Some machines defined by directed graphs

de Bruijn, N.G.

Published: 01/01/1979
SOME MACHINES DEFINED BY DIRECTED GRAPHS

by

N.G. de Bruijn.

University of Technology
Department of Mathematics
P.O.Box 513,
Eindhoven
The Netherlands.
Some machines defined by directed graphs
by N.G. de Bruijn.

1. Introduction. In a previous note ([2]) we showed that a certain sorting machine devised by P.Armstrong and M.Rem functions properly under all circumstances. It is a bit-processing machine, but it can be considered as a machine that operates on binary sequences (binary numbers with a fixed word length). This transition seems to be harder if we replace the very special graph of their machine by a more general one, as will be considered in this note. A bit-processing machine $M_1$ will be defined by means of a directed graph, and we shall see how it can be replaced by a machine $M_2$ that processes binary sequences. Finally we show (in section 5) how relatively simple diagrams (see figs. 3 and 6) can replace the confusing picture of the zeros and ones wriggling through a graph.

2. The machine $M_1$. We consider a quadruple $(G,E,\text{tail head})$, where $G$ and $E$ are sets, "tail" and "head" are mappings of $E$ into $G$. The elements of $G$ are called vertices, those of $E$ are called (oriented) edges. We say that the edge $e$ runs from tail($e$) to head($e$), and in the figures this will be indicated by an arrow.

If $P \in G$, the number of $e \in E$ with tail($e$) = $P$ is called out($P$) (the "out-degree" of $P$); the number of $e \in E$ with head($e$) = $P$ is called in($P$) (the indegree of $P$). We shall require that

$$\forall P \in G \quad \text{out}(P) = \text{in}(P) \quad (2.1)$$

(a directed graph with this property is usually called an Euler graph). We shall also require that all degrees are either 1 or 2; generalization to higher degrees is possible but not very attractive. If

$$\text{out}(P) = \text{in}(P) = 1$$

then $P$ will be called an **ordinary point**, if

$$\text{out}(P) = \text{in}(P) = 2$$

then $P$ will be called a **switch**.
At a switch $P$ we have $e_1, e_2$ with $\text{head}(e_1) = \text{head}(e_2) = P$; these are called the inputs of $P$. We label these two edges: one of them is called the high input, the other one the low input. Similarly we label the edges $e_3, e_4$ whose tail is $P$: one is called the high output, the other one the low output.

The situations at a switch will be described by means of a set with three elements: "through", "back", and "neutral".

A state of the machine is obtained by attaching a bit (taken from the set $\{0,1\}$) to each edge, and a switch setting (taken from the set \{"back", "through", "neutral"\}) to each switch.

In order to describe the actions of the machine, we refer to the set $T = \{0,1,2,\ldots\}$ as the set of time moments. To each $P \in G$ we attach a subset $T_P$ (the elements of $T_P$ are called the neutralization moments for $P$).

We shall describe how from a state of the machine (at time $t$) we get the next state (at time $t+1$). This implies that if the state is given at $t=0$, it is completely determined for all $t \in T$. Let us denote the bit on edge $e$ at time $t$ by $b(e,t)$.

At an ordinary point $P$ the rule is simple: the bit on the incoming edge is transferred to the outgoing edge. That is, if $P = \text{head}(e_1) = \text{tail}(e_2)$, then $b(e_2,t+1) = b(e_1,t)$.

If $P$ is a switch the rule is more complex. First assume $t \notin T_P$.

(i) If at time $t$ the switch setting is "through", then the bit on the high input is transferred to the high output, the bit on the low input is transferred to the low output. The switch setting remains "through".

(ii) If at time $t$ the switch setting is "back", then the bit on the high input is transferred to the low output, the bit on the low input to the high output. The switch setting remains "back".

(iii) If at time $t$ the switch setting is "neutral", we apply the following list:

<table>
<thead>
<tr>
<th>time $t$</th>
<th>time $t+1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>high input</td>
<td>low input</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
If, finally, \( t \in T_P \), the action is explained in two steps: first set the switch to neutral, then apply rule (iii).

We shall refer to the machine described here as \( M_1 \).

3. Processing binary sequences. We shall formulate a synchronization condition that guarantees that certain trains of bits running through the machine stay together as trains. In section 1 the sets \( T_P \) played a role only if \( P \) is a switch, but here \( T_P \)'s will be considered for ordinary points too. The synchronization condition is

\[
\forall e \in E \, \forall t \in T \ (t \in T_{\text{tail}}(e) \Rightarrow t+1 \in T_{\text{head}}(e)).
\]

From now on we assume the synchronization condition to be satisfied.

Let us call a pair \((e, t)\) (with \( e \in E, t \in T \)) a leader if \( t \in T_{\text{head}}(e) \). The leaders move stepwise if \( t \) proceeds: if \( e \) and \( e' \) are consecutive edges (in the sense that \( \text{head}(e) = \text{tail}(e') \)) then \((e, t)\) is a leader if and only if \((e', t+1)\) is a leader. In particular it follows that if \( e_1 \) and \( e_2 \) are the inputs of a switch, and if \((e_1, t)\) is a leader, then \((e_2, t)\) is a leader too.

If \((e, t)\) is a leader we consider the sequence

\[
\beta(e, t) = (b(e, t), b(e, t+1), b(e, t+2), \ldots, b(e, t+k))
\]

where \( k \) is such that \((e, t+k+1)\) is the first leader in the sequence \((e, t+1), (e, t+2), \ldots \). If there is no such \( k \), we take for \( \beta(e, t) \) the infinite sequence \((b(e, t), b(e, t+1), \ldots)\).

We can associate to \( \beta(e, t) \) the real number

\[
2^{-1}b(e, t) + 2^{-2}b(e, t+1) + \ldots + 2^{-k-1}b(e, t+k).
\]

We write \( \beta(e_1, t_1) < \beta(e_2, t_2) \) if \( \beta(e_1, t_1) \) is lower in the lexicographic order than \( \beta(e_2, t_2) \). In particular this happens if the inequality holds for the associated reals, but we also have inequalities like

\[
(0,1,1,1,1,1,\ldots) < (1,0,0,0,\ldots).
\]

If \( e \) and \( e' \) are consecutive edges, and if \( \text{head}(e) \) is an ordinary point, it is easy to see that

\[
\beta(e, t) = \beta(e', t+1) \quad \text{(3.1)}
\]

(in particular the sequences have the same length).

Let us next consider a switch with inputs \( e_1, e_2 \) and outputs \( e_3 \) (high), \( e_4 \) (low). From the rules of section 2 it follows that (if \((e_1, t)\) is a leader)
\[ \beta(e_3, t+1) = \max(\beta(e_1, t), \beta(e_2, t)), \]
\[ \beta(e_4, t+1) = \min(\beta(e_1, t), \beta(e_2, t)). \]

We also note that \( \beta(e_1, t) \) and \( \beta(e_2, t) \) have the same length. It should be remarked that sequences can have different length and yet have the same real associated to them, like \((1,0,0,0)\) and \((1,0)\).

4. A simpler machine \( M_2 \). We can describe the behaviour of the machine \( M_1 \) if we just look at the way the \( \beta \)'s move. We attach a sequence \( \beta(e, t) \) to \((e, t)\) if \((e, t)\) is a leader, but to non-leaders we do not attach anything. At ordinary points the \( \beta \)'s are just passed along (see (3.1)), at switches there are always two \( \beta \)'s arriving at the same time. If they are different, the larger takes the high output. If they are equal they are just passed on to the outputs. Let us call this machine \( M_2 \).

We remark that \( M_2 \) does not describe what happens in \( M_1 \) at an edge \( e \) before there has been any leader at that edge, i.e. at times \( t \) with \( \{1, 2, \ldots, t\} \cap T_{\text{head}(e)} = \emptyset \).

5. Swapping schemes for \( M_2 \)'s with constant word length. We shall specialize \( M_1 \) (and therefore \( M_2 \)) by requiring that there is a fixed positive integer \( w \) such that for every \( P \in G \) the set \( T_P \) has the form

\[ \{r_P, r_P+w, r_P+2w, \ldots\} \]

with some \( r_P \) satisfying \( 0 \leq r_P < w \). This guarantees that all sequences \( \beta(e, t) \) have length \( w \).

Let us define the notion "\( r \)-path". If \( r \) is an integer with \( 0 \leq r < w \), then an \( r \)-path is a sequence \( e_1, \ldots, e_w \) of consecutive edges (\( \text{head}(e_1) = \text{tail}(e_2), \ldots \)) starting at a point \( P \) (\( \text{tail}(e_1) = P \)) with \( r_P = r \). It follows that the \( r \)-path also ends at such a point.

It is easily seen that the set \( E \) if all edges can be considered as the disjoint union of a set of \( r \)-paths (all with the same \( r \)). An example is given in fig. 1, where we have \( w=5, r=2 \). In fig. 1 we have indicated the \( r \)'s; the graph is drawn a second time in fig. 2 in order to display a partition into \( r \)-paths, with names \( \ldots, A_1, A_0, A_1, \ldots \) assigned to them. In fig. 1 we have indicated the high outputs by heavy pieces of line.
If we consider the actions of the machine $M_2$, with $\beta$'s moving through the graph, we observe that at each moment each $r$-path contains exactly one $\beta$. Transitions of $\beta$'s from one path to another take place at two occasions:

(i) at times $t \equiv r \pmod{w}$ where the $\beta$ just steps into the next $r$-path,

(ii) at moments when two $\beta$'s get to a switch. Here the larger of the two goes to the $r$-path that contains the high output, the smaller goes to the other one. (For simplicity of the description we have chosen a first example where the cases (i) and (ii) never coincide, i.e. where no switch has its $r_p$ equal to $r$; a case where this does happen can be seen in figs. 4, 5, 6).

We can now represent the machine by an entirely different scheme, as displayed in fig. 3. Time proceeds from top to bottom, and we have represented a full period of length $w$. The $\beta$'s move along the (curved) vertical lines; at each moment all $\beta$'s lie on some horizontal cross section. If two $\beta$'s get to points connected by a horizontal arrow, the larger of the two goes to the head of that arrow, the smaller to the tail. In the curves, the $\beta$'s switch to other verticals; note that the $A$'s correspond to pure verticals, not to curved ones. For the $\beta$'s we have taken, instead of strings of zeros and ones, just integers $(4, 5, 8, 3, 2, 7, 1, 9, 8)$, written at various times to the right of the line on which they are moving.

This description of the actions of the machines give a much better survey than the original description of $M_1$ in section 1.

In figs. 4, 5, 6 we have done the same thing as in figs. 1, 2, 3 but now for a typical case of the Armstrong-Rem machine (see [1], [2]). The sets of lines in fig. 6 on the left and on the right both stretch to infinity in both directions. Those pieces of the figure have the form of the graph of $y = \cot x$. In this example we took $w = 5$, and we have chosen the four "rings" of the machine as $r$-paths (we took $r=0$). This leads to the very simple diagram of fig. 6. It is this kind of diagrams that were shown to in [2] to represent sorting machines.

References.