数理解析研究所講究録
716巻 1990年 117-123

A METHOD TO COMPUTE LOWER BOUNDS ON CIRCUIT-SIZE COMPLEXITY

Hajime MACHIDA
(町 田 元)

Hitotsubashi University

INTRODUCTION

In Boolean complexity theory, circuit-size complexity is one of the main targets of research ([1], [3]). **Circuit-size complexity** (which is also called **combinational complexity** or **network complexity**) of a Boolean function f over a base set B is defined to be the least number of gates contained in a Boolean circuit which is composed of gates in B and computes f. For most of the functions it is extremely difficult to get good estimate, not to mention the exact value, on the circuit-size complexity. Especially, good lower bounds are hard to obtain. It is, therefore, quite welcome to develop any new methods to derive good bounds on circuit-size complexity.

The purpose of this note is to exhibit a technique to derive a lower bound on the circuit-size complexity of a function f by counting, individually, for **each** gate in B the number of occurrences only of that gate appearing in any circuit computing f.

Note that this method is not new: It already appeared in Tiekenheinrich [2]. However, it is rarely used in the literature and so is worth mentioning here.

PRINCIPLE

Let $G = \{g_1, g_2, \ldots, g_r\}$ be a base set, i.e., a set consisting of gates (Boolean functions) which can be used in constructing a circuit. Let f be a Boolean function which is realizable by a circuit over G. Denote by $C_G(f)$ the circuit-size complexity of f over G, and by $C_{G,gi}(f)$, $1 \leq i \leq r$, the least number of occurrances of the gate g_i contained in any circuit computing f over G.

It is clear that

$$C_G(f) \geq \sum_{i=1}^{r} C_{G,gi}(f).$$

Consequently, if a lower bound m_i on $C_{G,gi}(f)$ can be derived for each $1 \leq i \leq r$, we have a lower bound on $C_G(f)$ as well. That is, if

$$C_{G,gi}(f) \geq m_i, \qquad 1 \leq i \leq r,$$

then

$$C_G(f) \geq \sum_{i=1}^{r} m_i \ .$$

RESULT

The above principle can be applied to the following function f to get (8/3)n lower bound on the circuit-size complexity over the monotone base {AND, OR}.

Let G = {AND(= $\wedge$),OR(= $\vee$)} and $f^{(n)}$ be an n-variable function, n>0, defined as follows.

$$f_1^{(n)}(x_1, \ldots, x_n) = \begin{cases} 1 & x_1 + \ldots + x_n \geq (1/3)n, \\ 0 & x_1 + \ldots + x_n < (1/3)n, \end{cases}$$

$$f_2^{(n)}(x_1, \ldots, x_n) = \begin{cases} 1 & x_1 + \ldots + x_n > (2/3)n, \\ 0 & x_1 + \ldots + x_n \leq (2/3)n, \end{cases}$$

and

$$f^{(n)}(x_1, \ldots, x_n, y) = (f_1^{(n)}(x_1, \ldots, x_n) \wedge y) \vee f_2^{(n)}(x_1, \ldots, x_n).$$

For the functions $f_1^{(n)}$ and $f_2^{(n)}$, we can prove the following.

Lemma 1 $C_{G,OR}(f_1^{(n)}) \geqq (4/3)n - 2.$

Lemma 2 $C_{G,AND}(f_2^{(n)}) \geqq (4/3)n - 2.$

As

$$f^{(n)}(x_1, \ldots, x_n, 1) = f_1^{(n)}(x_1, \ldots, x_n)$$

and

$$f^{(n)}(x_1, \ldots, x_n, 0) = f_2^{(n)}(x_1, \ldots, x_n),$$

we have

$$C_{G,OR}(f^{(n)}) \quad (\geqq C_{G,OR}(f_1^{(n)}) \ ) \quad \geqq (4/3)n - 2$$

and

$$C_{G,AND}(f^{(n)}) \ (\geqq C_{G,AND}(f_2^{(n)}) \ ) \geqq (4/3)n - 2.$$

The above principle can now be applied to get

Proposition $C_G(f^{(n)}) \geqq (8/3)n - 4.$

PROOF OF LEMMAS

Proof of Lemma1: Let γ be an optimal circuit computing $f_1^{(n)}$ over the base set G, where $n \geqq 2$.

Fact 1. For every input node x_i and every path p from x_i to the output node in γ, there exists at least one OR-gate in p.

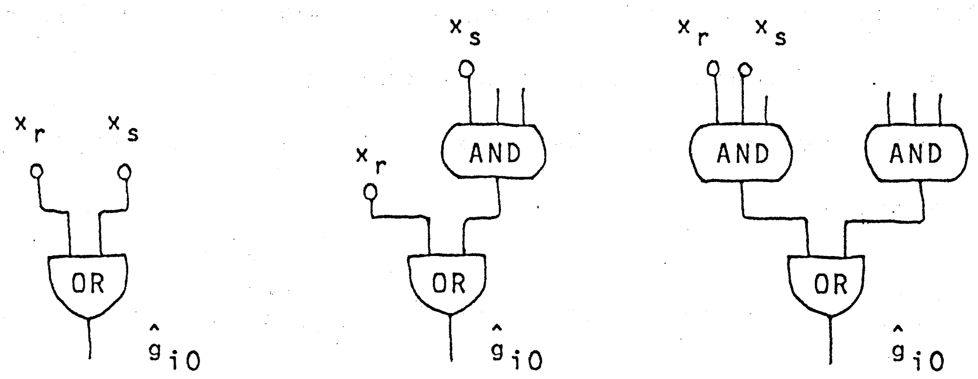
Proof. Suppose, on the contrary, that there is a path from x_i to the output node which contains only AND-gates. Then by setting $x_i = 0$ the output always takes the value 0, contradicting the definition of $f_1^{(n)}$.

Fact 2. For some input node x_i in γ, there exist at least two OR-gates g_1 and g_2 as well as two paths p_1 and p_2 each connecting x_i and the output node such that

1) g_j lies on p_j $(j = 1, 2)$ and

2) there is no other OR-gate on p_j between x_i and g_j $(j = 1, 2)$.

Proof. Assume that the statement is false, and consider, for each i, all the paths from the input node x_i to the output node. Then all of them must contain the same OR-gate, say $\hat{g}_i$, as the closest OR-gate to the input node x_i. Moreover, for some i_0 the



(a) (b) (c)

Fig. 1

gate $\hat{g}_{i0}$ has no OR-gate as its ancestors, that is, there is no OR-gate between $\hat{g}_{i0}$ and input nodes. The situation is shown in Fig. 1. In cases (a) and (b) the circuit is made not to depend on x_s by assigning $x_r = 1$, and in case (c) it is made not to depend on x_s by assigning $x_r = 0$. In either case this contradicts the definition of the function.

Now, Lemma 1 can be proved by mathematical induction. As the basis, note that

$$C_{G,OR}(f_1^{(1)}) = 0,$$

$$C_{G,OR}(f_1^{(2)}) = 1,$$

$$C_{G,OR}(f_1^{(3)}) = 2.$$

Suppose $C_{G,OR}(f_1^{(n)}) \geqq (4/3)n - 2$ holds for $n \geqq 3$ and consider a circuit $\gamma^{(n+3)}$ computing $f_1^{(n+3)}$. Choose an input node x_i in $\gamma^{(n+3)}$ as in Fact 2, and substitute the value 0 to x_i. By this procedure at least two OR-gates can be eliminated from $\gamma^{(n+3)}$ without affecting the result of the circuit. Repeat the same procedure to the resulting circuit, eliminating at least two OR-gates again. Finally, substitute the value 1 to any one of the rermaining input nodes. Then what we have is a circuit computing $f_1^{(n)}$, because

$$f_1^{(n+3)}(x_1, \ldots, x_n, 0, 0, 1) = f_1^{(n)}(x_1, \ldots, x_n)$$

holds. (Here, w.l.o.g., the inputs to which the value 0 or 1 is substituted are assumed to be x_{n+1}, x_{n+2} and x_{n+3}.) By hypothesis, we have

$$C_{G,OR}(f_1^{(n+3)}) \geq ((4/3)n - 2) + 2 + 2$$
$$= (4/3)(n+3) - 2.$$

This completes the proof of Lemma 1. QED

Proof of Lemma 2: In a circuit γ over the base G = {AND, OR} computing $f_2^{(n)}(x_1, \ldots, x_n)$, replace each AND-gate by an OR-gate and each OR-gate by an AND-gate. Then, by de Morgan's law, the resulting circuit γ' computes $\overline{f_2^{(n)}(\bar{x}_1, \ldots, \bar{x}_n)}$. Now it is easy to see that

$$\overline{f_2^{(n)}(\bar{x}_1, \ldots, \bar{x}_n)} = f_1^{(n)}(x_1, \ldots, x_n),$$

and Lemma 2 follows from Lemma 1. QED

REFERENCES

[1] Savage, J. E., The Complexity of Computing, John Wiley & Sons, 1976.

[2] Tiekenheinrich, J., A 4n-lower bound on the monotone network complexity of a one-output Boolean function, Information Processing Letters 18 (1984) 201-202.

[3] Wegener, I., The Complexity of Boolean Functions, Wiley-Teubner, 1987.