ESPRESSO: Advanced Steps

(i) Essentials #2
(ii) MAKE-SPARSE
ESPRESSO: “Essentials #2” Step

generating all essentials when not given all primes

“Essentials #2” Step

Example: given any “prime cover” F of a Boolean function f
(prime cover = cover using only prime implicants, i.e. fully-expanded cubes)

Q1. Does the cover include all essentials?

<table>
<thead>
<tr>
<th>yz</th>
<th>wx</th>
<th>00</th>
<th>01</th>
<th>11</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>A</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>01</td>
<td>B</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Example: given any “prime cover” $F$ of a Boolean function $f$

Q1. Does the cover include all essentials?

A1. YES! Every prime cover $F$ of a function $f$ includes all its essentials

Example: given any “prime cover” $F$ of a Boolean function $f$

Q2. Which cubes in the cover are essential?
Example: given any “prime cover” F of a Boolean function f

Q2. Which cubes in the cover are essential?

Note: cover F does not necessarily include all primes!

“Essentials #2” Step

Example: given any “prime cover” F of a Boolean function f

Q2. Which cubes in the cover are essential?
Example: given any “prime cover” $F$ of a Boolean function $f$

Q2. Which cubes in the cover are essential?

[Diagram showing a truth table with highlighted cubes and text explaining the concept of essentials.]

... how to identify all essentials when cover $F$ does not include all primes?
**“Essentials #2” Step**

Example: given any “prime cover” $F$ of a Boolean function $f$

<table>
<thead>
<tr>
<th>wx</th>
<th>00</th>
<th>01</th>
<th>11</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>yz</td>
<td>00</td>
<td>A</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>01</td>
<td>B</td>
<td>1</td>
<td>C</td>
</tr>
<tr>
<td></td>
<td>11</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

**Q2. Which cubes in the cover are essential?**

**CASE ANALYSIS:**

3 types of minterms in cubes

Each ON-set minterm is:

(i) covered by 2 or more cubes
Example: given any “prime cover” $F$ of a Boolean function $f$

Q2. Which cubes in the cover are essential?

CASE ANALYSIS:
3 types of minterms in cubes

Each ON-set minterm is:
(i) covered by 2 or more cubes
(ii) covered by only 1 cube

2 sub-cases: ...
Example: given any “prime cover” F of a Boolean function f

Q2. Which cubes in the cover are essential?

CASE ANALYSIS:
3 types of minterms in cubes

Each ON-set minterm is:
(i) covered by 2 or more cubes
⇒ (ii) covered by only 1 cube

2 sub-cases: ...
(a) ON-set minterm “isolated”: no neighboring cubes (only OFF-set)

(b) ON-set minterm not “isolated”: has neighboring adjacent cube(s)
Example: given any “prime cover” F of a Boolean function f

Q2. Which cubes in the cover are essential?

Test for Case #2(b):
not in “isolated regions” = has neighboring cubes

⇒ Grow “consensus” cubes:
... between each adjacent cube pair

X = CONSENSUS (cube B, cube C)
Example: given any “prime cover” $F$ of a Boolean function $f$

Q2. Which cubes in the cover are essential?

Test for Case #2(b):
not in “isolated regions” = Has Cube Neighbors

Intuition: experiment to “grow” missing implicants
- Consensus cube = “seed” which spans gap between adjacent cube pair (non-intersecting)
- Idea: used to locally generate the “core” of missing primes in cover (consensus often is non-prime)

Example: given any “prime cover” $F$ of a Boolean function $f$

A2. If a cube $w$ in cover $F$ is not entirely covered by the union of:
(i) the set of other cubes in cover, and
(ii) the set of consensus cubes (i.e. seeds of missing primes)
then it is essential!
“Essentials #2” Step

Example: given any “prime cover” $F$ of a Boolean function $f$

Q2. Which cubes in the cover are essential?

A2. If a cube $w$ in cover $F$ is not entirely covered by the union of:
   (i) the set of other cubes in cover, and
   (ii) the set of consensus cubes (i.e., seeds of missing primes)
then it is essential!

EXAMPLE #1: Cube A Essential

EXAMPLE #2: Cube C Not Essential
**“Essentials #2” Step**

Example: given any “prime cover” $F$ of a Boolean function $f$

**Q2. Which cubes in the cover are essential?**

A2. Algorithmic Formulation*: ESSEN #2
Given cover $F$, including cube ‘$e$’ (to check if $e$ essential):
1. Remove ‘$e$’ from $F$:
   $G = F - \{e\}$
2. Compute consensus of ‘$e$’ and each cube in $G$:
   $H = \text{consensus}(e, G)$
3. Formulate containment problem:
   Check if $e \leq G \cup H$

*equivalent to Hachtel/Somenzi, Theorem 5.4.1 (p. 204)

**ESPRESSO: “Make-Sparse” Step**
Multi-Output Minimization: Example #1

Multi-Output Function

Cover #1A: min-cost cover, using ONLY multi-output primes
Multi-Output Minimization: Example #1

Cover #1A: using ONLY multi-output primes

Cover #1B: using some NON-PRIMES!

COST = 15 gate inputs

COST = 14 gate inputs = better!
**Observation**

**Peculiar feature of multi-output primes (spanning multiple functions):**

Spanning more functions: (i) increases coverage, BUT (ii) also increases cost!

**ESPRESSO Strategy: using MAKE-SPARSE (simple version)**

**Basic idea:**
- Initial ESPRESSO goal: body of algorithm always 'expands' to multi-output primes
- Post-processing step (at end of algorithm) = MAKE-SPARSE
  - heuristically reduce final cover cost by DELETING UNNECESSARY OUTPUT CONNECTIONS
- Basic version of "MAKE-SPARSE":
  -- this step is a form of multi-output "REDUCE" = eliminate unnecessary AND-gate wire fanouts
  -- also known as "reduce output parts"

**See previous slides:**
- before MAKE-SPARSE: Cover #1A (cost = 15 gate inputs)
- after MAKE-SPARSE: Cover #1B (cost = 14 gate inputs)
**MAKE-SPARSE: advanced version**

*Used in Espresso-II:*
- Key Idea: improved "MAKE-SPARSE" can involve new expansion too!
  - Step #1. Use restricted "reduce": "REDUCE OUTPUT PARTS"
    - for each AND-gate, eliminate unnecessary output connections
  - Step #2. Use restricted "expand": "EXPAND INPUT PARTS"
    - for each AND-gate, expand if possible within current outputs

---

**Multi-Output Minimization: Example #2**

*Multi-Output Function: illustrates advanced MAKE-SPARSE operation*
Multi-Output Minimization: Example #2

Cover #2A: initial min-cost cover, using ONLY multi-output primes

STEP #1 -- Cover #2B: after RESTRICTED REDUCE of output part
= "REDUCE OUTPUT PART"

NOT a multi-output prime

multi-output prime
Multi-Output Minimization: Example #2

STEP #2 -- Cover #2C: after RESTRICTED EXPAND of input part
- no expansion to new outputs, only within current outputs!
  = "EXPAND INPUT PART"

FINAL COVER

Multi-Output Minimization: Example #2 -- details

Cover #2A: using ONLY multi-output primes

(COST = 15 gate inputs)

Final PLA Representation = "cubical complex"
Multi-Output Minimization: Example #2 -- details

STEP #1 -- Cover #2B: REDUCE OUTPUT PART

- PLA Representation of Cover = "cubical complex"

COST = 14 gate inputs = better!

STEP #2 -- Cover #2C: EXPAND INPUT PART

- PLA Representation of Cover = "cubical complex"

COST = 13 gate inputs = even better!