A simple analysis of the n^{th} order polyphase sort

I. R. MacCallum

Computing Centre, University of Essex, Wivenhoe Park, Colchester CO4 3SQ

Several descriptions of the polyphase sort have been published but with one exception the literature appears to be silent regarding its analysis. This note attempts to restrict the discussion to a simple notation and therefore hopefully find a wider readership. It concludes with a comparison between the polyphase and balanced merge sorts.

(Received July 1972)

Whereas the polyphase sort has been well-described in the literature (Manker, 1963; Malcolm, 1963; Gilstad, 1963; Flores, 1969; Martin, 1971; etc.) there appears to be little analysis of the method. Flores (1969) sets out in 32 pages a description and analysis of the polyphase sort which might deter a fair number of those most likely to use it. This note attempts to contain the discussion within fairly elementary mathematics and without too much reference to external sources. It might, therefore, be found useful for any course on Data Processing which emphasises the principles behind the practice.

Let the sort use $n + 1$ tape drives named $t_1, t_2, \ldots, t_n, t_{n+1}$. **Table 1** shows the number of unit strings on each tape following the final passes of the sort. It is later found to be convenient to extend the table by a further $n - 1$ imaginary passes according to the rules of the polyphase sort.

In each row of the table the zero corresponding to the tape which has to be rewound is in bold type. This zero is referred to as the 'pivotal' zero, to distinguish it from the other $n - 1$ zeros in the first n lines of the table.

Let $u_{i,j}$ be the jth element to the right of the 'pivotal' zero in the ith row, which is considered circular, e.g. $u_{i,n+1} = 0$ for all i.

The polyphase sort may now be defined by the recurrence relations

$$u_{n,1} = 1 \quad (1)$$

$$u_{n,j} = 0 \quad \text{for } j = 2, 3, \ldots, n + 1 \quad (2)$$

$$u_{i,j} = u_{i-1,1} + u_{i-1,j+1} \quad \text{for } i > 1 \text{ and } j = 1, 2, \ldots, n - 1 \quad (3)$$

$$u_{i,n} = u_{i-1,1} \quad \text{for } i > 1 \quad (4)$$

$$u_{i,n+1} = 0 \quad \text{for } i \geqslant 1. \quad (5)$$

The principal interest is in the row-sums $g_i = \sum_j u_{i,j}$ for it is these that represent the total number of strings on hand at each pass.

The first step is to show that the sequence g_i satisfies the recurrence relation

$$g_i = g_{i-1} + g_{i-2} + \ldots + g_{i-n} \quad \text{for } i > n$$

and

$$g_1 = g_2 = \ldots = g_n = 1 .$$

Writing k_i for $u_{i,1}$, it follows from (3), (4), and (5) that

$$g_{i+1} = g_i + (n - 1) k_i . \quad (6)$$

It therefore remains to be shown that k_i satisfies

$$k_i = k_{i-1} + k_{i-2} + \ldots + k_{i-n} \quad \text{for } i > n .$$

From the recurrence relation (3),

$$\left.\begin{aligned} u_{i,1} &= u_{i-1,1} + u_{i-1,2} \\ u_{i-1,2} &= u_{i-2,1} + u_{i-2,3} \\ u_{i-2,3} &= u_{i-3,1} + u_{i-3,4} \\ &\vdots \\ u_{i-n+2,n-1} &= u_{i-n+1,1} + u_{i-n+1,n} \end{aligned}\right\} \text{for } i > n \quad (7)$$

and from recurrence relation (4) the last term in the last equation above is given by

$$u_{i-n+1,n} = u_{i-n,1} . \quad (8)$$

Relations (7) and (8) now give

$$u_{i,1} = u_{i-1,1} + u_{i-2,1} + u_{i-3,1} + \ldots + u_{i-n+1,1} + u_{i-n,1}$$

which is

$$k_i = k_{i-1} + k_{i-2} + \ldots + k_{i-n} \quad \text{for } i > n . \quad (9)$$

The proof of

$$g_i = g_{i-1} + g_{i-2} + \ldots + g_{i-n} \quad \text{for } i > n$$

follows by induction since, from (6)

$$\begin{aligned} g_{i+1} &= g_i + (n - 1) k_i \\ &= g_{i-1} + g_{i-2} + \ldots + g_{i-n} \\ &\quad + (n - 1)(k_{i-1} + k_{i-2} + \ldots + k_{i-n}) \\ &= g_{i-1} + (n - 1) k_{i-1} \\ &\quad + g_{i-2} + (n - 1) k_{i-2} \\ &\quad + \ldots \\ &\quad + g_{i-n} + (n - 1) k_{i-n} \\ &= g_i + g_{i-1} + \ldots + g_{i-n+1} . \end{aligned}$$

From Table 1 it is clear that

$$g_1 = g_2 = \ldots = g_n = 1$$

and that $g_{n+1} = n$. Thus the recurrence relation is satisfied for its initial set of subscripts, namely

$$g_{n+1} = g_1 + g_2 + \ldots + g_n .$$

The general solution of the recurrence relation

$$g_i = g_{i-1} + g_{i-2} + \ldots + g_{i-n}$$

Table 1 Number of strings per tape following final passes of an *n*-tape polyphase sort

t_1	t_2	t_3	t_4	t_5	...	t_n	t_{n+1}	$\sum t_i$	PASS
1	**0**	0	0	0	...	0	0	g_1	imaginary passes.
1	0	**0**	0	0	...	0	0	g_2	
1	0	0	**0**	0	...	0	0	g_3	
1	0	0	0	**0**	...	0	0	g_4	
					⋮				
1	0	0	0	0	...	**0**	0	g_{n-1}	
1	0	0	0	0	...	0	**0**	g_n	last
0	1	1	1	1	...	1	1	g_{n+1}	last but 1
1	**0**	2	2	2	...	2	2	g_{n+2}	last but 2
3	2	**0**	4	4	...	4	4	g_{n+3}	last but 3
7	6	4	**0**	8	...	8	8	g_{n+4}	last but 4
15	14	12	8	**0**	...	16	16	g_{n+5}	last but 5
					⋮				

is

$$g_i = \sum_{r=1}^{n} A_r \alpha_r^i \qquad (10)$$

where $\alpha_1, \alpha_2, \ldots, \alpha_n$ are the roots of

$$p(x) = x^n - x^{n-1} - x^{n-2} - \ldots - x - 1 = 0$$

and $A_1, A_2, \ldots, A_n$ are arbitrary constants determined by the values of $g_1, g_2, \ldots, g_n$.

The polynomial $p(x)$ has one real root in the interval (1, 2) and $n - 1$ roots of modulus less than one. The proof of this important statement is given in Appendix 1. This is crucial to the analysis since it allows the solution (10) to be approximated to by $A\alpha^i$ where α is the dominant root, i.e. that in the interval (1, 2).

We now show, by expanding $p(x)$ in a Taylor series around the point 2, that α, the dominant root of $p(x)$ is given by

$$\alpha \doteq 2 - \frac{1}{2^n - 1} .$$

$$p(x) = x^n - x^{n-1} - x^{n-2} - \ldots - x - 1$$
$$= \frac{x^{n+1} - 2x^n + 1}{x - 1} \qquad \text{for } x \neq 1 .$$

Expanding $p(x)$ in a Taylor series around the point $x = 2$,

$$p(x) = p(2) + (x - 2)\, p'(2) + O(x - 2)^2 .$$

The root α is therefore given by

$$0 = p(2) + (\alpha - 2)\, p'(2) + O(\alpha - 2)^2 .$$

Hence

$$\alpha = 2 - \frac{p(2)}{p'(2)} + O(\alpha - 2)^2$$
$$= 2 - \frac{2^{n+1} - 2 \times 2^n + 1}{2^n - 1} + O(\alpha - 2)^2$$
$$= 2 - \frac{1}{2^n - 1} + O(2^{-2n}) .$$

Thus $\alpha \doteq 2 - \frac{1}{2^n - 1}$. Values of α, and $2 - \frac{1}{2^n - 1}$ are tabulated for $n = 2, 3, \ldots, 12$ in **Table 2.**

Hence

$$g_i \doteq A\alpha^i \text{ for some constant } A ,$$

and the number of passes required to sort S strings is approximately $[\log_\alpha S]$.

To compare the polyphase sort with, say, the balanced merge sort, it is necessary to know what fraction of the file is processed at each stage of the sort.

Using the notation of (6), the number of strings produced by the pass corresponding to a progression from row $i + 1$ to row i is k_i. To calculate how many *records* are processed we need to know the length of each string in terms of the lengths of the strings at the start of the sort which we define as *unit strings.* If, to Table 1, we add the lengths of each string in terms of unit strings (shown in parenthesis in **Table 3** where for simplicity n is taken to be 4), it is readily seen that the string length satisfies the same recurrence relation as does the total number of strings on hand at any stage. If the sort requires s passes then the string length corresponding to row i in the table is g_{s-i+n}.

The number of records processed at each pass is therefore $k_i . g_{s-i+n}$. But k_i has already been shown to have satisfied the recurrence relation (9).

Thus
$$k_i . g_{s-i+n} \doteq B . \alpha^i . A\alpha^{s-i+n}$$
$$= B . A . \alpha^{s+n}$$

which is a constant. Thus the number of *unit strings* processed at each pass is approximately the same.

In particular, on the first pass, the number is

$$n . k_{s-1} = n . \frac{g_s - g_{s-1}}{(n - 1)}$$

and the fraction processed, ϕ, is given by

$$\phi = \frac{n}{n - 1}\left(\frac{g_s - g_{s-1}}{g_s}\right)$$
$$= \frac{n(\alpha - 1)}{\alpha(n - 1)} .$$

(For the final pass, of course, $\phi = 1$). Thus the amount of processing is equivalent to processing the entire file about

$$\frac{n(\alpha - 1)}{\alpha(n - 1)} \log_\alpha S \text{ times} .$$

Finally we ask whether, given n (even) tapes, the polyphase sort is faster or slower than an $n/2$-way balanced merge sort.

Consider first large n. Then $\alpha \to 2$ and the equivalent power of the polyphase sort is approximately

$$\frac{n - 1}{2(n - 2)} \log_2 S .$$

For the balanced sort, the number of passes is approximately

$$\log_{n/2} S = \frac{\log_2 S}{\log_2\left(\frac{n}{2}\right)} .$$

Now the balanced sort is preferred to the polyphase sort if

Table 2 Comparison between dominant root of $p(x)$ and approximation given by Taylor expansion

n	α	$2 - \frac{1}{2^n - 1}$
2	1·61803399	1·66666667
3	1·83928676	1·85714286
4	1·92756198	1·93333333
5	1·96594824	1·96774194
6	1·98358284	1·98412698
7	1·99196420	1·99212598
8	1·99603118	1·99607843
9	1·99802947	1·99804305
10	1·99901863	1·99902248
11	1·99951040	1·99951148
12	1·99975550	1·99975580

Table 3 Number of strings per tape, and length of strings in terms of unit records, for the case of $n = 4$

t_1	t_2	t_3	t_4	t_5	i	g_i
85 (1)	56 (1)	**0**	108 (1)	100 (1)	12	349
29 (1)	**0**	56 (4)	52 (1)	44 (1)	11	181
0	29 (7)	27 (4)	23 (1)	15 (1)	10	94
15 (13)	14 (7)	12 (4)	8 (1)	**0**	9	49
7 (13)	6 (7)	4 (4)	**0**	8 (25)	8	25
3 (13)	2 (7)	**0**	4 (49)	4 (25)	7	13
1 (13)	**0**	2 (94)	2 (49)	2 (25)	6	7
0	1 (181)	1 (94)	1 (49)	1 (25)	5	4
1 (349)	0	0	0	**0**	4	1

$$\frac{\log_2 S}{\log_2 \left(\frac{n}{2}\right)} < \frac{n-1}{2(n-2)} \log_2 S ,$$

$$\log_2 \left(\frac{n}{2}\right) > 2\left(1 - \frac{1}{n-1}\right)$$

$$\frac{n}{2} > 2^{2\left(1 - \frac{1}{n-1}\right)}$$

$$n > 2^{3 - \frac{2}{n-1}}$$

i.e.

$$n > 6{\cdot}09$$

Thus for six tapes or less, the polyphase sort is faster than the balanced merge sort.

Acknowledgement

The author is indebted to Dr. M. Bloxham of the Mathematics Department, University of Essex, for suggesting the proof in the Appendix.

Appendix 1

To show that $p(z) = z^n - z^{n-1} - z^{n-2} - \ldots - z - 1$ has one real root in (1, 2) and $(n - 1)$ roots of modulus less than one.

$p(z)$ is continuous, and since $p(1) = -(n - 1) < 0$ and $p(2) = 1 > 0$ there is a real root in (1, 2).

Since $p(z)$ may be expressed as

$$p(z) = \begin{cases} \dfrac{z^n(z-2)+1}{(z-1)} & \text{for } z \neq 1 \\ -(n-1) & \text{for } z = 1 , \end{cases}$$

it follows that the zeroes of $p(z)$ coincide with the zeroes of $f(z) = z^n(z - 2) + 1$ with the exception of the root at $z = 1$ which is clearly not a root of $p(z)$. It is required, therefore, to show that $f(z)$ has precisely n zeroes in $|z| \leqslant 1$.

One proof of this employs a well-known theorem of complex analysis, a preamble to Rouché's theorem sometimes called the principle of the argument, which is used here in a form possibly most familiar as Nyquist analysis among electronic engineers.

Set $g(z) = z^n(z - 2)$. The image of the unit circle $|z| = 1$, described anticlockwise, is a curve that spirals out from $|g| = 1$ to $|g| = 3$ and back again, making n revolutions in all. This analysis is straightforward; a direct intuitive route is to see the image as the path of a moon that makes one orbit of its planet while the planet completes n around the sun. It crosses the negative real axis n times, at $g = -1$, $-(1 + \delta_1)$, $-(1 + \delta_2)$ etc., where δ_1 is a finite positive quantity, calculable for a given n.

Now consider $h(z) = g(z) + 1 + \varepsilon, 0 < \varepsilon < \delta_1$. The image of the unit circle C under the mapping $h(z)$ is the curve we have just described, displaced $1 + \varepsilon$ units to the right. It thus crosses the negative real axis $n - 1$ times (the intercept formerly at -1 now having moved across to $+\varepsilon$). Since the image of C now loops the origin $n - 1$ times there are, by the theorem, $n - 1$ zeroes of $h(z)$ inside C.

Finally, we note that with $\varepsilon \to 0$, $h(z)$ becomes $f(z)$ with a prescription that the zero of f at $z = 1$ is to be counted as lying outside C (it appears near $1 + \varepsilon/(n - 1)$ before the limit is taken).

References

FLORES, I. (1969). *Computer Sorting*, Prentice-Hall Inc., Englewood Cliffs, N.J.
GILSTAD, R. L. (1963). Read-Backward Polyphase Sorting. *CACM*, Vol. 6, pp. 220-223.
MALCOLM, W. D. (1963). String Distribution for the Polyphase Sort. *CACM*, Vol. 6, pp. 217-220.
MANKER, H. H. (1963). Multiphase Sorting, *CACM*, Vol. 6, pp. 214-217.
MARTIN, W. A. (1971). Sorting, *Computing Surveys*, Vol. 3, pp. 147-174.

Correspondence

To the Editor
The Computer Journal

Sir,

Two impressions following the BCS conference on *Computer Performance* just completed are, first, that debugging and performance measurement are very much the same thing and require similar techniques; and, second, that hardware designers have so far done precious little to help. I would like to suggest a facility which promises to help a great deal.

A common bug which can be difficult to find is the sudden appearance of a nonsensical value somewhere in the data space. With current hardware one must normally resort to intellectual exercise or executing the entire program interpretively. What I would like to suggest is hardware address traps which can be set by program. For example one might provide a set of program accessible registers, each containing two addresses, which would cause the hardware to interrupt to one address each time a reference to the other address was encountered. Such a facility would, I suggest, greatly facilitate the provision of decent debugging and monitoring facilities without large run-time or software implementation overheads.

Yours faithfully,
R. J. DAKIN

UKAEA Culham Laboratory
Abingdon
Berkshire
18 September 1972

To the Editor
The Computer Journal

Sir,

Words like 'compactifying' and 'digressionally' ought, in our opinion, to be thoroughly editorised.

Yours faithfully,
D. WHEELER and R. NEEDHAM

University of Cambridge Computer Laboratory
Corn Exchange Street
Cambridge CB2 3QG
12 October 1972

Editor's comment:

Whilst agreeing with the sentiments expressed by the writers of this letter, I wonder if they really feel that the editor has the right to change words which already have been author-ised.