Software Testing

Sibylle Schupp[1]

[1]Institute for Software Systems/Institut für Softwaresysteme
Hamburg University of Technology (TUHH)

Spring 2022

Lecture 9

Outline

Recall

- Logic coverage criteria (semantic): PC, CC, CoCC, ACC (GACC, CACC, RACC), ICC
- Semantic logic coverage is concerned with possible meanings (truth values) of a clause or predicate.
  - Advantage: tests are independent of the particular way (“syntax”) a predicate is written.
  - Limitation: syntax mistakes cannot be detected.

Outline

Disjunctive normal form (DNF)

DNF, take 2

Definition

- A literal is a clause or the negation of a clause.
- A term is a set of literals connected by a logical "and."
- A predicate is in disjunctive normal form (DNF) if it is a set of terms connected by logical "or."

Example (DNF)

Notation

- For DNF, $+$ and concatenation are commonly used as operators (for $\vee, \wedge$), negation is indicated by a bar and can only be applied to clauses ("atoms").
- Logic operators, in two different notations:

			example
and	$\wedge$	(concat)	ab
or	$\vee$	$+$	a+b
not	$\neg$	$\bar{.}$	$\bar{a}$

- Examples (DNF): ab, $a\bar{b}$, $a\bar{b} + \bar{a}$, $a\bar{c} + b\bar{c}$,
  - Counter example: $\overline{a+b}$, $(a+b)\bar{c}$
- As names for boolean values, we will use both True, False and 0,1.

Implicants

Definition

- A normal term is a term in which a clause (“atom”) occurs not more than once.
- An implicant of a predicate is a normal term that implies the predicate: whenever the implicant is true, the predicate is true.

Discussion

- By the previous definition of “term”, every term is normal.
- By the previous definition of DNF, every term is an implicant.

Implicant coverage (IC)

Definition

Given a DNF representation of a predicate f and its negation $\overline{f}$. For implicant coverage (IC) TR contains for every implicant in f and $\overline{f}$ the requirement that the implicant evaluates to true.

Example

$$f = ab + b\overline{c} \qquad \overline{f} = \overline{b} + \overline{a}c$$

- Implicants: ab, $b\overline{c}$, $\overline{b}$, $\overline{a}c$
- Possible TR: {TTF, FFT}

Boolean algebras, again

DNF notation

1. De Morgan
   1. $\overline{ab} = \overline{a} + \overline{b}$
   2. $\overline{a + b} = \overline{a}\overline{b}$
2. Associativity
   1. $(ab)c = a(bc)$
   2. $(a + b) + c = a + (b + c)$
3. Distributivity
   1. $a(b + c) = (ab) + (ac)$
   2. $a + (bc) = (a + b)(a + c)$
4. Commutativity
   1. $ab = ba$
   2. $a + b = b + a$
5. Absorption
   1. $a(a + b) = a + (ab) = a$
   2. $\overline{a} + (\overline{a}b) = \overline{a}$

Example (Boolean algebra)

$$
\begin{aligned}
\overline{f} &:= \overline{ab + b\overline{c}} \\
&= \overline{ab}\;\overline{b\overline{c}} \\
&= (\overline{a} + \overline{b})(\overline{b} + \overline{\overline{c}}) \\
&= \overline{a}\overline{b} + \overline{a}c + \overline{b}\,\overline{b} + \overline{b}c \\
&= \overline{a}\overline{b} + \overline{b}\,\overline{b} + \overline{b}c + \overline{a}c \\
&= (\overline{a} + \overline{b})\overline{b} + \overline{b}c + \overline{a}c \\
&= \overline{b} + \overline{b}c + \overline{a}c \\
&= \overline{b} + \overline{a}c
\end{aligned}
$$

Prime implicants and redundant implicants

Let f be a function in DNF, t be a term of f.

Definition

- A proper subterm of an implicant i is i with one or more literals removed.
  - Ex.: proper subterms of i=abc: ab, bc, ac, a, b, c.
- A prime implicant of f is an implicant such that no proper subterm is also an implicant of f.
  - Ex.: $f = abc + ab\overline{c} + b\overline{c}$
    - abc is not a prime implicant: if subterm ab true, then either abc true or $ab\overline{c}$ true, thus f true and ab implicant;
    - $ab\overline{c}$ not a prime implicant (subterms ab, $b\overline{c}$)
- An implicant of a predicate f is redundant if it can be omitted without changing the value of f.

Example (redundant implicants)

Ex.

- $f = ab + ac + b\overline{c} \equiv ac + b\overline{c}$, thus ab is redundant.
    - If ab true, then either ac or $b\overline{c}$ true.
- A prime implicant can be redundant: see above, ab is prime.

Minimal DNFs

Definition

A minimal DNF representation consists only of prime, non-redundant implicants.

Discussion

- Minimal DNFs can be computed automatically.
- In the following, we assume minimal DNFs (no practical restriction).
- The feasibility of some coverage criteria depends on minimality.

Unique true points

Given a minimal DNF representation of a predicate f.

Definition

A unique true point (UTP) with respect to a given implicant is a truth assignment such that the given implicant is true and all other implicants are false.

Examples

- Unique true points $f = ab + b\overline{c}$
  - ab: {TTT}
  - $b\overline{c}$: {FTF}
- Unique true points $\overline{f} = \overline{b} + \overline{a}c$
  - $\overline{b}$: {FFF, TFF, TFT}
  - $\overline{a}c$: {FTT}

Multiple unique true point coverage (MUTP)

Definition

Given a minimal DNF representation of a predicate f. For multiple unique true point coverage (MUTP) TR requires for each implicant i unique true points (UTPs) so that clauses **not** in i take on both values true and false.

Discussion

- Minimality guarantees the existence of at least one UTP for each implicant. MUTP, however, might be infeasible.
- Example $f = ab + b\overline{c}$
  - MUTP infeasible (both implicants have one UTP only)
- Example $\overline{f} = \overline{b} + \overline{a}c$
  - For implicant $\overline{b}$: could select FFF and TFT.
  - MUTP infeasible for the second implicant

Near false points

Let f be as above.

Definition

A near false point (NFP) with respect to a clause c in a given implicant i is a truth assignment such that f is false, but if c is negated and all other clauses remain unchanged, the implicant i (and hence f) evaluates to true.

- Example: $f = ab + cd$.
  - Consider clause a in implicant ab. A near false point is FTFF.
  - Check: f evaluates to false; for $\overline{\overline{a}}$, ab and f evaluate to true.
- At a near false point, a clause c determines the predicate.

Corresponding unique true point and near false point pair coverage (CUTPNFP)

Definition

Given a minimal DNF representation of a predicate f. For corresponding unique true point and near false point pair coverage (CUTPNFP) TR contains for each clause c in each implicant i a unique true point for i and a near false point for c such that the points differ only in the truth value of clause c.

Discussion

- CUTPNFP subsumes RACC.

Example (CUTPNFP)

$$f = ab + cd$$

- Implicant *ab* has 3 UTPs: {TTFF, TTFT, TTTF}
  - Ex.: for clause *a* pair UTP TTFF with NFP FTFF.
  - Ex.: for clause *b* pair UTP TTFF with NFP TFFF.
- Implicant *cd* has 3 UTPs: {FFTT, FTTT, TFTT}
  - For clause *c* pair UTP FFTT with NFP FFFT
  - For clause *d* pair UTP FFTT with NFP FFTF.
- CUTPNFP test requirements:
  {TTFF, FFTT, TFFF, FTFF, FFTF, FFFT }

In-class exercise

1. Determine the prime implicants of f.
   - A prime implicant of f is an implicant such that no proper subterm is an implicant.
2. Determine the UTPs for each implicant of f.
   - Let i be an implicant. A unique true point of i is a valuation so that i is true and all other implicants are false.
3. Specify the TR for MUTP coverage.
   - MUTP: for each implicant i choose UTPs so that clauses not in i take on both true and false.
4. Determine the NFPs for clause a in "the first" implicant i
   - Let i be an implicant, let c be a clause in i. A near false point of c is a valuation of f so that f is false, but if c is negated, i and f evaluate to true.
5. What is the test size for CUTPNFP coverage?
   - CUTPNFP: for each clause c in an implicant i choose a UTP for i and a NFP for c such that those points differ only in the truth value of c.

In-class exercise (cont'd)

In-class exercise

Fault detection: insertion faults

Let $f = ab + cd$ and assume a fault L such that $f = abL + cd$. L is called an insertion fault.

- MUTP = { TTFT, TTTF, FTTT, TFTT }
- MUTP detects all insertion faults:

L = a	no fault
L = $\overline{a}$	TTFT and TTTF fail
L = b	as for a
L = $\overline{b}$	as for $\overline{a}$
L = c	TTFT fails
L = $\overline{c}$	TTTF fails
L = d	TTTF fails
L = $\overline{d}$	TTFT fails

- MUTP cannot detect literal omission.

Fault detection: omission faults

Let $f = ab + cd$ and assume a fault such that $f' = ab + c$.

- CUTPNFP = {TTFF, FFTT, TFFF, FTFF, FFTF, FFFT }
- CUTPNFP detects all ommission faults. In the example:

  FFTT and FFTF for faulty f' both return true
- Insertion and omission faults span the whole fault space.
- Note, though, that MUTP and CUTPNFP are not always feasible.

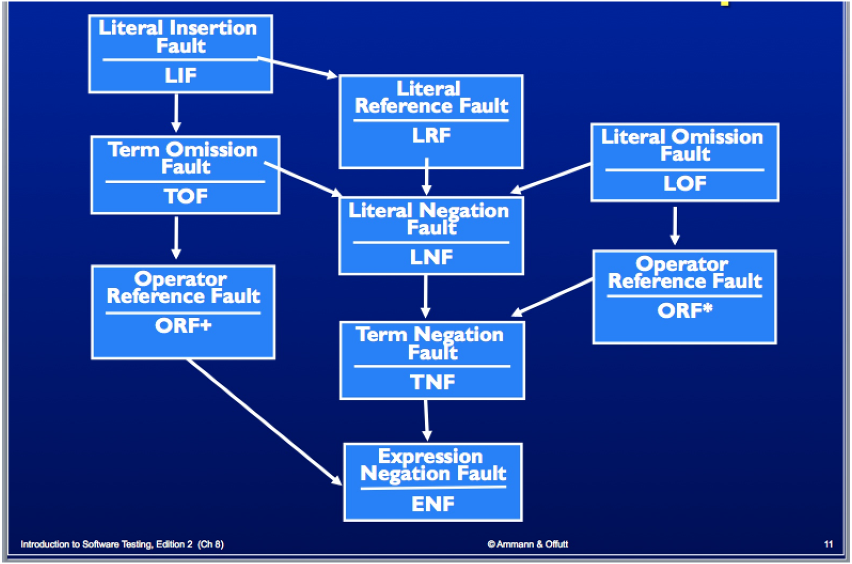
DNF fault classes

- Idea: classify faults
- Common faults (let $f = ab + c$).

Fault	Description
ENF	f incorrectly written as its negation: $\overline{ab + c}$
TNF	a term of f incorrectly written as its negation, e.g., $\overline{ab} + c$
LNF	a literal in f incorrectly written as its negation, e.g., $a\overline{b} + c$
TOF	a term of f incorrectly omitted, e.g., ab
LOF	a literal in f incorrectly omitted, e.g., $a + c$
LRF	a literal in f incorrectly replaced, e.g., $ac + c$
LIF	a literal in f incorrectly inserted, e.g., $ab + bc$
ORF+	an 'or' in f incorrectly replaced by 'and': abc
ORF*	an 'and' in f incorrectly replaced by 'or': $a + b + c$

- Dependencies?

Fault dectection relationships



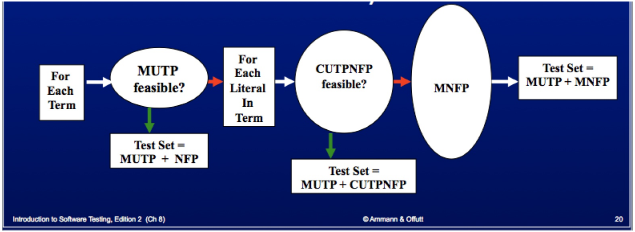
Multiple NFPs and MUMCUT

The MNFP criterion applies when MUTP or CUTPNFP are infeasible.

Definition

- Given a minimal DNF representation of a predicate f. For multiple near false point coverage (MNFP) *TR* contains for each literal in each implicant i near false points such that the clauses not in i take on both values true and false.
- Given a minimial DNF representation. For MUMCUT coverage *TR* combines the *TR*s of MUTP, CUTPNFP, and MNFP.

Minimal-MUMCUT



- One can show that the combined application of MUTP, CUTPNFP, and MNFP detects the entire fault hierarchy.

Outline

Karnaugh maps

Maurice Karnaugh (1953)

Karnaugh maps (K-map) are tabular representations of predicates.

- Boolean table
- Horizontal and vertical neighbors change 1 clause (“variable”) only
- Used for simplification of boolean expressions
- Suitable for 5+ clauses

Construction of K-maps

- Adjacent pairs differ in one truth value only.
- Think of the table in 3D: edge points are connected (toroidal mapping to 3D).
- Example (3 clauses ("variables")): $b\bar{c} + ac + \bar{b}c$

c\ ab	00	01	11	10
0		1	1	
1	1		1	1

- Example (4 clauses): ab + cd

cd \ ab	00	01	11	10
00			1	
01			1	
11	1	1	1	1
10			1	

Simplification

Simplification is done by grouping adjacent cells containing 1-entries.

- A *group* is a maximal rectangle of size 2^k.
  - Wrap-around and overlapping is permitted.
  - 1-entries that cannot be grouped form singletons (rectangles of size 1)
- Each rectangle corresponds to a predicate of the form $X(a + \bar{a})$, which is equivalent to X (absorption law).
  - Each maximal rectangle represents a prime implicant.
  - If its entries are covered by other rectangles, the implicant is redundant.
- Obtain minimal DNF by forming the disjunction of all non-redundant rectangles.

Applications

K-maps can be used for a variety of tasks in logic-based testing:

- Determination
- Negation
- Prime implicants and redundant implicants
- Unique true points
- Near false points

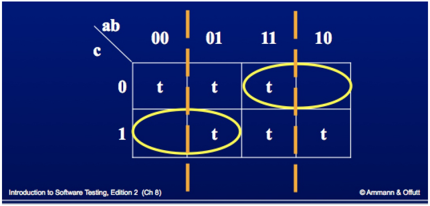
K-map: determination

Consider

$$f = b + \overline{ac} + ac$$

When does b determine f?

- K-Map



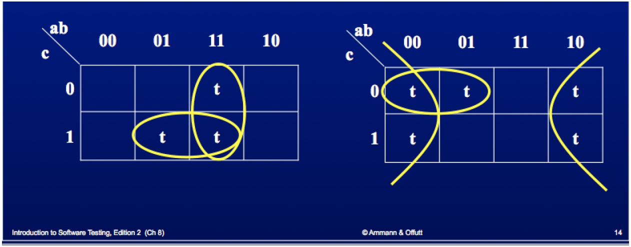
- Dashed line signals where b changes its value.
- If two cells joined by the dashed line have different values for f, then b determines f for those cells.
- Thus, b determines f for $\overline{a}c + a\overline{c}$ (but not at ac or $\overline{ac}$).

K-map: negation

Consider

$$f = ab + bc$$

- K-map



- Find groups
- Negation: $\overline{f} = \overline{b} + \overline{ac}$

K-map: Prime and redundant implicants

Consider

$$f = abc + ab\bar{d} + \bar{a}bcd + a\bar{b}c\bar{d} + a\bar{c}\bar{d}$$

- K-map

$cd \backslash ab$	00	01	11	10
00			1	1
01				
11		1	1	
10			1	1

$cd \backslash ab$	00	01	11	10
00			$ab\bar{d}, a\bar{c}\bar{d}$	$a\bar{c}\bar{d}$
01				
11		$\bar{a}bcd$	abc	
10			$abc, ab\bar{d}$	$a\bar{b}c\bar{d}$

- Prime implicants (1x size 4, 2x size 2)

$cd \backslash ab$	00	01	11	10
00			1	1
01				
11				
10			1	1

$cd \backslash ab$	00	01	11	10
00				
01				
11			1	
10			1	

$cd \backslash ab$	00	01	11	10
00				
01				
11		1	1	
10				

K-map: Prime and redundant implicants (cont'd)

- Again, the prime implicants of $f = abc + ab\bar{d} + \bar{a}bcd + a\bar{b}c\bar{d} + a\bar{c}\bar{d}$

cd\ab	00	01	11	10
00			1	1
01				
11				
10			1	1

cd\ab	00	01	11	10
00				
01				
11			1	
10			1	

cd\ab	00	01	11	10
00				
01				
11		1	1	
10				

- Redundant: the first implicant of size 2 is redundant
  - The 1111 entry is covered by the other implicant of size 2, the 1110 entry by the implicant of size 4
- The minimal DNF representation of f is $a\bar{d} + bcd$:

size 4	1100	size 2	0111
	1110		1111
	1000		
	1010		
	$a\bar{d}$		bcd

K-Map: unique true points

Consider f = ab + cd

- K-map

cd \ ab	00	01	11	10
00			1	
01			1	
11	1	1	1	1
10			1	

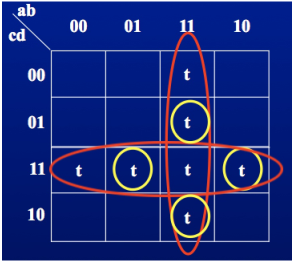
- Unique points for ab : $TTFF, TTFT, TTTF$
  - TTTT is true point, but not unique
- Unique points for cd : $FFTT, FTTT, TFTT$
  - TTTT is true point, but not unique

K-map: multiple unique true points

Recall MUTP coverage: given a minimal DNF representation, for each implicant *i* choose UTPs so that clauses not in *i* take on both true and false. Consider

$$f = ab + cd$$

- K-map:



- For implicant *ab* choose: {TTFT, TTTF}, for implicant *cd*: {FTTT, TFTT}. MUTP test set: {TTFT, TTFT, FTTT, TFTT}

K-map: CUTPNFP

- Consider again $f = ab + cd$ and recall CUTPNFP coverage
  - For each clause c in an implicant i choose a UTP for i and a NFP for c such that those points differ only in the truth value of c.
- K-map

cd \ ab	00	01	11	10
00		x	**1**	
01	x		**1**	x
11	**1**	1	1	1
10	x		1	

- Test requirements:

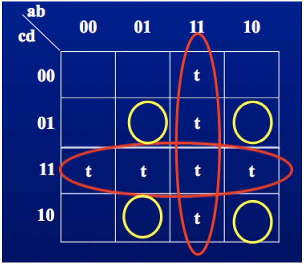
$$\begin{aligned} TR &= UTP \cup NFP \\ &= \{TTFF, TTFT, FFTT\} \cup \{FTFF, FFFT, TFFT, FFTF\} \end{aligned}$$

K-map: multiple near false points

Consider

$$f = ab + cd$$

- K-map:



- Recall MNFP: for each literal in each implicant *i* find NFPs so that clauses not in *i* take on values T and F.
- Implicant *ab*: choose {FTFT, FTTF} for *a*; for *b*: {TFFT, TFTF}
- MNFP *TR* : {TFTF, TFFT, FTTF, TFTF}

Summary (logic-based coverage)

- Semantic criteria
  - Active clauses (including MCDC)
  - Definitional method for determination (boolean laws)
- Syntactic criteria
  - Implicants, unique true points, near false points
  - Minimal DNFs
  - K-maps
- Applications
  - Predicates in programs
  - Finite state machines

References

- AO, Ch. 8.2