# **Apuntes – Módulo 7**

## **Curso: Modeling and Simulation of Natural Processes**

**Universidad de Ginebra - Prof. Bastien Chopard et al.** **Semana 7 – Simulación de Eventos Discretos**

## **1. Introducción a eventos discretos**

### Concepto base - Ejemplo de la partícula rebotando

* Problema: Partícula en caja 2D (16x16m) con colisiones elásticas
* Enfoque ingenuo: Usar Δt pequeño → 18,000 iteraciones, error 4%
* Enfoque inteligente: Calcular tiempos exactos de colisión

**Cálculo de colisiones:**

t\_norte = (L - y(t))/v\_y

t\_este = (L - x(t))/v\_x

t\_sur = -y(t)/v\_y

t\_oeste = -x(t)/v\_x

Resultado obtenido: Solo 34 iteraciones, solución exacta, >500x más rápido

### **Ventajas del enfoque discreto**

* Precisión exacta vs aproximada
* Eficiencia computacional dramática
* Salto directo entre eventos importantes

## **2. Definición formal de simulación de eventos discretos**

### Requisitos fundamentales

1. **Cálculo analítico:** Poder determinar estado del sistema entre eventos
2. **Tiempo determinista:** Cada evento ocurre en momento específico t ∈ ℝ⁺
3. **Unicidad temporal:** Solo un evento por instante de tiempo

### Componentes del sistema

* **Entidades:** Cada componente i con estado s\_i(t)
* **Estado global:** S(t) = {s\_i(t)} en tiempo t
* **Eventos:** Cada evento j tiene acción asociada a\_j: S → S

### **Tipos de eventos**

* **Endógenos:** Generados por evolución interna del sistema
* **Exógenos:** Originados fuera del sistema (condiciones límite)

Nota importante a considerar: Los eventos exógenos se generan típicamente con distribuciones probabilísticas obtenidas empíricamente.

### **Algoritmo básico DES**

Inicialización:

- t\_actual = t0

- s\_i = s\_i(t\_actual)

Evolución:

mientras no condición\_final:

eventos = f(s\_i) # calcular próximos eventos

e\_siguiente = g(eventos) # elegir más cercano en tiempo

t\_siguiente = e\_siguiente.t

s\_i = e\_siguiente.accion(s\_i) # ejecutar acción

t\_actual = t\_siguiente # saltar al siguiente tiempo

## **3. Problemas de optimización - Oficina de correos**

### **Descripción del modelo**

Ventanillas: n ventanillas con estados {CERRADA, ABIERTA, OCUPADA} Clientes: Cada cliente j con duración de procesamiento p\_j Cola: Sistema FIFO para clientes en espera

### **Preguntas de optimización típicas**

1. ¿Cómo organizar horarios para minimizar longitud de cola?
2. ¿Cómo reducir ventanillas abiertas manteniendo umbral de espera?
3. ¿Conviene cola separada para solicitudes simples?

### **Variables de estado**

* Vector W: Estados de n ventanillas
* Cola C: FIFO de clientes esperando (representados por p\_j)

### **Eventos del sistema**

**Endógenos (ventanillas):**

* Abrir(t,i)
* Procesar(t,i)
* Cerrar(t,i)

**Exógenos (clientes):**

* Llegada(t,p\_j)

### **Consideraciones para optimización**

* Los eventos exógenos determinan comportamiento del sistema
* Necesidad de múltiples ejecuciones si eventos son aleatorios (método Monte Carlo)
* DES funciona como calculadora de función objetivo
* Se puede combinar con algoritmos de optimización (genéticos, recocido simulado, etc.)

## **4. Cuestiones de implementación**

### **Cola de prioridad**

**Elemento clave:** Estructura de datos que ordena eventos por tiempo de ocurrencia

* Todos los eventos futuros se insertan en cola Q ordenada por tiempo
* Cada acción puede leer/modificar esta cola: a\_j: (S × Q) → (S × Q)

### **Estructuras recomendadas**

* **Calendar queue:** Eficiente para distribuciones temporales específicas
* **Pairing heap:** Buen balance entre velocidad y simplicidad de implementación

### **Algoritmo refinado**

Inicialización:

- t\_actual = t0

- Q = exogenos() # agregar eventos exógenos

- S = estadoInicial(t0)

Evolución:

mientras no condición\_final:

e\_siguiente = Q.siguiente() # próximo evento en Q

t\_siguiente = e\_siguiente.tiempo

(S,Q) = e\_siguiente.accion(S,Q) # ejecutar acción

t\_actual = t\_siguiente

### **Paralelización**

Problema:Causalidad hace difícil la paralelización

Estrategias:

* Optimista
* Pesimista
* Descuidada (careless)

## **5. Ejemplo: Intersección de tráfico**

### **Modelo simplificado**

Escenario: Calle principal (este-oeste) + calle secundaria (sur)

Objetivo: Mantener semáforo verde para calle principal, cambiar solo cuando hay coches en secundaria

### **Variables de estado**

* **f:** Estado semáforo calle secundaria {ROJO, VERDE}
* **C:** Número de coches esperando en calle secundaria

### **Parámetros del modelo**

* **a:** Latencia para cambiar semáforo (rojo→verde o verde→rojo)
* **b:** Duración de luz verde por coche esperando

### **Eventos**

**Llegada de coche:**

si C == 0 y f == "ROJO":

insertar R2G(t + a) en cola

si f == "VERDE":

pasar directo

sino:

C = C + 1

**Cambio rojo→verde:**

f = "VERDE"

insertar G2R(t + C\*b) en cola

C = 0

**Cambio verde→rojo:**

f = "ROJO"

### **Ejemplo de evolución (a=30, b=10)**

| **Tiempo** | **Evento** | **C** | **f** | **Cola** |
| --- | --- | --- | --- | --- |
| 0 | - | 0 | R | CAR(10) CAR(25) CAR(35) |
| 10 | CAR | 1 | R | CAR(25) CAR(35) R2G(40) |
| 40 | R2G | 0 | G | CAR(60) G2R(70) |
| 70 | G2R | 0 | R | CAR(75) |

## **6. Ejemplo avanzado: Balística volcánica**

### **Contexto del problema**

Peligro real: Bombas volcánicas (trozos de magma) expulsadas con trayectorias parabólicas

Características: Siguen movimiento uniformemente acelerado, pueden colisionar entre sí

### **Ecuaciones base**

Movimiento: r(t) = ½gt² + v(0)t + r(0) Colisión**:** d²[B₁,B₂](https://claude.ai/chat/t) = (r₁(t) - r₂(t))² ≤ (R₁ + R₂)²

### **Variables de estado**

* Por cada bomba: (v, r, R, m) - velocidad, posición, radio, masa
* L\_A**:** Lista de bombas en el aire
* L\_D**:** Lista de bombas depositadas

### **Eventos del sistema**

1. ERUPCION(t, distribución): Genera nuevas bombas
2. COLISION(t, b1, b2)**:** Colisión entre dos bombas
3. GROUND(t, b1)**:** Bomba toca el suelo

### **Complejidad del modelo**

* Sin solución analítica**:** Imposible resolver matemáticamente
* Solución continua**:** Computacionalmente costosa
* DES**:** Candidato ideal para este tipo de problema

### **Consideraciones prácticas**

* Cada evento puede generar múltiples eventos futuros
* Necesidad de "limpiar" cola cuando bombas colisionan o se depositan
* Modelo permite estudiar patrones de deposición y riesgo

**Consideraciones generales del módulo:**

### **Cuándo usar DES**

* Sistemas donde eventos discretos causan cambios significativos
* Cuando se puede calcular estado analíticamente entre eventos
* Problemas de optimización en sistemas complejos

### **Limitaciones**

* Requiere cálculo analítico entre eventos
* Un solo evento por instante de tiempo
* Complejidad en implementación paralela

### **Aplicaciones prácticas**

* Sistemas de colas (oficinas, hospitales)
* Tráfico y logística
* Fenómenos naturales (volcanes, sismos)
* Redes de comunicación
* Manufactura y producción

Ventaja principal: Eficiencia computacional dramática vs métodos continuos tradicionales