Logic: Scoring Efficiency of Circuits

Talk B. Dunham March 28, '58

Truth Table

Function

Binary (of inputs)

Subfunctions

Full function

Duplication of inputs

Interchange { equivalence classes }

p	q	R	
1	1	1	1
0	1	1	0
1	0	1	0
0	0	1	1
1	1	0	1
0	1	0	0
1	0	0	0
0	0	0	0

10001 1000

if R is biased ~~on~~ on get "if and only if." ≡ , if off get "and"

6 ways of biasing 3 input fns.

If q is on get 1 0 1 0 a 2 variable fn. reduced to 1 variable fn "various variable"

q is ~~off~~ get ⊅ "not if, Then"

If p is on, get 1 0 1 0

p is ~~off~~ get ⊅

got 3 full sub-functions

Duplication can put same signal in in two places — take pairs + duplicate

take p+R some values 1 0 0 0 same as "and" again

p+q " 1 1 1 0 "inclusive or" one more sub. fn.

q+R " 1 0 0 0 "and" again

Interchange:

P	q	⊃	
1	1	1	1
0	1	1	0
1	0	0	1
0	0	1	1

"if, Then" output 1101.

1011 is "interchange equivalent".

interchange p + q.

Efficiency ≡ count subfunctions, but leave out interchange classes

Ref: "Multipurpose Bias ~~Logic~~ Device" paper Dunlap: part I.

Term: primed octal nos.

(use these instead of using binary nos.)

— truth tables are in 4 bits, however

1011 0110 Take 3 & skip, Then put prime or no prime

5 | 3

write 5'3

What set of functions are best to build a machine from? What set is most efficient?

compare 3 types  no. of boxes needed to build circuits

evaluate:

First score — no duplication of inputs

Second score — with duplication of inputs.

can build expressions out of "neither-nor" only — very long eg.

using "and, or, not" — get much shorter eg.

— mirror image — (read most significant as least)

— denial — (change all 0's to 1's & 1's to 0's) If a sub-fns has score of 4, then denial will also have 4

— cycle — (class of these images & denials) mirror image also has same score — hard to prove, but so,

1101 0101 → 1010 1011

no. of wrys 2^{2^n} n = no of variable 256 fns for 3 variables.

— scoring all 3 variable fns. only one has score of 6 (= = =) all others are part of some cycle.

Example of 6 score

P	Q	R	output
1	1	1	1
0	1	1	1
1	0	1	0
0	0	1	1
1	1	0	0
0	1	0	0
1	0	0	1
0	0	0	0

get ⊃ ≠ ↔ =

duplicate inputs add ∨ •

for 4 variables : 65 K subfns.

max score could be 14, (8 singles, 6 pairs)

There are actually 2 cycles which have 14 score

for 5 variables : — use search program "change bits" prog,

use one set, change one bit & rescore go up to dead end,

max. score = 40 out of a possible 68. although 0 1st scores of 36 are higher

2 or 4 cycles.

2nd scores 5 are up to 54,

for 6 variables :

68, 60 have found cases of these.

2nd scoring not yet done.

Many output case;   etc.

Example : full adder.

80 cases { 45 require 3 units
35 " 2 or less units,

also the simpler ones seem to take less components too.