

BAB II

TINJAUAN PUSTAKA

2.1 Teknik Pengukuran Waktu

Secara garis besar, pengukuran waktu standar dapat dikelompokkan menjadi dua (Wignjosoebroto, 2006:132).

1. Pengukuran Waktu Secara Langsung.

Teknik pengukuran ini disebut pengukuran waktu kerja secara langsung karena pengukuran dilakukan dengan mengamati secara langsung di lokasi kerja pada pekerjaan yang akan di ukur. Dalam pengukuran waktu secara langsung ada dua metode yaitu :

- a. Pengukuran waktu kerja dengan jam henti (*Stopwatch Time Study*).
- b. Pengukuran waktu kerja *Work Sampling*.

2. Pengukuran Waktu Secara Tak Langsung.

Dalam pengukuran waktu kerja secara tak langsung terdapat 2 metode yaitu :

- a. Pengukuran waktu kerja dengan metode *Standard Data*.
- b. Pengukuran waktu kerja *Predetermined Motion Time System*.

2.1.1 Pengukuran Waktu Kerja Jam Henti (*Stopwatch Time Study*).

Pengukuran waktu kerja dengan jam henti dapat diaplikasikan untuk pekerjaan yang singkat dan *repetitive*. Pengukuran waktu kerja dengan jam henti merupakan cara yang obyektif karena waktu yang ditetapkan berdasarkan fakta terjadi di lapangan, bukan dari hasil estimasi. Pengukuran ini dilakukan dengan cara mengamati dan mencatat waktu kerja yang dilakukan oleh seorang operator sesuai siklus operasi kerja. Secara umum, langkah-langkah pengukuran kerja jam henti dapat dijelaskan sebagai berikut (Wignjosoebroto, 2006:171).

1. Prosedur pelaksanaan dan alat yang digunakan.
 - a. Sebelum melakukan pengukuran, haruslah ada penetapan tujuan, untuk kepentingan hasil pengukuran tersebut digunakan.

- b. Persiapan awal pengukuran waktu kerja. Dalam tahap ini, kondisi kerja pada pekerjaan yang akan diukur hendaklah seperti pada kondisi normal, baik kecepatan kerja, spesifikasi material, proses-proses yang dikerjakan maupun kondisi kerja yang lainnya, sehingga waktu baku yang ditetapkan diperoleh dari kondisi kerja dan metode yang baik.
- c. Setelah melakukan persiapan awal, tahap berikutnya yaitu memilih operator yang memiliki *skill* yang normal dengan tujuan hasil dari penetapan waktu baku dapat diikuti oleh operator lain.
- d. Langkah selanjutnya yaitu mempersiapkan alat pengukuran kerja. Alat yang digunakan untuk pengukuran kerja dengan jam henti antara lain *stopwatch*, lembar pengamatan, alat tulis dan kalkulator.
- e. Pembagian operasi menjadi elemen-elemen kerja dengan tujuan memperoleh gambaran operasi secara detail dan mempermudah pengukuran dan pencatatan waktu bakunya.
- f. Cara pengukuran dan pencatatan waktu kerja. Ada tiga cara untuk melakukan pengukuran dan pencatatan waktu kerja, yaitu pengukuran waktu secara terus menerus, pengukuran waktu secara berulang, dan pengukuran waktu akumulasi.
- g. Penetapan jumlah siklus kerja yang akan diamati.

Pada dasarnya, pengukuran waktu kerja merupakan proses *sampling* dengan konsekuensi semakin besar jumlah siklus kerja yang diamati atau diukur, semakin mendekati kebenaran data waktu yang diperoleh. Maka dari itu perlu adanya penetapan jumlah pengamatan atau biasa disebut uji kecukupan data. Namun sebelum menentukan berapa jumlah observasi yang dibutuhkan, terlebih dahulu ditetapkan tingkat kepercayaan dan derajat ketelitian. Rumus untuk menetapkan jumlah observasi seperti berikut (Wignjosoebroto, 200:134):

$$N' = \left[\frac{\frac{K}{S} \sqrt{N \sum X^2 - (\sum X)^2}}{\sum X} \right]^2$$

Keterangan :

K = tingkat kepercayaan

S = tingkat ketelitian

X = waktu tiap pengamatan

N = jumlah pengamatan

N' = jumlah pengamatan yang dibutuhkan.

Apabila $N > N'$, maka data pengamatan bisa dikatakan cukup. Namun apabila $N < N'$, maka perlu melakukan penambahan jumlah pengamatan.

h. Uji keseragaman data

Uji keseragaman data perlu dilakukan untuk melihat apakah ada penyimpangan data yang ekstrim atau tidak jika dibandingkan dengan data yang lainnya. Data yang terlalu ekstrim yang diperoleh dapat disebabkan oleh kesalahan pengamat saat membaca *stopwatch*, kekeliruan pada saat menulis atau karena situasi/kondisi kerja yang tidak wajar. Untuk uji keseragaman data, umumnya digunakan peta control (*control chart*). Dalam peta kontrol yang digunakan adalah rata-rata (*Mean*) dari seluruh data pengamatan, Batas Kontrol Atas (BKA) dan Batas Kontrol Bawah (BKB). Untuk menghitung mean digunakan rumus sebagai berikut (Wignjosoebroto, 2006:184):

$$\bar{X} = \frac{\sum X_i}{N}$$

Keterangan :

$$\bar{X} = \text{mean}$$

$\sum X_i$ = total waktu pengamatan

N = jumlah pengamatan

Sedangkan untuk menghitung standar deviasi digunakan rumus sebagai berikut (Purnomo, 2004:47).

$$SD = \sqrt{\frac{\sum(x_i - \bar{x})^2}{N-1}} \quad (2.3)$$

Keterangan :

SD = standar deviasi

X_i = waktu pengamatan ke -i

\bar{X} = *mean*

N = jumlah pengamatan

Kemudian menentukan batas kontrol atas dan batas kontrol bawah dengan rumus sebagai berikut (Wignjosoebroto, 2006:195)

$$BKA = \bar{X} + (3SD)$$

$$BKB = \bar{X} - (3SD)$$

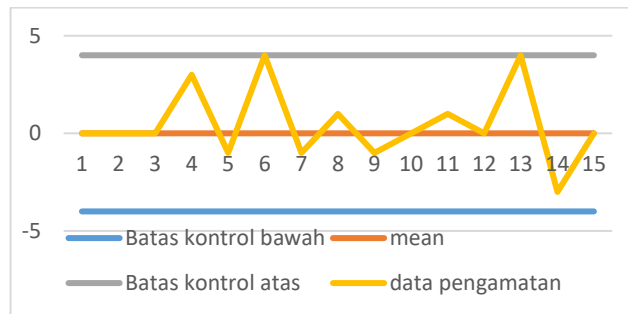
Keterangan :

BKA = Batas kontrol atas

BKB = Batas kontrol bawah

SD = Standar deviasi

Kemudian setelah didapatkan *mean*, batas kontrol atas dan batas kontrol bawah, data pengamatan di plot pada peta kontrol. Contoh peta kontrol seperti pada Gambar 2.1.



Gambar 2. 1 Contoh Peta Kontrol untuk Uji Keseragaman Data

Data dapat dikatakan seragam apabila keseluruhan data berada dalam batas kontrol. Begitu pula sebaliknya, data dikatakan tidak seragam jika terdapat data yang berada di luar batas kontrol. Maka data yang berada diluar batas control akan dieliminasi.

i. Penyesuaian waktu dengan *Rating Performance* Kerja

Rating Performance dilakukan untuk menormalkan waktu kerja yang diperoleh dari hasil pengamatan. Dalam penyesuaian dengan *Westing House System's Ratings* terdapat empat faktor yang akan dilakukan penilaian, yaitu :

1. *Skill* : kemampuan mengikuti cara kerja yang telah ditetapkan.
2. *Effort* : kesungguhan yang ditunjukkan atau diberikan operator ketika melakukan pekerjaannya.
3. *Condition* : kondisi fisik lingkungan kerja seperti keadaan, pencahayaan, temperatur dan kebisingan ruangan.
4. *Consistency* : waktu penyelesaian yang selalu tetap dari satu waktu ke waktu lain.

Untuk kelas dan nilai *rating factor* pada *Westing House System's Ratings* dapat dilihat pada Tabel 2.1.

Tabel 2. 1 Tabel *Performance Ratings* dengan sistem *Westinghouse*

<i>Skill</i>		<i>Effort</i>	
+ 0,15 A1	Superskill	+ 0,13 A1	Superskill
+ 0,13 A2		+ 0,12 A2	
+ 0,11 B1	Excellent	+ 0,10 B1	Excellent
+ 0,08 B2		+ 0,08 B2	
+ 0,06 C1	Good	+ 0,05 C1	Good
+ 0,03 C2		+ 0,02 C2	
0,00 D	Average	0,00 D	Average
-0,05 E1	Fair	-0,04 E1	Fair
-0,10 E2		-0,08 E2	
-0,16 F1	Poor	-0,12 F1	Poor
-0,22 F2		-0,17 F2	Poor
<i>Condition</i>		<i>Consistency</i>	
+ 0,06 A	Ideal	+ 0,04 A	Ideal
+ 0,04 B	Excellent	+ 0,03 B	Excellent
+ 0,02 C	Good	+ 0,01 C	Good
0,00 D	Average	0,00 D	Average
-0,03 E	Fair	-0,02 E	Fair
-0,07 F	Poor	-0,04 F	Poor

(Sumber : Wigjosoebroto, 2006:198)

Menurut Sutalaksana dkk (2006) : Keterampilan atau *skill* didefinisikan sebagai kemampuan mengikuti cara kerja yang diterapkan. Untuk keperluan penyesuaian, keterampilan dibagi menjadi enam kelas dengan ciri-ciri dari setiap kelas yang dikemukakan berikut ini:

Super skill :

1. Secara bawaan cocok sekali dengan pekerjaannya.
2. Bekerja dengan sempurna.
3. Gerakan – gerakannya halus tetapi sangat cepat sehingga sifat untuk diikuti.
4. Tampak seperti telah terlatih dengan cepat sehingga sangat sulit untuk diikuti.
5. Kadang-kadang terkesan tidak berbeda dengan gerakan-gerakan mesin.
6. Perpindahan dari satu elemen pekerjaan ke elemen lainnya tidak terlampau terlihat karena lancarnya.
7. Tidak terkesan adanya gerakan-gerakan berfikir dan merencanakan tentang apa yang dikerjakan (sudah sangat otomatis).
8. Secara umum dapat dikatakan bahwa pekerja yang bersangkutan adalah pekerja yang sangat baik.

Excellent skill:

1. Percaya pada diri sendiri.

2. Tampak cocok dengan pekerjaannya.
3. Terlihat telah terlatih baik.
4. Bekerjanya teliti dengan tidak banyak melakukan atau pemeriksaan lagi.
5. Gerakan-gerakan kerjanya beserta urutan-urutannya dijalankan tanpa kesalahan.
6. Menggunakan peralatan dengan baik.
7. Bekerjanya cepat tanpa mengorbankan mutu.
8. Bekerjanya cepat tapi halus.
9. Bekerjanya berirama dan berkomondasi.

Good skill:

1. Kualitas hasil baik.
2. Bekerjanya tampak lebih baik daripada kebanyakan pekerja pada umumnya.
3. Dapat memberi petunjuk-petunjuk pada pekerja lain yang keterampilannya lebih rendah.
4. Tampak jelas sebagai pekerja yang cakap.
5. Tidak memerlukan banyak pengawasan.
6. Tiada keraguan.
7. Kerjanya “stabil”.
8. Gerakan-gerakan terkoordinasi dengan baik.
9. Gerakan-gerakannya cepat.

Average skill:

1. Tampak adanya kepercayaan pada diri sendiri.
2. Gerakannya cepat tetapi tidak lambat.
3. Terlihat adanya pekerjaan-pekerjaan perencanaan.
4. Tampak sebagai pekerja yang cakap.
5. Gerakan-gerakan cukup menunjukkan tidak ada keraguan.
6. Mengkoordinasi tangan dan pikiran dengan cukup baik.
7. Tampak cukup terlatih dan karenanya mengetahui seluk beluk pekerjaannya.
8. Bekerja cukup teliti.
9. Secara keseluruhan cukup memuaskan.

Fair skill:

1. Tampak terlatih tetapi belum cukup baik.
2. Mengenal peralatan dan lingkungan secukupnya.
3. Terlihat adanya perencanaan-perencanaan sebelum melakukan gerakan-gerakan.
4. Tidak mempunyai kepercayaan diri yang cukup.
5. Tampaknya seperti tidak cocok dengan pekerjaannya tetapi telah dipekerjakan di bagian itu sejak lama.

6. Mengetahui apa-apa yang dilakukan dan harus dilakukan tapi tampak tidak selalu yakin.
7. Sebagian waktunya terbuang karena kesalahan-kesalahan sendiri.
8. Jika tidak bekerja secara sungguh-sungguh outputnya akan sangat rendah.
9. Biasanya tidak ragu-ragu dalam menjalankan gerakan-gerakannya.

Poor skill:

1. Tidak bias mengkoordinasikan tangan dan pikiran.
2. Gerakan-gerakannya kaku.
3. Kelihatan ketidakyakinannya pada urutan-urutan gerakan.
4. Seperti yang tidak terlatih untuk pekerjaan yang bersangkutan.
5. Tidak terlihat adanya kecocokan dengan pekerjaannya.
6. Ragu-ragu dalam melaksanakan gerakan-gerakan kerja.
7. Sering melakukan kesalahan-kesalahan.
8. Tidak adanya kepercayaan pada diri sendiri.

Untuk usaha atau *effort* cara Westing House membagi juga kelas-kelas dengan ciri-ciri tersendiri. Yang dimaksud usaha disini adalah kesungguhan yang ditunjukkan atau diberikan operator ketika melakukan pekerjaannya (Sutalaksana dkk, 2006). Berikut ini ada enam kelas usaha dengan ciri-cirinya, yaitu:

Excessive effort:

1. Kesempatan sangat berlebihan.
2. Usahnya sangat bersungguh-sungguh tetapi dapat membahayakan kesehatannya.
3. Kecepatan yang ditimbulkannya tidak dapat dipertahankan sepanjang hari kerja.

Excellent effort:

1. Jelas terlihat kecepatannya sangat tinggi.
2. Gerakan-gerakan lebih ekonomis daripada operator-operator biasa.
3. Penuh perhatian pada pekerjaannya.
4. Banyak memberi saran.
5. Menerima saran-saran petunjuk dengan senang.
6. Tidak bertahan lebih dari beberapa hari
7. Bangga atas kelebihannya.
8. Gerakan-gerakan yang salah terjadi sangat jarang sekali.
9. Bekerjanya sangat sistematis

Good effort:

1. Bekerja berirama.
2. Saat-saat mengganggu sangat sedikit, bahkan kadang-kadang tidak ada.
3. Penuh perhatian pada pekerjaannya.
4. Senang pada pekerjaannya.
5. Kecepatannya baik dan dapat dipertahankan sepanjang hari.
6. Percaya pada pekerjaannya.
7. Menerima saran-saran dan petunjuk dengan senang.

Average effort:

1. Tidak sebaik *good*, tapi lebih baik dari *poor*.
2. Bekerja dengan stabil.
3. Menerima saran-saran tapi tidak melaksanakannya.
4. *Set up* dilaksanakan dengan baik.
5. Melakukan kegiatan-kegiatan perencanaan.

Fair effort:

1. Saran-saran perbaikan diterima dengan kesal/
2. Kadang-kadang perhatian tidak ditunjukkan pada pekerjaannya.
3. Kurang sungguh-sungguh.
4. Tidak mengeluarkan tenaga dengan secukupnya.
5. Terjadi sedikit penyimpangan dari cara kerja baku.

Poor effort:

1. Banyak membuang waktu.
2. Tidak memperlihatkan adanya minat bekerja
3. Tidak mau menerima saran-saran.
4. Tampak malas dan lambat bekerja.
5. Melakukan gerakan-gerakan yang tidak perlu untuk mengambil alat-alat dan bahan.
6. *Set up* kerjanya terlihat tidak rapi.

Yang dimaksud dengan kondisi kerja atau *Condition* pada cara Westing House adalah kondisi fisik lingkungannya seperti keadaan pencahayaan, suhu, dan kebisingan ruangan. Bila tiga faktor lainnya, yaitu keterampilan, usaha, dan konsistensi merupakan sesuatu yang dicerminkan operator, maka kondisi kerja merupakan sesuatu di luar operator yang diterima apa adanya oleh operator tanpa banyak kemampuan mengubahnya. Oleh sebab itu, faktor kondisi sering disebut

sebagai faktor manajemen, karena pihak inilah yang dapat dan berwenang mengubah atau memperbaikinya (Sutalaksana dkk, 2006).

Menurut Sutalaksana dkk (2006), Kondisi kerja dibagi menjadi enam kelas yaitu *Ideal, Excellent, Good, Average, Fair, Poor*. Kondisi yang *ideal* tidak selalu sama bagi setiap pekerjaan karena berdasarkan karakterlistiknya masing-masing pekerja membutuhkan kondisi *ideal* sendiri-sendiri. Satu kondisi yang dianggap *good* untuk satu pekerjaan dapat saja dirasakan *fair* atau bahkan *poor* bagi pekerjaan yang lain. Pada dasarnya kondisi *ideal* adalah kondisi yang paling cocok untuk pekerjaan yang bersangkutan, yaitu yang memungkinkan kinerja maksimal dari pekerja. Sebaliknya, kondisi *poor* adalah kondisi lingkungan yang tidak membantu jalannya pekerjaan atau bahkan sangat menghambat pencapaian kinerja yang baik. Sudah tentu suatu pengetahuan tentang kriteria yang disebut *ideal*, dan kriteria yang disebut *poor* perlu dimiliki agar penilaian terhadap kondisi kerja dalam rangka melakukan penyesuaian dapat dilakukan dengan seteliti mungkin.

Faktor lain yang harus diperhatikan adalah konsistensi atau *Consistency*. Faktor ini perlu diperhatikan karena pada setiap pengukuran waktu angka-angka yang dicatat tidak pernah semuanya sama, waktu penyelesaian yang ditunjukkan pekerja selalu berubah-ubah dari satu siklus ke siklus lainnya, dari jam ke jam, bahkan dari hari ke hari. Selama ini masih dalam batas-batas kewajaran masalah tidak timbul, tetapi jika variabilitasnya tinggi maka hal tersebut harus diperhatikan. Sebagaimana halnya faktor-faktor lain, konsistensi juga dibagi menjadi enam kelas yaitu *Perfect, Excellent, Good, Average, Fair* dan *Poor*. Seseorang yang bekerja *Perfect* adalah yang dapat bekerja dengan waktu penyelesaian yang boleh dikatakan tetap dari saat ke saat. Sebaliknya konsistensi yang *Poor* terjadi bila waktu-waktu penyelesaiannya berselisih jauh dari rata-rata secara acak. Konsistensi rata-rata atau *Average* adalah bila selisih antara waktu penyelesaian dengan rata-ratanya tidak besar walaupun ada satu dua yang “letaknya” jauh (Sutalaksana dkk, 2006).

Sebagai contoh, apabila, diketahui bahwa waktu rata-rata yang diukur terhadap suatu elemen kerja adalah 0,50 menit dan rating performance operator adalah memenuhi klasifikasi berikut:

-	<i>Excellent skill (B2)</i>	: +0,08
-	<i>Good Effort (C2)</i>	: +0,02
-	<i>Good Condition (C)</i>	: +0,02
-	<i>Good Consistency (C)</i>	: +0,01
	Total	: <u>+0,13</u>

Maka waktu normal untuk elemen kerja ini adalah: $0,50 \times 1,13 = 0,565$ menit.

Untuk menghitung nilai *Rating Performance* menggunakan rumus sebagai berikut :

$$RF = 1 + \text{total rating factor}$$

Setelah dihitung nilai performansinya, untuk menghitung waktu normal didapatkan dari waktu pengamatan dikalikan dengan jumlah nilai faktor *skill*, *effort*, *condition* dan *consistency*, dengan rumus sebagai berikut (Wignjosoebroto, 2006:200):

$$WN = \text{Waktu Pengamatan} \times RF$$

j. *Allowance* (Kelonggaran Waktu)

Dalam pengukuran kerja diberikan kelonggaran atau toleransi waktu untuk *Personal Needs*, *Fatigue Allowance*, dan *Delay Allowance*. Ada 3 kategori dalam *Allowance* yang dibutuhkan oleh setiap operator, yaitu:

1. *Personal need* yang merupakan kelonggaran waktu yang diberikan oleh perusahaan kepada setiap operator untuk melakukan kebutuhan pribadi, misalnya waktu untuk beribadah, mengambil minum, dan pergi ke toilet.
2. *Fatigue*, merupakan kelonggaran waktu untuk melepaskan kelelahan kerja kepada setiap operator.

3. *Delay*, merupakan kelonggaran waktu atas tertundanya proses produksi dikarenakan oleh hambatan yang tidak dapat dihindari, misalnya aliran listrik tiba-tiba mati, macetnya mesin jahit, penggulangan benang jahit, mengasah gunting potong, mengambil pensil dan lain sebagainya. Selain itu ada pula delay untuk setiap operator untuk melakukan koordinasi, mendapatkan informasi ataupun konfirmasi tentang pekerjaan yang sedang maupun akan dilakukan. Untuk menghitung *allowance* digunakan rumus sebagai berikut :

$$Allowance = \frac{Personal\ need + Fatigue + Delay}{Jam\ Kerja\ Efektif} \times 100\% \quad (2.8)$$

Untuk menghitung waktu baku atau waktu standar menggunakan rumus berikut (Wignjosoebroto, 2006:203) :

$$WS = WN \times \frac{100\%}{100\% - Allowance\ \%}$$

Keterangan :

WS = Waktu Standar

WN = Waktu normal

Allowance = Kelonggaran waktu

2.1.2 Penetapan Waktu Longgar

Dalam hal ini waktu longgar dapat diklarifikasikan menjadi tiga macam, yaitu *personal allowance*, *fatigue allowance* dan *delay allowance*.

Personal Allowance adalah jumlah waktu longgar untuk kebutuhan personil dapat ditetapkan dengan jalan melaksanakan aktivitas *time study* sehari kerja penuh atau dengan metode sampling kerja. Untuk pekerjaan-pekerjaan yang relatif ringan dimana operator bekerja selama 8 jam per hari tanpa jam istirahat yang resmi sekitar

2 sampai 5% (atau 10 sampai 24 menit) setiap jari akan dipergunakan untuk kebutuhan-kebutuhan yang bersifat personal ini (Wignjosoebroto S, 2006).

Yang termasuk ke dalam kebutuhan pribadi disini adalah hal-hal seperti minum sekedarnya untuk menghilangkan rasa haus, ke kamar kecil, bercakap-cakap dengan teman sekerja sekedar untuk menghilangkan ketegangan ataupun kejemuhan dalam kerja (Sutalaksana dkk, 2006).

Fatigue Allowance adalah kelelahan fisik manusia bisa disebabkan oleh beberapa penyebab diantaranya adalah karena kerja yang membutuhkan pikiran banyak dan kerja fisik. Untuk pekerjaan-pekerjaan berat, masalah kebutuhan istirahat untuk melepaskan lelah sudah banyak berkurang karena disini sudah mulai diaplikasikan penggunaan peralatan atau mesin yang serba mekanis dan otomatis secara besar-besaran, sehingga mengurangi peranan manusia. Sebagai konsekuensinya maka kebutuhan waktu longgar untuk istirahat melepaskan lelah ini dapat pula dihilangkan (Wignjosoebroto S, 2006).

Delay Allowance adalah keterlambatan atau *delay* bisa disebabkan oleh faktor-faktor yang sulit untuk dihindarkan, tetapi juga bisa disebabkan oleh faktor-faktor yang masih bisa dihindari. Keterlambatan yang terlalu besar/lama tidak diperhitungkan sebagai dasar untuk menetapkan waktu baku (Wignjosoebroto S, 2006).

Menurut Sutalaksana dkk (2006), beberapa contoh yang termasuk ke dalam hambatan tak terhindarkan adalah:

- a. Menerima atau meminta petunjuk kepada pengawas
- b. Melakukan penyesuaian-penyesuaian mesin
- c. Memperbaiki kemacetan-kemacetan singkat seperti mengganti alat potong yang patah, memasang kembali ban yang lepas dan sebagainya.
- d. Mengasah peralatan potong.
- e. Mengambil alat-alat khusus atau bahan-bahan khusus dari gudang.
- f. Mesin berhenti karena matinya aliran listrik.
- g. Hambatan-hambatan karena kesalahan pemakaian alat ataupun bahan.

Apabila ketiga jenis kelonggaran waktu tersebut diaplikasikan secara bersamaan untuk seluruh elemen kerja, maka hal ini bisa menyederhanakan perhitungan yang harus dilakukan.

2.2 Line Balancing

Line Balancing adalah serangkaian stasiun kerja (mesin dan peralatan) yang dipergunakan untuk membuat produk yang biasanya terdiri dari sejumlah area kerja yang ditangani oleh satu pekerja atau lebih. Tujuan utama dari penyusunan Line Balancing ini adalah untuk menyeimbangkan beban kerja yang terdapat pada setiap tempat kerja. Hal ini dilakukan untuk mengurangi efisiensi yang akan terjadi akibat adanya ketidakseimbangan beban kerja.

Persyaratan umum yang harus digunakan dalam suatu keseimbangan lintasan produksi adalah dengan meminimumkan waktu menganggur (*idle time*) dan meminimumkan pula keseimbangan waktu senggang (*balance delay*). Sedangkan tujuan dari lintasan produksi yang seimbang adalah sebagai berikut (Gasperz, 1998):

1. Menyeimbangkan beban kerja yang dialokasikan pada setiap stasiun kerja sehingga setiap stasiun kerja selesai pada waktu yang seimbang dan mencegah terjadinya kemacetan.
2. Menjaga agar pelintasan perakitan tetap lancar dan berlangsung terus menerus dan Meningkatkan efisiensi/produktifitas.

Manajemen industri dalam menyelesaikan masalah *line balancing* harus mengetahui tentang metode kerja, peralatan peralatan, mesin-mesin, dan personil yang digunakan dalam proses kerja. Data yang diperlukan adalah informasi tentang waktu yang dibutuhkan untuk setiap *assembly line* dan *precedence relationship*. Aktivitas-aktivitas yang merupakan susunan dan urutan dari berbagai tugas yang perlu dilakukan, manajemen industri perlu menetapkan tingkat produksi per hari yang disesuaikan dengan tingkat permintaan total, kemudian membaginya ke dalam waktu produktif yang tersedia per hari. Hasil ini adalah *cycle time* yang merupakan

waktu dari produk yang tersedia pada setiap stasiun kerja (*work station*) (Baroto, 2002).

2.2.1 Tujuan Line Balancing

Banyaknya pendapat yang dilontarkan mengenai tujuan keseimbangan lini diantaranya adalah menurut (James L. Rigg, 1983) yang mengatakan untuk meminimumkan waktu menganggur dari operasi yang ditetapkan adalah dengan bekerja menurut prosedur yang berurutan. Pendapat yang hampir sama pula dilontarkan oleh (James M. Moore, 1959) yang mengatakan bahwa tujuan dari keseimbangan lini adalah untuk meminimumkan waktu menganggur pada suatu lini dari seluruh stasiun kerja dengan cara tertentu. Sedangkan tujuan dari lintasan produksi yang seimbang menurut (Gasperz, 1998), adalah sebagai berikut:

1. Menyeimbangkan beban kerja yang dialokasikan pada setiap stasiun kerja sehingga setiap stasiun kerja selesai pada waktu yang seimbang dan mencegah terjadinya kemacetan.
2. Menjaga agar pelintasan perakitan tetap lancar dan berlangsung terus menerus.
3. Meningkatkan efisiensi/produktifitas

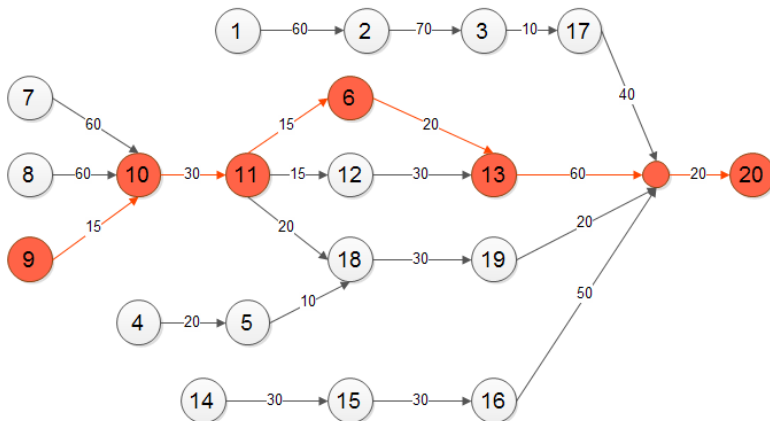
Dengan demikian dapat disimpulkan bahwa tujuan dari keseimbangan lini adalah untuk menghindarkan adanya waktu menganggur dari satu tingkat proses ke tingkat proses lainnya, dengan cara mengefektifkan sejumlah mesin yang ada serta menghindari bertumpuknya bahan dalam proses-proses tertentu, yang pada akhirnya akan memperlancar jalannya proses produksi secara keseluruhan.

a) Precedence Constraint

Precedence constraint adalah batasan terhadap urutan pengerjaan elemen kerja. Kendala precedence dapat digambarkan secara grafis dalam bentuk diagram precedence. Precedence diagram merupakan diagram sederhana sebagai prosedur dasar untuk mengalokasikan elemen-elemen aktivitas dimana memperlihatkan

hubungan suatu aktivitas untuk mendahului aktivitas yang lain. Precedence diagram berfungsi untuk mempermudah penjelasan dari elemen-elemen aktivitas yang ditempatkan dalam suatu stasiun kerja. Dimana pada pada proses assembly ada dua kondisi yang biasa muncul, yaitu (Ginting, 2007,p.11) :

1. Tidak ada ketergantungan dari komponen-komponen dalam proses pengerjaan, jadi setiap komponen mempunyai kesamaan yang sama untuk dikerjakan terlebih dahulu.
2. Apabila suatu komponen telah dipilih terlebih dahulu untuk suatu assembly, maka urutan pengerjaan komponen-komponen lain dimulai. Disinilah dinyatakan batasan precedence untuk pengerjaan komponen-komponen.



Gambar 2.1 Precedence Diagram

Keterangan dari gambar:

1. Simbol lingkaran serta nomor didalamnya untuk mempermudah identifikasi asli dari suatu proses produksi.
2. Tanda panah menunjukkan ketergantungan dalam urutan proses produksi.
3. Angka diatas simbol lingkaran adalah waktu standar yang diperlukan untuk menyelesaikan setiap proses produksi.

b) Waktu Siklus

Waktu siklus (*cycle time*) tersebut merupakan suatu total waktu dari awal hingga akhir dari proses kegiatan, termasuk waktu tunggu. Waktu yang diperlukan untuk melaksanakan elemen-elemen kerja pada umumnya sedikit berbeda dari siklus ke siklus sekalipun operator berkerja pada kecepatan normal atau seragam, karena setaip elemen taip siklus berbeda dan tidak bisa diselesaikan dalam waktu yang sama.

$$\text{Waktu siklus} = \frac{\text{waktu produksi yg tersedia per hari}}{\text{unit yang dihasilkan per hari}}$$

c) Balance Delay

Balance Delay merupakan ukuran dari ketidakefisienan lintasan yang dihasilkan dari waktu menganggur sebenarnya yang disebabkan karena pengalokasian yang kurang sempurna di antara stasiun-stasiun kerja.

$$BD = \frac{(n \times CT) - \sum ti}{(n \times CT)} \times 100\%$$

Dimana :

n : jumlah stasiun kerja

CT : waktu siklus terbesar dalam stasiun kerja

$\sum ti$: jumlah waktu operasi dari semua operasi

ti : waktu operasi

BD : balance delay (%)

d) Line Efficiency

Line Efficiency merupakan rasio dari total waktu stasiun kerja dibagi dengan siklus dikalikan jumlah stasiun kerja. *Line Efficiency* dapat menunjukkan tingkat efisiensi suatu lintasan.

$$LE = \frac{\sum Sti}{(K \times CT)} \times 100 \%$$

Dimana :

K : jumlah stasiun kerja

CT: waktu siklus terbesar dalam stasiun kerja

Sti : waktu stasiun dari stasiun ke-1

e) Workstation

Work Station merupakan tempat pada lini perakitan di mana proses perakitan dilakukan. Setelah menentukan interval waktu siklus, maka jumlah stasiun kerja yang efisien dapat ditetapkan dengan rumus.

$$Kmin = \frac{\sum ti}{(CT)}$$

Dimana :

Kmin : jumlah stasiun kerja minimum

CT : waktu siklus terbesar dalam stasiun kerja

ti : waktu operasi / elemen (i=1,2,3,4,...,n)

2.2.2 Metode Umum Line Balancing

Metode Umum Line Balancing Terdapat beberapa metode yang digunakan untuk menyeimbangkan lintasan produksi, diantaranya yaitu :

1. Metode Analitik (Matematik), Merupakan metode yang dapat menghasilkan suatu solusi optimal. Contoh: Branch and Bound.

2. Metode Heuristik. Metode Heuristik bersal dari kata Yunani yang berarti menemukan. Model heuristik ini pertama kali dilakukan oleh Simon dan Newil untuk menggambarkan pendekatan tertentu untuk memecahkan masalah dan membuat keputusan. Model Heuristik menggunakan aturan-aturan yang logis dalam memecahkan masalah.

2.2.3 Langkah-langkah Pemecahan Line Balancing

Menurut Gaspersz (2004), terdapat sejumlah langkah pemecahan masalah line balancing. Berikut ini merupakan langkah-langkah pemecahan masalah adalah sebagai berikut :

1. Mengidentifikasi tugas-tugas *individual* atau aktivitas yang akan dilakukan.
2. Menentukan waktu yang dibutuhkan untuk melaksanakan setiap tugas itu.
3. Menetapkan *precedence constraints*, jika ada yang berkaitan dengan setiap tugas itu.
4. Menentukan *output* dari *assembly* line yang dibutuhkan.
5. Menentukan waktu total yang tersedia untuk memproduksi output.
6. Menghitung *cycle time* yang dibutuhkan, misalnya: waktu diantara penyelesaian produk yang dibutuhkan untuk menyelesaikan output yang diinginkan dalam batas toleransi dari waktu (batas waktu yang yang diijinkan).
7. Memberikan tugas-tugas kepada pekerja atau mesin.
8. Menetapkan minimum banyaknya stasiun kerja (*work stasion*) yang dibutuhkan untuk memproduksi output yang diinginkan.
9. Menilai efektifitas dan efisiensi dari solusi.
10. Mencari terobosan-terobosan untuk perbaiki proses terus- menerus (*continous process improvement*).

2.2.4 Istilah-istilah Line Balancing

Ada beberapa istilah yang lazim digunakan dalam line balancing. Berikut adalah istilah-istilah yang dimaksud (Baroto, 2002) :

1. Precedence diagram
2. Assemble Product
3. Keseimbangan Waktu Senggang (Balance Delay)
4. Efisiensi stasiun kerja merupakan rasio antara waktu operasi tiap stasiun kerja (W_i) dan waktu operasi stasiun kerja terbesar (W_s).
5. Line efficiency merupakan rasio dari total waktu stasiun kerja dibagi dengan siklus dikalikan jumlah stasiun kerja atau jumlah efisiensi stasiun kerja dibagi jumlah stasiun kerja.
6. Work Station merupakan tempat pada lini perakitan dimana proses perakitan dilakukan.
7. Smoothes index (SI) adalah suatu indeks yang menunjukkan kelancaran relatif dari penyeimbangan lini perakitan tertentu

2.2.5 Metode Ranked Positional Weight (RPW)

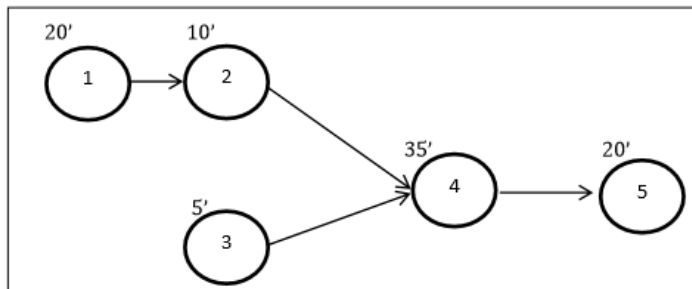
Menurut Gaspersz (1998), fokus penyeimbangan lintasan adalah pada upaya meminimumkan waktu menganggur (idle time) dan menyeimbangkan waktu senggang (balance delay) sehingga permasalahan ini dicoba diselesaikan dengan metode Ranked Positional Weight (RPWM). Menurut Tam (1999) model ini dikembangkan oleh Helgesson-Birnie pada tahun 1961. Dyah (2008) menjelaskan pula bahwa metode ini dianggap mampu memecahkan permasalahan pada keseimbangan lini dan menemukan solusi dengan cepat.

Metode *Renked Positional Weight* (RPW) merupakan metode gabungan antara metode *Large Candidate Ruler* dengan metode *region approach*. Nilai RPW merupakan perhitungan antara elemen kerja tersebut dengan posisi masing-masing elemen kerja dalam *precedence diagram*. Langkahlangkah dari metode RPW adalah sebagai berikut :

- Membuat precedence diagram sesuai dengan keadaan yang sebenarnya.
- Menghitung waktu siklus.
- Membuat lintasan berdasarkan precedencediagram.
- Menentukan positional Weight (bobot posisi) untuk setiap elemen pekerjaannya dari suatu operasi dengan memperhatikan precedence diagram. Berikut cara menentukan bobot posisinya:

$RPW = Waktu\ proses\ operasi + Waktu\ proses\ operasi\ berikutnya$

Contoh:



Gambar 2.2 *Precedence Diagram*

Penentuan bobot posisi :

$$\text{Bobot posisi operasi 1} = 20 + 10 + 35 + 20 = 85'$$

$$\text{Bobot posisi operasi 2} = 10 + 35 + 20 = 65'$$

$$\text{Bobot posisi operasi 3} = 5 + 35 + 20 = 60'$$

Dan seterusnya

- Urutkan elemen operasi berdasarkan bobot posisi yang telah didapatkan pada langkah kedua. Pengurutan dimulai dari elemen operasi yang memiliki bobot posisi terbesar.

- f. Jika pada stasiun kerja terdapat waktu yang berlebihan (waktu stasiun kerja melebihi waktu maksimum yang telah ditetapkan), maka pindahkan elemen operasi terakhir ke stasiun berikutnya.
 - g. Ulangi langkah ke 3 dan ke 4 diatas sampai seluruh element operasi telah ditempatkan kedalam stasiun kerja.
 - h. Menghitung jumlah stasiun kerja minimum.
 - i. Membuat flow diagram untuk stasiun kerja minimum tersebut lalu lakukan pembebanan operasi pada stasiun kerja mulai dari bobot operasi terbesar sampai dengan terkecil, dengan kriteria total waktu operasi lebih kecil dari waktu siklus yang diinginkan.
 - j. Melakukan trial and error untuk mendapatkan efisiensi lintasan yang paling tinggi.
 - k. Menghitung balance delay lintasan.
1. Perhitungan Ranked Positional Weight (RPW)
- a. Waktu Siklus : Merupakan waktu yang dibutuhkan untuk memproduksi suatu produk mulai dari bahan baku awal diproses di tempat kerja.
 - b. Waktu Normal : Waktu normal merupakan waktu yang dibutuhkan oleh operator dalam kondisi wajar atau kemampuan rata-rata untuk memproduksi secara normal.
 - c. Waktu Baku : Waktu baku merupakan waktu yang dibutuhkan oleh operator untuk melakukan pekerjaan sesuai standar.

2.2.6 Metode Region Approach (Killbridge Wester Heursitik)

Killbridge Wester adalah metode yang dirancang oleh M. Killbridge dan L. Wester sebagai pendekatan lain untuk mengatasi permasalahan keseimbangan lintasan perakitan. Metode ini dilakukan dengan pengelompokan tugas kedalam sejumlah kelompok yang mempunyai tingkat keterhubungan yang sama. Langkah-langkah yang digunakan sebagai berikut.

1. Buat *precedence* diagram dari persoalan yang dihadapi. Bagi tiap elemen kerja dalam diagram tersebut ke dalam kolom dari kiri ke kanan. Kolom I adalah elemen-elemen kerja yang tidak memiliki elemen kerja pendahuluan. Kolom II adalah elemen-elemen kerja yang memiliki pendahulaun di kolom I. Begitu seterusnya dengan kolom selanjutnya.
2. Tempatkan elemen-elemen kerja ke stasiun kerja sehingga total waktu elemen kerja tidak melebihi waktu siklus.
3. Bila penempatan suatu elemen kerja mengakibatkan total waktu elemen kerja melebihi waktu siklus maka elemen kerja tersebut ditempatkan di stasiun kerja berikutnya.
4. Ulangi langkah 3 dan 4 sampai seluruh elemen kerja ditempatkan.

2.3 Simulasi Sistem Diskrit

Semua aktivitas di dunia manufaktur didominasi oleh peristiwa yang bersifat diskrit dan probabilistik. Munculnya produk-produk cacat, tidak menentunya kedatangan bahan baku, peristiwa break down mesin, proses welding yang dilakukan oleh operator, aktivitas inspeksi, jumlah demand dan lain sebagainya merupakan contoh-contoh yang nyata akan kedua sifat tersebut. Sebaliknya pada industri penyulingan minyak, petrokimia dan industri lain sejenis hampir seluruh prosesnya bersifat kontinyu sehingga tingkat probabilitasnya jauh lebih rendah dibanding dengan industri manufaktur. Kedua persoalan ini tingkat kerumitannya sangat jauh berbeda, yaitu persoalan manufaktur jauh lebih rumit dibanding dengan persoalan industri kontinyu karena itu pada bab ini akan dipaparkan konsep dasar kediskritan dalam simulasi. Kelak dengan konsep dasar ini dapat dipakai sebagai modal awal pembuatan program simulasi.

Guna menciptakan nuansa kediskritan dan keprobabilistikan sistem dalam simulasi, kemampuan untuk membangkitkan bilangan acak menjadi kebutuhan vital. Dengan bilangan-bilangan acak ini dapat dilahirkan pula berbagai macam variabel acak yang mengikuti pola distribusi probabilitas yang sesuai. Namun dalam hal-hal tertentu perlu kejelian didalam menafsirkan suatu sistem apakah tergolong diskrit

atau kontinyu, karena sifat kediskritan suatu sistem bisa terlihat sebagai perilaku kontinyu. Suatu misal, dalam sistem lalu lintas, untuk lintasan/jalan yang lengang sampai dengan ramai, kedatangan kendaraan bisa dikatakan diskrit. Akan tetapi untuk lintasan yang sangat padat, sehingga kendaraan nyaris berimpitan, maka tidak terlalu salah kalau dikatakan sebagai sistem yang kontinyu. Nampak bahwa perubahan ini dipengaruhi oleh kenisbian pengamatan. Hal ini dapat juga terjadi pada dunia industri manufaktur. Pada mesin (stasiun kerja) penghancur batu misalnya, kedatangan bongkahan batu besar akan bersifat diskrit karena mesin akan menghancurkan bongkahan itu satu per satu dan proses penghancurannya cukup lama, begitu pula saat batu kerikil keluar maka juga bersifat diskrit. Akan tetapi karena kerikil yang keluar dalam jumlah banyak dan satu dengan yang lainnya keluar secara berimpitan maka dapat dipandang sebagai kontinyu.

2.3.1 Sistem Diskrit dan Sistem Kontinyu

Satu hal yang sangat penting dalam mengkaji suatu sistem adalah mengetahui perilaku sistem, yaitu aktivitas sistem yang dinyatakan dalam bentuk keluaran sebagai perwujudan respon sistem atas rangsangan-rangsangan yang datang. Ada banyak macam perilaku yang dimunculkan sistem satu diantaranya adalah perilaku yang bersifat diskrit. Kediskritan sistem dapat dilihat dari perubahan keadaan (state) sistem dari waktu ke waktu. Jika perubahan state yang terjadi hanya pada titik-titik waktu tertentu dan bukannya pada setiap titik waktu maka dikatakan sebagai state yang diskrit. Jika sebaliknya, maka dikatakan kontinyu.

Perubahan state dari sistem tidak muncul dengan sendirinya tanpa sebab. Penyebab utama sehingga muncul perubahan state adalah munculnya kejadian (event). Karena itu sifat diskrit pada state sistem juga disebabkan oleh sifat diskrit pada eventnya. Pada kebanyakan sistem kapan datangnya event tidak menentu sehingga kapan terjadi perubahan state juga bersifat tidak menentu. Kalau ketidakmenentuan itu dapat diduga dengan pola-pola probabilistik, maka sistem yang demikian dikatakan sebagai sistem diskrit yang probabilistik.

Simulasi dengan kejadian/event yang diskrit (discrete event simulation) atau disebut simulasi sistem diskrit akan mensimulasikan sistem-sistem yang mana perubahan statenya terjadi pada titik-titik waktu yang diskrit.. Kemampuan pengidentifikasi sifat perilaku sistem dan kemudian mengejawentahkan sifat perilaku itu ke simulasi inilah sebenarnya yang menjadi konsep dasar dari pada simulasi sistem dengan event diskrit probabilistik. Sebaliknya sistem dimana perubahan statenya berlangsung secara kontinyu, maka dikatakan sebagai sistem kontinyu. Sistem pengendalian posisi satelit misalnya, akan selalu memonitor di posisi mana keberadaan satelit. Perubahan posisi satelit dari tempat kedudukan semula ke tempat kedudukan berikutnya terjadi secara kontinyu, maka respon dari sistem pengendalian itu juga bersifat kontinyu. Guna menyelesaikan persoalan yang bersifat kontinyu seperti itu juga harus menggunakan simulasi kontinyu.

Pada sistem diskrit akan dijumpai variabel-variabel diskrit dan variabel-variabel kontinyu. Guna menandai apakah suatu variabel berbentuk diskrit atau kontinyu dapat diperiksa dari bagaimana nilai dari variabel itu didapat. Variabel kontinyu nilainya didapatkan dengan cara mengukur dengan menggunakan alat ukur misalnya berat kemasan, tekanan udara, waktu antar kedatangan mesin rusak, waktu proses dan lain sebagainya adalah variabel-variabel kontinyu sedang variabel diskrit nilainya diperoleh dengan cara mencacah (menghitung), banyaknya produk cacat, jumlah sumber daya manusia yang dibutuhkan, banyaknya mesin yang break down dan lain-lainnya adalah variabel diskrit.

Dalam realita sering kali dijumpai sistem yang sifat kediskritannya bisa berubah menjadi kontinyu atau sebaliknya. Perubahan itu bisa terjadi karena tujuan kajian sistem yang berbeda, pendekatan kajian yang berbeda atau tingkat agregasi sistem yang diambil. Pada stasiun kerja penghancur batu misalnya, kajian kerikil yang keluar secara diskrit bisa berubah menjadi kontinyu karena adanya anggapan bahwa kerikil-kerikil itu keluar secara berimpitan (interval waktu yang mendekati nol) begitu pula pada kajian sistem lalu lintas, tingkat kedatangan kendaraan yang tinggi sekali dapat dianggap sebagai aliran yang kontinyu. Sedang tingkat

pertumbuhan/kelahiran bayi secara global dipandang sebagai peristiwa kontinyu karena tingkat agregasi sistem yang sangat tinggi. Oleh karena itu identifikasi awal atas sistem nyata yang menjadi kajian sangat kritis sifatnya

2.3.2 Mekanisme Penggeseran Jam Simulasi

Keuntungan dari pada simulasi adalah kecepatan dalam menirukan sistem sebenarnya (*real system*) dari kondisi awal hingga diperoleh jawaban atas sistem itu di penghujung waktu sebagai cerminan hasil akhir simulasi. Bahkan pengulangan siklus proses tersebut dalam jumlah yang banyak sekali dapat dilakukan dengan cepat sekali. Masing-masing pengulangan itu akan dilakukan selama selang waktu tertentu. Sementara itu pada kebanyakan persoalan simulasi adalah mengamati dan mencatat bagaimana perubahan state sistem terjadi disepanjang waktu simulasi menjadi keharusan. Untuk itu penggambaran penggeseran waktu simulasi dari awal hingga akhir menjadi sangat penting untuk dilakukan.

Jam simulasi merupakan salah satu besaran yang penting, karena dengan jam ini akan diketahui kapan *event* selanjutnya terjadi dan juga dapat diketahui performansi sistem pada saat itu yang pada akhirnya akan diketahui pula berapa lama simulasi dijalankan. Pada hakekatnya jam simulasi ini adalah sama seperti lazimnya jam, yaitu berubah setiap saat secara periodik (biasanya setiap satu detik). Untuk itu dalam simulasi pun ada mekanisme yang dipakai untuk menjalankan jam simulasi. Mekanisme itu juga bekerja secara periodik, namun tidak selalu/harus setiap satu detik seperti halnya jam biasa. Ada dua macam perioda penggeseran, yaitu pertama dengan perioda penggeserannya yang konstan dan tertentu. (*fixed-increment time advance*) dan yang kedua dengan perioda penggeseran berdasarkan pada setiap munculnya *event* yang berikutnya (*next event time advance*). Masing-masing cara mempunyai kelebihan dan kekurangannya, namun cara yang kedua yang lazim dipakai.

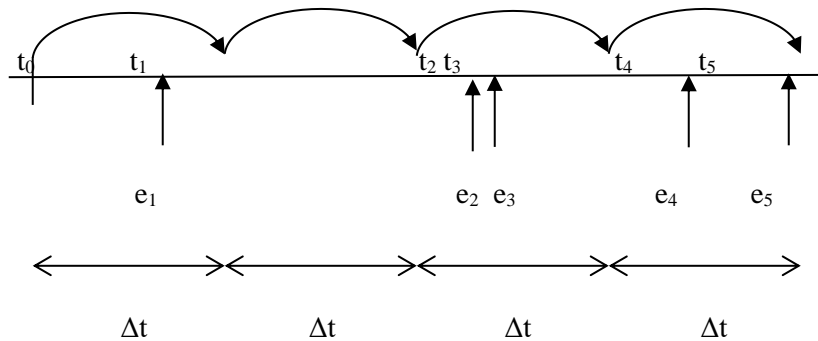
2.3.3 Penggeseran Jam Dengan Perioda Konstan

Guna menjelaskan mekanisme penggeseran waktu dalam simulasi, Perhatikan sistem pemberangkatan kereta api. Kereta bisa dianggap sebagai pelayan yang melayani penumpang yang akan bepergian. Jadwal pemberangkatannya sudah tertentu misal setiap jam 07.00 pagi. Meskipun jadwal sudah tertentu, akan tetapi kedatangan penumpang tidak pasti. Ada yang datang jauh sebelum jam pemberangkatan ada yang beberapa menit sebelum pemberangkatan bahkan banyak pula yang datang tepat saat pemberangkatan. Kapanpun penumpang-penumpang itu datang, semuanya akan diberangkatkan (dilayani) jam 07.00 pagi dan mereka tidak mendongkol hatinya kalau baru diberangkatkan jam 07.00 pagi meskipun mereka sudah datang puluhan menit sebelumnya. Selanjutnya pemberangkatan kereta berikutnya adalah esuk hari dengan jam pemberangkatan yang sama pula. Coba bandingkan dengan pelayanan di loket teller sebuah bank yang melayani nasabah-nasabahnya sungguh sangat berbeda. Metoda penggeseran jam simulasi dengan perioda konstan mirip dengan pemberangkatan kereta api tersebut.

Penggeseran jam secara konstan artinya jam simulasi akan digeser (dimajukan) kedepan secara periodik dengan lebar penggeseran Δt yang konstan. Nilai dari Δt bisa berharga satu detik, sepuluh detik, dua jam atau berapapun. Setiap setelah penggeseran jam simulasi dilakukan, selalu segera diikuti pemeriksaan atas event-event yang terjadi selama interval waktu Δt . Jika ada satu atau lebih event yang telah terjadi dalam interval waktu tersebut, maka event-event tersebut segera diproses di penghujung interval waktu Δt tersebut.

Perhatikan gambar (2.3). Anak panah lengkung menyatakan penggeseran jam simulasi sebesar Δt , sedang t_i menyatakan titik waktu dimana kejadian ke i , yaitu e_i telah terjadi. Dalam interval waktu $[0, \Delta t]$ telah terjadi event e_1 pada waktu t_1 . Event ini baru segera diproses di akhir interval $[0, \Delta t]$, sehingga telah terjadi manipulasi waktu kedatangan event dari t_1 menjadi ke akhir interval $[0, \Delta t]$. Tidak ada event yang muncul selama interval $[\Delta t, 2\Delta t]$ akan tetapi pemeriksaan tetap dilakukan padahal itu tidak perlu. Selanjutnya dalam interval $[2\Delta t, 3\Delta t]$ dua event

terjadi dan baru diproses di akhir interval $[2\Delta t, 3\Delta t]$ begitu seterusnya. Kalau dikaitkan dengan sistem pemberangkatan kereta api, maka Δt berarti interval waktu antar pemberangkatan (24 jam) sedang t_i adalah jam kedatangan penumpang ke i , atau pada jam t_i itulah event ke i (penumpang ke i) terjadi/datang.



Gambar 2.3 Penggeseran Jam Simulasi Secara Konstan

Dengan cara penggeseran ini akan mengalami dua persoalan, yaitu yang pertama adalah keharusan memeriksa event pada setiap interval penggeseran waktu meskipun selama interval itu tidak muncul event. Yang kedua telah terjadi manipulasi waktu kedatangan event, yaitu waktu event telah dimundurkan ke akhir interval, padahal eventnya sendiri terjadi sebelum titik ujung interval itu. Semakin besar interval yang ditetapkan semakin besar kesalahan itu. Kesalahan ini memang bisa diminimalkan, yaitu dengan memperkecil lebar interval penggeseran akan tetapi konsekuensinya adalah persoalan yang pertama menjadi lebih serius karena semakin sering melakukan pemeriksaan. Dengan alasan inilah maka metoda ini jarang dipakai dan walaupun dipakai benar-benar terbatas, yaitu hanya pada sistem yang benar-benar semua eventnya terjadi di setiap akhir perioda penggeseran. Misalnya sistem pelaporan akuntansi yang selalu diberlakukan pada akhir tahun.

2.3.4 Penggeseran Jam Berdasarkan Event Berikutnya

Berbeda dengan pelayanan/pemberangkatan penumpang kereta api, pelayanan nasabah bank akan sesegera mungkin dilakukan setiap nasabah datang

dan tidak pernah ada pelayanan bersama-sama seperti pemberangkatan penumpang kereta api dan tidak ada pelayanan nasabah bank yang bersifat periodik setiap satu jam misalnya. Oleh karena itu cara kedua ini akan menggeser jam simulasi ke depan sejauh kapan event tercepat yang akan terjadi. Dikatakan tercepat, sebab selalu akan ada banyak event yang akan terjadi di masa akan datang. Mengingat bahwa kedatangan event tercepat yang akan datang adalah tidak menentu (*probabilistik*), maka lebar interval penggeserannya juga tidak menentu, ada kalanya sangat sempit dan adakalanya lebar sekali. Perhatikan gambar 4.2. Misalkan dibuatkan notasi-notasi sebagai berikut:

t_i = waktu kedatangan customer ke i ($t_0 = 0$).

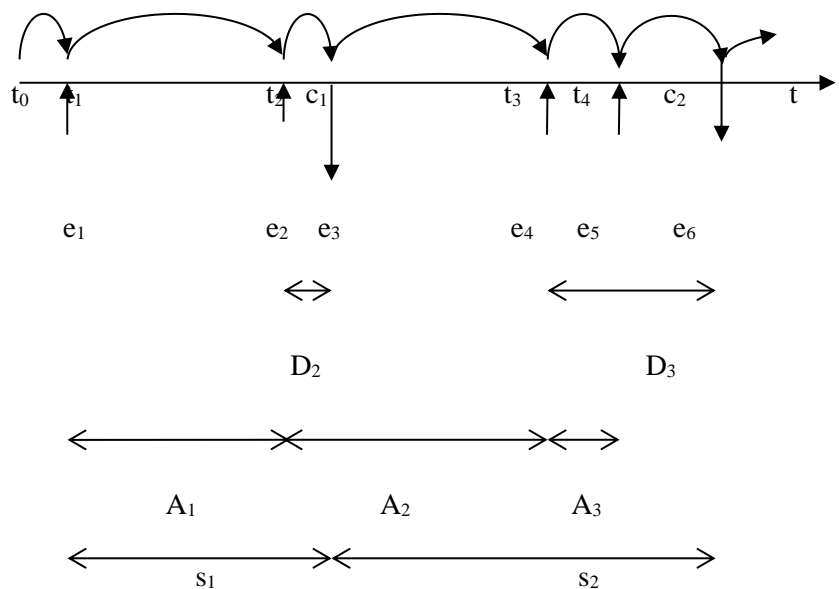
e_i = kejadian event ke i (segala jenis event).

$A_i = t_i - t_{i-1}$ = waktu antar kedatangan customer ke $i-1$ dan customer ke i .

S_i = waktu lamanya server melayani customer ke i .

D_i = waktu tunggu (delay) customer ke i dalam antrian.

$c_i = t_i + D_i + S_i$ = waktu bahwa customer ke i selesai dilayani dan pergi.



Gambar 2.4 Penggeseran waktu berdasarkan event berikutnya

Pada saat jam simulasi sama dengan t_0 , kondisi awal server yaitu teller bank dalam keadaan idle. Event tercepat pertama e_1 , yaitu kedatangan nasabah terjadi waktu t_1 kemudian, maka jam simulasi dimajukan ke t_1 . Pada saat yang sama server mulai melakukan pelayanan pada pendaang pertama tersebut. Disaat pelayanan pertama belum selesai, telah datang nasabah berikutnya yaitu event e_2 pada waktu t_2 , sehingga jam simulasi dimajukan ke t_2 . Selanjutnya ternyata sebelum ada kedatangan ke tiga pada waktu t_3 , ternyata pelayanan pertama sudah selesai. Oleh karena itu sebelum dimajukan ke t_3 , jam simulasi dimajukan lebih dulu ke c_1 . Pada saat pelayanan pertama selesai ternyata sudah ada antrian, sehingga pelayanan berikutnya (nasabah kedua) langsung diberikan. Dan seterusnya proses penggeseran jam simulasi dilakukan terus sampai kondisi selesai atau kondisi lain yang mengisaratkan bahwa simulasi harus diakhiri yaitu saat jam kantor tutup atau saat semua nasabah yang datang sebelum jam kantor tutup terlayani.

Dari dua metoda penggeseran waktu di atas, nampak bahwa metoda penggeseran waktu yang berdasarkan event lebih bersifat general, yaitu metoda itu bisa juga memerankan diri dalam penggeseran waktu secara periodik kalau memang event-event yang muncul pada sistem nyatanya bersifat demikian. Dan hal itu dapat dilakukan tanpa harus ada perubahan-perubahan yang dilakukan. Dengan begitu cara penggeseran yang kedua itulah yang banyak dipakai orang untuk membangun simulasi.

2.3.5 Bilangan Acak

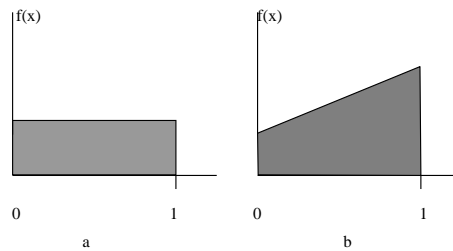
Semakin banyak komponen sistem berperilaku probabilistik, maka semakin tinggi derajat ketidakpastiannya. akan tetapi justru fenomena semacam itu yang sering dijumpai pada kebanyakan sistem nyata. Persoalan penjadwalan produksi, pengendalian persediaan, pengaturan loket-loket layanan di bank, kajian kelayakan pabrik dan masih banyak lagi persoalan-persoalan baik besar maupun kecil yang bersifat probabilistik. Lantas pertanyaan mendasar yang perlu di jawab adalah

bagaimana sebenarnya guna menggambarkan fenomena probabilistik ini dalam simulasi?. Tidak terlalu sulit untuk menggambarkan fenomena tersebut dalam simulasi yaitu, dengan membangkitkan bilangan acak (random number). Selanjutnya dengan bilangan acak yang telah dibangkitkan ini dapat dipakai untuk membangkitkan variabel acak yang dapat menggambarkan keprobabilistikan suatu sistem, apakah keprobabilistikan itu mengikuti pola yang sudah baku seperti pola Normal, Weibull, Gamma, Binomial, Geometrik atau bahkan membentuk pola-pola lain yang belum dikenal selama ini.

Untuk menggambarkan fenomena probabilistik dalam simulasi, bilangan acak menjadi kuncinya. bilangan acak adalah bilangan sembarang tetapi tidak sembarangan. Ada beberapa kriteria dasar yang membuat bilangan sembarang tersebut adalah bilangan acak yang tidak sembarangan, yaitu:

a. Bilangan acak tersebut harus mempunyai distribusi serba sama (uniform)

Semua bilangan acak yang ada mempunyai kemungkinan yang sama untuk dipilih. Perhatikan bilangan-bilangan yang berada diantara 0.0 sampai dengan 1.0, maka sebenarnya akan ada bilangan dalam jumlah banyak sekali (tak berhingga) yang dapat diambil sebagai bilangan acak baik secara acak kontinu maupun acak diskrit, maka masing-masing bilangan itu mempunyai peluang terambil sama besar. Distribusi kontinu yang sederhana ini ternyata menjadi distribusi yang sangat penting guna memperoleh variabel acak yang mempunyai distribusi tertentu seperti normal, gamma, binomial dan lain sebagainya, yaitu dengan mentransformasikan bilangan acak tersebut ke distribusi lain yang diinginkan.



Gambar 2.5 Distribusi bilangan acak

b. Masing-masing bilangan acak tidak bergantung

Artinya terambilnya suatu bilangan acak tidak dipengaruhi bilangan acak sebelumnya dan tidak mempengaruhi munculnya bilangan acak berikutnya. sehingga bilangan acak yang terambil tersebut semata karena sifat keacakan pengambilan. Bila mengamati hasil suatu proses produksi model batc, akan dijumpai sejumlah produk yang cacat. banyaknya produk cacat setiap batc produksi bisa sama dan bisa tidak sama. Jumlah produk cacat dalam suatu batc tidak ditentukan/dipengaruhi oleh banyaknya produk cacat dalam batc yang lain. Jumlah itu benar-benar independent. Ini adalah sebuah contoh yang baik sekali untuk mengilustrasikan akan pengertian bilangan acak yang tidak bergantung. Hal ini akan berbeda sekali manakala jumlah kecacatan itu dikaitkan dengan kecepatan kerja operator. Semakin cepat kerja seorang operator, maka akan semakin cenderung melakukan kesalahan dan kesalahan itu semakin menjadi-jadi manakala kecepatannya melampaui batas kecepatan alamiah operator. Dengan begitu jumlah produk cacat sangat tergantung pada seberapa cepat operator bekerja.

2.3.6 Metoda Pembangkitan Bilangan Acak

Guna memperoleh bilangan acak dapat diperoleh dari mana saja dan dengan cara apa saja. Tidak menjadi persoalan yang mendasar apakah bilangan itu berasal dari angkaangka pada sebuah alat ukur, angka-angka yang melekat pada pelat nomer kendaraan, angka-angka nomer rumah sepanjang jalan atau angka apapun. Sementara itu bagaimana untuk memperolehnya pada awalnya juga tidak ada

keharusan dengan menggunakan cara tertentu, yang terpenting adalah bahwa bilangan-bilangan acak yang dihasilkan itu memenuhi kriteria/syarat.

Metoda pembangkitan bilangan acak mempunyai sejarah yang cukup panjang sehingga banyak sekali cara bermunculan bagaimana sebenarnya bilangan acak itu diperoleh. Ada yang melakukan dengan cara yang sederhana yaitu, melempar dadu berkali-kali kemudian mencatat semua mata dadu yang muncul sebagai cerminan bilangan acak. Ada pula dengan memutar bola-bola angka dalam permainan lotere, memutar piringan bilangan dan lain sebagainya masih banyak lagi, bahkan boleh-boleh saja muncul lagi metoda-metoda lain yang barang kali sangat jauh berbeda dari metodametoda yang sudah ada. Mengingat pentingnya peranan bilangan acak pada waktu itu sehingga banyak orang berupaya menciptakan alat-alat pembangkit bilangan acak dalam bentuk yang lebih sempurna, maka munculah mesin-mesin pembangkit bilangan acak yang lebih maju seperti ERNIE (*Electronic Random Number Indicator Equipment*). Bahkan dengan kemajuan dunia komputer yang semakin canggih, ternyata pembangkitan bilangan acak ini masih merupakan salah satu materi pemikiran yang sangat serius. Karena ternyata pemakaian bilangan acak ini juga semakin merambah pada perangkatperangkat lunak computer.

a. Metoda Midsquare

Metoda ini diperkenalkan oleh Neumann dan Metropolis di tahun 1940-an. Guna memperoleh bilangan acak, konsep dasar dari metoda ini adalah dengan cara mengambil bagian tengah dari hasil kuadrat suatu bilangan, selanjutnya bilangan acak berikutnya diperoleh dari bilangan tengah dari hasil kuadrat bilangan tengah sebelumnya. Begitu seterusnya akan diperoleh sejumlah bilangan acak yang diinginkan. Sebagai contoh, guna membangkitkan bilangan acak dalam interval $U(0,1)$ perhatikan tabel (2.1). Dengan menetapkan bilangan awal sebanyak empat angka misal bilangan 7182, kemudian angka tersebut dikuadratkan sehingga menjadi 51581124 selanjutnya ambil empat angka yang ditengah yaitu, 5811 sebagai bilangan empat angka berikutnya.

Dari bilangan pertama diperoleh bilangan acak 0.5811. Sementara itu dari bilangan kedua diperoleh bilangan acaknya 0,7677. Dan seterusnya dengan cara yang sama akan diperoleh n buah bilangan acak. Secara intuitiv cara ini memberikan upaya pengacakan angka-angka dengan baik guna memperoleh angka berikutnya, sehingga bisa membangkitkan bilangan acak secara baik. Kenyataanya cara ini tidak bekerja dengan baik. Satu persoalan serius adalah bahwa cara ini mempunyai kecenderungan menuju nol (ganti Zo dengan 1009 misalnya). Ternyata cara ini menjadi kurang menarik, sebab disamping mempunyai kecenderungan nilai, juga bahwa nilai-nilai acak U_i berikutnya sebenarnya dapat diduga manakala telah diketahui Z_0 (nilai awalnya).

Table 2.1 Pembangkitan Bilangan Acak Metoda Mid Square

i	Z_i	Z_i^2	U_i
0	7182	51581124	0.5811
1	5811	33767721	0.7677
2	7677	58936329	0.9363
3	9363	87665769	0.6657
4	6657	44315649	0.3156
5	dan seterusnya		

b. Metoda L C G

Kebanyakan pembangkit bilangan acak yang dipakai orang saat ini adalah Linear Congruential Generators (LCG). Sejumlah urutan bilangan integer $Z_1, Z_2, Z_3, \dots, Z_n$ diperoleh dengan rumus berikut:

$$Z_i = (aZ_{i-1} + c) \pmod{m}$$

Dimana a = konstanta pengali.

c = konstanta pergeseran.

m = konstanta modulus.

Z_0 = bilangan awal (*seed number*).

Dari hubungan di atas terlihat bahwa Z_i selalu merupakan sisa hasil pembagian bilangan $(aZ_{i-1} + c)$ dengan modulus m yang berarti $0 \leq Z_i \leq m-1$ dan bilangan acaknya $U_i(0,1) = Z_i/m$. Guna memperoleh nilai yang positif, maka hendaknya $0 < m$, $a < m$, $c < m$, dan $Z_0 < m$. Sebagai contoh ambil $m = 16$, $a = 5$, $c = 3$ dan $Z_0 = 7$, maka hasil U_i ($i = 1, 2, \dots$).

Table 2.2 Hasil U_i untuk $m = 16$, dengan $a = 5$, $c = 3$ $Z_0 = 7$

i	Z_i	U_i	i	Z_i	U_i
0	7	-	10	9	0.563
1	6	0.375	11	0	0.000
2	1	0.063	12	3	0.188
3	8	0.500	13	2	0.125
4	11	0.688	14	13	0.813
5	10	0.625	15	4	0.250
6	5	0.313	16	7	0.438
7	12	0.750	17	6	0.375
8	15	0.938	18	1	0.063
9	14	0.875	19	8	0.500

Ditunjukkan dalam tabel (2.2). Ternyata dari contoh itu terlihat bahwa untuk nilai U yang ke 17, ke 18 dan seterusnya nilainya merupakan pengulangan nilai U yang ke 1, ke 2 dan seterusnya. Panjang dari rentangan siklus pengulangan itu disebut dengan perioda pembangkitan. Pada metoda LCG nilai Z_i hanya tergantung pada Z_{i-1} , dan selagi nilainya $0 \leq Z_i \leq m$, maka perioda itu paling panjang adalah m dan bila perioda itu sama dengan m , maka LCG mempunyai perioda penuh. Jelas bahwasanya jika pembangkit bilangan acak mempunyai perioda penuh, maka pemilihan Z_0 yang lain juga akan menghasilkan siklus penuh. Perioda yang panjang mempunyai makna bahwa cacah bilangan acak yang ditampilkan juga akan banyak. Jadi semakin panjang perioda itu, maka semakin baik keberadaan metoda itu.

c. **Metoda Pembangkit Bauran (Mixed LCG)**

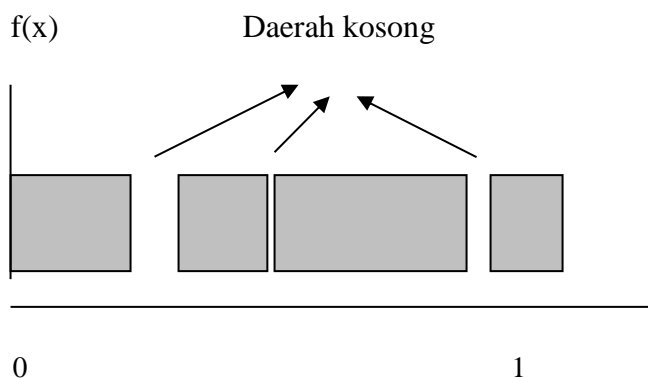
Metoda ini sebagai pengembangan atas metoda LCG, yaitu mencoba memperpanjang perioda siklus pembangkitan bilangan acak disamping meningkatkan kepadatan munculnya bilangan acak dalam perioda itu. Pengembangan ini mengingat bahwa pada proyek simulasi bersekala besar, kebutuhan bilangan acak dalam jumlah besar (bisa mencapai ratusan ribu) merupakan perwujudan akan kebutuhan metoda dengan perioda yang panjang. Semakin panjang perioda dan semakin tinggi kepadatan munculnya bilangan acak, maka konsekuensi logis dari hubungan $Z_i = (aZ_{i-1} + c)(\text{mod } m)$ menghendaki harga m yang semakin besar pula. Namun demikian seberapa besar harga m yang diberikan juga harus memperhatikan kemampuan komputer yang akan dipakai. Karena itu harga m hendaknya sebesar 2^b dengan b adalah jumlah bit dalam satu kata (word) pada komputer yang digunakan.

d. **Pembangkit Multiplikatif (Multiplicative Generators)**

Jika konstanta c pada $Z_i = (aZ_{i-1} + c)(\text{mod } m)$ dipilih harga nol, maka alat pembangkit ini dikenal dengan Multiplicative Generators dan masih dimungkinkan mendapatkan perioda $m-1$ jika harga m dan a diambil dengan hati-hati. Sebenarnya multiplicative LCG sudah lebih dahulu dari pada mixed LCG dan

telah dikaji secara intensif sehingga yang digunakan sampai sekarangpun kebanyakan metoda multiplikativ.

Seperti pada mixed generator, maka pada multiplikativ juga menetapkan harga $m = 2^b$, namun ternyata periodanya tinggal 2^{b-2} atau tinggal 1/4 bagian dari $m-1$ sebagai Z_i . Lebih lanjut sangat disayangkan bahwa dimana Z_i berada ternyata tidak diketahui. Dikawatirkan ada daerah-daerah kosong diantara perioda itu (lihat gambar 4.4). Munculnya daerah-daerah kosong ini berarti bilangan acak tidak menyebar secara merata disepanjang perioda dan secara statistik bilangan tidak berdistribusi serba sama. Dari kesulitan-kesulitan ini kemudian menetapkan m sebagai bilangan integer terbesar di bawah 2^b . Misalkan



nilai $b = 31$, maka $m = 2^b - 1$, yaitu sama dengan 2.147.483.647. Lebih lanjut cara ini lebih dikenal dengan metoda Prime Modulus Multiplicative LCG (PMMLCG).

2.3.7 Pengujian Atas Pembangkit Bilangan Acak

Di awal pembicaraan tentang pembangkitan bilangan acak, telah diungkap bahwa bilangan acak adalah sembarang bilangan yang bisa dibangkitkan dengan cara atau alat apapun, namun yang harus menjadi persyaratan utama dari pada bilangan acak yang dihasilkan, adalah bahwa bilangan-bilangan acak yang dihasilkan itu hendaknya bilangan-bilangan yang IID $U(0,1)$. Karena itu pengujian atas alat

pembangkit bilangan acak dimaksudkan untuk mengetahui apakah alat pembangkit tersebut telah mampu menghasilkan bilangan acak yang IID $U(0,1)$ atau tidak.

Ada dua macam alat uji yang bisa dipakai untuk menguji alat pembangkit, yaitu uji empiris (*empirical test*) yang berbasis pada uji statistik dan uji teoritis (*theoretical test*) yang mengacu pada parameter numeris dari pembangkit. Namun dalam buku hanya akan disampaikan alat uji jenis empiris. Barangkali cara yang lazim digunakan dalam uji empiris adalah menetapkan U_i sebagai hasil keluaran alat pembangkit bilangan acak guna mengetahui seberapa dekat bilangan-bilangan random itu menyerupai variabel acak yang IID $U(0,1)$. Tersedia beberapa macam alat uji jenis empiris yang akan dibicarakan diantaranya, uji Chi-Square, uji Kolmogorov-Smirnov (K-S), uji Seri dan uji Runs-up.

2.3.8 Pembangkitan Variabel Acak

Sifat probabilistik pada sistem nyata selalu mempunyai pola distribusi probabilistik tertentu. Distribusi probabilistik tertentu yang dimaksud adalah suatu distribusi yang menggambarkan bagaimana sebenarnya pola munculnya ketidakpastian dari pada sistem nyatanya. Distribusi itu bisa bersifat kontinyu dengan bentuk pola yang baku seperti distribusi Normal, Eksponensial, Gamma dan lain sebagainya dan bisa juga bersifat diskrit dengan bentuk pola yang baku pula seperti Binomial, Geometrik, Bernoulli dan lain sebagainya bahkan bisa membentuk pola lain yang belum baku sama sekali atau belum ada sebelumnya

Dalam simulasi komputer penggambaran fenomena probabilistik dengan pola-pola itu mutlak diperlukan. Lantas bagaimana guna menggambarkan pola-pola itu dalam simulasi ?. Hal itu dapat digambarkan dengan menggunakan variabel acak yang mempunyai pola distribusi seperti yang diinginkan ke dalam perhitungan komputer. Selanjutnya bagaimana mendapatkan variabel acak yang mempunyai pola distribusi tertentu itu ?. Seperti pada pembahasan sebelumnya telah diketahui bahwa sebagai bahan utama pembangkitan variabel acak adalah bilangan acak $U(0,1)$. Dari bilangan acak ini kemudian ditransformasikan ke suatu distribusi

probabilitas tertentu, sehingga diperoleh variabel acak yang berdistribusi tertentu pula. Akhirnya variabel acak inilah yang akan diolah dalam simulasi, sehingga gambaran fenomena probabilistik dapat diwujudkan dalam simulasi komputer.

Ada beberapa algoritma pembangkitan variabel acak yang dapat digunakan, namun untuk menetapkan pilihan algoritma mana yang akan dipakai harus difikirkan apakah algoritma itu memang cocok untuk mendapatkan variabel acak yang berdistribusi tertentu, sebab pemilihan algoritma yang salah akan membuat upaya pembangkitan variabel acak itu berkepanjangan. Berikut beberapa algoritma yang dapat dipakai guna mendapatkan variabel acak yang mempunyai pola distribusi tertentu.

2.3.9 Daftar Kejadian (Event List)

Ada dua macam mekanisme penanganan (pemrosesan) atas setiap event yang muncul dalam program simulasi. Yang pertama, penanganan secara langsung, artinya setiap muncul event semua konsekuensi logis dari kemunculan event seperti perubahan status, perubahan statistik dan lain sebagainya segera diproses. Pemrosesan berikutnya menunggu setelah event berikutnya telah muncul dan begitu muncul pemrosesan segera dilakukan. Begitu seterusnya sampai simulasi berakhir. Sedang yang kedua penanganan secara tidak langsung, yaitu dibangkitkan terlebih dahulu semua event yang diperkirakan muncul selama selang waktu simulasi berlangsung.

Kemudian event-event itu dimasukkan dalam daftar event (event list), yaitu suatu daftar yang berisi semua jenis event yang dimungkinkan terjadi dalam simulasi. Semua atribut yang dimiliki masing-masing event yang berada dalam daftar itu juga dicantumkan. Selanjutnya setelah daftar dimiliki barulah bagian eksekutif program akan memeriksa masing-masing atribut (waktu) yang dimiliki oleh masing-masing event. Waktu event apa yang tercepat, maka event itulah yang akan dieksekusi, yaitu dengan melakukan proses perubahan pada status, statistik dan lain sebagainya. Setelah selesai melakukan operasi-operasi itu, event tersebut

segera dihapus dari daftar dan dilanjutkan pemeriksaan kembali untuk mengetahui waktu event tercepat berikutnya dan seterusnya sampai event terakhir.

Table 2.3 Daftar event sisten antrian sederhana

NO.	TIPE EVENT	JADWAL KEDATANGAN	LAMA LAYANAN
1		0	
2	1	0.4	
3	1	1.6	
4	1	2.1	
5	2		2.4
6	2		0.7
7	2		0.2
8	1	3.8	
9	1	4.0	
10	2		1.6
11	1	5.6	
12	1	5.8	
13	1	7.2	
14	2		3.7
15	2		2.4
16	1	11.1	
17	2		1.5
18	2		1.0

2.4 Pola Distribusi Masukan

Hampir semua *real system* yang ada mengandung unsur keacakan. Adakalanya hanya memiliki satu unsur keacakan tetapi adakalanya memiliki banyak unsur keacakan. Karena itu dalam kajian simulasi, membicarakan persoalan keacakan menjadi bagian yang tidak bisa ditinggalkan. Aktualisasi keacakan dalam simulasi sering dinyatakan dengan fungsi distribusi probabilitas. Sehingga mengetahui macam-macam pola fungsi distribusi probabilitas adalah salah satu keharusan. Kesalahan atau paling tidak kekurangtepatan memilih fungsi distribusi probabilitas guna menggambarkan keacakan *real system* akan berakibat fatal pada hasil keluaran simulasi. Pada akhirnya kesimpulan yang diperoleh akan menjadi bias.

Sebagai awal untuk mengetahui pola fungsi distribusi probabilitas atas variabel acak adalah dengan mengumpulkan data masa lalu (historis) variabel tersebut. Dan selanjutnya berbekal dari data itu baru dicari fungsi distribusi probabilitasnya. Ada beberapa pendekatan yang bisa dilakukan guna mendapatkan fungsi kepadatan probabilitas, yaitu:

- a. Menggunakan langsung data historis variabel acak yang diperoleh dalam mengeksekusi simulasi. Dengan begitu keacakan variabel tersebut dengan sendirinya juga akan mewarnai simulasinya.
- b. Pendekatan empiris. Dengan metoda heuristik dicoba didapatkan fungsi distribusi empirisnya. Dengan fungsi empiris ini selanjutnya dipakai untuk melakukan sampling data dalam simulasi.
- c. Pendekatan teoritis, yaitu dengan teknik-teknik statistik inferensif (uji kebaikan suai) yang sudah baku akan didapatkan fungsi distribusi teoritis.

2.4.1 Pendugaan Keluarga Pola Distribusi

Langkah awal untuk mengetahui pola distribusi probabilitas dari suatu data adalah mencari tahu keluarga pola distribusinya. Hal ini dimaksudkan untuk mengkerucutkan dugaan yang akan dilakukan. Perlu diketahui bahwa banyak sekali

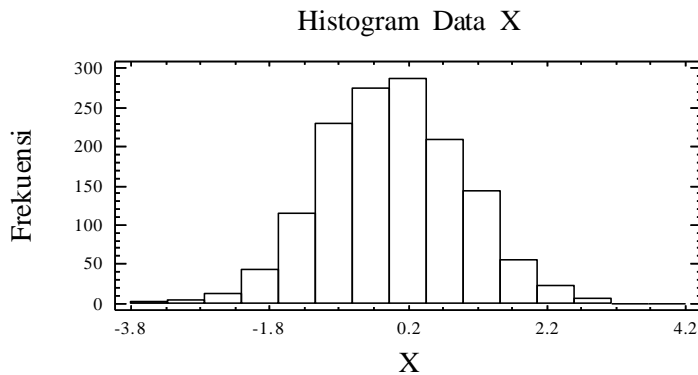
model pola distribusi yang ada baik itu yang sudah baku maupun yang belum. Ada beberapa cara yang bisa ditempuh untuk menduga keluarga distribusi ini yaitu:

1. Ringkasan Statistik

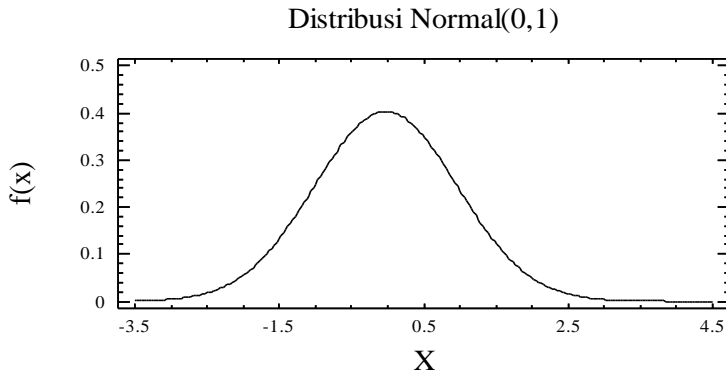
- a. Beberapa distribusi dapat dikarakteristikan paling tidak oleh ringkasan statistik datanya. Dari ringkasan ini dapat diketahui keluarga distribusinya. Nilai-nilai pemusatan merupakan besaran statistik yang cukup penting guna menduga keluarga distribusi. Mean dan median misalnya, pada distribusi kontinyu jika nilai keduanya sama, maka sudah dapat dipastikan bahwa distribusi data berbentuk simetris. Namun pada distribusi diskrit keduanya berharga hampir sama.
- b. Koefisien varian ($cv=s/x$) juga mempunyai peranan yang penting dalam menduga keluarga distribusi. Untuk nilai koefisien varian berharga $cv=1$, dapat diduga bahwa data berdistribusi eksponensial. Jika nilai koefisien varian tersebut berharga $cv<1$, maka data diduga berdistribusi Gamma atau Weibull dengan parameter $cv>1$. sedang jika nilai koefisien varian berharga $cv>1$, maka dugaan mengarah pada distribusi Gamma atau Weibull dengan parameter $cv<1$.
- c. Untuk distribusi diskrit, rasio lexis $\tau = s^2/\bar{x}$ memainkan peran yang sama seperti koefisien varian cv dalam distribusi kontinyu. Jika nilai $\tau = 1$, maka dugaan data berdistribusi Poisson sangat kuat dan jika nilai $\tau < 1$ dugaan akan mengarah pada distribusi Binomial sedang jika nilai $\tau > 1$ distribusi Binomial Negatif menjadi pilihannya.
- d. Kelandaian distribusi (skewness) $v = \frac{\sum(x-\bar{x})^2}{n} = \left(\frac{1}{s}\right)^{\frac{3}{2}}$ Adalah mengukur kelandaian distribusi. Untuk distribusi yang simetris misalnya Normal, nilai v akan berharga nol. Sedang untuk $v > 0$ distribusi akan menjulur ke kanan dan sebaliknya ke kiri. Misal nilai v sama dengan 2, berarti data berdistribusi eksponensial.

2. Histogram dan Grafik Garis

Pada data kontinyu, histogram pada dasarnya merupakan grafik dugaan atas fungsi kepadatan data. Karena itu pembuatan histogram yang tepat akan dapat menggambarkan dengan jelas atas fungsi kepadatan data. Perhatikan gambar di bawah ini. Gambar 2.2 adalah grafik histogram data yang akan diperkirakan pola distribusinya. Guna memperkirakannya dengan cara membandingkan histogram itu dengan pola-pola distribusi baku. Gambar 2.3 adalah kurva fungsi normal (pola baku/pembanding). Manakala histogram yang dimiliki memang seperti itu bentuknya, maka sudah hampir dipastikan bahwa fungsi kepadatan data itu berpola normal.



Gambar 2.6 Histogram Data



Gambar 2.7 Bentuk kurva

2.4.2 Estimasi Parameter

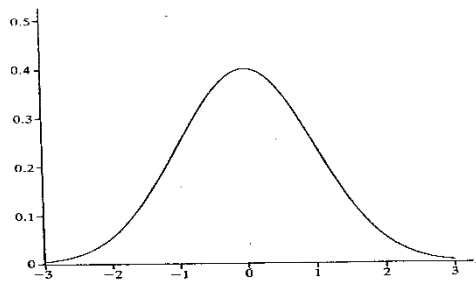
Setelah hasil perkiraan fungsi distribusi probabilitasnya diperoleh, selanjutnya estimasi akan parameter fungsi itu harus dilakukan. Sebelum bicara banyak tentang bagaimana mengestimasi nilai parameter, terlebih dahulu perlu dikenali macam parameter. Ada tiga macam parameter, yaitu:

1. Parameter Lokasi (γ).

Parameter lokasi menunjukkan tempat kedudukan (absis) nilai range distribusi (biasanya titik tengah). Parameter ini sering pula disebut dengan parameter pergeseran (shift parameter). Seiring berubahnya nilai parameter γ maka distribusi akan bergeser ke kanan atau ke kiri sepanjang sumbu x. Perhatikan fungsi kepadatan distribusi normal berikut :

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(x-\mu)^2/2\sigma^2} \text{ untuk semua bilangan riil } x$$

Klau dibuat geafiknya bentuk tampilan dari pola distribusi ini seperti ditunjukkan oleh gambar 2.9.



Gambar 2.8 : Fungsi kepadatan Normal(μ, σ^2).

2. Parameter Skala (β)

Parameter β akan menjelaskan skala/unit pengukuran dari nilai dalam interval distribusi. Perubahan nilai β akan serta merta mengubah ukuran bentuk distribusi ke bentuk mengembang atau menyusut dari bentuk dasarnya.

3. Parameter Bentuk(α).

Sedang parameter α akan berdampak pada perubahan bentuk dasar dari distribusinya. Perubahan harga α dengan sendirinya akan mengubah bentuk dasar distribusi atau akan mengubah sifat dasarnya.

2.4.3 Uji Kebaikan Suai Distribusi

Setelah bentuk fungsi distribusi didapatkan, yaitu pada tahap pendugaan di atas, tahap selanjutnya yang tidak kalah penting adalah mengetahui seberapa baik dan sesuai fungsi itu dapat mencerminkan pola populasinya. Ada beberapa cara yang bisa dipakai, yaitu :

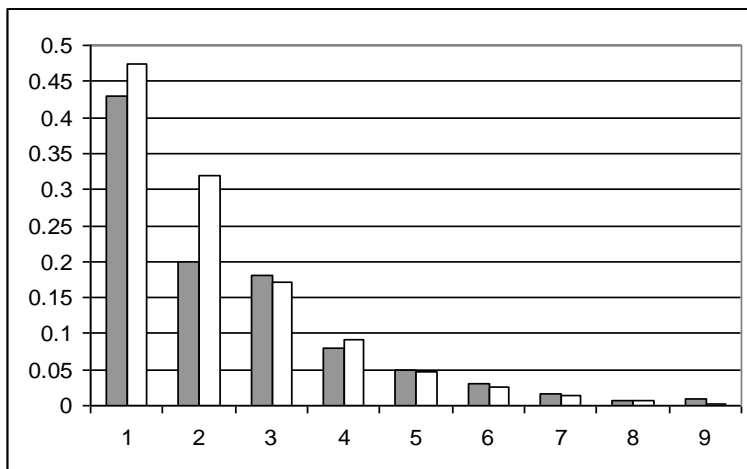
1. Cara Heuristik

Cara ini dilakukan dengan cara membandingkan histogram data dengan kurva dari fungsi distribusi $\hat{f}(x)$ yang akan diuji. Misal $[b_0, b_1]$, $[b_1, b_2]$, $[b_2, b_3]$. . . $[b_{k-1}, b_k]$ adalah k buah interval histogram dengan lebar masing-masing intervalnya Δb , dan anggap h_j adalah proporsi pengamatan x_i yang jatuh di interval ke j . Dan r_j adalah proporsi harapan dari n observasi

yang akan jatuh di interval j jika $\hat{f}(x)$ yang diuji adalah benar dan r_j dituliskan sebagai :

$$r_j = \int_{b_{j-1}}^{b_j} \hat{f}(x) dx$$

maka perbandingan frekuensi dapat dilakukan dengan menggambarkan kedua nilai itu bersama-sama pada interval histogram yang ke j dengan $j = 1, 2, \dots, k$. Kebaikan dan kesesuaiannya dapat dilihat dari seberapa sama dan sebangunnya kedua grafik yang telah digambarkan tadi (lihat gambar 2.10).



Gambar 2.9 Histogram Proporsi Pengamatan dan Harapan

2. Uji Chi-Square

Uji kebaikan suai Chi Square merupakan metoda uji yang tertua. Uji ini dapat difikirkan sebagai perbandingan formal antara histogram dengan kurva fungsi distribusi yang ingin diuji. Pada awalnya data yang dimiliki dibagi ke dalam k buah interval yang berdekatan. Kemudian data di tally ke dalam masing-masing interval tersebut sehingga didapat nilai f_j yaitu banyaknya data yang berada pada interval j .

Ingat bahwa $\sum f_j = n$. Selanjutnya jika $f(x)$ adalah memang fungsi distribusi yang diperkirakan, maka dihitung proporsi harapan dari x_i yang jatuh pada interval j dengan rumus :

$$p_j = \int_{a_{j-1}}^{a_j} \hat{f}(x) dx$$

Nilai statistik Chi-Square dapat dihitung dengan rumus $X^2 = \sum \{(f_j - np_j)^2 / np_j\}$. Jika fungsi distribusi $f(x)$ memang baik dan sesuai, maka diharapkan nilai X^2 kecil (lebih kecil dari $X^2_{k-1,1-\alpha}$), sehingga H_0 , yaitu bahwa x_i adalah data acak yang berasal dari fungsi distribusi $f(x)$ akan ditolak jika X^2 terlalu besar.

3. Uji Kolmogorov–Smirnov (K-S)

Uji K-S, berprinsip pada ingin membandingkan seberapa baik dan sesuai antara fungsi distribusi empiris data dengan fungsi distribusi yang dihipotesakan. Dibanding dengan uji Chi-Square, maka uji K-S ini mempunyai beberapa kelebihan, yaitu pertama, pada uji ini tidak perlu lagi dilakukan pengelompokan data sehingga tidak akan dijumpai kesulitan dalam upaya pengelompokan itu dan bahkan yang lebih penting adalah tidak dijumpai informasi yang hilang akibat pengelompokan ini. Kedua, uji ini akan tetap valid dengan berbagai macam ukuran sampel n . Dan yang ketiga, uji ini lebih berdaya menghadapi berbagai macam fungsi distribusi. Namun kelemahannya adalah, bahwa uji ini hanya cocok untuk fungsi distribusi yang kontinu dan telah diketahui parameter-parameternya. Bentuk fungsi empirik dari data x_1, x_2, \dots, x_n dalam uji K-S telah didapatkan bentuknya adalah :

$$f_n(x) = (\text{Banyaknya } x_i \leq x) / n$$

untuk semua bilangan real x . Maka $f_n(x)$ akan berbentuk fungsi step dengan demikian $f_n(x_{(i)}) = i/n$ untuk $i = 1, 2, 3, \dots, n$. Jika $\hat{f}(x)$ adalah fungsi distribusi yang ingin diuji, maka K-S akan mengukur seberapa dekat antara $f_n(x)$ dengan $\hat{f}(x)$

, yaitu dengan cara mencari selisihnya. Jika selisih terbesar D_n terlalu besar (lebih besar dari $d_{n,1-\alpha}$), maka hal ini menandakan kedua fungsi tersebut tidak cukup untuk dikatakan baik dan sesuai satu dengan yang lainnya. Nilai D_n didapatkan dari

$$D_n = \text{Max}\{D_n^+, D_n^-\}$$

$$\text{Dengan } D_n^+ = \text{Max}\{i/n - \hat{f}(x_{(i)})\} \quad \text{untuk } i = 1, 2, \dots, n$$

$$D_n^- = \text{Max}\{\hat{f}(x_{(i)}) - (i-1)/n\} \quad \text{untuk } i = 2, \dots, n$$

4. Uji Anderson-Darling (A-D)

Satu kelemahan yang ada dalam uji K-S adalah bahwa metoda tersebut memberikan bobot yang sama pada $[f_n(x) - \hat{f}(x)]$ untuk setiap nilai x , sedangkan banyak distribusi mempunyai ekor (tail) yang berbeda. Uji A-D dirancang untuk mendeteksi ketidaksesuaian di ekor dan mempunyai daya yang lebih tinggi dari pada uji K-S menghadapi berbagai distribusi alternatif. Statistik A-D didefinisikan dengan

$$A_n^2 = \int_{-\infty}^{\infty} [f_n(x) - \hat{f}(x)]^2 \psi(x) \hat{f}(x) dx$$

$$\text{Dimana fungsi bobot } \psi(x) = \frac{1}{\{\hat{f}(x)[1 - \hat{f}(x)]\}}$$

Maka A_n^2 adalah rata-rata berbobot dari kwadrat selisih $[f_n(x) - \hat{f}(x)]^2$ dan bobot itu terbesar untuk $\hat{f}(x)$ mendekati 1 (ekor kanan) dan $\hat{f}(x)$ mendekati 0 (ekor kiri). Jika kita ambil $Z_i = \hat{F}(x_{(i)})$ untuk $i = 1, 2, 3, \dots, n$ dapat diperoleh :

$$A_n^2 = \frac{-\sum_{i=1}^n (2i-1)(\ln Z_i + \ln(1 - Z_{n+1-i}))}{n} - n$$

Adalah bentuk statistik yang digunakan. Selagi A_n^2 adalah “jarak terbobot” bentuk dari uji ini akan menolak H_0 jika A_n^2 melebihi nilai dari $a_{n,1-\alpha}$ (tabel 5.4) dengan α adalah tingkat uji.

Table 2.4 Nilai kritis uji A-D ($a_{n,1-\alpha}$)

Kasus	Uji Statistik	1 - α			
		0.900	0.950	0.975	0.990
1	A_n^2 untuk $n \geq 5$	1.933	2.492	3.070	3.857
2	$\left(1 + \frac{4}{n} - \frac{25}{n^2}\right) A_n^2$	0.632	0.751	0.870	1.029
3	$\left(1 + \frac{0.6}{n}\right) A_n^2$	1.070	1.326	1.587	1.943
4	$\left(1 + \frac{0.2}{\sqrt{n}}\right) A_n^2$	0.637	0.757	0.877	1.038

2.4.4 Beberapa Distribusi Penting

Ada banyak pola-pola distribusi baku yang sudah ada baik pola distribusi kontinyu maupun pola distribusi diskrit. Beberapa diantaranya yang dirasa cukup penting adalah:

1. Distribusi Uniform(a,b): Distribusi ini penting dalam upaya pembangkitan bilangan acak untuk semua distribusi variabel acak.
2. Distribusi Eksponensial(β): seringkali dikaitkan dengan waktu antar kedatangan pelanggan ke suatu sistem layanan yang terjadi pada laju kedatangan yang konstan.

3. Distribusi Gamma(α, β): Distribusi ini acap kali cocok pada persoalan lama waktu layanan misal di loket layanan pelanggan, layanan perbaikan mesin dan lain sebagainya.
4. Distribusi Weibull(α, β): Distribusi ini sering terkait dengan waktu layanan seperti pada distribusi Gamma(α, β). Disamping itu juga sering terkait dengan waktu kerusakan sebuah komponen dalam masalah perawatan.
5. Distribusi Normal(μ, σ^2): Distribusi ini dapat dijumpai pada hampir semua kejadian di sekitar kita, misal pada hasil inspeksi produk cacat, penyebaran nilai ujian mahasiswa, tingkat pertumbuhan tanaman dan lain sebagainya.
6. Distribusi Lognormal(μ, σ^2): Distribusi ini banyak digunakan pada penggambaran waktu layanan sepertihalnya pada distribusi Gamma(α, β) dan distribusi Weibull(α, β).
7. Distribusi Uniform Diskrit(i, j).
8. Distribusi Binomial(n, p): Banyak digunakan untuk menyatakan banyaknya produk cacat dalam batch, banyak item yang dibutuhkan dari gudang.
9. Distribusi Poisson(λ): Distribusi ini erat kaitannya dengan banyak kejadian per satuan waktu atau per satuan luasan.
10. Distribusi Geometrik(p): Banyak digunakan dalam pengendalian kualitas, yaitu untuk menyatakan jumlah item yang diinspeksi sebelum item cacat yang pertama ditemukan. Jumlah item dalam batch dengan ukurannya yang acak.

2.5 Macam-macam Simulasi Ditinjau dari Hasil Keluarannya

2.5.1 Terminating Simulation

Terminating simulation adalah simulasi yang mempunyai batas akhir eksekusinya. Batas akhir ini ditandai dengan munculnya event alamiah E dimana saat itu sudah tidak ada lagi informasi yang berguna bagi analisa hasil. Perhatikan beberapa contoh berikut:

- a. Layanan Kasir supermarket.

Kasir supermarket dibuka setiap hari mulai jam 09.00 pagi sampai jam 17.00. Akan diukur kinerja pelayanan atas kasir supermarket tersebut setiap harinya. Sebagai titik terminal E simulasi dalam contoh ini adalah event perginya semua customer yang datang setidaknya jam 17.00.

b. Industri pesawat

Sebuah industri pesawat menerima kontrak kerja untuk membuat 100 unit pesawat terbang yang harus selesai dalam kurun waktu 18 bulan kedepan. Perusahaan akan mensimulasikan berbagai variasi konfigurasi operasi manufaktur guna mengetahui konfigurasi mana yang paling mampu memenuhi deadline penyerahan 100 pesawat. Dalam contoh ini berarti E adalah 100 unit pesawat yang lengkap.

c. Perusahaan manufaktur

Sebuah perusahaan manufaktur beroperasi selama 16 jam per hari (2 shift). Simulasi dibangun dalam rangka mengetahui seberapa banyak penumpukan *work in process* dari hari ke hari. Apakah ini termasuk terminating simulation dengan E adalah 16 jam simulasi yang telah terlewatkan? Kalau proses kegiatan manufaktur ini berkelanjutan untuk hari-hari berikutnya dengan anggapan bahwa penumpukan work in proses di akhir suatu hari kerja menjadi kondisi awal hari kerja berikutnya, maka ini tidak tergolong terminating simulation.

2.5.2 Non Terminating Simulation

Adalah simulasi yang tidak mempunyai event alamiah E yang dipakai sebagai patokan untuk mengakhiri simulasi. Pengukuran kinerja simulasi ini dilakukan melalui parameter kondisi mantabnya (steady state parameter). Setelah kondisi mantab tercapai, serta merta dapat diestimasi nilai rata-rata parameter kondisi mantabnya (steady state mean). Perhatikan beberapa contoh berikut:

d. Perusahaan manufaktur

Sebuah perusahaan sedang membangun sistem manufaktur. Guna mengetahui rata-rata *throughput* setiap jam dari sistem itu. Perusahaan telah membuat simulasinya. Misal N_i adalah jumlah part yang dapat dihasilkan di hari ke i , maka sebenarnya nilai N_i merupakan proses yang stokastik. Jika N_i mempunyai distribusi steady state, maka baru bisa diestimasi rata-ratanya (mean) $\mu = E(N)$. Sehingga simulasinya adalah non terminating. Tetapi kalau pihak perusahaan ingin mengetahui seberapa lama dari awal sistem dapat bekerja secara normal, maka simulasinya menjadi terminating dengan E adalah sistem berjalan secara normal. Dari contoh ini terlihat bahwa terminating atau non terminating sangat ditentukan oleh tujuan macam apa yang ingin dicapai dalam simulasi tersebut.

e. Perusahaan jasa

Pandang sebuah perusahaan telephone jarak jauh yang mana customer harus men-dial nomor telepon lokal untuk bisa mengakses system. Anggap laju kedatangan penelpon ke sistem berubah terhadap waktu dalam sehari dan hari ke minggu, tapi dengan pola laju kedatangan yang identik dari minggu ke minggu. Misal D_i adalah delay penelpon ke i antara saat men-dial nomer lokal sampai dengan teraksesnya sistem. Proses stokastik D_1, D_2, D_3 , tidak mempunyai distribusi yang *steady state*. Maka jika simulasi diberlakukan pada sistem ini dalam rangka mengestimasi nilai harapan delay dalam mingguan, akan terjadi non terminating simulation.

2.6 Program arena

Program Arena adalah salah satu program aplikasi sistem operasi Microsoft Window yang didesain sedemikian rupa sehingga cepat bersahabat dengan pemakai dengan segala keistimewannya. Dibandingkan dengan window software yang lain, seperti paket word processor, spreadsheet dan CAD, Arena tidak kalah familiernya, sehingga pemakai dapat dengan mudah mempelajarinya. Secara kusus Arena dibuat

untuk aplikasi simulasi. Bukan hanya itu ternyata Arena melengkapi diri dengan sebuah animasi sehingga apapun yang disimulasikan dengan arena dapat sekaligus simulasi itu diwujudkan dalam sebuah animasi yang menarik.

Untuk memperlancar dalam menjalankan paket Arena, pemakai disarankan untuk sudah menguasai operasi-operasi dasar windows. Karena dalam buku ini tidak dibahas secara khusus akan hal itu.

2.6.1 Modul Pembangun Model Arena

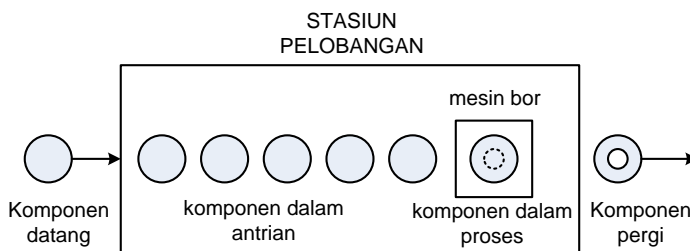
Untuk membangun sebuah model simulasi dalam Arena diperlukan blok bangunan (*Building block*) yang lebih lanjut dikenal dengan modul flowchart. Dengan modul ini model simulasi dapat dibangun dengan mudah dan cepat. Model tidak harus disusun satu perintah demi satu perintah sebagaimana dalam program pascal, namun cukup merangkai modul-modul yang telah tersedia di project bar window sehingga membentuk sebuah flowchart. Selanjutnya masing-masing modul tinggal dilengkapi data dan parameternya masing-masing. Ada banyak modul flowchart yang telah disediakan Arena dan tinggal memakainya. Untuk kali ini adalah modul flowchart dasar yang terangkum dalam panel Basic Process.

Guna mempermudah pemahaman akan bagaimana membangun model simulasi dalam Arena, diambil sebuah contoh sistem produksi sederhana, yaitu stasiun kerja pengeboran komponen mobil. Stasiun kerja tersebut bertugas melubangi/mengebor (*drilling*) salah satu komponen mobil. Mesin bor yang digunakan masih bersifat manual. Kali ini mesin yang dioperasikan sebanyak satu unit. Komponen yang akan dilubangi berasal dari proses sebelumnya.

Ketika komponen datang saat mesin bor dalam keadaan menganggur, seketika itu komponen langsung memasuki pusat layanan pemrosesan dan pengeboran segera dimulai; sebaliknya bila komponen datang saat mesin bor sedang aktif melakukan proses pengeboran, maka komponen yang baru saja tiba harus menunggu terlebih dahulu dalam antrian yang mengikuti azas First In First Out (FIFO). Dari sisi pandang yang lain, ketika ada komponen datang maka mesin bor

akan segera memproses komponen itu hingga selesai. Ketika proses selesai sementara dalam antrian terdapat komponen yang sedang antri, proses dilanjutkan untuk komponen selanjutnya sedang sebaliknya ketika sudah tidak ada lagi komponen yang menunggu untuk diproses dengan segera mesin akan tinggal diam/menganggur (idle) menunggu kedatangan komponen berikutnya.

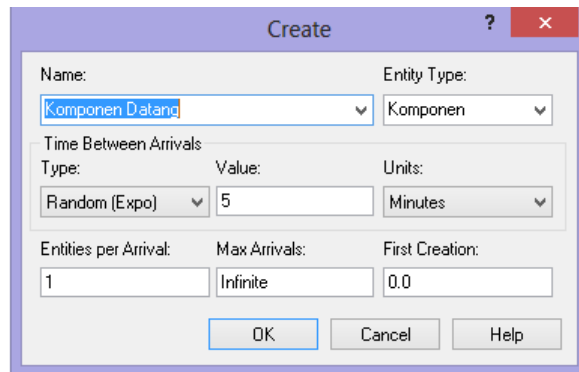
Inilah kerangka berfikir atau logika dari model sistem antrian dalam proses manufaktur yang dicontohkan kali ini.



Gambar 2.10 Proses Manufaktur Sederhana

1. Modul Create 

Modul ini digunakan untuk menciptakan (create) atau membangkitkan kedatangan entiti. Untuk memindahkan modul create ke jendela model cukup dilakukan dengan cara menyeret (dragging) modul creat dari jendela project bar ke jendela model. Setelah berada di jendela model, selanjutny diisi data dan parameternya. Isian dilakukan sesuai persoalan (real system) yang ada. Untuk mengisinya dapat dilakukan melalui kotak dialognya. Kotak dialog (gambar 2.12) dapat ditampilkan dengan melaukan *double click* pada modul create.

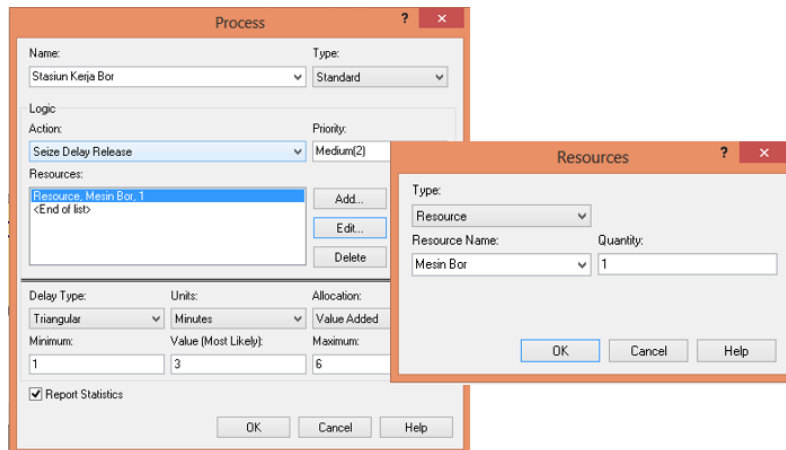


Gambar 2.11 Kotak Dialog Modul Create

- a. *Name* : tempat memberikan nama modul create misal diberi nama *Komponen Datang*.
- b. *Entity Type* : menentukan tipe entiti apakah berupa dokumen, suku cadang dan lain sebagainya misal bertipe *Komponen*.
- c. *Time Between Arrival* : untuk menenentukan pola distribusi waktu antar kedatangan entiti.
- d. *Type* : Tempat menuliskan tipe pola distribusinya waktu antar kedatangan komponen. Ada banyak pilihan. Misal berdistribusi exponential dengan parameter beta 5 menit.
- e. *Value* : menetapkan nilai parameter distribusinya, yaitu nilai beta.
- f. *Units* : satuan waktu yang digunakan, yaitu menit
- g. *Entities per Arrival* : banyak entiti setiap kali datang misal 1 unit setiap kali datang.
- h. *Max Arrival* : jumlah maksimum komponen yang akan dibangkitkan oleh modul create. Ketika jumlah itu telah tercapai, maka pembangkitan akan berhenti. Misal diberi masukkan infinite.
- i. *First Creation* : waktu kapan pertama kali entiti datang. Misal pada menit ke nol.

2. Modul Process 

Modul ini digunakan untuk melakukan pemrosesan terhadap entiti yang datang. Agar membentuk aliran flowchart yang baik, modul ini di *dragging* ke sebelah kanan modul create. Adapun kotak dialognya adalah seperti pada Gambar (2.13).



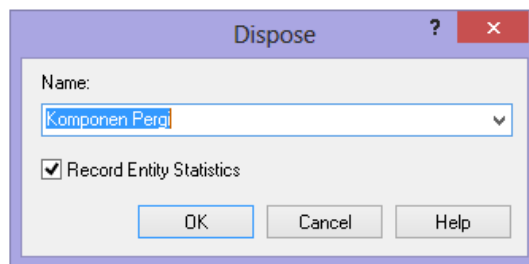
Gambar 2.12 Kotak Dialog Modul Process

- a. *Name* : tempat memberikan nama modul process misal diisi Stasiun Kerja Bor
- b. *Type* : tempat pemilihan logika proses yang ada di dalam modul. Misal dipilih standard.
- c. *Logic Action* : Jenis pengolahan yang akan terjadi dalam modul. Ada empat pilihan action, yaitu (a). *Delay* hanya menunjukkan bahwa penundaan proses akan terjadi dengan tidak ada keterbatasan sumber daya. *Seize Delay* menunjukkan bahwa sumber daya akan dialokasikan dalam modul ini dan penundaan akan terjadi, tapi pelepasan sumber daya itu akan terjadi di kemudian hari. *Seize Delay Release* menunjukkan bahwa sumber daya akan dialokasikan diikuti dengan penundaan proses dan kemudian sumber daya yang dialokasikan akan dilepas. *Delay Release* menunjukkan bahwa sumber daya sebelumnya telah dialokasikan dan bahwa entitas dengan sederhana akan menunda dan melepaskan sumber daya tertentu. Pilihan-pilihan itu berlaku ketika isian *Type* adalah Standard.
- d. *Priority* : nilai prioritas dari entitas yang menunggu di modul ini untuk sumber daya yang ditentukan. Digunakan ketika satu atau lebih entitas dari modul lain sedang menunggu sumber daya yang sama. Tidak berlaku ketika Action adalah Delay atau *Delay Release*, atau jika Type berisi *Submodel*.
- e. *Resources* : tempat menjelaskan sumber daya apa yang akan digunakan dan berapa jumlahnya. Untuk mengisinya dilakukan klik pada tombol Add. Selanjutnya diisi nama sumber dayanya dan jumlahnya. Dalam contoh ini sumber dayanya Mesin Bor dan jumlahnya satu unit.

- f. *Delay Type* : Isian ini digunakan untuk menyatakan pola distribusi lama waktu pemrosesan berlangsung. Misal berdistribusi segi tiga dengan parameternya nilai terkecil satu nilai rata-ratanya tiga dan nilai terbesarnya enam.
- g. *Units* : satuan waktu yang digunakan misalkan menit.
- h. *Allocation* : pengalokasian waktu pada biaya proses.
- i. *Minimum, value, maximum* : tempat-tempat memasukkan nilai parameter distribsinya

3. Modul Dispose

Modul ini digunakan untuk menyatakan kepergian entiti yang telah selesai mengalami pemrosesan. Adapun kotak dialognya ditunjukkan pada Gambar (2.14). Tidak banyak yang harus diisi pada kotak dialog modul dispose ini, yaitu



Gambar 2.13 Kotak Dialog Modul Dispose

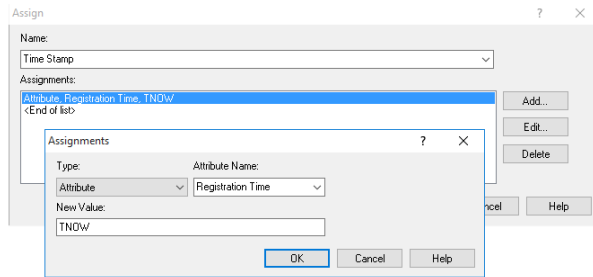
Dimana:

- a. Name : nama modul dispoe
- b. Record Entity Statistic : Kalau pilihan ini dicontreng maka Arena akan merekam statistik entiti.

4. Modul Assign

Pada diri sebuah entity sangat dimungkinkan mempunyai banyak atribut. Atribut sebuah entity identic dengan narasi yang ada di label yang menempel pada sebuah produk. Pada label itu terdapat banyak informasi berkaitan dengan produknya. Misal dalam label tertulis no batch, tanggal kadaluwrsa, beratnya dan

lain sebagainya. Salah satu cara menempelkan atribut pada sebuah entity menggunakan modul assign. Namun demikian secara umum modul ini digunakan untuk memberikan nilai baru ke variabel, atribut entity, tipe entity, gambar entity, atau variabel sistem lainnya. Pemberian nilai secara serentak dapat dilakukan dengan modul Assign tunggal.

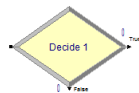


Gambar 2.14 Kotak dialog modul assign

Pada contoh isian Gambar 12 ini menampilkan bagaimana atribut, bernama Registration Time, dapat diberi nilai. Ketika entity memasuki modul, attribute Registration Time diatur ke waktu simulasi saat ini (TNOW). Atribut ini nantinya dapat digunakan dalam modul Record, dengan menggunakan statistik Interval, untuk memasukkan interval antara waktu entity melewati modul Record dan waktu yang ditentukan oleh atribut Registration Time.

Contoh lain, ketika assignment type dipilih entity picture, maka tampilan animasi entity akan berupa gambar yaitu gambar yang berada dalam file Picture.Report. Perwujudan gambar yang berada di file Picture.Report dapat dilihat dalam pustaka gambar melalui menu Edit>Entity Pictures). Perwujudan gambar itu jika diinginkan dapat diganti dengan perwujudan gambar lain yang ada dalam pustaka gambar ini pula.

5. Modul Decide



Dengan modul ini dimungkinkan proses pengambilan keputusan dalam sistem dilakukan. Hanya ada dua macam keputusan yang dapat diambil, yaitu keputusan yang diambil ketika kondisi benar. Keputusan ini akan diwujudkan

kedalam sejumlah aktivitas melalui pintu keluaran yang ada di sisi kanan, sedang yang lain adalah keputusan yang diambil ketika kondisi salah. Keputusan ini akan diwujudkan kedalam sejumlah aktivitas melalui pintu keluaran yang ada di sisi bawah.

The screenshot shows a dialog box titled 'Decide' with a question mark icon and a close button. It contains the following fields:

- Name:** A dropdown menu with the text 'Seleksi produk cacat'.
- Type:** A dropdown menu with the text '2-way by Chance'.
- Percent True (0-100):** A dropdown menu with the text '10' and a percentage symbol (%) to its right.

At the bottom of the dialog box are three buttons: 'OK', 'Cancel', and 'Help'.

Gambar 2.15 : Kotak isian modul decide untuk 2-way by change

Dalam contoh ini, modul decide digunakan untuk menyeleksi produk cacat. Misal telah diketahui 10% produk yang cacat dilakukan proses pengerjaan ulang, maka percent true berisi 10 dan aliran aktivitas selanjutnya yaitu pengerjaan ulang akan diteruskan melalui pintu keluaran di sisi kanan modul (true). Sedang produk baik sebanyak 90% akan dilakukan proses pengemasan yang aliran aktivitasnya tersambung dengan pintu keluaran yang berada di sisi bawah (false).

The screenshot shows a dialog box titled 'Decide' with a question mark icon and a close button. It contains the following fields:

- Name:** A dropdown menu with the text 'Seleksi produk cacat'.
- Type:** A dropdown menu with the text '2-way by Condition'.
- If:** A dropdown menu with the text 'Variable'.
- Named:** A dropdown menu with the text 'Berat'.
- Is:** A dropdown menu with the text '<>'.
- Value:** A text input field containing the number '125'.

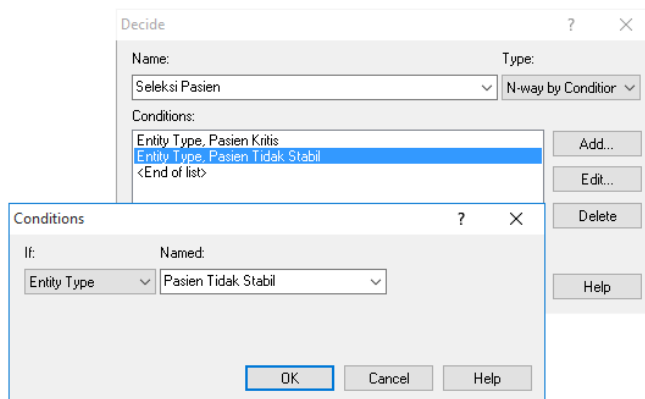
At the bottom of the dialog box are three buttons: 'OK', 'Cancel', and 'Help'.

Gambar 2.16 Isian modul decide untuk 2-way by Condition

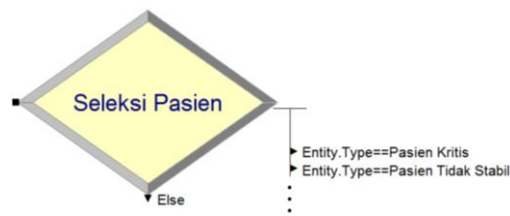
Bisa jadi penyeleksian produk cacat dilakukan berdasarkan nilai kondisi/syarat tertentu. Misal pada proses pengemasan gula pasir dalam karung.

Ketika isi karung tidak sama dengan 125 kg, maka dikatakan bahwa pengisiannya salah dan harus disesuaikan. Maka dalam kotak isian dituliskan jika (If) Berat \diamond 125. Ketika benar bahwa Berat \diamond 125, maka proses-proses selanjutnya yang akan dilakukan adalah proses-proses yang tersambung dengan pintu keluar yang ada di sisi kanan.

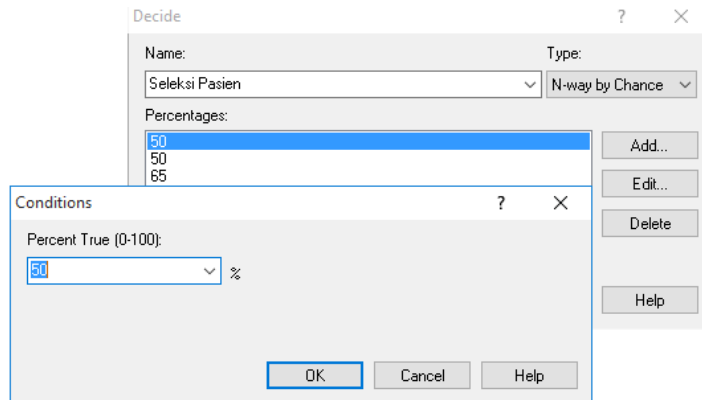
Gambar 2.18 menunjukkan isian dengan N-way by Condition. Dicontohkan bahwa ketika pasien dalam kondisi kritis pasien harus masuk ke ruang ICU kelas A. Dari modul decide pasien kritis keluar melalui pintu kanan yang pertama (lihat Gambar 16). Pasien yang kondisi tidak stabil, maka pasien harus masuk ke ruang ICU kelas B, sehingga dari modul decide keluar melalui pintu kanan yang kedua. Sedang pasien yang sudah stabil boleh di bawa ke ruang regular. Banyaknya kondisi bias ditambah. Dengan begitu akan muncul banyak pintu keluar di sisi kanan modul. Pengertian yang serupa jika menggunakan tipe N-way by Chance (lihat Gambar 2.17).



Gambar 2.17 Isian modul decide untuk N-way by Condition.

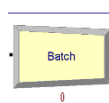


Gambar 2.18 N buah pintu keluar dari N-way by Condition



Gambar 2.19 Isian modul decide untuk N-way by Chance

6. Modul Batch



Modul ini digunakan sebagai mekanisme upaya pengelompokan entity dalam model simulasi. Batch entity dapat dikelompokkan secara permanen atau sementara. Jika memilih Batch yang sementara maka kelak batch data diurai kembali dengan menggunakan modul Separate. Sebagai contoh misalnya pada proses pembuatan rokok. Setiap 10 batang rokok yang dihasilkan dikemas secara permanen dalam satu pak. Contoh lain, dalam sebuah lintasan produksi pembuatan produk yang dimensinya kecil, maka untuk memindahkan produk-produk setengah jadi ke proses berikutnya seringkali diangkat dengan menggunakan keranjang dalam jumlah tertentu. Setelah sampai di tempat tujuan isi keranjang kemudian dibongkar. Untuk contoh yang kedua menggunakan Batch yang sementara.

Batch dibentuk dengan cara mengumpulkan entity dalam jumlah tertentu dalam antrian setelah jumlah tertentu telah tercapai maka muncul entity baru yang mewakili batch. Semula entity batangan rokok yang terkumpul dalam antrian. Setelah terkumpul sepuluh batang rokok, maka penyebutannya bukan lagi batangan rokok tetapi bungkus rokok yang berisi sepuluh batangan rokok.

Gambar 2.20 Kotak isian modul Batch

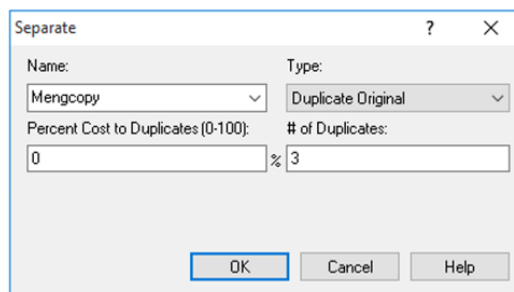
Perhatikan kotak isian modul Batch pada Gambar (2.20). Entity Batang rokok tiba di modul Pembungkusan dan ditempatkan dalam antrian bernama Pembungkusan.Queue. Ketika sepuluh Batang rokok dengan tipe entity yang sama terakumulasi, satu entitty perwakilan tetap/permanent yaitu Bungkusan rokok meninggalkan modul. Tetapi jika tidak ada jenis entity perwakilan yang ditentukan, maka entity yang dikelompokkan akan mempertahankan jenis entitty dari entity terakhir yang memasuki antrian yaitu Batang rokok.

7. Modul Separate

Modul ini dapat digunakan disamping untuk menyalin sebuah entity masuk menjadi multi entity juga untuk memisahkan/mengurai entity yang sebelumnya dalam bentuk batch dikelompokkan dalam batch. Aturan untuk mengalokasikan biaya dan waktu untuk menduplikat ditentukan. Aturan untuk penugasan atribut ke anggota entitas juga ditentukan.

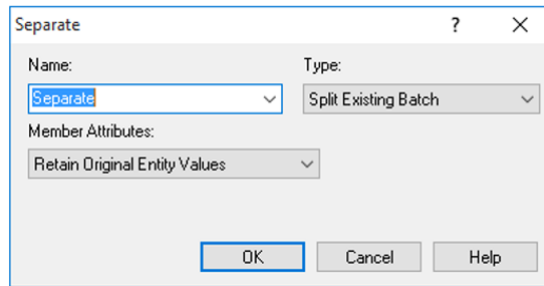
Saat memisahkan/mengurai batch, entity perwakilan sementara yang dibentuk dibuang dan entity asli yang membentuk kelompok tersebut dipulihkan/dimunculkan kembali. Entitas melanjutkan perjalanan secara berurutan dari modul dengan urutan yang sama dengan saat entity ditambahkan ke batch. Saat menduplikasi entitas, jumlah salinan yang ditentukan dibuat dan dikirim dari modul. Entitas masuk asli juga meninggalkan modul.

Dalam contoh pada Gambar 2.22, modul Mengcopy akan menyalin entity masuk yang asli ke dalam tiga entity duplikat. Sebanyak empat entity akan meninggalkan modul. Entity asli akan menyimpan semua informasi biaya dan waktu. Tiga duplikat yang lain akan mulai tanpa akumulasi biaya atau waktu, karena biaya untuk duplikat ditentukan sebagai 0.



Gambar 2.21 Kotak isian modul Separate-Duplicate

Contoh pada Gambar 2.21, modul Separate akan mengambil entity perwakilan batched yang masuk dan membaginya/mengurai-nya menjadi komponen aslinya. Entity perwakilan kemudian dibuang. Entitas asli akan mempertahankan nilai atribut tertentu dari sebelum mereka dikumpulkan dalam batch. Ini termasuk atribut mempertahankan Entity.Type, Entity.Picture, Entity.Station, Entity.Sequence, Entity.JobStep, dan Entity.HoldCostRate, dan semua atribut yang ditetapkan oleh pengguna.

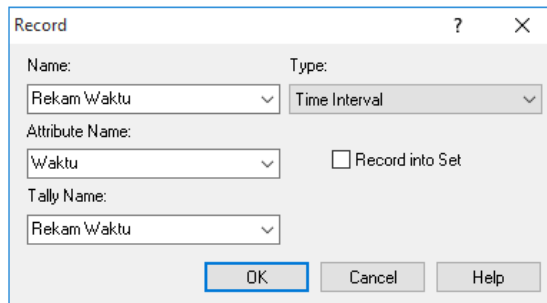


Gambar 2.22 Kotak isian modul Separate-Split

8. Modul Record




Modul ini digunakan untuk mengumpulkan statistik dalam model simulasi. Berbagai jenis statistik hasil observasi tersedia, termasuk waktu antara keluar melalui modul, statistik entity (waktu, biaya, dan yang lainnya), Observasi umum, dan statistik interval (dari beberapa waktu sampai pada waktu simulasi terkini). Jenis hitungan statistik tersedia juga. Tally dan Counter set juga bisa ditentukan.



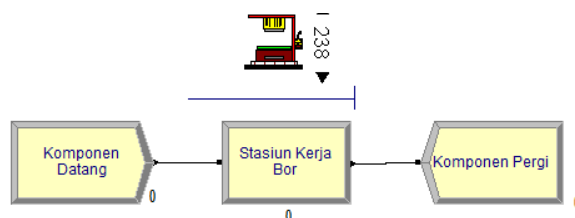
Gambar 2.23 Kotak isian modul Record

Dalam contoh pada Gambar 21, pada modul Record yang bernama Rekam Waktu setiap kali sebuah entitas tiba di modul Record, selisih antara waktu simulasi saat ini (TNOW) dan nilai atribut entity yang bernama Waktu dicatat dalam penghitungan bernama Rekam Waktu.

2.6.2 Aliran Komponen

Untuk melengkapi penggambaran sebuah aliran proses, maka modul Create, Process dan Release dihubungkan dari kiri ke kanan dengan garis penyambung. Dengan garis ini menunjukkan urutan aliran komponen dari modul flowchart ke modul flowchart yang lain. Untuk membuat penyambungan dapat dilakukan dengan meng klik menu connect  atau melalui menu *Object>Connect* selanjutnya klik pada titik keluar dari modul (ada tanda kotak hijau) dan ikuti klik pada titik masuk pada modul yang lain (ditandai kotak coklat). Ada berapa opsi pilihan penyambungan bisa dilakukan :

- a. Jika memilih *Object>Auto Connect* berarti menyambung secara otomatis titik keluaran modul yang ditunjuk ke titik masukan modul yang lain yang ditunjuk namun dengan garis sambung yang tidak tertata rapi.
- b. Jika memilih *Object>Smart Connect* berarti menyambung secara otomatis seperti pada *Object>Auto Connect* namun dengan garis sambung yang tertata rapi, yaitu tertata secara horisonal dan vertical.
- c. Jika memilih *Object>Animate Connector* Arena akan menunjukkan icon entiti bergerak menyusuri garis penghubung selama simulasi. Hal ini hanya sekedar menunjukkan bahwa perpindahan entiti telah berlangsung.

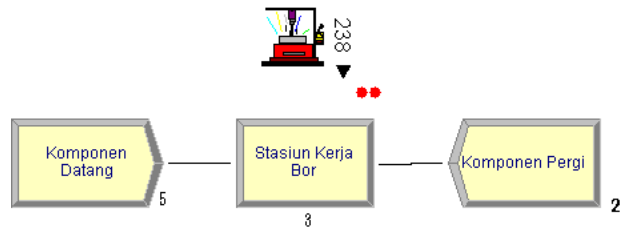


Gambar 2.24 Model simulasi stasiun kerja pengeboran

2.6.3 Perancangan Percobaan

Perancangan percobaan dapat dilakukan melalui pengaturan run simulasi. Guna mengatur run simulasi dapat dilakukan melalui menu *Run>Setup*, dengan menu ini akan menampilkan kotak dialog dengan tujuh halaman pengaturan yang meliputi:

- a. Parameter pengulangan (*Replication Parameters*).
Parameter pengulangan meliputi penetapan akan banyaknya replikasi, panjang replikasi, periode warm-up dan satuan waktu yang digunakan.
- b. Parameter proyek (*Project Parameter*).
Pada halaman parameter proyek mencakup penetapan judul dan deskripsi proyek serta data statistik yang diinginkan.
- c. Kecepatan simulasi (*Run Speed*).
Pengaturan tentang seberapa cepat simulasi berlangsung, dan pengaturan pengupdatean aktu simulasi dapat dilakukan pada halaman ini
- d. Pengendalian simulasi (*Run Control*).
Barangkali akan menyimulasikan program simulasi dari luar, atau diinginkan simulasinya tampil secara penuh, dan juga simulasi dihentikan sementara ketika peringatan-peringatan serta ketika setiap kali replikasi berikutnya dilakukan. Dan beberapa pengendalian lain dapat dilakukan di sini.
- e. Laporan (*Reports*).
Laporan dapat dibangkitkan di akhir run simulasi dengan mengklik komponen laporan di *Report Panel*. Sebagai tambahan, Arena dapat dikonfigurasi ke tampilan yang otomatis atas laporan yang spesifik ketika simulasi dirun. Ada tiga macam moda tampilan laporan, yaitu pertama, laporan selalu (*always*) ditampilkan setiap run selesai, ke dua, laporan tidak pernah ditampilkan setelah run



Gambar 2.25 Tampilan simulasi setelah beberapa saat dieksekusi

Resource

Usage

Instantaneous Utilization				
	Average	Half Width	Minimum Value	Maximum Value
Mesin Bor	0.9171	(Insufficient)	0.00	1.0000
Number Busy				
	Average	Half Width	Minimum Value	Maximum Value
Mesin Bor	0.9171	(Insufficient)	0.00	1.0000
Number Scheduled				
	Average	Half Width	Minimum Value	Maximum Value
Mesin Bor	1.0000	(Insufficient)	1.0000	1.0000
Scheduled Utilization				
	Value			
Mesin Bor	0.9171			
Total Number Seized				
	Value			
Mesin Bor	6.0000			

Gambar 2.26 Tampilan hasil simulasi untuk resource

10:59:23AM

Category Overview

February 16, 2017

Proses Manufaktur Sederhana

Replications: 1 Time Units: Minutes

Entity

Time

VA Time	Average	Half Width	Minimum Value	Maximum Value
Komponen	3.4056	(Insufficient)	1.7642	4.5167
NVA Time	Average	Half Width	Minimum Value	Maximum Value
Komponen	0.00	(Insufficient)	0.00	0.00
Wait Time	Average	Half Width	Minimum Value	Maximum Value
Komponen	3.0340	(Insufficient)	0.00	8.1598
Transfer Time	Average	Half Width	Minimum Value	Maximum Value
Komponen	0.00	(Insufficient)	0.00	0.00

Gambar 2.27 Tampilan hasil simulasi untuk entiti

2.7 Penelitian terdahulu

Table 2.5 Tabel Penelitian terdahulu

No.	Peneliti	Tahun	Judul	Metode	Hasil
1	Salammia L.A. dan Dedy Ariyanto	2010	Simulasi Keseimbangan Lintasan Proses Dalam Upaya Mengoptimalkan Waktu Proses Produksi Eternit	Line Balancing dan Simulasi	Hasil analisis yang dilakukan menunjukkan adanya peningkatan produksi, dari produk 655 yang terdiri dari 631 produk baik dan 24 cacat dan mengalami antrian sebanyak 5 lembar pada pencetakan dari model awal

No.	Peneliti	Tahun	Judul	Metode	Hasil
					menjadi produk yang dihasilkan 660 lembar terdiri dari 642 produk baik dan 18 produk tanpa ada antrian pada stasiun kerja dengan menambah 1 operator pada pencetakan dan 1 mesin potong
2	Vickri Fiesta Daelima, Evi Febianti, dan Muhammad Adha Ilhami	2013	Analisis Keseimbangan Lintasan untuk Meningkatkan Kapasitas Produksi dengan Pendekatan Line Balancing dan Simulasi	<i>Line Balancing</i> dan Simulasi	Hasil simulasi didapatkan bahwa kondisi usulan lebih baik daripada kondisi <i>eksisting</i> , hal tersebut dapat dilihat dari hasil output produknya dimana untuk <i>existing</i> hanya 1732 unit/ <i>shift</i> , sedangkan untuk model usulan CT maksimum 2043 unit/ <i>shift</i> , <i>takt time</i> permintaan 2043 unit/ <i>shift</i> dan <i>takt</i>

No.	Peneliti	Tahun	Judul	Metode	Hasil
					<i>time</i> produksi 1963 unit/ <i>shift</i> dengan menambah mesin sebanyak 1 mesin pada mesin <i>water</i> dan mesin <i>heater</i>
3	Reza Primadhana, Parwadi Moengin, dan Sucipto Adisuwiryo	2013	Evaluasi dan Usulan Perbaikan Keseimbangan Lintasan Produksi untuk Mencapai Target Produksi dengan Pendekatan Simulasi pada <i>Workshop</i> 3 di PT. Faco Global Engineering	<i>Line Balancing</i> dan Simulasi	Pemindahan 2 operator <i>painting</i> ke stasiun kerja <i>cutting wheel</i> dan 1 operator <i>fullfitting</i> ke stasiun kerja <i>drilling</i> dengan output produk 2 unit per minggu dan utilitas rata-ratanya menjadi 31,51%.

(Halaman ini sengaja dikosongkan)