IWLS 2018 programming contest

Mathias Soeken, *EPFL*   Alan Mishchenko, *UC Berkeley*

Boolean logic networks. We consider single output Boolean logic networks that realize Boolean functions $f(x_1, \dots, x_n)$. A k-feasible n-input Boolean logic network is a sequence of gates

$$x_i = g_i(x_{s(i,1)}, \dots, x_{s(i,k)}) \qquad \text{for } n < i \leq n + r,$$

where g_i is any k-input Boolean function with inputs from previous gates or primary inputs $x_{s(i,k)}$, with $1 \leq s(i,j) < i$ for all $1 \leq j \leq k$. The Boolean function realized by the network is obtained from the value to x_{n+r} for all assignments to the primary inputs $x_1, \dots, x_n$.

For convenience, we define $S_i = \{s(i,1), \dots, s(i,k)\}$ as the support variables of gate i.

A Boolean logic network is *normalized*, if all of the following conditions hold.

(i) $g_i(0, 0, \dots, 0) = 0$ for all $n < i \leq n + r$.

(ii) $s(i,1) \leq s(i,2) \leq \dots \leq s(i,k)$ for all $n < i \leq n + r$.

(iii) If $s(i+1,k) \neq i$, then $(s(i,1), s(i,2), \dots, s(i,k)) \preceq (s(i+1,1), s(i+1,2), \dots, s(i+1,k))$.

(iv) If $S_i = S_{i+1}$, then $g_i \prec g_{i+1}$

(v) If the primary inputs x_j and x_l are symmetric, then $\arg\min_i\{j \in S_i\} \leq \arg\min_i\{l \in S_i\}$.

Note that normalized Boolean logic networks cannot represent all functions f, but only those for which $f(0, 0, \dots, 0) = 0$.

Condition (iii) states, that if two successive gates are independent from each other (i.e., $s(i+1) \neq i$), then we require the two gates to be lexicographically ordered based on their support. For example, $(1,3,3) \prec (2,2,3)$, and $(1,2,2) \prec (1,2,3)$, but $(2,3,3) \not\prec (1,3,3)$.

Condition (iv) states, that if two successive gates have the same support, the gates must be ordered according to their functions. We do this by considering their truth tables as bitstrings of size 2^k. For example, 2-input AND ($1000_2 = 8_{10}$) is smaller than 2-input OR ($1110_2 = 14_{10}$).

Finally, condition (v) states that if two primary inputs x_j and x_l are symmetric (i.e., we can swap the two inputs without changing the function), then input x_j must be used by a gate before input x_l is used.

Task. Write a computer program, that for a given a Boolean function $f(x_1, \dots, x_n)$, a number k, and a number of gates r, finds as many k-feasible normalized Boolean logic networks as possible that realize f with r gates. The program should terminate within one hour and should write all found logic networks into a single file called `<func>`-k-r`.bln`. In that filename, `<func>` is the hexadecimal truth table representation of the function (from the given list of contest functions). Each network is described with a line for each gate formatted as follows:

chr(i) `= <binary truth table of` g_i`>` chr($s(i,1)$) … chr($s(i,k)$)

where chr(i) is the letter in the alphabet at position i, i.e., chr(1) = `a`, chr(2) = `b`, and so on. For convenience, we use lower case letters for primary inputs ($i \leq n$), and upper case letters for gates ($i > n$). As an example, the file `e8-2-4.bln` for logic networks of the majority function may contain the 4-gate solution

$$x_4 = x_1 \oplus x_2, \;\; x_5 = x_1 \vee x_2, \;\; x_6 = x_3 < x_4, \;\; x_7 = x_5 > x_6$$

formatted as

```
D = 0110 a b
E = 1110 a b
F = 0100 c D
G = 0010 E F
```

All gates should be in order, and multiple solutions should be seperated from each other by an empty line. Also note that $x_{s(i,1)}$ is the least significant input to function g_i. To illustrate that, the truth table for implication is `1101`.