State Assignment: Rules vs. Partition Pairs

Jacob Boles

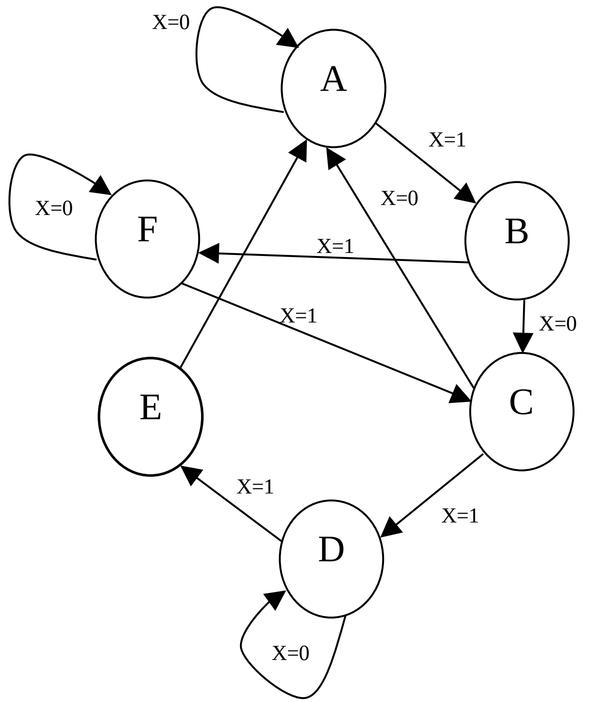
Ece 572

Fall 99

Introduction

- In this presentation I will show an example of state assignment by heuristic rules and compare it to the assignment down by partition pairs.

- So that my example is more relevant and unique, I will use the simplified state machine from my project.



CS	NS	
	X=0	X=1
A	A	B
B	C	F
C	E	D
D	D	E
E	A	A
F	F	C

State Assignment by Rules

- Rule 1
  - States with most incoming branches should be assignment least number of 1’s in code.

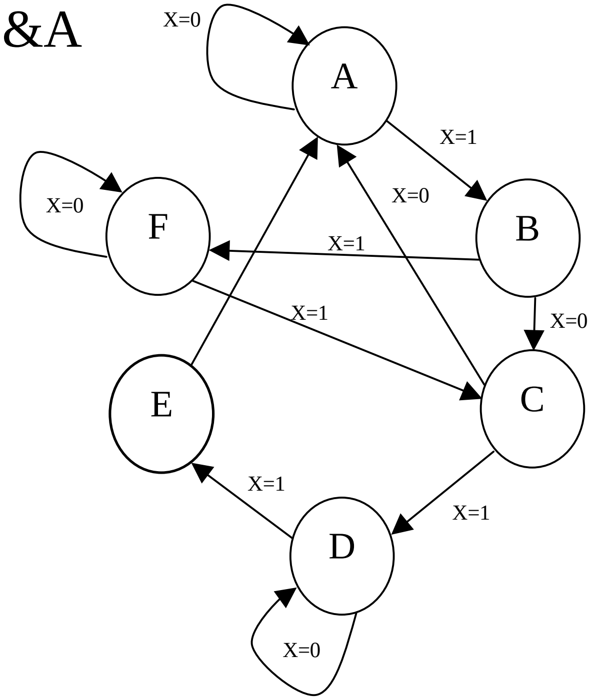
  - This implies that state A which has the most incoming branches by far should be zero. All the other states have about the same number of incoming branches so we take no precedence

A <= 000

State Assignment by Rules

- Rule 2
  - State with common next state on the same input condition should be assigned adjacent codes.
  - In my example this only occurs for E&C&A

E & C & A should be adjacent to each other

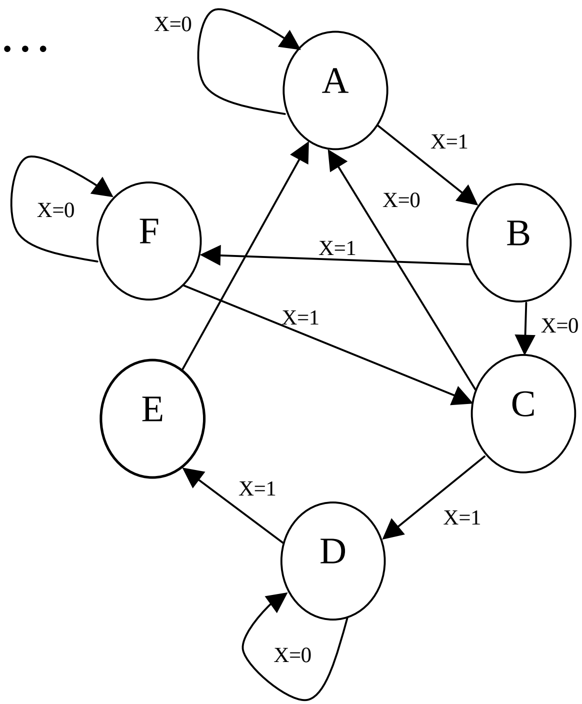


State Assignment by Rules

- Rule 3
  - Next state of same state should be adjacent codes according to adjacency of branch conditions.
  - This is a little harder to see but implies …

Impossible to do all these with 3 bits!

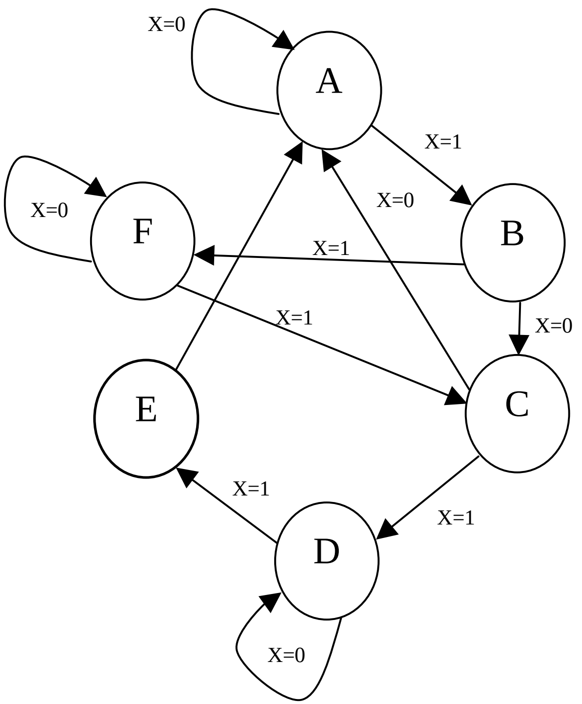
A adj. B
A adj. D
D adj. E
F adj. C



State Assignment by Rules

- Rule 4
  - States that form a chain on same branch should be adjacent codes.

Two chains:
Chain A->B->F->C->D->E
Chain B->C->A



State Assignment by Rules

- Our assignment …

Impossible w/o violating Rule 1

Violates only 1 rules

A 000
B 001
C 010
D 100
E 101
F 011

Rule 1:
A <=000

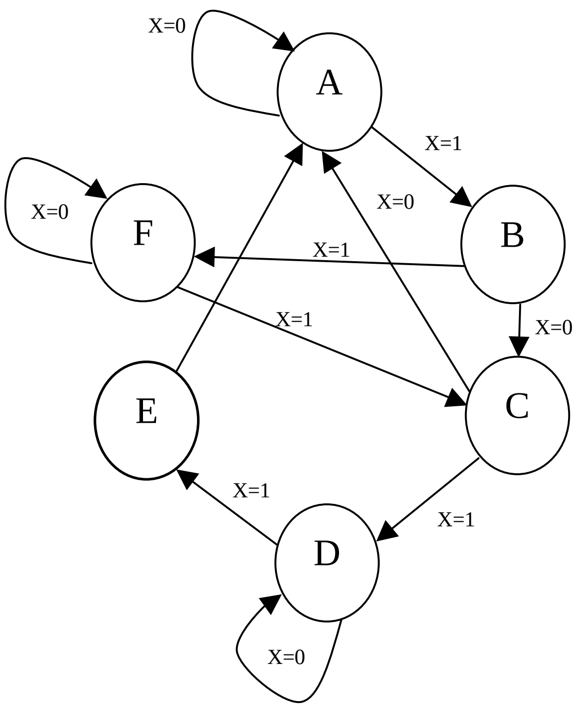
Rule 2:
E&C&A adjacent

Rule 3
A adj. B
A adj. D
D adj. E
F adj. C

Rule 4:
Chain A->B->F->C->D->E
Chain B->C->A

Violates this rule

State Assignment by Rules



A 000
B 001
C 010
D 100
E 101
F 011

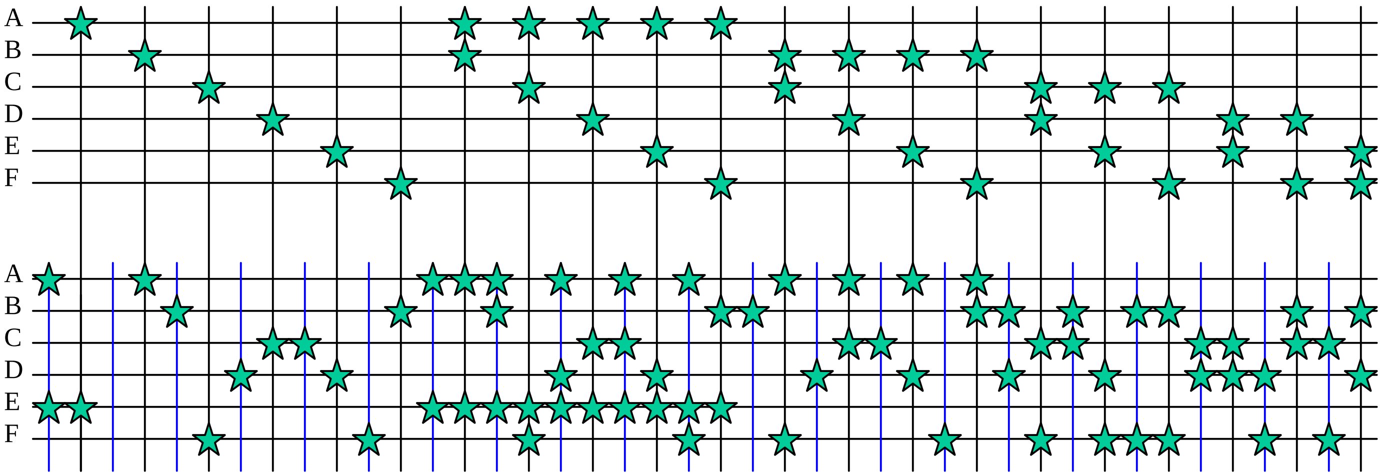
Q0 = XC + X'D + [XD]

Q1 = X'B + XF + [X'F + XB]

Q2 = XA + [XD] + [X'F + XB]

Assuming sharing of common logic:
gates = 5+4+3 = 12

State Assign. using Partitions



CS	NS	
	X=0	X=1
A	A	B
B	C	F
C	E	D
D	D	E
E	A	A
F	F	C

(A, BCDF, E) $\rightarrow$ T_A $\rightarrow$ T_B $\nearrow$ T_E

(AE, BCDF) $\rightarrow$ T_{AB}

(ACDE, B, F) $\rightarrow$ T_C

(AB, CD, E, F) $\rightarrow$ T_{AC}

(AD, BF, C, E) $\rightarrow$ T_{AD}

(AC, BF, D, E) $\rightarrow$ T_{AE}

(AF, B, CD, E) $\rightarrow$ T_{AF}

(AF, B, CDE) $\rightarrow$ T_{BC}

(ABEF, C, D) $\rightarrow$ T_D

(ACDE, B, F) $\rightarrow$ T_F

(AC, BEF, D) $\rightarrow$ T_{BD}

(AD, BEF, C, D) $\rightarrow$ T_{BE}

(A, B, CEF, D) $\rightarrow$ T_{BF}

(AE, BD, CF) $\rightarrow$ T_{CD}

(AE, BC, DF) $\rightarrow$ T_{CE}

(AE, BF, CD) $\rightarrow$ T_{CF}

(ABE, C, D, F) $\rightarrow$ T_{DE}

(ADE, B, C, F) $\rightarrow$ T_{DF}

(AE, BD, CF) $\rightarrow$ T_{EF}

Not very good partition. Doesn’t tell us anything w/o trying every possibility.

Comparison of results

	Rules and heuristics	Partitioning
Advantages	• Easy to do • Fast • Efficient for small problems with limited number of variables	• Will always find best solution if given time • Better than trying every possibility
Disadvantages	• Rules may not always hold true • Inefficient for large variable problems.	• More complex • Can be slow if problem is large or bad partition