INFORMATION FLOW AND WIDTH OF BRANCHING PROGRAMS

(Extended Abstract)

S.P. Jukna
Institute of Mathematics
Lithuanian Academy of Sciences
Vilnius, 232021, USSR

In this report some quantitative observations on the effect of information-flow restrictions to the width of branching programs are given.

A branching program over the set of Boolean variables $X = \{x_1, \dots, x_n\}$ is a labeled acyclic digraph G with the following properties:

(i) There is exactly one source.

(ii) Every vertex has outdegree at most 2.

(iii) Every edge is labelled by a contact x^a, where $x \in X$ and $a \in \{0,1\}$.

(iv) For every edge of outdegree 2, one of the leaving edges is labelled by a variable x and the other by its complement $\neg x$.

The branching program computes a Boolean function defined by the disjunction of all the monoms associated with the paths from the source to leaves . The length of a path is the number of distinct variables in it. The height of a vertex v in G is the maximal length of a path to v. For $k \geq 0$, let $G(k) = \{v \in G : \text{height}(v) = k\}$ and put $\text{Width}(G) = \max\{|G(k)| : 0 \leq k \leq n\}$. A path is a null path if it contains some pair of contrary contacts. Let Inf(G,v) denote the number of variables $x \in X$ such that for some $a \in \{0,1\}$ the following holds: there are two non-null paths P_1 and P_2 from the source of G to v and a path P_3 from v to a leaf of G such that $x^a \in P_1$, $\neg x^a \in P_3$ and paths P_2P_3 and $(P_1 - \{x^a\})P_3$ are both non-null. Informally, Inf(G,v) expresses the amount of information which is necessary to determine the value of the function when computation is started in v. Let G_v denote the subprogram of G generated by all the paths from v to the leaves of G.(Thus v is a source of G_v). For $0 \leq r,k \leq n$, let

$$\text{Inf}(G,r,k) = \min \max \text{Inf}(G_v,u),$$

where "min" is over all $v \in G(k)$ and "max" is over all $u \in G_v(r)$. Put also Inf(G,r) = Inf(G,r,0). Note that for all $i,j \geq 0$,

$$0 \leq \text{Inf}(G,r-i,k+j) \leq \text{Inf}(G,r,k) \leq \text{Inf}(G,r) \leq \text{Inf}(G,n) \leq n.$$

In what follows the term "almost all" refers to a (1 - o(1)) fraction.

The method of synthesis by O.B. Lupanov [8] implies the following.

Fact: For almost any n-ary Boolean function, the set of its minimal branching programs contains a program G with $\mathrm{Inf}(G,r) = 0$ for any $r \le n(1 - o(n^{-1/2}))$.

Thus almost all Boolean functions have minimal programs with a small information flow. However, for some concrete functions the information flaw is rather complicated in all their minimal programs.

An assignment is a function $w\colon X \longrightarrow X \cup \{0,1\}$ such that for every $x \in X$, $w(x) \in \{0,1,x\}$; $\mathrm{dom}(w) = w^{-1}(0) \cup w^{-1}(1)$ is a domain of w ; $|\mathrm{dom}(w)|$ is a rank of w. For a Boolean function $f(X)$, set $f^w = f(w(x_1),\dots,w(x_n))$. Let $Q(f,p)$ denote the minimal number q such that there exists a set W of q assignments of rank p, with $\bigcap\{\mathrm{dom}(w) : w \in W\} \ne \emptyset$,possessing the representation $f = \bigvee \{ f^w : w \in W\}$. A Boolean function $f(X)$ is (weakly) m-mixed ([6,7]) if for any $Y \subseteq X$, with $|Y| \le m$, and any two assignments w,z over the domain Y, it holds that (either $f^w = f^z = 0$ or) $f^w \ne f^z$. The class of mixed functions is sufficiently rich: for any $\varepsilon > 0$ and $m \le n - (1+\varepsilon)\log_2 n$, almost all n-ary Boolean functions are m-mixed.

Theorem: For any branching program G computing a weakly m-mixed function f and any integers $r,k \ge 0$, with $k + 2r \le m$, it holds that

$$\log_2 \mathrm{Width}(G) + 1.6\ \mathrm{Inf}(G,r,k) \ge b,$$

where $b = r$ if f is m-mixed and $b = \log_2 Q(f,r+k)$ otherwise.

In [6,7] a uniform argument is given to generate concrete $O(\sqrt{n})$-mixed n-ary Boolean functions from NP (and even from P). In particular, the following two n-ary Boolean functions f_n and g_n are $\sqrt{n}/2$-mixed and belong to P ([4,11]): for a (0,1)-matrix X of order $\sqrt{n}$, let $f_n(X) = 1$ iff $\mathrm{Per}(X) > 0$, and $g_n(X) = \mathrm{Per}(X) \pmod 2$, where $\mathrm{Per}(X)$ is the permanent of X.So, $\log_2 \mathrm{Width}(G) \ge \sqrt{n}/4 - 1.6\ \mathrm{Inf}(G,\sqrt{n}/4)$ for any program G computing f_n or g_n. Next, the "exactly-half-clique" Boolean function h_n (see, e.g.[6]) is computable by a polynomial size branching program and $Q(h_n,\sqrt{n}/4) \ge 2^{c\sqrt{n}}$, $c > 0$. Thus, $\mathrm{Inf}(G,\sqrt{n}/4) \ge n^{1/2-\varepsilon}$ for any minimal program G computing h_n.

On the other hand, width restrictions do not increase the size of programs drastically. A branching program is called stratified if all the edges leaving the vertexes of any given height are labelled by contacts of the same variable. From [2,10] it follows that for any sequence $\{F_n\}$ of Boolean formulae over the basis $\{\&,\vee,\neg\}$ there is a sequence $\{G_n\}$ of stratified branching programs of width ≤ 5 such that:

$$\mathrm{size}(G_n) \le \mathrm{size}(F_n)^{O(1)}.$$

Although the transition from the (unrestricted) formulae to constant width branching programs (and vice versa) does not increse the size drastically, the information flow may become more complicated.

Corollary: Let G be a stratified branching program of width $\leq$ d. If G computes an m-mixed Boolean function then for any $r,k \geq 0$, with $k + r \leq m$, it holds that $r - \log_2 d \leq Inf(G,r,k) \leq r$.

Thus, to prove non-trivial lower bounds for the complexity, a new insight into the information flow is desirable. One of the possible ways is to use Ramsey-like arguments. Some work in this direction was done in [1,3,9], where an $\Omega(n \log_2 n)$ lower bounds for symmetric n-variable Boolean functions were proved. Another way (proposed in [5-7]) is to look for appropriate measures of "distance" for subfunctions. We conjecture that the functions with "highly distant" subfunctions have minimal networks with a small information flow.

References:

1. M. Ajtai, L. Babai, P. Hajnal, J. Komlós, P. Pudlák, V. Rödl, E. Szemerédi and G. Turán, Two lower bounds for branching programs, Proc. 18-th ACM STOC (1986) 30-38.
2. D.A. Barrington, Bounded-width polynomial size branching programs recognize exactly those languages in NC^1, Proc. 18-th ACM STOC (1986) 1-5
3. A.K. Chandra, M.L. Furst and R.J. Lipton, Multiparty protocols, Proc. 15-th ACM STOC (1983) 94-99.
4. J.E. Hapcroft and R.M. Karp, An $n^{5/2}$ algorithm for maximum maching in bipartite graphs, SIAM J. Comput. 2 (1973) 225-231.
5. S.P. Jukna, An entropic method of obtaining lower bounds for the complexity of Boolean functions, to appear in Dokl. Akad. Nauk SSSR (1987).
6. ------, Lower bounds on the complexity of local circuits, Proc. 12-th Int. Symp. MFCS, LNCS 233 (1986) 440-448.
7. ------, Entropy of Boolean networks and lower bounds on their complexity, to appear in Theoretical Computer Science.
8. O.B. Lupanov, On the synthesis of switching networks, Dokl. Akad. Nauk SSSR 119, n.1 (1958) 23-26.
9. P. Pudlák, A lower bound on the complexity of branching programs, Proc. 11-th Int. Symp. MFCS, LNCS 176 (1984) 480-489.
10. P.M. Spira, On time-hardware tradeoffs for Boolean functions, Proc. 4-th Hawaii Int. Symp. on System Sciences (1971) 525-527.
11. L.G. Valiant, The complexity of computing the permanent, Theoretical Computer Science 21 (1982) 181- 201.