

Especialización en sistemas embebidos (CESE)


Testing de software embebido Clase 5




Clase 5

- Técnicas de diseño de Test:
 - State transition testing (STT)
 - Evolutionary algorithms (EA)

ing. Alejandro Permingeat



REPASO



¿tipo de prueba
vs
nivel de prueba?

Tipos de prueba vs nivel prueba

Tipos de prueba



qué se va a probar



características de calidad
(funcionalidad, interfaces,
rendimiento . . .)

Nivel de prueba



quién lo va a probar
y cuándo



nivel de detalle.
(pruebas unitarias, de
sistema, de aceptación)

Test type	Description	Quality characteristics included
Functionality	Testing functional behavior (includes dealing with input errors)	Functionality
Interfaces	Testing interaction with other systems	Connectivity
Load and stress	Allowing large quantities and numbers to be processed	Continuity, performance
Support (manual)	Providing the expected support in the system's intended environment (such as matching with the user manual procedures)	Suitability
Production	Test production procedures	Operability, continuity
Recovery	Testing recovery and restart facilities	Recoverability
Regression	Testing whether all components function correctly after the system has been changed	All
Security	Testing security	Security
Standards	Testing compliance to standards	Security, user-friendliness
Resources	Measuring the required amount of resources (memory, data communication, power, ...)	Efficiency

Test level	Level	Environment	Purpose
Hardware unit test	Low	Laboratory	Testing the behavior of hardware component in isolation
Hardware integration test	Low	Laboratory	Testing hardware connections and protocols
Model in the loop	High/low	Simulation models	Proof of concept; testing control laws; design optimization
Software unit test, host/target test	Low	Laboratory, host + target processor	Testing the behavior of software components in isolation
Software integration test	Low	Laboratory, host + target processor	Testing interaction between software components
Hardware/software integration test	High	Laboratory, target processor	Testing interaction between hardware and software components
System test	High	Simulated real life	Testing that the system works as specified
Acceptance test	High	Simulated real life	Testing that the system fulfills its purpose for the user/customer
Field test	High	Real life	Testing that the system keeps working under real-life conditions.

Técnicas diseño casos de prueba

- State transition testing (STT)
- Control flow test (CFT)
- Elementary comparison test (ECT)
- Classification-tree method (CTM)
- Evolutionary algorithms (EA)
- Statistical usage testing (SUT)
- Rare event testing (RET)
- Mutation analysis (MA)

Técnicas diseño casos de prueba

State transition testing (STT)

- Al probar funcionalidad (comportamiento) que depende de estados, se debe verificar que en respuesta a ciertos eventos de entrada:
 - se ejecutan las acciones correctas
 - el sistema cambia al estado correcto
- La técnica que se describe a continuación se aplica sobre diagramas de estados planos (es decir, no es aplicable para diagramas de estados jerárquicos).

Técnicas diseño casos de prueba

State transition testing (STT)

Pasos:

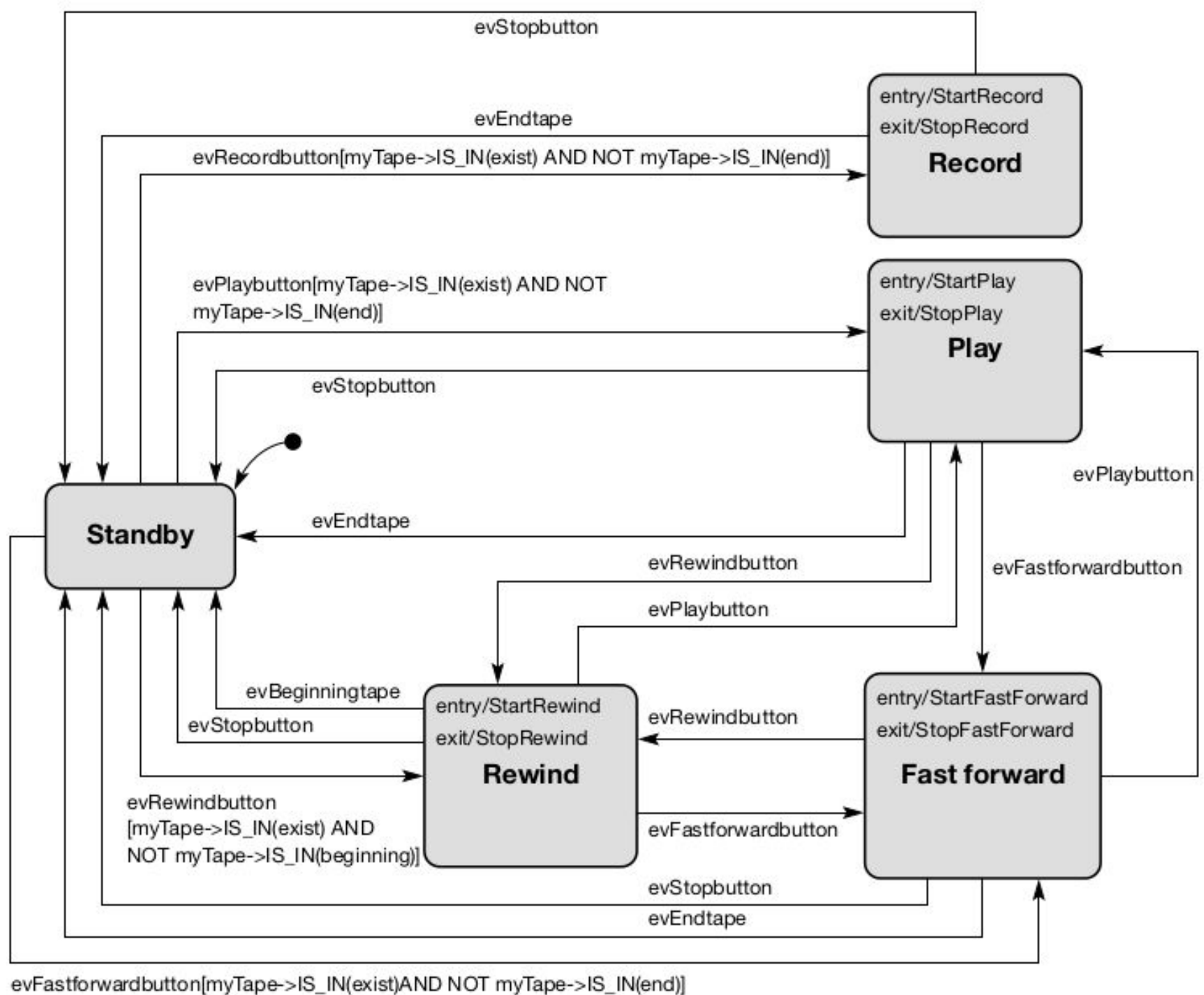
1. Construir tabla de estados/eventos
2. Construir el árbol de transiciones
3. Construir el test script de casos de prueba legales.
4. Construir el test script de casos de prueba ilegales.
5. Construir el test script de guardas.

Técnicas diseño casos de prueba

State transition testing (STT)

Para aplicar esta técnica de diseño es muy importante que se cuente con una **máquina de estados** (statechart) **completamente definida**

Ejemplo: Sistema de control para una video-casetera (VCR)



Técnicas diseño casos de prueba

State transition testing (STT)

1 - Construir tabla de estados/eventos

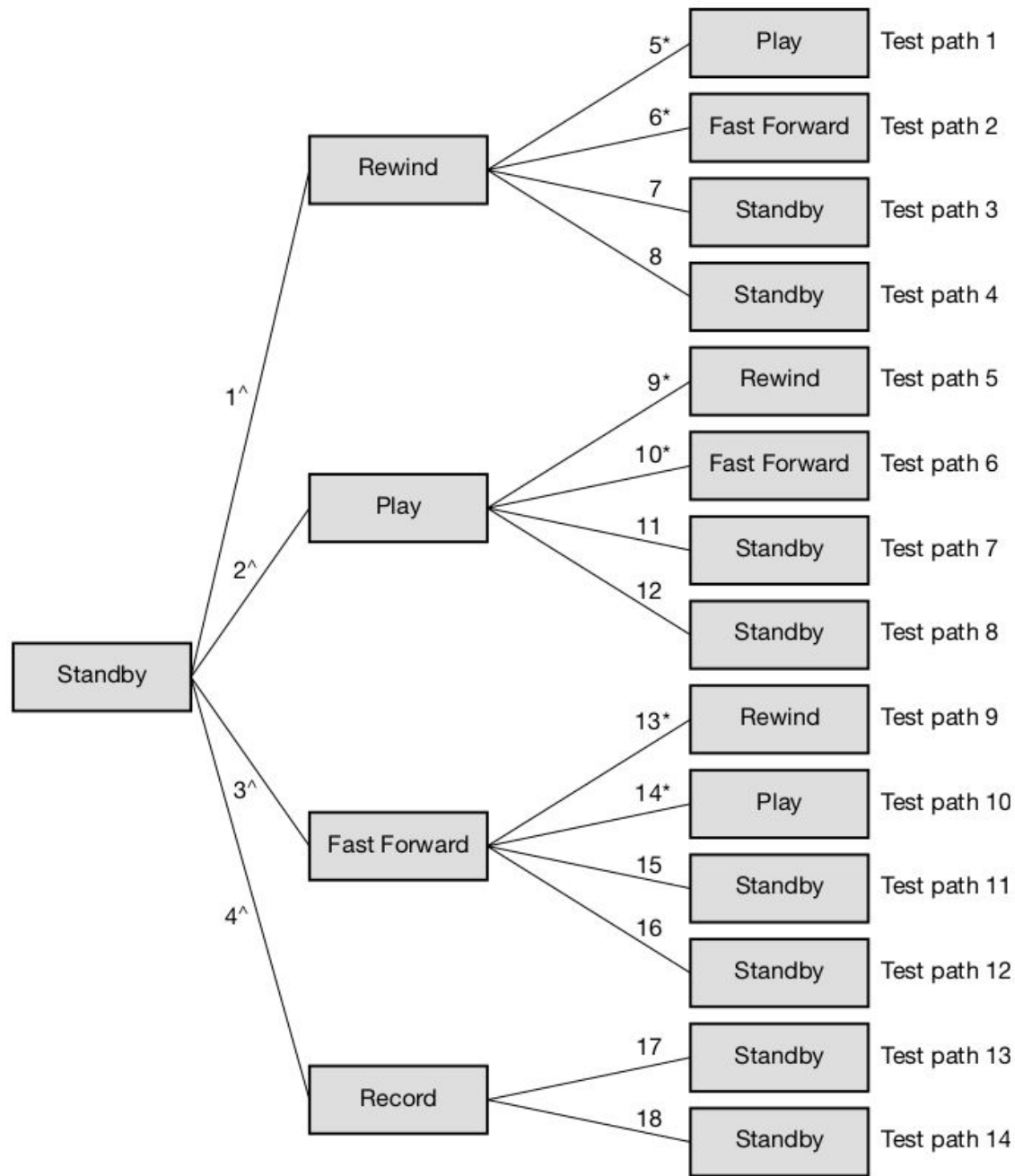
	Standby	Rewind	Play	Fast forward	Record
evRewindbutton	1-Rewind	●	9-Rewind	13-Rewind	●
evPlaybutton	2-Play	5-Play	●	14-Play	●
evFastforwardbutton	3-Fast forward	6-Fast forward	10-Fast forward	●	●
evRecord	4-Record	●	●	●	●
evStopbutton	●	7-Standby	11-Standby	15-Standby	17-Standby
evEndtape	●	●	12-Standby	16-Standby	18-Standby
evBeginningtape	●	8-Standby	●	●	●
● Illegal combinations (sneak path)					

Técnicas diseño casos de prueba

State transition testing (STT)

2 - Construir el árbol de transiciones

- Se comienza por el estado inicial (raíz árbol)
- De allí todas las transiciones de salida y estados a los que lleva son agregados al árbol
- Se respeta numeración de la tabla estados/eventos
- De los estados alcanzados, se agrega al árbol el siguiente nivel de transiciones y estados
- Se repite lo anterior hasta que:
 - Cada camino llegue a un estado final
 - El estado inicial es alcanzado
 - Se alcance un estado ya presente en el árbol



* :Provisional terminal point

^ :Guarded

Técnicas diseño casos de prueba

State transition testing (STT)

3 - Construir el test script de casos de prueba legales.

- Teniendo como guía a la tabla de estados/eventos y el árbol de transiciones se crean los scripts de pruebas que solo cubren **casos de prueba legales**
- Cada script debe finalizar en el estado inicial o en el estado final de la máquina de estados.
- Cada línea debe contener:
 - el evento
 - la guarda
 - la acción esperada
 - el estado resultado

Input			Expected result	
ID	Event	Guard	Action	State
L1.1	evRewindbutton	Tape exists, tape not at the beginning	StartRewind	Rewind
L1.2	evPlaybutton		StopRewind; StartPlay	Play
L1.3	evStopbutton		StopPlay	Standby
L2.1	EvRewindbutton	Tape exists, tape not at the beginning	StartRewind	Rewind
L2.2	evFastforwardbutton		StopRewind; StartFastForward	Fast forward
L2.3	evStopbutton		StopFastForward	Standby
L3.1	EvRewind	Tape exists, tape not at the beginning	StartRewind	Rewind
L3.2	EvStopbutton		StopRewind	Standby
L4.1	EvRewind	Tape exists, tape not at the beginning	StartRewind	Rewind
L4.2	EvBeginningtape		StopRewind	Standby
L5.1	EvPlay	Tape exists, tape not at the end	StartPlay	Play
L5.2	evRewind		StopPlay; StartRewind	Rewind
L5.3	evStopbutton		StopRewind	Standby
L6.1	evPlaybutton	Tape exists, tape not at the end	StartPlay	Play
L6.2	evFastforwardbutton		StopPlay; StartFastForward	Fast forward
L6.3	evStopbutton		StopFastForward	Standby

L7.1	evPlaybutton	Tape exists, tape not at the end	StartPlay	Play
L7.2	evStopbutton		StopPlay	Standby
L8.1	evPlaybutton	Tape exists, tape not at the end	StartPlay	Play
L8.2	evEndtape		StopPlay	Standby
L9.1	evFastforwardbutton	Tape exists, tape not at the end	StartFastForward	Fast forward
L9.2	evRewind		StopFastForward; StartRewind	Rewind
L9.3	evStopbutton		StopRewind	Standby
L10.1	evFastforwardbutton	Tape exists, tape not at the end	StartFastForward	Fast forward
L10.2	evPlaybutton		StopFastForward; StartPlay	Play
L10.3	evStopbutton		StopPlay	Standby
L11.1	evFastforwardbutton	Tape exists, tape not at the end	StartFastForward	Fast forward
L11.2	evStopbutton		StopFastForward	Standby
L12.1	evFastforwardbutton	Tape exists, tape not at the end	StartFastForward	Fast forward
L12.2	evEndtape		StopFastForward	Standby
L13.1	evRecord	Tape exists, tape not at the end	StartRecord	Record
L13.2	evStopbutton		StopRecord	Standby
L14.1	evRecordbutton	Tape exists, tape not at the end	StartRecord	Record
L14.2	evEndtape		StopRecord	Standby

Técnicas diseño casos de prueba

State transition testing (STT)

- 4 - Construir el test script de casos de prueba ilegales.
 - Las combinaciones estado/evento ilegales se pueden obtener desde la tabla de estado/eventos
 - Una combinación ilegal es aquella en la cual el software no debe reaccionar ante un evento
 - El resultado esperado para un casos de prueba ilegal es que el sistema no responda y se quede en el mismo estado en el que se encontraba antes de dicho evento.

ID	Setup	State	Event	Result
I1		Standby	evStopbutton	
I2		Standby	evEndtape	
I3		Standby	evBeginningtape	
I4	L1.1	Rewind	evRewind	
I5	L1.1	Rewind	evRecord	
I6	L1.1	Rewind	evEndtape	
I7	L5.1	Play	evPlay	
I8	L5.1	Play	evPlay	
I9	L5.1	Play	evBeginningtape	
I10	L9.1	Fast forward	evFastforwardbutton	
I11	L9.1	Fast forward	evRecordbutton	
I12	L9.1	Fast forward	evBeginningtape	
I13	L13.1	Record	evRewindbutton	
I14	L13.1	Record	evFastforwardbutton	
I15	L13.1	Record	evRecordbutton	
I16	L13.1	Record	evBeginningtape	

Técnicas diseño casos de prueba

State transition testing (STT)

5 - Construir el test script de guardas.

- Si las guardas consisten en condiciones de “valores de borde”, se crea un caso de prueba para los valores por debajo y otro por encima de la condición de borde.
- Si las guardas consisten en condiciones complejas, se emplea la técnica Elementary Comparison Test (ECT)

Técnicas diseño casos de prueba

Evolutionary algorithms (EA)

- También conocidos como **algoritmos genéticos**.
- Teoría de **Darwin**: “Supervivencia del más apto”
- Algoritmos utilizados para problemas de optimización
- *Ejemplo*: muy utilizados para encontrar casos de prueba en los cuales el sistema viola sus restricciones de tiempo

Técnicas diseño casos de prueba

Evolutionary algorithms (EA)

**Técnicas de diseño
convencionales**



foco en
*casos de pruebas
individuales*

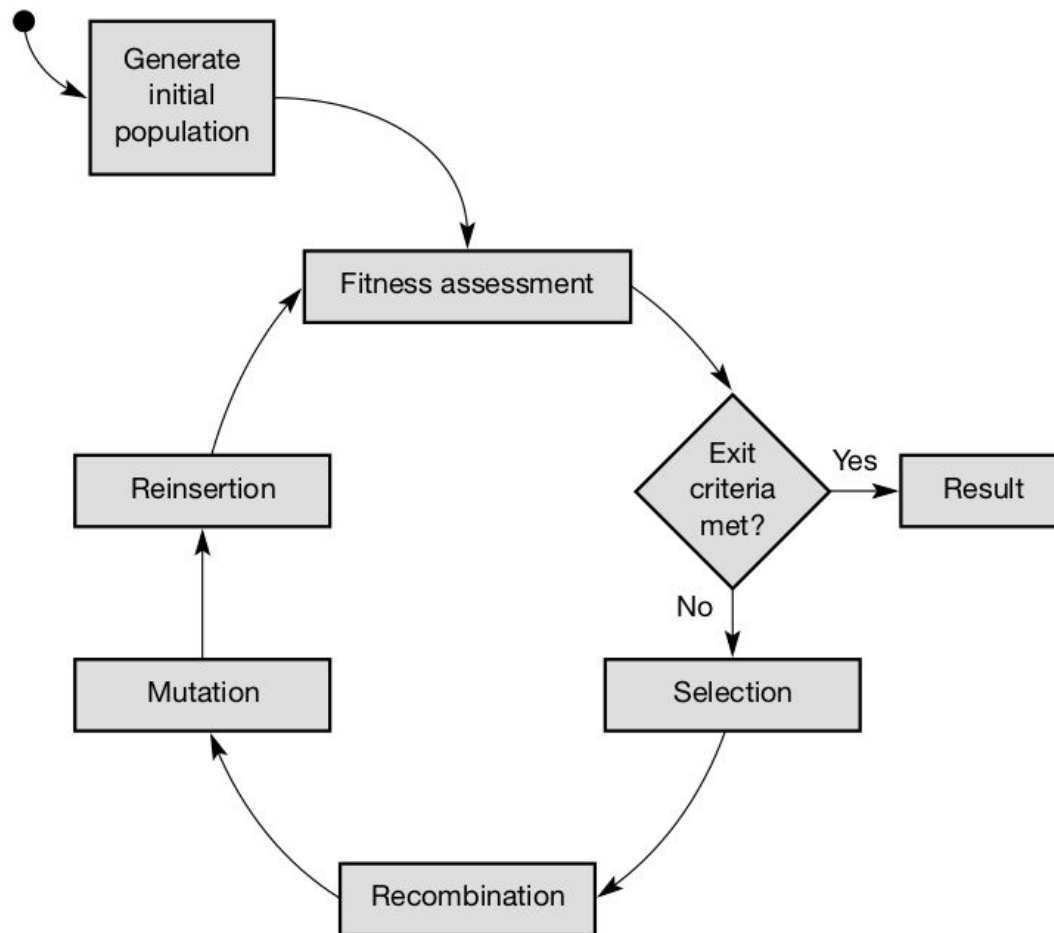
**Algoritmos
evolutivos**



involucra
“población” y “aptitud”
de *casos de pruebas
individuales*

Técnicas diseño casos de prueba

Evolutionary algorithms (EA)



Técnicas diseño casos de prueba

Evolutionary algorithms (EA)

Población inicial


- Cada caso de prueba es un individuo de la población
- La población inicial es elegida aleatoriamente
- Típicamente entre 50 y 100 miembros

Técnicas diseño casos de prueba

Evolutionary algorithms (EA)

Población inicial

Ejemplo: Un sistema embebido con 4 ADC que debe calcular una señal de salida en menos de 100 ns.



Individual	P_1	Valores de ADC generados			
1	41 03 90 93				
2	04 72 95 31				
3	44 31 45 93				
4	93 45 32 91				

Técnicas diseño casos de prueba

Evolutionary algorithms (EA)

Evaluación de la aptitud

- La aptitud expresa con bueno es el caso de prueba en relación al objetivo óptimo
- La aptitud es calculada para cada caso de prueba (función de aptitud)
- Ejemplo: Si el objetivo es encontrar casos de prueba para estresar el sistema, la aptitud podría ser el tiempo que tardó el sistema en responder ante la ejecución del caso de prueba

Técnicas diseño casos de prueba

Evolutionary algorithms (EA)

Evaluación de la aptitud

Tiempo de
respuesta en ns

Individual	P_1	fi
1	41 03 90 93	13
2	04 72 95 31	6
3	44 31 45 93	31
4	93 45 32 91	8
Total fitness		58

Técnicas diseño casos de prueba

Evolutionary algorithms (EA)

Selección

- Se seleccionan casos de prueba para formar nuevos casos de prueba (padres de la nueva generación)
- Puede ser
 - aleatoria
 - teniendo en cuenta la “aptitud” de los casos de prueba

Técnicas diseño casos de prueba

Evolutionary algorithms (EA)

Selección

Individual	P_1	fi
1	41 03 90 93	13
2	04 72 95 31	6
3	44 31 45 93	31
4	93 45 32 91	8
Total fitness		58

Técnicas diseño casos de prueba

Evolutionary algorithms (EA)

Recombinación

- Dos casos de pruebas (padres) se recombinan para producir dos casos de prueba
- Se pueden realizar los siguientes “entrecruzamientos”:
 - simples (padres partidos en un solo lugar)
 - dobles (padres partidos en dos lugares)
 - uniformes (cantidad de partes aleatorias)

Técnicas diseño casos de prueba

Evolutionary algorithms (EA)

Recombinación

				<i>f</i>
44	31	45	93	
41	03	90	93	
31	85	32	91	
44	31	45	93	
44	31	90	93	22
41	03	45	93	22
31	85	32	91	17
44	31	45	93	22
				83

Recombinación simple

Técnicas diseño casos de prueba

Evolutionary algorithms (EA)

Mutación

- Es un cambio aleatorio de un elemento dentro de un caso de prueba

44 31 90 93	0.637 –	44 31 89 93
41 03 45 93	0.352 +	41 04 45 93
31 85 32 93	0.712 +	31 85 33 93
44 31 45 91	0.998 –	44 31 45 90

Técnicas diseño casos de prueba

Evolutionary algorithms (EA)

Reinserción

- No todos los nuevos casos de prueba generados serán parte de la “nueva generación” de casos de prueba
- La “supervivencia” del nuevo caso de prueba dependerá en su aptitud y en la tasa de reinserción
- La **tasa de reinserción** es la probabilidad de supervivencia de la descendencia

Nuevos individuos

Individual	O_1	f_i
1	44 31 89 93	22
2	41 04 45 93	22
3	31 85 33 93	17
4	44 31 45 90	22
Total fitness		83

Reinserción

Ej. con tasa de supervivencia de 25%, sobrevive solo 1 nuevo individuo elegido aleatoriamente entre los más aptos

Nueva población

Individual	P_2	f_i
1	41 03 90 93	13
2	44 31 45 90	22
3	44 31 45 93	31
4	93 45 32 91	8
Total fitness		74

Población anterior

Individual	P_1	f_i
1	41 03 90 93	13
2	04 72 95 31	6
3	44 31 45 93	31
4	93 45 32 91	8
Total fitness		58

De la población anterior no sobrevive el/los menos aptos para darle lugar a los “recién nacidos”

Bibliografía

- “Testing Embedded Software”. Autores: Bart Broekman y Edwin Notenboom. Año 2003.

Especialización en sistemas embebidos (CESE)

Testing de software embebido Clase 5

