For the record: Batcher's Baffler

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

$$OK.i.j \equiv f.i \le f.j$$

and in the following quantifications over it are implicitly constrained by $0 \le i < j < N$. 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

$|[$ N: int $\{N \ge 0\}$
; $|[$ $f(i: 0 \le i < N)$ array of int $\{bag.f = B\}$
  ; Batcher's Baffler
   $\{bag.f = B \wedge (\mathbf{A}i :: OK.i.(i+1))\}$
  $]|$
$]|$

In the following we shall no longer mention the invariance of $bag.f = B$; it is trivially maintained as the algorithm only interferes with the array f with the operation

Ord.i.j = **if** OK.i.j $\rightarrow$ skip
  $[]$ $\neg$OK.i.j $\rightarrow$ f:swap(i,j)
  fi $\{OK.i.j\}$

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

P0: $(\underline{A}i :: OK.i.(i+t))$

which suggests for Batcher's Baffler the form

"establish $t \geq N$" {P0}
; do $t \neq 1$ → "decrease t under invariance of P0" od.

Let the decrease of t under invariance of P0 involve the transition from $t = t'$ to $t = t''$ with $t'' < t'$. It would be nice if we could exploit the precondition $(\underline{A}i :: OK.i.(i+t'))$ by keeping it invariant; we can only hope to do so provided it is implied by the postcondition $(\underline{A}i :: OK.i.(i+t''))$. On account of the the transitivity 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 P0 as little as possible— is halving t; thus we suggest for the invariant P1, given by

P1: P0 $\wedge$ t is a power of 2

and for Batcher's Baffler the form

$t := 1$; do $t < N$ → $t := 2 \cdot t$ od {P1}
; do $t \neq 1$ → $t := t/2$ $\{(\underline{A}i :: OK.i.(i+2 \cdot t))\}$
"restore P0" $\{(\underline{A}i :: OK.i.(i+t))\}$
od

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

$\{P2\colon (\underline{A} i :: OK.i.(i+2\cdot t))\}$
restore Po
$\{P0\colon (\underline{A} i :: OK.i.(i+t))\}$;

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

With $e.i \equiv (i \underline{\text{mod}}\ 2\cdot t) < t$

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

and rewrite P0 as $P3 \wedge P4$ with

P3: $(\underline{A} i : e.i : OK.i.(i+t))$ and

P4: $(\underline{A} i : \neg\, e.i : OK.i.(i+t))$ or, equivalently

P4: $(\underline{A} i : e.i : OK.(i+t).(i+2\cdot t))$.

The above dichotomy of the desired OK-relations is of interest because the argument pairs $i, (i+t)$ in P3 —and also those in P4— are disjoint on account of $e.i \equiv \neg\, e.(i+t)$. Hence P3 (or P4) can be established by a whole bunch of Ord-operations that could be performed concurrently. (It is the potential for concurrency that makes Batcher's Baffler of interest.) Using $\|$ to denote the potentially concurrent combination, we have

(0) $\{true\}\ (\| i : e.i : Ord.i.(i+t))\ \{P3\}$

and similarly

(1) $\{true\}\ (\| : e.i : Ord.(i+t).(i+2\cdot t))\ \{P4\}$.

But one cannot achieve $P3 \wedge P4$ 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 P3. For our analysis we generalize (1) by replacing the constant 2 by the parameter v, restricted to <u>even</u> 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) $\quad (\| i : e.i : \mathrm{Ord}.(i+t).(i+v \cdot t))$

and would like it not to falsify the OK-relations that constitute P3. In our analysis we shall use the lemmata and – grudgingly – the notation of EWD 932.

(i) OK-relations with no argument involved in an Ord-operation are maintained on account of EWD 932, Lemma 0.

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

$$\begin{tikzcd} i \arrow[d] & i+v\cdot t \\ i+t \arrow[ur, dashed] & \end{tikzcd} \qquad\qquad \begin{tikzcd} & i+v\cdot t \arrow[d] \\ i+t \arrow[ur, dashed] & i+v\cdot t+t \end{tikzcd} ,$$

neither of which, however, is a lemma. However

$$\begin{tikzcd} i \arrow[r] \arrow[d] & i+v\cdot t \\ i+t \arrow[ur, dashed] & \end{tikzcd} \qquad\qquad \begin{tikzcd} & i+v\cdot t \arrow[d] \\ i+t \arrow[ur, dashed] \arrow[r] & i+v\cdot t+t \end{tikzcd}$$

represent EWD 932, Lemma 3. Note that, v being even, the two OK-relations added are implied by P2. (Here we are getting our first glimpse of how to exploit precondition $P2$.)

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

$$\begin{tikzcd}
 & i+v\cdot t \arrow[d] & i+2\cdot v\cdot t \\
i+t \arrow[ur, dashed] & i+v\cdot t+t \arrow[ur, dashed] &
\end{tikzcd}$$

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

$$\begin{tikzcd}
 & i+v\cdot t \arrow[d] \arrow[r] & i+2\cdot v\cdot t \\
i+t \arrow[ur, dashed] \arrow[r] \arrow[urr] & i+v\cdot t+t \arrow[ur, dashed] &
\end{tikzcd}$$

Two of the added OK-relations are again implied by $P2$, the third one

$$(\underline{A}\, i : e.i : OK.(i+t).(i+2\cdot v\cdot t))$$

has to be implied by the precondition of (2), which, besides then leaving $P3$ invariant, establishes

$$(\underline{A}\, i : e.i : OK.(i+t).(i+v\cdot t)) \quad ,$$

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

$P5$: $\quad (\underline{A}\, i : e.i : OK.(i+t).(i+v\cdot t)) \wedge v \geq 2 \wedge$
$\quad\quad v$ is a power of 2 .

for "restore $P0$"; 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 $P2$.

For invariance under (0) we have OK - relations

(i) with 0 arguments involved

$i \longrightarrow i+2\cdot t$ EWD932, Lemma 0

(ii) with 1 argument involved

$$\begin{tikzcd} i \arrow[r] \arrow[d, dashed] & i+2\cdot t \\ i+t & \end{tikzcd}$$

EWD932, Lemma 1

(iii) with 2 arguments involved

$$\begin{tikzcd} i \arrow[r] \arrow[d, dashed] & i+2\cdot t \arrow[d, dashed] \\ i+t \arrow[r] & i+3\cdot t \end{tikzcd}$$

EWD932, Lemma 2

For invariance under (2) we have OK-relations

(i) with 0 arguments involved

$i \longrightarrow i+2\cdot t$ EWD932, Lemma 0

$i+t \longrightarrow i+3\cdot t$ "

(ii) with 1 argument involved

$$\begin{tikzcd} i+v\cdot t-2\cdot t \arrow[r] & i+v\cdot t \\ i+t \arrow[ur, dashed] & \end{tikzcd}$$

EWD932, Lemma 1

$$\begin{tikzcd} & i+v\cdot t \\ i+t \arrow[ur, dashed] \arrow[r] & i+3\cdot t \end{tikzcd}$$

"

(iii) with 2 arguments involved

$$\begin{tikzcd} & i+v\cdot t \arrow[r] & i+v\cdot 2+2\cdot t \\ i+t \arrow[ur, dashed] \arrow[rr] & & i+3\cdot t \arrow[u, dashed] \end{tikzcd}$$

EWD932, Lemma 2

And now we are ready for the final version of Batcher's Baffler

$[\![$ t, v0: int
; t:=1 ; do t < N → t := 2·t od ; v0:=1 $\{P1 \wedge t \cdot v0 \geq N\}$
; do t ≠ 1 → t, v0 := t/2, 2*v0 $\{P2 \wedge t \cdot v0 \geq N\}$
    ; (‖i: e.i : Ord.i.(i+t)) $\{P2 \wedge P3 \wedge t \cdot v0 \geq N\}$
    ; $[\![$ v: int ; v := v0 $\{P2 \wedge P3 \wedge P5\}$
      ; do v ≠ 2 → v := v/2
          ; (‖i: e.i : Ord.(i+t).(i+v·t))
      od
    $]\!]$
  od
$]\!]$

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

I am endebted 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