# Kombinatorno testiranje

# Uvod

- Na ponašanje aplikacije utiče puno faktora, npr. ulazne vrednosti, konfiguracije okruženja.
- Tehnike kao što je podela na klase ekvivalencije ili analiza graničnih vrednosti identifikuju moguće vrednosti pojedinih faktora
- Od značaja su i kombinacije vrednosti, ali nije praktično testirati sve moguće kombinacije

# Primer: servis za naručivanje pice SNP

- SNP prima naružbe, proverava validnost i inicira slanje pice.
- Mušterija mora da navede četiri podatka: veličina pice, preliv, adresa dostave i kučni broj telefona.

# PDS: Model prostora ulaznih podataka

| Factor | Levels | | |
|---|---|---|---|
| Size | Large | Medium | Small |
| Toppings | Custom | Preset | |
| Address | Valid | Invalid | |
| Phone | Valid | Invalid | |

- Ukupan broj kombinacija vrednosti faktora je $3*2^3=24$
- Drugačije vrednosti faktora bi još povećale broj kombinacija

# Primer: Testiranje korisničkog interfejsa

GUI aplikacije sastoji se od menija File, Edit i Format.

| Factor | Levels | | | |
|---|---|---|---|---|
| File | New | Open | Save | Close |
| Edit | Cut | Copy | Paste | Select |
| Typeset | LaTex | BibTex | PlainTeX | MakeIndex |

Ima tri faktora, a svaki može da se postavi na bilo koji od 4 nivoa. Ukupno $4^3=64$ kombinacije.

# Primer: sort komanda na UNIXu

**sort** [-cmu] [-ooutput] [-Tdirectory] [-y [ kmem]] [-zrecsz] [-dfiMnr] [-b] [ tchar] [-kkeydef] [+pos1[-pos2]] [file...]

- Može se identifikovati 20 faktora for za sort komandu.
- Uzimajući u obzir njihove moguće vrednosti, tj. sve nivoe, postoji oko $1.9 \times 10^9$ kombinacija.

# Testiranje kompatibilnosti

- npr. testiranje web aplikacije na različitim platformama i browserima
- 3*5*5=75 kombinacija, ali nisu sve u praksi moguće

| Hardware | Operating System | Browser |
|---|---|---|
| Dell Dimension Series | Windows Server 2003-Web Edition | Internet Explorer 6.0 |
| Apple G4 | Windows Server 2003-64-bit Enterprise Edition | Internet Explorer 5.5 |
| Apple G5 | Windows XP Home Edition | Netscape 7.3 |
| | OS 10.2 | Safari 1.2.4 |
| | OS 10.3 | Enhanced Mosaic |

# Primer otkaza usled interakcije vrednosti

```
// pretpostavka x, y ∈ {-1, 1}, and z ∈ {0, 1}
begin
        int x, y, z, p;
        input (x, y, z);
        p = (x + y) * z // trebalo bi da bude p = (x – y) * z
        if (p >= 0)
                output (f(x, y, z));
        else
                output (g(x, y));
end
```

- Otkaz iniciraju sve vrednosti koje zadovoljavaju
  $x+y \neq x-y$ i $z \neq 0$.
- Međutim, otkaz otkrivaju samo dve od mogućih osam kombinacija ulaza: x=-1, y=1, z=1 i x=-1, y=-1, z=1.

# Kombinatorno testiranje

- Umesto testiranja svih mogućih kombinacija, uzima se podskup kombinacija koji zadovoljava neku od definisanih kombinatornih strategija testiranja.
- Ne utiče svaki ulazni faktor na svaki otkaz, često je otkaz iniciran interakcijama malog broja faktora
- Zbog toga kombinatorne strategije mogu dramatično redukovati broj kombinacija koje treba pokriti ali pri tom ostaju vrlo efikasne u detekciji otkaza.

# Kombinatorne strategije testiranja

- Pokrivanje svih kombinacija
- Pokrivanje svih pojedinih vrednosti
- **Pokrivanje parova vrednosti**
- **Pokrivanje n-torki vrednosti**

# Pokrivanje svih kombinacija

- Svaka moguća kombinacija vrednosti svih ulaznih parametara mora biti pokrivena. Primer:
  - Pretpostavimo da program koji se testira ima tri ulazne promenljive, od kojih svaka uzima jednu od dve vrednosti
  - Neka su X, Y i Z ulazne promenljive a {X1, X2}, {Y1, Y2}, {Z1, Z2} njihovi skupovi vrednosti. Svi mogući skupovi kombinacija ($2^3$) su:

$$\begin{array}{ll} (X_1, Y_1, Z_1) & (X_1, Y_1, Z_2) \\ (X_1, Y_2, Z_1) & (X_1, Y_2, Z_2) \\ (X_2, Y_1, Z_1) & (X_2, Y_1, Z_2) \\ (X_2, Y_2, Z_1) & (X_2, Y_2, Z_2) \end{array}$$

# Pokrivanje svih pojedinačnih vrednosti

- Svaka vrednost svakog parametra mora biti pokrivena bar jedanput.
  - Za prethodni primer, serija testova koja zadovoljava je:
    $\{(X_1, Y_1, Z_1), (X_2, Y_2, Z_2)\}$

# Pokrivanje parova vrednosti

- Za bilo koja dva parametra, sve kombinacije vrednosti ovih parametara moraju biti pokrivene bar u jednom test primeru

  - Za posmatrani primer: X, Y i Z ulazne promenljive a {X1, X2}, {Y1, Y2}, {Z1, Z2} njihovi skupovi vrednosti
  - ima 12 parova vrednosti: (X1, Y1), (X1, Y2), (X1, Z1), (X1, Z2), (X2, Y1), (X2, Y2), (X2, Z1), (X2, Z2), (Y1, Z1), (Y1, Z2), (Y2, Z1) i (Y2, Z2).

# Pokrivanje parova vrednosti - primer

- Sledeće četiri kombinacije pokrivaju sve parove:

$$(X_1, Y_1, Z_2) \quad (X_1, Y_2, Z_1)$$
$$(X_2, Y_1, Z_1) \quad (X_2, Y_2, Z_2)$$

- *Moguće je naći i druge skupove od četiri kombinacije koje pokrivaju svih 12 parova.*
- Radi se o balansiranom skupu kombinacija jer se svaka vrednost svakog parametra pojavljuje tačno jednak broj puta.

# Pokrivanje n-torki vrednosti

- Za bilo koji podskup od n parametara, sve kombinacije vrednosti tih n parametra moraju biti pokrivene bar jednim test primerom

- Primetiti da su prethodni kriterijumi samo specijalni slučajevi ovog najopštijeg kriterijuma

# Konstrukcija kombinatornih tabela

- Metodi zasnovani na pretrazi u prostoru stanja uglavnom razvijaju računarci
  - AETG (from Telcordia), TCG (from JPL/NASA), DDA (from ASU), PairTest
- Algebarski metodi koje su razvili matematičari
  - Ortogonalni nizovi (orthogonal arrays)
  - Pokrivajući nizovi (covering arrays)

# Ortogonalni nizovi

| Run | $F_1$ | $F_2$ | $F_3$ |
|---|---|---|---|
| 1 | 1 | 1 | 1 |
| 2 | 1 | 2 | 2 |
| 3 | 2 | 1 | 2 |
| 4 | 2 | 2 | 1 |

OA(N, k, s, t)
N=4 kombinacija (runs)
k=3 faktora
s=2 vrednosti (levels, symbols)
t=2 jačina (strength)

- Ortogonalni niz je N x k matrica popunjena vrednostima iz konačnog skupa od s simbola, sa svojstvom (balansa) da bilo koja N x t podmatrica sadrži proizvoljnu t-torku vrednosti uvek isti broj puta..

- Taj broj naziva se indeks ortogonalnog niza i računa kao $\lambda = N / s^t$

# Ortogonalni nizovi: Indeks

| Run | $F_1$ | $F_2$ | $F_3$ |
|---|---|---|---|
| 1 | 1 | 1 | 1 |
| 2 | 1 | 2 | 2 |
| 3 | 2 | 1 | 2 |
| 4 | 2 | 2 | 1 |

*OA*(4, 3, 2, 2)

N=4 kombinacija (runs)

k=3 faktora

s=2 vrednosti (levels, symbols)

t=2 jačina (strength)

$\lambda = N/s^t = 4/2^2 = 1$ što znači da se svaki par pojavljuje tačno jednom u bilo kome 4x2 podnizu.

Ima ukupno $s^t = 2^2 = 4$ para: (1, 1), (1, 2), (2, 1), (2, 2).

# Ortogonalni nizovi: drugi primer

| Run | $F_1$ | $F_2$ | $F_3$ | $F_4$ |
|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 |
| 2 | 1 | 2 | 2 | 3 |
| 3 | 1 | 3 | 3 | 2 |
| 4 | 2 | 1 | 2 | 2 |
| 5 | 2 | 2 | 3 | 1 |
| 6 | 2 | 3 | 1 | 3 |
| 7 | 3 | 1 | 3 | 3 |
| 8 | 3 | 2 | 1 | 2 |
| 9 | 3 | 3 | 2 | 1 |

OA(9, 4, 3, 2)

OA(runs, factors, levels, strength)
indeks $\lambda$=1.

# Ortogonalni nizovi: Alternativno obeležavanje

$L_N(s^k)$

Ortogonalni niz sa N kombinacija gde k faktora uzima neku od s vrednosti.

$L_4(2^3)$

| Run | $F_1$ | $F_2$ | $F_3$ |
|---|---|---|---|
| 1 | 1 | 1 | 1 |
| 2 | 1 | 2 | 2 |
| 3 | 2 | 1 | 2 |
| 4 | 2 | 2 | 1 |

$L_9(3^4)$

| Run | $F_1$ | $F_2$ | $F_3$ | $F_4$ |
|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 |
| 2 | 1 | 2 | 2 | 3 |
| 3 | 1 | 3 | 3 | 2 |
| 4 | 2 | 1 | 2 | 2 |
| 5 | 2 | 2 | 3 | 1 |
| 6 | 2 | 3 | 1 | 3 |
| 7 | 3 | 1 | 3 | 3 |
| 8 | 3 | 2 | 1 | 2 |
| 9 | 3 | 3 | 2 | 1 |

# Konstrukcija OA koristeći polje Galoa

Aritmetika po modulu 3, sabiranje i množenje – GF(3)

| + | 0 | 1 | 2 |
|---|---|---|---|
| 0 | 0 | 1 | 2 |
| 1 | 1 | 2 | 0 |
| 2 | 2 | 0 | 1 |

| x | 0 | 1 | 2 |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 2 |
| 2 | 0 | 2 | 1 |

Zameniti svaku nulu u tabeli množenja nultom kolonom tabele sabiranja. Isto to uraditi sa 1 i 2 i odgovarajućim kolonama. Tako ćemo dobiti OA sa 9 redova i tri kolone

# Ortogonalni niz $L_9(3^3)$

Zamenili smo svaki element x tabele odgovarajućom kolonom + tabele

| + | 0 | 1 | 2 |
|---|---|---|---|
| 0 | 0 | 1 | 2 |
| 1 | 1 | 2 | 0 |
| 2 | 2 | 0 | 1 |

$L_9(3^3)$

| x | 0 | 1 | 2 |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| | 1 | 1 | 1 |
| | 2 | 2 | 2 |
| 1 | 0 | 1 | 2 |
| | 1 | 2 | 0 |
| | 2 | 0 | 1 |
| 2 | 0 | 2 | 1 |
| | 1 | 0 | 2 |
| | 2 | 1 | 0 |

# Dobijanje ortogonalnog niza $L_9(3^4)$

Dodati jednu kolonu. Popuniti tako da svaka tri reda promenimo vrednost u njoj dok se ne izređaju sve tri vrednosti

| | | | | | |
|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | |
| | 0 | 1 | 1 | 1 | |
| | 0 | 2 | 2 | 2 | |
| | 1 | 0 | 1 | 2 | |
| | 1 | 1 | 2 | 0 | |
| | 1 | 2 | 0 | 1 | |
| | 2 | 0 | 2 | 1 | |
| | 2 | 1 | 0 | 2 | |
| | 2 | 2 | 1 | 0 | |

# OA mešovitih nivoa (Mixed level Orthogonal arrays)

- Prethodni *OA* bili su sa fiksnim nivoom, što znači da su svi faktori uzimali vrednosti iz istog skupa.

- U praktičnoj primeni faktori uzimaju vrednosti iz različitih skupova.

- Obeležavanje: $MA(N, s_1^{k_1} s_2^{k_2} \ldots s_p^{k_p}, t)$

  Strength=t. Runs=N.
  k1 faktora ima s1 nivoa, k2 ima s2 nivoa, itd.

Ukupan broj faktora: $\sum_{i=1}^{p} k_i$

# OA mešovitih nivoa (nastavak)

Osobina balansa za ove nizove takođe važi, tj. bilo koja N x t podmatrica sadrži proizvoljnu t-torku vrednosti, uvek isti broj puta, a taj broj je $\lambda$.

Indeks $\lambda$ ne može se međutim računati po ranije navedenoj formuli.

# OA mešovitih nivoa: Primer

$$MA(8, 2^4 4^1, 2)$$

| Run | $F_1$ | $F_2$ | $F_3$ | $F_4$ | $F_5$ |
|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 |
| 2 | 2 | 2 | 2 | 2 | 1 |
| 3 | 1 | 1 | 2 | 2 | 2 |
| 4 | 2 | 2 | 1 | 1 | 2 |
| 5 | 1 | 2 | 1 | 2 | 3 |
| 6 | 2 | 1 | 2 | 1 | 3 |
| 7 | 1 | 2 | 2 | 1 | 4 |
| 8 | 2 | 1 | 1 | 2 | 4 |

Balans: U bilo kojoj podmatrici 8 x 2, svaki par vrednosti (strength=2) pojavljuje se tačno jednom.

# Generisanje testova na osnovu MA: Primer dostave pice

| Factor | Levels | | |
|---|---|---|---|
| Size | Large | Medium | Small |
| Toppings | Custom | Preset | |
| Address | Valid | Invalid | |
| Phone | Valid | Invalid | |

Postoje 3 binarna faktora i jedan faktor sa 3 nivoa.
U skladu sa tim upotrebićemo sledeći niz:

$$MA(12, 2^3, 3^1, 2)$$

# Generisanje testova na osnovu MA: Primer dostave pice

$$MA(12, 2^3, 3^1, 2)$$

*Svi postojeći parovi vrednosti faktora su pokriveni.*

| Run | Size | Toppings | Address | Phone |
|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 |
| 2 | 1 | 1 | 2 | 1 |
| 3 | 1 | 2 | 1 | 2 |
| 4 | 1 | 2 | 2 | 2 |
| 5 | 2 | 1 | 1 | 2 |
| 6 | 2 | 1 | 2 | 2 |
| 7 | 2 | 2 | 1 | 1 |
| 8 | 2 | 2 | 2 | 1 |
| 9 | 3 | 1 | 1 | 2 |
| 10 | 3 | 1 | 2 | 1 |
| 11 | 3 | 2 | 1 | 1 |
| 12 | 3 | 2 | 2 | 2 |

# Generisanje testova na osnovu MA: Primer dostave pice

Testovi

| Run | Size | Toppings | Address | Phone |
|---|---|---|---|---|
| 1 | Large | Custom | Valid | Valid |
| 2 | Large | Custom | Invalid | Valid |
| 3 | Large | Preset | Valid | Invalid |
| 4 | Large | Preset | Invalid | Invalid |
| 5 | Medium | Custom | Valid | Invalid |
| 6 | Medium | Custom | Invalid | Invalid |
| 7 | Medium | Preset | Valid | Valid |
| 8 | Medium | Preset | Invalid | Valid |
| 9 | Small | Custom | Valid | Invalid |
| 10 | Small | Custom | Invalid | Valid |
| 11 | Small | Preset | Valid | Valid |
| 12 | Small | Preset | Invalid | Invalid |

# Pokrivajući nizovi

- Pokrivajući niz *CA*(N, k, s, t) je N x k matrica sa ulazima iz konačnog skupa od s symbola tako da svaka N x t podmatrica sadrži svaku moguću t-torku **najmanje** $\lambda$ puta.

- N – broj kombinacija (runs), k – broj faktora, s – broj nivoa za faktore, t je jačina, a $\lambda$ je indeks.

- *CA* su generalizacija *OA*. Svaki *OA* je *CA*, ali obrnuto ne važi.

- Za potrebe generisanja softverskih testova, razmatramo samo $\lambda$=1. Za razliku od *OA*, *CA* nisu balansirani i zato obično imaju manje N za iste ostale faktore.

# Pokrivajući nizovi: Primer

- Balansirani *OA* jačine 2 za 5 binarnih faktora, zahteva 8 kombinacija.
- Međutim, *CA* istih karakteristika zahteva samo 6.

$OA(8, 5, 2, 2)$=

| Run | $F_1$ | $F_2$ | $F_3$ | $F_4$ | $F_5$ |
|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 |
| 2 | 2 | 1 | 1 | 2 | 2 |
| 3 | 1 | 2 | 1 | 2 | 1 |
| 4 | 1 | 1 | 2 | 1 | 2 |
| 5 | 2 | 2 | 1 | 1 | 2 |
| 6 | 2 | 1 | 2 | 2 | 1 |
| 7 | 1 | 2 | 2 | 2 | 2 |
| 8 | 2 | 2 | 2 | 1 | 1 |

$CA(6, 5, 2, 2)$=

| Run | $F_1$ | $F_2$ | $F_3$ | $F_4$ | $F_5$ |
|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 |
| 2 | 2 | 2 | 1 | 2 | 1 |
| 3 | 1 | 2 | 2 | 1 | 2 |
| 4 | 2 | 1 | 2 | 2 | 2 |
| 5 | 2 | 2 | 1 | 1 | 2 |
| 6 | 1 | 1 | 1 | 2 | 2 |

# Pokrivajući nizovi sa mešovitim nivoima

- Pokrivajući niz sa mešovitim nivoima označava se kao

$$MCA(N, s_1^{k_1} s_2^{k_2} \ldots s_p^{k_p}, t)$$

i odnosi na N x Q matricu tako da važi $Q = \sum_{i=1}^{p} k_i$ i svaka N x t podmatrica sadrži bar jednu pojavu svake t-torke vrednosti.

- s1, s2,… označavaju broj nivoa za svaki odgovarajući faktor.

# Pokrivajući nizovi sa mešovitim nivoima: Primer

MCA su generalno manji od MOA i samim tim pogodniji za upotrebu u testiranju.

$MCA(6, 2^3 3^1, 2)$

| Run | Size | Toppings | Address | Phone |
|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 |
| 2 | 2 | 2 | 1 | 2 |
| 3 | 3 | 1 | 2 | 2 |
| 4 | 1 | 2 | 2 | 2 |
| 5 | 2 | 1 | 2 | 1 |
| 6 | 3 | 2 | 1 | 1 |

U poređenju sa ranijim $MA(12, 2^3 3^1, 2)$ uočava se redukcija za 6 kombinacija.

*Is the above array balanced?*

# Konstrukcija CA: Terminologija

- N-tostruka serija testova -> N-tostruki pokrivajući niz
- Testovi -> Redovi
- Parametri -> Faktori ili kolone
- Vrednosti -> Nivoi (Levels)

# In Parameter Order Strategija

- Konstruiše t-struku seriju testova na inkrementalni način
  - t-struka serija testova se konstruiše za prvih t parametara,
  - Onda se serija testova proširuje da generiše t-struku seriju testova za prvih t + 1 parametara
  - Prethodni koraci se ponavljaju za svaki sledeći parametar
- U svakom proširenju za novi parametar preduzimaju se dva koraka:
  - Horizontalni rast: proširuje svaki postojeći test dodajući jednu vrednost za novi parametar
  - Vertikalni rast: dodati novi test, ako je to potrebno

# IPO strategija (nastavak)

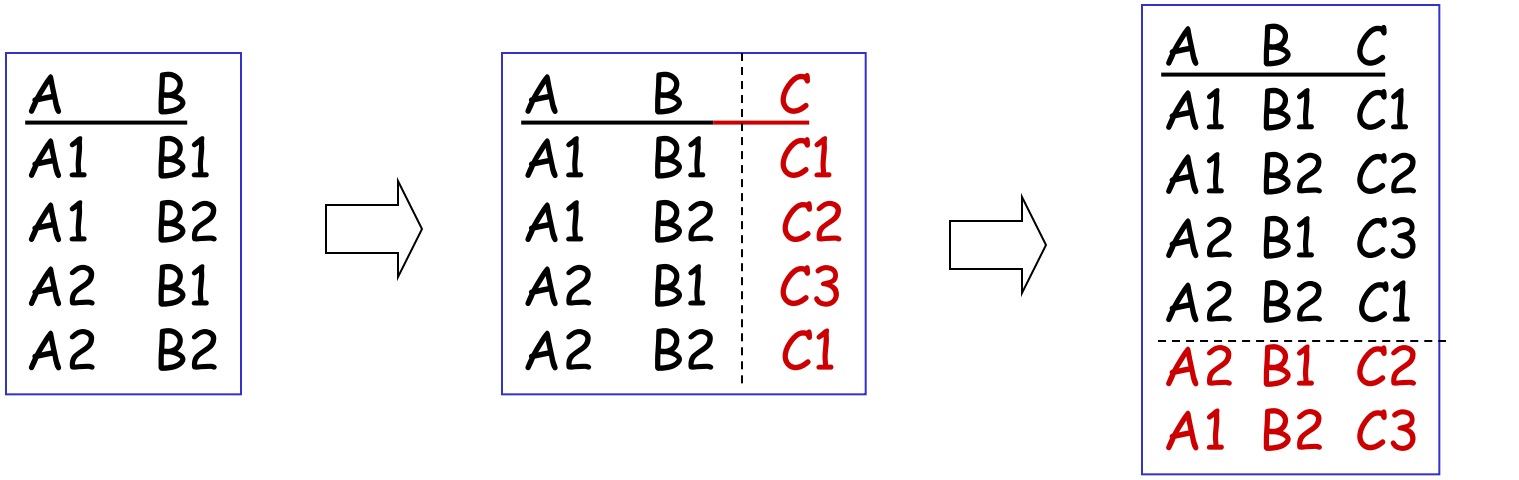
```
Strategy In-Parameter-Order
begin
   /* for the first t parameters p1, p2 , …,  pt*/
   T := {(v1, v2, …, vt) | v1, v2, …, vt are values of p1, p2, …, Pk, respectively}
   if n = t then stop;
   /* for the remaining parameters */
   for parameter pi, i = t + 1, …, n do
   begin
      /* horizontal growth */
      for each test (v1, v2, …, vi-1) in T do
          replace it with (v1, v2, …, vi-1, vi), where vi is a value of pi
       /* vertical growth */
       while T does not cover all the interactions between pi and
             each of p1, p2, …, pi-1 do
          add a new test for p1, p2, …, pi to T;
    end
end
```

# Primer

Razmotrimo sistem sa sledećim parametrima:

- parametar *A* ima vrednosti *A1* i *A2*
- parametar *B* ima vrednosti *B1* i *B2*
- parametar *C* ima vrednosti *C1*, *C2* i *C3*

# Primer (nastavak)



| A | B |
|---|---|
| A1 | B1 |
| A1 | B2 |
| A2 | B1 |
| A2 | B2 |

| A | B | C |
|---|---|---|
| A1 | B1 | C1 |
| A1 | B2 | C2 |
| A2 | B1 | C3 |
| A2 | B2 | C1 |

| A | B | C |
|---|---|---|
| A1 | B1 | C1 |
| A1 | B2 | C2 |
| A2 | B1 | C3 |
| A2 | B2 | C1 |
| A2 | B1 | C2 |
| A1 | B2 | C3 |

Horizontalni rast

Vertikalni rast