (D, 15) ($s^* = 1$) Schedule next D	15	2
16	0	0	(A, 18) (E, 60)	Fifth D occurs: (D, 16)	16	2
18	0	1	(D, 23) (A, 23) (E, 60)	Sixth A occurs ($a^* = 5$) Schedule next A ($s^* = 5$) Schedule next D	16	2
23	0	1	(A, 25) (D, 27) (E, 60)	Seventh A occurs: (A, 23) ($a^* = 2$) Schedule next Arrival Sixth D occurs: (D, 23)	21	2

Table 11 Model Responses and Simulation Table (first 11 customers) for the Grocery Store Simulation

	A	B	C	D	E	F	G	H	I	J
15		**TOTALS**	420		320		163		483	106
16		**AVERAGES**	4.24		3.20		1.63		4.83	1.07
17				**Number of Customers=**	100					
18		**Simulation Table**								
19		**Step**	**Activity**	**Clock**	**Activity**	**Clock**	**Output**	**Clock**	**Output**	**Output**
20			**Interarrival**		**Service**	**Time**	**Waiting Time**	**Time**	**Time Customer**	**Idle Time**
21			**Time**		**Time**	**Service**	**in Queue**	**Service**	**Spends in System**	**of Server**
22		**Customer**	**(Minutes)**	**Arrival Time**	**(Minutes)**	**Begins**	**(Minutes)**	**Ends**	**(Minutes)**	**(Minutes)**
23		1	0	0	2	0	0	2	2	
24		2	5	5	2	5	0	7	2	3
25		3	5	10	4	10	0	14	4	3
26		4	4	14	4	14	0	18	4	0
27		5	2	16	3	18	2	21	5	0
28		6	8	24	2	24	0	26	2	3
29		7	7	31	3	31	0	34	3	5
30		8	8	39	5	39	0	44	5	5
31		9	5	44	1	44	0	45	1	0
32		10	2	46	6	46	0	52	6	1
33		11	1	47	4	52	5	56	9	0

Example 4: The Checkout-Counter Simulation, Continued

Suppose that, in the simulation of the checkout counter in Example 3, the simulation analyst desires to estimate mean response time and mean proportion of customers who spend 5 or more minutes in the system. A response time is the length of time a customer spends in the system. In order to estimate these customer averages, it is necessary to expand the model in Example 3 to represent the individual customers explicitly. In addition, to be able to compute an individual customer's response time when that customer departs, it will be necessary to know that customer's arrival time. Therefore, a customer entity with arrival time as an attribute will be added to the list of model components in Example 3. These customer entities will be stored in a list to be called "CHECKOUT LINE"; they will be called $C1, C2, C3, \ldots$ Finally, the event notices on the FEL will be expanded to indicate which customer is affected. For example, (D,4,$C1$) means that customer $C1$ will depart at time 4. The additional model components are the following:

Example 4: The Checkout-Counter Simulation, Continued

Entities

(Ci, t), representing customer Ci who arrived at time t.

Event notices

(A, t, Ci), the arrival of customer Ci at future time t;

(D, t, Cj), the departure of customer Cj at future time t,

Set

"CHECKOUT LINE," the set of all customers currently at the checkout counter (being served or waiting to be served), ordered by time of arrival.

Example 4: The Checkout-Counter Simulation, Continued

Three new cumulative statistics will be collected: S, the sum of customer response times for all customers who have departed by the current time; F, the total number of customers who spend 5 or more minutes at the checkout counter; and N_D, the total number of departures up to the current simulation time. These three cumulative statistics will be updated whenever the departure event occurs; the logic for collecting these statistics would be incorporated into Step 5 of the departure event in Figure 6.

Example 4: The Checkout-Counter Simulation, Continued

Table 2 Simulation Table for Example 4

CLOCK	System state $LQ(t)$	System state $LS(t)$	CHECKOUT LINE	Future event list	Cumulative statistics S	Cumulative statistics N_D	Cumulative statistics F
0	0	1	(*C*1,0)	(A,1,*C*2) (D,4,*C*1) (E,60)	0	0	0
1	1	1	(*C*1,0) (*C*2,1)	(A,2,*C*3) (D,4,*C*1) (E,60)	0	0	0
2	2	1	(*C*1,0) (*C*2,1) (*C*3,2)	(D,4,*C*1) (A,8,*C*4) (E,60)	0	0	0
4	1	1	(*C*2,1) (*C*3,2)	(D,6,*C*2) (A,8,*C*4) (E,60)	4	1	0
6	0	1	(*C*3,2)	(A,8,*C*4) (D,11,*C*3) (E,60)	9	2	1
8	1	1	(*C*3,2) (*C*4,8)	(D,11,*C*3) (A,11,*C*5) (E,60)	9	2	1
11	1	1	(*C*4,8) (*C*5,11)	(D,15,*C*4) (A,18,*C*6) (E,60)	18	3	2
15	0	1	(*C*5,11)	(D,16,*C*5) (A,18,*C*6) (E,60)	25	4	3
16	0	0		(A,18,*C*6) (E,60)	30	5	4
18	0	1	(*C*6,18)	(D,23,*C*6) (A,23,*C*7) (E,60)	30	5	4
23	0	1	(*C*7,23)	(A,25,*C*8) (D,27,*C*7) (E,60)	35	6	5

Example 4: The Checkout-Counter Simulation, Continued

The simulation table for Example 4 is shown in Table 2. The same data for interarrival and service times will be used again; so Table 2 essentially repeats Table 1, except that the new components are included (and the comment column has been deleted). These new components are needed for the computation of the cumulative statistics S, F, and N_D. For example, at time 4, a departure event occurs for customer $C1$. The customer entity $C1$ is removed from the list called "CHECKOUT LINE"; the attribute "time of arrival" is noted to be 0, so the response time for this customer was 4 minutes. Hence, S is incremented by 4 minutes. N_D is incremented by one customer, but F is not incremented, for the time in system was less than five minutes. Similarly, at time 23, when the departure event $(D, 23, C6)$ is being executed, the response time for customer $C6$ is computed by

$$\begin{aligned} \text{Response time} &= \text{CLOCK TIME} - \text{attribute "time of arrival"} \\ &= 23 - 18 \\ &= 5 \text{ minutes} \end{aligned}$$

Then S is incremented by 5 minutes, and F and N_D by one customer.

Example 4: The Checkout-Counter Simulation, Continued

For a simulation run length of 23 minutes, the average response time was $S/N_D = 35/6 = 5.83$ minutes, and the observed proportion of customers who spent 5 or more minutes in the system was $F/N_D = 0.83$. Again, this simulation was far too short to regard these estimates as having any degree of accuracy. The purpose of Example 4, however, was to illustrate the notion that, in many simulation models, the information desired from the simulation (such as the statistics S/N_D and F/N_D) to some extent dictates the structure of the model.

Example 5: The Dump-Truck Problem

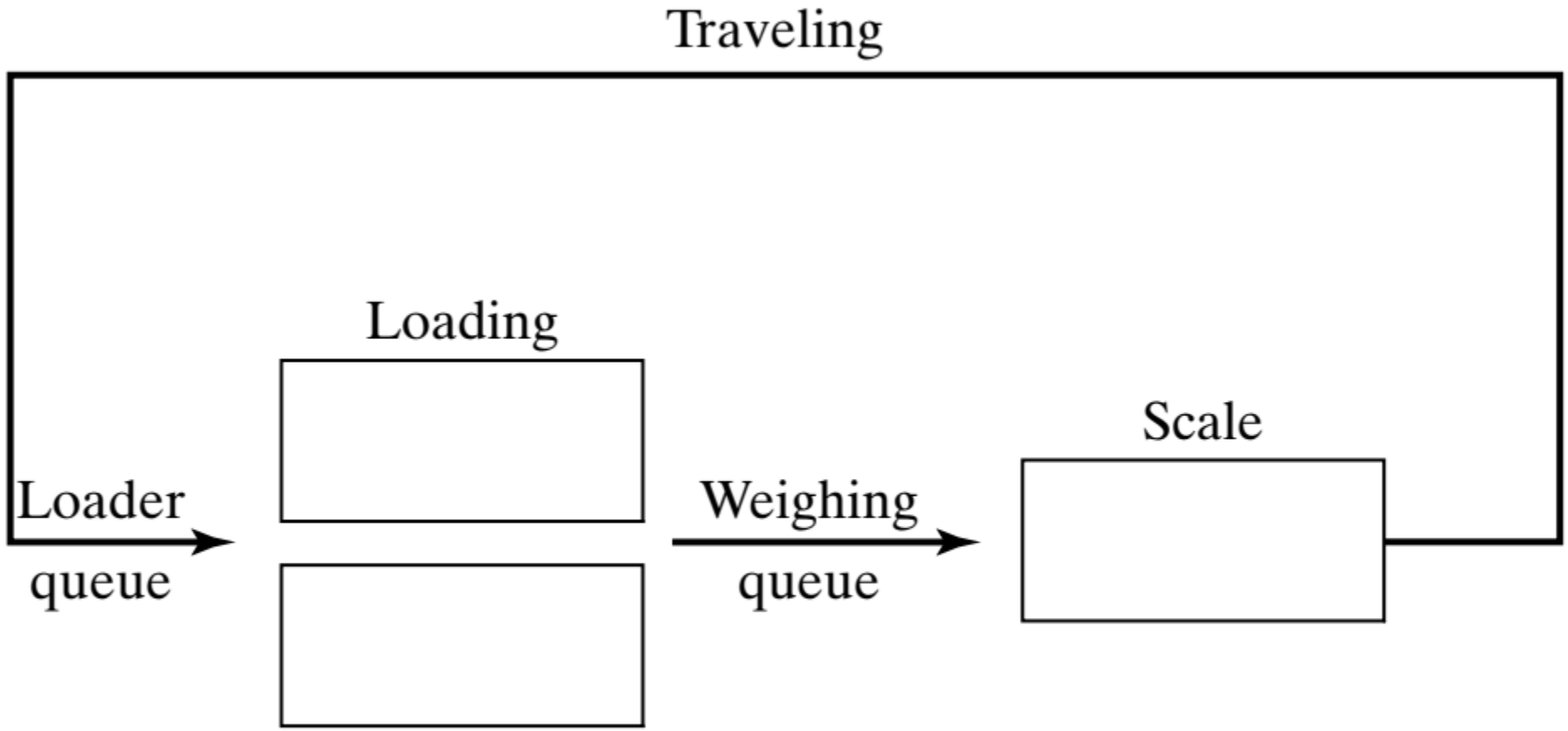

Figure 7 Dump truck problem.

Example 5: The Dump-Truck Problem

Six dump trucks are used to haul coal from the entrance of a small mine to the railroad. Figure 7 provides a schematic of the dump-truck operation. Each truck is loaded by one of two loaders. After a loading, the truck immediately moves to the scale, to be weighed as soon as possible. Both the loaders and the scale have a first-come–first-served waiting line (or queue) for trucks. Travel time from a loader to the scale is considered negligible. After being weighed, a truck begins a travel time (during which time the truck unloads) and then afterward returns to the loader queue.

The purpose of the simulation is to estimate the loader and scale utilizations (percentage of time busy). The model has the following components:

Example 5: The Dump-Truck Problem

System state

$[LQ(t), L(t), WQ(t), W(t)]$, where

$LQ(t)$ = number of trucks in loader queue;

$L(t)$ = number of trucks (0, 1, or 2) being loaded

$WQ(t)$ = number of trucks in weigh queue;

$W(t)$ = number of trucks (0 or 1) being weighed, all at simulation time t.

Entities

The six dump trucks ($DT1, \ldots, DT6$).

Event notices

(ALQ, t, DTi), DTi arrives at loader queue (ALQ) at time t;

(EL, t, DTi), DTi ends loading (EL) at time t;

(EW, t, DTi), DTi ends weighing (EW) at time t.

Example 5: The Dump-Truck Problem

Lists

Loader queue, all trucks waiting to begin loading, ordered on a first-come–first-served basis;
Weigh queue, all trucks waiting to be weighed, ordered on a first-come–first-served basis.

Activities

Loading time, weighing time, and travel time.

Delays

Delay at loader queue, and delay at scale.

Example 5: The Dump-Truck Problem

The simulation table is given in Table 6. To initialize the table's first row, we assume that, at time 0, five trucks are at the loaders and one is at the scale. For simplicity, we take the (randomly generated) activity times from the following list as needed:

Loading Time	10	5	5	10	15	10	10
Weighing Time	12	12	12	16	12	16	
Travel Time	60	100	40	40	80		

Example 5: The Dump-Truck Problem

Table 6 Simulation Table for Dump-Truck Operation

CLOCK t	System state				Lists		Future event list	Cumulative statistics	
	$LQ(t)$	$L(t)$	$WQ(t)$	$W(t)$	Loader queue	Weigh queue		B_L	B_S
0	3	2	0	1	DT4 DT5 DT6		(EL, 5, DT3) (EL, 10, DT2) (EW, 12, DT1)	0	0
5	2	2	1	1	DT5 DT6	DT3	(EL, 10, DT2) (EL, 5 + 5, DT4) (EW, 12, DT1)	10	5
10	1	2	2	1	DT6	DT3 DT2	(EL, 10, DT4) (EW, 12, DT1) (EL, 10 + 10, DT5)	20	10
10	0	2	3	1		DT3 DT2 DT4	(EW, 12, DT1) (EL, 20, DT5) (EL, 10 + 15, DT6)	20	10
12	0	2	2	1		DT2 DT4	(EL, 20, DT5) (EW, 12 + 12, DT3) (EL, 25, DT6) (ALQ, 12 + 60, DT1)	24	12

Example 5: The Dump-Truck Problem

Table 6 Simulation Table for Dump-Truck Operation

CLOCK	System state				Lists		Future event list	Cumulative statistics	
t	$LQ(t)$	$L(t)$	$WQ(t)$	$W(t)$	Loader queue	Weigh queue		B_L	B_S
0	3	2	0	1	DT4 DT5 DT6		(EL, 5, DT3) (EL, 10, DT2) (EW, 12, DT1)	0	0
5	2	2	1	1	DT5 DT6	DT3	(EL, 10, DT2) (EL, 5 + 5, DT4) (EW, 12, DT1)	10	5
10	1	2	2	1	DT6	DT3 DT2	(EL, 10, DT4) (EW, 12, DT1) (EL, 10 + 10, DT5)	20	10
10	0	2	3	1		DT3 DT2 DT4	(EW, 12, DT1) (EL, 20, DT5) (EL, 10 + 15, DT6)	20	10
12	0	2	2	1		DT2 DT4	(EL, 20, DT5) (EW, 12 + 12, DT3) (EL, 25, DT6) (ALQ, 12 + 60, DT1)	24	12
20	0	1	3	1		DT2 DT4 DT5	(EW, 24, DT3) (EL, 25, DT6) (ALQ, 72, DT1)	40	20
24	0	1	2	1		DT4 DT5	(EL, 25, DT6) (EW, 24 + 12, DT2) (ALQ, 72, DT1) (ALQ, 24 + 100, DT3)	44	24
25	0	0	3	1		DT4 DT5 DT6	(EW, 36, DT2) (ALQ, 72, DT1) (ALQ, 124, DT3)	45	25
36	0	0	2	1		DT5 DT6	(EW, 36 + 16, DT4) (ALQ, 72, DT1) (ALQ, 36 + 40, DT2) (ALQ, 124, DT3)	45	36
52	0	0	1	1		DT6	(EW, 52 + 12, DT5) (ALQ, 72, DT1) (ALQ, 76, DT2) (ALQ, 52 + 40, DT4) (ALQ, 124, DT3)	45	52
64	0	0	0	1			(ALQ, 72, DT1) (ALQ, 76, DT2) (EW, 64 + 16, DT6) (ALQ, 92, DT4) (ALQ, 124, DT3) (ALQ, 64 + 80, DT5)	45	64
72	0	1	0	1			(ALQ, 76, DT2) (EW, 80, DT6) (EL, 72 + 10, DT1) (ALQ, 92, DT4) (ALQ, 124, DT3) (ALQ, 144, DT5)	45	72
76	0	2	0	1			(EW, 80, DT6) (EL, 82, DT1) (EL, 76 + 10, DT2) (ALQ, 92, DT4) (ALQ, 124, DT3) (ALQ, 144, DT5)	49	76

Example 5: The Dump-Truck Problem

In order to estimate the loader and scale utilizations, two cumulative statistics are maintained:

$$B_L = \text{ total busy time of both loaders from time 0 to time } t$$
$$B_S = \text{ total busy time of the scale from time 0 to time } t$$

Both loaders are busy from time 0 to time 20, so $B_L = 40$ at time $t = 20$—but, from time 20 to time 24, only one loader is busy; thus, B_L increases by only 4 minutes over the time interval [20, 24]. Similarly, from time 25 to time 36, both loaders are idle ($L(25) = 0$), so B_L does not change. For the relatively short simulation in Table 6, the utilizations are estimated as follows:

$$\text{Average loader utilization} = \frac{49/2}{76} = 0.32$$
$$\text{Average scale utilization} = \frac{76}{76} = 1.00$$