Department of Systems Engineering

George Mason University

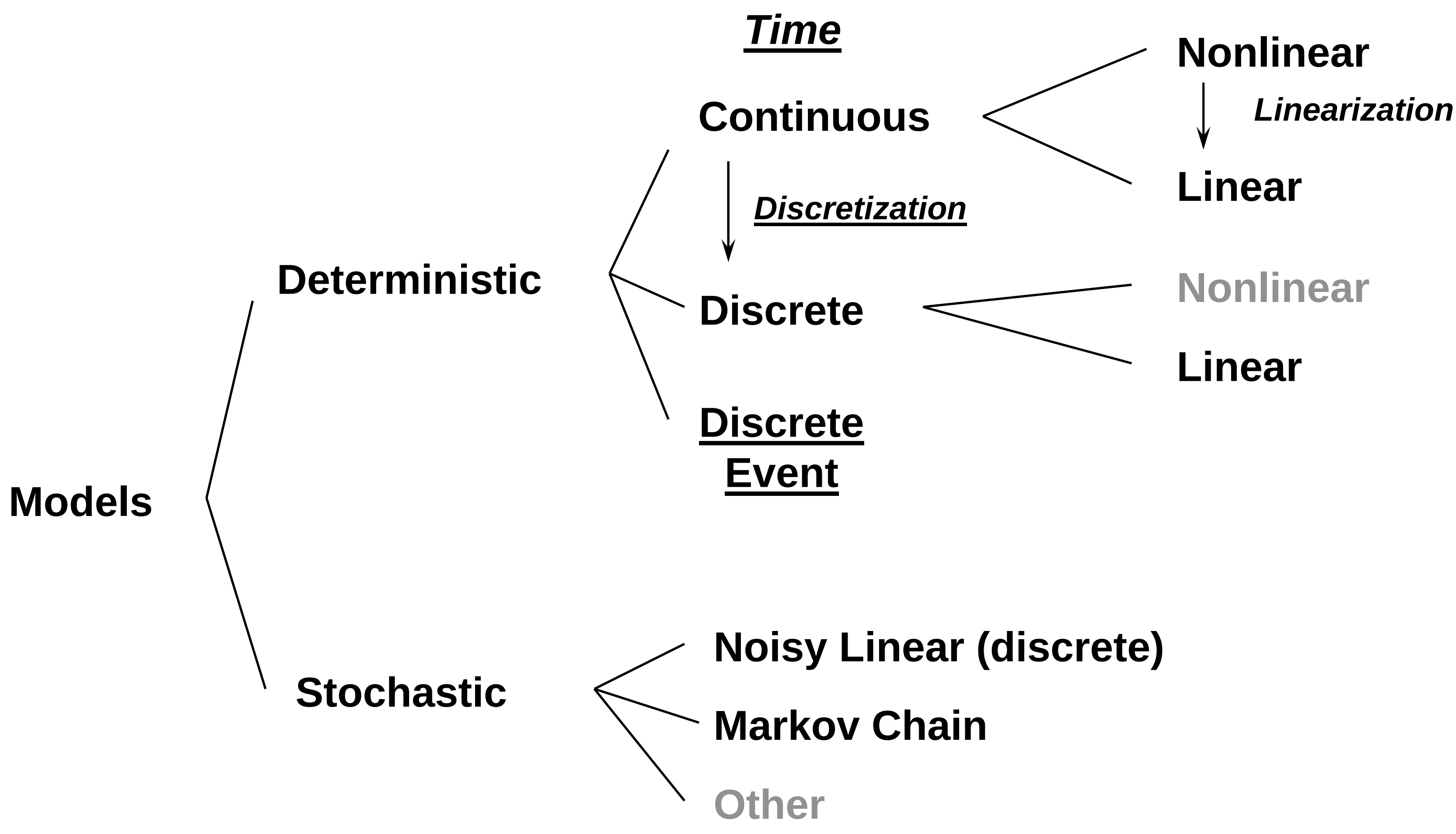
SYST611: Systems Methodology and Modeling #6

Kuo-Chu Chang
Fairfax, Virginia

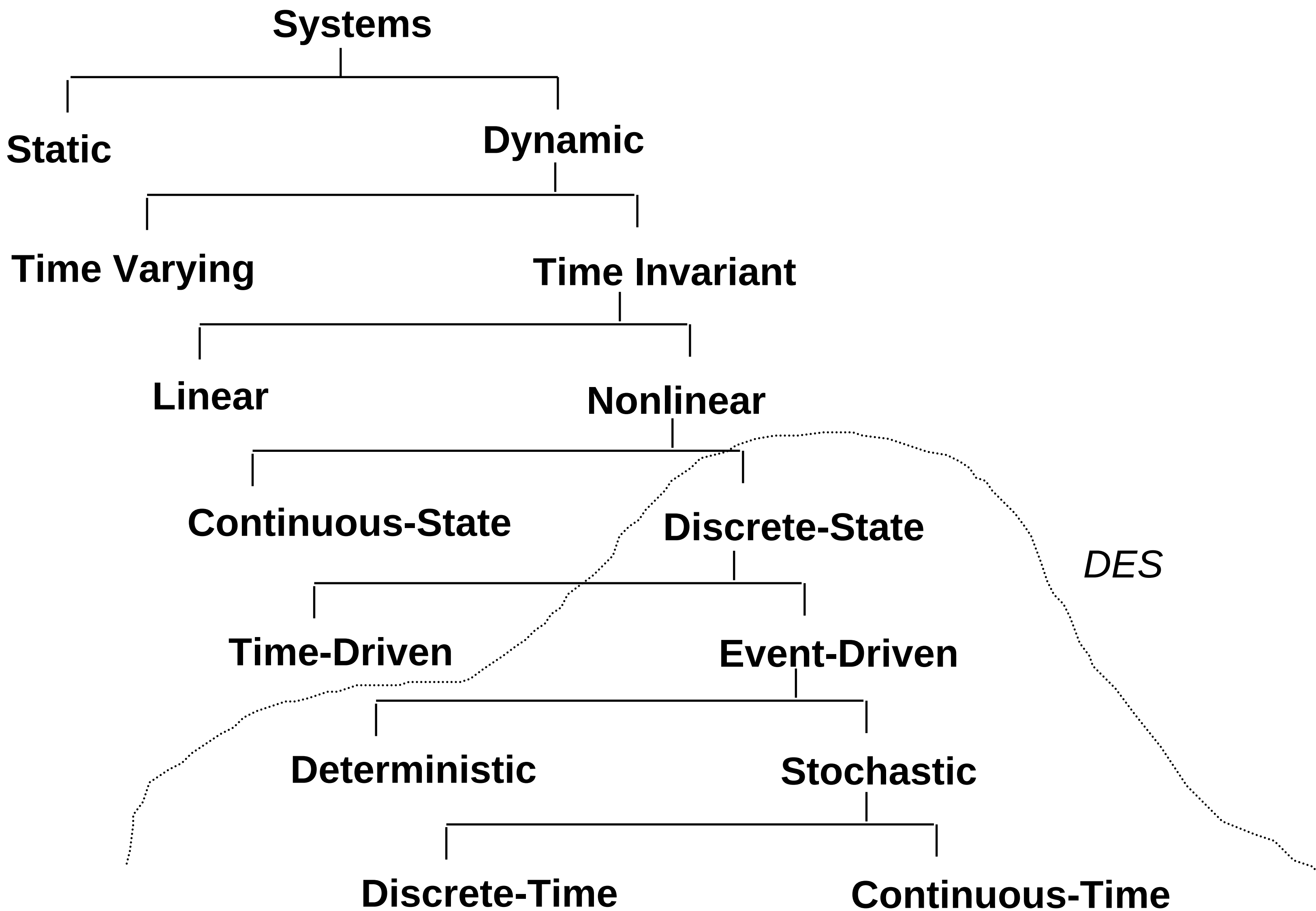
Discrete Event Systems and Systems Overview

- Discrete Event Systems
- Timed and Untimed Systems
- Petri Nets Model
- Queuing System Example
- Analyzing an DES
- Overview on Dynamic Systems

Taxonomy of Models



System Classification



Discrete Event Systems

- **Continuous-Variable Dynamic Systems (CVDS)**
  - Continuous-state systems
  - The state transition mechanism is time-driven
- **Discrete Event Systems (DES)**
  - The state space is a discrete set
  - The state transition mechanism is event-driven
- **Definition**
  - A DES is a discrete-state, event-driven system, that is, its state evolution depends entirely on the occurrence of asynchronous discrete events over time
  - Examples: state of a machine, a computer, inventory, board games, queuing systems, etc.

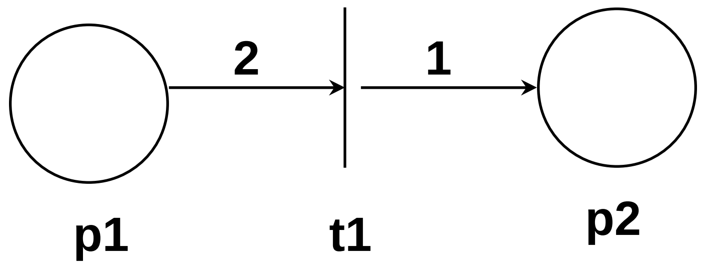
Timed and Untimed Models

- **DES Essential Elements**
  - **A discrete state space**
  - **A discrete-event set**
- **Untimed Models**
  - **An event sequence** $\{e_1, e_2, \ldots\}$ **determines the input without any information about the occurrence times**
  - **Focus on logical behavior of the system**
  - ***State automata*** **and** ***Petri nets***
- **Timed Models**
  - **A timed event sequence** $\{(e_1, t_1), (e_2, t_2), \ldots\}$ **determines the input**
  - **Can answer questions related to time**
  - ***Timed State automata, Timed Petri nets, Dioid Algebra***

Petri Nets Model

- **Petri Net**
  - A device to manipulate events according to certain rules
  - Include explicit conditions to enable an event
  - Can represent very general DES
- **Notations**
  - Transitions: events (bar)
  - Places: contain conditions for transition to occur (circle)
  - Arcs: transition to place or place to transition
  - Weights: one for each arc
- **Definitions**
  - Petri net is a four-tuple (P,T,A,w)
  - P: a finite set of places
  - T: a finite set of transitions
  - A: a set of arcs (PxT)U(TxP)
  - w: a positive integer weight function

Petri Nets Example



P={p1,p2},

T={t1},

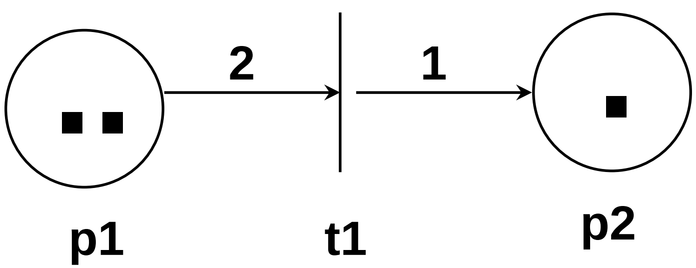
A={(p1,t1),(t1,p2)}

w(p1,t1)=2,

w(t1,p2)=1

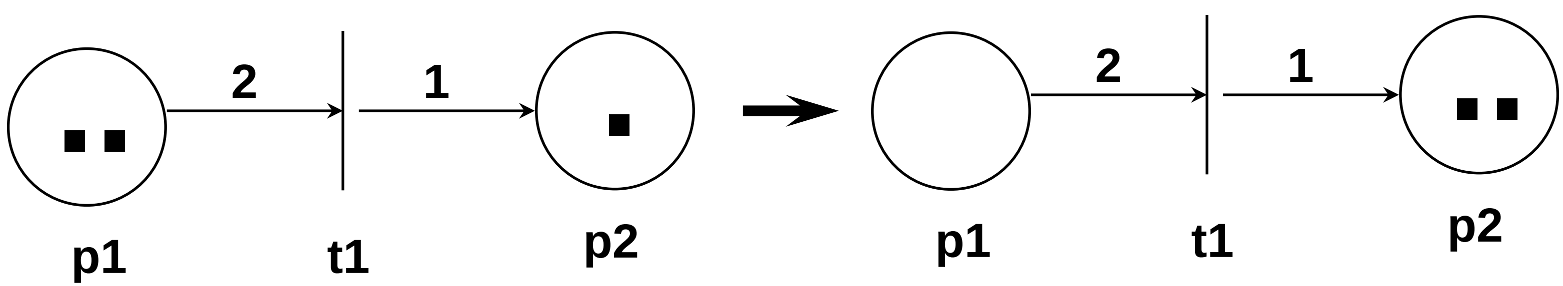
Petri Nets Marking

- **Marking**
  - Assign tokens (nonnegative integer) to places
  - Indicate the conditions described by the place is satisfied
  - Define a row vector [x(p1), x(p2), …, x(pn)]
- **Definitions**
  - A marked Petri net is a five-tuple (P,T,A,w,x0)
  - x0 is an initial marking
- **Example**
  - x0 =[2, 1]



Petri Nets Dynamics

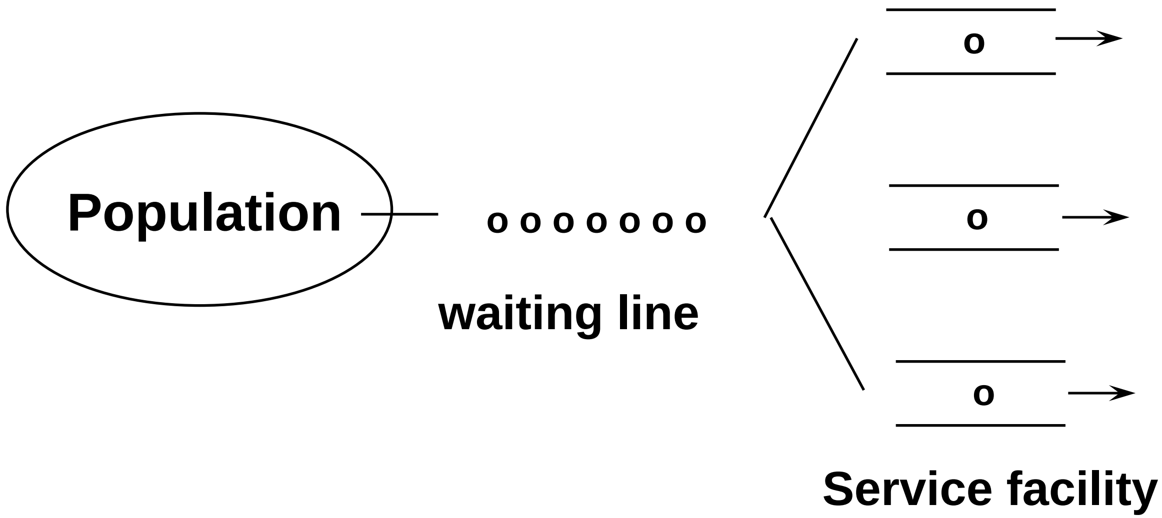
- **Definitions**
  - State: marking of the net, x=[x(p1), x(p2), …, x(pn)]
  - Enabled: A transition tj is *enabled (can fire)* if x(pi)>=w(pi,tj)
- **State Dynamics**
  - The state transition function for tj is defined when x(pi)>=w(pi,tj)
  - Firing a transition causes the change in the marking
  - x'(pi) = x(pi) - w(pi,tj) + w(tj,pi)
- **Example**
  - x0 =[2, 1],   x1=[0,2]



Queuing Systems

- **The Queuing System (Waiting-Line)**
  - A facility or a group of facilities for service
  - A population of individuals or units which form a line to be served with a pre-determined waiting-line discipline
  - Single-channel or multiple-channel
  - Single-stage or multiple-stage
- **Common Examples**
  - The public forms waiting-lines at grocery stores, theaters, doctor’s office, etc.
  - Items in process produces a waiting-line at each machine center in a production system
  - Automobiles form waiting-lines at toll-gates, traffic signals, docks in a transportation system

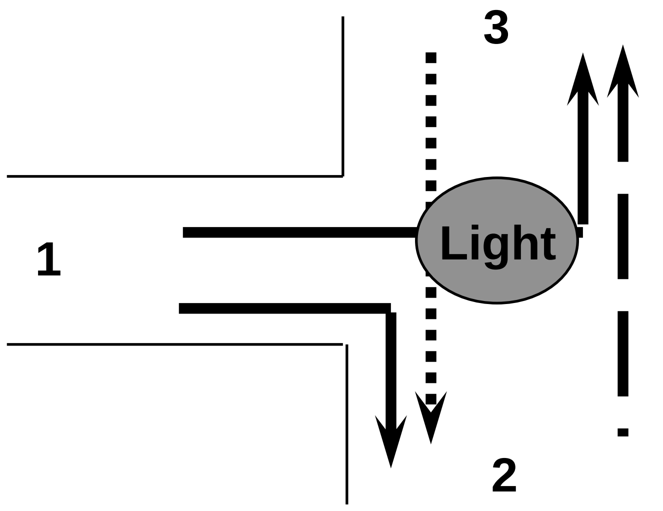
Queuing Examples



Event set = { arrival, service completion 1, .. , service completion 3 }

State space X = { 0, 1, 2, 3, ... } : length of queuing line

Traffic Systems Example



Event set = { a12, a13, a23, a32, d12, d13, d23, d32, g, r }

aij: arrival i to j, dij: departure i to j
g,r: green or red light for 12 and 13

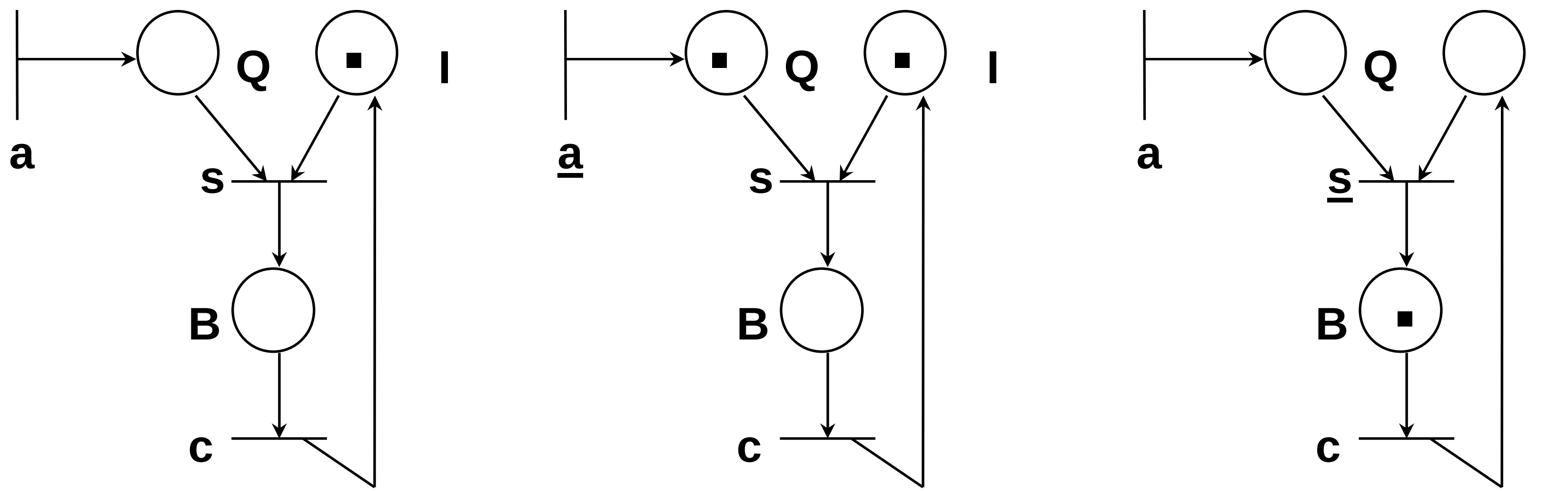
State space X = { (x12,x13,x23,x32,y): x12,x13,x23,x32>=0, y: {g,r} }

xij: queue lengths for each path

Petri Net Model for Queuing Systems

- **Net Model**
  - Places: Q(queue), I (idle), B(busy)
  - Transitions (events): a (customer arrives), s (service starts), c (service completes)
- **State Transition**

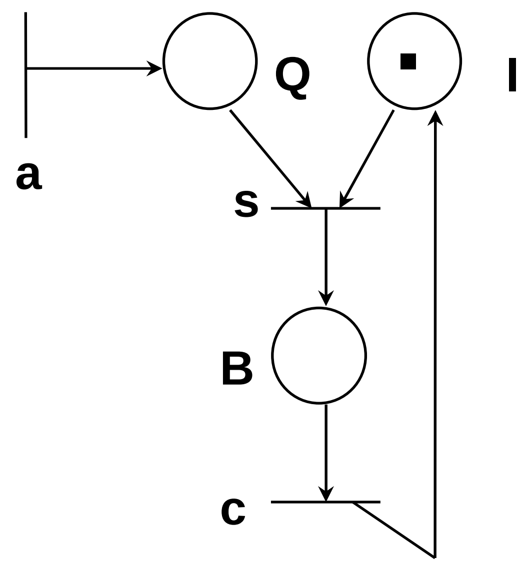
Firing sequence { a, s }
x0 = [0,1,0], x1 = [1,1,0], x2 = [0,0,1]



State Transition Equation

- **State Dynamic**
  - Define a firing vector given a particular transition firing u=[0,0,..,0,1,0,…0]
  - Define an incidence matrix A= [aji] = [w(tj,pi)-w(pi,tj)]
  - State equation: x' = x +uA
- **Example**

$$A = \begin{bmatrix} 1 & 0 & 0 \\ -1 & -1 & 1 \\ 0 & 1 & -1 \end{bmatrix}$$



State Transition Example

- **Firing Sequence**
  - **Given initial state x0 = [0 1 0]**
  - **Firing sequence {a, s, a, a, c, s, a}**

$$x_1 = x_0 + u_0 A = [0\ 1\ 0] + [1\ 0\ 0]\begin{bmatrix} 1 & 0 & 0 \\ -1 & -1 & 1 \\ 0 & 1 & -1 \end{bmatrix} = [1\ 1\ 0]$$

$$x_2 = x_1 + u_1 A = [1\ 1\ 0] + [0\ 1\ 0]A = [0\ 0\ 1]$$

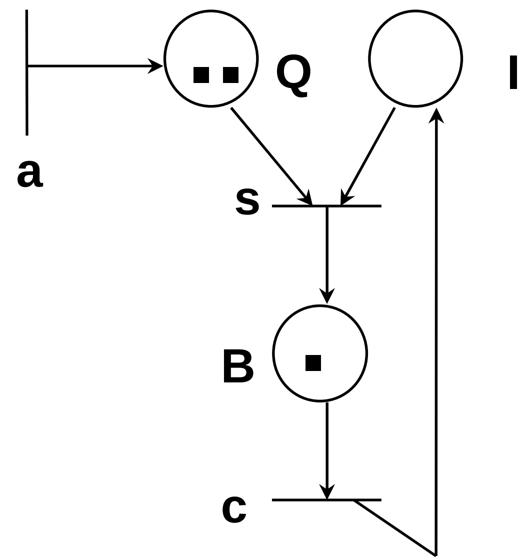
$$x_3 = x_2 + u_2 A = [0\ 0\ 1] + [1\ 0\ 0]A = [1\ 0\ 1]$$

$$x_4 = x_3 + u_3 A = [1\ 0\ 1] + [1\ 0\ 0]A = [2\ 0\ 1]$$

$$x_5 = x_4 + u_4 A = [2\ 0\ 1] + [0\ 0\ 1]A = [2\ 1\ 0]$$

$$x_6 = x_5 + u_5 A = [2\ 1\ 0] + [0\ 1\ 0]A = [1\ 0\ 1]$$

$$x_7 = x_6 + u_6 A = [1\ 0\ 1] + [1\ 0\ 0]A = [2\ 0\ 1]$$



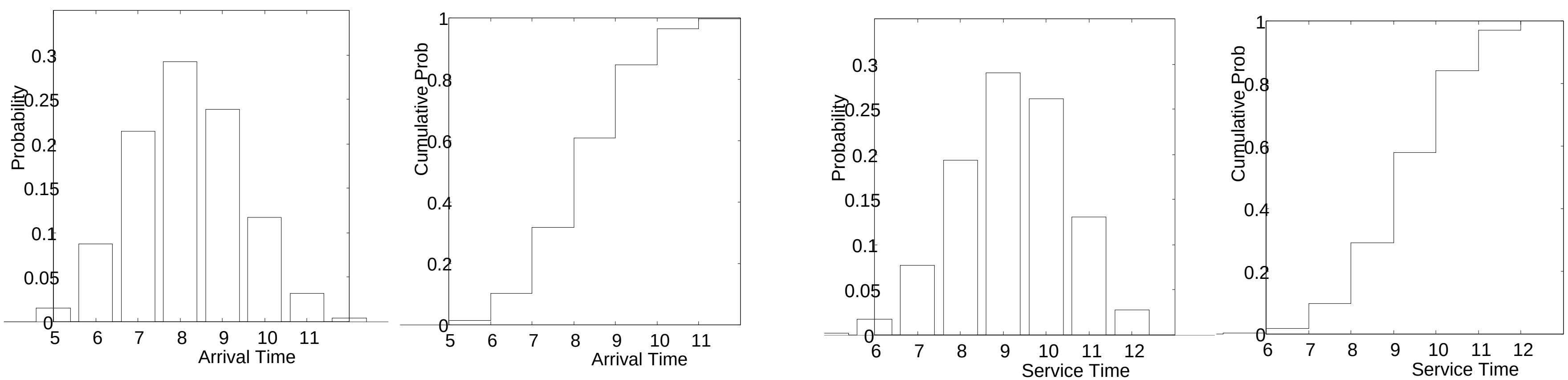
Analyzing a DES

- **DES are inherently complex**
  - Hard to analyze regardless of the modeling framework adopted
  - Lack of mathematical techniques for analysis and design
- **Discrete-Event Simulation**
  - Timed and untimed models provide solid foundations for analysis but still too complex to manage most of the time, they are good for special cases such as Markov chains
  - *Discrete-event simulation* is the only universal tool
- **Simulation Techniques**
  - Sensitivity estimation
  - Sample path constructability techniques

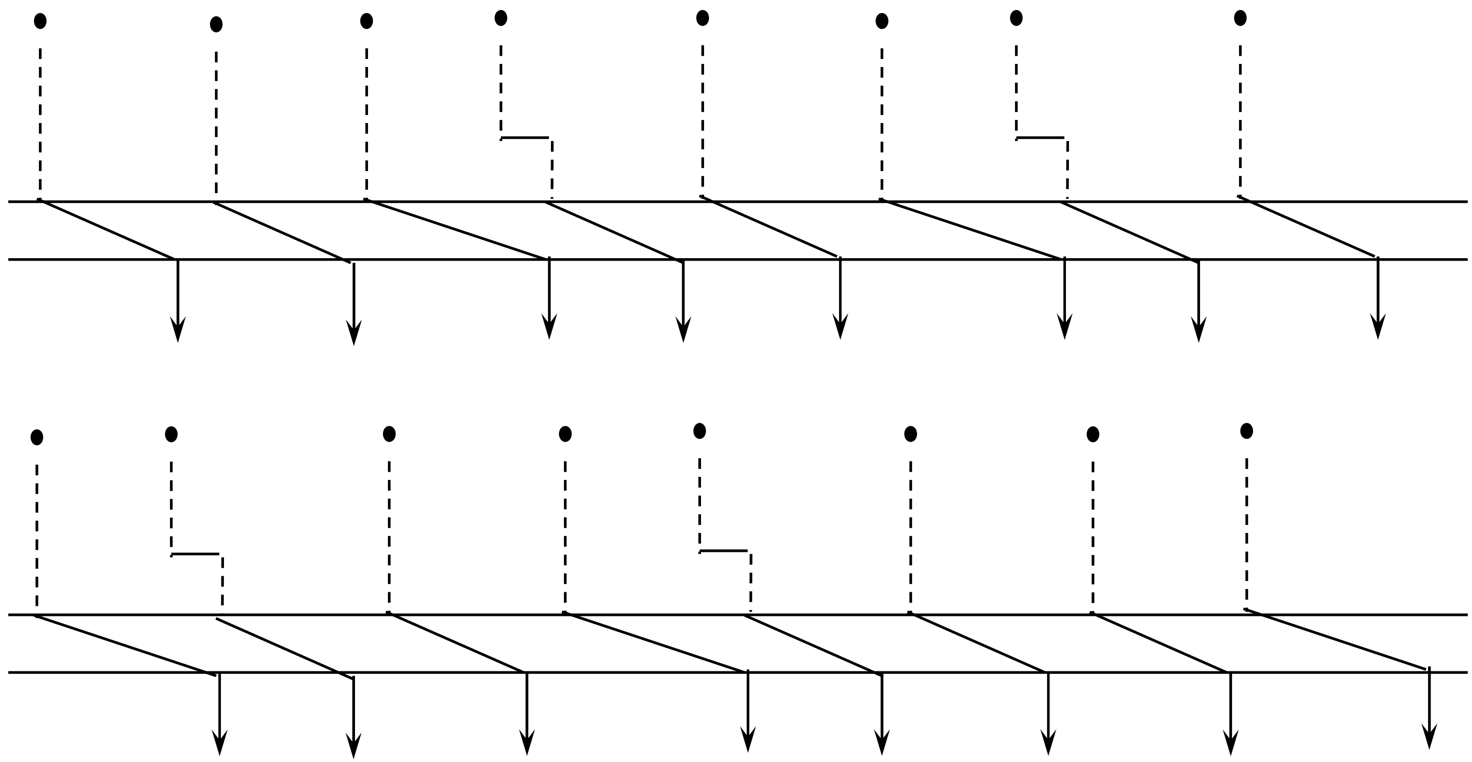
Ref: Discrete Event Systems - Modeling and performance Analysis, Christos G. Cassandras, IRWIN, Boston, 1993

Monte Carlo Analysis

- **The Mechanism**
  - **Uncertain Waiting-line decision models usually based on assumptions of the arrival and service-time distributions**
  - **Formal mathematic solutions may be difficult or impossible in general**
  - **Monte Carlo analysis does not require these distributions to obey certain theoretical forms**
  - **Provides analysis tool through simulation**



Single Channel Queuing MC Analysis



Economic Analysis:

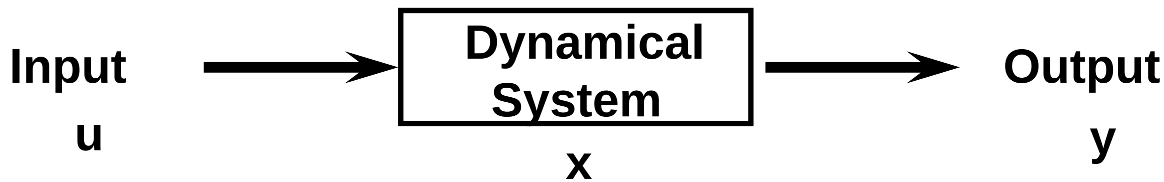
Total cost = unit-waiting-cost-per-time-period * (waiting-time-in-queue + waiting-time-in-service)
+ service-cost-per-time-period * service-time

Modeling Overview for Dynamical Systems

- **Model Uncertainties**
  - **<u>Deterministic models:</u>** knowledge of the state at time ***t0*** and the input for (***t0,t1***) yields the state and the output at time ***t1***
  - **<u>Stochastic models:</u>** knowledge of the state at time ***t0*** and the input for (***t0,t1***) yields a probabilistic description of the state and the output at time ***t1***
- **Time**
  - **<u>Continuous-time system:</u>** defined over a continuous time interval
  - **<u>Discrete-time system:</u>** defined only over a discrete time set
- **Mathematical Form**
  - **<u>Linear:</u>** approximations over operating range
  - **<u>Nonlinear:</u>** almost all physical systems are nonlinear

System Representation

- **Input-Output representation**
  - **<u>Inputs:</u> external forces**
  - **<u>Outputs:</u> observable measures of the resulting behavior**

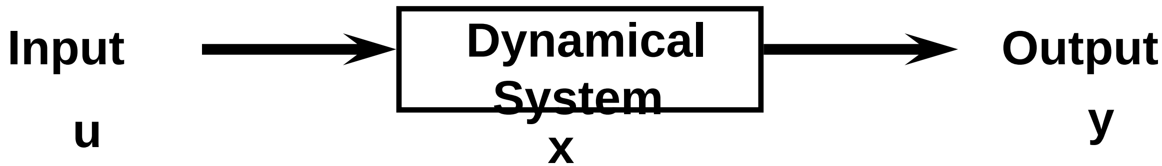


- **State Representation**
  - **State: a set of minimum number of variables**
  - **Knowledge of the value of this set of variables at a time *t0*, and of the forcing functions from *t0* to *t1*, where *t1* > *t0*, is sufficient information to determine the output of this system at any time *t1* > *t0***

- **Mathematical Model of a Physical Dynamical System (PDS)**
  - **Describes the inter-relationship between the state, the input, and the output**
  - **Independent variable: time, discrete or continuous**

Basic Systems Concept

- **Linearity and Superposition**
  - **If an input u1(t) produces an output y1(t) and an input u2(t) produces an output y2(t), then input c1u1(t)+c2u2(t) produces an output c1y1(t)+c2y2(t) for all pairs of inputs u1(t) and u2(t) and all pairs of constants c1 and c2**
  - **It's called the *principle of superposition***

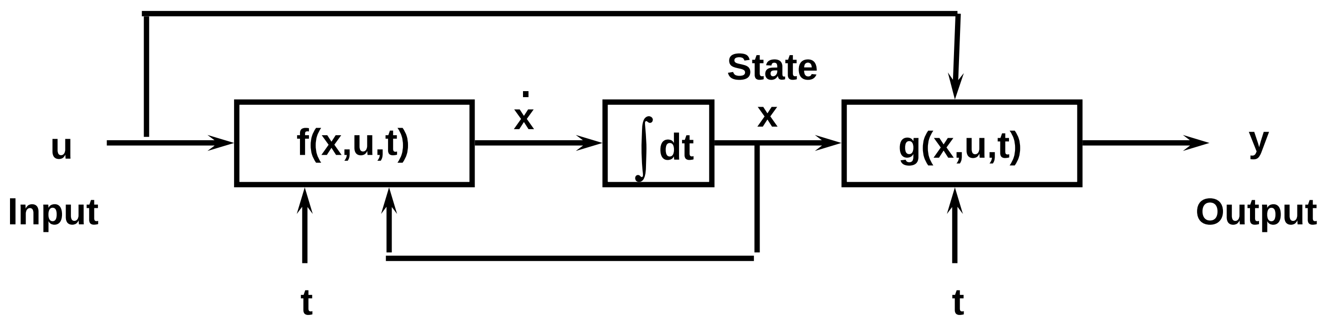


- **Causality and Physically Realizable Systems**
  - **A system is called *causal* if the output depends only on the present and the past values of the input.**
  - **Causal systems are sometimes called *physically realizable systems***

System models

- **Continuous-Time Systems**
  - **Ordinary differential equations**
  - $\dot{x}(t) = f(x, u, t)$

    $y = g(x, u, t)$



- **Discrete-Time Systems**
  - **Difference equations**

    $x_{k+1} = f(x_k, u_k, k)$

    $y_k = g(x_k, u_k, k)$

Input-Output Representation

- **Discrete Linear Systems: Difference Equations**

$$a_n y(k+n) + a_{n-1} y(k+n-1) + \cdots + a_0 y(k) = b_m u(k+m) + b_{m-1} u(k+m-1) + \cdots + b_0 u(k)$$

$$\Rightarrow \sum_{i=0}^{n} a_i y(k+i) = \sum_{j=0}^{m} b_j u(k+j) \Rightarrow \left(\sum_{i=0}^{n} a_i z^i\right) Y(z) = \left(\sum_{j=0}^{m} b_j z^j\right) U(z) \Rightarrow Y(z) = H(z) U(z)$$

$$H(z) = \frac{\left(\sum_{j=0}^{r} b_j z^j\right)}{\left(\sum_{i=0}^{n} a_i z^i\right)} \text{ is the } Transfer\ \ Function, \left(\sum_{i=0}^{n} a_i z^i\right) \text{ is the } Characteristic\ Polynomial$$

- **Continuous Linear Systems: Differential Equations**

$$a_n y^{(n)}(t) + a_{n-1} y^{(n-1)}(t) + \cdots + a_1 y^{(1)}(t) + a_0 y(t) = b_m u^{(m)}(t) + b_{m-1} u^{(m-1)}(t) + \cdots + b_0 u(t)$$

$$\Rightarrow \sum_{i=0}^{n} a_i \frac{d^i y}{dt^i} = \sum_{i=0}^{m} b_i \frac{d^i u}{dt^i} \Rightarrow \left(\sum_{i=0}^{n} a_i s^i\right) Y(s) = \left(\sum_{i=0}^{m} b_i s^i\right) U(s) \Rightarrow Y(s) = H(s) U(s)$$

$$H(s) = \frac{\left(\sum_{i=0}^{m} b_i s^i\right)}{\left(\sum_{i=0}^{n} a_i s^i\right)} \text{ is the } Transfer\ \ Function, \left(\sum_{i=0}^{n} a_i s^i\right) \text{ is the } characteristic\ polynomial$$

Note: for causal systems, n is greater or equal to m

State Representation

- **Discrete Linear Systems: Vector Difference Equations**

$$\boldsymbol{x}(k+1) = \boldsymbol{F}\boldsymbol{x}(k) + \boldsymbol{b}u(k), \quad \text{Note: } \det\left[\boldsymbol{F} - \lambda I\right] = \left(\sum_{i=0}^{n} a_i \lambda^i\right)$$

- **Continuous Linear Systems: Vector Differential Equations**

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{b}u(t), \quad y(t) = \mathbf{c}\boldsymbol{x}(t) \quad \text{Note: } \det\left[\boldsymbol{A} - \lambda I\right] = \left(\sum_{i=0}^{n} a_i \lambda^i\right)$$

- **From input-output to state representation (observable canonical form)**

$$\mathbf{A} = \begin{bmatrix} -a_{n-1}/a_n & 1 & 0 & \cdots & 0 \\ -a_{n-2}/a_n & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -a_1/a_n & 0 & 0 & \cdots & 1 \\ -a_0/a_n & 0 & 0 & \cdots & 0 \end{bmatrix} \qquad \mathbf{b} = \begin{bmatrix} b_{n-1}/a_n \\ b_{n-2}/a_n \\ \vdots \\ b_1/a_n \\ b_0/a_n \end{bmatrix}$$

$$y = x_1 = \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix}\mathbf{x}$$

- **From state to input-output representation**

$$y = \mathbf{c}(\mathbf{DI} - \mathbf{A})^{-1}\mathbf{b}\,u$$

Linear System Solutions

Continuous Linear Systems:

$$\dot{x}(t) = Ax(t) + Bu(t); \quad x(0) = x_0$$

Solution:

$$x(t) = x_h(t) + x_p(t) = e^{At}x_0 + \int_0^t e^{A(t-\tau)} Bu(\tau)d\tau$$

Discrete Linear Systems:

$$x_{k+1} = Fx_k + Gu_k; \quad x_0 \ given$$

Solution:

$$x_k = F^k x_0 + \sum_{i=0}^{k-1} F^{k-1-i} Gu_i$$

Stability

- **<u>Stable</u>: A system is *stable* if *unforced* solution x(t) is bounded for all t**
  - **<u>Asymptotically Stable</u>: If a system is stable and x(t) tends to 0 when t goes to infinite**
  - **<u>Marginally Stable</u>: If a system is stable and not asymptotically stable**
- **Stability Criteria - Discrete Systems**

$$\|\lambda_i\| \le 1, \ \forall\, i \rightarrow \ Stable$$
$$\|\lambda_i\| < 1, \ \forall\, i \rightarrow \ Asymptotically \quad Stable$$

- **Stability Criteria - Continuous Systems**

$$\mathrm{Re}\,(\lambda_i) \le 0, \ \forall\, i \rightarrow \ Stable$$
$$\mathrm{Re}\,(\lambda_i) < 0, \ \forall\, i \rightarrow \ Asymptotically \quad Stable$$

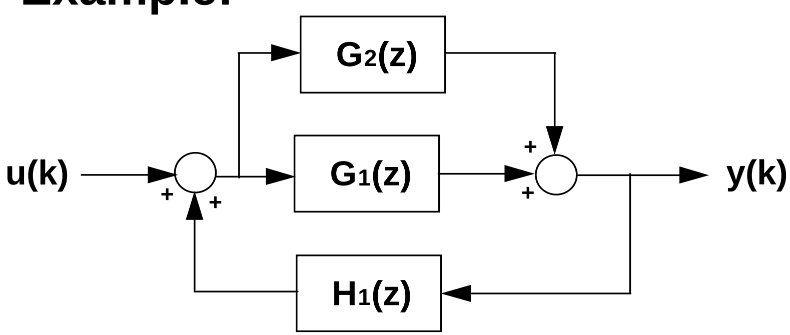
Note: Routh test can be used to check stability criteria

Interconnected Systems and Block Diagram

- **A Short-Hand Pictorial Representation of System I/O Relationship**
  - Describes Interconnected systems
  - Decomposes complex system problems into simpler ones
  - Characterizes the functional relationship between system components

- **Derive Overall Transfer Function**

Example:



$$\Rightarrow H(z) = \frac{G_1(z) + G_2(z)}{1 - H_1(G_1(z) + G_2(z))}$$

$$Y(z) = H(z)U(z) \Rightarrow y(k) = Z^{-1}(Y(z))$$

Systems Analysis and Design

- **System Characteristics**
  - Degree or extent of system stability
  - Steady state performance (response doesn't approach zero)
  - Transient performance (response approaches zero when t $\rightarrow \infty$)
- **Analysis Methods**
  - Determine transfer functions for each system components
  - Formulate system model by integrating components (Block Diagram)
  - Determine the system response characteristics (Root-locus, Bode-plot, Nyquist, Nichols)
- **Design Objectives**
  - Meet performance specifications
  - Frequency/Time domain responses
  - Speed of response, relative stability, system accuracy

Discretization of Continuous Systems

- **Motivation**
  - System simulation and analysis using digital computers and numerical methods
  - Digital signal processing and control for real-time applications
- **Discretization Methods**
  - Sample and hold (input stays constant)

    $$x_d((k+1)h) = Fx_d(kh) + Gu(kh), \quad F = e^{Ah},\ G = A^{-1}(F-I)B$$

  - Forward difference (replace derivative by difference quotient)

    $$x_d((k+1)h) = (I + hA)x_d(kh) + hBu(kh)$$

  - Backward difference

    $$x_d((k+1)h) = (I - hA)^{-1}x_d(kh) + h(I - hA)^{-1}Bu((k+1)h)$$

  - Trapezoidal method

    $$x_d((k+1)h) = \left(I - \frac{hA}{2}\right)^{-1}\left(I + \frac{hA}{2}\right)x_d(kh) + \frac{1}{2}\left(I - \frac{hA}{2}\right)^{-1}Bh\big(u((k+1)h) + u(kh)\big)$$

Issues in Discretization

- **Accuracy**
  - Sample-and-hold is exact when system input remains constant between sampling times
  - Forward and Backward difference provide first-order approximation
  - Trapezoidal method provides second-order accuracy
- **Stability**
  - Sample-and-hold preserves stability and instability
  - Forward discretization preserves instability but not stability
  - Backward discretization preserves stability but not instability
  - Trapezoidal method preserves both stability and instability
- **Computational Requirements**
  - Smaller sampling interval requires more computation

Nonlinear Systems and Linearization

- **Nonlinear System Equations**

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)); \qquad \mathbf{x}(0) = \mathbf{x}_0$$

$$y(t) = h(\mathbf{x}(t))$$

- **Linearization about Equilibrium Point**

With $\mathbf{x} = \mathbf{x_0} + \Delta\mathbf{x},$

$$\mathbf{f}(\mathbf{x}) \approx \mathbf{f}(\mathbf{x_0}) + \mathbf{f}'(\mathbf{x_0})(\mathbf{x} - \mathbf{x_0})$$

but $\dfrac{d}{dt}\Delta\mathbf{x}(t) = \dot{\mathbf{x}}(t) - \dot{\mathbf{x}}_\mathbf{0}(t) = \mathbf{f}(\mathbf{x}(t)) - \mathbf{f}(\mathbf{x_0}(t))$

$\Rightarrow \dfrac{d}{dt}\Delta\mathbf{x}(t) \approx \mathbf{f}'(\mathbf{x_0}(t))\Delta\mathbf{x}(t)$ linearized system w.r.t. $\mathbf{x_0}(t)$

Discretize Nonlinear Systems

Given $\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t))$

Forward Difference Method:

$$\mathbf{x}((k+1)h) = \mathbf{x}(kh) + h\mathbf{f}(\mathbf{x}(kh), \mathbf{u}(kh))$$

Backward Difference Method:

$$\mathbf{x}((k+1)h) = \mathbf{x}(kh) + h\mathbf{f}(\mathbf{x}((k+1)h), \mathbf{u}((k+1)h))$$

Let $\mathbf{G}(\mathbf{x}_{k+1}) = \mathbf{x}_{k+1} - \mathbf{x}_k - h\mathbf{f}(\mathbf{x}_{k+1}, \mathbf{u}_{k+1}) = 0$

Apply Newton's iterative method to solve $\mathbf{x}_{k+1}$

Midpoint Discretization:

$$\mathbf{x}((k+1)h) = \mathbf{x}(kh) + h\mathbf{f}\left(\frac{\mathbf{x}((k+1)h) + \mathbf{x}(kh)}{2}\right)$$

Let $\mathbf{x} = \mathbf{x}_{k+1} = \mathbf{K}(\mathbf{x}) = \mathbf{x}(kh) + h\mathbf{f}\left(\frac{\mathbf{x} + \mathbf{x}(kh)}{2}\right)$

Apply successive approximation to solve $\mathbf{x}$

Stability and Lyapunov's Theorem

For the nonlinear system $\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t))$,

wolg, assume $\mathbf{x_e} = 0$ is an equilibrium solution

- **The equilibrium solution is *stable* if for every choice of a bound M, there is a sufficiently small neighborhood surrounding the equilibrium, $\{\mathbf{x}_i : \|\mathbf{x}_i\| < \delta\}$, such that every initial condition in this region produces a solution bounded by M for all times.**
- **The equilibrium solution is *asymptotically stable* if it is stable, and the solution tends to 0 as $t \to \infty$**
- **The equilibrium solution is *unstable* if it is not stable**

Lyapunov's Theorem:

the asymptotic stability property of $\mathrm{x_e}$ for $\dfrac{d}{dt}\Delta\mathbf{x}(t) = \mathbf{f}'(\mathbf{x_0}(t))\Delta\mathbf{x}(t)$

implies that the same properties hold for the original system

System Behavior and Phase Plane Analysis

- **System Behavior**
  - **Focus on local behavior about the equilibrium points**
  - **Study stability property as well as other nonlinear behavior such as *limit cycles***
- **Phase Plane Analysis**
  - **For two-dimensional (second-order) systems**
  - **Obtain linear system equation by linearizing the nonlinear system at the singular point**
  - **Solution trajectories corresponding to curves, parameterized by time t**
  - **Different types of equilibrium points exhibit different local system behaviors**
  - **Lyapunov theorem indicates that except for the case of a center, the behavior of a system in a sufficiently small neighborhood of a singular point is identical to the behavior of its linearized version**

Example

The simplified Volterra-Lotka population model describe the population of two species predator and prey in an ecosystem is, for example,

$$\dot{x}_1(t) = (x_2(t) - 1)x_1(t)$$
$$\dot{x}_2(t) = (1 - x_1(t))x_2(t)$$

$\Rightarrow$ Equilibrium point: $\mathbf{x}_0 = [1 \;\; 1]'$

$$f'(\mathbf{x}_0) = \begin{bmatrix} \dfrac{\partial f_1}{\partial x_1} & \dfrac{\partial f_1}{\partial x_2} \\ \dfrac{\partial f_2}{\partial x_1} & \dfrac{\partial f_2}{\partial x_2} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

$\Rightarrow \lambda = \pm j$

A center, the stability of the nonlinear system is unknown

Needs to rely on the phase plan analysis

Mid-Term

- In Class (10/16/07)
- Open Books
- 7:20 – 9:50 PM
- 6 – 7 problems
- Evenly spread over all the topics covered
- Homework problems are good examples