# DNF Criteria

Isaac Griffith

CS 4422 and CS 5599
Department of Computer Science
Idaho State University

Idaho State University
Computer Science

# Outcomes

At the end of Today's Lecture you will be able to:

- Understand what Karnaugh maps are, but will need additional practice.
- Understand the basic concepts surrounding Disjunctive Normal Form and its use.
- Understand DNF based coverage criteria.

ROAR

# Inspiration

“I don’t care if it works on your machine! We are not shipping your machine!” – Vidiu Platon

# Disjunctive Normal Form

- Common Representation for Boolean Functions
  - Slightly Different Notation for Operators
  - Slightly Different Terminology
- Basics:
  - A **literal** is a clause or the negation (overstrike) of a clause
    - Examples: $a$, $\overline{a}$
  - A **term** is a set of literals connected by logical "and"
    - "and" is denoted by adjacency instead of $\wedge$
    - Examples: $ab$, $a\overline{b}$, $\overline{ab}$ for $a \wedge b, a \wedge \neg b, \neg a \wedge \neg b$
  - A **(disjunctive normal form) predicate** is a set of terms connected by "or"
    - "or" is denoted by $+$ instead of $\vee$
    - Examples: $abc + \overline{a}b + a\overline{c}$
    - Terms are also called "implicants" if a term is true, that **implies** the predicate is true

# Implicant Coverage

- Obvious coverage idea: Make each implicant evaluate to "true"
  - Problem: Only tests "true" cases for the predicates
  - Solution: Include DNF representations for negation

## Implicant Coverage (IC)

Given DNF representations of a predicate $f$ and its negation $\overline{f}$, for each implicant in $f$ and $\overline{f}$, $TR$ contains the requirement that the implicant evaluate 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 test set: {TTF, FFT}
- Observation: IC is relatively weak

# Improving on Implicant Coverage

Additional Definitions:

- A **proper subterm** is a term with one or more clauses removed
  - Example: $abc$ has 6 proper subterms: $a$, $b$, $c$, $ab$, $ac$, $bc$
- A **prime implicant** is an implicant such that no proper subterm is also an implicant
  - Example: $f = ab + a\bar{b}c$
  - Impliant ab is a prime implicant
  - Implicant $a\bar{b}c$ is not a prime implicant (due to proper subterm $ac$)
- A **redundant implicant** is an implicant that can be removed without changing the value of the predicate
  - Example: $f = ab + ac + b\bar{c}$
  - ab is redundant
  - Predicate can be written: $ac + b\bar{c}$

# Unique True Points

- A **minimal DNF representation** is one with only prime, non-redundant implicants
- A **unique true point** with respect to a given implicant is an assignment of truth values so that
  - The given implicant is true, and
  - All other implicants are false
- A unique true point test focuses on just one implicant
- A minimal representation guarantees the existence of at least one unique true point for each implicant

**Multiple Unique true Point Coverage (MUTP):**

Given minimal DNF representation of a predicate $f$, for each implicant $i$, choose unique true points (UTPs) such that clauses not in $i$ take on values $T$ and $F$.

# Unique True Point Example

- Consider again: $f = ab + b\overline{c}$
  - Implicants: {$ab$, $b\overline{c}$}
  - Each implicant is prime
  - No implicant is redundant
- Unique true points:
  - $ab$: {TTT}
  - $b\overline{c}$: {FTF}
  - MUTP requires both of these
- But MUTP is still infeasible for both implicants
  - Not enough UTPs for clauses to take on all truth values
  - Later, we will have an example where MUTP is feasible

# Near False Points

- A **near false point** with respect to a clause $c$ in implicant $i$ is an assignment of truth values such that $f$ is false, but if $c$ is negated (and all other clauses left as is), $i$ (and hence $f$) evaluates to true
- Relation to **determination**: at a near false point: $c$ determines $f$
  - Hence we should expect relationship to ACC criteria

**Unique True Point and Near False Point Pair Coverage (CUTPNFP):**

Given a minimal DNF representation of a predicate $f$, for each clause $c$ in each implicant $i$, TR contains a unique true point for $i$ and a near false point for $c$ such that the points differ only in the truth value of $c$.

- Note that definition only mentions $f$, and not $\overline{f}$
- Clearly, CUTPNFP subsumes RACC

# CUTPNFP Example

- Consider $f = ab + cd$
  - Implicant $ab$ has 3 unique true points: {TTFF, TTFT, TTTF}
    - For clause $a$, we can pair unique true point $\underline{\mathrm{T}}$TFF with near false point $\underline{\mathrm{F}}$TFF
    - For clause $b$, we can pair unique true point T$\underline{\mathrm{T}}$FF with near false point T$\underline{\mathrm{F}}$FF
  - Implicant $cd$ has 3 unique true points: {FFTT, FTTT, TFTT}
    - For clause $c$, we can pair unique true point FF$\underline{\mathrm{T}}$T with near false point FF$\underline{\mathrm{F}}$T
    - For clasue $d$, we can pair unique true point FFT$\underline{\mathrm{T}}$ with near false point FFT$\underline{\mathrm{F}}$
- CUTPNFP set: {TTFF, FFTT, TFFF, FTFF, FFTF, FFFT}
  - First two tests are unique true points; others are near false points
- Rough number of tests required: # implicants * # literals

# The MNFP Criterion

The next two criteria provide enough scaffolding to make guarantees about fault detection

**Multiple Near False Point Coverage (MNFP):**

Given a minimal DNF representation of a predicate $f$, for each literal $c$ in each implicant $i$, TR choose near false points (NFPs) such that clauses not in $i$ take on values T and F.

# MNFP Example

- Consider again: $f = ab + b\overline{c}$
  - Implicants: {$ab$, $b\overline{c}$}
- Unique true points:
  - ab:
    - NFP for where $c = T : FTT$
    - Infeasible NFP for $a$ where $c = F$
    - NFPs for $b$ where $c = T, F : TFT, TFF$
  - bc:
    - NFPs for $b$ where $a = T, F : TFF, FFF$
    - NFP for $\overline{c}$ where $a = F : FTT$
    - Infeasible NFP for $\overline{c}$ where $a = T$
- Resulting MNFP set = **{FTT, TFT, TFF, FFF}**

# The MUMCUT Criterion

Together, these three criteria provide enough scaffolding to make guarantees about fault detection

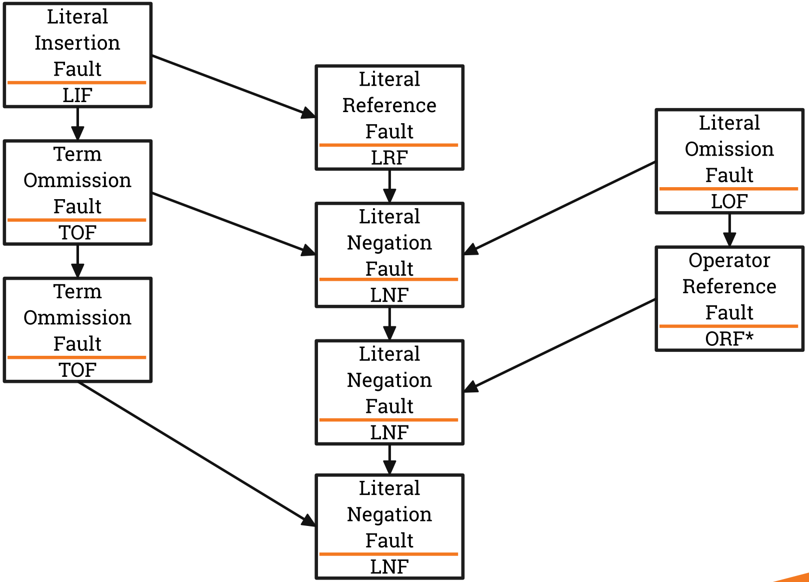
**MUMCUT:**

Given a minimal DNF representation of a predicate $f$, apply MUTP, CUTPNFP, and MNFP.

# DNF Fault Classes

- ENF: Expression Negation Fault $f = ab + c \quad f' = \overline{ab + c}$
- TNF: Term Negation Fault $f = ab + c \quad f' = \overline{ab} + c$
- TOF: Term Omission Fault $f = ab + c \quad f' = ab$
- LNF: Literal Negation Fault $f = ab + c \quad f' = a\bar{b} + c$
- LRF: Literal Reference Fault $f = ab + bcd \quad f' = ad + bcd$
- LOF: Literal Omission Fault $f = ab + c \quad f' = a + c$
- LIF: Literal Insertion Fault $f = ab + c \quad f' = ab + bc$
- ORF+: Operator Reference Fault $f = ab + c \quad f' = abc$
- ORF*: Operator Reference Fault $f = ab + c \quad f' = a + b + c$
- Key idea is that fault classes are related with respect to testing:
  - Test sets guaranteed to detect certain faults are also guaranteed to detect additional faults

# Fault Detection Relationships



# Karnaugh Maps

- Fair Warning
  - We use, rather than teach, Karnaugh Maps
  - Newcomers to K-Maps probably need a tutorial
    - Suggestion: Google "Karnaugh Map Tutorial"
- Our goal: Apply Karnaugh Maps to concepts used to test logic expressions
  - Identify when a clause determines a predicate
  - Identify the negation of a predicate
  - Identify prime implicants and redundant implicants
  - Identify unique true points
  - Identify unique true point / near false point pairs
- No new material here on testing
  - Just fast shortcuts for concepts already presented

# K-Map

## A Clause Determines a Predicate

- Consider the predicate:
  $f = b + \overline{ac} + ac$
- Suppose we want to identify when $b$ determines $f$
- The dashed line highlights where $b$ changes value
  - If two cells joined by the dashed line have different values for $f$, then $b$ determines $f$ for those two cells
  - $b$ determines $f$ : $\overline{a}c + a\overline{c}$ (but NOT at $ac$ or $\overline{ac}$)
- Repeat for clauses $a$ and $c$

| c \ ab | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | t | t | t | |
| 1 | | t | t | t |

Idaho State University
Computer Science

# K-Map

## Negation of a predicate

- Consider the predicate: $f = ab + bc$
- Draw the Karnaugh Map for the negation
  - Identify groups
  - Write down negation: $\overline{f} = \overline{b} + \overline{ac}$

| c \ ab | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | | | t | |
| 1 | | t | t | |

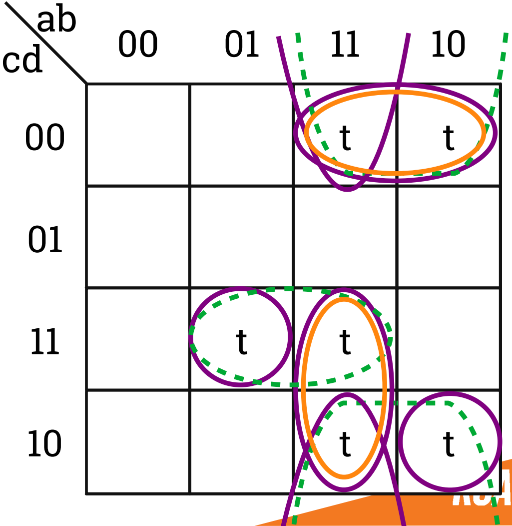
| c \ ab | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | t | t | | t |
| 1 | t | | | t |

ROAR

# K-Map

## Prime and Redundant Implicants

- Consider the prediate:
  $f = abc + ab\bar{d} + \bar{a}bcd + \bar{a}bc\bar{d} + a\overline{cd}$
- Draw the Karnaugh Map
- Implicants that are not prime:
  $ab\bar{d}$, $\bar{a}bcd$, $\bar{a}bc\bar{d}$, $a\overline{cd}$
- redundant implicant: $ab\bar{d}$
- Prime implicants:
  - Three: $a\bar{d}$, $bcd$, $abc$
  - The last is redundant
  - Minimal DNF representation
    - $f = a\bar{d} + bcd$

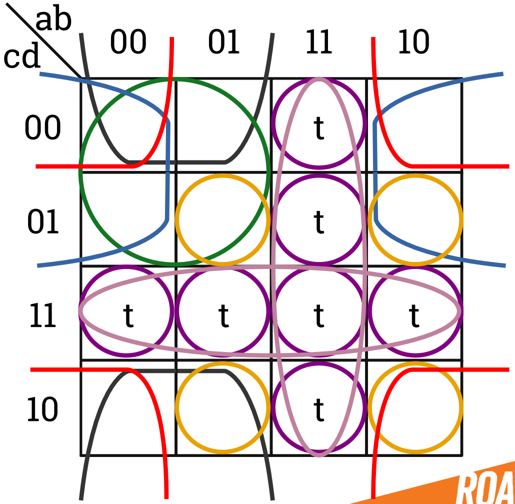


Idaho State University Computer Science

# K-Map

## Unique True Points

- Consider the predicate
  $f = ab + cd$
- Three unique true points for $ab$
  - TTFF, TTFT, TTTF
  - TTTT is a true point, but not a unique true point
- Three unique true points for $cd$
  - FFTT, FTTT, TFTT
- Unique true points for $\overline{f}$
  $\overline{f} = a\overline{c} + \overline{b}c + \overline{a}d + \overline{b}d$
  - FTFT, TFFT, FTTF, TFTF




ROAR

# MUTP

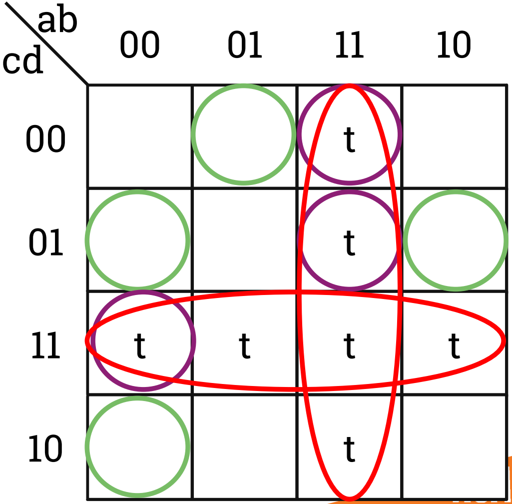
- For each implicant find unique true points (UTPs) so that
  - Literals no in implicant take on values T and F
- Consider the DNF predicate
  - $f = ab = cd$
- For implicant $ab$
  - Choose TTFT, TTTF
- For implicant $cd$
  - Choose FTTT, TFTT
- MUTP test set
  - {TTFT, TTTF, FTTT, TFTT}

| cd \ ab | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | | | t | |
| 01 | | | t | |
| 11 | t | t | t | t |
| 10 | | | t | |

# CUTPNFP

- Consider the DNF predicate: $f = ab + cd$
- For implicant $ab$
  - For $a$, choose UTP, NFP apir
    - TTFF, FTFF
  - For $b$, choose UTP, NFP pair
    - TTFT, TFFT
- For implicant $cd$
  - For $c$, choose UTP, NFP pair
    - FFTT, FFFT
  - For $d$, choose UTP, NFP pair
    - FFTT, FFTF
- Possible CUTPNFP test set

```
– {TTFF, TTFT, FFTT  // UTPS
  FTFF, TFFT, FFFT, FFTF  //
  NFPS}
```



# MNFP

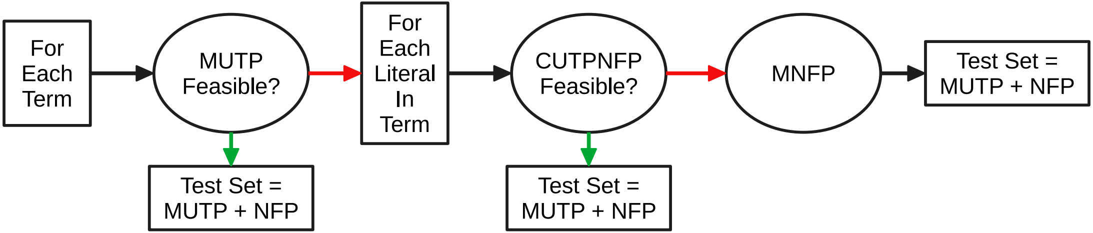
- Find NFP tests for each literal such that all literals not in the term attain F and T
- Consider the DNF predicate:
  - $f = ab + cd$
- For implicant $ab$
  - Choose FTFT, FTTF for $a$
  - Choose TFFT, TFTF for $b$
- For implicant $cd$
  - Choose FTFT, TFFT for $c$
  - Choose FTTF, TFTF for $d$
- MNFP test set
  - {TFTF, TFFT, FTTF, TFTF}
- Example is small, but generally MNFP is large

| cd \ ab | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | | | t | |
| 01 | | | t | |
| 11 | t | t | t | t |
| 10 | | | t | |

# Minimal-MUMCUT Criterion

Kaminski et al (ICST 2009)

- Minimal-MUMCUT uses low level **criterion feasibility analysis**
  - Adds CUTPNFP and MNFP only when necessary
- Minimsl-MUMCUT guarantees detecting LIF, LRF, LOF
  - And thus all 0 faults in the hierarchy



# Are there any questions?