For the record: Batcher's Baffler

In this note we consider a special sorting routine for array \( f(i:0 \leq i < N) \). The predicate OK is given by

\[
\text{OK}.i.j \equiv f.i \leq f.j
\]

and in the following quantifications over it are implicitly constrained by \( 0 \leq i < j < N \). The specification for Batcher's Baffler—the algorithm has been invented by K.E. Batcher and has been given this name by David Gries, presumably because he was baffled by it—is

\[
\begin{align*}
\{ & N: \text{int} \{ N \geq 0 \} \\
 & f(i:0 \leq i < N) \text{ array of int} \{ \text{bag}.f = B \} \\
 & \text{Batcher's Baffler} \\
 & \{ \text{bag}.f = B \land (A_i :: \text{OK}.i.(i+1)) \}
\end{align*}
\]

In the following we shall no longer mention the invariance of \( \text{bag}.f = B \); it is trivially maintained as the algorithm only interferes with the array \( f \) with the operation

\[
\text{Ord}.i.j = \begin{cases} 
\text{if} \ \text{OK}.i.j \rightarrow \text{skip} \\
\text{if} \ \text{OK}.i.j \rightarrow f:\text{swap}(i,j) \\
\{ \text{OK}.i.j \}
\end{cases}
\]
There are all sorts of ways of expressing our postcondition that \( f \) is ascending, but this one is a nice starting point for the invariant

\[ \text{Po: } (\text{Ai:: OK.i.(i+t))} \]

which suggests for Batcher's Baffler the form

"establish \( t > N \)" \{ Po \}

\( \text{; do } t \neq 1 \rightarrow \text{"decrease } t \text{ under invariance of Po \}} \ad \)

Let the decrease of \( t \) under invariance of \( \text{Po} \) involve the transition from \( t=t' \) to \( t=t'' \) with \( t'' < t' \). It would be nice if we could exploit the precondition \( (\text{Ai:: OK.i.(i+t'))} \) by keeping it invariant; we can only hope to do so provided it is implied by the postcondition \( (\text{Ai:: OK.i.(i+t''))} \).

On account of the transition of \( \leq \), this implication holds if \( t'' \) is a factor of \( t' \). Under that constraint the most modest decrease of \( t \) - i.e. the one that strengthens \( \text{Po} \) as little as possible - is halving \( t \); thus we suggest for the invariant \( \text{Pi} \), given by

\[ \text{Pi: } \text{Po} \land t \text{ is a power of 2} \]

and for Batcher's Baffler the form

\[ t:=1; \text{ do } t < N \rightarrow t := 2 \cdot t \text{ ad } \{ \text{Pi} \} \]

\[ \text{; do } t \neq 1 \rightarrow t := t/2 \text{ } \{ (\text{Ai:: OK.i.(i+2 \cdot t))} \} \]

"restore \( \text{Po} \)" \{ (Ai:: OK.i.(i+t)) \} \ad

The rest of this note is concerned with the
algorithm for "restore Po", constrained by

\{ P_2 : (A_i \equiv \text{OK}.i.(i+2:t)) \}

restore Po

\{ P_0 : (A_i \equiv \text{OK}.i.(i+t)) \}

; it treats \( t \) as a constant (for which its being a power of 2 is no longer significant).

with \( e.i \equiv (i \mod 2^t) < t \)

we observe \( e.i \equiv \neg e.(i+t) \)

and rewrite \( P_0 \) as \( P_3 \land P_4 \) with

\( P_3 : (A_i \equiv e.i \equiv \text{OK}.i.(i+t)) \)

and

\( P_4 : (A_i \equiv \neg e.i \equiv \text{OK}.i.(i+t)) \) or, equivalently

\( P_4 : (A_i \equiv e.i \equiv \text{OK}.(i+t).(i+2:t)) \)

The above dichotomy of the desired \( \text{OK} \)-relations is of interest because the argument pairs \( i, (i+t) \)
in \( P_3 \) -and also those in \( P_4 \)-are disjoint on account of \( e.i \equiv \neg e.(i+t) \). Hence \( P_3 \) (or \( P_4 \))
can be established by a whole bunch of \( \text{Ord} \)-operations that could be performed concurrently. (It is
the potential for concurrency that makes Batcher's Baffler of interest.) Using \( \| \) to denote the po-
tentially concurrent combination, we have

(0) \( \{ \text{true} \} (\| i : e.i : \text{Ord}.i.(i+t)) \) \{ P_3 \}

and similarly

(1) \( \{ \text{true} \} (\| : e.i : \text{Ord}.(i+t).(i+2:t)) \) \{ P_4 \}

But one cannot achieve \( P_3 \land P_4 \) by executing the above two consecutively: the second operation
would in general destroy what the first one has accomplished. Let us therefore investigate a strengthening of its precondition such that (1) does not falsify \( P_8 \). For our analysis we generalize (1) by replacing the constant 2 by the parameter \( v \), restricted to even values so as to ensure that all pairs \( i, (i + v \cdot t) \) with \( e.i \) are disjoint. That is, we consider the operation

\[
(2) \quad \left\| i : e.i : \text{Ord.}(i + t), (i + v \cdot t) \right\|
\]

and would like it not to falsify the OK-relations that constitute \( P_3 \). In our analysis we shall use the lemmata and - grudgingly - the notation of EWD932.

(i) OK-relations with no argument involved in an Ord-operation are maintained on account of EWD932, Lemma 0.

(ii) Of OK-relations with one argument involved in an Ord-operation we have two types:

Neither of which, however, is a lemma. However,

represent EWD932, Lemma 3. Note that, \( v \) being even, the two OK-relations added are implied by \( P_2 \). (Here we are getting our first glimpse of how
to exploit precondition $P_2$.

(iii) Finally we investigate the $OK$-relation with both arguments involved in an $Ord$-operation:

$$
\begin{align*}
& i + v \cdot t \\
& \downarrow \\
& i + t \\
& \downarrow \\
& i + v \cdot t + t
\end{align*}
$$

This, again, is not a lemma, but we can recognize the sequence $\rightarrow \rightarrow \rightarrow$ in EWD932, Lemma 4

$$
\begin{align*}
& i + v \cdot t \\
& \downarrow \\
& i + v \cdot t + t
\end{align*}
$$

Two of the added $OK$-relations are again implied by $P_2$, the third one

$$(A_1 : e.i.: OK. (i + t). (i + 2 \cdot v \cdot t))$$

has to be implied by the precondition of (2), which, besides then leaving $P_3$ invariant, establishes

$$(A_1 : e.i.: OK. (i + t). (i + v \cdot t))$$

i.e. the same formula after $v := v/2$. We have found the invariant

$P_5$: $$(A_1 : e.i.: OK. (i + t). (i + v \cdot t)) \land v > 2 \land v \text{ is a power of 2}$$

for "restore $P_0$"; it can be initialized by seeing to it that $v \cdot t \geq N$.

We are left with the duty to demonstrate that (0) and (2) maintain $P_2$. 


For invariance under (0) we have OK-relations

(i) with 0 arguments involved
\[ i \rightarrow i + 2 \cdot t \]  
EWD932, Lemma 0

(ii) with 1 argument involved
\[ i \rightarrow i + 2 \cdot t \]  
EWD932, Lemma 1

(iii) with 2 arguments involved
\[ i \rightarrow i + 2 \cdot t \]  
EWD932, Lemma 2

For invariance under (2) we have OK-relations

(i) with 0 arguments involved
\[ i \rightarrow i + 2 \cdot t \]  
EWD932, Lemma 0
\[ i + t \rightarrow i + 3 \cdot t \]

(ii) with 1 argument involved
\[ i + v \cdot t - 2 \cdot t \rightarrow i + v \cdot t \]  
EWD932, Lemma 1

(iii) with 2 arguments involved
\[ i + v \cdot t \rightarrow i + v \cdot 2 + 2 \cdot t \]  
EWD932, Lemma 2
\[ i + t \rightarrow i + 3 \cdot t \]
And now we are ready for the final version of Batcher's Baffler

\[
\begin{align*}
&\text{if } t, v_0: \text{int} \\
&; t := 1; \text{ do } t < N \rightarrow t := 2 \cdot t \text{ od } \text{; } v_0 := 1 \{P_1 \wedge t \cdot v_0 \geq N\} \\
&; \text{ do } t \neq 1 \rightarrow t, v_0 := t/2, 2 \cdot v_0 \{P_2 \wedge t \cdot v_0 \geq N\} \\
&; (|| i: \text{e.i.: Ord. } i + t \}) \{P_2 \wedge P_3 \wedge t \cdot v_0 \geq N\} \\
&; || v: \text{int } \text{; } v := v_0 \{P_2 \wedge P_3 \wedge P_5\} \\
&; \text{ do } v \neq 2 \rightarrow v := v/2 \\
&; (|| i: \text{e.i.: Ord. } i + t \cdot (i + v \cdot t)) \text{ od } \\
&\text{ od } \\
&\text{ od }
\end{align*}
\]

Remark For \(N = 4\), Batcher's Baffler generates the operations described on EWD 932-2. (End of Remark.)

I am indebted to David Gries and the members of the ETAC (= Eindhoven Tuesday Afternoon Club).

Austin, 9 September 1985

prof. dr. Edsger W. Dijkstra
Department of Computer Sciences
The University of Texas at Austin
Austin, Texas, 78712-1188
United States of America