

Corso di Laurea in Ingegneria Informatica

Corso di Ingegneria del Software

Test Combinatoriale

Sommario

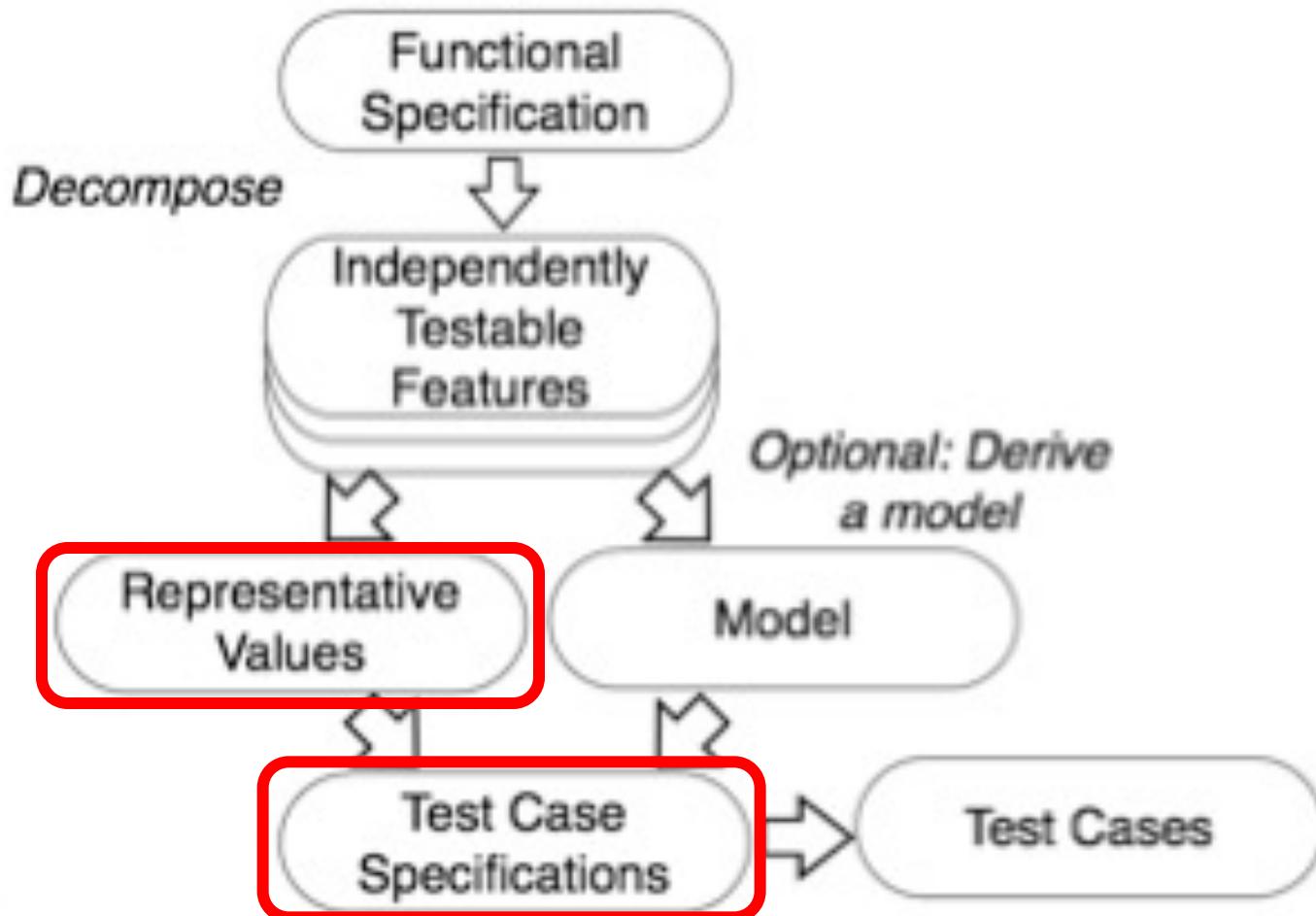
- Test Combinatoriale
 - Approcci: Category-Partition testing e Pairwise combinatorial testing
- Category partition Testing
 - Parametri
 - Categorie
 - Vincoli
- Pairwise combinatorial testing

Riferimenti:

M.Pezzè, M. Young; Software Testing and Analysis. Process, principles, and techniques.

Cap. 11 § § 11.1, 11.2, 11.3

Dalla specifica ai casi di test



Pezzè & Young, Software Testing And Analysis: Process, Principles And Techniques, 2008

Approcci al test funzionale

- ❖ Data una specifica ci sono più approcci per derivare i casi di test
- ❖ Due approcci per derivare delle **specifiche di casi di test** (da cui si generano i test case) sono le tecniche di tipo combinatoriale (combinatorial testing):
 - ❖ **Category-partition testing**
 - ❖ **Pairwise testing**

Test Combinatoriale

- ★ **Test Combinatoriale (Combinatorial Testing):** il test combinatoriale consiste nella strutturazione (manuale) delle specifiche di test in un insieme di proprietà o attributi che possono essere sistematicamente variate in insiemi di valori. **Vantaggi:**
 - I test sono automatizzabili
 - I test sono caratterizzati con attributi che possono essere combinati fra loro per ottenere delle combinazioni “rappresentative” per il test di una funzionalità.
 - Lo spazio dei possibili valori dei singoli attributi può essere ridotto ad un subset.

Test Combinatoriale: Idea di Base

- ★ **Identificazione di attributi distinti**
 - ★ Nell'insieme dei dati, nel sistema, nella configurazione
 - ★ Esempio: il browser può essere IE o Firefox, il sistema operativo Vista, XP, o Linux, ...
- ★ **Generazione delle combinazioni degli attributi per il testing**
 - ★ Combinazione 1: IE su Vista, Firefox su XP, Firefox su Linux
 - ★ Combinazione 2: IE su XP, Firefox su Vista, Firefox su Linux

Approcci per il combinatorial testing

- ★ **Category partition testing:** identificazione manuale di attributi che caratterizzano lo spazio di input. Gli attributi variano all'interno di un set di valori. Applicazione di vincoli per ridurre il numero dei casi di test
- ★ **Pairwise Testing:** identificazione di attributi che caratterizzano lo spazio di input. Gli input variano su un set di valori opportunamente ridotto, considerando **copie** dei valori degli attributi (a due a due). Si possono considerare anche combinazioni n-arie per $n > 2$ (a tre a tre, a quattro a quattro, ecc.).

Category Partition

1. Decomposizione delle funzionalità che possono essere testate indipendentemente

- Per ogni funzionalità si individuano:
 - I parametri che la descrivono e gli elementi da cui dipende
 - Per ogni parametro ed elemento si individuano un insieme di caratteristiche elementari, dette **categorie**

2. Identificazione di valori rappresentativi

- Per ogni categoria si identifica classi di valori rappresentativi:
 - *Normal values*
 - *Special Values*
 - *Boundary Values*
 - *Error Values*

3. Introduzione di vincoli sui valori individuati

- Si impone che le combinazioni possano avere al più un errore
- Si valuta se i valori sono compatibilmente combinati (nell'esempio precedente non si può avere IE su Linux)

Esempio: Controllo configurazione

- ❖ Si consideri un sito web per la vendita di computer online. Il sistema ha diverse funzionalità quali: registrazione dei clienti, acquisto di singoli componenti per il computer, visualizzazione delle offerte, controllo della configurazione del computer che si sta ordinando
- ❖ Si testa con approccio **category partition** la funzionalità “controllo della configurazione” (che può essere testata indipendentemente dalle altre). Questa funzionalità permette di controllare se la configurazione del computer è valida. Ad esempio, se il sistema che si sta comprando ha una scheda madre su cui si può installare un processore *dual core* oppure si è obbligati a usare un processore *quad core*.
 - I parametri che si possono individuare sono:
 - **Modello**
 - **Insieme dei componenti**
 - L'elemento di sistema è il **database** in cui sono memorizzati i componenti e i modelli disponibili

Scelta dei parametri (1/2)

Il parametro **modello**

Il modello identifica uno specifico prodotto e determina un insieme di vincoli sui componenti usati per l'assemblaggio. I modelli sono caratterizzati da slot logici che possono essere occupati da alcuni componenti. Gli slot logici possono essere implementati anche in hardware. Alcuni slot possono essere lasciati anche vuoti.

- Il modello *Chipmunk C20* ha come slot logici lo schermo, il processore, l'hard disk, la memoria e il sistema operativo (di questi slot logici solo hard disk e memoria sono implementati con slot fisici)

Scelta dei parametri (2/2)

Il parametro **insieme dei componenti**

Una coppia (slot,componente) indica lo slot e il componente che lo occupa. La scelta del componente da inserire è vincolata al modello che si desidera assemblare e alle incompatibilità che possono eventualmente, esistere tra i componenti.

- Il modello *Chipmunk C20* può montare nello slot hard disk il componente hard disk da 20GB (default) oppure si possono installare anche hard disk da 30 e 40 GB. La scelta di lasciare lo slot logico hard disk vuoto non è possibile. Il sistema operativo di default è RodentOS 3.2 personal edition, ma si può anche scegliere RodentOS3.2 mobile edition che richiede un hard disk da 30 GB.
 - Si noti che lo slot logico hard disk non può essere vuoto (**vincolo del modello**) e che se si usa il componente RodentOS 3.2 mobile edition si deve usare un hard disk da 30 GB (**vincolo fra componenti**)

Elemento del sistema

⊕ L'elemento database dei prodotti

Il database dei prodotti è un elemento di sistema che non è menzionato nelle specifiche funzionali, ma è richiesto per l'esecuzione dei test (in quanto influisce sulla specifica funzionale dei test*).

Nota: non si vuole testare la funzionalità con uno specifico set di valori inseriti nel database, ma testare il sistema con un database i cui valori possono variare.

Identificazione delle categorie

- ★ **Le categorie rappresentano delle proprietà del parametro.**
- ★ Per il parametro modello:
 - Numero del modello
 - Numero di slot logici richiesti
 - Numero di slot logici opzionali
- Per il parametro insieme dei componenti:
 - Corrispondenza del componente selezionato con gli slot del modello
 - Numero di componenti richiesti
 - I componenti richiesti selezionati
 - Numero di componenti opzionali
 - Componenti opzionali selezionati
- Elementi di sistema:
 - Numero di modelli nel database
 - Numero di componenti nel database

La scelta delle categorie è un'attività che richiede esperienza e la conoscenza del dominio. In generale non esistono regole per la definizione di categorie.

Identificazione classi di valori

- ❖ Per ogni categoria si individua una classe di valori che la rappresenta
- ❖ **Boundary value:**
 - Valori che superano di “poco” il *bound*
 - Valori interni al *bound*
 - Valori interni in prossimità del *bound*
- ❖ **Erroneous value:**
 - Valori che sono esterni al dominio della categoria

Identificazione valori: Modello

⊕ Numero del modello:

- Malformato
- Non presente nel database
- Valido

⊕ Numero di slot logici richiesti per il modello

- 0
- 1
- Molti

⊕ Numero di slot logici opzionali per il modello

- 0
- 1
- Molti

Identificazione valori: Insieme dei Componenti

- ★ **Corrispondenza del componente selezionato con gli slot del modello**
 - Slot omesso
 - Extra slot
 - Slot incompatibile con il componente
 - Completa corrispondenza
- ★ **Numero di componenti richiesti**
 - 0
 - < numero slot richiesti
 - = numero slot richiesti
- ★ **Componenti richiesti selezionati**
 - Alcuni sono default
 - Tutti validi
 - Incompatibile con gli slot
 - Incompatibile con un altro componente scelto
 - Incompatibile con il modello
 - Non nel database
- ★ **Componenti opzionali selezionati**
 - Alcuni di default
 - Tutti validi
 - Incompatibile con gli slot
 - Incompatibile con un altro componente scelto
 - Incompatibile con il modello
 - Non nel database
- ★ **Numero di componenti opzionali**
 - 0
 - < numero slot opzionali
 - = numero slot opzionali

Identificazione valori: Database

- ★ **Numero di modelli nel database:**
 - 0
 - 1
 - Molti

- ★ **Numero di componenti nel database:**
 - 0
 - 1
 - Molti

- ★ **Nota:** 0 e 1 sono in genere valori insoliti (valori speciali). Possono causare un' insolita combinazione.

Introduzione dei vincoli

- ❖ Il test combinatoriale può essere eseguito su tutte le combinazioni di valori delle categorie. In questo caso il numero di test case può essere notevole (è il prodotto cartesiano dei valori delle categorie).
 - Nell'esempio il numero di test case è pari al prodotto della cardinalità delle singole categorie, 314.298 test case.
 - 7 cat. con 3 classi di valori, 2 cat. con 6 classi, una cat. con 4 val.
 - $3^7 \times 6^2 \times 4 = 314.928$
 - Alcune combinazioni non sono compatibili, infatti non ha senso una combinazione in cui compaiono “zero slot” e “componente incompatibile” (se non ci sono slot non può essere inserito nessun componente).
- ❖ Si possono introdurre dei vincoli che riducono notevolmente il numero di test case
 - **Vincolo error**
 - **Vincolo single**
 - **Vincolo property**

Vincolo: errore

- ❖ Il vincolo **errore** [error] indica una classe di valori che dà luogo ad un errore.
 - Si può ipotizzare che sia sufficiente un solo errore per combinazione per osservare un potenziale fallimento del sistema.

- ❖ **Numero del modello:**

- Malformato [error]
- Non presente nel database [error]
- Valido

- ❖ **Corrispondenza del componente selezionato con gli slot del modello**

- Slot omesso [error]
- Extra slot [error]
- Slod incompatibile con il componente [error]
- Completa corrispondenza

- ❖ **Numero di componenti richiesti**

- 0 [error]
- < numero slot richiesti [error]

Il vincolo errore riduce
il numero di test a
2.711

- ❖ **Numero di modelli nel database:**

- 0 [error]

- ❖ **Numero di componenti nel database:**

- 0 [error]

Vincolo: proprietà (1/2)

- ★ Il vincolo **proprietà** [property] [if-property] elimina le combinazioni invalide dei valori delle categorie
 - [property] raggruppa i valori di un componente che hanno una proprietà in comune
 - [if-property] il valore può essere in combinazione solo con valori di altre categorie che hanno la label [property]
 - Ad esempio il valore “ **= numero slot richiesti** ” della categoria di Numero di componenti richiesti ha senso se il numero di slot logici richiesti per il modello vale “**molti**”
 - Per introdurre questo vincolo si assegna un nome alla proprietà, ad esempio RSMANY e la si associa con una label al valore “molti” mentre si assegna la label [if RSMANY] al valore “= numero slot richiesti della categoria”

Vincolo: proprietà (2/2)

❖ **Numero di slot logici richiesti per il modello**

- 1 [property RSNE]
- Molti [property RSNE] [property RSMANY]

❖ **Numero di slot logici opzionali per il modello**

- 1 [property OSNE]
- Molti [property OSNE] [property OSMANY]

❖ **Numero di componenti richiesti**

- 0 [if RSNE]
- < numero slot richiesti [if RSNE]
- = numero slot richiesti [if RSMANY]

❖ **Numero di componenti opzionali**

- < numero slot opzionali [if OSNE]
- = numero slot opzionali [if OSMANY]

Il vincolo proprietà riduce il numero di test da 2.711 a 908

Vincolo: single

- ★ Il vincolo **single** [single] indica una classe di valori che si desidera testare una sola volta e non su tutte le possibili combinazioni per ridurre il numero di test case.
 - Nel calcolo del numero di test case si comporta come il vincolo [error] ma ha una motivazione diversa che ne giustifica l'uso
- ★ **Componenti opzionali selezionati**
 - Alcuni sono default [single]
- ★ **Componenti richiesti selezionati**
 - Alcuni sono default [single]
- ★ **Numero di modelli nel database:**
 - 1 [single]
- ★ **Numero di componenti nel database:**
 - 1 [single]

Il vincolo single riduce il numero di test da 908 a 69

Pairwise Combinational Testing

- ★ Il *Category partition* è un test sistematico che individua in maniera esaustiva tutte le possibili combinazioni dei valori per un test. Per ridurre il numero di combinazioni si possono applicare dei vincoli che agiscono sui valore delle categorie. Tuttavia, talvolta inserire dei vincoli non è facile e si rischia di inserirne alcuni senza una motivazione razionale.
- ★ **Pairwise Testing**: test di tipo combinatorio che usa un numero limitato dei valori di input (coppie, triple).
 - L'idea alla base è che la maggior parte dei fallimenti software sono dovuti a combinazioni di pochi input (coppie, triple).
 - Si copre “la maggior” parte dei fault con pochi test
 - L'individuazione di coppie (triple) di valori di input può essere proibitiva se esistono diversi parametri del sistema, tuttavia si usano euristiche che semplificano la ricerca dei parametri.

Esempio: Display Control (1/4)

- Si consideri un display control descritto dai parametri: display mode, language, fonts, color, screen, size.

Display Mode	Language	Fonts	Color	Screen size
full-graphics	English	Minimal	Monochrome	Hand-held
text-only	French	Standard	Color-map	Laptop
limited-bandwidth	Spanish	Document-loaded	16-bit	Full-size
	Portuguese		True-color	

Pezzè & Young, Software Testing And Analysis: Process, Principles And Techniques, 2008

- Se si considerano tutte le possibili combinazioni si ottengono $3 \times 4 \times 3 \times 4 \times 3 = 432$ test case.

Esempio: Display Control (2/4)

- ❖ Se si osservano tutte le tuple che si ottengono dal prodotto dei valori di input, è facile rendersi conto che ci sono coppie di valori che si presentano più volte. Ad esempio:
 - <full-graphics, English, Minimal, Monochrome, Hand-held>
 - <full-graphics, English, Minimal, Monochrome, Laptop>
 - <full-graphics, English, Minimal, Monochrome, Full-Size>
- ❖ La coppia <full-graphics, English> compare, nel nostro esempio 3 volte (se calcolassimo tutte le combinazioni avremmo 36 tuple in cui essa compare), analogamente per le coppie <English, Minimal> , <Minimal, Monochrome>.
- ❖ I test pairwise richiedono di testare una sola volta ogni coppia.
- ❖ Se si scelgono le tuple in maniera tale da avere più coppie nella stessa tupla e far ripetere la tupla una sola volta, si può ridurre significativamente il numero di test. In genere si usano degli algoritmi e/o tecniche per calcolare le tuple in maniera che non ripetano più volte la stessa coppia.

Esempio: Display Control (3/4)

Considerando solo coppie di valori per tutti i parametri si ottengono 17 test case

Language	Color	Display Mode	Fonts	Screen Size
English	Monochrome	Full-graphics	Minimal	Hand-held
English	Color-map	Text-only	Standard	Full-size
English	16-bit	Limited-bandwidth	-	Full-size
English	True-color	Text-only	Document-loaded	Laptop
French	Monochrome	Limited-bandwidth	Standard	Laptop
French	Color-map	Full-graphics	Document-loaded	Full-size
French	16-bit	Text-only	Minimal	-
French	True-color	-	-	Hand-held
Spanish	Monochrome	-	Document-loaded	Full-size
Spanish	Color-map	Limited-bandwidth	Minimal	Hand-held
Spanish	16-bit	Full-graphics	Standard	Laptop
Spanish	True-color	Text-only	-	Hand-held
Portuguese	-	-	Monochrome	Text-only
Portuguese	Color-map	-	Minimal	Laptop
Portuguese	16-bit	Limited-bandwidth	Document-loaded	Hand-held
Portuguese	True-color	Full-graphics	Minimal	Full-size
Portuguese	True-color	Limited-bandwidth	Standard	Hand-held

Pezzè & Young, Software Testing And Analysis: Process, Principles And Techniques, 2008

Esempio: Display Control (4/4)

- È possibile ridurre ulteriormente il numero di test se si introducono dei vincoli (come per il category partition testing). Ad es., si può imporre che il colore monocromatico non sia compatibile con screen laptop e full size

Display Mode	Language	Fonts	Color	Screen size
full-graphics	English	Minimal	Monochrome	Hand-held
text-only	French	Standard	Color-map	
limited-bandwidth	Spanish	Document-loaded	16-bit	
	Portuguese		True-color	

Display Mode	Language	Fonts	Color	Screen size
full-graphics	English	Minimal		
text-only	French	Standard	Color-map	Laptop
limited-bandwidth	Spanish	Document-loaded	16-bit	Full-size
	Portuguese		True-color	

SOFTWARE TESTING
AND ANALYSIS

Pezzè & Young, Software Testing And Analysis: Process, Principles And Techniques, 2008

Sommario

- ★ I test combinatoriali caratterizzano i test con un insieme di parametri a cui è possibile associare un set di valori. I test case sono generati combinando i valori associati ai parametri
- ★ Il category partition è un test combinatoriale in cui è possibile introdurre dei vincoli per ridurre il numero dei test. I vincoli sono introdotti secondo ipotesi di compatibilità, di errore o singolarità.
- ★ Il Pairwise testing è un test combinatoriale che si limita a considerare coppie di valori di input. Si basa sull'idea che una coppia di valori è sufficiente a trovare errori (dual failure mode). Possono essere usati come alternativa al category partition testing se l'introduzione dei vincoli per quest'ultimo è onerosa (oppure se le categorie sono scarsamente vincolate).