## Cofactor

- First test to see if there is intersection
  - Then perform $\alpha_1+\beta_1'$  $\alpha_2+\beta_2'$  … $\alpha_n+\beta_n'$
  - E.g. $f=x'y'+xy$
    - Take cofactor w.r.t. x = [01 11]
    - First row – void, no intersection
    - Second row 11 01 = y
      - = $f_a$

10 10
01 01



## Operations on logic covers

- Recursive Paradigm
  - Base on the cofactor expansion theorem
    - E.g. $f+g=x(f_x+g_x)+x'(f_{x'}+g_{x'})$
  - Expand about a variable (possibly multi-valued)
  - Apply operation to cofactors
  - Merge Results

$$f \odot g = \sum_{k=0}^{p-1} x^{\{k\}} \cdot (f_{x^{\{k\}}} \odot g_{x^{\{k\}}})$$

- Unate Heuristics
  - Operations on unate functions are simpler
  - Select variables so that cofactors become unate functions
- Unate is roughly equal to monotonic increasing, but with more precise definition within Boolean algebra
- Unate Recursive Paradigm [Brayton et. al]



## Unate function and unate cover

- A bv-function is positve unate in variable x if $f_x \supseteq f_{x'}$
- A bv-function is negatie unate in variable x if $f_x \subseteq f_{x'}$
- A cover F is positive unate in x if all its implicants have 11 or 01 in the corresponding field
  - E.g. xy+xz'
- A cover F is negative unate in x if all its implicants have 11 or 10 in the corresponding field
  - E.g. x'y+x'z'
- A cover F is unate in x implies the function f is unate in x
  - The other way is not true!
    - x'+x is a unate function, not a unate cover!
- Function vs cover = {minterms} vs {implicants} = f vs F



## Unate mvi-functions

- Strong Unateness (not very useful, but correspond to bv case)
  - A function f is positive unate in variable x if $f_x^{\{j\}} \supseteq f_x^{\{k\}}$ $\forall j,k \in (0,1,\ldots,p-1) : j>k$
  - A function f is negative unate in variable x if $f_x^{\{j\}} \subseteq f_x^{\{k\}}$ $\forall j,k \in (0,1,\ldots,p-1) : j>k$
  - E.g. Given all variable can have {0,1,2}
    - $f=x^{\{2\}}y^{\{0,1\}}+x^{\{1\}}y^{\{0\}}$
    - $f_x^{\{2\}}=y^{\{0,1\}} \supseteq f_x^{\{1\}}=y^{\{0\}} \supseteq f_x^{\{0\}}=\varnothing$
    - Strongly unate in x

001 110
010 100

  - E.g. Given all variable can have {0,1,2}
    - $f=x^{\{2\}}y^{\{0,1\}}+x^{\{1\}}y^{\{0,2\}}+x^{\{0\}}y^{\{1,2\}}$
    - $f_x^{\{2\}}=y^{\{0,1\}} \not\supseteq f_x^{\{1\}}=y^{\{0,2\}} \not\supseteq f_x^{\{0\}}= y^{\{1,2\}}$
    - Not Strongly unate in x

001 110
010 101
100 011

There exists no permutation of values that the cofactor will form containment relation



## Unate mvi-functions

- Weak Unateness
  - A function f is weakly unate in variable x iff there exists a value j such that, for all other values k, $f_x^{\{k\}} \supseteq f_x^{\{j\}}$
  - A cover F is weakly unate in x iff subset of all implicants depend on variable x has a column of 0's in the x field
- A cover F is weakly unate only if f is weakly unate

11 10
01 11
01 10

$y'+x+xy'$

$f_x^{\{1\}}=1 \supseteq f_x^{\{0\}}=y'$

$f_y^{\{0\}}=1 \supseteq f_y^{\{1\}}=x$

---

11 101
01 111
01 001

$y^{\{0,2\}}+x^{\{1\}}y^{\{0,1,2\}}+x^{\{1\}}y^{\{2\}}$

$f_x^{\{1\}}=1 \supseteq f_x^{\{0\}}= y^{\{0,2\}}$
$f_y^{\{2\}}=1 \supseteq f_y^{\{1\}}=x^{\{1\}}$
$f_y^{\{0\}}=1 \supseteq f_y^{\{1\}}=x^{\{1\}}$

Weakly unate in all variables



## Tautology

- Check if a function is always TRUE (1)
- Recursive paradigm
  - $f = x\, f_x + x'\, f_{x'}$
    - So if both $f_x$ and $f_{x'}$ are 1, then f=1
    - Therefore, recursively testing to see if $f_x$ and $f_{x'}$ are tautology
  - Expand about a (possibly mv)-variable to break problem into small ones
  - If all cofactors are tautology, then function is a tautology
    - f = x + x'
    - $f_x=1$, $f_{x'}=1$, therefore, f=1
  - If any of the cofactors are $\varnothing$, then function is not a tautology
    - Can terminate very quickly
    - f = x + x
    - $f_x=1$, $f_{x'}=\varnothing$, therefore, f is not a tautology
  - Need to define other end cases of recursion

## Tautology

- Tautology of unate cover is easy!!!
  - If F is a weakly unate cover in variable x and G is the subset of F that does not depend on x, then F is a tautology if and only if G is a tautology
    - y'+x+xy' 11 10
    - Not a tautology 01 11
      01 10

    A tautology? 11 101
    01 111
    11 010
  - Expand about a mv-variable
    - Heuristic: choosing one that has the most implicants depend on it
      - Not all 1's
    - Trying to break problem into unate subproblems



## Recursive tautology

- End Cases:
  - The cover has a row of all 1's, e.g. [11 11 111]
    - Return Tautology
  - The cover has a column of 0's
    - Return NOT tautology

    01 11 11
    01 10 11
    a+ab'
  - The cover depends on one variable, and there is no 0 column
    - Return Tautology

    10 11 111
    01 11 111
    a'+a



## Example

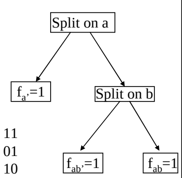
- f=ab+ac+ab'c'+a'
  - All variables are binate
  - a affect the most column
    - No "11"
  - Take cofactor w.r.t. a'=10 11 11
    - $f_{a'}$= 11 11 11
  - Take cofactor w.r.t. a=01 11 11
  - Still binate, split along b
  - Take cofactor w.r.t. b'=11 10 11
    - A tautology
  - Take cofactor w.r.t. b=11 01 11
    - A tautology
- Therefore, f is a tautology!

01 01 11
01 11 01
01 10 10
10 11 11

11 01 11
11 11 01
11 10 10

11 11 01
11 11 10

11 11 11
11 11 01






## Example

- F=ab+ac+a'
  - Unate in b
    - Check submatrix doesn't depend on b
  - Unate in c
    - Check submatrix doesn't depend on c
    - [10 11 11] is not a tautology
    - Submatrix doesn't depend on b is not a tautology
    - F is not a tautology

01 01 11
01 11 01
10 11 11

01 11 01
10 11 11



## Containment

- A cover F contains an implicant $\alpha$ if and only if $F_\alpha$ is a tautology
- E.g. f= ab+ac+a'

  01 01 11
  01 11 01
  10 11 11
  - Want to know if implicant bc [11 01 01] is contained in f
  - Take cofactor, obtain

    01 11 11
    01 11 11
    10 11 11
  - Which is a tautology
    - Therefore, bc is contained in F



## Complementation

- Recursive paradigm
  - $F'=x f'_x + x' f'_{x'}$
    - Proof: let $g= x f'_x + x' f'_{x'}$
    - $f=xf_x+x'f_{x'}$
    - $fg= (x f_x + x' f_{x'})(x f'_x + x' f'_{x'})= 0$
    - $f+g=x(f_x+f'_x)+x'(f_{x'} +f'_{x'})=1$
    - Therefore g=f'
- Steps:
  - Select variable
  - Compute cofactors
  - Complement cofactors
  - Assemble the complement
- Recur until cofactors can be complemented in a straightforward way

## Termination rules

- The cover F is void, its complement is the universal cube
  - 0'=1
- The cover F has a row of 1's, its complement is void
  - 1'=0
- The cover F consists of one implicant, use De Morgan's law
  - (ab'cdf')' = a' + b + c' + d' + f
- All implicants of F depend on a single variable, and there is no column of 0's, it's complement is void

00111 11
10110 11
01000 11

Tautology, therefore the complement is void



## Unate functions

- Useful shortcuts
  - If f is positive unate in x, $f' = f'_x + x' f'_{x'}$
    - $f' = x f'_x + x' f'_{x'} = f'_x + x' f'_{x'}$ because $f_x \supseteq f_{x'}$, $f'_{x'} \supseteq f'_x$
  - If f is negative unate in x, $f' = x f'_x + f'_{x'}$
- Consequence:
  - Complement computation is simpler
- Heuristic
  - Select variables to make the cofactors unate
    - Select binate variable

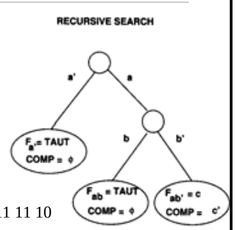


## Example

- F=ab+ac+a'
- Select binate variable a
- Compute cofactors
  - $F_{a'}$ is a tautology, hence $F'_{a'}$ is void
  - $F_a$ yields
- Select variable b
- Compute cofactors
  - $F_{ab}$ is a tautology, Hence $F'_{ab}$ is void
  - $F_{ab'}$ = 11 11 01 and its complement is 11 11 10
- Re-construct complement
  - 11 11 10 intersected with b'=11 10 11 yields 11 10 10
  - 11 10 10 intersected with a=01 11 11 yields 01 10 10
  - Complement F' – 01 10 10

01 01 11
01 11 01
10 11 11

11 01 11
11 11 01



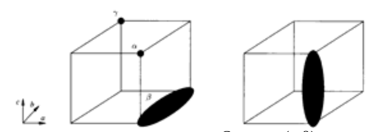



## Consensus

- If two implicants has distance larger or equal to 2
  - Consensus return 0
- If two implicants has distance equal to 1
  - Consensus return a single implicant
    - Parts that is "really close"

$$CONSENSUS(\alpha,\beta) = \begin{cases} a_1 + b_1 & a_2 \cdot b_2 & \dots & a_n \cdot b_n \\ a_1 \cdot b_1 & a_2 + b_2 & \dots & a_n \cdot b_n \\ \dots & \dots & \dots & \dots \\ a_1 \cdot b_1 & a_2 \cdot b_2 & \dots & a_n + b_n \end{cases}$$

α 01 10 01
β 01 11 10
γ 10 01 01

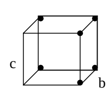


Consensus(α,β)



## Compute all prime

- Given any two orthonormal basis $x^A$ and $x^B$, A prime of f is
  - A prime of $x^A f_x^A$
  - A prime of $x^B f_x^B$
  - A prime of consensus of $x^A f_x^A$ and $x^B f_x^B$
    - $x^A f_x^A$ and $x^B f_x^B$ don't intersect
    - Its consensus is the corresponding part of the minterms that is distance 1
  - May be redundant in a cover (contained in union of multiple prime)
- E.g. = abc+abc'+ab'c+ab'c'+a'bc+a'bc'
  - Let the basis be a and a'
  - $af_a$= a 1 = a
  - $a'f_{a'}$ = a' b
  - Consensus(01 11, 10 01) is 1101 = b
  - SCC (a, a'b, b) = a, b
    - Because union make a'b not prime
  - There are two prime a, b
  - Both are essential, since the function is unate






## Unate

- Unate function
  - All primes are essential
  - A unate cover, minimal w.r.t. SCC, is a prime cover
    - Computing of prime is easy!
- Example f = ab+ac+a' → 01 01 11, 01 11 01, 10 11 11
  - a is binate, split along a
    - $f_{a'}$ = 1, $a'f_{a'}$=a' → 10 11 11
      - Unate, so the only prime is a'
    - $f_a$=b+c, $af_a$=ab+ac → 01 01 11, 01 11 01
      - Unate, so the only primes are ab, ac
    - Consensus(a', ab+ac) = b+c → 11 01 11, 11 11 01
      - Unate, so the only primes are b, c
    - SCC (a', ab, ac, b, c) = a',b,c → 10 11 11, 11 01 11, 11 11 10

## Operations in espresso

- Expand
- Reduce
- Irredundant
- Essential

- Basic operation
  - Tautology
  - Complement
  - Containment
  - Generate all primes (espresso –Dexact)



## Espresso, anyone?

```
ESPRESSO(F^ON, F^DC){
    F^OFF = complement(F^ON ∪ F^DC);
    F = expand(F^ON, F^OFF);
    F = irredundant(F, F^DC);
    E = essentials(F, F^DC);
    F = F − E;                                   /* remove essentials */
    F^DC = F^DC ∪ E;          /* place essentials temporarily in don't care set*/
    repeat {
        φ2 = cost(F);
        repeat {
            φ1 = |F|;
            F = reduce(F, F^DC);
            F = expand(F, F^OFF);
            F = irredundant(F, F^DC);
        } until ( |F| ≥ φ1);
        F = last_gasp(F, F^DC, F^OFF);   /* apply different heuristics as last gasp */
    } until (cost(F) ≥ φ2);
    F = F ∪ E;                                   /* restore essentials */
    F^DC = F^DC − E;
    F = make_sparse(F, F^DC, F^OFF);
}
```



## Espresso algorithm

- Compute the complement
- Extract essentials
- Iterate
  - Expand, irredundant, reduce
- Cost functions:
  - 1. Cover cardinality
  - 2. Weighted sum of cube and literal count
- Make_sparse
  - Reduce output part to reduce OR-plane transistors in PLA
  - http://embedded.eecs.berkeley.edu/pubs/downloads/



## 1