BAB II
TINJAUAN PUSTAKA

Dalam penelitian yang dilaksanakan, diperlukan dasar-dasar argumen ilmiah yang berhubungan dengan konsep-konsep yang dipermasalahkan dalam penelitian dan akan dipakai dalam analisis. Dalam bab ini akan dijelaskan beberapa dasar-dasar argumentasi atau teori yang digunakan dalam penelitian.

2.1 Penelitian Terdahulu

Penelitian terdahulu memaparkan beberapa konsep relevan yang berhubungan dengan penelitian ini. Rangkuman penelitian terdahulu dan perbandingan dengan penelitian saat ini terdapat pada Tabel 2.1

1. Indrayana (2010) melakukan penelitian untuk mengoptimalkan waktu tunggu antrian kendaraan di Perempatan Pingit, Yogyakarta. Peneliti menggunakan *tools* simulasi dengan bantuan *software* ARENA. Hasil yang didapatkan adalah penurunan waktu tunggu antrian dari model awal dengan model usulan dengan presentase sebesar 41,71%-58,19%.
2. Vandahardhika dan Yopi (2011) melakukan penelitian untuk mengoptimalkan antrian konsumen pada loket dan utilitas petugas loket di PT. Pos Indonesia (Persero), Mojokerto. Peneliti menggunakan *tools* simulasi dengan bantuan *software* ARENA. Hasil yang didapatkan adalah dengan model awal yaitu 2 loket dan jumlah antrian pada loket 1 dan 2 sebesar 524 dengan utilitas petugas loket sebesar 99,7%. Sedangkan pada model usulan, yaitu 4 loket dan jumlah antrian pada 4 loket tersebut sebanyak 10 konsumen dengan utilitas petugas loket sebesar 72,5%.
3. Wardhani dan Arie Restu (2012) melakukan penelitian untuk mengoptimalkan biaya antrian pelanggan di Indomaret Borobudur, Kota Malang. Peneliti menggunakan *tools* simulasi dengan bantuan *software* ARENA. Hasil yang didapatkan adalah waktu tunggu kondisi eksisting sebesar 0.24902 menit, sedangkan waktu tunggu kondisi usulan sebesar 0.24082 menit yang berarti kondisi usulan memiliki waktu 0.0082 menit lebih cepat bila dibandingkan dengan kondisi eksisting. Sementara itu total biaya antrian kondisi usulan sebesar Rp. 60.063,166 per hari, hampir dua kali lipat total biaya antrian kondisi eksisting yaitu sebesar Rp 30.063,166 per hari.

Tabel 2.1 Penelitian Terdahulu

Peneliti	Objek Penelitian	Tujuan Penelitian	Metode Penelitian	Hasil Penelitian
Indrayana (2010)	Perempatan Pingit Yogyakarta	Mengoptimalkan waktu tunggu antrian kendaraan	Simulasi, ARENA	Setelah dilakukan eksperimen terhadap model simulasi perempatan Pingit, maka diperoleh strategi pengaturan durasi lalu lintas. Perbandingan waktu tunggu antrian maksimum antara model awal dengan model usulan menunjukkan penurunan dengan presentase 41,71% -58,19%.
Vandahardhika, Yopi (2011)	PT. Pos Indonesia (Persero) Mojokerto	Mengoptimalkan antrian pada loket dan utilitas petugas loket	Simulasi, ARENA	Dari hasil penelitian dan pengolahan data dapat ditarik kesimpulan bahwa antrian di PT. POS (Persero) Mojokerto adalah sebagai berikut: Dengan 2 (dua) loket masih terdapat antrian pada loket 1 dan 2 sebesar : 524 konsumen dengan rata – rata tingkat utilitas sebesar 99,7%, maka pelayanan terhadap pelanggan belum optimal dan terjadi antrian yang panjang dalam sistem pelayanannya sedangkan usulan 4 (empat) loket diperoleh antrian sebesar 10 konsumen, dengan rata-rata tingkat utilitas sebesar 72,5% demikian pelayanan terhadap pelanggan bisa lebih optimal dan tingkat utilitas yang dialami oleh petugas loket sudah mendekati standar dari PT. POS (Persero) Mojokerto yaitu 80%.
Wardhani, Arie Restu (2012)	Indomaret Borobudur Kota Malang	Mengoptimalkan biaya antrian pelanggan	Simulasi, ARENA	Berdasarkan hasil analisa, waktu tunggu kondisi eksisting sebesar 0.24902 menit, sedangkan waktu tunggu kondisi usulan sebesar 0.24082 menit yang berarti kondisi usulan memiliki waktu 0.0082 menit lebih cepat bila dibandingkan dengan kondisi eksisting dalam hal waktu menunggu (*waiting time*). Sementara itu total biaya antrian kondisi usulan sebesar Rp. 60.063,166 per hari, hampir dua kali lipat total biaya antrian kondisi eksisting yaitu sebesar Rp 30.063,166 per hari. Oleh sebab itu, perusahaan sebaiknya mempertahankan kondisi eksisting, karena kondisi usulan lebih mahal, sementara selisih waktunya sangat sedikit.
Eliana (2016)	Persimpangan Jalan MT. Haryono Jalan Gajayana, Malang	Mengurangi durasi antrian dengan skenario durasi lampu lalu lintas	Simulasi, ARENA	-

2.2 Lalu Lintas Persimpangan

Persimpangan merupakan salah satu bagian dari sistem lalu lintas yang menjadi faktor penting dalam penentuan kapasitas jalan. Persimpangan adalah lokasi atau daerah dimana dua atau lebih jalan, bergabung, berpotongan, atau bersilang (Sony S., 1997). Secara umum, persimpangan dibagi menjadi 4 jenis, yaitu: persimpangan tidak teratur (*uncontrolled)*, persimpangan diatur dengan prioritas (*give way, stop*), bundaran

(*roundabot*), persimpangan diatur dengan alat pemberi sinyal lalu lintas/pemisahan bertingkat (*grade-separated*) (O'Flaherty, 1997).

Adanya persimpangan tidak lepas dari sistem pengontrolan lalu lintas karena apabila tidak ada fungsi kontrol dalam lalu lintas, maka sistem lalu lintas tidak akan berjalan dengan baik. Sistem pengontrolan lalu lintas merupakan pengaturan lalu lintas yang berupa perintah atau larangan berupa lampu lalu lintas, rambu lalu lintas, atau marka jalan (Munawar A., 2004). Sistem pengontrolan lalu lintas pada persimpangan jalan meliputi:

1. Optimalisasi lampu lalu lintas

   Dengan pengaturan *cycle time* antara lampu merah, kuning, dan hijau dari lampu lalu lintas serta jumlah V fase
2. Pemasangan atau pemindahan lampu lalu lintas

   Dengan memasang lampu lalu lintas di tempat-tempat dengan arus lalu lintas yang tinggi
3. Prioritas kepada bus kota pada persimpangan dengan lampu lalu lintas

   Dengan antena pemancar pada bus kota sehingga jika terdapat bus kota mendekati lampu lalu lintas, lampu akan selalu hijau
4. Koordinasi lampu lalu lintas

   Dengan koordinasi antara lampu-lampu lalu lintas sehingga sebagian kendaraan akan melewati beberapa lampu lalu lintas tanpa berhenti

2.3 Teori Antrian

Teori antrian merupakan studi matematika dari antrian atau kejadian garis tunggu (*waiting lines*), yaitu suatu garis tunggu dari pelanggan yang memerlukan layanan dari sistem yang ada (Christoper, 2000). Komponen dasar antrian adalah:

1. Kedatangan

   Setiap masalah antrian melibatkan kedatangan, misalnya orang, mobil, atau panggilan telepon untuk dilayani. Unsur ini sering disebut proses *input*. Proses *input* meliputi sumber kedatangan atau biasa dinamakan *calling population*, dan cara terjadinya kedatangan yang umumnya merupakan proses *random*. Terdapat 3 perilaku antrian, yaitu:
   a. *Reneging* (pembatalan) adalah meninggalkan antrian sebelum dilayani.
   b. *Balking* adalah orang yang langsung pergi ketika melihat panjangnya antrian, menolak untuk memasuki antrian.
   c. *Jockeying* adalah orang yang berpindah-pindah dari satu antrian ke antrian lain

karena ingin dilayani lebih cepat.

2. Pelayanan

Pelayan atau mekanisme pelayanan dapat terdiri dari satu atau lebih pelayan, atau satu atau lebih fasilitas pelayanan. Contohnya pada sebuah *check out counter* dari suatu supermarket terkadang hanya ada seorang pelayan, tetapi bisa juga diisi seorang kasir dengan pembantunya untuk memasukkan barang-barang ke kantong plastik. Sebuah bank dapat mempekerjakan seorang atau banyak *teller*. Disamping itu, perlu diketahui cara pelayanan diselesaikan, yang kadang-kadang merupakan proses *random*.

Ada 3 aspek yang harus diperhatikan dalam mekanisme pelayanan, yaitu:

a. Tersedianya Pelayanan

Mekanisme pelayanan tidak selalu tersedia untuk setiap saat. Misalnya dalam pertunjukan bioskop, loket penjualan karcis masuk hanya dibuka pada waktu tertentu antara satu pertunjukan dengan pertunjukan berikutnya. Sehingga pada saat loket ditutup, mekanisme pelayanan terhenti dan petugas pelayanan istirahat

b. Kapasitas Pelayanan

Kapasitas dari mekanisme pelayanan diukur berdasarkan jumlah pelanggan yang dapat dilayani secara bersama-sama. Kapasitas pelayanan tidak selalu sama untuk setiap saat, ada yang tetap, tetapi ada juga yang berubah-ubah. Karena itu, fasilitas pelayanan dapat memiliki satu atau lebih saluran. Fasilitas yang mempunyai satu saluran disebut saluran tunggal atau sistem pelayanan tunggal dan fasilitas yang mempunyai lebih dari satu saluran disebut saluran ganda atau sistem pelayanan ganda.

c. Lamanya pelayanan

Lamanya pelayanan adalah waktu yang dibutuhkan untuk melayani seorang pelanggan. Lama pelayanan harus dinyatakan secara pasti. Oleh karena itu, waktu pelayanan boleh tetap dari waktu ke waktu untuk semua pelanggan atau boleh juga berupa variabel acak. Umumnya untuk keperluan analisis, waktu pelayanan dianggap sebagai variabel acak yang terpencar secara bebas dan sama serta tidak tergantung pada waktu kedatangan.

3. Komponen Antrian

Munculnya antrian tergantung dari sifat kedatangan dan proses pelayanan. Penentu lain yang penting dalam antrian adalah disiplin antrian. Disiplin antrian adalah aturan keputusan yang menjelaskan cara melayani pengantri, misalnya datang

awal dilayani dulu yang lebih dikenal dengan singkatan FCFS, datang terakhir dilayani dulu LCFS, berdasar prioritas, berdasar abjad, berdasar janji, dan lain-lain. Jika tak ada antrian berarti terdapat pelayan yang menganggur atau kelebihan fasilitas pelayanan.

2.4 Sistem, Model, Dan Simulasi

Dalam penelitian ini sistem, model, dan simulasi memiliki keterkaitan karena sebuah sistem nyata (lalu lintas) akan dimodelkan menggunakan *software* pemodelan yang kemudian dilakukan simulasi untuk mengetahui dampak yang terjadi pada model ketika parameter dalam model diubah. Pengertian sistem, model, dan simulasi akan diuraikan setelahnya.

2.4.1 Sistem

Sistem adalah sekumpulan objek yang digabungkan bersama dalam suatu interaksi dan saling interdependensi untuk memenuhi suatu tujuan (Banks, et al., 2004). Dalam memodelkan sistem, penentuan batasan antara sistem dan lingkungannya perlu dilakukan dan disesuaikan dengan tujuan pembuatan model tersebut.

Sistem dapat dikategorikan sebagai sistem diskrit ataupun sistem kontinu. Pada kenyataannya, hanya sedikit sistem yang dapat dikategorikan sebagai sistem yang sepenuhnya diskrit atau sepenuhnya kontinu, biasanya sebuah sistem memiliki kecenderungan diskrit atau kecenderungan kontinu sehingga inilah yang menjadi dasar dalam menentukan sebuah sistem bersifat diskrit atau kontinu (Law dan Kelton, 2000). Sebuah sistem diskrit adalah ketika keadaan (*state*) dalam sistem berubah dalam waktu yang diskrit, sementara sistem kontinu adalah bila keadaaan atau *state* berubah secara kontinu seiring berjalannya waktu (Banks, et al., 2004).

2.4.2 Model suatu sistem

Sebuah model didefinisikan sebagai suatu representasi dari sebuah sistem dengan tujuan untuk mempelajari sistem tersebut (Banks, et al., 2004). Sehingga dapat dikatakan bahwa model adalah pola (contoh, acuan, ragam) dari sesuatu yang akan dibuat atau dihasilkan. Model dapat diklasifikasikan dalam empat kategori. Berikut penjelasan mengenai model menurut klasifikasi tersebut (Banks, et al., 2004).

1. Model matematis atau model fisik

   Model matematis menggunakan notasi dan perhitungan matematis untuk merepresentasikan sistem. Sedangkan model fisik menggunakan benda nyata berbentuk fisik untuk merepresentasikan sistem.

2. Model statis atau model dinamis

   Model statis merepresentasikan sebuah sistem pada satu titik waktu. Contohnya adalah simulasi Monte Carlo. Sedangkan model dinamis merepresentasikan sistem seiring berjalannya waktu. Contohnya adalah simulasi bank dari jam 09:00 AM sampai dengan 04:00 PM.

3. Model deterministik atau model stokastik

   Model deterministik tidak memiliki pembangkitan bilangan acak, model deterministik memiliki kedatangan atau *input* yang telah diketahui yang akan menghasilkan *output* yang menyesuaikan. Contohnya adalah kedatangan pasien pada dokter gigi sesuai dengan jadwal *appointment*. Sedangkan model stokastik memiliki satu atau lebih variabel *random* sebagai *input* sistem. *Random input* akan menghasilkan *random output*. Contohnya adalah simulasi bank dimana akan menggunakan waktu antar-kedatangan *random*.

4. Model diskrit atau kontinu

   Model diskrit adalah model dari sistem yang bersifat diskrit yaitu model yang keadaannya berubah secara diskrit seiring waktu. Sedangkan model kontinu adalah model dari sistem yang bersifat kontinu yaitu model yang keadaannya berubah secara kontinu seiring waktu.

2.4.3 Simulasi

Simulasi adalah suatu metodologi untuk melaksanakan percobaan dengan menggunakan menggunakan model dari suatu sistem nyata (Siagian, 1987). Beberapa kegunaan dari simulasi adalah memungkinkan pembelajaran, eksperimental, dan interaksi internal dari suatu sistem yang kompleks, mengamati perubahan informasi, dan memberikan saran perbaikan terhadap sistem nyata yang sedang diamati (Banks, et al., 2004). Dari pandangan perspektif simulasi, sebuah sistem dapat dikatakan terdiri dari entitas, aktivitas, *resource*, dan *control* (Harrel, et al., 2004). Berikut adalah penjelasan elemen-elemen pada simulasi:

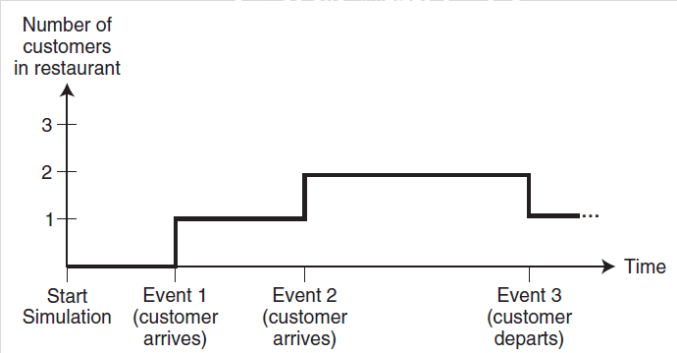
1. Entitas: merupakan segala sesuatu yang diproses dalam sistem seperti produk, konsumen, dan dokumen.

2. Aktivitas: merupakan tugas yang dilakukan di dalam sistem baik secara langsung atau tidak langsung terlibat dalam pengolahan entitas.
3. *Resources*: merupakan alat atau operator untuk menjalankan aktivitas.
4. Kontrol: merupakan segala sesuatu yang menentukan bagaimana, kapan, dan dimana aktivitas dijalankan.

2.5 *Discrete-Event Simulation*

Discrete-Event Simulation merupakan salah satu perubahan status yang terjadi pada titik-titik diskrit dalam waktu yang dipicu oleh peristiwa (Harrel, et al., 2004). Tipe kejadian dalam simulasi meliputi:

1. Kedatangan entitas pada *workstation*
2. Kegagalan dari sebuah *resource*
3. Penyelesaian suatu aktivitas
4. Berakhirnya sebuah *shift*



Gambar 2.1 *Discrete-Events* menyebabkan perubahan status
Sumber: Harrel, et al., (2004:72)

2.6 Pengumpulan Data *Input* Simulasi

Input dari model simulasi adalah distribusi tertentu dari parameter yang ingin dimodelkan. Angka acak akan dibangkitkan oleh perangkat model simulasi sesuai dengan distribusi yang telah dimasukan. Untuk itu perlu proses pengumpulan data yang baik untuk mendapatkan distribusi yang dapat merepresentasikan sistem nyata. Konsep *Garbage-in-garbage-out* (GIGO) menyatakan bahwa kendatipun struktur dari model sudah teruji validitas, namun bila input data tidak dikumpulkan atau dianalisis secara tepat serta tidak merepresentasikan sistem, *output* simulasi akan menjadi rancu sehingga dapat merusak proses simulasi dan penentuan keputusan (Banks, et al., 2004). Langkah-langkah dalam mengumpulkan data simulasi adalah sebagai berikut.

1. Mengumpulkan data dari sistem nyata yang akan dimodelkan.

   Proses ini memerlukan waktu dan tenaga yang banyak. Untuk memastikan bahwa data yang dikumpulkan cukup untuk pengolahan, dilakukan uji kecukupan data (N').

$$N' = \left(\frac{k/s \sqrt{N \sum X_i^2 - (X_i)^2}}{\sum X_i} \right)^2 \qquad (2-1)$$

   dengan:

   N' = jumlah pengamatan yang seharusnya dilakukan
   N = jumlah pengamatan yang telah dilakukan
   x_i = data ke-i (i = 1, 2, 3, …, dst.)
   k = indeks tingkat kepercayaan ($k = Z_{\alpha/2} = Z_{(1-0,95)/2} = 1,96$)
   s = indeks tingkat ketelitian (0,05)

   Jumlah pengamatan yang seharusnya dilakukan didapatkan dari perhitungan indeks tingkat kepercayaan yang ditentukan peneliti dibagi dengan indeks tingkat ketelitian, kemudian dikalikan dengan akar pangkat dua dari perkalian jumlah pengamatan yang telah dilakukan dengan pengurangan jumlah data ke i yang dikuadratkan masing-masing dan jumlah data ke i yang dikuadratkan secara keseluruhan. Kemudian dibagi dengan jumlah dari seluruh data sampai data ke i.

   Setelah dilakukan uji kecukupan, dilakukan uji *independent* untuk memastikan bahwa antar data-data tidak memiliki ketergantungan. Uji *independent* dilakukan dengan *runs test*, melihat *scatter plot* atau grafik autokorelasi (Harrel, Ghosh & Bowden, 2004). Pada penelitian ini uji *independent* akan dilakukan dengan *runs test* atau melihat grafik autokorelasi. Bila data *independent*, maka dapat digunakan langsung sebagai *input* simulasi, namun bila data *dependent*, pemodel harus meninjau ulang dampak dari ketergantungan tersebut sebelum dimasukkan ke dalam model simulasi (Banks et al., 2004).

2. Mengidentifikasi distribusi probabilitas untuk merepresentasikan *input* proses.

   Setelah data telah layak untuk digunakan, maka tahap selanjutnya adalah mengubah data tersebut sehingga dapat menjadi *input* model simulasi (Harrel, et al., 2004). Distribusi probabilitas yang dapat dipilih seperti normal, *uniform*, *triangular*, *exponential*, *poisson*, dan sebagainya. Tahap ini dilakukan dengan *goodness of fit test*. *Goodness of fit test* yang dilakukan adalah Chi-Square karena lebih sesuai untuk data-data yang memiliki bentuk histogram *modal*, *bimodal*, *plateau*, dan *triangular*.

3. Mendapatkan nilai parameter untuk merepresentasikan data dari distribusi.

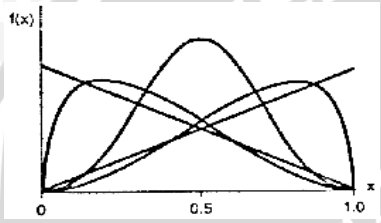
   Parameter distribusi ditentukan sesuai dengan distribusi yang dipilih. Contohnya bila menentukan sebuah *input* proses memiliki distribusi normal, maka perlu menentukan berapa rata-rata dan standar deviasi yang sesuai. Dengan demikian telah mengonversikan data sehingga bisa menjadi *input* simulasi.

4. Melakukan evaluasi distribusi dan parameter distribusi.

   Keyakinan terhadap distribusi dapat dilihat dari nilai *error* antara distribusi yang dipilih dengan data aktual yang dikumpulkan.

Dalam menentukan distribusi yang paling sesuai untuk jenis *input* proses (langkah 2), dapat digunakan dasar yang tercantum berikut:

1. Distribusi Beta

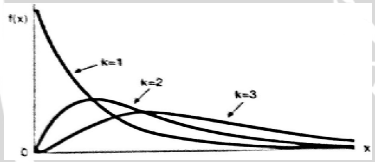


Distribusi beta digunakan untuk merepresentasikan distribusi yang memiliki karakteristik acak seperti jumlah produk cacat. Rumus distribusi beta dapat dilihat pada Persamaan (2 – 2).

$$f(x) = \frac{x^{a_1-1}(1-X)^{a_2-1}}{B(a_1,a_2)} \quad \text{jika } 0 < X < 1 \qquad (2-2)$$

Sumber: Pedgen, et al., (1995)

2. Distribusi Erlang

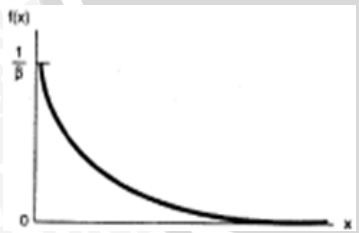


Distribusi erlang memiliki fungsi yang sama seperti distribusi gamma, untuk menggambarkan waktu melakukan pekerjaan. Rumus terdapat pada Persamaan (2 – 3).

$$f(x) = \frac{\beta^{-k} x^{k-1} e^{-x/\beta}}{(k-1)!} \qquad (2-3)$$

Sumber: Pedgen, et al., (1995)

3. Distribusi Eksponensial

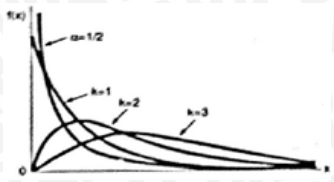


Fungsi eksponensial adalah salah satu fungsi yang paling penting dalam matematika. Variabel acak *exponential* biasanya digunakan untuk mempresentasikan *interarrival* pelanggan, banyaknya kegagalan, dan sebagainya. Rumus distribusi eksponensial dapat dilihat pada Persamaan (2 – 4).

$$f(x) = \frac{1}{\beta} e^{-x/\beta} \quad \text{jika } x > 0 \qquad (2-4)$$

Sumber: Pedgen, et al., (1995)

4. Distribusi Gamma

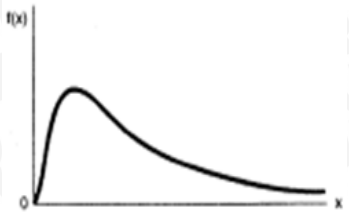


Distribusi gamma memiliki fungsi seperti distribusi erlang, untuk menggambarkan waktu melakukan pekerjaan. Rumus distribusi gamma dapat dilihat pada Persamaan (2 – 5).

$$f(x) = \frac{\beta^{-\alpha} x^{\alpha-1} e^{-x/\beta}}{\tau(\alpha)} \quad \text{jika } x > 0 \qquad (2-5)$$

Sumber: Pedgen, et al., (1995)

5. Distribusi Lognormal

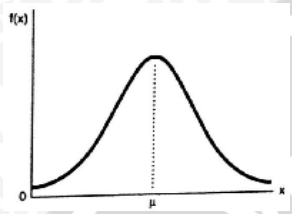


Distribusi lognormal merupakan distribusi probabilitas sebuah variabel acak yang logaritmanya tersebar secara normal. Distribusi lognormal pada umumnya digunakan untuk merepresentasikan waktu kerusakan. Rumus distribusi lognormal dapat dilihat pada Persamaan (2 – 6).

$$f(x) = \frac{1}{\alpha x\sqrt{2\pi}} e^{-(\ln(x)-\mu)^2/2\alpha^2} \quad \text{jika } x > 0 \qquad (2-6)$$

Sumber: Pedgen, et al., (1995)

6. Distribusi Normal

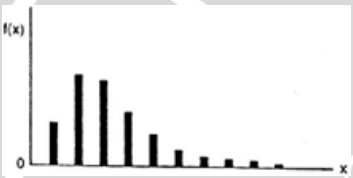


Distribusi normal merupakan sebuah fungsi yang berbentuk lonceng dengan parameter μ (mean) dan σ (standar deviasi). Distribusi normal pada umumnya digunakan untuk menggambarkan proses. Rumus distribusi normal dapat dilihat pada Persamaan (2 – 7).

$$f(x) = \frac{1}{\delta\sqrt{2x}} e^{-(x-\mu)^2/2\delta^2} \qquad (2-7)$$

Sumber: Pedgen, et al., (1995)

7. Distribusi Poisson

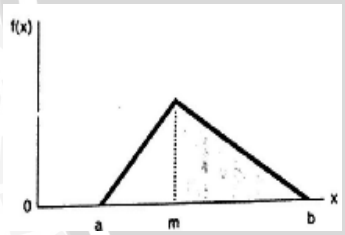


Distribusi Poisson pada umumnya digunakan untuk menggambarkan jumlah kedatangan dalam satu satuan waktu. Rumus distribusi poisson dapat dilihat pada Persamaan (2 – 8).

$$p(x) = \frac{e^{-\lambda}\lambda^x}{x!} \qquad (2-8)$$

Sumber: Pedgen, et al., (1995)

8. Distribusi Triangular



Distribusi triangular digunakan ketika distribusi yang mendasari tidak diketahui, tetapi masuk akal untuk mengasumsikan bahwa nilai berkisar dari beberapa nilai minimal, biasanya untuk mempresentasikan proses. Rumus distribusi triangular dapat dilihat pada Persamaan (2 – 9).

$$f(x) = \frac{2(x-a)}{(m-a)(b-a)} \quad \text{jika } a \le x \le m \qquad (2-9)$$

$$= \frac{2(b-x)}{(b-m)(b-a)} \quad \text{jika } m \le x \le b$$

Sumber: Pedgen, et al., (1995)

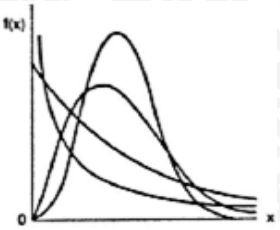
9. Distribusi *Uniform*

Distribusi *uniform* sesuai pada umumnya digunakan variabel acak seragam (*uniform*) umum digunakan karena tidak adanya informasi tentang distribusi yang mendasari yang dimodelkan. Rumus terdapat pada Persamaan (2 – 10).

$$f(x) = \frac{a}{b-a} \quad \text{jika } a \le x \le b \qquad (2-10)$$

Sumber: Pedgen, et al., (1995)

10. Distribusi Weibull



Distribusi weibull biasanya digunakan untuk menyelesaikan masalah-masalah yang menyangkut lama waktu (umur) suatu objek yang mampu bertahan hingga akhirnya objek tersebut tidak berfungsi sebagaimana mestinya (rusak atau mati). Rumus terdapat pada Persamaan (2 – 11).

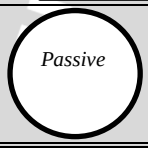
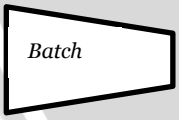
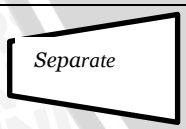
$$f(x) = \alpha\beta^{-\alpha}x^{\alpha-1}e^{-(x/\beta)^{\alpha}} \text{ jika } x > 0 \qquad (2-11)$$

Sumber: Pedgen, et al., (1995)

2.7 Model Konseptual (*Activity Cycle Diagram*)

Activity cycle diagram (ACD) adalah bahasa grafik atau gambar yang memodelkan sistem dengan menunjukkan hubungan interaksi antar elemen dengan perubahan secara diskrit terhadap waktu. Entitas pada ACD ada dua, permanen dan sementara. Sedangkan aktivitas pada ACD ada dua, pasif dan aktif. Pada ACD terdapat simbol-simbol yang dirangkum dalam Tabel 2.2.

Tabel 2.2 Simbol pada *Activity Cycle Diagram*

No.	Simbol	Nama	Fungsi
1.		Segilima ke kanan	Merepresentasikan menciptakan (*create*) atau membangkitkan (*generate*) entitas.
2.		Segilima ke kiri	Merepresentasikan membuang *(dispose*) atau memberhentikan (*terminate*) entitas.
3.		Lingkaran (*passive state*)	Merepresentasikan aktivitas pasif.
4.		Segi empat (*active state*)	Merepresentasikan aktivitas aktif.
5.		Panah (*connect*)	Merepresentasikan relasi urutan antar *node* yang menunjukkan bahwa status atau aktivitas pendahulu berubah atau berlanjut menjadi status atau aktivitas berikutnya.
6.		Belah ketupat (*alternate*)	Merepresentasikan kondisi (*condition*) pilihan dua alternatif kemungkinan yang perlu diputuskan (*decide*).
7.		Trapesium kanan (*assembly* atau *batch*)	Merepresentasikan aktivitas aktif yang melibatkan dua entitas (atau lebih) dan bertransformasi menjadi satu entitas (lain).
8.		Trapesium kiri (*dispers* atau *separate*)	Merepresentasikan aktivitas aktif yang mentransformasikan satu entitas menjadi dua entitas (atau lebih).

2.8 *Software* Simulasi Arena 5.0

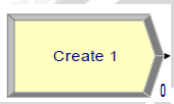
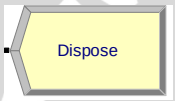
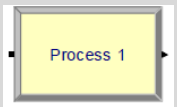
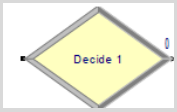
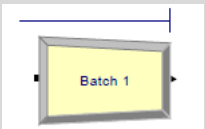
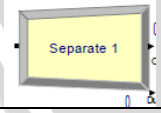
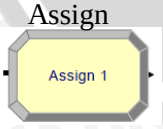
ARENA merupakan sebuah program penyusun model dan juga merupakan simulator. ARENA merupakan kombinasi antara kemudahan pemakaian yang dimiliki *high level program* dan fleksibilitas/kelenturan yang menjadi ciri *general purpose simulation language* (Kelton, 2003).

2.8.1 Modul arena 5.0 *simulation software*

Pembangunan model simulasi pada *software* ARENA 5.0 dilakukan dengan menggunakan modul yang dibagi menjadi *Basic Process Panel* dan *Advanced Process Panel.*

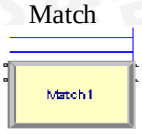
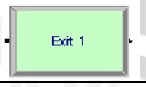
Basic Process Panel pada ARENA 5.0 berisikan modul-modul yang digunakan untuk memodelkan simulai sebuah sistem. Berikut merupakan modul yang terdapat dalam *Basic Process Panel*.

Tabel 2.3 Modul *Basic Process Panel* pada *software* ARENA 5.0

No	Modul	Keterangan
1	*Create* 	Modul ini sebagai titik awal untuk entitas dalam model simulasi. Entitas dibuat menggunakan jadwal atau berdasarkan waktu antara kedatangan. Entitas kemudian meninggalkan modul untuk memulai proses melalui sistem.
2	*Dispose* 	Modul ini sebagai titik awal untuk entitas dalam model simulasi. Entitas dibuat menggunakan jadwal atau berdasarkan waktu antara kedatangan. Entitas kemudian meninggalkan modul untuk memulai proses melalui sistem.
3	*Process* 	*Process module* digunakan untuk mendefinisikan langkah-langkah proses. Server dapat berupa sebuah *resource* atau transporter.
4	Decide 	Modul ini untuk pengambilan keputusan proses dalam sistem. Ini termasuk pilihan untuk membuat keputusan berdasarkan satu atau lebih kondisi atau berdasarkan pada satu atau lebih probabilitas.
5	Batch 	Modul ini sebagai mekanisme pengelompokan dalam model simulasi. *Batch* dapat dilakukan dengan memasukkan sejumlah entitas atau dapat dicocokkan bersama berdasarkan atribut
6	Separate 	Modul ini digunakan untuk menyalin baik entitas yang masuk menjadi beberapa entitas atau untuk membagi sebuah entitas sebelumnya di *batch*.
7	Assign 	Modul ini digunakan untuk menyalin baik entitas yang masuk menjadi beberapa entitas atau untuk membagi sebuah entitas sebelumnya di*batch*.
8	Record	Modul ini digunakan untuk mengumpulkan data statistik dalam model simulasi. Berbagai jenis statistik observasional adalah waktu antara keluar melalui statistik modul, entitas (waktu, biaya).

Advanced Process Panel adalah panel yang memiliki beberapa modul yang memiliki fungsi dan aplikasi proses yang lebih bervariasi dari pada *Basic Process Panel*. Berikut merupakan modul yang terdapat dalam *Advanced Process Panel*.

Tabel 2.4 Modul *Advanced Process Panel* pada *software* ARENA 5.0

No	Modul	Keterangan
1	Hold	Modul ini akan menahan sebuah entitas dalam sebuah antrian dengan beberapa pilihan, yakni: untuk menunggu sinyal, menunggu untuk kondisi tertentu yang diinginkan kemudian dilakukan pemindaian, atau tertahan selama waktu yang tidak terbatas (menjadi hilang kemudian dengan *Remove module*).
2	Match 	*Match module* membawa beberapa entitas sekaligus untuk menunggu di antrian yang berbeda. Saat entitas datang pada *Match Module*, entitas akan tetap pada antriannya sampai terjadi kecocokan. Disaat ada satu kecocokan, satu entitas pada setiap antrian akan lepas dan bertemu. Entitas yang bertemu akan tersinkronisasi untuk meninggalkan modul.
3	Signal	*Signal module* mengirimkan sebuah *signal* atau tanda untuk setiap *hold module* dalam model yang sedang menunggu sinyal untuk melepaskan entitas tertentu. Saat entitas datang ke *signal module*, sinyal dievaluasi dan kode sinyal dikirimkan. pada saat itu, entitas di *Hold Module* yang menunggu untuk sinyal yang sama akan hilang dari antrian.
4	Delay	*Delay module* menunda sebuah entitas dengan spesifikasi waktu tertentu. Saat sebuah entitas datang pada *Delay module*, pernyataan waktu tunda dievaluasi dan entitas tetap pada modul sampai periode waktu tertentu.
5	Release	*Release module* digunakan untuk melepaskan beberapa *resource* yang telah memproses entitas sebelumnya.
6	Seize	*Seize Module* digunakan untuk mengalokasikan unit satuan atau lebih *resource* menjadi entitas. *Seize module* dapat digunakan untuk meraih, menangkap unit dari *resource* khusus, anggota dari *resource set*, atau *resource* yang terdefinisi sebagai metode alternatif, seperti atribut atau ekspresi.
7	Station	*Station module* mendefinisikan sebuah *station* (atau kumpulan *station*) yang cocok secara fisik atau logis lokasi dimana proses dilakukan. Jika *station module* terdefinisi sebagai sebuah kumpulan *station*, maka secara efektif akan menjadi lokasi yang multi proses.
8	Route	*Route module* mentransferkan entitas ke *station* tertentu atau ke *station* selanjutnya di rangkaian *station* kunjungan tertentu untuk entitas. Didalam *route* diasumsikan bahwa *resource* tersedia setiap saat.
9	Access	*Access module* mengalokasikan satu atau lebih *cell* dari *conveyor* ke entitas untuk pemindahan dari satu *station* ke yang lain. Saat entitas dikendalikan dari *cell* pada *conveyor*, kemudian dapat dibawa ke *station* selanjutnya.
10	Convey	*Convey module* memindahkan entitas pada *conveyor* dari lokasi tertentu ke *station* tujuan. Waktu tunda untuk membawa entitas dari satu *station* ke selanjutnya berdasarkan kecepatan *conveyor* jarak antar *station*.
11	Exit 	*Exit module* melepaskan entitas *cell* di *conveyor* tertentu. Jika entitas yang lain menunggu dalam antrian *conveyor* di *station* yang sama ketika *cell* itu lepas, entitas kemudian masuk ke *conveyor*.
12	Request	Sebuah unit *transporter* spesifik dapat ditentukan atau seleksi berdasarkan aturan. Ketika entitas tiba di*request module*, maka entitas akan dialokasikan *transporter* ketika salah satu tersedia. Entitas tetap pada *request module* sampai unit *transporter* telah mencapai stasiun entitas. Kemudian entitas bergerak keluar dari *module Request*.
13	Transport	*Transport module* mentransfer entitas pengendali ke stasiun tujuan. Setelah penundaan waktu yang diperlukan untuk transportasi, entitas muncul kembali dalam model di *module Station*.
14	Free	*Free module* melepaskan *entity* yang terakhir dialokasikan di *transporter* unit. Jika entitas lain sedang menunggu dalam antrian untuk diminta atau mengalokasikan *transporter*, maka *transporter* akan diberikan kepada entitas tersebut.

2.8.2 *Input analyzer*

Input Analyzer merupakan bagian dari *software* ARENA. *Tools* ini digunakan untuk menentukan fungsi distribusi probabilitas dari data *input.* Selain itu juga dapat digunakan untuk mencocokkan fungsi spesifik dari distribusi data *file* dan membandingkan fungsi distribusi atau untuk menampilkan efek dari perubahan parameter untuk distribusi yang sama. *Input Analyzer* menampilkan *input* data acak tersebut yang kemudian dapat dianalisis menggunakan fitur perangkat lunak *fitting distribution* untuk mencari bentuk distribusi yang cocok menggambarkan data tersebut. Data yang akan dimasukkan sebelumnya harus disimpan dalam *Notepad* dengan format.*txt* karena *Input Analyzer* ARENA hanya dapat membaca masukan dari format.*txt*.

2.8.3 *Process analyzer*

Process Analyzer alat yang mendukung dalam mengevaluasi alternatif yang disajikan oleh eksekusi model untuk membuat, menjalankan, dan membandingkan skenario simulasi yang berbeda, dengan demkian dapat diamati efek dari skenario yang telah diatur sebelumnnya. Hal ini berguna untuk pengembangan model simulasi, serta pembuatan keputusan dimana tidak dikenal dengan model, namun akrab disebut dengan menangani solusi model simulasi. Biasanya *process analyzer* untuk menentukan skenario mana yang cocok sehingga bisa mendapatkan tujuan yang diinginkan.

2.9 Manual Kapasitas Jalan Indonesia (MKJI)

Manual Kapasitas Jalan Indonesia (MKJI) memuat fasilitas jalan perkotaan dan semi perkotaan dan juga jalan luar kota dan jalan bebas hambatan. Manual ini menggantikan manual sementara untuk fasilitas lalu lintas perkotaan (Januari 1993) dan jalan luar kota (Agustus 1994) yang telah diterbitkan lebih dahulu dalam proyek MKJI. Berikut merupakan beberapa faktor penentuan waktu sinyal pada simpang bersinyal (Manual Kapasitas Jalan Indonesia, 1997):

1. Derajat Kejenuhan

   Digunakan untuk mengetahui kepadatan/kejenuhan suatu arus jalan yang dapat dihitung dengan menggunakan persamaan berikut

   $$DS = Q/C = (Q \times c)/(S \times g) \qquad (2 - 12)$$

   Sumber: Manual Kapasitas Jalan Indonesia (1997)

   dengan:

   Q = arus lalu lintas smp/jam

   C = kapasitas smp/jam

2. Waktu siklus

Digunakan untuk pengendalian waktu tetap yang dapat dihitung dengan menggunakan persamaan sebagai berikut

$$c_{ua} = (1{,}5 \times LTI + 5)/(1 - IFR) \quad (2 - 13)$$

Sumber: Manual Kapasitas Jalan Indonesia (1997)

dengan:

c_{ua} = waktu siklus sebelum penyesuaian sinyal (det)
LTI = waktu hilang total per siklus (det)
IFR = rasio arus simpang

3. Waktu hijau

Untuk mendapatkan waktu hijau dapat menggunakan persamaan berikut

$$g_i = (C_{ua} - LTI) \times PR_i \quad (2 - 14)$$

Sumber: Manual Kapasitas Jalan Indonesia (1997)

dengan:

g_i = tampilan waktu hijau pada fase ke i (det)
c_{ua} = waktu siklus sebelum penyesuaian sinyal (det)
LTI = waktu hilang total per siklus (det)
PR_i = rasio fase

2.10 Verifikasi Dan Uji Validitas Model

Verifikasi dilakukan untuk memastikan model telah terbangun dengan tepat pada *software* simulasi, hal ini dapat dilakukan dengan beberapa cara yaitu (Banks, et al., 2004):

1. Membuat diagram alir konseptual yang mampu menggambarkan setiap logika proses dari sistem dan membandingkannya dengan model pada *software* simulasi.
2. Melihat rangkuman proses pada model untuk melakukan pengecekan terhadap *input* parameter, satuan, serta variabel yang digunakan pada model.
3. Melakukan pengecekan terhadap animasi dari model untuk melihat apakah jalannya model mampu meniru dan sesuai dengan sistem nyata.
4. Melakukan kompilasi *error* dari model simulasi yang dibuat. Pada umumnya, *software* simulasi memiliki *Interactive Run Controller* (IRC) atau *debugger* yang secara otomatis melakukan pengecekan terhadap model yang telah dibuat dan mengidentifikasi *error* pada model.

Uji validitas dapat dilakukan dengan *face validity* dan membandingkan hasil *output* terhadap *input* yang diberikan antara model dengan sistem nyata menggunakan uji statistik, pada umumnya digunakan uji-t (Banks, et al., 2004).

2.11Penentuan Jumlah Replikasi

Penentuan jumlah replikasi yang sebaiknya dilakukan saat menjalankan simulasi memiliki hubungan terhadap *confidence interval*. Adapun langkah-langkah dalam menentukan jumlah replikasi adalah sebagai berikut:

1. *Run* model dengan jumlah replikasi kecil
2. Dari *report* simulasi, dapat melihat *output* simulasi untuk menghitung standar deviasi.
3. Selanjutnya melakukan perhitungan terhadap berapa jumlah replikasi yang sebaiknya dilakukan untuk mendapatkan *error* tertentu. Caranya adalah dengan memasukannya ke dalam rumus perhitungan *half-width*. Rumus dapat dilihat sebagai berikut:

$$Half\ width = \frac{\left(t_{n-1,\frac{\alpha}{2}}\right) \text{x s}}{\sqrt{n}} \qquad (2-15)$$

dengan:

$t_{n-1;\alpha/2}$ = Nilai pada tabel T
n = Jumlah replikasi awal
α = Error yang diinginkan
s = Standar deviasi hasil simulasi