The Saddleback Search

The origin of this algorithm is unknown; its name has been invented by David Gries. It solves a problem that can be stated in many variations; we shall first solve it in one of its straightforward versions and then discuss several variations.

We are given an integer function \( f \) of two natural arguments that is increasing in both its arguments and takes on the value \( F \) at least once. Saddleback Search has to locate such an occurrence, more precisely, the occurrence with the smallest value of the first argument. Because \( f \) is increasing in both its arguments, this is at the same time the occurrence with the largest value of the second argument. Thus we are lead to the following formal specification

\[
\begin{align*}
&\text{IF: } \text{int}; \ f(i,j: 0 \leq i \land 0 \leq j) \ \text{array of int} \\
&\{ (\exists i,j, ii, jj: 0 \leq i < ii \land 0 \leq j < jj; \ f(i,j) < f(ii,j) \land f(i,j) < f(i, jj)) \land \\
&f(X,Y) = F \land (\exists i,j: 0 \leq i < X \land j > Y; \ f(i,j) \neq F) \} \\
\text{; IF x, y: int} \end{align*}
\]

; Saddleback Search

\[
\{ \text{R: } x, y = X, Y \}
\]

;
The analogy with the Linear Search now suggests to approach \( X \) from below and \( Y \) from above, i.e. to iterate with the invariant \( P \), given by
\[
P: \quad 0 \leq x < X \land y \geq Y
\]
Our first approximation of Saddleback Search keeps the analogy to the Linear Search as close as possible:

\[
\text{"establish } P \" \\
; \text{ do } x < X \rightarrow x := x + 1 \{ P \} \quad \text{if} \quad y > Y \rightarrow y := y - 1 \{ P \} \text{ od } \{ R \}.
\]
We note that this program differs in two respects from the corresponding approximation — see EWD930 — of the Linear Search: Firstly — on account of the conjunct \( y \geq Y \) — the initialisation cannot be done independently of \( f \) ("\( y := +\infty \)" is not acceptable) and secondly the repetition here is nondeterministic.

In order to relate our inequalities involving \( X \) and \( Y \) to \( f \), we observe the

Lemma \( x \leq X \land y \geq Y = \)
\[
(\forall i,j : 0 \leq i < x \lor j > y : f(i,j) \neq F)
\]
Proof
\[
x \leq X \land y \geq Y \\
\Rightarrow \{ (\forall i,j : 0 \leq i < x \lor j > y : f(i,j) \neq F) \\
(\forall i,j : 0 \leq i < x \lor j > y : f(i,j) \neq F) \\
\Rightarrow \{ f(X,Y = F \land X \geq 0 \} \\
X \geq x \land Y \leq y
\]
(End of Proof)

With the above Lemma in our hands we now tackle the guards of the repetition, strengthened by the invariant:
\[ \text{P} \land x < x \]
\[ = \{ \text{arithmetic and definition of P} \} \]
\[ P \land x + 1 \leq x \land y > y \]  
\[ = \{ \text{Lemma} \} \]
\[ P \land (\forall i: 0 \leq i < x + 1 \lor j \geq y: f_i.j \neq F) \]
\[ = \{ \text{definition of P, Lemma and predicate calculus} \} \]
\[ P \land (\forall i: 0 \leq i \leq y: f_i.j \neq F) \]
\[ \equiv \{ f \text{ is increasing in its second argument} \} \]
\[ P \land f_i.x.y < F \]  
\[ \text{and} \]
\[ P \land y > x \]
\[ = \{ \text{arithmetic and definition of P} \} \]
\[ P \land x < x \land y - 1 > y \]  
\[ = \{ \text{Lemma} \} \]
\[ P \land (\exists i: 0 \leq i < x \lor j \geq y - 1: f_i.j \neq F) \]
\[ = \{ \text{definition of P, Lemma and predicate calculus} \} \]
\[ P \land (\exists i: i > x: f_i.y \neq F) \]
\[ \equiv \{ f \text{ is increasing in its first argument} \} \]
\[ P \land f_i.x.y > F \]

Hence we find ourselves invited to consider the second approximation with the (conditionally) strengthened guards

"establish P"

\[ ; \text{do } f_i.x.y < F \rightarrow x := x + 1 \{ P \}_3 \land f_i.x.y > F \rightarrow y := y - 1 \{ P \}_3 \text{ od} \]

**for which we have to check that, though the guards have been strengthened, the final conclusion \( R \) is still justified. Indeed:**

\[ P \land f_i.x.y > F \land f_i.x.y < F \]
\[ = \{ \text{arithmetic} \} \]
\[ P \land f_i.x.y = F \]
\[ = \{ \text{definition of P and } X, Y \} \]
\[ x, y = X, Y \]

Because \( f \) is increasing in both arguments, \( x = 0 \land f(y) \geq F \Rightarrow P \). Thus we arrive at a complete program for the Saddleback Search:

\[
\begin{align*}
  & x, y := 0, 0 \\
  & \text{do } f(x, y) < F \rightarrow y := y + 1 \text{ od } \{ P \} \\
  & \text{do } f(x, y) < F \rightarrow x := x + 1 \\
  & \text{while } f(x, y) > F \rightarrow y := y - 1 \\
  & \text{od } \{ R \}
\end{align*}
\]

Convergence of the first repetition is guaranteed by the fact that \( f(x, y) \) is increasing in its second component.

\[ \star \quad \star \quad \star \]

We observed at the beginning that the occurrence of \( F \) with the smallest value of the first argument is also that with the largest value of the second argument. Consequently we could also have defined \( X, Y \) as the solution of \( x, y : (f(x, y) = F) \) with the minimum value for \( x - y \):

\[ P(x, y) = F \land (\exists i, j : 0 \leq i \land j \geq 0 \land i - j < x - y \land f(i, j) = F) \]

The disadvantage of this definition is that, for the proof of our lemma, another appeal to \( f \)'s double monotonicity would be required. It has the advantage that a first approximation —viz. to investigate values of \( f(x, y) \) for increasing values of \( x - y \)— would have been a more direct analogue of the Linear Search. The approach has a further heuristic virtue.
Under the invariant \( x - y \leq X - Y \) the search would continue until \( x - y = X - Y \). But those two conditions do not imply \( x, y = X, Y \) ! However

\[
x, y = X, Y \equiv x - y = X - Y \land x \leq X \land y \geq Y
\]

- a nice little theorem I did not know - and we are thus led to the stronger invariant \( x \leq X \land y \geq Y \). (I did this derivation as well, and it was kind of nice; it was essentially the case analysis needed in the proof of the Lemma, that put me off.)

* * *

The first variation is Saddleback Count, which, instead of locating an occurrence, counts the number of occurrences. It does so in the order of increasing \( x - y \). Formally specified

\[
\begin{align*}
[F: \text{int} ; f(i,j): 0 \leq i \land 0 \leq j) \text{ array of int} \\
\{ (i, j, ii, jj: 0 \leq i < ii \land 0 \leq j < jj) ; f(i, j) < f(ii, j) \land f(i, jj) \land \\
K = (N (i, j): 0 \leq i \land 0 \leq j ; f(i, j) = F) \}
\end{align*}
\]

; \[ k: \text{int} \]

; Saddleback Count

\[
\{ R: k = K \}
\]

The invariant \( P \) is given by

\[
P: k = (N (i, j): 0 \leq i < x \lor j > y : f(i, j) = F)
\]
or, equivalently
\[ P: k + (\mathbb{N}(i,j): i \geq x \land 0 \leq j \leq y: f(i,j) = F) = k \]

A solution for Saddleback Count is

\[
\begin{align*}
\text{let } & x,y: \text{ int} \\
& x,y,k := 0,0,0; \ \text{ do } f(x,y < F \rightarrow y := y+1 \ \text{ od } \{P\} \\
& \text{ do } y \geq 0 \rightarrow \text{ if } f(x,y < F \rightarrow x := x+1} \\\n& \quad \text{ if } f(x,y > F \rightarrow y := y-1} \\\n& \quad \text{ if } f(x,y = F \rightarrow x,y,k := x+1,y-1,k+1} \\
& \text{ od } \{P\} \\
& \text{ od } \{R\} \\
\end{align*}
\]

which, I trust, now requires no further explanation.

\[ \ast \ast \ast \]

A next variation is that in the declaration of \( f \) the first argument is bounded by \( 0 \leq i < I \) and/or the second argument is bound by \( 0 \leq j < J \).

\( 0 \leq i < I \): this bound has no influence on the text of Saddleback Search; for Saddleback Count the guard \( y \geq 0 \) of the last repetition has to be replaced by the stronger \( x < I \land y \geq 0 \) so as to prevent "index out of bounds". The proper reformulation of the invariants is left as an exercise to the reader.

\( 0 \leq j < J \): in Saddleback Search \( P \) is established by \( x,y := 0,J-1 \), in Saddleback Count by \( x,y,k := 0,J-1,0 \).
The next variation to consider is a weakening of the monotonicity requirements on $f$ from increasing to ascending.

Saddleback Search is also okay for an $f$ that is ascending in its first and increasing in its second argument. If $f$ is only given to be ascending in both arguments, the program for bounded second argument is still okay, but for unbounded second argument the establishment of $P$ has to be effectuated by

$$x,y := 0,0$$
$$\text{do } f.x.y \leq F \rightarrow y := y+1 \text{ od}$$

and $f$ has to be such that this repetition converges, i.e., for increasing $y$, $f.0.y$ has to grow beyond $F$.

In the case of Saddleback Count we have in any case to insist that $K$ exists (i.e. is finite), which is the case if both arguments are bounded from above or $f.x.y$ grows beyond $F$ for increasing $x+y$. If, in addition, $f$ is increasing in one of its arguments, its second one, say, it suffices to modify the third guarded command of the alternative construct from

$$f.x.y = F \rightarrow x,y,k := x+1, y-1, k+1$$

into

$$f.x.y = F \rightarrow x,k := x+1, k+1$$

(the original being an optimization of the latter which is valid if $f$ is increasing in its first argument).
For Saddleback Count applied to an \( f \) given to be ascending in both unbounded arguments and \( f \cdot x \cdot y \) growing beyond \( F \) for increasing \( x+y \), we strengthen the invariant to \( P \land Q \) with \( P \) (as before) given by

\[
\begin{align*}
P : \quad & k + (N(i,j) : i \geq x \land 0 \leq j \leq y : f \cdot i \cdot j = F) = k \\
\text{and } \quad & Q \text{ given by} \\
Q : \quad & (A_i : x \leq i < z : f \cdot i \cdot y \leq F)
\end{align*}
\]

Invariant \( Q \) states the relevant property of \( z \) which has been introduced for the purpose of efficiency. Note that neither \( x := x+1 \) nor \( y := y-1 \) falsifies \( Q \), nor \( z := z \max x \).

\[
; x, y, z, k := 0, 0, 0, 0 ; \text{ do } f \cdot x \cdot y \leq F \rightarrow y := y+1 \text{ od } \{ P \land Q_3 \} \\
; \text{do } y > 0 \rightarrow \\
\begin{align*}
& \text{if } f \cdot x \cdot y < F \rightarrow x := x+1 \text{ } \{ P \land Q_3 \} \\
& \text{if } f \cdot x \cdot y > F \rightarrow y := y-1 \text{ } \{ P \land Q_3 \} \\
& \text{if } f \cdot x \cdot y = F \rightarrow z := z \max x \text{ } \{ Q' : (\exists i : x \leq i < z : f \cdot i \cdot y = F) \} \\
& \text{do } f \cdot z \cdot y = F \rightarrow z := z+1 \text{ od } \{ Q' \land f \cdot z \cdot y \neq F \} \\
& y, k := y-1, k+z-x \text{ } \{ P \land Q_3 \}
\end{align*}
\]

;
We finally mention that it may be worthwhile to replace the linear searches by logarithmic ones. We could replace, for instance,
\[ \text{do } f.x.y < F \rightarrow y := y + 1 \text{ od} \]
by
\[ \text{if } f.x.y > F \rightarrow \text{skip} \]
\[ \text{if } f.x.y \leq F \rightarrow \]
\[ \text{if } v : \text{int} ; v := 1 \{ f.x.y \leq F \text{ and } v \text{ is power of } 2 \} \]
\[ ; \text{do } F \geq f.x.(y+v) \rightarrow v := 2 \times v \text{ od} \]
\[ \{ f.x.y \leq F < f.x.(y+v) \text{ and } v \text{ is power of } 2 \} \]
\[ ; \text{do } v \neq 1 \rightarrow v := v/2 \]
\[ \text{if } f.x.(y+v) \leq F \rightarrow y := y + v \]
\[ \text{if } F < f.x.(y+v) \rightarrow \text{skip} \]
\[ \text{od} \; ; y := y + 1 \{ f.x.(y-1) \leq F < f.x.y \} \]
\[ \} \]

The other three linear searches can be treated similarly. Note that the worst case remains linear, viz. when in the original execution we have a long execution of alternations of \( x := x + 1 \) and \( y := y - 1 \).

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

Austin 5 Sep. 1985