DISCRETE EVENT SYSTEMS

MODELING AND PERFORMANCE ANALYSIS

C. G. Cassandras

Division of Systems Engineering
and Dept. of Electrical and Computer Engineering
and Center for Information and Systems Engineering
Boston University

OUTLINE

- Why are DISCRETE EVENT SYSTEMS important?
- Models for Discrete Event Systems (DES) and Hybrid Systems (HS)
- DES Simulation
- Control and Optimization in DES: Resource Contention problems
- “Rapid Learning”:

  Perturbation Analysis (PA) and Concurrent Estimation (CE)
- Applications
- Dealing with Complexity – Fundamental Complexity Limits
- Abstraction through Hybrid Systems:

  Stochastic Flow Models (SFM), the IPA Calculus

WHY DISCRETE EVENT SYSTEMS ?

- Many systems are naturally Discrete Event Systems (DES) (e.g., Internet)
  → *all* state transitions are event-driven
- Most of the rest are Hybrid Systems (HS)
  → *some* state transitions are event-driven
- Many systems are distributed
  → components interact asynchronously (through events)
- Time-driven sampling inherently inefficient ("open loop" sampling)
  → event-driven control

REASONS FOR *EVENT-DRIVEN* MODELS, CONTROL, OPTIMIZATION

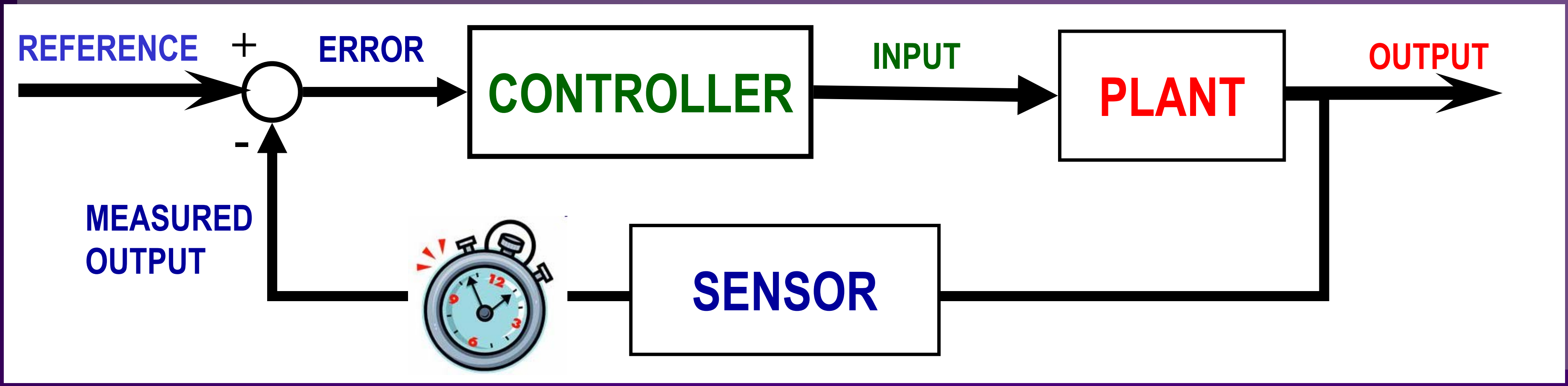
- Many systems are **stochastic**
  $\longrightarrow$ actions needed in response to random events

- Event-driven methods provide significant advantages in **computation** and **estimation** quality

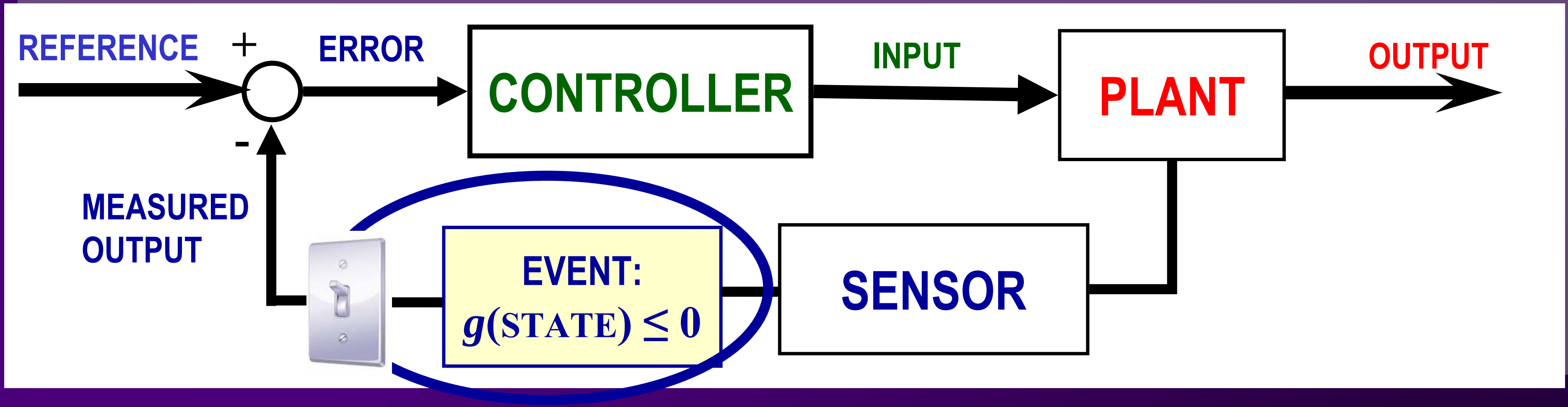
- System performance is often **more sensitive to event-driven** components than to time-driven components

- Many systems are **wirelessly networked** $\longrightarrow$ energy constrained
  $\longrightarrow$ time-driven communication consumes energy unnecessarily
  $\longrightarrow$ use **event-driven control**

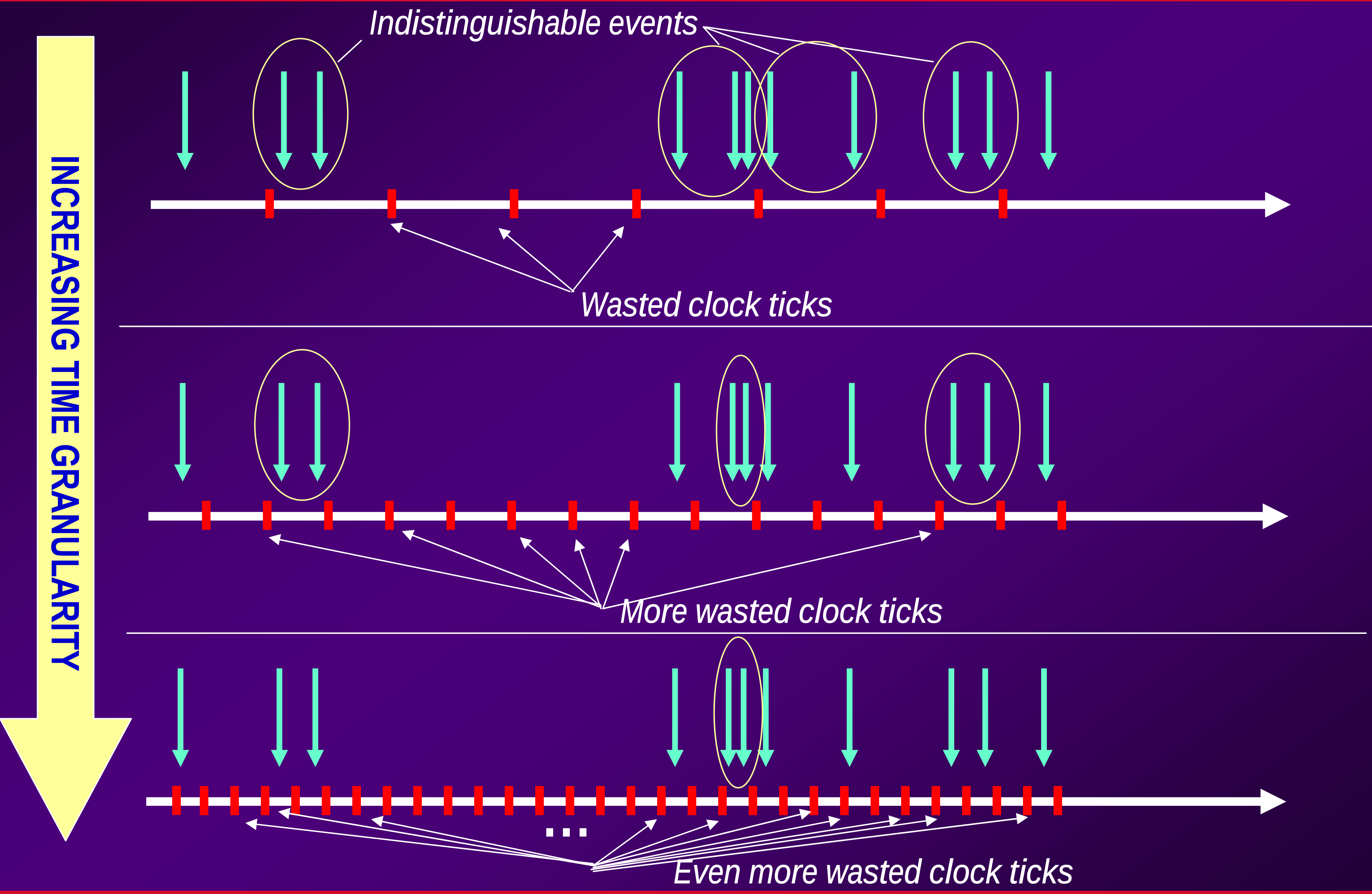
TIME-DRIVEN v EVENT-DRIVEN CONTROL



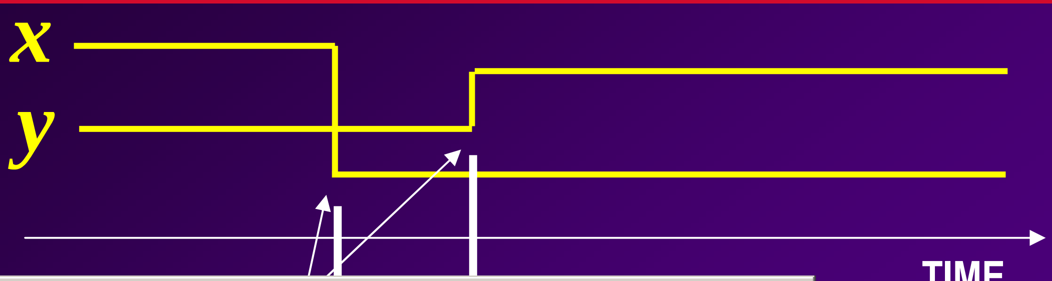
EVENT-DRIVEN CONTROL: Act *only when needed* (or on **TIMEOUT**) - not based on a clock

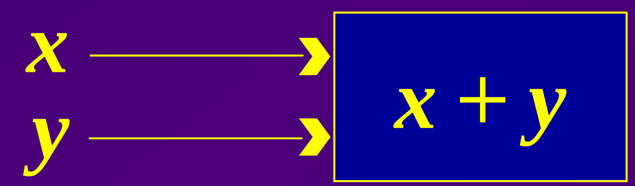


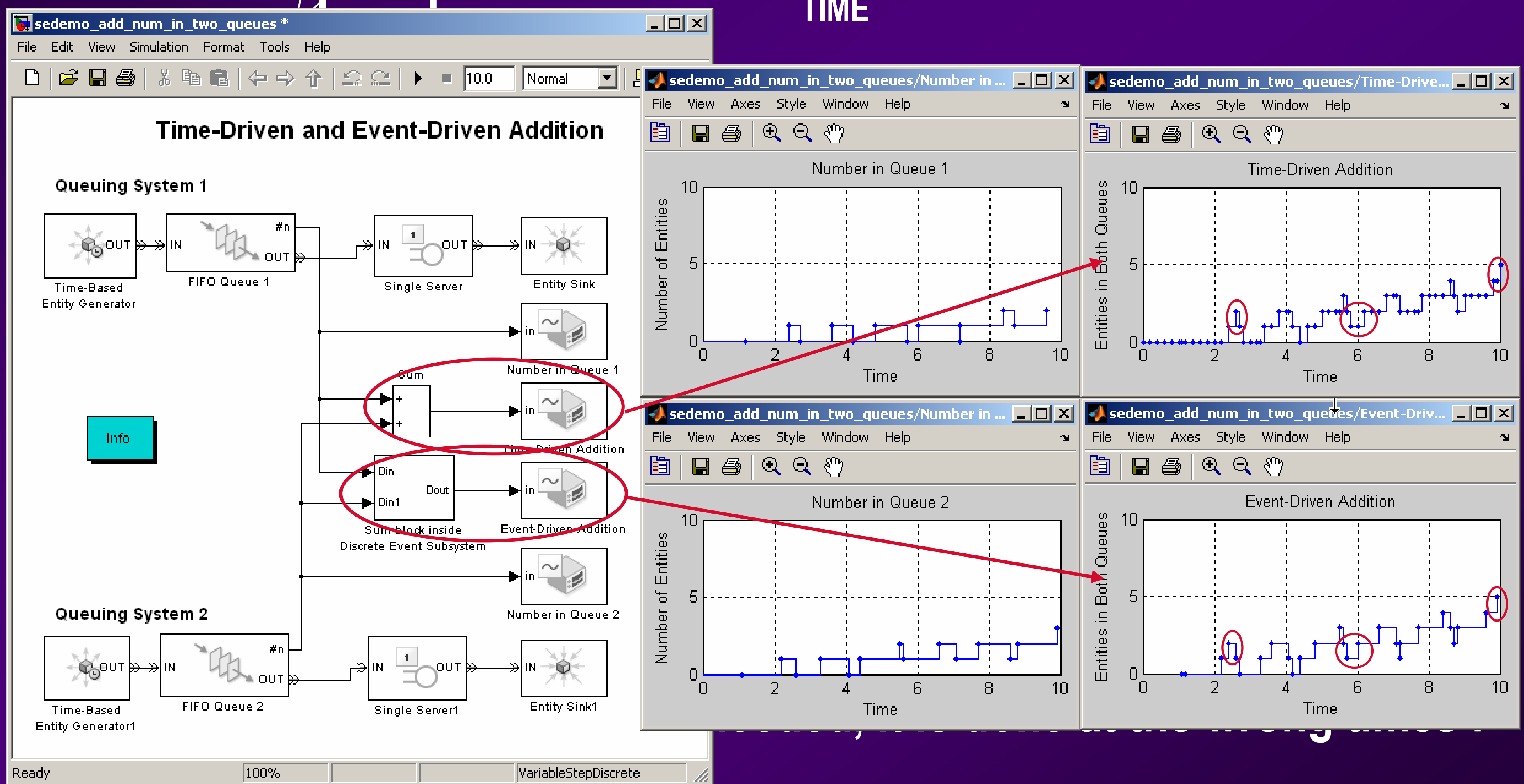
SYNCHRONOUS v ASYNCHRONOUS BEHAVIOR



SYNCHRONOUS v ASYNCHRONOUS COMPUTATION







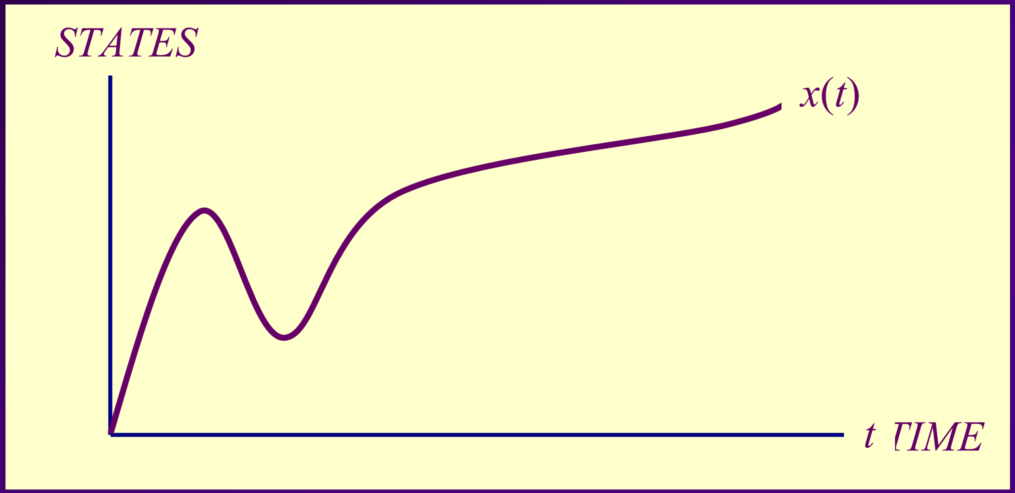
MODELING DES AND HS:
-Timed Automata
- Hybrid Automata

OTHER MODELS NOT COVERED HERE:

- Timed Petri Nets
- Max-Plus Algebra
- Temporal Logic

TIME-DRIVEN v EVENT-DRIVEN SYSTEMS

TIME-DRIVEN SYSTEM



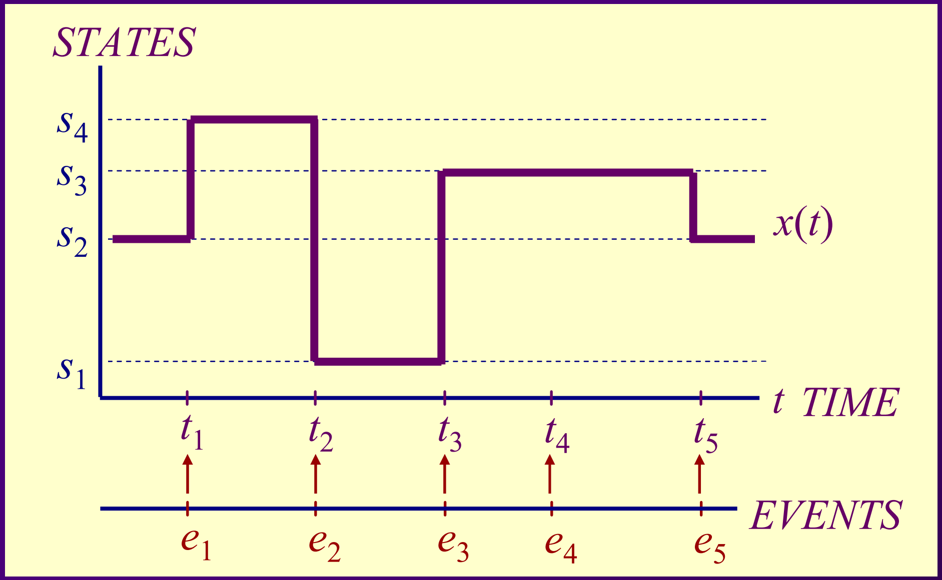
STATE SPACE:

$$X = \Re$$

DYNAMICS:

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

EVENT-DRIVEN SYSTEM



STATE SPACE:

$$X = \{s_1, s_2, s_3, s_4\}$$

DYNAMICS:

$$x' = f(x, e)$$

AUTOMATON

AUTOMATON: (E, X, Γ, f, x_0)

E : Event Set

X : State Space

$\Gamma(x)$: Set of *feasible* or *enabled* events at state x

f : State Transition Function $f : X \times E \to X$
(undefined for events $e \notin \Gamma(x)$)

x_0 : Initial State, $x_0 \in X$

$$\{e_1, e_2, \ldots\} \longrightarrow \boxed{f(x,e) = x'} \longrightarrow \{x_1, x_2, \ldots\}$$

TIMED AUTOMATON

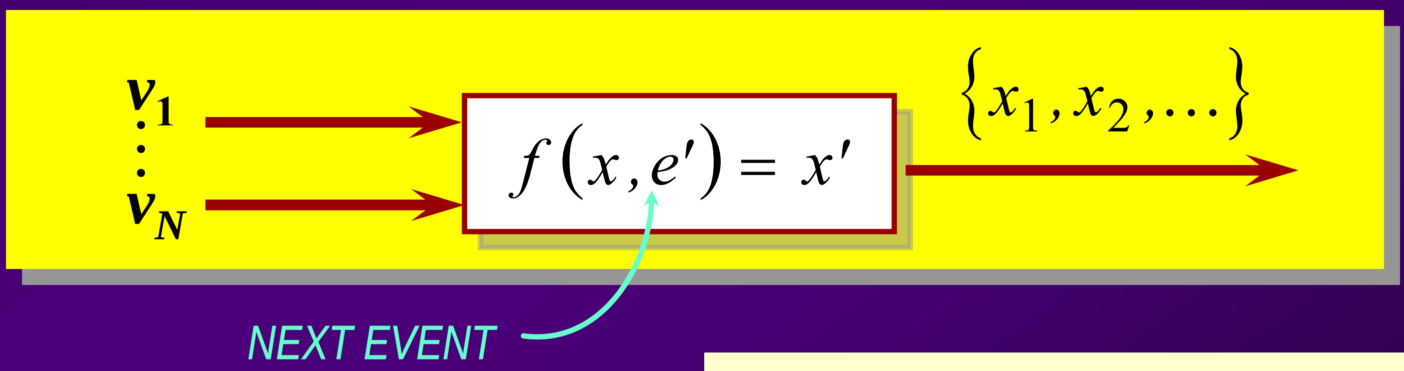
Add a ***Clock Structure V*** to the automaton: $(E, X, \Gamma, f, x_0, \boldsymbol{V})$

where:

$$\boldsymbol{V} = \{\boldsymbol{v}_i : i \in E\}$$

and $\boldsymbol{v}_i$ is a ***Clock or Lifetime sequence***: $\boldsymbol{v}_i = \{v_{i1}, v_{i2}, \ldots\}$

one for each event i



Need an *internal mechanism* to determine *NEXT EVENT e′* and hence *NEXT STATE* $x' = f(x, e')$

HOW THE TIMED AUTOMATON WORKS...

- **CURRENT STATE**

  $x \in X$ with feasible event set $\Gamma(x)$

- **CURRENT EVENT**

  e that caused transition into x

- **CURRENT EVENT TIME**

  t associated with e

Associate a
CLOCK VALUE/RESIDUAL LIFETIME y_i
with each feasible event $i \in \Gamma(x)$

HOW THE TIMED AUTOMATON WORKS...

- **NEXT/TRIGGERING EVENT** e' **:**

$$e' = \arg \min_{i \in \Gamma(x)} \{y_i\}$$

- **NEXT EVENT TIME** t' **:**

$$t' = t + y^*$$

$$\text{where:} \quad y^* = \min_{i \in \Gamma(x)} \{y_i\}$$

- **NEXT STATE** x' **:**

$$x' = f(x, e')$$

HOW THE TIMED AUTOMATON WORKS...

Determine new ***CLOCK VALUES*** y_i'
for every event $i \in \Gamma(x)$

$$y_i' = \begin{cases} y_i - y^* & i \in \Gamma(x'), i \in \Gamma(x), i \neq e' \\ v_{ij} & i \in \Gamma(x') - \{\Gamma(x) - e'\} \\ 0 & \textit{otherwise} \end{cases}$$

where: v_{ij} = new lifetime for event i

EVENT CLOCKS
ARE STATE VARIABLES

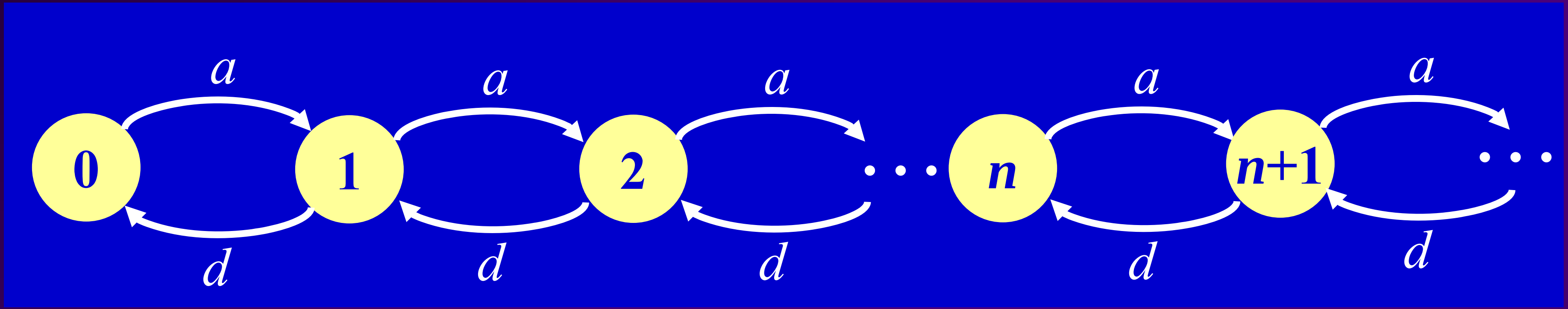
$\mathbf{v}_1$ ⋮ $\mathbf{v}_N$

$$x' = f(x, e'), \quad e' = \arg \min_{i \in \Gamma(x)} \{y_i\}$$

$$\mathbf{y}' = \mathbf{g}(\mathbf{y}, x, V)$$

$\{x_1, x_2, \ldots\}$

TIMED AUTOMATON – AN EXAMPLE



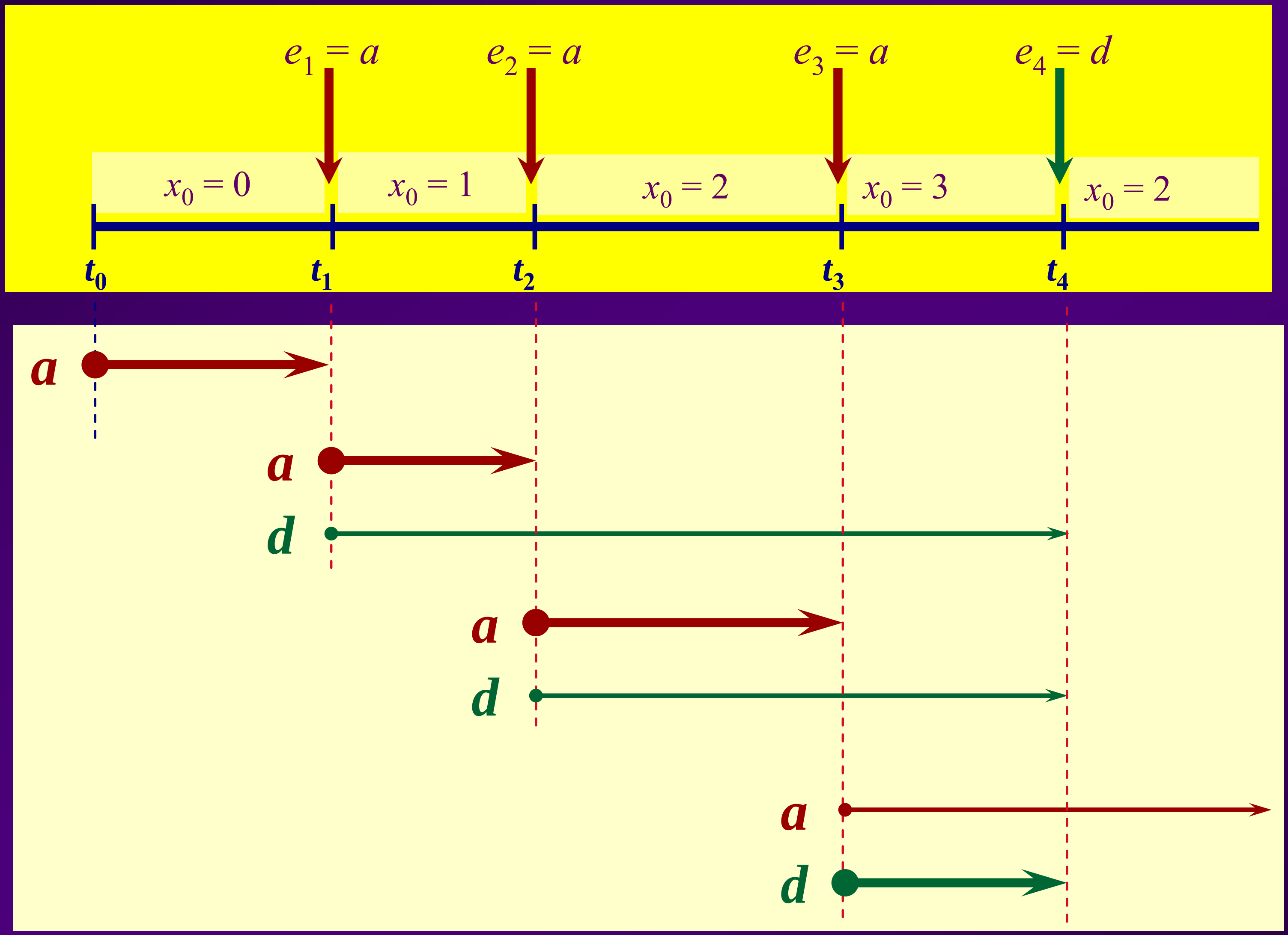
$$E = \{a, d\}$$
$$X = \{0, 1, 2, \ldots\}$$

$$\Gamma(x) = \{a, d\}, \quad \text{for all } x > 0$$
$$\Gamma(0) = \{a\}$$

$$f(x, e') = \begin{cases} x+1 & e' = a \\ x-1 & e' = d, \; x > 0 \end{cases}$$

$$\text{Given input}: \quad \mathbf{v}_a = \{v_{a1}, v_{a2}, \ldots\}, \; \mathbf{v}_d = \{v_{d1}, v_{d2}, \ldots\}$$

TIMED AUTOMATON – A STATE TRAJECTORY



STOCHASTIC TIMED AUTOMATON

- Same idea with the Clock Structure consisting of ***Stochastic Processes***

- Associate with each event *i a* ***Lifetime Distribution*** based on which $\boldsymbol{v}_i$ is generated

↓

Generalized Semi-Markov Process
(GSMP)

In a simulator, $\boldsymbol{v}_i$ is generated through a pseudorandom number generator

HYBRID AUTOMATA

$$G_h = (Q, X, E, U, f, \phi, Inv, guard, \rho, q_0, \mathbf{x}_0)$$

Q: set of discrete states (modes)

X: set of continuous states (normally R^n)

E: set of events

U: set of admissible controls

f: vector field, $f : Q \times X \times U \to X$

ϕ: discrete state transition function, $\phi : Q \times X \times E \to Q$

Inv: set defining an invariant condition (domain), $Inv \subseteq Q \times X$

$guard$: set defining a guard condition, $guard \subseteq Q \times Q \times X$

ρ: reset function, $\rho : Q \times Q \times X \times E \to X$

q_0: initial discrete state

$\mathbf{x}_0$: initial continuous state

Key features:

Transition MAY occur

Guard condition: Subset of X in which a transition from q to q' is enabled, defined through ϕ

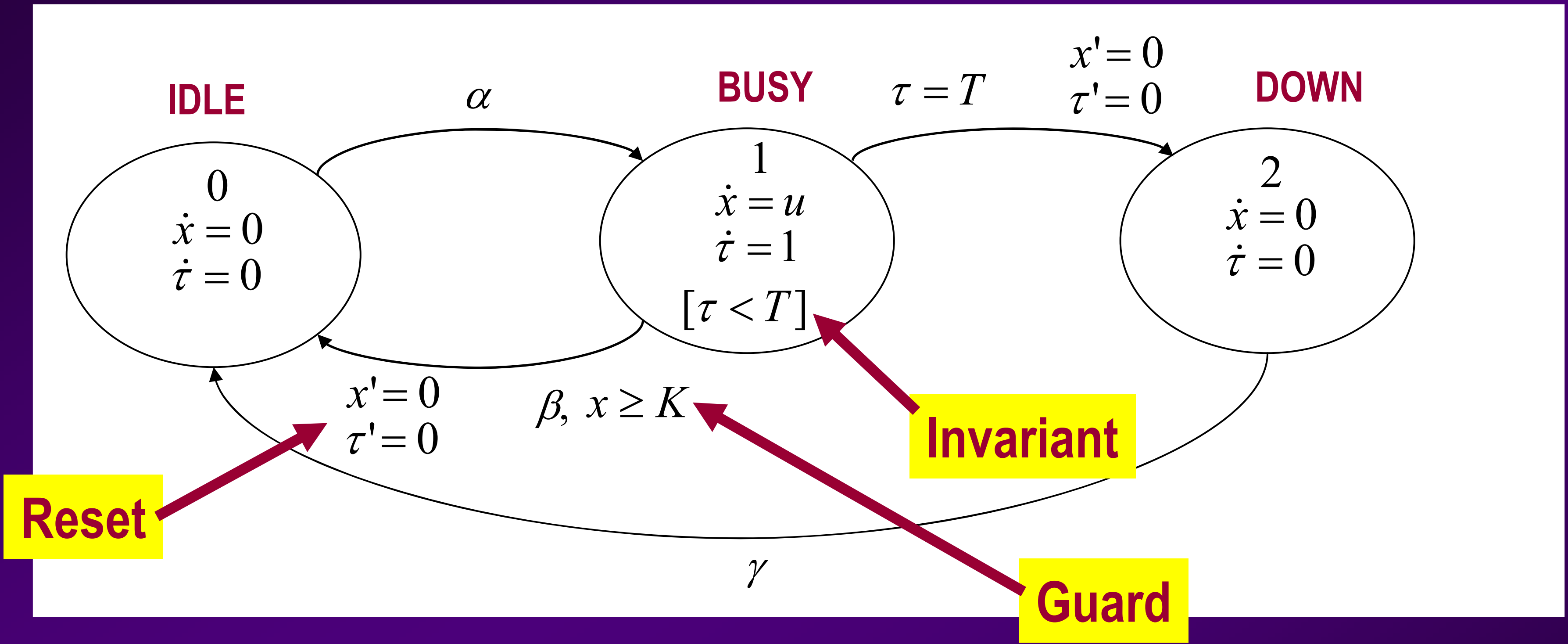
Transition MUST occur

Invariant condition: (domain) Subset of X to which x must belong in order to remain in q. If this condition no longer holds, a transition to some q' must occur, defined through ϕ

Reset condition: New value x' at q' when transition occurs from (x,q)

HYBRID AUTOMATA

Unreliable machine with timeouts

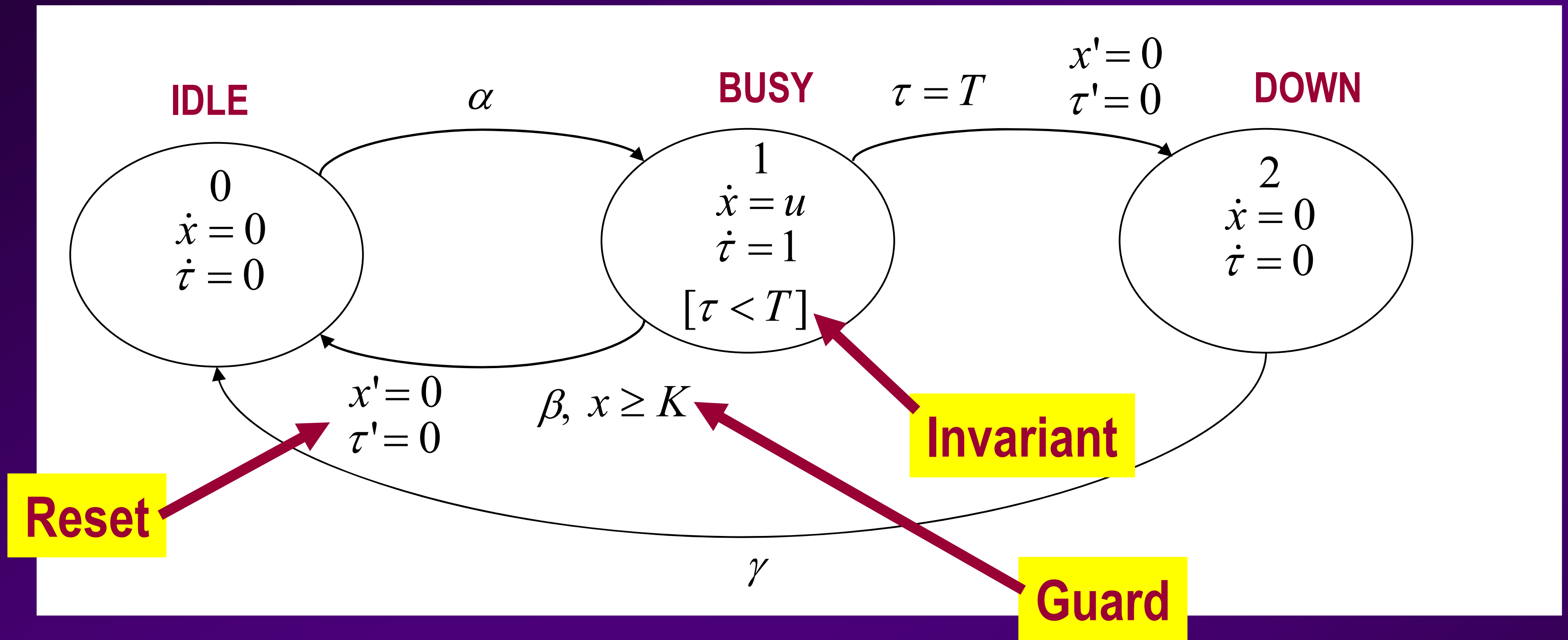


$x(t)$: physical state of part in machine

$\tau(t)$: clock

α : START, β : STOP, γ : REPAIR

HYBRID AUTOMATA



$$\phi(0; x, \tau; e) = \begin{cases} 1 & e = \alpha \\ 0 & \text{otherwise} \end{cases}$$

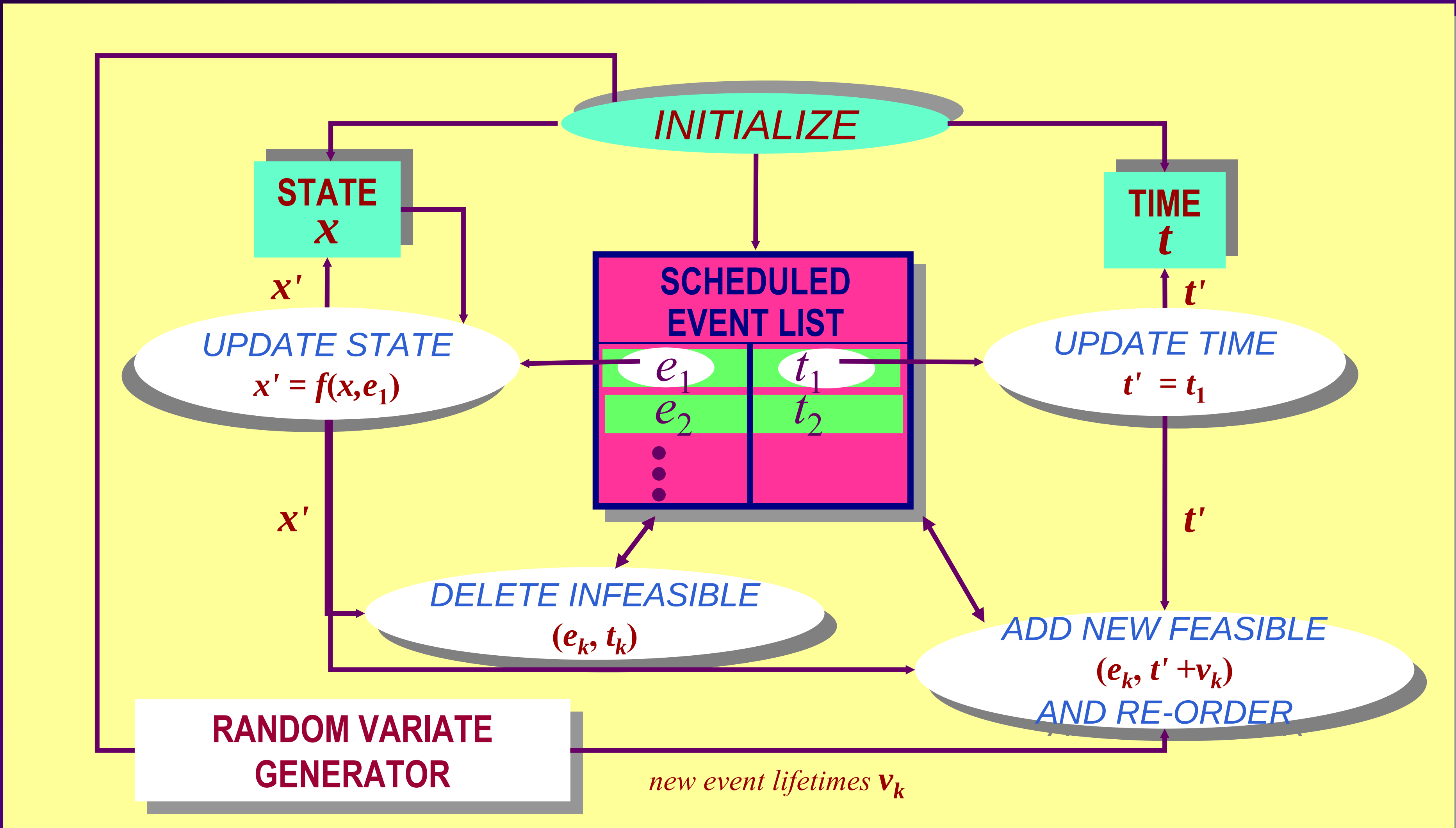
$$\phi(1; x, \tau; e) = \begin{cases} 2 & \tau = T \\ 0 & x \geq K,\ e = \beta \\ 1 & \text{otherwise} \end{cases}$$

$$\phi(2; x, \tau; e) = \begin{cases} 0 & e = \gamma \\ 2 & \text{otherwise} \end{cases}$$

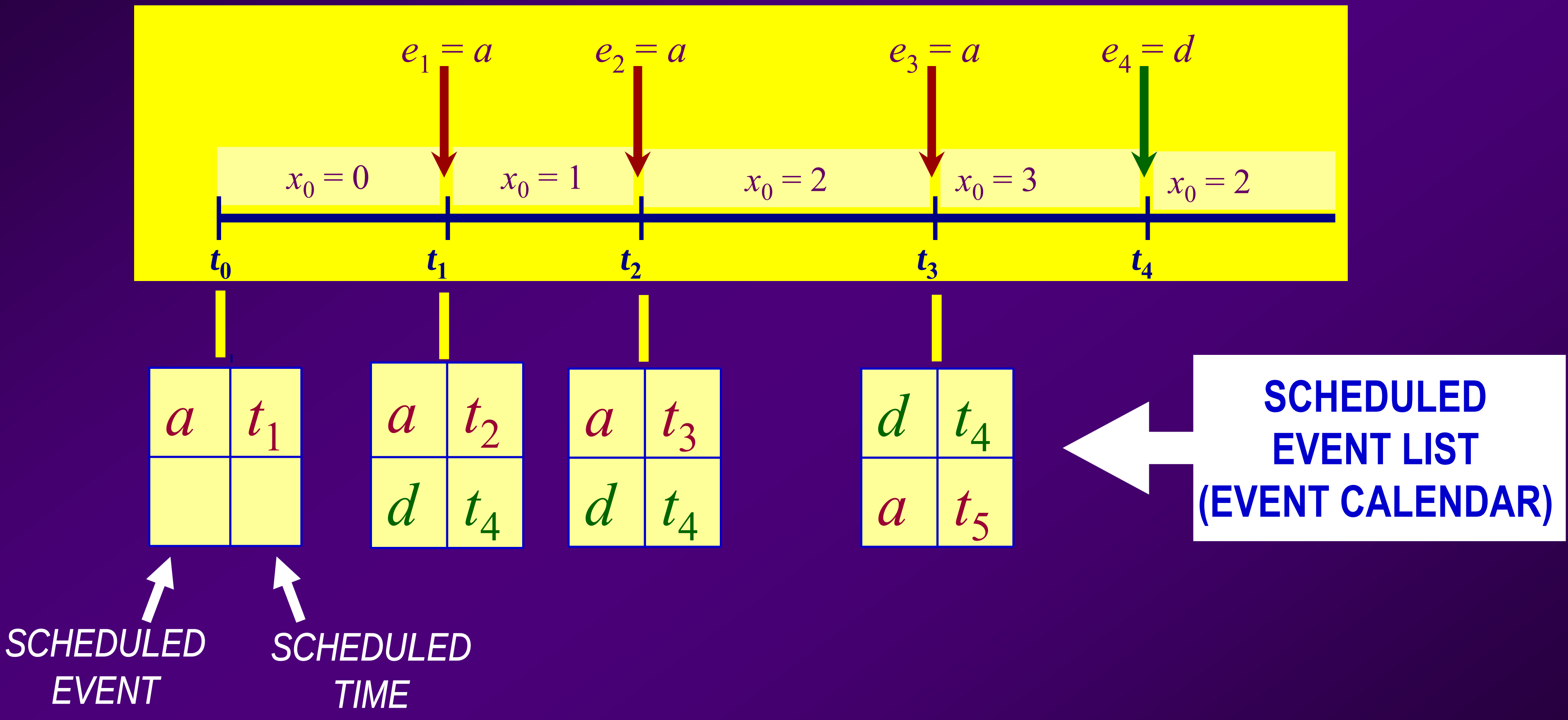
DES SIMULATION

DISCRETE EVENT SIMULATION

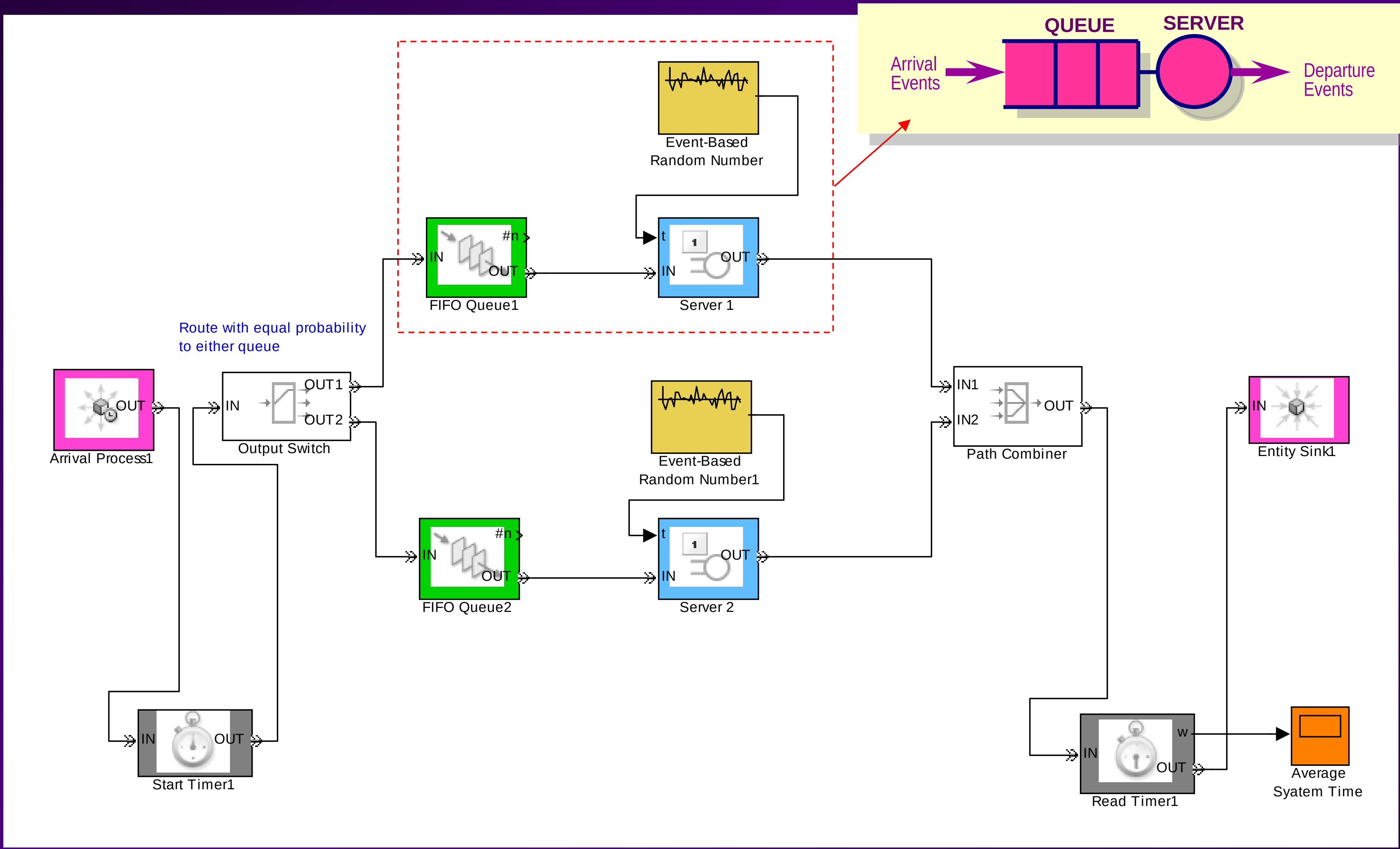
…is simply a computer-based implementation of the DES sample path generation mechanism described so far



DISCRETE EVENT SIMULATION - EXAMPLE

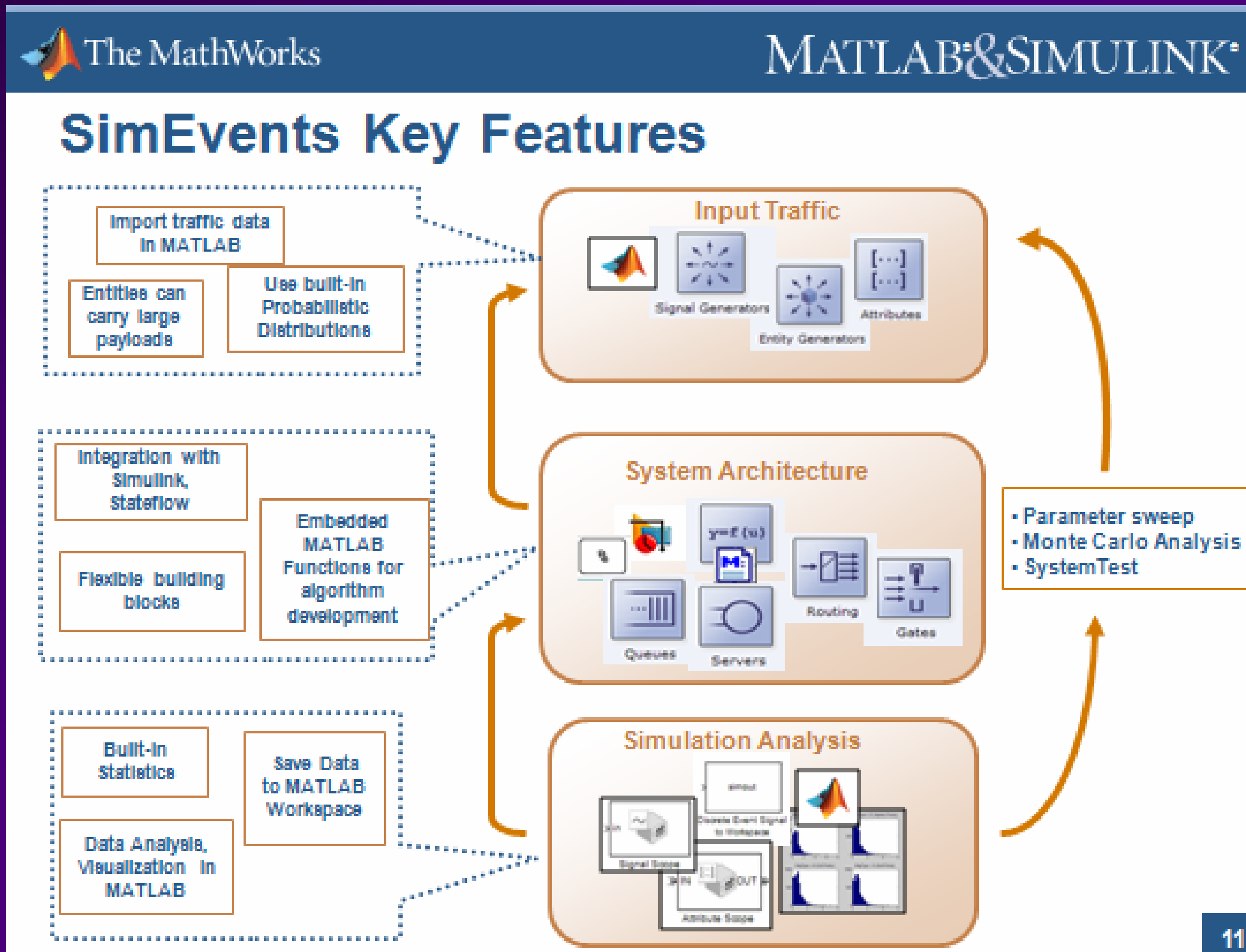


DISCRETE EVENT SIMULATION - EXAMPLE



DISCRETE EVENT SIMULATION - *SimEvents*

www.mathworks.com/products/simevents/



[Cassandras, Clune, and Mosterman, 2006]

SOFTWARE IMPLEMENTATION ISSUES

- Object-oriented design (Libraries)
- Flexibility (user can easily define new objects)
- Hierarchical (macro) capability [e.g., a *G/G/m/n* block]
- Random Variate generation accuracy
- Execution speed
- User interface
- Animation (useful sometimes – distracting/slow other times)
- Integrating with operational control software
  (e.g., *scheduling, resource allocation, flow control*)
- Web-based simulation: *Google-like capability* !?

SELECTED REFERENCES - MODELING

Timed Automata, Timed Petri Nets, Max-Plus Algebra

– Alur, R., and D.L. Dill, “A Theory of Timed Automata,” Theoretical Computer Science, No. 126, pp. 183-235, 1994.
– Cassandras, C.G, and S. Lafortune, “Introduction to Discrete Event Systems,” Springer, 2008.
– Wang, J., “Timed Petri Nets - Theory and Application,” Kluwer Academic Publishers, Boston, 1998.
– Heidergott, B., G.J. Olsder, and J. van der Woude, “Max Plus at Work – Modeling and Analysis of Synchronized Systems: A Course on Max-Plus Algebra and its Applications,” Princeton University Press, 2006

Hybrid Systems

– Bemporad, A. and M. Morari, “Control of Systems Integrating Logic Dynamics and Constraints,” Automatica, Vol. 35, No. 3, pp.407-427, 1999.
– Branicky, M.S., V.S. Borkar, and S.K. Mitter, “A Unified Framework for Hybrid Control: Model and Optimal Control Theory,”IEEE Trans. on Automatic Control, Vol. 43, No. 1, pp. 31-45, 1998.
– Cassandras, C.G., and J. Lygeros, “Stochastic Hybrid Systems,” Taylor and Francis, 2007.
– Hristu-Varsakelis, D. and W.S. Levine, Handbook of Networked and Embedded Control Systems, Birkhauser, Boston, 2005.

CONTROL AND OPTIMIZATION IN DES

DES CONTROL

ENABLE/DISABLE controllable events in order to achieve desired LOGICAL BEHAVIOR

- Reach desirable states
- Avoid undesirable states
- Prevent system deadlock

SUPERVISORY CONTROL

ENABLE/DISABLE controllable events in order to achieve desired PERFORMANCE

- N events of type 1 within T time units
- nth type 1 event occurs before mth type 2 event
- No more than N type 1 events after T time units

RESOURCE CONTENTION PROBLEMS IN DES

RESOURCE CONTENTION

USERS competing for limited *RESOURCES* in an event-driven dynamic environment

Examples:

- *MESSAGES* competing for *SWITCHES* in networks
- *TASKS* competing for *PROCESSORS* in computers
- *PARTS* competing for *EQUIPMENT* in manufacturing

TYPICAL GOALS:

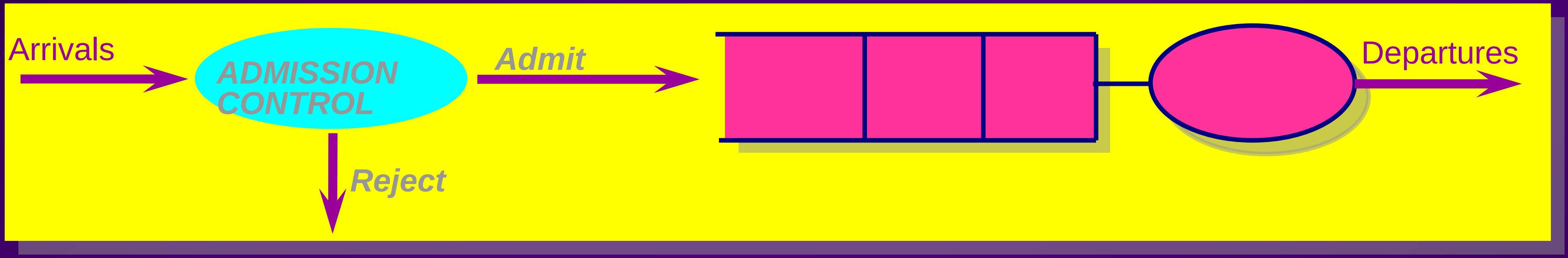
- User requests satisfied on “best effort” basis
- User requests satisfied on “guaranteed performance” basis
- Users treated “fairly”

RESOURCE CONTENTION

1. ADMISSION CONTROL

Admit or Reject user requests

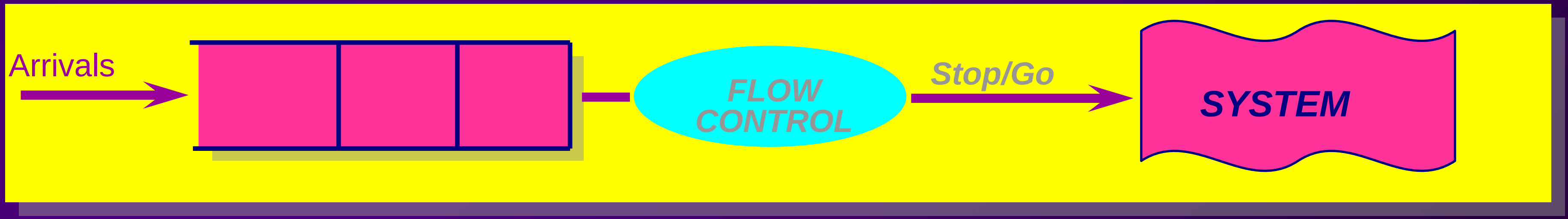
(e.g., to ensure good quality of service for admitted users)



2. FLOW CONTROL

Control *when* to accept user requests

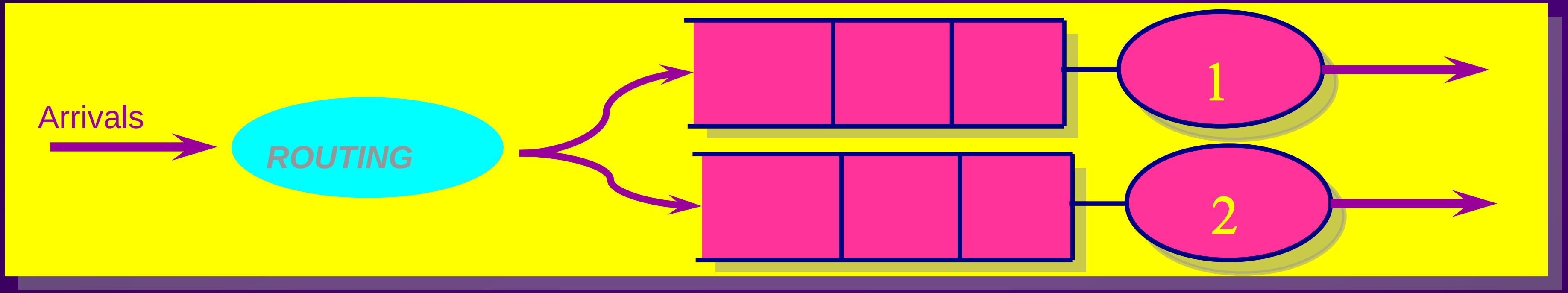
(e.g., to "smooth" bursty demand, prevent congestion)



RESOURCE CONTENTION

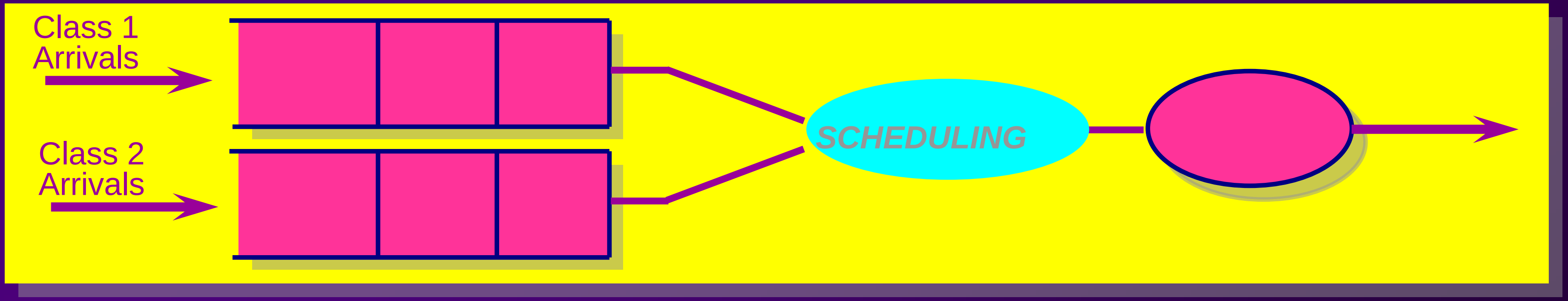
3. ROUTING

User selects a resource
(e.g., route to shortest queue)



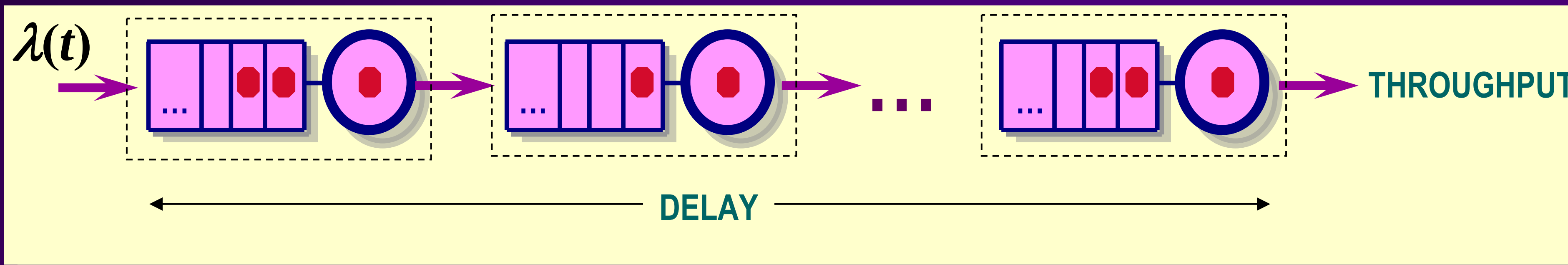
4. SCHEDULING

Resource selects user
(e.g., serve longest queue first)



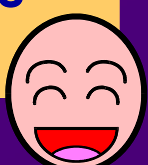
RESOURCE ALLOCATION

Manufacturing system with *N* sequential operations:

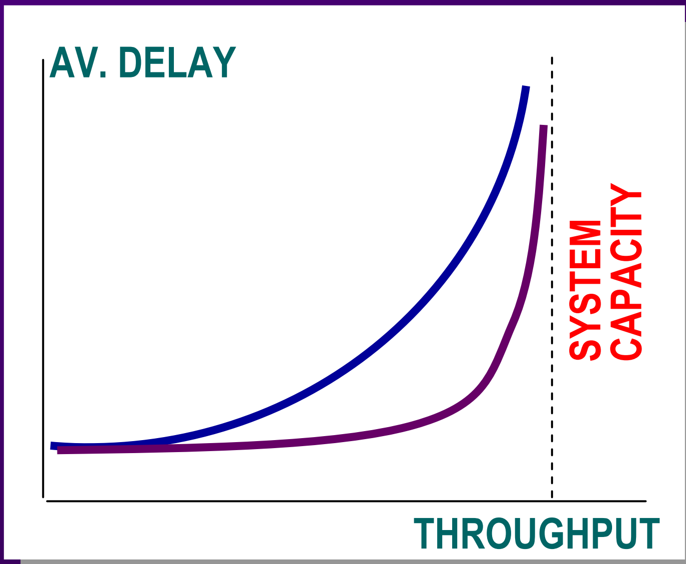


INCREASE $\lambda(t)$

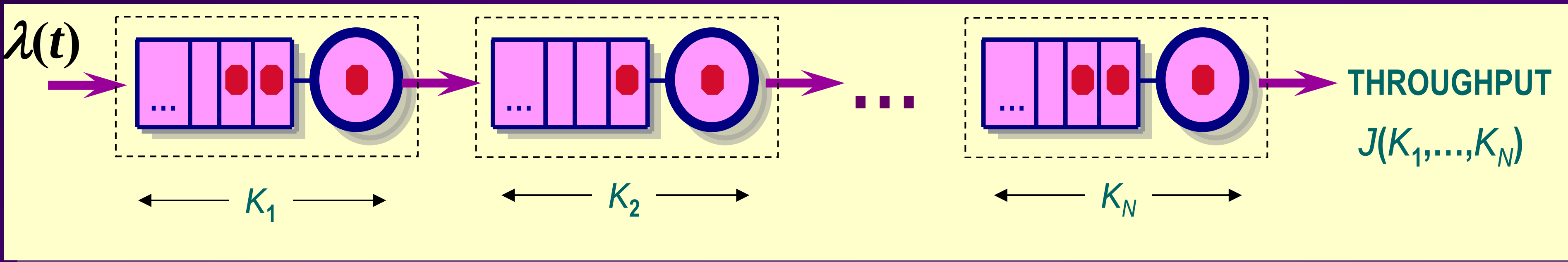
THROUGHPUT increases (GOOD)



average DELAY increases (BAD)

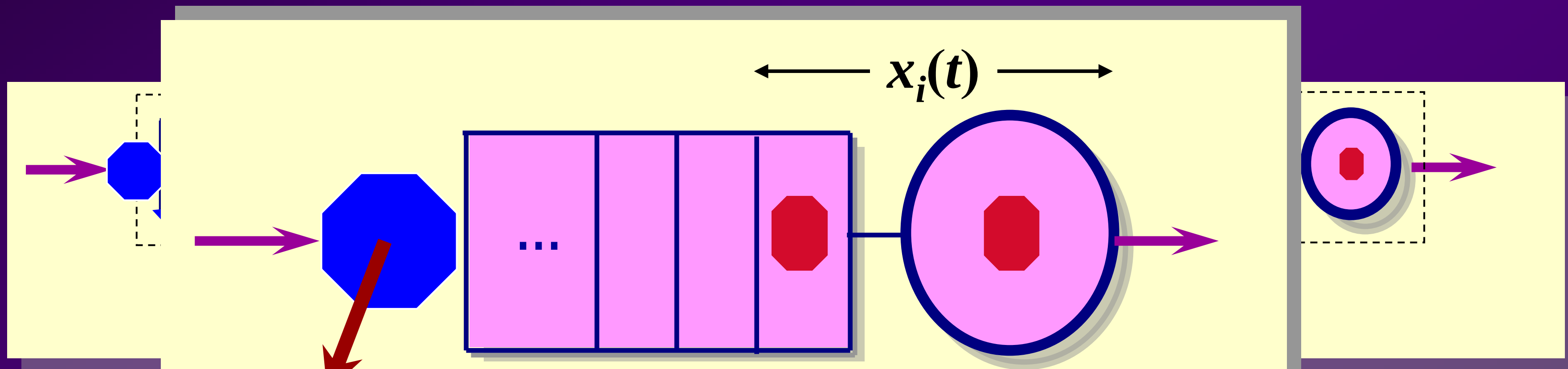


KANBAN (OR *BUFFER*) *ALLOCATION*



$$\max_{K_1,\dots,K_N} J(K_1,\dots,K_N) \text{ s.t.} \sum_{i=1}^{N} K_i = C$$

RESOURCE ALLOCATION

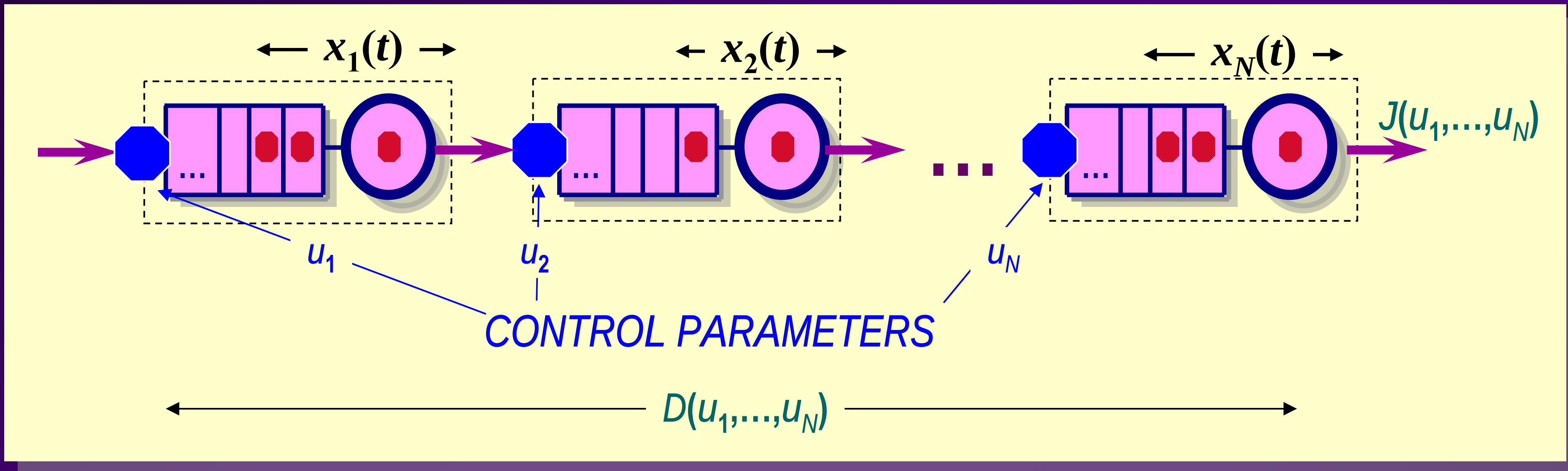


GO **if** $x_i(t) < K_i$, ***STOP*** **otherwise**

FEEDBACK

No. of *KANBAN (tickets)* allocated to stage *i*

RESOURCE ALLOCATION



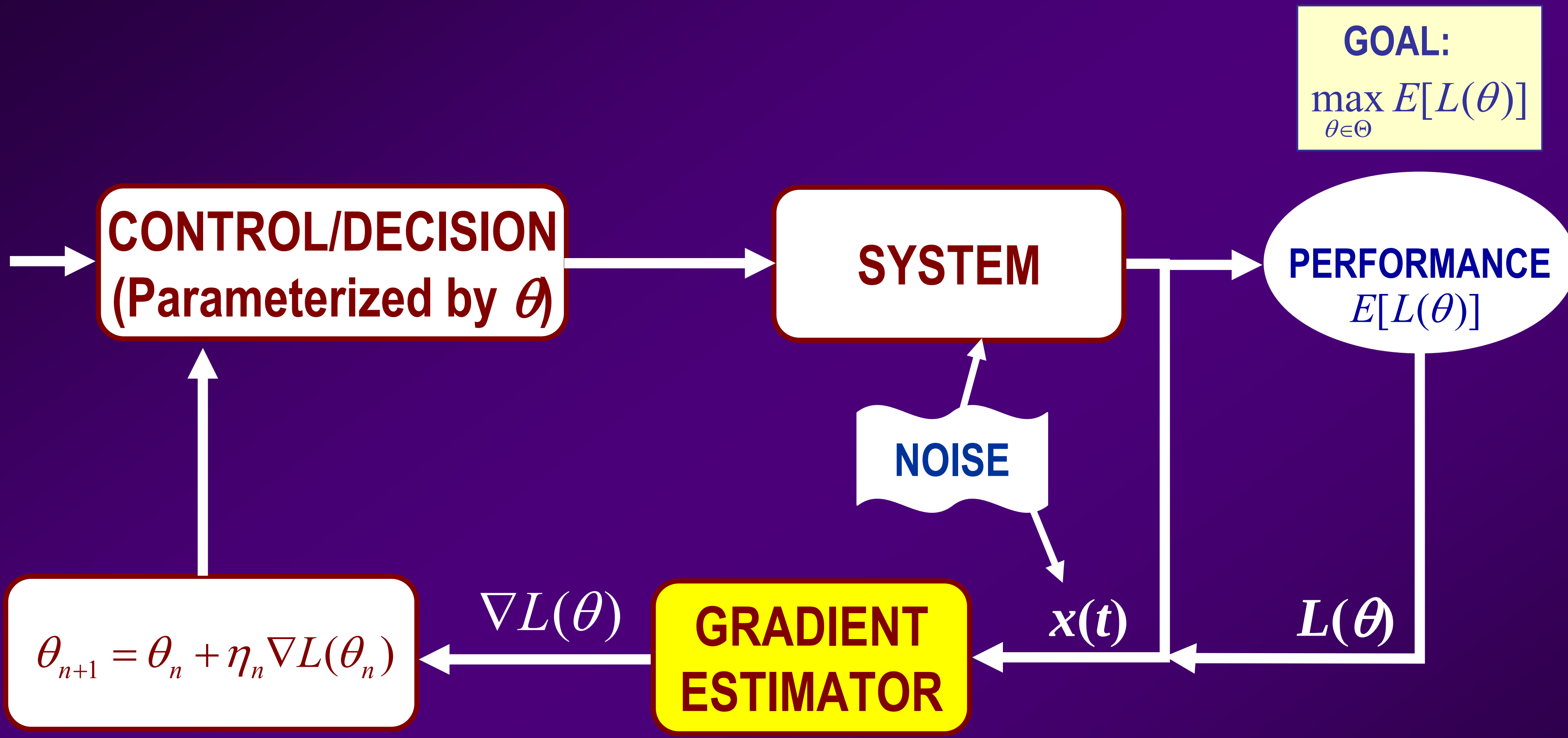
$$\max_{u_1,\ldots,u_N} J(u_1,\ldots,u_N) \;\; \text{s.t.} \begin{cases} D(u_1,\ldots,u_N) \le C \\ \text{system dynamics} \end{cases}$$

SOLUTION METHODOLOGIES

- **Queueing** models and analysis: *Descriptive*, not *prescriptive*
- **Markov Decision Processes** (MDPs):
  - Decisions planned ahead
  - Need accurate stochastic models
  - Curse of dimensionality (Dynamic Programming)
- **Approximate Dynamic Programming** (ADP) techniques
- **Data-driven** techniques:
  - Gradient Estimation
  - Rapid Learning

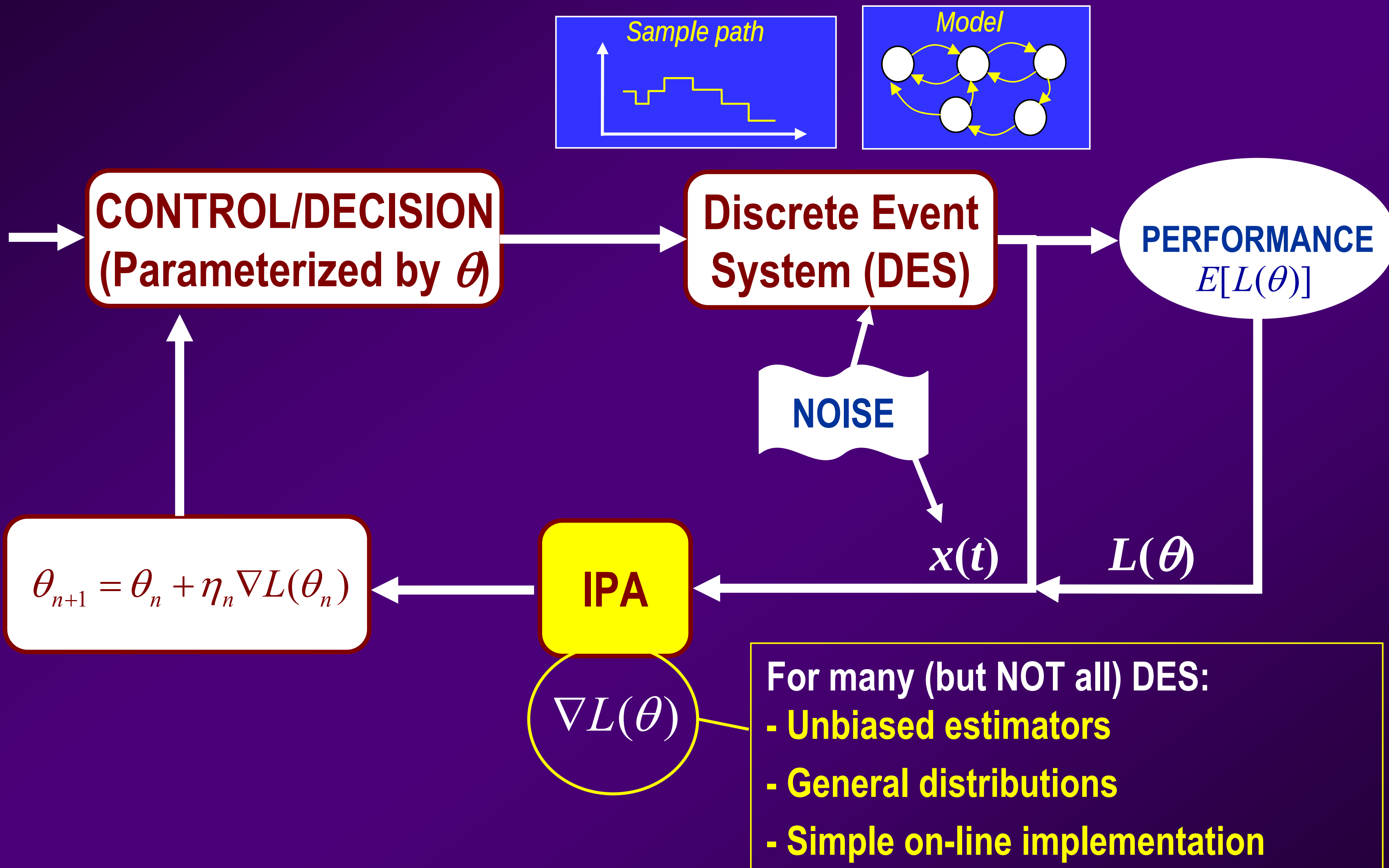
STOCHASTIC CONTROL AND OPTIMIZATION: PERTURBATION ANALYSIS, RAPID LEARNING

REAL-TIME STOCHASTIC CONTROL AND OPTIMIZATION



DIFFICULTIES:
- $E[L(\theta)]$ NOT available in closed form
- $\nabla L(\theta)$ not easy to evaluate
- $\nabla L(\theta)$ may not be a good estimate of $\nabla E[L(\theta)]$

THE CASE OF *DES*: INFINITESIMAL PERTURBATION ANALYSIS (IPA)



[*Ho and Cao, 1991*], [*Glasserman, 1991*], [*Cassandras, 1993*], [*Cassandras and Lafortune, 2008*]

RAPID LEARNING

LEARNING BY TRIAL-AND-ERROR

CONVENTIONAL TRIAL-AND-ERROR ANALYSIS (e.g., simulation)

- *Repeatedly change parameters/operating policies*
- *Test different conditions*
- *Answer multiple WHAT IF questions*

LEARNING THROUGH *CONCURRENT ESTIMATION*

Design Parameters

Operating Policies, Control Parameters

SYSTEM

Performance Measures

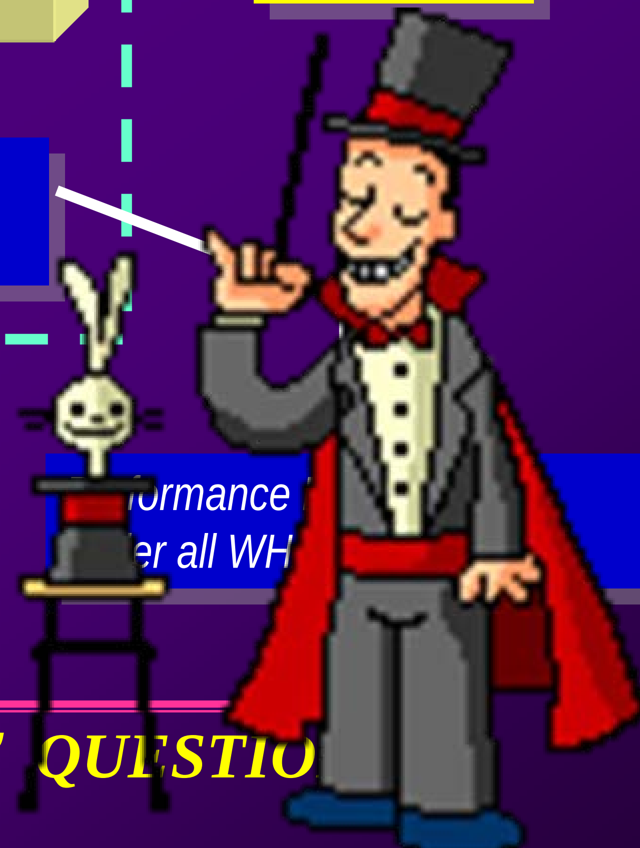
CONCURRENT ESTIMATION

WHAT IF…

- Parameter $p_1 = a$ were replaced by $p_1 = b$
- Parameter $p_2 = c$ were replaced by $p_2 = d$

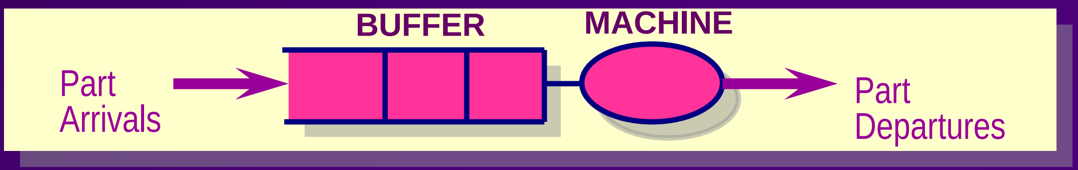
  ⋮

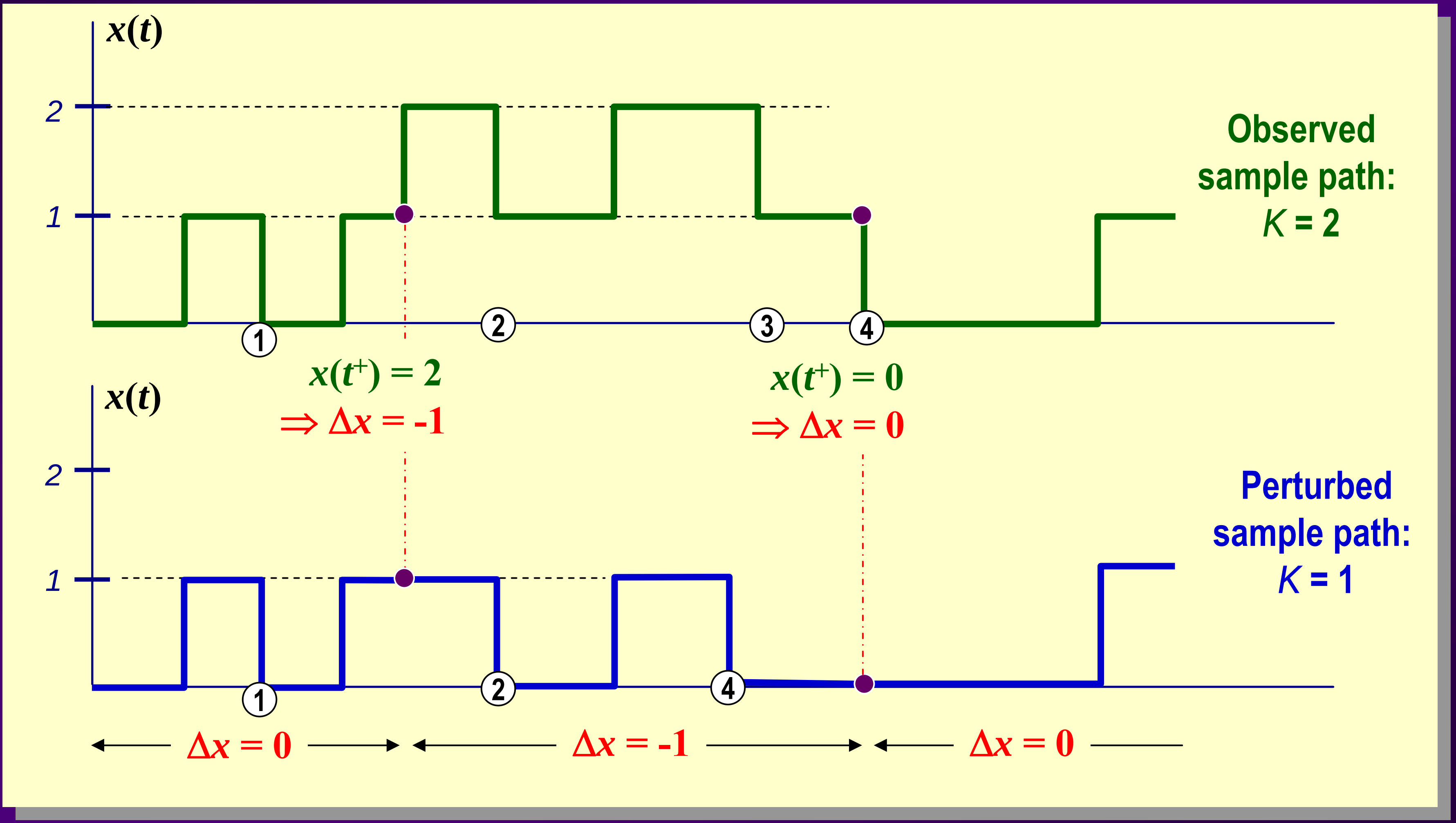
ormance
er all WH



ANSWERS TO MULTIPLE "WHAT IF" QUESTIO
AUTOMATICALLY PROVIDED

LEARNING THROUGH *CONCURRENT ESTIMATION*

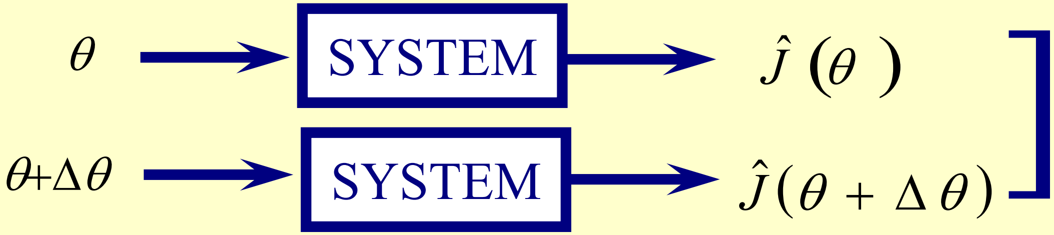




PERTURBATION ANALYSIS

IPA vs "BRUTE FORCE" DERIVATIVE ESTIMATION

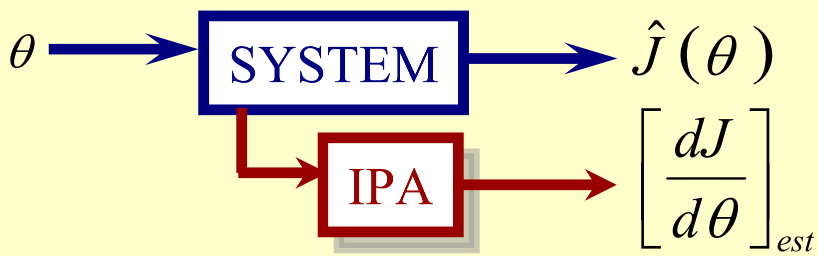
"Brute Force" Derivative Estimation:



$$\Rightarrow \left[\frac{dJ}{d\theta}\right]_{est} = \frac{\hat{J}(\theta+\Delta\theta)-\hat{J}(\theta)}{\Delta\theta}$$

DRAWBACKS:
- **Intrusive**: actively introduces perturbation $\Delta\theta$
- **Computationally costly**: 2 observation processes [$(N+1)$ for N-dim θ]
- **Inherently inaccurate**: $\Delta\theta$ large $\Rightarrow$ poor derivative approx.
  $\Delta\theta$ small $\Rightarrow$ numerical instability

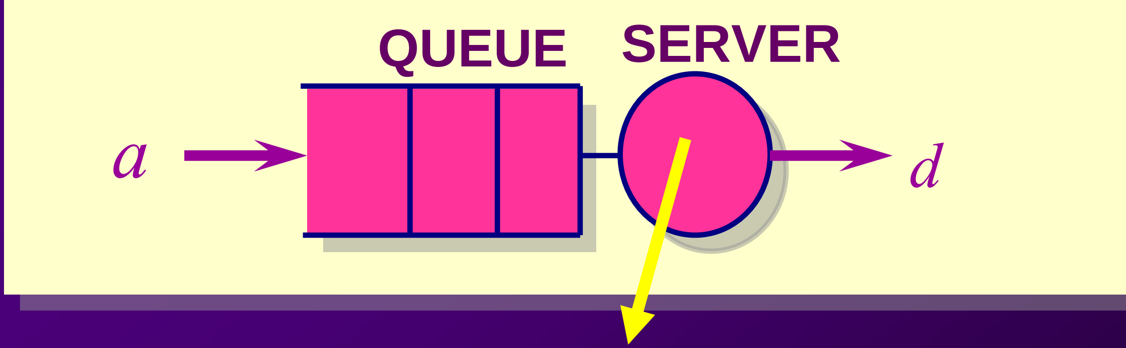
Infinitesimal Perturbation Analysis (IPA):



PERTURBATION GENERATION

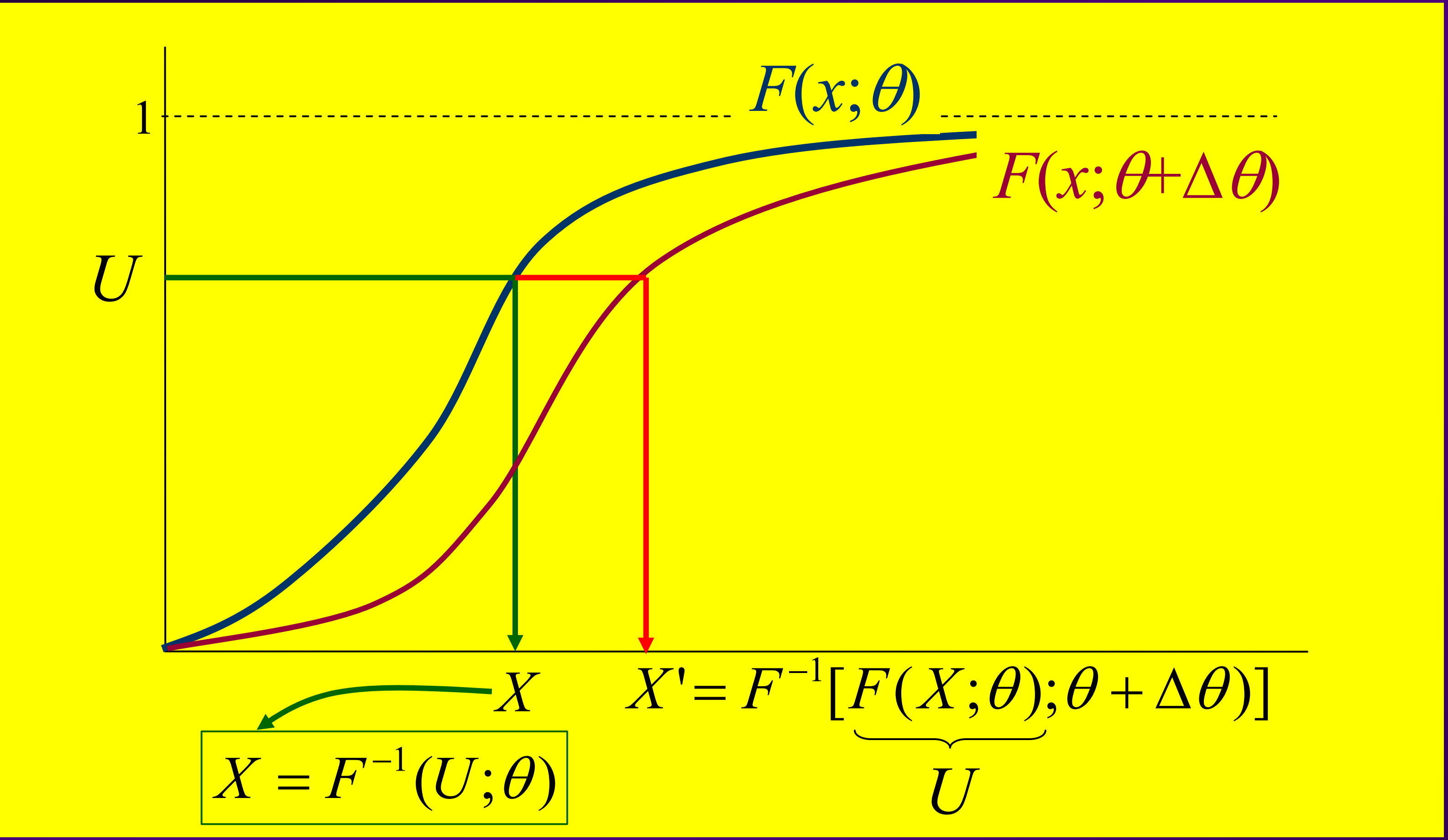
- DES parameter θ perturbed by $\Delta\theta$
- Suppose θ is a parameter of some cdf $F(x;\theta)$
- OBSERVED sample path: $X(\theta)$
- What is $\Delta X(\theta)$?
- What is $\dfrac{dX(\theta)}{d\theta}$?

EXAMPLE:



- Mean Service Time: θ
- Service Times: $X_1(\theta), X_2(\theta), \ldots$
- What happens when θ is changed by $\Delta\theta$?

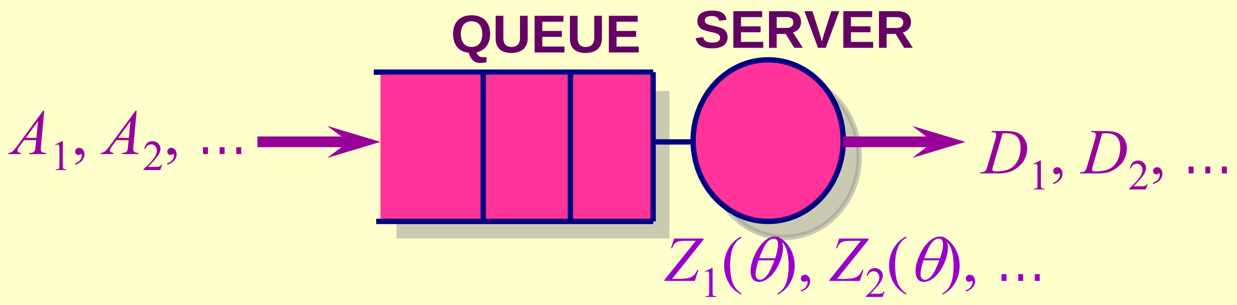
PERTURBATION GENERATION



$$\Delta X = F^{-1}[F(X;\theta);\theta+\Delta\theta)] - X$$

$$\frac{dX}{d\theta} = -\frac{\left[\partial F(x;\theta)/\partial\theta\right]_{(X,\theta)}}{\left[\partial F(x;\theta)/\partial x\right]_{(X,\theta)}}$$

PERTURBATION PROPAGATION



Suppose θ is perturbed by $\Delta\theta \;\Rightarrow\; \Delta D_k = D_k' - D_k$

Lindley Equation for any queueing system:

$$D_k = \max\{A_k, D_{k-1}\} + Z_k$$

Max-Plus Algebra
(One of many dioid algebras)
Cunningham-Greene, R.A., Minimax Algebra, 1979.

$$\Rightarrow \quad \Delta D_k = \max\{A_k, D_{k-1}'\} - \max\{A_k, D_{k-1}\} + \Delta Z_k$$
$$= \max\{A_k, D_{k-1} + \Delta D_{k-1}\} - \max\{A_k, D_{k-1}\} + \Delta Z_k$$

Iterative relationship that depends ONLY on observed sample path data !

PERTURBATION PROPAGATION

Four cases to consider… After some algebra:

$$\Delta D_k = \Delta Z_k + \begin{cases} \Delta D_{k-1} & \text{if } I_k \le 0 \text{ and } \Delta D_{k-1} \ge I_k \\ 0 & \text{if } I_k > 0 \text{ and } \Delta D_{k-1} \le I_k \\ I_k & \text{if } I_k \le 0 \text{ and } \Delta D_{k-1} \le I_k \\ \Delta D_{k-1} - I_k & \text{if } I_k > 0 \text{ and } \Delta D_{k-1} \ge I_k \end{cases}$$

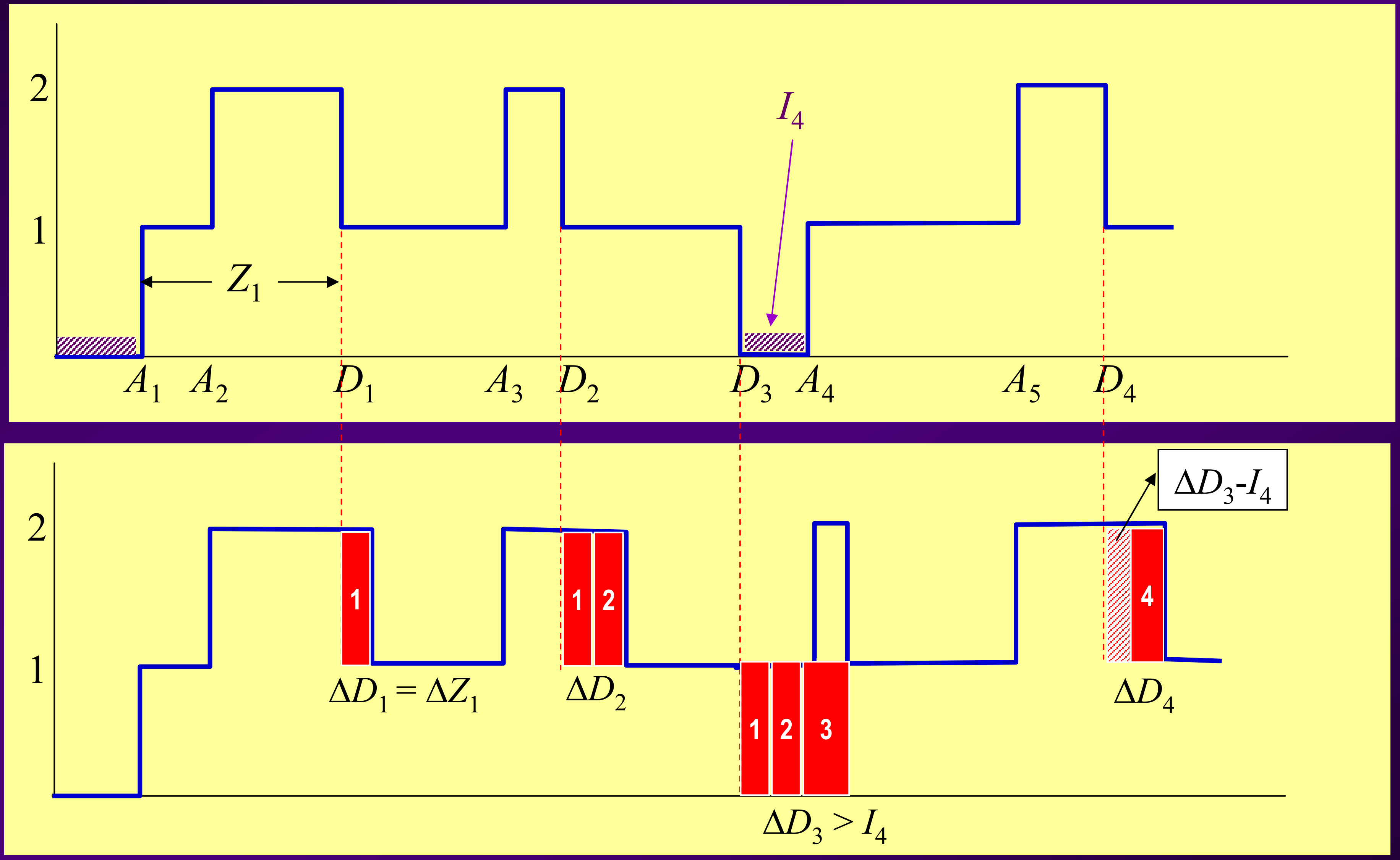
where: $I_k = A_k - D_{k-1}$ **(idle period length if > 0)**

I_k **observed**

ΔZ_k **calculated from Z_k as described earlier (PERTURBATION GENERATION)**

PERTURBATION PROPAGATION

EXAMPLE: Perturbations in mean service time $\Rightarrow \Delta Z_1(\theta), \Delta Z_2(\theta), \ldots$



INFINITESIMAL PERTURBATION ANALYSIS (IPA)

If $\Delta\theta$ is so "small" as to ensure that $\Delta D_{k\text{-}1} \leq I_k$ then

$$\Delta D_k = \Delta Z_k + \begin{cases} \Delta D_{k-1} & \text{if } I_k \leq 0 \\ 0 & \text{otherwise} \end{cases}$$

Perturbation Generation

Perturbation Propagation

If derivatives are used, this can be rigorously shown:

$$\frac{dD_k}{d\theta} = \frac{dZ_k}{d\theta} + \begin{cases} \dfrac{dD_{k-1}}{d\theta} & \text{if } I_k \leq 0 \\ 0 & \text{otherwise} \end{cases}$$

UNIVERSAL IPA ALGORITHM

1. INITIALIZATION: If α feasible at x_0: $\Delta_\alpha := \dfrac{dV_{\alpha,1}}{d\theta}$

Else for all other α: $\Delta_\alpha := 0$

2. WHENEVER β IS OBSERVED (including $\beta = \alpha$) :

If α activated with new lifetime V_α :

2.1. Compute $dV_\alpha / d\theta$

2.2. $\Delta_\alpha := \Delta_\beta + \dfrac{dV_\alpha}{d\theta}$

IMPLEMENTATION: Simple, *non-intrusive*, overhead negligible

See *http://people.bu.edu/cgc/IPA/*

INFINITESIMAL PERTURBATION ANALYSIS (IPA)

QUESTION: Are IPA sensitivity estimators "good" ?

- Unbiasedness
- Consistency

ANSWER: Yes, for a large class of DES that includes

- G/G/1 queueing systems
- Jackson-like queueing networks
  (e.g., no blocking allowed)

NOTE: IPA applies to REAL-VALUED parameters of some event process distribution (e.g., mean interarrival and service times)

IPA UNBIASEDNESS

Theorem. For a given sample function $L(\theta)$, suppose that

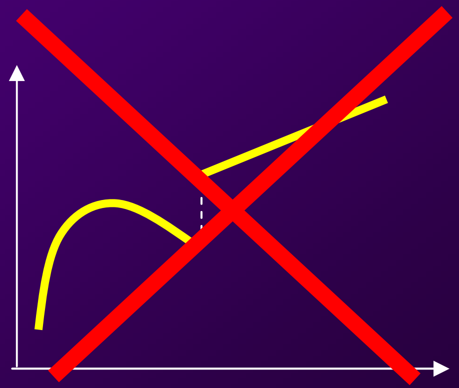
(a) the sample derivative $dL(\theta)/d\theta$ exists w.p. 1 for every $\theta \in \Theta$ where Θ is a closed bounded set, and

(b) $L(\theta)$ is Lipschitz continuous w.p. 1 and the Lipschitz constant has finite first moment.

Then,

$$\frac{d}{d\theta} E[L(\theta)] = E\left[\frac{dL(\theta)}{d\theta}\right]$$

[Rubinstein and Shapiro, 1993]



COMMUTING CONDITION

States x, y, z_1 and events α, β such that

$$p(z_1; x, \alpha) \cdot p(y; z_1, \beta) > 0$$

Then, for some z_2

$$p(z_2; x, \beta) = p(y; z_1, \beta) \text{ and } p(y; z_2, \alpha) = p(z_1; x, \alpha)$$

Moreover, for x, z_1, z_2 s.t. $p(z_1;x,\alpha) = p(z_2;x,\alpha) > 0$, $z_1 = z_2$

$$\begin{tikzcd} & z_1 \arrow[dr, "\beta"] & \\ x \arrow[ur, "\alpha"] \arrow[dr, "\beta"'] & & y \\ & z_2 \arrow[ur, "\alpha"'] & \end{tikzcd}$$

[*Glasserman, 1991*]

EXTENSIONS OF IPA

There are several generalizations, at the expense of more simulation overhead (still, very minimal)

e.g.,

- routing probabilities
- systems with real-time constraints
- some scheduling policies

IMPLEMENTATION:

Still very straightforward and *non-intrusive*

CONCURRENT ESTIMATION

THE CONSTRUCTABILITY PROBLEM

Consider a Discrete Event System (DES) operating under parameter settings $\{u_1, \ldots, u_m\}$

INPUT: $\omega =$ all event lifetimes

$u_1 =$ a parameter setting

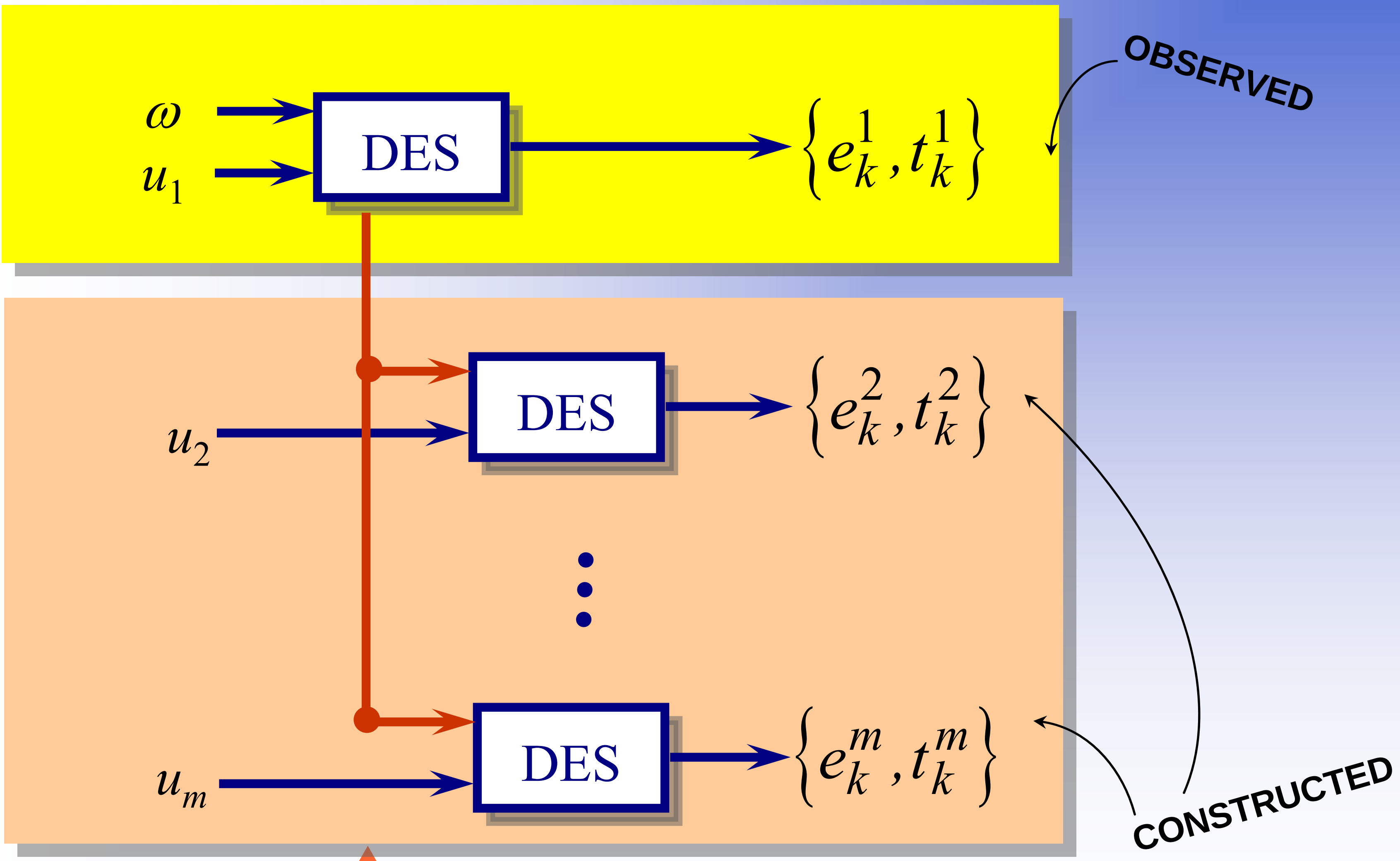
OUTPUT: $\{e_k^1, t_k^1\} =$ observed sample path under u_1

PROBLEM:

Construct sample paths under all $\{u_2, \ldots, u_m\}$ based only on

- The input ω
- The observed sample path $\{e_k^1, t_k^1\}$
- Observed state history

THE CONSTRUCTABILITY PROBLEM *CONTINUED*



THE CONSTRUCTABILITY PROBLEM *CONTINUED*

- **OBSERVABILITY**:

A sample path $\left\{e_k^j, t_k^j\right\}$ is *observable* with respect to $\left\{e_k, t_k\right\}$ if

$$\Gamma\left(x_k^j\right) \subseteq \Gamma\left(x_k\right) \quad \text{for all } k = 0,1,\ldots$$

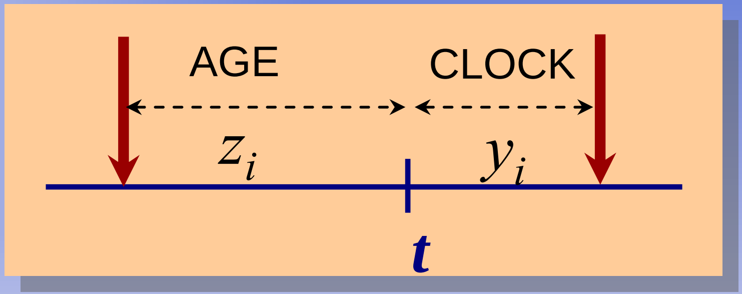
INTERPRETATION: For every state x_k^j visited in the constructed sample path, all feasible events $i \in \Gamma\left(x_k^j\right)$ are also feasible in the corresponding observed state x_k

Ref: *Cassandras and Strickland, 1989*

THE CONSTRUCTABILITY PROBLEM *CONTINUED*

Define the conditional cdf of an *event clock* given the *event age*:

$$H(t, z_i) = P[y_i \le t \mid z_i]$$



CONSTRUCTABILITY:

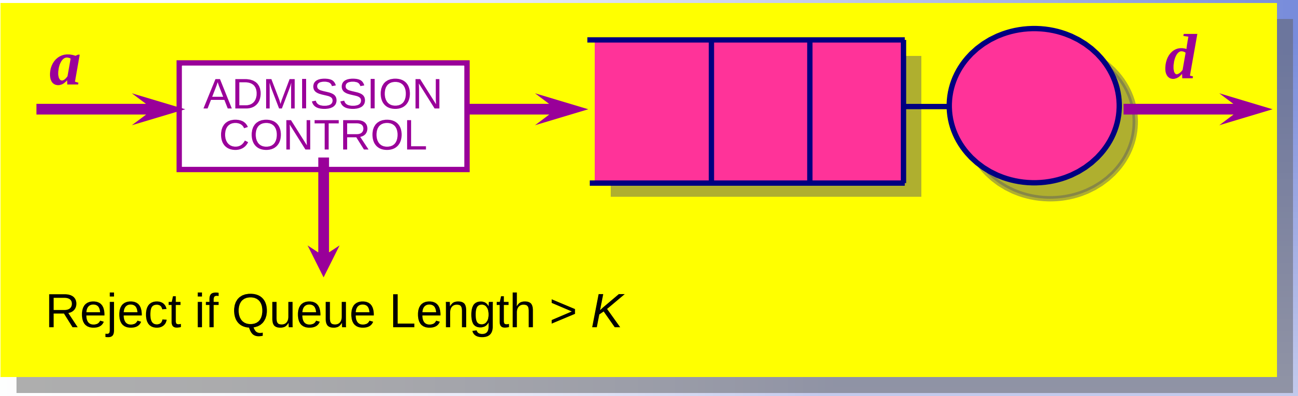
A sample path $\{e_k^j, t_k^j\}$ is *constructable* with respect to $\{e_k, t_k\}$ if

1. $\Gamma(x_k^j) \subseteq \Gamma(x_k)$ for all $k = 0, 1, \ldots$

2. $H^j(t, z_{i,k}^j) = H(t, z_{i,k})$ for all $i \in \Gamma(x_k^j)$, $k = 0, 1, \ldots$

CONSTRUCTABILITY: AN EXAMPLE

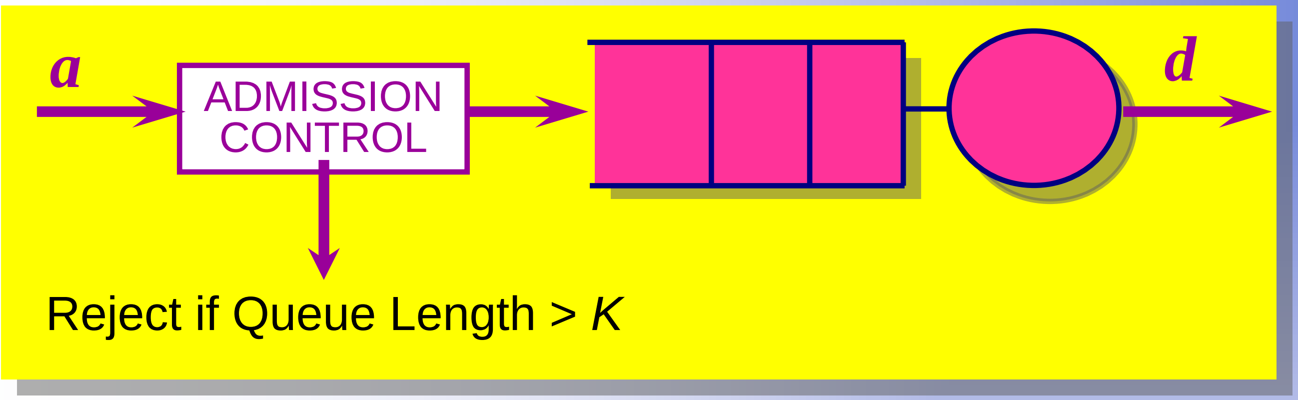
A simple design/optimization problem



PROBLEM:

- How does the system behave under different choices of *K*?
- What is the *optimal K*?

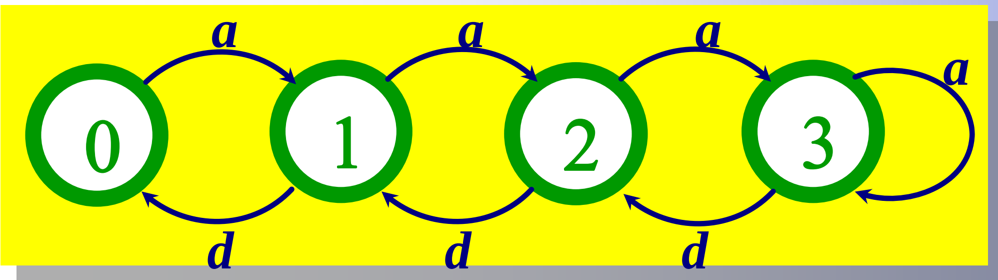
CONSTRUCTABILITY: AN EXAMPLE *CONTINUED*



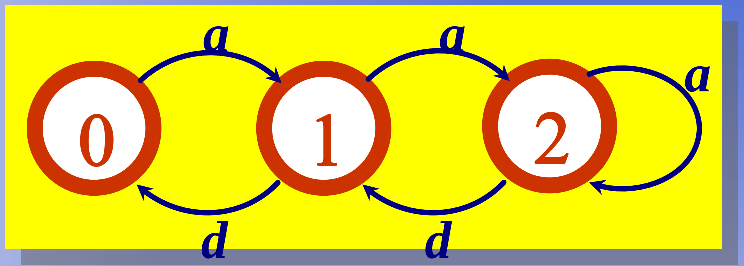
CONCURRENT ESTIMATION APPROACH:

- Choose any K
- Simulate (or observe *actual system*) under K
- Apply Concurrent Simulation to *LEARN* effect of all other feasible K

CONSTRUCTABILITY: AN EXAMPLE *CONTINUED*

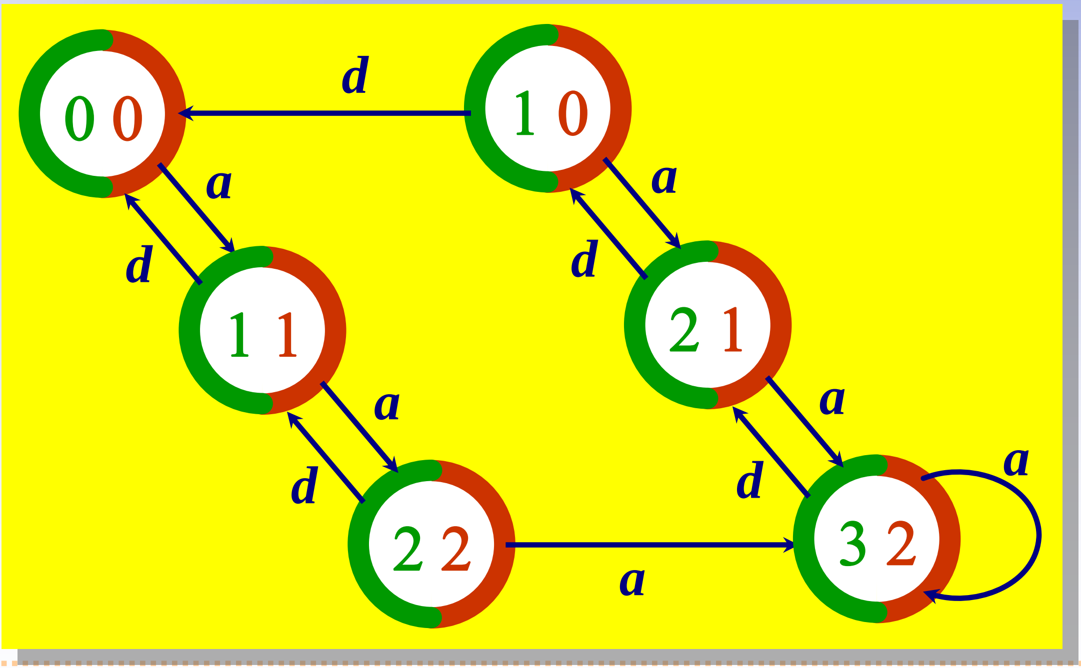


NOMINAL SYSTEM: $K = 3$

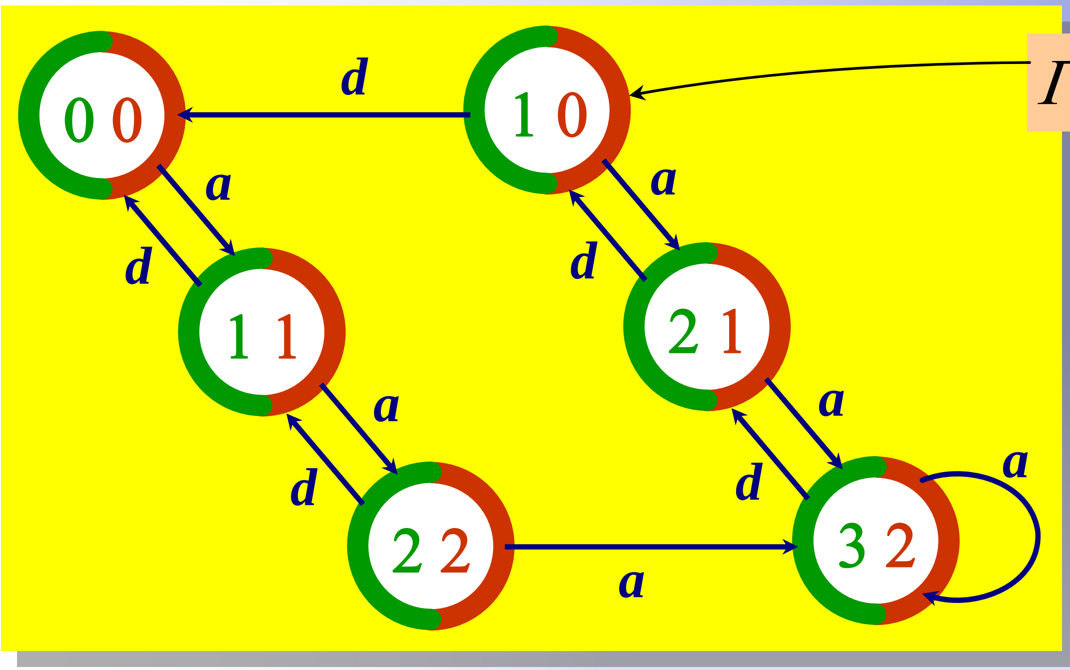


PERTURBED SYSTEM: $K = 2$

AUGMENTED SYSTEM:
Check for OBSERVABILITY



CONSTRUCTABILITY: AN EXAMPLE *CONTINUED*



$$\Gamma(0) = \{a\} \subset \Gamma(1) = \{a, d\}$$

OBSERVABILITY satisfied!

However, if roles of NOMINAL and PERTURBED are reversed, then OBSERVABILITY is ***not*** satisfied...

SOLVING CONSTRUCTABILITY

- Constructability is ***not*** easily satisfied, in general
- However, the CONSTRUCTABILITY PROBLEM can be solved a some cost

SOLUTION METHODOLOGIES:

- STANDARD CLOCK (SC) methodology for Markovian systems

Ref: *Vakili, 1991*

- AUGMENTED SYSTEM ANALYSIS (ASA) for systems with at most one non-Markovian event

Ref: *Cassandras and Strickland, 1989*

- TIME WARPING ALGORITHM (TWA) for arbitrary systems

Ref: *Cassandras and Panayiotou, 1996*

AUGMENTED SYSTEM ANALYSIS (ASA)

- Assume Markovian DES $\Rightarrow$ only OBSERVABILITY required
- Observe sample path of Σ_1
- As state sequence unfolds, check for OBSERVABILITY for every constructed $n = 2,\ldots,m$
- If OBSERVABILITY violated for some n, ***suspend*** nth sample path construction
- Wait until a state of Σ_1 is entered s.t. OBSERVABILITY satisfied for ***suspended*** nth sample path and resume construction

EVENT MATCHING ALGORITHM

INITIALIZE: $A = \{2,\ldots,m\}$ = ACTIVE sample paths

1. Observe every event e in sample path of Σ_1

2. Update state: $x'_n = f(x_n, e)$

3. For each $n = 2,\ldots,m$, check if [is OBSERVABILITY satisfied?]

3.1. If $\Gamma(x'_1) \supseteq \Gamma(x'_n)$, add n to A (if $n \in A$, leave n in A)

3.2. Else, remove n from A and leave x'_n unchanged

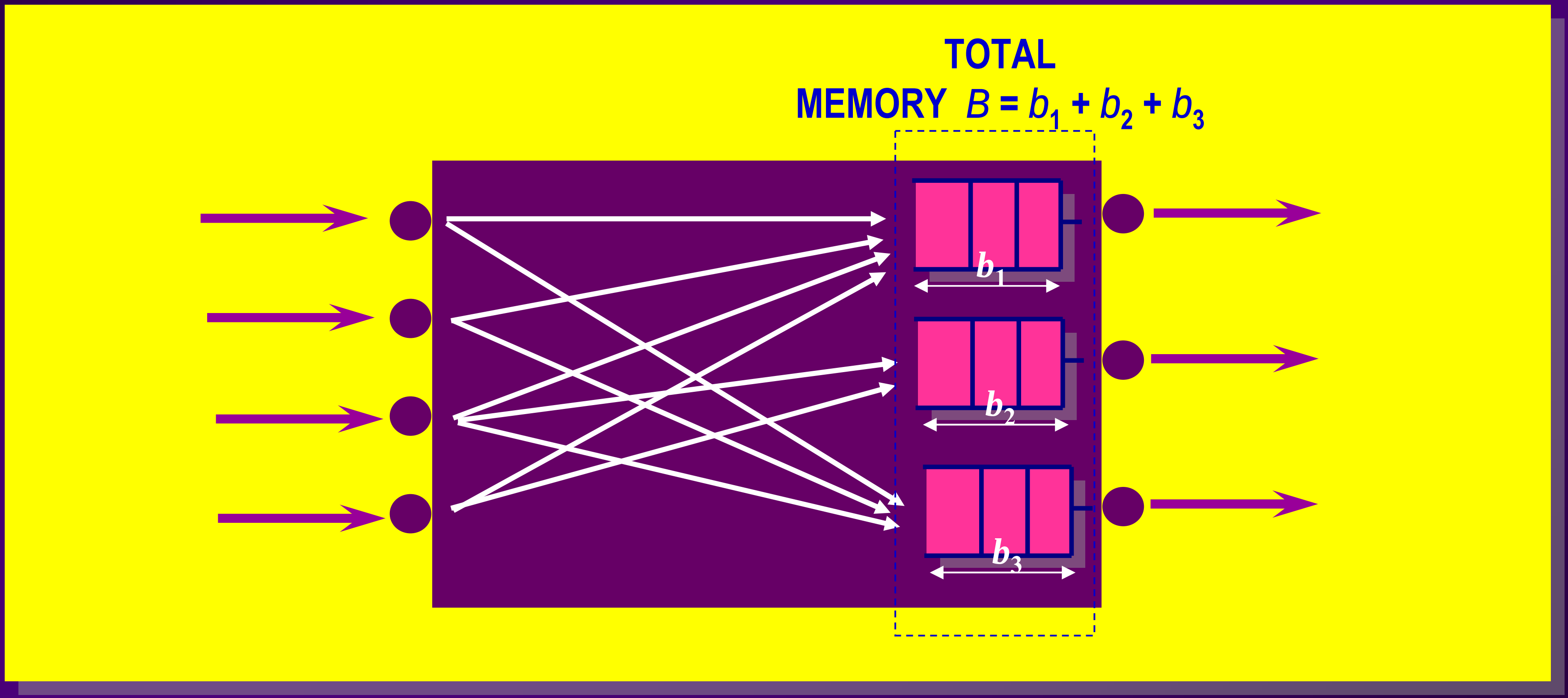
3. Return to **Step 1** for next event in sample path of Σ_1

- *Price to pay in ASA*: long suspensions possible in **Step 3.1**

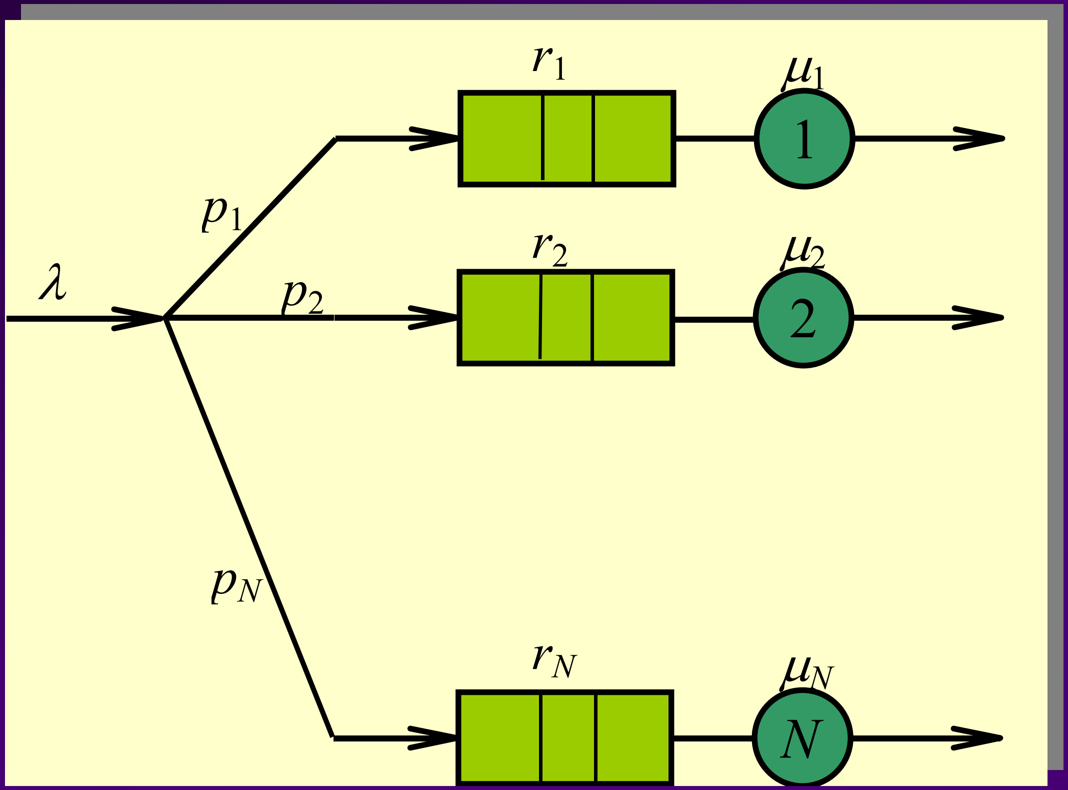
SOME APPLICATIONS

BUFFER ALLOCATION IN A SWITCH

NETWORK SWITCH MEMORY ALLOCATION OVER OUTPUT PORTS



BUFFER ALLOCATION IN A SWITCH

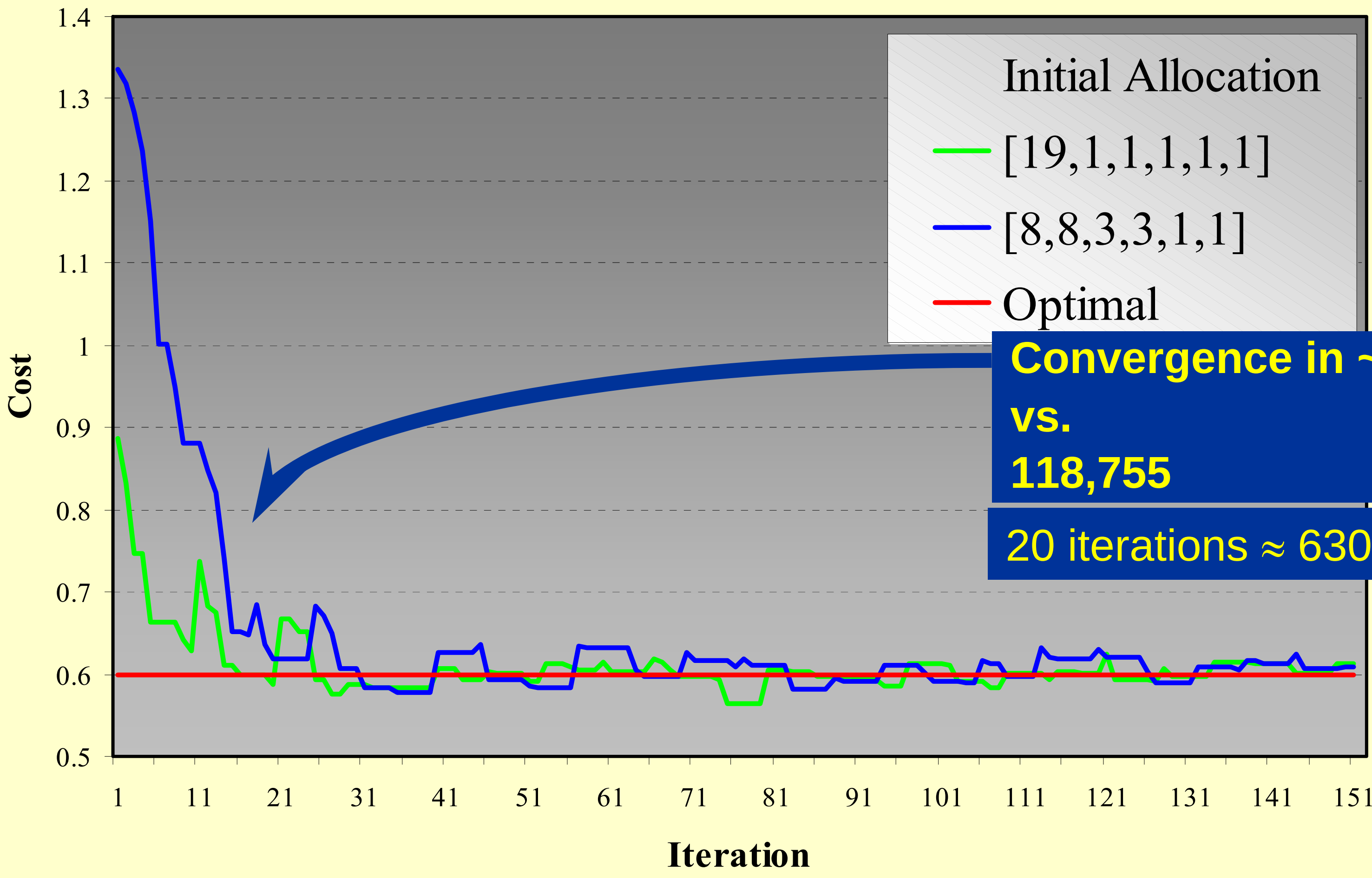


Allocate K buffer slots (RESOURCES) over N servers (USERS) so as to minimize *BLOCKING PROBABILITY*

subject to $\sum_{i=1}^{N} r_i = K$

For N=6, K=24 → Possible allocations = **118,755**

CONCURRENT ESTIMATION + OPTIMIZATION

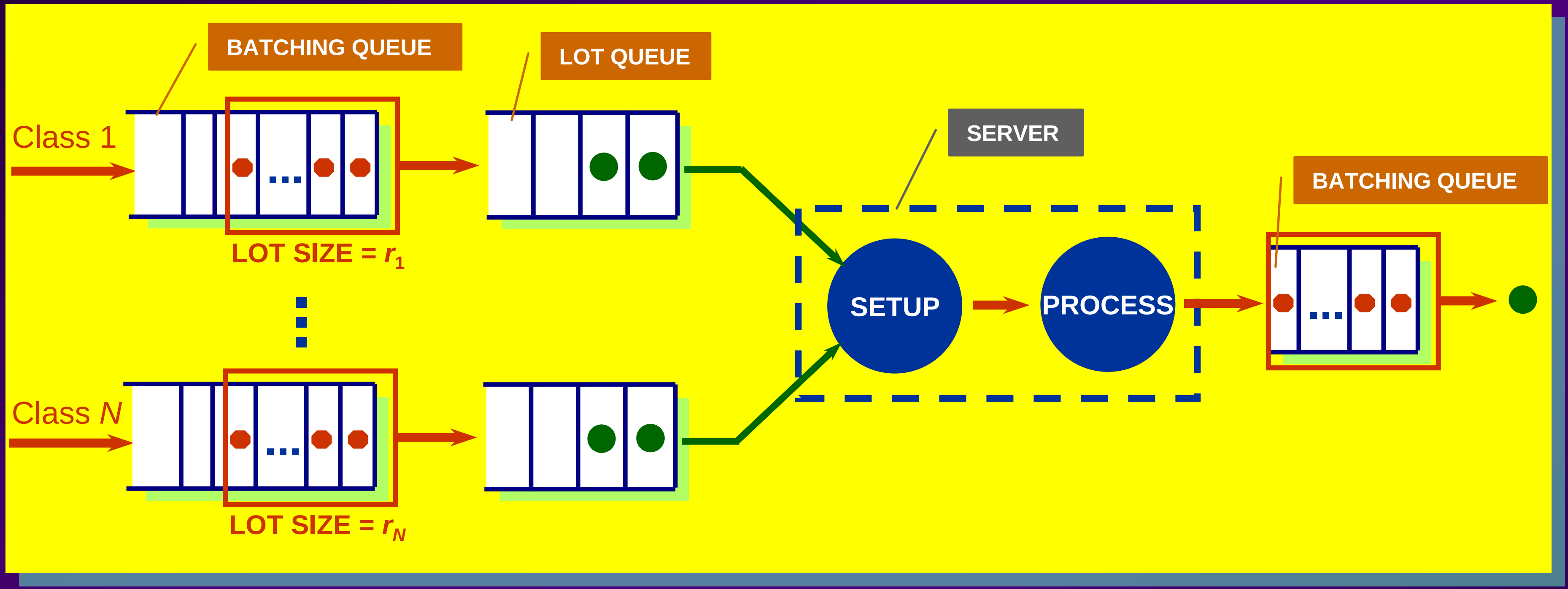


Convergence in ~ 20 iterations vs. 118,755

20 iterations ≈ 630,000 events

- $\lambda = 5.0$, $\mu_i = 1.0$, $p_i = 1/6$ for all $i = 1,\dots,6$
- Optimal = [4,4,4,4,4,4]

LOT SIZING IN MANUFACTURING

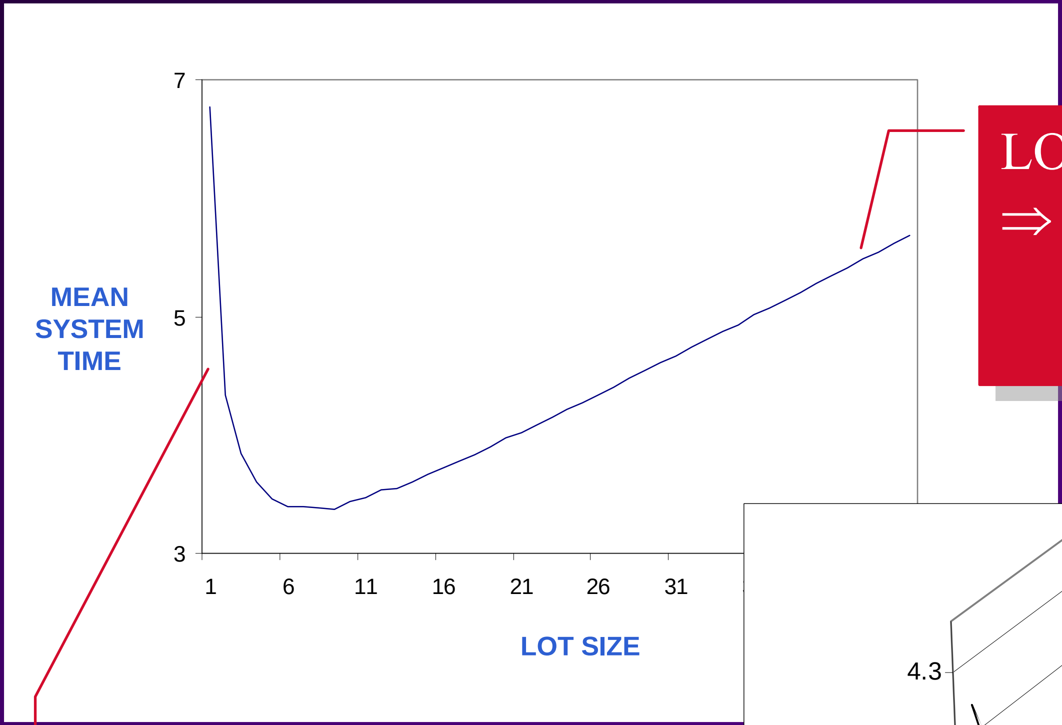


LOT SIZING: Determine Lot Sizes to minimize Average Lead Time

COMPLEXITY: For 3 product classes and lot sizes in [1,100] :

10^6 possible allocations

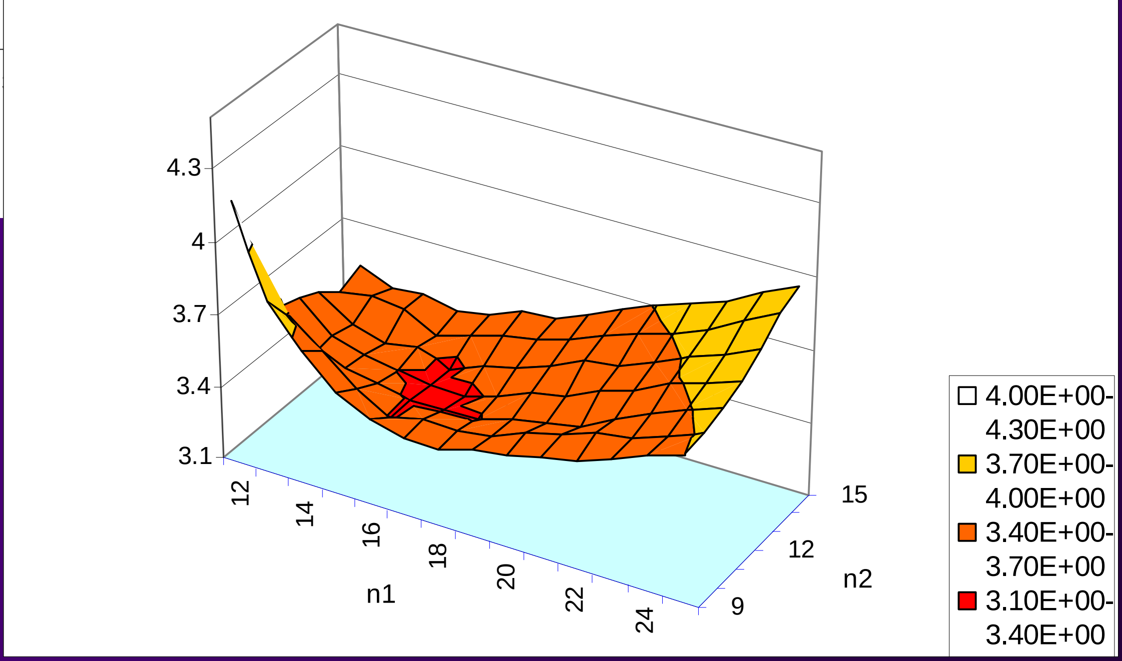
LOT SIZING IN MANUFACTURING



LOT SIZE r_i too ***large***

$\Rightarrow$ All other classes delayed

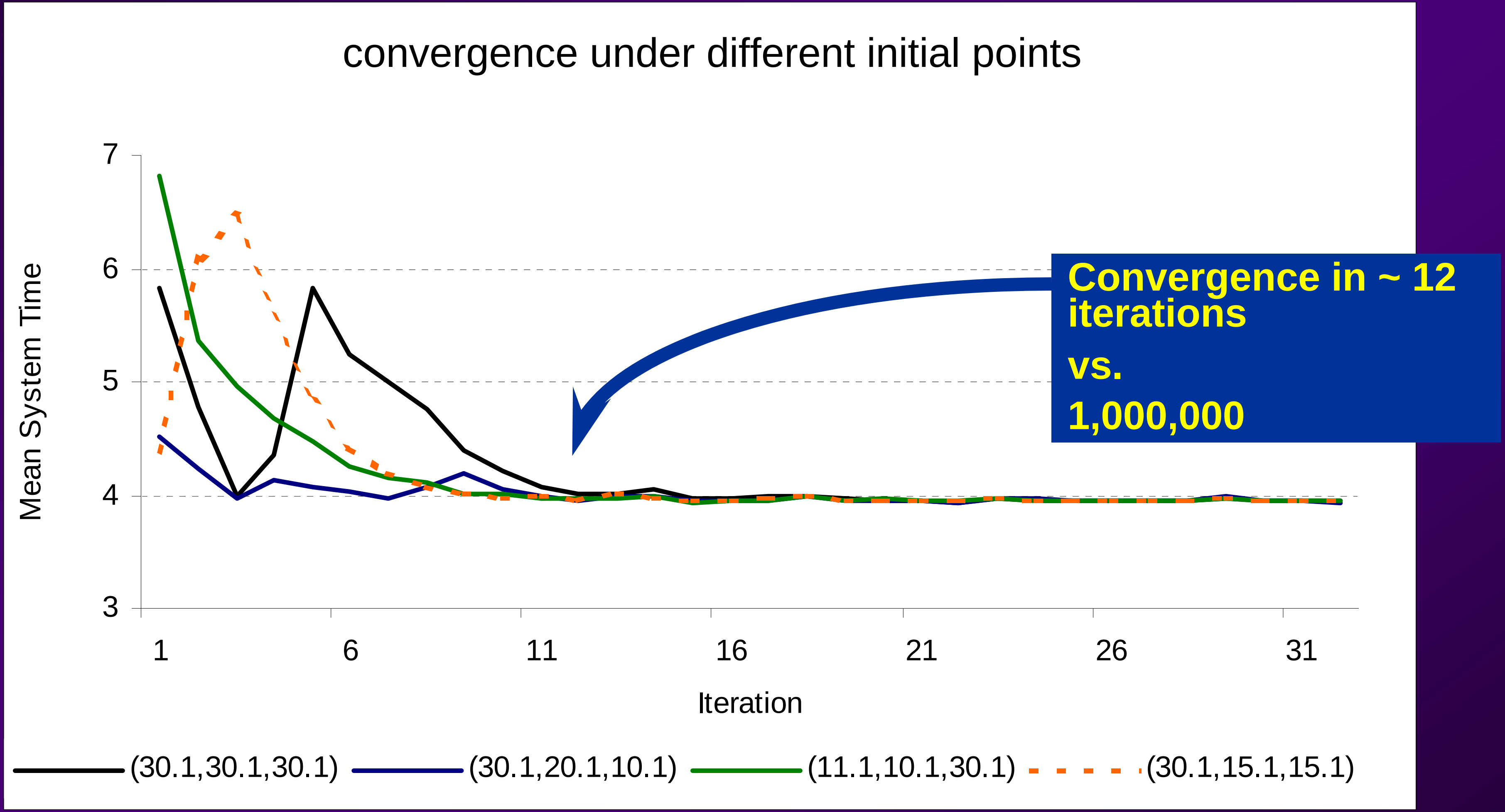
+ class i parts delayed waiting for lot at output



LOT SIZE r_i too ***small***

$\Rightarrow$ Too many setups, instability possible

CONCURRENT SIMULATION + OPTIMIZATION



ABSTRACTION (AGGREGATION) OF DES

THREE FUNDAMENTAL *COMPLEXITY LIMITS*

$1/T^{1/2}$ LIMIT

NP-HARD LIMIT

O.

CE

C

(e.

Tradeoff between
GENERALITY and **EFFICIENCY**
of an algorithm

[*Wolpert and Macready, IEEE TEC, 1997*]

NO-FREE-LUNCH LIMIT

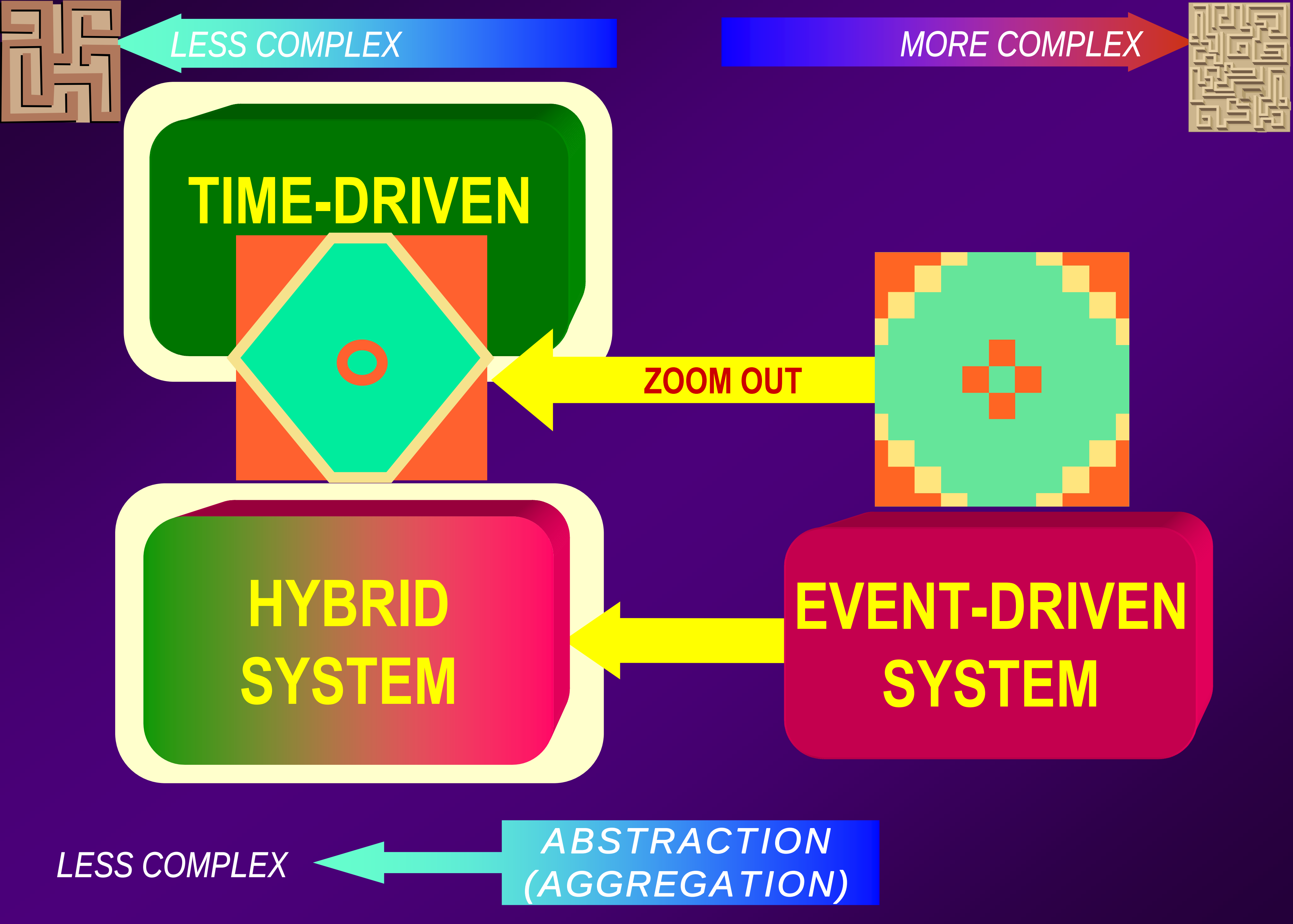
THREE FUNDAMENTAL *COMPLEXITY LIMITS*

$1/T^{1/2}$ LIMIT

NP-HARD LIMIT

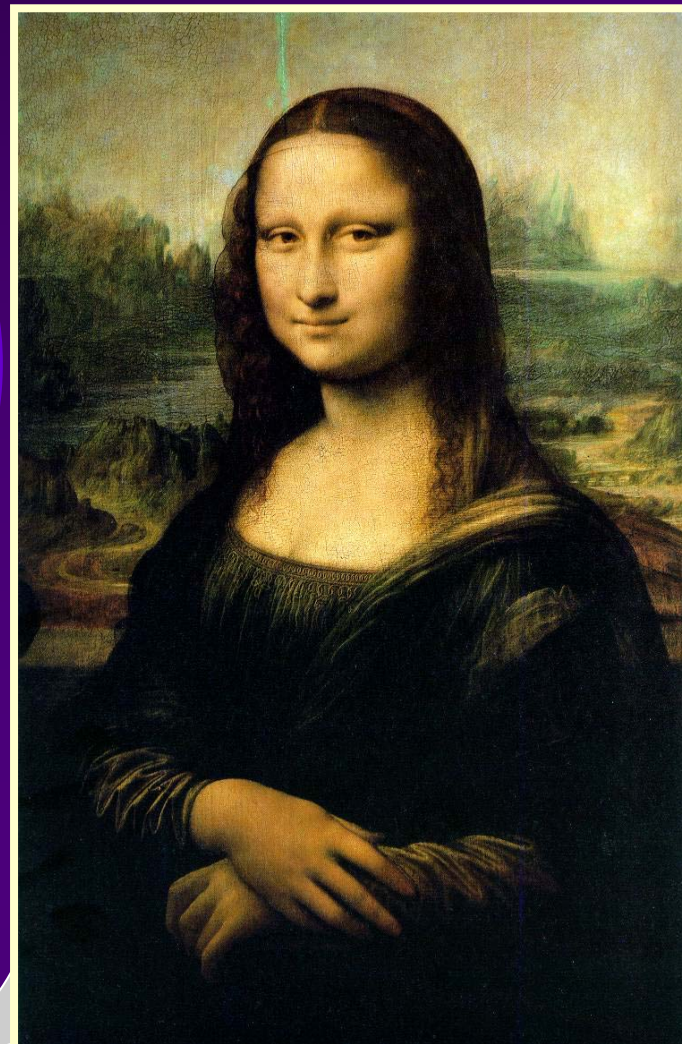
Effect is MULTIPLICATIVE!

NO-FREE-LUNCH LIMIT

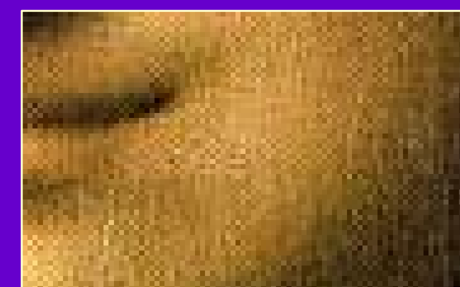

LESS COMPLEX
MORE COMPLEX
TIME-DRIVEN
ZOOM OUT
HYBRID SYSTEM
EVENT-DRIVEN SYSTEM
LESS COMPLEX
ABSTRACTION (AGGREGATION)

WHAT IS THE RIGHT ABSTRACTION LEVEL ?

TOO FAR…
model not detailed enough



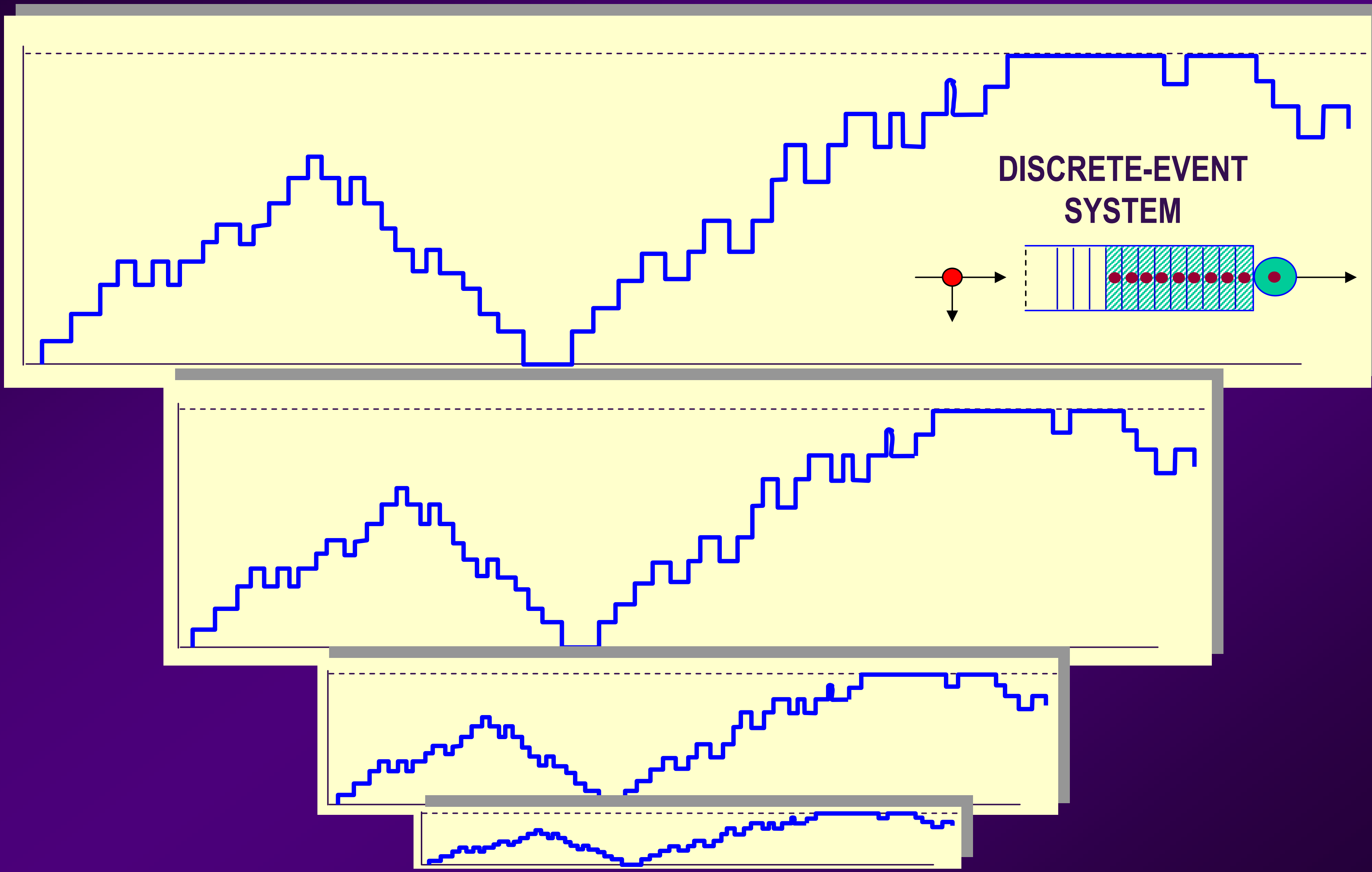
JUST RIGHT…
good model



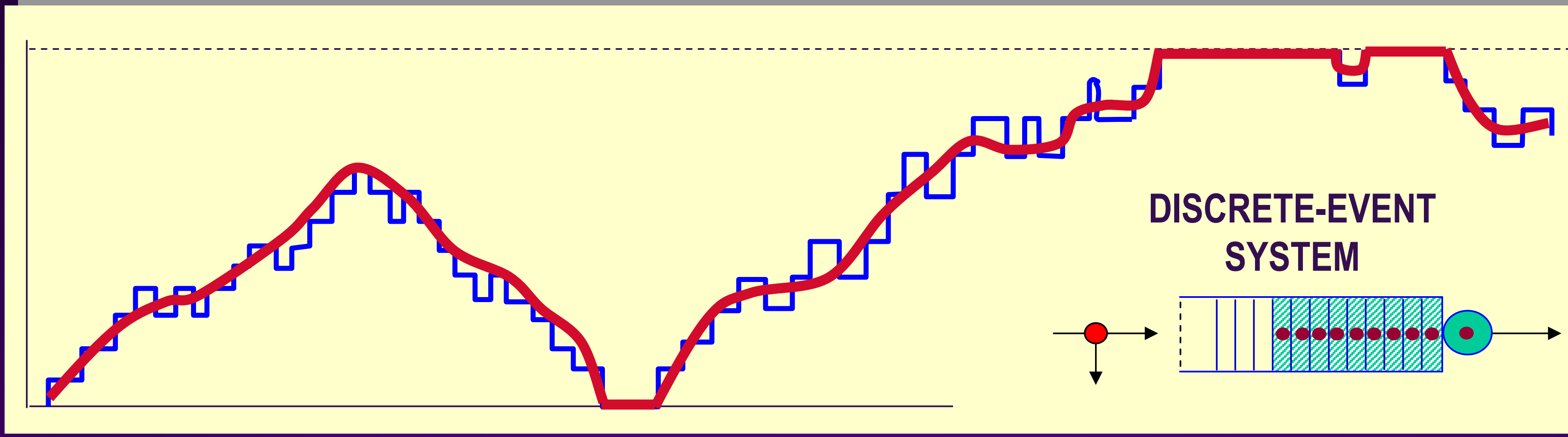
TOO CLOSE…
too much undesirable detail

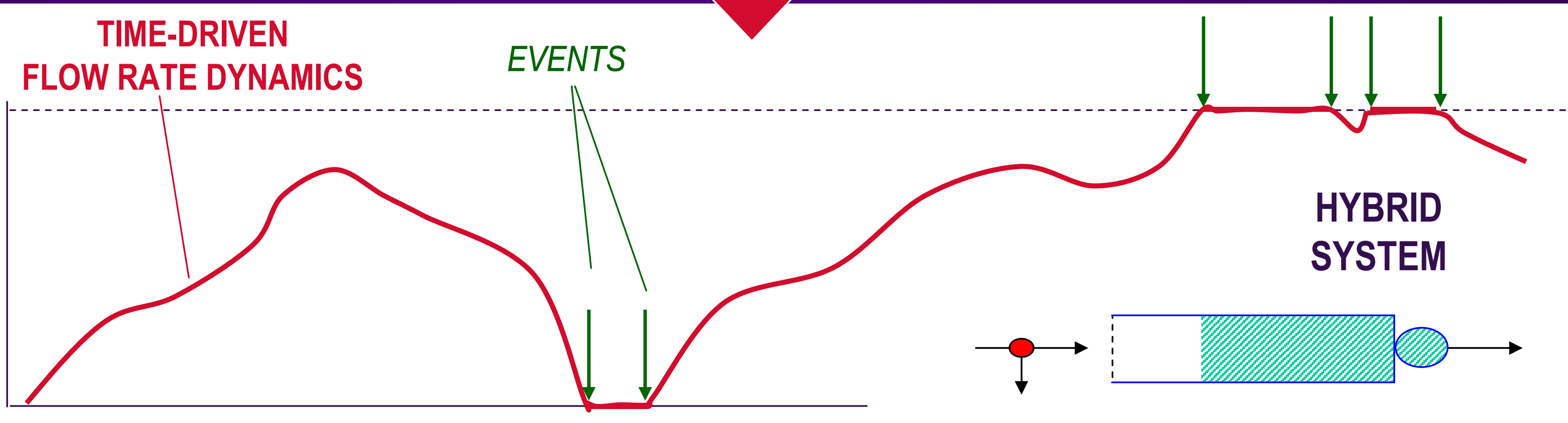
CREDIT: W.B. Gong

ABSTRACTION OF A DISCRETE-EVENT SYSTEM

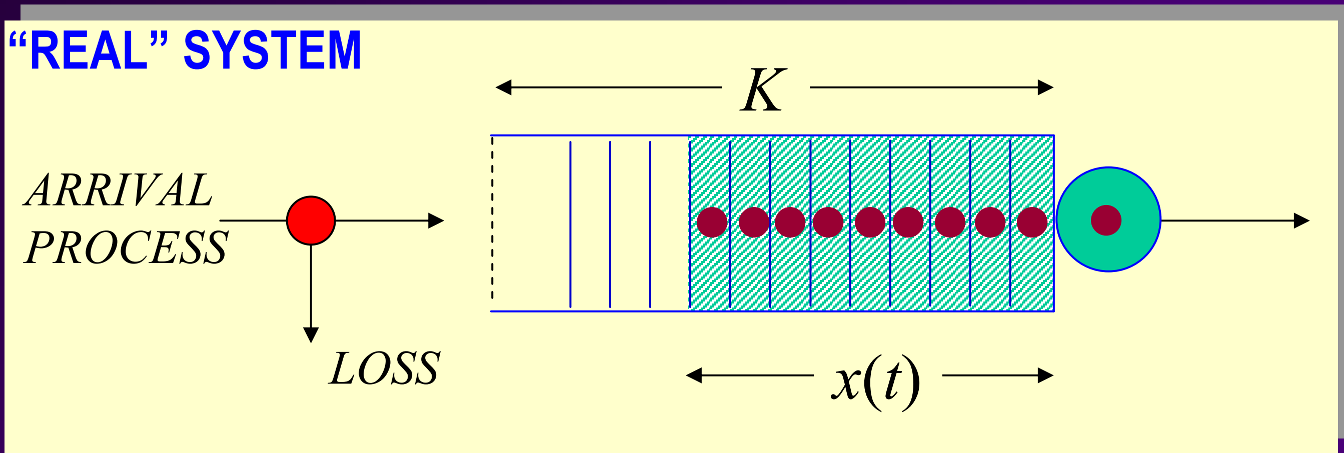


ABSTRACTION OF A DISCRETE-EVENT SYSTEM





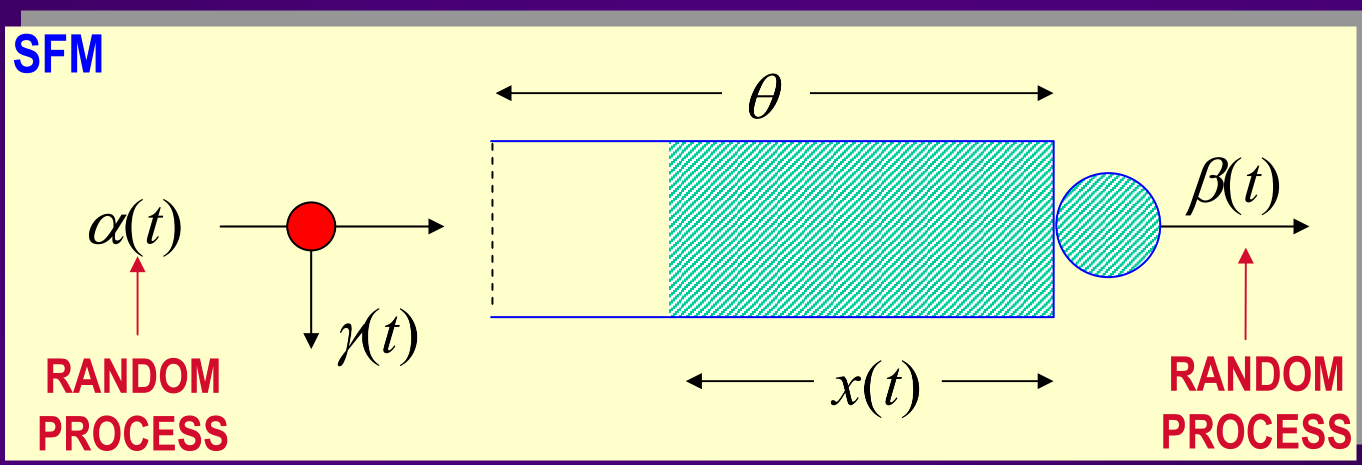
THRESHOLD BASED BUFFER CONTROL



$L(K)$: Loss Rate

$Q(K)$: Mean Queue Lenth

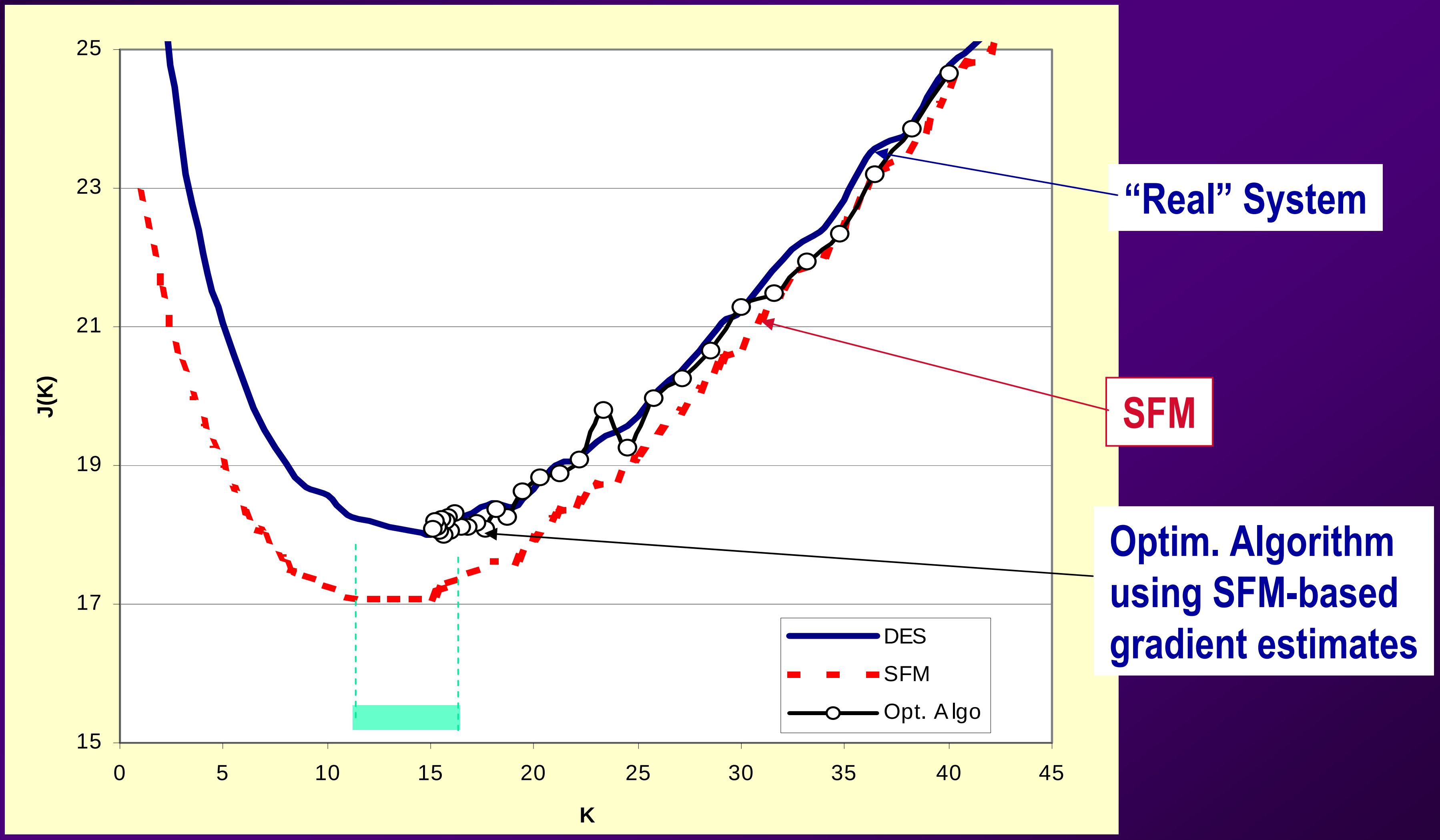
PROBLEM: Determine K to minimize $[Q(K) + R \cdot L(K)]$



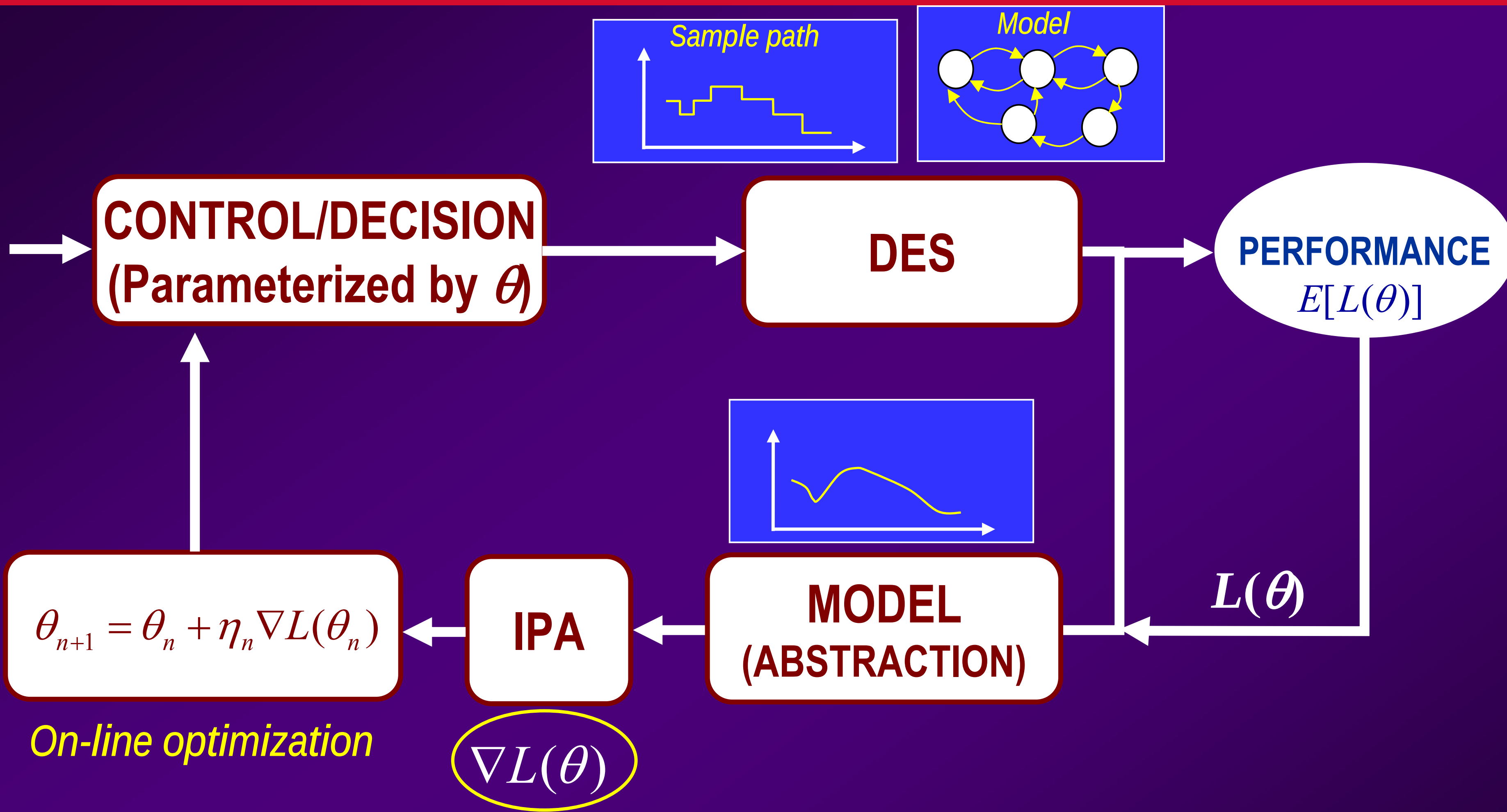
SURROGATE PROBLEM: Determine θ to minimize $[Q^{SFM}(\theta) + R \cdot L^{SFM}(\theta)]$

THRESHOLD BASED BUFFER CONTROL

CONTINUED



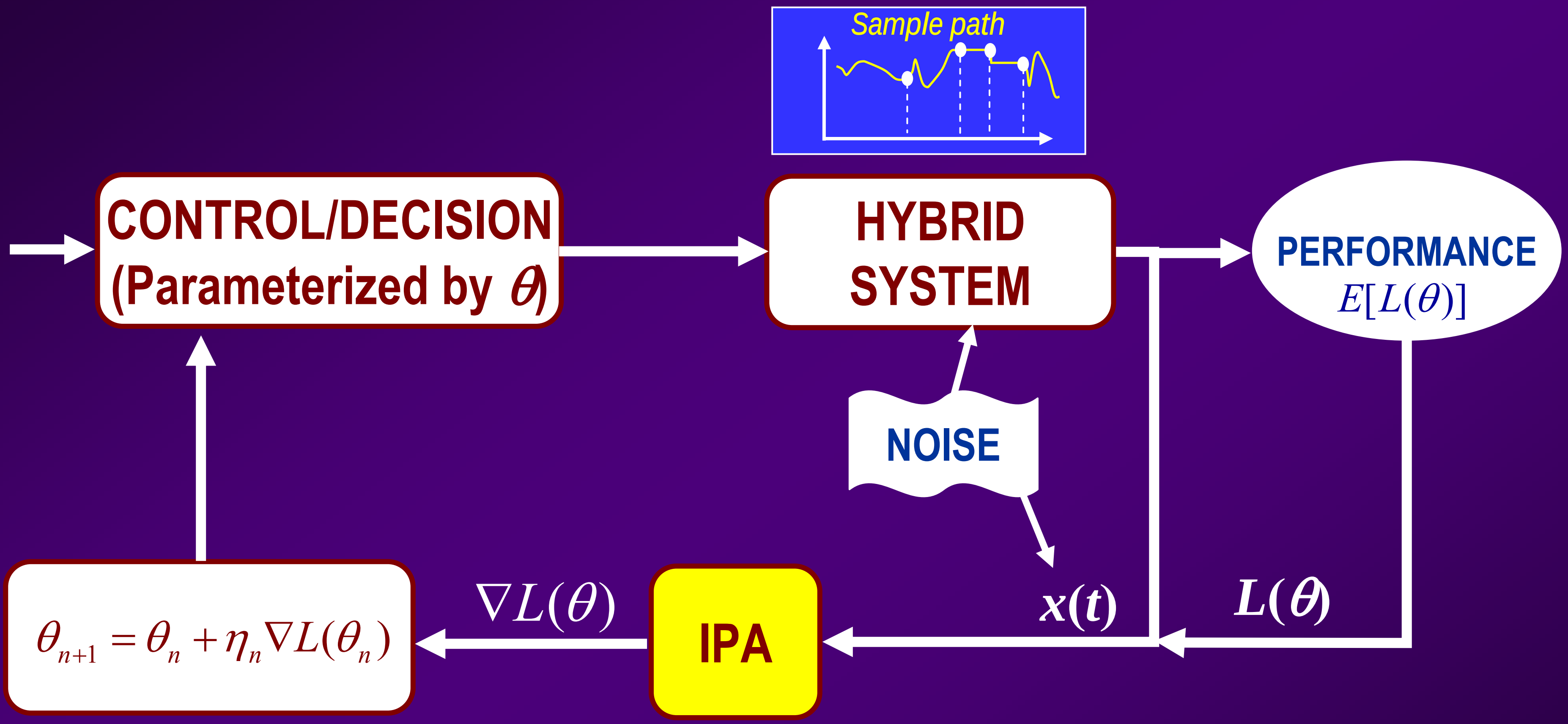
REAL-TIME STOCHASTIC OPTIMIZATION



- **Unbiased estimators**
- **General distributions**
- **Simple on-line implementation**

Not always true in DES !

REAL-TIME STOCHASTIC OPTIMIZATION: *HYBRID SYSTEMS*



A general framework for an IPA theory in Hybrid Systems?

PERFORMANCE OPTIMIZATION AND IPA

Performance metric (objective function):

$$J\left(\theta; x(\theta,0), T\right) = E\left[L\left(\theta; x(\theta,0), T\right)\right]$$

$$\Downarrow$$

$$L\left(\theta\right) = \sum_{k=0}^{N} \int_{\tau_k}^{\tau_{k+1}} L_k(x, \theta, t)\,dt$$

IPA goal:

- Obtain unbiased estimates of $\dfrac{dJ\left(\theta; x(\theta,0), T\right)}{d\theta}$, normally $\dfrac{dL\left(\theta\right)}{d\theta}$

- Then: $\theta_{n+1} = \theta_n + \eta_n \dfrac{dL(\theta_n)}{d\theta}$

NOTATION: $x'\left(t\right) = \dfrac{\partial x\left(\theta, t\right)}{\partial\theta}, \quad \tau'_k = \dfrac{\partial \tau_k\left(\theta\right)}{\partial\theta}$

RECALL: HYBRID AUTOMATA

$$G_h = (Q, X, E, U, f, \phi, Inv, guard, \rho, q_0, \mathbf{x}_0)$$

Q: set of discrete states (modes)

X: set of continuous states (normally R^n)

E: set of events

U: set of admissible controls

f: vector field, $f : Q \times X \times U \to X$

ϕ: discrete state transition function, $\phi : Q \times X \times E \to Q$

Inv: set defining an invariant condition (domain), $Inv \subseteq Q \times X$

$guard$: set defining a guard condition, $guard \subseteq Q \times Q \times X$

ρ: reset function, $\rho : Q \times Q \times X \times E \to X$

q_0: initial discrete state

$\mathbf{x}_0$: initial continuous state

THE IPA CALCULUS

IPA: *THREE FUNDAMENTAL EQUATIONS*

System dynamics over $(\tau_k(\theta),\ \tau_{k+1}(\theta)]$: $\quad \dot{x} = f_k(x,\theta,t)$

NOTATION: $\quad x'(t) = \dfrac{\partial x(\theta,t)}{\partial \theta}, \quad \tau_k' = \dfrac{\partial \tau_k(\theta)}{\partial \theta}$

---

1. **Continuity at events:** $x(\tau_k^+) = x(\tau_k^-)$

Take $d/d\theta$:

$$x'(\tau_k^+) = x'(\tau_k^-) + [f_{k-1}(\tau_k^-) - f_k(\tau_k^+)]\tau'_k$$

If no continuity, use reset condition $\Rightarrow$

$$x'(\tau_k^+) = \frac{d\rho(q,q',x,\upsilon,\delta)}{d\theta}$$

IPA: *THREE FUNDAMENTAL EQUATIONS*

2. Take $d/d\theta$ of system dynamics $\dot{x} = f_k(x,\theta,t)$ over $(\tau_k(\theta),\ \tau_{k+1}(\theta)]$:

$$\frac{dx'(t)}{dt} = \frac{\partial f_k(t)}{\partial x} x'(t) + \frac{\partial f_k(t)}{\partial \theta}$$

Solve $\dfrac{dx'(t)}{dt} = \dfrac{\partial f_k(t)}{\partial x} x'(t) + \dfrac{\partial f_k(t)}{\partial \theta}$ **over** $(\tau_k(\theta),\ \tau_{k+1}(\theta)]$:

$$x'(t) = e^{\int_{\tau_k}^{t} \frac{\partial f_k(u)}{\partial x} du} \left[\int_{\tau_k}^{t} \frac{\partial f_k(v)}{\partial \theta} e^{-\int_{\tau_k}^{v} \frac{\partial f_k(u)}{\partial x} du} dv + x'(\tau_k^+) \right]$$

initial condition from 1 above

NOTE: If there are no events (pure time-driven system), IPA reduces to this equation

IPA: *THREE FUNDAMENTAL EQUATIONS*

3. Get τ_k' depending on the event type:

- **Exogenous** event: By definition, $\boxed{\tau_k' = 0}$

- **Endogenous** event: occurs when $g_k(x(\theta,\tau_k),\theta) = 0$

$$\boxed{\tau_k' = -\left[\frac{\partial g}{\partial x} f_k(\tau_k^-)\right]^{-1}\left(\frac{\partial g}{\partial \theta} + \frac{\partial g}{\partial x} x'(\tau_k^-)\right)}$$

- **Induced** events:

$$\boxed{\tau_k' = -\left[\frac{\partial y_k(\tau_k)}{\partial t}\right]^{-1} y_k'(\tau_k^+)}$$

IPA: *THREE FUNDAMENTAL EQUATIONS*

Ignoring resets and induced events:

1. $x'(\tau_k^+) = x'(\tau_k^-) + [f_{k-1}(\tau_k^-) - f_k(\tau_k^+)] \cdot \tau'_k$

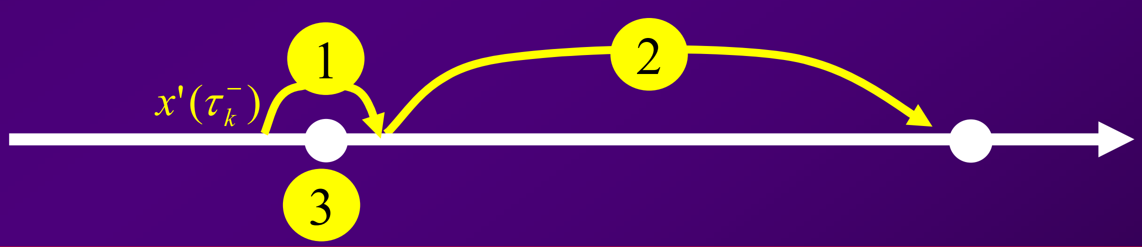
2. $$x'(t) = e^{\int_{\tau_k}^{t} \frac{\partial f_k(u)}{\partial x} du} \left[\int_{\tau_k}^{t} \frac{\partial f_k(v)}{\partial \theta} e^{-\int_{\tau_k}^{v} \frac{\partial f_k(u)}{\partial x} du} dv + x'(\tau_k^+) \right]$$

3. $\tau'_k = 0$ **or** $$\tau'_k = -\left[\frac{\partial g}{\partial x} f_k(\tau_k^-) \right]^{-1} \left(\frac{\partial g}{\partial \theta} + \frac{\partial g}{\partial x} x'(\tau_k^-) \right)$$

Recall:

$$x'(t) = \frac{\partial x(\theta, t)}{\partial \theta}$$

$$\tau'_k = \frac{\partial \tau_k(\theta)}{\partial \theta}$$



Cassandras et al, Europ. J. Control, 2010

IPA PROPERTIES

Back to performance metric: $$L(\theta) = \sum_{k=0}^{N} \int_{\tau_k}^{\tau_{k+1}} L_k(x,\theta,t)\,dt$$

NOTATION: $$L_k'(x,\theta,t) = \frac{\partial L_k(x,\theta,t)}{\partial \theta}$$

Then: $$\frac{dL(\theta)}{d\theta} = \sum_{k=0}^{N} \Bigg[\underbrace{\tau_{k+1}' \cdot L_k(\tau_{k+1}) - \tau_k' \cdot L_k(\tau_k)}_{\text{What happens at event times}} + \underbrace{\int_{\tau_k}^{\tau_{k+1}} L_k'(x,\theta,t)\,dt}_{\text{What happens between event times}} \Bigg]$$

IPA PROPERTIES

THEOREM 1: If either 1,2 holds, then $dL(\theta)/d\theta$ depends only on information available at event times τ_k:

1. $L(x,\theta,t)$ **is independent of** t **over** $[\tau_k(\theta),\ \tau_{k+1}(\theta)]$ **for all** k

2. $L(x,\theta,t)$ **is only a function of** x **and for all** t **over** $[\tau_k(\theta),\ \tau_{k+1}(\theta)]$**:**

$$\frac{d}{dt}\frac{\partial L_k}{\partial x} = \frac{d}{dt}\frac{\partial f_k}{\partial x} = \frac{d}{dt}\frac{\partial f_k}{\partial \theta} = 0$$

[Yao and Cassandras, 2010]

$$\frac{dL(\theta)}{d\theta} = \sum_{k=0}^{N}\left[\tau'_{k+1}\cdot L_k(\tau_{k+1}) - \tau'_k \cdot L_k(\tau_k) + \cancel{\int_{\tau_k}^{\tau_{k+1}} L'_k(x,\theta,t)dt}\right]$$

IMPLICATION: - Performance sensitivities can be obtained from information limited to event times, which is easily observed

- *No need to track system in between events !*

IPA PROPERTIES

EXAMPLE WHERE THEOREM 1 APPLIES (simple tracking problem):

$$\min_{\theta,\phi} E\left[\int_0^T [x(t) - g(\phi)]dt\right] \quad \Rightarrow \frac{\partial L}{\partial x} = 1$$

$$\text{s.t.} \quad \dot{x}_k = a_k x_k(t) + u_k(\theta_k) + w_k(t) \quad \Rightarrow \frac{\partial f_k}{\partial x_k} = a_k, \quad \frac{\partial f_k}{\partial \theta_k} = \frac{du_k}{d\theta_k}$$

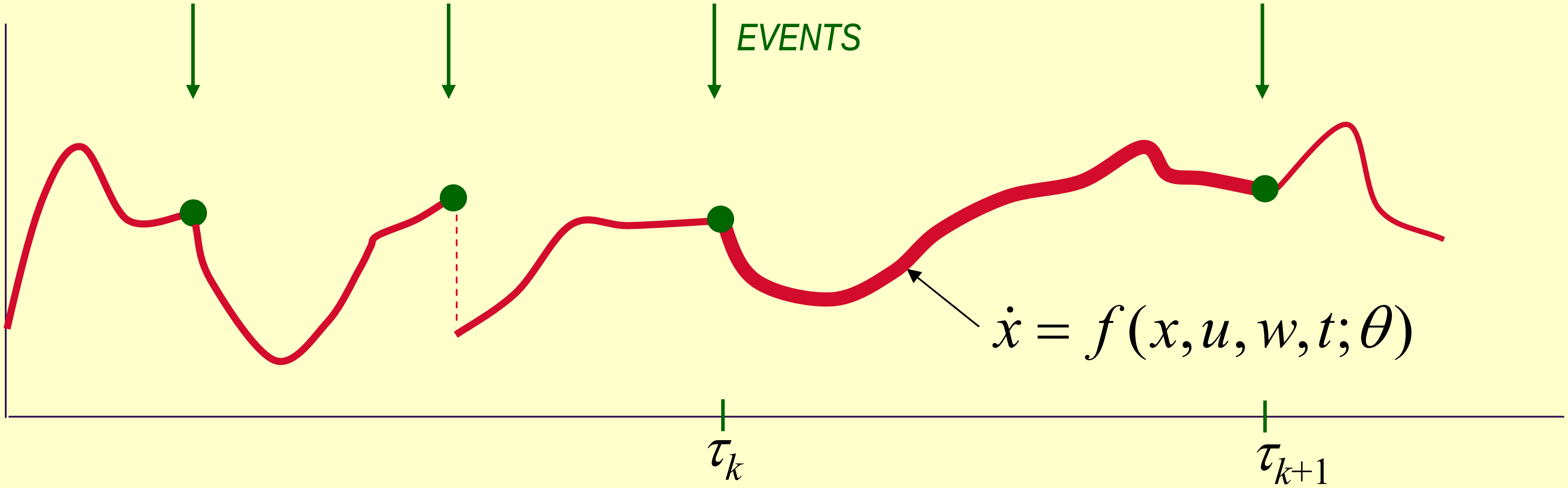
$$k = 1, \ldots, N$$

NOTE: THEOREM 1 provides *sufficient* conditions only.
IPA still depends on info. limited to event times if

$$\dot{x}_k = a_k x_k(t) + u_k(\theta_k, t) + w_k(t)$$
$$k = 1, \ldots, N$$

for "nice" functions $u_k(\theta_k,t)$, e.g., $b_k\theta t$

IPA PROPERTIES



Evaluating $x(t;\theta)$ requires full knowledge of w and f values (obvious)

However, $\dfrac{dx(t;\theta)}{d\theta}$ may be *independent* of w and f values (*NOT* obvious)

It often depends only on:
- event times τ_k
- possibly $f(\tau_{k+1}^-)$

IPA PROPERTIES

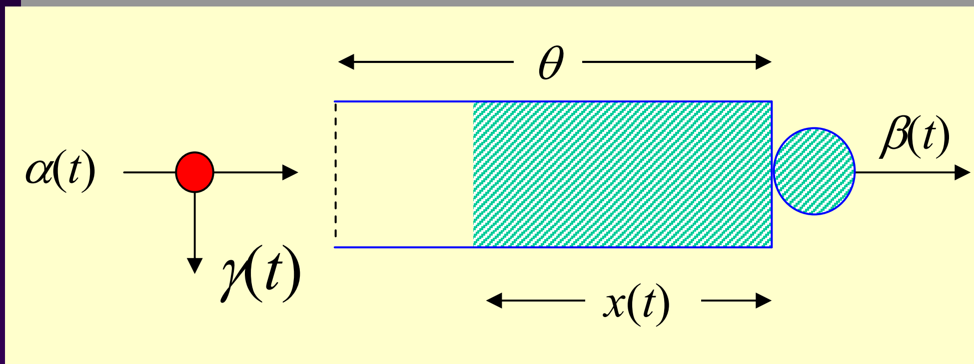
In many cases:

- *No need for a detailed model* (captured by f_k) to describe state behavior in between events

- This explains why *simple abstractions of a complex stochastic system* can be adequate to perform sensitivity analysis and optimization, as long as event times are accurately observed and local system behavior at these event times can also be measured.

- This is true in *abstractions of DES as HS* since:
  Common performance metrics (e.g., workload) satisfy THEOREM 1

THRESHOLD-BASED ADMISSION CONTROL



$$\frac{dx(t)}{dt} = \begin{cases} 0 & x(t)=0, \alpha(t) \le \beta(t) \\ 0 & x(t)=\theta, \alpha(t) \ge \beta(t) \\ \alpha(t)-\beta(t) & \text{otherwise} \end{cases}$$

$$J_T(\theta) = Q_T(\theta) + RL_T(\theta)$$

WORKLOAD

$$Q_T(\theta) = \int_0^T x(t,\theta)dt = \sum_{k\in\Omega} \int_{\tau_k}^{\tau_{k+1}} x(t,\theta)dt$$

$$\Omega = \{k : x_i(t) > 0 \text{ for all } t \in [\tau_k, \tau_{k+1}]\}$$

LOSS

$$L_T(\theta) = \sum_{k\in\Psi} \int_{\tau_k}^{\tau_{k+1}} [\alpha(t) - \beta(t)]dt$$

$$\Psi = \{k : x_i(t) = \theta_i \text{ for all } t \in [\tau_k, \tau_{k+1}]\}$$

Assume: $\{\alpha(t)\}$, $\{\beta(t)\}$ **piecewise continuously differentiable, independent of** θ

IPA FOR LOSS WITH RESPECT TO θ

$$L_T(\theta) = \sum_{k\in\Psi} \int_{\tau_k}^{\tau_{k+1}} \left[\alpha(t) - \beta(t)\right] dt$$

$$\frac{\partial L_T(\theta)}{\partial\theta} = \sum_{k\in\Psi} \left[\alpha(\tau_{k+1}^-) - \beta(\tau_{k+1}^-)\right]\tau'_{k+1} - \sum_{k\in\Psi} \left[\alpha(\tau_k^+) - \beta(\tau_k^+)\right]\tau'_k$$

Need to obtain EVENT TIME derivatives

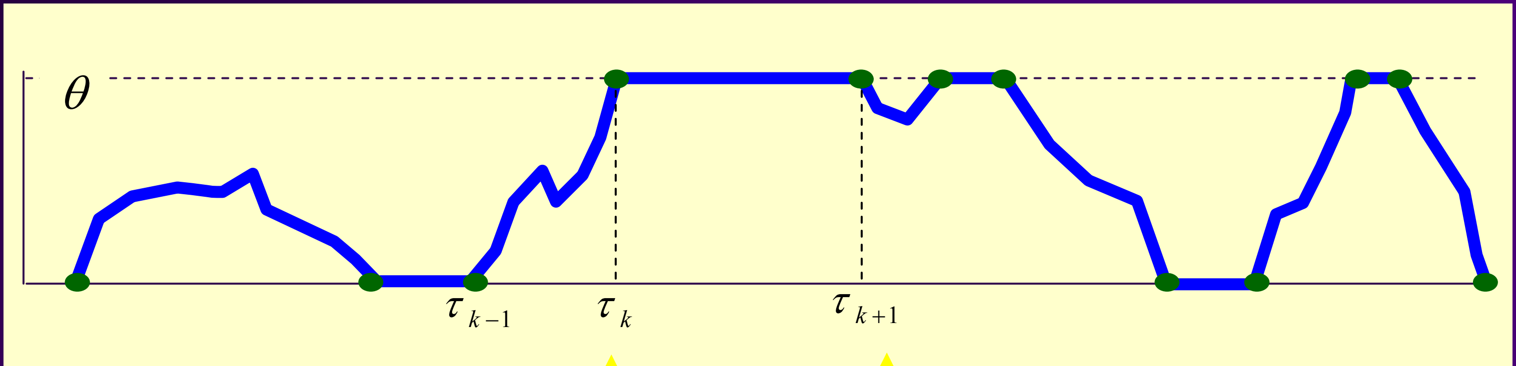
Apply:

1. $x'(\tau_k^+) = x'(\tau_k^-) + [f_{k-1}(\tau_k^-) - f_k(\tau_k^+)] \cdot \tau'_k$

2. $x'(t) = e^{\int_{\tau_k}^{t} \frac{\partial f_k(u)}{\partial x} du} \left[\int_{\tau_k}^{t} \frac{\partial f_k(v)}{\partial\theta} e^{-\int_{\tau_k}^{v} \frac{\partial f_k(u)}{\partial x} du} dv + x(\tau_k^+) \right]$

3. $\tau'_k = 0$ **or** $\tau'_k = -\left[\frac{\partial g}{\partial x} f_k(\tau_k^-)\right]^{-1} \left(\frac{\partial g}{\partial\theta} + \frac{\partial g}{\partial x} x'(\tau_k^-)\right)$

IPA WITH RESPECT TO θ



Endogenous event:

$$\tau_k' = -\left[\frac{\partial g}{\partial x} f_k(\tau_k^-)\right]^{-1}\left(\frac{\partial g}{\partial \theta} + \frac{\partial g}{\partial x} x'(\tau_k^-)\right)$$

Exogenous event:

$$\tau_{k+1}' = 0$$

Here:

$$g_k = x(\tau_k) - \theta \Rightarrow \tau_k' = \frac{1 - x'(\tau_k^-)}{\alpha(\tau_k) - \beta(\tau_k)}$$

$$x'(\tau_k^-) = e^{\int_{\tau_{k-1}}^{\tau_k} 0\,du}\left[\int_{\tau_{k-1}}^{\tau_k} 0 \cdot e^{-\int_{\tau_{k-1}}^{v} 0\,du} dv + 0\right] = 0$$

$$\frac{\partial L_T(\theta)}{\partial \theta} = -\sum_{k \in \Psi}\left[\alpha(\tau_k) - \beta(\tau_k)\right]\tau_k'$$

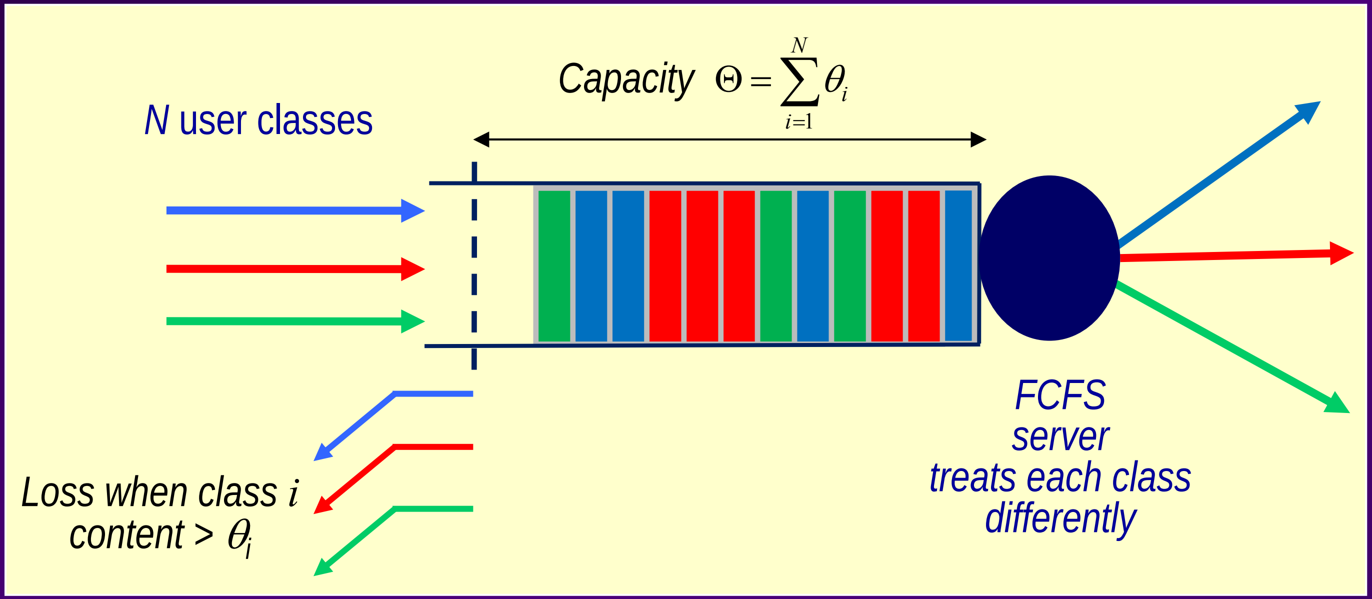
$$\boxed{\frac{\partial L_T(\theta)}{\partial \theta} = -|\Psi|}$$

Just count Overflow intervals

RESOURCE CONTENTION GAMES

MULTIPLE USERS COMPETE FOR RESOURCE

PROBLEM: Determine $(\theta_1,\ldots,\theta_N)$ to optimize system performance



SYSTEM-CENTRIC OPTIMIZATION:

System optimizes $J(\theta_1,\ldots,\theta_N)$ by controlling $(\theta_1,\ldots,\theta_N)$

vs

USER-CENTRIC OPTIMIZATION:

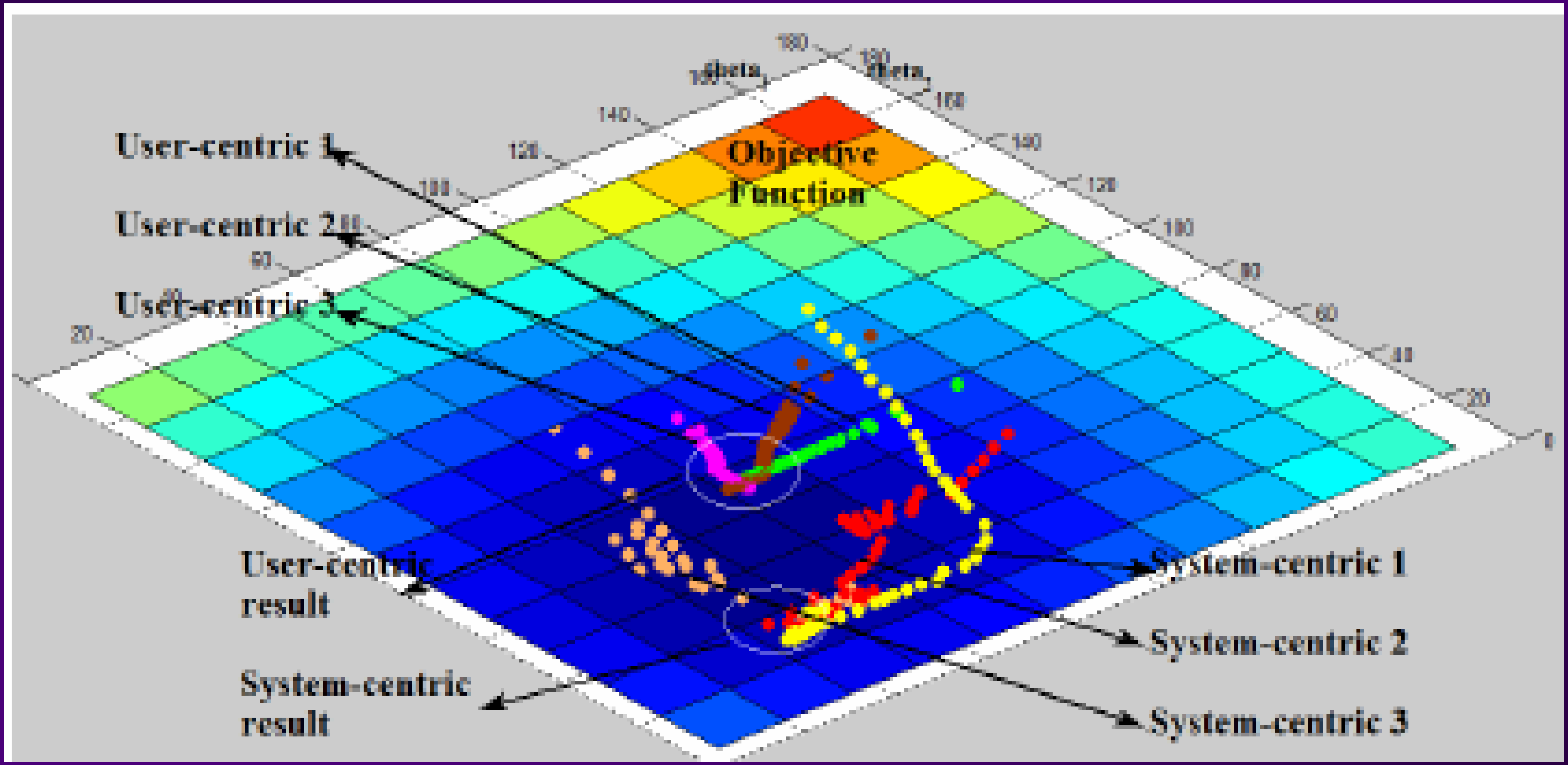
Each user optimizes $J_i(\theta_1,\ldots,\theta_N)$ by controlling *only* θ_i

$\Rightarrow$ ***Resource Contention Game***

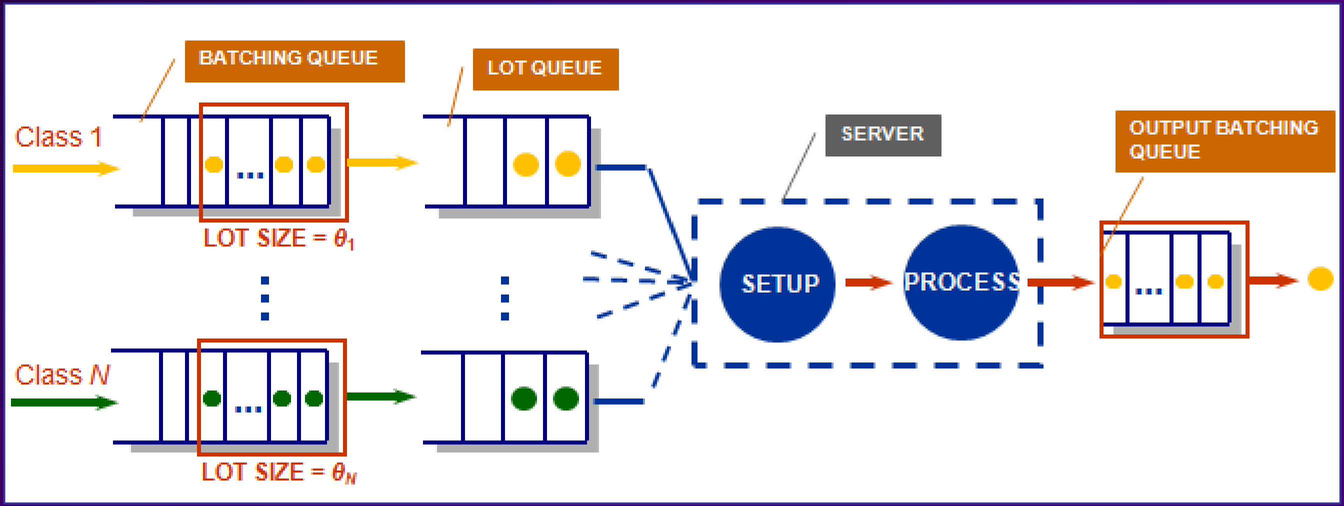
SYSTEM-CENTRIC v GAME SOLUTIONS

Generally, GAME solution is worse than SYSTEM-CENTRIC solution

→ “*the price of anarchy*”



THE OPTIMAL LOT-SIZING PROBLEM

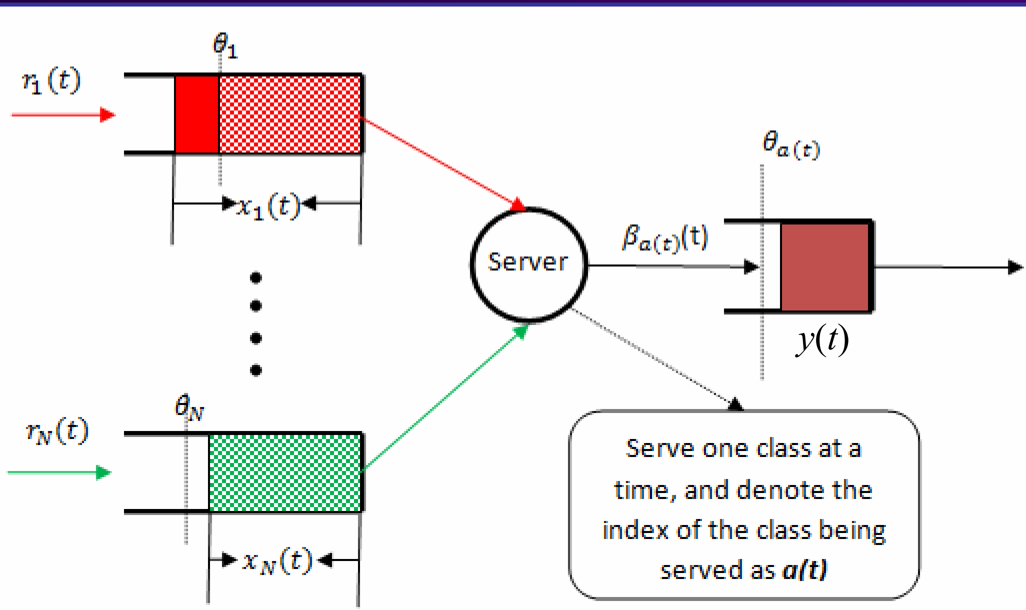


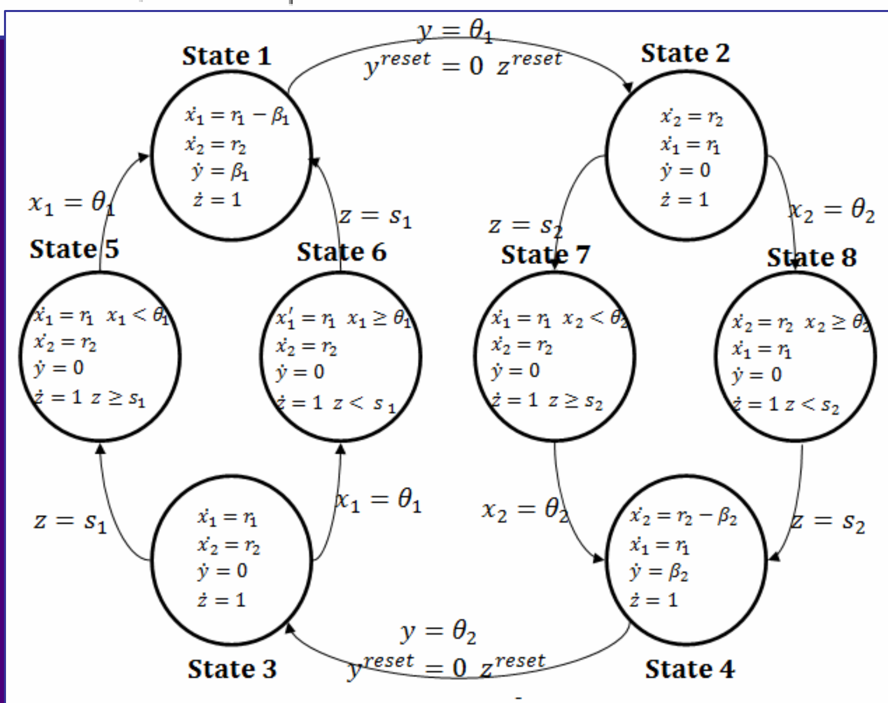
SYSTEM-CENTRIC OPTIMIZATION: Determine N lot size parameters to minimize overall MEAN DELAY

USER-CENTRIC OPTIMIZATION: Determine *ith* lot size parameter to minimize *ith* user MEAN DELAY

K. Baker, P.S. Dixon, M.J. Magazine, and E.A. Silver. *Management Science*, 1978.
P. Afentakis, B. Gavish, and U. Karmarkar. *Management Science*, 1984.
U.S. Karmarkar. *Management Science*, 1987.
J. Maes and L.V. Wassenhove. *Journal of Operations Research*, 1988.
G. Belvaux and L.A. Wolsey. *Management Science*, 2000.
N. Absi and S. Kedad-Sidhoum. *Operations Research*, 2007.

THE OPTIMAL LOT-SIZING PROBLEM





$$\frac{dx_i(t)}{dt} = \begin{cases} r_i(t) - \beta_i(t) & a(t) = i \text{ and } z(t) \geq s_{a(t)} \\ & \text{and } x_i(t) + y(t) \geq \theta_i(t) \\ r_i(t) & \text{otherwise} \end{cases}$$

$$\frac{dy(t)}{dt} = \begin{cases} \beta_{a(t)}(t) & y(t) < \theta_{a(t)} \text{ and } z(t) \geq s_{a(t)} \\ & \text{and } x_{a(t)}(t) + y(t) \geq \theta_{a(t)} \\ r_i(t) & \text{otherwise} \end{cases}$$

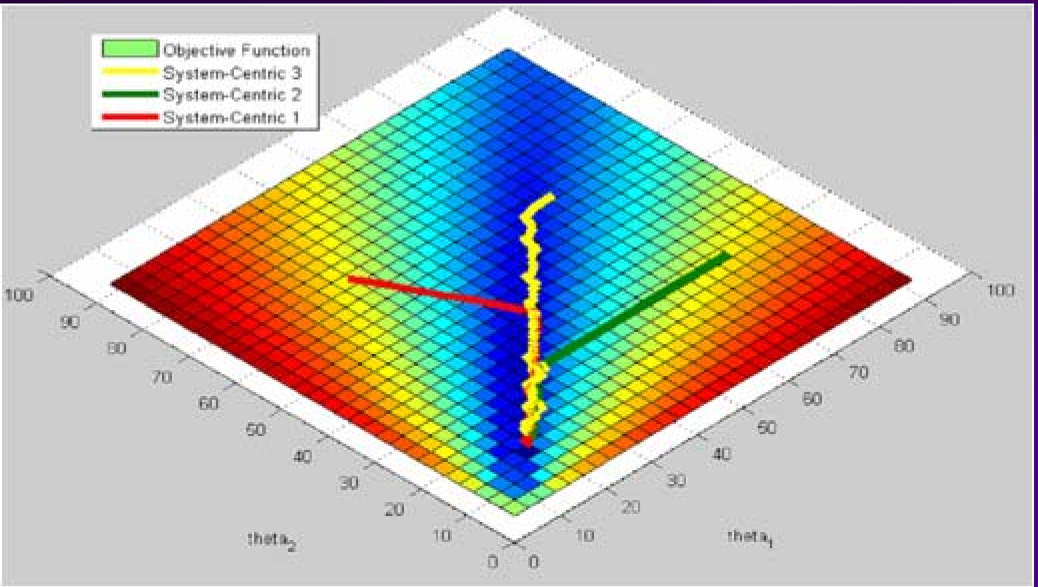
$$y(t) = 0 \text{ if } y(t^-) = \theta_{a(t^-)}$$

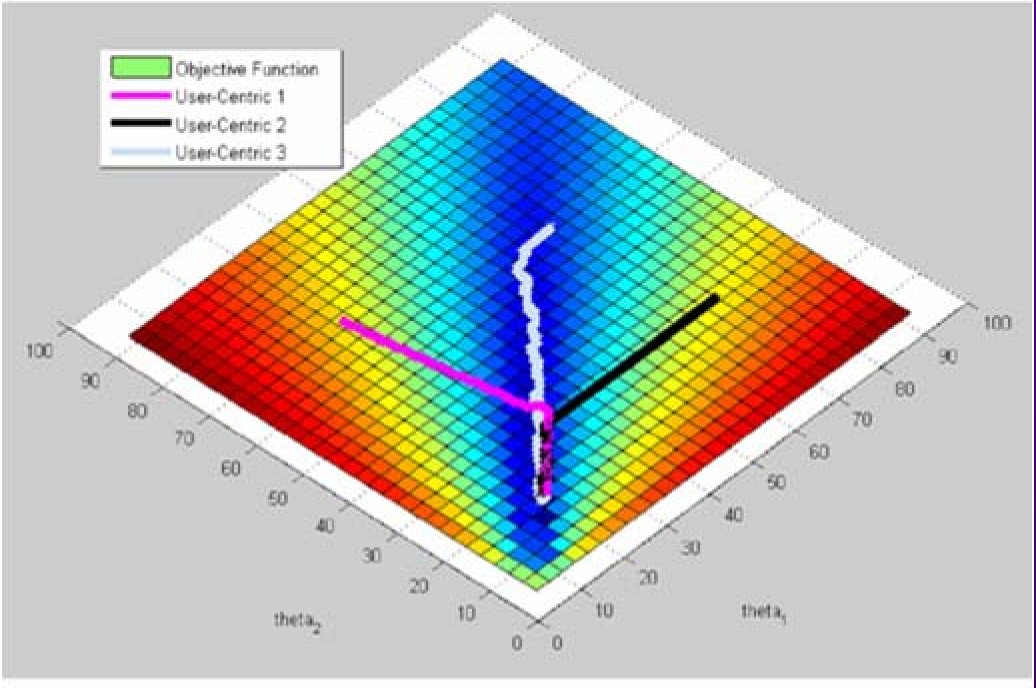
$$\frac{dz(t)}{dt} = 1 \text{ if } y(t) < \theta_{a(t)}$$

$$z(t) = 0 \text{ if } y(t^-) = \theta_{a(t^-)}$$

[*Yao and Cassandras, IEEE CDC, 2010; IEEE TASE, 2012*]

SYSTEM-CENTRIC v GAME SOLUTIONS





SYSTEM-CENTRIC solution…

…coincides with

GAME (USER-CENTRIC) solution!

→ *ZERO "price of anarchy"*

Proof obtained for DETERMINISTIC version

CYBER-PHYSICAL SYSTEMS: *THE NEXT FRONTIER*