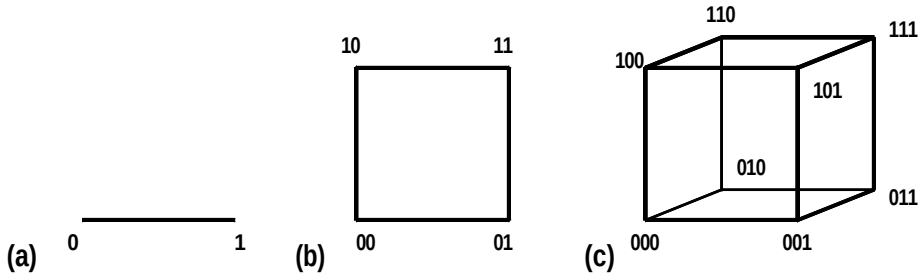
- The ESPRESSO program is an example of a HEURISTIC algorithm.

- The espresso algorithm consists of three basic steps called:
  - Expand
  - Irredundant cover.
  - Reduce.

- A program calles SIS is available on the net which implements the ESPRESSO algorithm,

Some problems are said to be NP-complete.
Traveling salesman problem: A salesman visits n cities and minimizes thes distance traveled.
There are $n!$ (factorial n) routes.
For 50 cities, there are 3×10^{64} possible routes
It is no longer possible to calculate the total distance for all possible routes and be sure that the optimum route has been obtained.
A good algorithm for the solution of this type of problem will give what is called a *near optimal solution*.

Second difficulty with large problems
They can no longer be solved analytically
Near optimal solutions can only be obtained by the use of computer programs
operating with a set of algorithms which have been shown to give good solutions by extensive trials on other similar problems.
Heuristic programs.
SIS (Sequential Interactive Synthesis) package of programs from The Department of Electrical Engineering and Computer Science of the University of California.



ESPRESSO algorithm uses the cube notation.

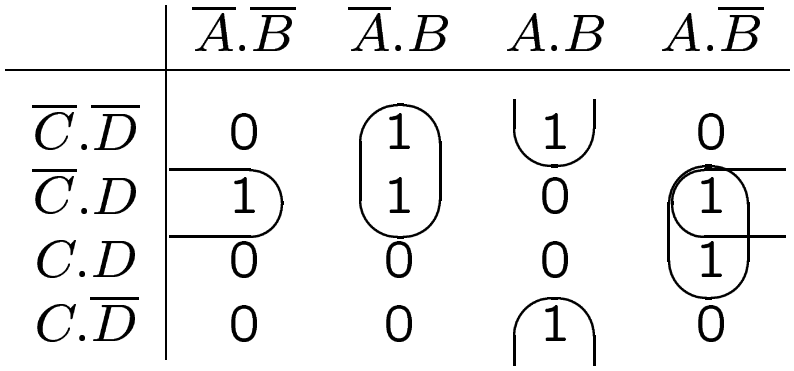
Boolean function occupies vertices of n dimensionsl cube.

Two sub cubes are considered to intersect if they have at least one common term or vertex.

A prime cube is a cube that cannot be expanded without including a vertex for which the Boolean expression is off—a vertex corresponding to a member of the off-set of minterms.

	$\overline{A}.\overline{B}$	$\overline{A}.B$	$A.B$	$A.\overline{B}$
$\overline{C}.\overline{D}$	0	1	1	0
$\overline{C}.D$	1	1	0	1
$C.D$	0	0	0	1
$C.\overline{D}$	0	0	1	0

1. Expand. A cube is expanded until no further expansion is possible without including a vertex of the off-set,
that is, the cube is expanded until it is prime. This operation involves complementing each input, in turn, to test if the new vertex is a member of the off-set or on-set of the Boolean expression.



2. Irredundant_Cover. Reduce the number of prime cubes to the minimum

The primes are classified into *relatively_essential* and *redundant* cubes

Partition into a totally redundant set and a partially redundant set

3. Reduce. The Expand and Irredundant_Cover give locally optimal solution
But not global optimum solution.

Remember that this is an NP-Complete problem where the number of solutions is so large that it is not possible to test all solutions for optimality. The Reduce procedure transforms a prime cover into a new cover by replacing each cube by a smaller cube contained within it.

These three procedures of Expand, Irredundant_Cover and Reduce are iterated with different starting points until there is no further improvement in the optimality of the reduction.

The ESPRESSO command is then completed by the preparation of a file which contains the optimum output configuration for the Boolean function.

	$\overline{A}.\overline{B}$	$\overline{A}.B$	$A.B$	$A.\overline{B}$
$\overline{C}.\overline{D}$	0	0	1	1
$\overline{C}.D$	0	1	0	0
$C.D$	0	1	0	0
$C.\overline{D}$	1	0	1	1

Input File name blexpt1

```
.i  4                    ;4 inputs
.o  1                    ;One output
1100 1                   ;Truth table
1000 1
0101 1
0111 1
0010 1
1110 1
1010 1
```

SIS system started
Read input file blexpt1
espresso program is run. output to blexpt1o in Berkeley Logic Interchange Format (blif) file

```
sis
read_pla blexpt1
espresso
write_blif blexpt1o
quit
```

Output available in file blexpt1o

```
.model blexpt1              ;Name of the input file
.inputs v0 v1 v2 v3         ;SIS  A, B, C, D
.outputs v4.0               ;Only output
.names v0 v1 v3 [1]         ;Product term, ABD
011 1                       ;ABD = 011 output 1
.names v0 v3 [2]            ;Product term, AD
10 1                        ;AD = 10 output 1
.names v1 v2 v3 [3]         ;Product term BCD
010 1                       ;BCD = 010 output 1
.names [1] [2] [3] [4]      ;Output product of 3 compl
000 1                       ;having value 000 = 1
.names [4] v4.0             ;Complemented give S of P
0 1                         ;by De Morgan's Theorem
.end
```

We can now interpret this file as follows:
$\overline{A}.B.D$ is represented by lines 4 and 5 100 1
$A.\overline{D}$ is represented by lines 6 and 7
$\overline{B}.C.\overline{D}$ is represented by lines 8 and 9
which gives in lines 10 to 13 the Boolean expression:

$$\begin{aligned} Q &= \overline{\overline{(\overline{A}.B.D)}\,\overline{(A.\overline{D})}\,\overline{(\overline{B}.C.\overline{D})}} \\ &= \overline{A}.B.D + A.\overline{D} + \overline{B}.C.\overline{D} \end{aligned}$$

```
.i 12
.o 8
----1------0 10000000
----0-----0- 10000000
0---------- 10000000
-----1-----0 01000000
-----0----0- 01000000
-0--------- 01000000
------1----0 00100000
------0---0- 00100000
--0-------- 00100000
-------1---0 00010000
-------0--0- 00010000
---0------- 00010000
0---1---0--- 00001000
0---0----0-- 00001000
```

```
.model alu1
.inputs v0 v1 v2 v3 v4 v5 v6 v7 v8 v9 v10 v11
.outputs v12.0 v12.1 v12.2 v12.3 v12.4 v12.5 v12.6
.names v3 v7 v8 [8]
010 1
.names v0 v4 v9 [9]
000 1
.names v1 v5 v9 [10]
000 1
.names v2 v6 v9 [11]
000 1
.names v0 v4 v8 [12]
010 1
.names v1 v5 v8 [13]
010 1
.names v2 v6 v8 [14]
010 1
.names v7 v10 [15]
```

```
.names [9] [12] [35]
00 1
.names [35] v12.4
0 1
.names [10] [13] [37]
00 1
.names [37] v12.5
0 1
.names [11] [14] [39]
00 1
.names [39] v12.6
0 1
.names [8] [41]
0 1
.names [41] v12.7
0 1
.end
```