Synthesis and Verification of Finite State Machines

Sungho Kang

Yonsei University

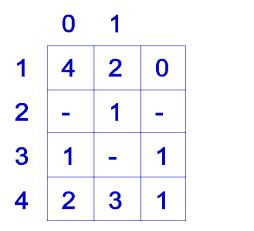
Outline

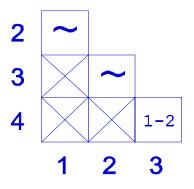
- Minimization of Incompletely Specified Machines
- Binate Covering Problem
- State Encoding
- Decomposition and Encoding

- We have learned basic methods for minimizing, encoding, checking equivalence, and synthesizing circuits for realizing completely specified FSMs
- Now we must learn to deal with the more practical case of incomplete specification
- Our goal is thus to find a least cost circuit that satisfies a partial specification

Use don't-cares to merge states.

Merged states must have same output sequences.

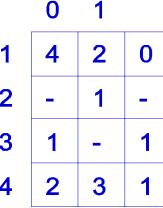


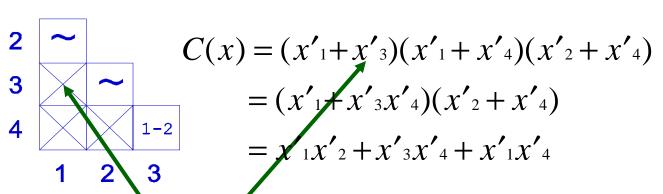


Flow Table Compatibility Table
Note each constraint represents pair
(incompatibility)

- Derive all prime sets of compatible states
- Solve a covering problem to obtain minimum states.

Compatibility relation: conjunction of constraints (one for each "X")





Note each constraint represents pair (incompatibility)

$$(x'_1 + x'_3) \Leftrightarrow (x'_1 \Rightarrow x'_3) \Leftrightarrow (x'_3 \Rightarrow x'_1)$$

By recursive multiplication method, like computing the Complete Sum:

$$C(x) = (x'_1 + x'_3)(x'_1 + x'_4)(x'_2 + x'_4)$$

$$= (x'_1 + x'_3 x'_4)(x'_2 + x'_4)$$

$$= x'_1 x'_2 + x'_3 x'_4 + x'_2 x'_3 x'_4 + x'_1 x'_4$$

$$= x'_1 x'_2 + x'_3 x'_4 + x'_1 x'_4$$

The (complete) constraint sums are multiplied out, dropping absorbed terms when they arise.

$$x'_{1}x'_{2} + x'_{3}x'_{4} + x'_{1}x'_{4}$$

 $x'_{1}x'_{2} \Rightarrow \{S_{3}, S_{4}\}$

Maximal compatibles are "Prime".

(No superset of these state sets are also pairwise c ompatible).

 $e.g., x'_1 \Rightarrow \{s_2, s_3, s_4\}$ but $\{s_2, s_4\}$ are not compatible

- Unfortunately, some subsets of the maximal compatibles pairs are also prime compatibles.
- Because, selection of one compatible pair may imply selection of other compatible pairs.

$$\{S_3, S_4\} \Longrightarrow \{S_1, S_2\}$$

• A compatible C_s is prime if and only if there is no other compatible C_q which contains it or whose class s et Γ_q contains class set Γ_q of C_s That is, C_s is prime if and only if

$$\neg \exists C_q$$
 such that
$$(1) \ C_q \supset C_s$$

$$(2) \ \Gamma_s \supseteq \Gamma_q$$

(Bigger compatible, smaller class set)

Subsets with smaller class sets are acceptable.

- In minimization, we desire a minimum number of compatible sets that cover all original states. Pick fr om primes.
- Choice of conditionally compatible set implies choosing all implied pairs.
- Set of implied compatibles pairs is called the class set, e.g., $\{S_1, S_2\}$ is the class set of $\{S_3, S_4\}$

$$CS_{(s,t)} = \{(s,t)\}$$

- We just derived maximal compatibles that are prime
- Derive remaining prime compatibles
- Solve a covering problem

$$\Gamma((a,b)) = \{(a,d)\} \\ \Gamma((b,e)) = \{(d,e),(a,b),(a,e)\}$$

$$\begin{bmatrix} a \\ b \\ b,0 \\ d,1 \\ a,- \\ c \\ b,0 \\ d,1 \\ a,- \\ c \\ b,0 \\ d,1 \\ a,- \\ a,-$$

f,0

$$\Gamma(\{c,f,g\}) = \{(c,d),(e,h)\}$$

$$\Gamma(\{c,f\}) = \{(c,d)\}$$

Note $\{c, f\}$ is prime: although $\{c, f, g\} \supset \{c, f\},\$

$$\Gamma(\{c,f\}) \subset \Gamma(\{c,f,g\})$$

d,e a,b a,e, a,b a,d X

a,d

X

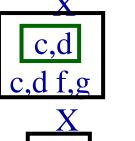
h

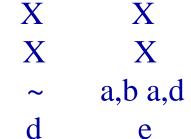
d

X

X b

a





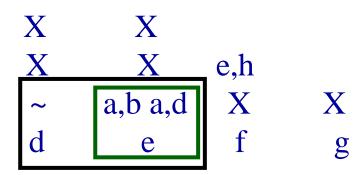
$$\Gamma(\{d,e,h\}) = \{(a,b),(c,d)\}$$

$$\Gamma(\{e,h\}) = \{(a,b),(c,d)\}$$

Note $\{e,h\}$ is not prime:

$$\{d,e,h\}\supset\{e,h\},$$

$$\Gamma(\lbrace e,h\rbrace)\supseteq\Gamma(\lbrace d,e,h\rbrace)$$



$$\Gamma(\{a,b\}) = \{(a,d)\}$$
 $\Gamma(\{b,e\}) = \{(d,e),(a,b),(a,e)\}$
 $\Gamma(\{a,b,e\}) = \{(a,d),(d,e)\}$
 $\Gamma(\{a,b,d,e\}) = \emptyset$

$$\Gamma(\{c,f\}) = \{(c,d)\}$$

$$\Gamma(\{c,f,g\}) = \{(c,d),(e,h)\}$$

```
h
     a,d
      X
d
      b,e
              a,b d,e
                         d,e a,g
             d,e a,b a,e,
    a,b a,d
                           c,d
       X
                 X
                         c,d f,g
                 X
       X
                  b
       a
```

Note $\{c, f\}$ is prime: $\{c, f, g\} \supset \{c, f\}$, but $\Gamma(\{c, f\}) \subset \Gamma(\{c, f, g\})$

X X e,h
~ a,b a,d X X
d e f g

maximal		class	
compatibles		set	
1	${a,b,d,e}$	{}	
2	$\{b,c,d\}$	$\{\{a,b\},\{a,g\},\{d,e\}$	}
3	$\{c,f,g\}$	$\{\{c,d\}, \{e,h\}\}$	
4	$\{d,e,h\}$	$\{\{a,b\}, \{a,d\}\}$	
11	$\{a,g\}$	{}	
oth	ner PCs		
5	{b,c}	{}	Note sub-compatibles $\{b,c\}$
6	$\{c,d\}$	$\{\{a,g\}, \{d,e\}\}$	through $\{d,h\}$ are added to the
7	$\{c,f\}$	$\{\{c,d\}\}$	
8	$\{c,g\}$	$\{\{c,d\}, \{f,g\}\}$	list of prime compatibles
9	$\{f,g\}$	$\{\{e,h\}\}$	before maximal compatible
10	$\{d,h\}$	{}	$\{a,g\}$
10	(f)		$[\alpha, \delta]$

12 {f}

```
maximal
               class
compatible
             set
   \{a,b,d,e\}
              {}
2 \{b,c,d\} \{a,b\},\{a,g\},\{d,e\}\}
3 \{c,f,g\} \{\{c,d\},\{e,h\}\}
\{d,e,h\} {{a,b}, {a,d}}
11 \{a,g\}
other PCs
5 {b,c}
         {}
\{c,d\} \{\{a,g\},\{d,e\}\}
7 \{c,f\} \{\{c,d\}\}
\{c,g\} \{\{c,d\},\{f,g\}\}
9 \{f,g\}
        \{\{e,h\}\}
10 \{d,h\}
12 {f}
```

Note that subsets $\{b,d\}$ and $\{d,e\}$ are not prime because they are contained in $\{a,b,d,e\}$, which has an empty class set

```
maximal
               class
compatible
             set
   \{a,b,d,e\}
              {}
2 \{b,c,d\} \{a,b\},\{a,g\},\{d,e\}\}
3 \{c,f,g\} \{\{c,d\},\{e,h\}\}
\{d,e,h\} \{\{a,b\},\{a,d\}\}
11 \{a,g\}
other PCs
5 {b,c}
            {}
\{c,d\} \{\{a,g\},\{d,e\}\}
7 \{c,f\} \{\{c,d\}\}
\{c,g\} \{\{c,d\},\{f,g\}\}
9 \{f,g\}
         \{\{e,h\}\}
10 \{d,h\}
12 {f}
```

Note that subset $\{e,h\}$, with class set $\{\{a,b\},\{a,d\}\}$, is not prime because it is contained in $\{d,e,h\}$, whose class set is the same. $A\exists q$ such that

 $(1) q \supset s$

(2) $\Gamma_s \supseteq \Gamma_a$

```
maximal
              class
compatible
              set
   \{a,b,d,e\}
```

2 $\{b,c,d\}$ $\{a,b\},\{a,g\},\{d,e\}\}$

 $11 \{a,g\}$ other PCs

 $\{c,g\}$ $\{\{c,d\},\{f,g\}\}$

 $10 \{d,h\}$ 12 {f}

After treating subsets of size 2, we still have to check all subsets of size 1, which have

empty class sets.

Note

 ${a},{b},{c},{d},{e},{g}$ are all contained in primes

with empty class sets, so they are not prime.

But $\{f\}$ is not, so it is prime.

```
maximal
                class
compatible
                 set
   \{a,b,d,e\}
              { }
2 \{b,c,d\} \{a,b\},\{a,g\},\{d,e\}\}
3 \{c,f,g\} \{\{c,d\},\{e,h\}\}
\{d,e,h\} \{\{a,b\},\{a,d\}\}
11 \{a,g\}
other PCs
5 {b,c}
6 \{c,d\}
          \{\{a,g\}, \{d,e\}\}
7 \{c,f\}
         \{ \{ c,d \} \}
          \{\{c,d\}, \{f,g\}\}
8 \{c,g\}
9 \{f,g\}
          \{\{e,h\}\}
10 \{d,h\}
12 {f}
```

Maximal Compatibles are prime.

Other prime compatibles are **subsets** of primes such that:

s is prime iff its class set does not contain the class set of a larger prime $s' \supset s$.

e.g., $\{e,h\} \rightarrow \{(a,b),(a,d)\}$ is not prime

```
Procedure(MAXCOMPS,CM) {
   p = LARGEST(MAXCOMPS); k_{max} = |p|
  for(k = k_{max}; k \ge 1; k - -)  {
      Q = SELECT_BY_SIZE(MAXCOMPS, k)
                                                     Enqueue known
      \mathbf{for}(q \in Q) \text{ ENQUEUE}(P,q)
      foreach(p \in P; |p| = k) {
                                                     primes
        CS_p = CLASS\_SET(CM, p)
                                                     of size k
        if (CS_p = \emptyset) continue
        S_p = MAX_SUBSETS(p)
        \mathbf{for}(s \in S_p) {
        if (DONE(s)) continue
        CS_s = CLASS\_SET(CM, s)
        prime = 1
        foreach(q \in P; |q| \ge k) {
                                                    Test subcompatibles
           if (s \subset q) {
                                                    for primality
             CS_q = CLASS\_SET(CM, q)
             if (CS_s \supseteq CS_q) { prime = 0; break}
         if (prime = 1) ENQUEUE(P, s)
         HASH_TABLE_INSERT(DONE, s)
```

```
Procedure (MAXCOMPS, CM) {
    p = \text{LARGEST}(MAXCOMPS); k_{\text{max}} = |p|

1    \mathbf{for}(k = k_{\text{max}}; k \ge 1; k - -) {
    Q = \text{SELECT\_BY\_SIZE}(MAXCOMPS, k)

    \mathbf{for}(q \in Q) \text{ ENQUEUE}(P, q)

2    \mathbf{foreach}(p \in P; |p| = k) {
    CS_p = \text{CLASS\_SET}(CM, p)

    \mathbf{if}(CS_p = \emptyset) \text{ continue}

    S_p = \text{MAX\_SUBSETS}(p) the maximum of the maximum of the second or the se
```

For each value of k, the for-loop of Line 1 puts the maximal compatibles of size k onto the queue of primes, P.

For k = 4, only $\{a,b,d,e\}$ is enqueued For k = 3, $\{b,c,d\}$, $\{c,f,g\}$, $\{d,e,h\}$ are enqueued

```
S_p = \text{MAX\_SUBSETS}(p)
\mathbf{for}(s \in S_p) {
if (DONE(s)) continue
CS_s = CLASS\_SET(CM, s)
prime = 1
foreach(q \in P; |q| \ge k) {
   if (s \subset q) {
      CS_q = CLASS\_SET(CM,q)
      if (CS_s \supseteq CS_a) { prime = 0; break}
 if (prime = 1) ENQUEUE(P, s)
 HASH_TABLE_INSERT(DONE, s)
```

For each enqueued prime p (of size k), we check every subset of size k-1.

s is a prime com patible if and only if

 $\neg \exists q \text{ such that}$ $(1) \ q \supset s$ $(2) \ \Gamma_s \supseteq \Gamma_q$

$$\{c_1, c_4, c_5, c_9\}$$

$$c_1 = \{a, b, d, e\}$$

$$c_4 = \{d, e, h\}$$

$$c_5 = \{b, c\}$$

$$c_9 = \{f, g\}$$

-,1

4,1

9,1

$$c_1 = \{a,b,d,e\}$$

$$c_4 = \{d,e,h\}$$

$$c_5 = \{b,c\}$$

$$c_9 = \{f,g\}$$

Where there is a choice, choose 1 (as in x2-successor of compatible 1): $\{d,e\}$ contained in C_1 or C_4 .

- Closed Cover: Choosing Compatibles
- Every state of the original machine must be covered
- Every implied compatible must be present in the solution

maximal		class	Let's check if the follow	Let's check if the following							
compatibles		set	set of compatibles forn	set of compatibles forms a							
1	${a,b,d,e}$	{}	closed cover: $\{c_1, c_4, \ldots, c_4, \ldots, c_4, \ldots, c_4, \ldots, c_4, \ldots, \ldots, c_4, $	$\{c_{5},c_{6}\}$							
2	$\{b,c,d\}$	$\{\{a,b\},\{a,g\},\{a,$	d,e }}	$a \in a$							
3	$\{c,f,g\}$	$\{\{c,d\}, \{e,h\}\}$									
4	$\{d,e,h\}$	$\{\{a,b\},\{a,d\}\}$		$b, c \in c$							
11	$\{a,g\}$	{}	Coverage:	$d, e \in c$							
other PCs											
5	$\{b,c\}$	{}		$ f,g \in c$							
	$\{c,d\}$	$\{\{a,g\}, \{d,e\}\}$	Closure:	$h \in c$							
7	$\{c,f\}$	$\{\{c,d\}\}\$	$\Gamma(c_1)$:								
8	$\{c,g\}$	$\{\{c,d\}, \{f,g\}\}\$	1 (61).								
9	$\{f,g\}$	$\{\{e,h\}\}$	$\Gamma(c_4): \{a,b\} \in c_1 \{a,b\} $	$\{d\} \in C_1$							
10	$\{d,h\}$	{}	$\Gamma(c_5)$:								
12	{f}	{}	$\Gamma(c_9): \{e, h\} \in c_4$								

9

maximal	class	
IIIaXIIIIaI	Class	• Eve
compatibles	set	
$1 \{a,b,d,e\}$	{ }	mac
2 {b,c,d}	$\{\{a,b\},\{a,g\},\{d,e\}\}$	
$3 \{c,f,g\}$	$\{\{c,d\}, \{e,h\}\}$	(0.4
$4 \{d,e,h\}$	$\{\{a,b\}, \{a,d\}\}$	$(c_1 +$
$11 \{a,g\}$	{}	$(c_2 \dashv$
other PCs		·
5 {b,c}	{}	$(c_1 +$
6 {c,d}	$\{\{a,g\}, \{d,e\}\}$	$(c_1 +$
$7 \{c,f\}$	$\{\{c,d\}\}$	(C1 1
$8 \{c,g\}$	$\{\{c,d\}, \{f,g\}\}$	$(c_3 \dashv$
$9 \{f,g\}$	$\{\{e,h\}\}$	
10 {d,h}	{}	$(C_4 \dashv$
12 {f}	{}	

Every state of the original machine must be covered.

$$(c_1 + c_{11})(c_1 + c_2 + c_5)$$

$$(c_2 + c_3 + c_5 + c_6 + c_7 + c_8)$$

$$(c_1 + c_2 + c_4 + c_6 + c_{10})$$

$$(c_1 + c_4)(c_3 + c_7 + c_9 + c_{12})$$

$$(c_3 + c_8 + c_9 + c_{11})$$

$$(c_4 + c_{11}) = 1$$

```
class
 maximal
compatibles
                          set
    \{a,b,d,e\}
   \{b,c,d\}
                   \{\{a,b\},\{a,g\},\{d,e\}\}
                   \{\{c,d\},\{e,h\}\}
   \{c,f,g\}
                   \{\{a,b\},\{a,d\}\}
    \{d,e,h\}
11 \{a,g\}
other PCs
    {b,c}
   \{c,d\}
                  \{\{a,g\}, \{d,e\}\}
                  \{\{c,d\}\}
                  \{\{c,d\},\{f,g\}\}
   \{c,g\}
    \{f,g\}
                  \{\{e,h\}\}
    \{d,h\}
12 {f}
```

 Every state of the original machine must be covered.

$$(c_{1} + {}^{a}c_{11})(c_{1} + c_{2}^{b} + c_{5})$$

$$(c_{2} + c_{3} + c_{5} + c_{6} + c_{7})$$

$$(c_{1} + c_{2} + {}^{d}c_{4} + c_{6} + c_{10})$$

$$(c_{1} + c_{2} + c_{4})(c_{3} + c_{7}^{f} + c_{9} + c_{12})$$

$$(c_{3} + c_{8}^{g} + c_{9} + c_{11})$$

$$(c_{4} + c_{10})$$

Finding a Minimum Closed Cover

- Associate a variable c_i to the *ith* prime compatible
 For each s∈ S, form the coverage constraint ∏_{s∈S} (∑_{s∈c_i} c_i)

```
\{a,b,d,e\}
2 \{b,c,d\} \{\{a,b\},\{a,g\},\{d,e\}\}
                                              (c_1+c_2)
\{c,f,g\} {\{c,d\},\{e,h\}\}
  \{d,e,h\}
          \{\{a,b\},\{a,d\}\}
                                          e f g h
 c_1(c_1+c_2)(c_2+c_3)(c_1+c_2+c_3)(c_1+c_4)c_3c_3c_4
           This cover is not closed, since c_2 is excluded
```

 C_{Γ} is the set of prime compatibles with non-empty class sets

```
Note (c_i \Longrightarrow c_j) \iff (c'_i + c_j)
      C_{\Gamma}
                    class sets
1 \{a,b,d,e\}
2 {b,c,d} {{a,b},{a,g},{d,e}} (c'_2+c_1)
                                                             \{a,b\}\subset\{a,b,d,e\}
3 \{c,f,g\}
                  \{\{c,d\},\{e,h\}\}
                                         (c'_2 + c_{11})
                                                            \{a,g\}\subseteq\{a,g\}
4 \{d,e,h\}
                  \{\{a,b\},\{a,d\}\}
                                         (c'_2+c_1+c_4)\{d,e\}\subset\{a,b,d,e\}
11 \{a,g\}
5 {b,c}
                                         (c'_3 + c_4)... {d,e}\subset{d,e,h}
6 \{c,d\}
                  \{\{a,g\},\{d,e\}\}
7 \{c,f\}
                  \{\{c,d\}\}
```

Covering and Closure Constraints--POS FORM

$$(c_1 + c_{11})(c_1 + c_2 + c_5)(c_2 + c_3 + c_5 + c_6 + c_7 + c_8)$$

$$(c_1 + c_2 + c_4 + c_6 + c_{10})(c_1 + c_4)(c_3 + c_7 + c_9 + c_{12})$$

$$(c_3 + c_8 + c_9 + c_{11})(c_4 + c_{11})$$

$$(c'_2 + c_1)(c'_2 + c_{11})$$

$$(c'_2 + c_1 + c_4)(c'_3 + c_2 + c_6)(c'_3 + c_4)(c'_4 + c_1)(c'_6 + c_{11})$$

$$(c'_6 + c_1 + c_4)(c'_7 + c_2 + c_6)(c'_8 + c_2 + c_6)(c'_8 + c_3 + c_9)$$

$$(c'_9 + c_4) = 1$$

```
class sets
                                     For each pair p_i in the class
   \{a,b,d,e\}
                \{a,b\},\{a,g\},\{d,e\}\} set of each compatible c_i,
  {b,c,d}
   \{c,f,g\}
                \{\{c,d\},\{e,h\}\}
                                      form the clause
                \{\{a,b\},\{a,d\}\}
    {d,e,h}
                                                   c_i' + \sum_k c_k
11 \{a,g\}
5 {b,c}
                                      where k ranges over the
6 {c,d}
                \{\{a,g\}, \{d,e\}\}
                                     indices of compatibles that
7 \{c,f\}
                \{\{c,d\}\}
                                      contain p_i.
 (c'_2 + c_1)
                           c_1 c_2 c_3 c_4 c_5 c_6 c_7 c_8 c_9 c_{10} c_{11} c_{12}
 (c'_2 + c_{11})
(c'_2+c_1+c_4)
```

Cover rows by including a 1-col OR excluding a 0-col

$c_i' \Longrightarrow c_j$		\boldsymbol{c}	\boldsymbol{c}	c_{\perp}	\boldsymbol{c}	c_{\perp}	\boldsymbol{c}	C_{-}	\mathcal{C}_{a}	\mathcal{C}_{\perp}	<i>c</i> 10	c_{\perp}	c_{\perp}
$c_{i} - c_{j}$ $c_{2} + c_{1}$ $c_{2} + c_{11}$ $c_{2} + c_{1} + c_{4}$ $c_{3} + c_{4}$ $c_{3} + c_{2} + c_{6}$ $c_{4} + c_{1}$ $c_{5} + c_{11}$ $c_{6} + c_{11}$ $c_{6} + c_{1} + c_{4}$ $c_{7} + c_{2} + c_{6}$ $c_{8} + c_{1} + c_{6}$	2	c_{1}	$c \\ 0$	3	4	5	6	7	8	9	10	11	12
$c_2^2 + c_{11}$	2		0									1	
$c_2^7 + c_1^7 + c_4$	2	1	0		1								
$c_3^7 + c_4^7$	3		1	0			1						
$c_3' + c_2' + c_6$	3			0	1								
$c_{4}' + c_{1}$	4	1			0								
$c_{4}' + c_{1}$	(d) 4	1			0								
$c_{6}' + c_{11}$	6 F						0					1	
$c_{6}' + c_{1} + c_{4}$	6 6	1			1		0						
$c_{7} + c_{2} + c_{6}$	w 7		1				1	0					
$c_{8} + c_{2} + c_{6}$	(m) 8		1				1		0				
$c_8 + c_2 + c_6$ $c_8 + c_3 + c_9$	(m) 8			1					0	1			
$c_9' + c_4$	9				1					0			



Find a minimum set of columns which cover all rows: {1,4,5,9}

Closure

Constraints

A row is covered by either including a 1-col or excluding a 0-col.

- Similar to unate covering
- Matrix
 - Variables on columns
 - Sum expressions on the rows
- Solution may not exist when product is 0

- Note: *M* replaced by *F* to emphasize POS semantics
- Also there is one addition (for empty solution space)

```
Procedure BCP(F,\mathcal{U}, currentSol){
      (F, currentSol) = REDUCE(M, currentSol)
      if (terminalCase(F)){
                                                             \setminus \setminus ||F|| = 0
          if (F \neq 0 \text{ and } COST(currentSol) < U){
               \overline{U = COST(currentSol)}
               return (currentSol)
          else return("no (better) solution (in this subspace)")
      L = LOWER\_BOUND(F, currentSol)
     if (L \ge U) return ("no (better) solution (in this subspace)")
     x_i = \text{CHOOSE\_VAR}(F)
                                                              \ \longest column
     S^1 = BCP(F_{x_i}, U, currentSol \cup \{x_i\})
     if(COST(S^1) = L) return(S^1)
     S^0 = BCP(F_{x'_i}, U, currentSol)
     return BEST_SOLUTION (S^1, S^0)
8
```

When x'_2 is <u>essential</u> we say that x_2 is <u>unacceptable</u>

When x'_i is essential, we may delete all rows of the matrix which has a zero in the i^{th} column

$$F = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 \\ \hline 0 & 1 & 0 & - & f_1 \\ - & 1 & 0 & - & f_2 \\ 1 & - & - & 1 & f_3 \\ 1 & 0 & 1 & 0 & f_4 \end{bmatrix}$$

$$(x'_3 + x_2)(x'_3 + x_2 + x'_1)$$

= $(x'_3 + x_2)$

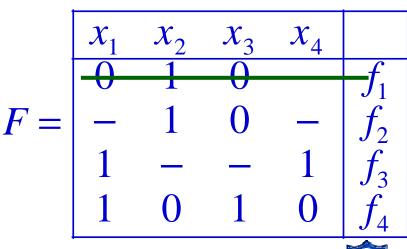
Row 1 (f_1) dominates row 2 (f_2) since row 2 matches row 1 at all care entries. Row 1 may be deleted.

Formally: Row f_1 dominates row f_2 if f_1 is satisfied, in a Boolean sense, whenever f_2 is satisfied, that is, $f_1 \le f_2$

Let F_j and F_k be two columns of F. We say that F_j dominates F_k if, for each row f_i of F, one of the following conditions hold:

- (1) $f_{ij} = 1$
- $(2) \quad f_{ij} = \text{ and } f_{ik} \neq 1$
- (3) $f_{ii} = 0$ and $f_{ik} = 0$

Example: reduced column F_1 dominates F_4



- Two rows are independent if it is not possible to sat isfy both clauses by assigning one variable to 1
- Thus in finding the MIS, we ignore rows (clauses) that contain 0s, since these are satisfied by assigning variables to 0

x_1	x_2	x_3	x_4	
1	1	_	_	f_1
_	1	1	_	$ f_2 $
_	0	_	1	f_3

$$MIS = \{f_1\}$$

x_1	x_2	x_3	x_4	
1	0			$ f_1 $
0	1			$ f_2 $
	0	1	_	$\left f_3 \right $
_	_	0	1	$ f_4 $

cyclic, $MIS = \{\}$

F = 0 cannot occur in original problem (first call to the recursive procedure). But it can happen after one or more recursions:

$$F = \begin{bmatrix} x_1 & x_2 \\ 1 & 1 \\ 0 & 1 \\ 1 & 0 \\ 0 & 0 \end{bmatrix} \equiv (x_1 + x_2)(x_1' + x_2)(x_1 + x_2')(x_1' + x_2') = 0$$
This is detected by REDUCTION, which

This is detected by REDUCTION, which discovers that both x_2 and x_2' are essential

Reduction

Binate Covering

$$f_1$$
 dominates f_2

$$F_1$$
 dominates F_4

$$x_4 = 0$$

x_1	x_2	x_3	\mathcal{X}_4	
0	1	0	_	f_1
_	1	0	_	f_2 f_3
1			1	$ f_3 $
1	0	1	0	$ f_4 $

x_1 is essential

$$F_2$$
 dominates F_3

F_2 dominates F_3				
x_1	x_2	x_3	x_4	
0	1	0	_	$ f_1 $
_	1	0	_	$ f_2 $
1	_	_	1	$ f_3 $
1	0	1	0	f_4

$$x_3 = 0$$

Solution:
 $x = (1,0,0,0)$

- The number of possible assignments is very high
- If one uses k bits to encode p states, there are (2^k)!/(2^k - p)! possible assignments
- If one considers two assignments obtained by permutation or complementation of some of the bits as essentially the same assignment, then there are (2^k-1)! / (2^k p)! k! distinct assignments

- Mustang tries to identify pairs of states by receiving adjacent pairs
 - Two codes are adjacent if they only differ in one bit
- The first objective is to build a graph representing the attraction between each pair of states
 - Two states that have a strong attraction should be given adjacent codes
- How to build attraction graph
 - In the fanout-oriented algorithm, whenever two states, si and sj have a common fanout state, the weight of the edge (si, sj) of the attraction graph is increased
 - In the fanin-oriented algorithm, if si and sj have a common fanin state, the weight of the edge (si, sj) of the attraction graph is increased
 - Once the graph of the attractions is found, we try to assign codes to pairs of states that have strong attractions

- Build two matrices
 - The first with one row for each present state and one column for each next state
 - The second with one row for each present state and one column for each output

- Assign codes to states
 - Select first the node for which the sum of the weights of the Nb heaviest incident edges is maximum

- Build two matrices
 - The first with one row for each next state and one column for each present state
 - The second with one row for each next state and two columns for each output
 - One column is for the true input and the other is for the complement

- Rather than aiming directly at minimizing the number of literals in the next-state functions, one may actually try to minimize the support of the functions
- Reduction of the number of literals and simplification of the interconnections

- A partition π is on a set S is a collection of disjoint subsets of S whose set union is S, i.e. π = { B_a } such that B_a \cap B_b = Φ for a \neq b and \cup { B_a} = S
- Each subset is called a block of the partition
- If π_1 and π_2 are partitions on S, then π_1 π_2 is the partition on S such that $s \equiv t(\pi_1$ π_2) if and only if $s \equiv t(\pi_1$) and $s \equiv (\pi_2)$, whereas, $\pi_1 + \pi_2$ is the partition on S such that $s \equiv t(\pi_1 + \pi_2)$ if and only if there exists a sequence in S $s = s_0$ s_1 s_2 ... $s_n = t$ for which either $s_i \equiv s_{i+1}$ (π_1) or $s_i \equiv s_{i+1}$ (π_2) , $0 \le i \le n-1$

- A partition π on the set of states of the machine is said to have the substitution property if and only if $s \equiv t(\pi)$ implies that $\delta(s,a) \equiv \delta(t,a)$ (π) $\forall a \in I$
- A sequential machine M has a non-trivial parallel decomposition of its state behavior if and only if there exist two nontrivial S.P. partitions π_1 and π_2 on M such that π_1 π_2 = 0
- Independent component
- Dependent component

- First generate the minimal SP partitions and then sum them until considering all possible sums
- The minimal partitions are those obtained by requiring that two states only are included in a block

- Need to resort to something more general than SP partitions, namely, partition pairs
- A partition pair (π, π') on the machine is an ordered pair of partitions on S such that

$$s \equiv t(\pi)$$
 implies that $\delta(s,a) \equiv \delta(t,a)$ (π ') $\forall a \in I$

- The knowledge of the block of π containing the present state and of the current input allows one to compute the block π ' of that will contain the next state.
- It is evident that if (π, π) is a partition pair, then π has substitution property
 - Partition pairs generalize SP partitions