CONDITIONS FOR INCREMENTAL ITERATION: EXAMPLES AND COUNTEREXAMPLES

B.G. RYDER,* T.J. MARLOWE and M.C. PAULL

Department of Computer Science, Rutgers University, New Brunswick, NJ 08903, USA

Communicated by R. Sethi
Received May 1986
Revised March 1987

Abstract. Iterative algorithms for fixed points of systems of equations are of importance in graph algorithms, data flow analysis and other areas of computer science. One commonly-sought extension is an incremental update procedure, which responds to small changes in problem parameters by obtaining the new fixed point from perturbation of the previous solution. One approach which has been suggested is to iterate for the new fixed point beginning at that previous solution, possibly after some small modifications. Our results show that this procedure is not in general correct. We give sufficient conditions for correctness, and give counterexamples in Boolean algebra and data flow analysis showing that difficulties with the proposed algorithms can occur in practice.

1. Introduction

Iteration is a solution procedure often used for problems formulated as systems of equations. Iteration is applicable to a wide range of problems, since it requires only weak conditions on the form and operators of the system of equations and the underlying domain [9]. Application areas range from the solution of numerical equations to data flow analysis.

Iterative solution of a system of equations is a standard technique in data flow analysis of computer programs [11, 12]. In static analysis of large, evolving software systems, it is too expensive to fully reanalyze a system each time it changes; instead, incremental update techniques have been developed which update the data flow information consistent with the current state of the system [4, 8, 10, 14, 15, 16, 17, 19]. Some of these suggested algorithms use iteration and recompute data flow information incrementally, by restarting iteration from the previous solution after a problem change [8, 10]. The results presented here address the issue of when this technique can succeed and when it cannot. We give theoretical results which delineate a class of cases for which this procedure will work; we demonstrate that it fails in other cases. To explore this issue, we consider changes in the context of least fixed points on partially ordered sets.

* Research supported by National Science Foundation grant NSF DCR85-13730.

Suppose that a change in some underlying parameters transforms an equation \( X = A(X) \) into a related equation \( Y = B(Y) \). The question of interest is "When can an iterative solution to the new equation \( Y = B(Y) \) be found by restarting iteration from a solution to the previous equation \( X = A(X) \) (or one of its iterates)?"

\( Q \) is a *bounded set* if \( Q \) is partially ordered by a relation \( \leq \) and has a least element \( 0. \) A function \( F: Q \to Q \) is *monotonic* if for \( X \) and \( Y \) in \( Q \), \( X \leq Y \) implies \( F(X) \leq F(Y) \). A *fixed point* of \( F \) is an element \( L \) of \( Q \) such that \( F(L) = L \). A fixed point accessible by iteration is called a *limit*.

The iterative approach to computing the least fixed point of the equation \( X = F(X) \) is to successively apply \( F \) to \( 0 \), computing \( F^j(0) \), for \( j \geq 0 \), where we use the standard convention:

\[
F^0(X) = X, \quad F^{i+1}(X) = F(F^i(X)).
\]

A fixed point \( L \) of any monotonic function \( F \) must be "above" each \( F^j(0) \); that is, \( L \geq F^j(0) \), for all \( j \geq 0 \). This follows by induction and monotonicity of \( F \) since \( L \geq 0 \) gives

\[
L = F(L) \geq 0.
\]

The main theorem of this paper applies to related systems of equations \( X = A(X) \) and \( Y = B(Y) \) satisfying \( B(Z) \geq A(Z) \) for all \( Z \) in \( Q \). We show that a fixed point \( L_B \) of \( B \), if one exists, can be found by iterating \( A \) starting at \( 0 \), reaching some value \( X_A \), and then iterating \( B \) starting at \( X_A \). Furthermore, this approach involves at most as many applications of \( B \) to attain \( L_B \) as successive application of \( B \) to 0. In other words, the sequence \( \langle B^j(A^i(0)) \rangle_{i,j} \) attains its limit at least as rapidly as the sequence \( \langle B^j(0) \rangle \).

Thus, if a system of equations is changed after a limit to the previous problem has been found, or after a number of iterations have occurred (e.g., because of changes in the underlying problem), we may continue iteration from the current value if the above criterion is met, or in certain other cases. We show however that iteration cannot always be restarted with that value if the criterion is not met.\(^2\)

In the iterative approach, one of four situations must apply to the sequence \( \langle F^j(0) \rangle \):

1. \( \langle F^j(0) \rangle \) converges in a finite number of steps to a fixed point \( L \) of \( F \); it is easy to show that \( L \) must then be the least fixed point of \( F \).
2. \( \langle F^j(0) \rangle \) is infinite, has a least upper bound \( L \) in \( Q \), and \( L \) is a fixed point; we say that \( L \) is the *infinite limit* of the sequence.
3. \( \langle F^j(0) \rangle \) has no least upper bound in \( Q \); that is, \( Q \) is not a "complete partial order."

\(^1\) \( 1 \) is the greatest element of \( Q \) if one exists.

\(^2\) Thus, our theorem gives a sufficient but not necessary condition for restarting iteration.
Conditions for incremental iterations

(4) $\langle F'(0) \rangle$ has a least upper bound which is not a fixed point of $F$; in this case, $F$ is not a continuous function.\(^3\)

Examples of the first three cases are:

1. The transitive closure of a finite graph $G = \langle V, E \rangle$ for the equation $W(X) = X \circ E \cup E$, where $\circ$ is composition of relations, and $\emptyset = \{ \}$. In this case, the sequence reaches the least fixed point when $j$ is the length of the longest simple path or cycle in $G$.

2. The Kleene closure $A^*$ for a subset $A$ of a language $S$ and the equation $W(X) = X \cdot A \cup A$, where $\cdot$ is concatenation, and $\emptyset$ is the empty string $A$.

3. The function $W(X) = X + 1$ on the integers.

It should be noted that in case (4), a function could have a least fixed point which is not a limit, even in the sense of infinite limit. We will see an example of this situation in Section 5.

In the remainder of this paper, we present examples drawn from graph algorithms, set equations and data flow analysis, which illustrate successful and unsuccessful restarting of iteration as an incremental technique. We give counterexamples for two standard data flow problems. Then we prove our theoretical results and extend them to deal with infinite limits.

2. Two examples

The two problems in this section illustrate case (1). The shortest path problem for a directed graph finds the minimal length path between two nodes in a graph whose edges are marked with positive distances. Incremental iteration trivially succeeds, since this function has a unique fixed point, and the bounded set of distances has a 1, namely, the sum of all of the edge distances. Indeed, after a change to an edge distance, we can restart iteration at any point and still converge to the unique fixed point. The union-intersection set problem seeks the solution of a set equation whose operators are set union and intersection, defined over a finite domain. Such equations arise in graph and data flow problems. Here, the fixed point of the equation is not unique. Unlike the shortest path example, there are some changes which cannot be accommodated by restarting iteration at any point, as shown below.

2.1. Shortest path in a directed graph

Consider the problem of finding the shortest path from $v_1$ to $v_6$ in the directed graph of Fig. 1. We trace the solution of the shortest path equations by iteration (see [1, Section 5.6]), before and after a change in the problem parameters.

\(^3\) In the sense of Scott [12, 21].
Letting $x_i$ be the minimum distance from $v_i$ to $v_n$, we find the following set of equations:

$$
\begin{align*}
  x_1 &= \min\{1 + x_2, 4 + x_5\}, \\
  x_2 &= 1 + x_4, \\
  x_3 &= 1 + x_5, \\
  x_4 &= \min\{1 + x_3, 4 + x_5\}, \\
  x_5 &= \min\{1 + x_2, 4 + x_6\}, \\
  x_6 &= 0.
\end{align*}
$$

Define:

$$X = (x_1, x_2, \ldots, x_6) \text{ and } X^0 = 0 = (0, 0, 0, 0, 0, 0).$$

Then:

$$W(X^i) = X^{i+1} = (\min\{1 + x_2^{(i)}, 4 + x_5^{(i)}\}, 1 + x_4^{(i)}, 1 + x_5^{(i)},$$

$$\min\{1 + x_3^{(i)}, 4 + x_5^{(i)}\}, \min\{1 + x_2^{(i)}, 4 + x_6^{(i)}\}, 0) \text{ for } i \geq 0.$$

We can then show:

$$
\begin{align*}
  W^1(0) &= (1, 1, 1, 1, 1, 0), \\
  W^2(0) &= (2, 2, 2, 2, 2, 0), \\
  W^3(0) &= (3, 3, 3, 3, 3, 0), \\
  W^4(0) &= (4, 4, 4, 4, 4, 0), \\
  W^5(0) &= (5, 5, 5, 5, 4, 0), \\
  W^6(0) &= (6, 6, 5, 6, 4, 0), \\
  W^7(0) &= (7, 7, 5, 6, 4, 0), \\
  W^8(0) &= W^8(0) = (8, 7, 5, 6, 4, 0)
\end{align*}
$$

whence $(8, 7, 5, 6, 4, 0)$ is a fixed point of $W$.

Now suppose the weights on the edges $(3, 4)$ and $(5, 4)$ are changed so that the initial equation for $x_4$ becomes:

$$x_4 = \min\{2 + x_3, 4 + x_3\}.$$
be guaranteed to arrive at the correct answer. Can we begin iteration instead with
the solution we have already obtained for $W$, and still obtain the proper fixed point
for $V$?

Using initial value $W^8(0) = X^8 = (8, 7, 5, 6, 4, 0)$, we find:

$$V'(X^8) = (8, 7, 5, 7, 4, 0), \quad V'(X^8) = V'(X^8) = (8, 8, 5, 7, 4, 0).$$

Thus certainly $(8, 8, 5, 7, 4, 0)$ is a fixed point of $V$, and, in fact, the same fixed point
we would find had we started at 0 (or in this case, anywhere else).

### 2.2. Union-intersection set equations

For union-intersection equations, we will not always be able to restart iteration.
Systems of "linear" union-intersection equations occur in data flow analysis, in
graph problems such as reachability and dominators, and in Boolean algebra.

Consider the set equation:

$$W(X) = X \cap A \cup B$$

where $A$ and $B$ are subsets of the finite set $S = \{1, 2, 3, 4, 5\}$. The partial order is set
inclusion and $\emptyset = \{\}$. We see that for any $A$ and $B$ chosen:

$$W^0(\emptyset) = \{ \}, \quad W^2(\emptyset) = W'(\emptyset) = B,$$

therefore, the least fixed point of $W$ is $B$. If we take $A = \{1, 2\}$ and $B = \{2, 3, 4\}$ we
find the fixed point $W^2(\emptyset) = \{2, 3, 4\}$. We change the equation to:

$$U(X) = X \cap A \cup C$$

where $B \subseteq C, \quad C = \{2, 3, 4, 5\}$. Then, if we start iteration from 0, we find $U^2(\emptyset) = \{2, 3, 4, 5\}$, the least fixed point of $U$. Now if we restart iteration of $U$ from the old
fixed point of $W$:

$$U^1(\{2, 3, 4\}) = \{1, 2\} \cap \{2, 3, 4\} \cup \{2, 3, 4, 5\} = \{2, 3, 4, 5\},$$

$$U^2(\{2, 3, 4, 5\}) = \{1, 2\} \cap \{2, 3, 4, 5\} \cup \{2, 3, 4, 5\},$$

obtaining the same least fixed point, $\{2, 3, 4, 5\}$. However, for $C = \{3, 4\}$ we obtain
$U^2(\emptyset) = \{2, 4\}$, whereas, by restarting iteration at the fixed point of $W$, we obtain:

$$U^1(\{2, 3, 4\}) = \{1, 2\} \cap \{2, 3, 4\} \cup \{3, 4\} = \{2, 3, 4\}$$

which is an "erroneous" least fixed point. We can show that whenever $B \subseteq C$ as in
the first case, the fixed point of equation (2.1) is less than the fixed point of (2.2).
If, as in the second case this condition fails, starting iteration in (2.2) at the fixed
point of (2.1) will not necessarily converge to the least fixed point of (2.2).

### 3. Applications to data flow analysis

As mentioned in Section 1, incremental algorithms for data flow analysis have
been designed to deal with changes in the programs being analyzed. Our theorems

\footnote{Actually $A \cap B \subseteq C$ insures convergence here.}
demonstrate that one cannot always restart iteration from an old fixed point (or an
old iterate), although this strategy was suggested for the formal bound set problem
in [8] and the aliasing problem in [10]. In this section we present two counter-
examples from data flow analysis: calculating the bound sets of formal call-by-
reference parameters and calculating the definitions of variables which reach a node
in a flow graph. The latter is representative of the four classical data flow problems:
reaching definitions, live uses of variables, available expressions and very busy
expressions, whose solutions are used in compiler optimizations [2,11]. A related
counterexample based on our work applies to the finding of aliases among global
variables and formal parameters in a program [4]. Further discussion of these data
flow algorithms can be found in [5]. Some generalizations arising from these
counterexamples demonstrate the practical impact of the results in Section 4.

One of the standard problems in interprocedural data flow analysis is determina-
tion of the formal bound set of a procedure \( P \), that is, finding all possible dynamic
associations between each formal reference parameter of \( P \) and other formal refer-
ence parameters [4,8]. A formal reference parameter is a parameter of a procedure
which is passed using a call-by-reference mechanism [2]. This problem is solved by
taking the transitive closure of \((\text{argument}, \text{formal reference parameter})\) pairs in
procedure calls where the argument itself is a formal reference parameter. For a
more formal definition of this problem, let \( G \) be a directed graph with vertices \( V \)
and edges \( E \). Let \( Q \) be the power set of \( V \times V \) ordered by set inclusion with \( \emptyset = \{ \} \)
and \( V = V \times V \). Let

\[
A \circ B = \{(x, z) \mid \exists y (x, y) \in A \land (y, z) \in B\},
\]

\[
W(X) = X \circ E \cup E,
\]

where \( V \) = the set of formal parameters of the program, and \( E \) = the set of initial
bound pairs, that is, all those pairs directly discernible from the bindings at some
call site in the program. These equations can be solved by using iterative [8] or
elimination methods [4].

A proposed algorithm for incremental update of the iterative solution [8] suggests
 recomputation of the formal bound set after deletion of a call site, initializing \( X \)
as the bound set found in the previous invocation. As shown in Theorem 4.3, there
are cases for which this procedure is correct; however, the following example shows
that there are situations in which this initialization will lead to the wrong solution.

Consider the fragment of program \( A \) given in Fig. 2, where \( q, r, \) and \( s \) are mutually
recursive procedures also containing code for termination of the loops. In addition,
suppose program \( A \) is then transformed into program \( B \) by replacement of statement
\( L \) by:

\[
L' : r(a + 1); \quad \{\text{this eliminates graph edge } (a, b)\}.
\]

Define functions for the formal bound sets of program \( A \) (i.e., \( W \)) and program \( B \)
i.e., \( U \)):

\[
W(X) = X \circ E \cup E, \quad U(X) = X \circ F \cup F,
\]
Program fragment:

```plaintext
procedure p (var d: integer);
  . . .
  q(d);
  . . .
end p;

procedure q (var a: integer);
  . . .
L: r(a);
  . . .
end q;

procedure r (var b: integer);
  . . .
  q(b);
  . . .
s(b);
  . . .
end r;

procedure s (var c: integer);
  . . .
  q(c);
  . . .
s(c);
  . . .
end s
```

Graph for bound set problem:

```
    d
     /
    /  
   /    
  a ----
    \
    /  
   /    
  c --- b
```

Graph for modified problem:

```
    d
     /
    /  
   /    
  a ----
    \
    /  
   /    
  c --- b
```

Fig. 2. Program fragment and formal bound set graph.

where \( E = \{(a, b), (b, a), (b, c), (c, a), (c, b), (d, a)\} \) and \( F = E - \{(a, b)\} \). Using \( 0 = \{ \} \) and iterating \( W \), we find:

\[
W^1(0) = E = \{(a, b), (b, a), (b, c), (c, a), (c, b), (d, a)\},
\]

\[
W^2(0) = E \cup \{(a, a), (a, c), (b, b), (c, c), (d, b)\},
\]

\[
W^3(0) = E \cup \{(a, a), (a, c), (b, b), (c, c), (d, b), (d, c)\},
\]

\[
W^4(0) = W^2(0) = \{(a, a), (a, b), (a, c), (b, a), (b, b), (b, c), (c, a), (c, b),
(c, c), (d, a), (d, b), (d, c)\}
\]

\[
= L_A.
\]
Likewise iterating $U$ from $0$ we find:

$$0 = \{ \{ \}, \}$$

$$U^1(0) = F = \{(b, a), (b, c), (c, a), (c, b), (d, a)\},$$

$$U^2(0) = F \cup \{(b, b), (c, c)\},$$

$$U^3(0) = U^2(0)$$

$$= \{(b, a), (b, b), (b, c), (c, a), (c, b), (c, c), (d, a)\}$$

$$= L_B.$$

However, beginning iteration for $U$ at $L_A$, we find:

$$U^0(L_A) = L_A,$$

$$U^1(L_A) = \{(a, a), (a, b), (a, c), (b, a), (b, b), (b, c), (c, a), (c, b), (c, c), (d, a), (d, b), (d, c)\},$$

$$U^1(L_A) = L_A \neq L_B,$$

so that $U$ restarted at $L_A$ converges to the wrong limit. Also, beginning at $W^1(0) = E$, we have:

$$U^0(E) = E = \{(a, b), (b, a), (b, c), (c, a), (c, b), (d, a)\},$$

$$U^1(E) = \{(a, a), (a, c), (b, a), (b, b), (b, c), (c, a), (c, b), (c, c), (d, a)\},$$

$$U^2(E) = \{(a, a), (a, b), (b, a), (b, b), (b, c), (c, a), (c, b), (c, c), (d, a)\},$$

$$U^3(E) = U^1(E) \neq U^2(E),$$

so that iteration finds a cycle of length two, and does not converge to any limit.

Thus restarting iteration at the old fixed point or an old iterate for the formal bound set problem after deletion of one or more call sites sometimes fails, and the incremental algorithm proposed by Cooper and Kennedy [8] must be replaced or modified to handle the case of edge deletion. Burke, Carroll and Ryder have developed incremental elimination algorithms which can be applied to the formal bound set problem. Burke’s update algorithm is interval analysis based [4]. Ryder’s initial work was also based on interval analysis [15, 17], but recently Carroll and Ryder have designed an incremental algorithm using dominators which can handle any type of structural change, including edge deletions [6, 7, 16].

The four classical data flow problems all have equations similar in form to (2.1). The variables are data flow solutions on entry to, or exit from, each node of a flow graph of a program. To define the reaching definitions problem, let $S$ be the set of all variable definitions in the program, $Q = S \times S$ with set inclusion as the partial

\footnote{Note: even if we removed the initial bound pairs introduced by the deleted call site from $L_A$, we still cannot restart iteration and obtain $L_B$, because there are pairs still in $L_A$ which only resulted from those deleted initial pairs; such pairs will remain in the fixed point obtained by this restarted iteration.}
Conditions for incremental iterations

For each node \( n \) in the flow graph, except the source, for which \( R_{\text{source}} = \{ \} \), we have:

\[
R_n = \bigcup_{j \in \text{pred}(n)} \{ p_j \cap R_j \cup d_j \}
\]

where \( p_j \) is the set of definitions preserved through node \( j \) and \( d_j \) is the set of definitions generated at and available on exit from node \( j \). A solution to these equations is an \( m \)-tuple \( (R_1, R_2, \ldots, R_m) \) where \( m \) is the number of flow graph nodes; it can be obtained by iteration using the initial condition \( R_j = \{ \} \) for all \( j \).

Consider the flow graph shown in Fig. 3 where the definition of variable \( a \) at node 1 is represented as \( a \), the definition of variable \( b \) at node 5 is represented as \( b \) and all the variable definitions are shown.

For this graph, \( p_j = \{ a, b \} \) for \( j = \{ 2, 3, 4, 6 \}, p_1 = \{ b \} \) and \( p_5 = \{ a \} \). Using iteration from \( \emptyset \), we reach the least fixed point \( (\{ \}, ab, ab, ab, ab, ab) \). Suppose we remove the edge \((6, 2)\). Then \( (\{ \}, a, a, a, a, ab) \) is the new fixed point of the altered system of equations. However, if we try to calculate the fixed point of the altered system by restarting iteration on the new system of equations at the old fixed point, we converge to \( (\{ \}, ab, ab, ab, ab, ab) \), an incorrect value. This result is not dependent on the type of program change introduced. If there were a new definition of \( a \) at node 2, by updating the \( p_j \) to indicate whether they preserve this definition, and restarting the iteration at the old fixed point, we again would obtain the wrong fixed point for the new system.

Fig. 3. Reaching definitions counterexample.
Similar examples can be obtained for the other classical data flow problems. For union problems (i.e., reaching definitions, live uses of variables) a change introduced to the system resulting in the solution becoming “bigger” means restarting iteration always is valid. However, if the change results in the solution becoming “smaller” as in our reaching definitions counterexample, we cannot restart iteration. For intersection problems (i.e., available expressions, very busy expressions), if the solution is decreased in size then restarting iteration is always valid, whereas if the solution is increased then we cannot restart iteration. This duality between union and intersection problems is natural.

All of these examples of invalidly restarting iteration include a cycle in the flow graph preserving some data flow information that reached that cycle in the original problem but would never reach it in the altered problem. In some sense, the cycle “keeps” that information and erroneously introduces it into the altered problem. We are currently investigating further generalizations of these results.

4. Theoretical results

In this section we present theorems which prove the classification of functions given in Section 1 and explore the behavior of function iteration from initial points other than 0, demonstrated in the examples of Sections 2 and 3. We give conditions under which this iteration converges to the same limit as obtained starting from 0. From this we derive a sufficient condition for successful use of restarting iteration in incremental updating. We describe specific situations in which restarting iteration will work as well as two general problems for which it will fail.

**Theorem 4.1.** Let $Q$ be a bounded set and $F$ a monotone function. Consider the sequence $(F^{i}(0))$. Then exactly one of the following applies:

1. There is a least $i$ so that $F^{i+1}(0) = F^{i}(0)$. In this case, $F^{i}(0) = L$ is the least fixed point of $F$.
2. $(F^{i}(0))$ is infinite, has a least upper bound $L$ in $Q$, and $L$ is a fixed point. Further, $L$ is the least fixed point of $F$.
3. $(F^{i}(0))$ has no least upper bound in $Q$.
4. $(F^{i}(0))$ has a least upper bound which is not a fixed point of $F$, and $F$ is not continuous.

**Proof.** Consider the sequence $(F^{i}(0))$. If there is a $j$ with $F^{j+1}(0) = F^{j}(0) = L$, and $X$ is any fixed point of $F$, then $F^{j}(X) \geq F^{j}(0) = L$. Thus $L$ is the least fixed point of $F$. Thus case (1) holds.

---

With simple modifications for changes that introduce or delete definitions in the program (i.e., which change $Q$), it is easy to deduce if a change (e.g., add/delete a node or edge) increases or decreases the solution.
Conditions for incremental iterations

If no such \( j \) exists, then if there is no least upper bound for the sequence, case (3) holds.

Suppose there is a least upper bound \( L \) and it is a fixed point. Let \( X \) be any other fixed point. Again \( X = F^j(X) \geq F^j(0) \) for all \( j \), so \( X \) is an upper bound of the sequence. Thus \( L \leq X \), and case (2) holds.

Finally suppose that the upper bound \( L \) is not a fixed point. Then \( F(L) = F(\text{lub}\{F'(0)\}) \neq \text{lub}\{F'^{j+1}(0)\} = L \). Thus \( F \) is not continuous and case (4) holds.

Under the assumption that one of the first two cases holds, we give a criterion which guarantees when restarting iteration is "safe" and suggests situations in which it will prove to be invalid.

**Theorem 4.2.** Let \( F \) be a monotone function, and suppose that the sequence \( \langle F^j(0) \rangle \) converges to its least fixed point \( L \). Let \( X < L \). Then the sequence \( \langle F^j(X) \rangle \) also converges to \( L \). Further, the sequence \( \langle F^j(X) \rangle \) is at most as long as \( \langle F^j(0) \rangle \). Also, if \( L \) is an infinite limit and \( X \) is less than some iterate \( F'(0) \), then the sequence \( F'(X) \) is also infinite.

**Proof.** By induction, \( F^j(0) \leq F^j(X) \leq F^j(L) = L \).

Thus \( L \) is an upper bound for the sequence \( \langle F^j(X) \rangle \). But clearly it must be the least upper bound. Since \( L \) is a fixed point, the theorem is proved. \( \Box \)

**Theorem 4.3.** Suppose \( X = A(X) \) and \( Y = B(Y) \) are systems of equations defined on a bounded set \( Q \). Suppose further that \( A \) and \( B \) are monotone functions of \( n \)-tuples of \( Q \), and that \( B(Z) \geq A(Z) \) for all \( Z \in Q \). Let \( X^* \) be an iterate or the infinite limit of the sequence \( \langle A'(0) \rangle \). If the limit \( L_B \) of \( B \) exists, then the sequence \( \langle B^j(X^*) \rangle \) converges to \( L_B \), and the sequence \( \langle B^j(X^*) \rangle \) is at most as long as the sequence \( \langle B^j(0) \rangle \).\(^7\)

**Proof.** This follows from Theorem 4.2 if \( X^* \) is an iterate.

If \( X^* \) is an infinite limit, note that \( L_B \) is an upper bound of \( \langle A'(0) \rangle \), hence \( X^* \leq L_B \), and the theorem still applies. \( \Box \)

If the hypotheses of Theorem 4.3 do not hold then even if \( L_B \) exists but \( X^* \geq L_B \), essentially any behavior can occur. Namely,

(i) The sequence may converge to the correct fixed point.
(ii) The sequence may converge to some other fixed point.
(iii) The sequence may have a least upper bound which is not a fixed point.
(iv) The sequence may not converge, and may in fact oscillate.

Examples in previous sections illustrate several of these cases. Case (i) is illustrated by shortest path, case (ii) by reaching definitions, and case (iv) by the last bound set example. We can give some more specific results.

\(^7\) The conclusion of the theorem continues to hold if \( A \) falls into case 4 and \( X^* \) is the lub of the sequence of iterates for \( A \).
(1) Suppose $F$ is monotone and $X$ and $F(X)$ are always comparable. Suppose in addition that $F$ satisfies the ascending and descending chain conditions on chains of iterates [3]. Then restarting iteration at any point always leads to a fixed point. If $F$ has only one fixed point, as in the shortest path example, then iteration from any point $X$ after a change in parameters will always reach the correct fixed point.

(2) Suppose $F$ is continuous and $Q$ is bounded above as well as below. Then if $F$ has only one fixed point, it will be the limit when iterating from any point $X$.

(3) Suppose $F$ is a function $F: V \rightarrow 2^U$ defined for finite graphs $G = (V, E)$ with values subsets of a set $U$, given by a system of union-intersection equations

$$F(X) = \bigcup_{(X, Y) \in E} \left[ a(X, Y) \cap F(Y) \right] \cup b(X, Y)$$

where $\Omega$ is union or intersection and $a, b: V \rightarrow 2^U$ are fixed functions; that is, $F(X)$ is a "linear" function of the $F(Y)$ at the neighbors of $X$, as in the four classical problems of data flow analysis. For any such problem there is a graph, an $X$, and a "small" change in the problem parameters (i.e., any of $a, b$ or $G$) on which restarting iteration at $X$ will lead to an incorrect solution.

(4) Similarly, if $F$ is a function given by a system of union-composition equations

$$F(X) = \bigcup_{(X, Y) \in E} \left[ a(X, Y) \circ F(Y) \right] \cup b(X, Y),$$

that is, a "transitive closure-like" function, there is again a graph, an $X$ and a "small" change which will give an incorrect solution upon continuing iteration from $X$.

5. Infinite limits and incremental iteration

As a partial converse to Theorem 4.3, we show that restarting iteration can give a misleading result only if $X^*$ is an infinite limit and $B$ is not continuous, that is, $L_B$ does not exist.

**Theorem 5.1.** Suppose $X = A(X)$ and $Y = B(Y)$ are systems of equations defined on a bounded set $Q$. Suppose further that $A$ and $B$ are monotone functions of $n$-tuples of $Q$, and that $B(Z) \geq A(Z)$ for all $Z \in Q$. Let $X^*$ be an iterate or the infinite limit (or lub) of the sequence $\langle A'(0) \rangle$, and suppose the sequence $\langle B'(X^*) \rangle$ converges to $L$. Then either $L$ is the least fixed point of $B$ and the limit of the sequence $B'(0)$, or $X^*$ is the infinite limit (or lub) of $A$ and the least upper bound for the sequence $\langle B'(0) \rangle$, and $B$ is not continuous.

**Proof.** If $X^* = A'(0)$ for some $j$, then $\langle B'(X^*) \rangle$ and $\langle B'(0) \rangle$ show the same behavior.

If $X^*$ is the infinite limit for $A$, then if $X^*$ is not an upper bound for $\langle B'(0) \rangle$, then the two sequences of iterates again show the same behavior. If $X^*$ is an upper bound for $\langle B'(0) \rangle$, then $X^*$ is the least upper bound for $B$. But if $L_B$ exists, this implies that $X^* = L_B$, and $X^*$ is a fixed point.
Thus if $L_B$ does not exist then $L$ is not the limit for $B'(0)$ and the least upper bound $X^*$ of the sequence $B'(0)$ cannot be a fixed point. Thus we are in case (4), and $B$ is not continuous. □

The following examples illustrate iteration and incremental iteration with infinite limits. Example 5.1 is an example of case (2) and Example 5.2 of case (4). Example 5.3 shows the three possibilities occurring when $X^*$ is an infinite limit:

(i) $X^*$ is not an upper bound for $\langle B'(0) \rangle$,
(ii) $X^*$ is the infinite limit for $\langle B'(0) \rangle$, or
(iii) $B$ is a case (4) function and no limit exists.

Example 5.1. Let $Q$ be the set of non-empty closed subintervals of the real interval $[0, 2]$ ordered by $[a, b] \geq [c, d]$ if $a \geq c$ and $b \geq d$. Then $0 = [0, 0]$. Let $W$ be defined as follows:

$$W[a, b] = \begin{cases} [a, (b+2)/3], & b \leq 1, \\ [a, (b+2)/2], & b > 1. \end{cases}$$

Then $[0, 1]$ is the infinite limit for the sequence $\langle W^i(0) \rangle$. (Suppose $[a, b]$ is an upper bound. Clearly $a \geq 0$ and $b \geq 1$ since $W^k([0, 0]) = [0, 1-(1/3)^k].$)

This should be contrasted to the following seemingly similar example, which is an example of a non-continuous $W$ (case (4)):

Example 5.2. Let $Q$ be as in Example 5.1 and

$$U[a, b] = \begin{cases} [a, (b+2)/3], & b < 1, \\ [a, (b+2)/2], & b \geq 1. \end{cases}$$

$[0, 1]$ is again the least upper bound for $U$, but is not a fixed point. In fact, only intervals of the form $[a, 2]$ are fixed by $U$, and $[0, 2]$ is the least fixed point. But $[0, 2] \succ [0, 1]$, so $[0, 2]$ is not the infinite limit.

Example 5.3. Let $Q$ be as in Example 5.1. In the terminology of Theorem 5.1, let $A$ be the function

$$A[a, b] = (\min(a, 1), (b+1)/2).$$

$X^* = L_A = [0, 1]$ is the infinite limit. Then:

(i) For $B[a, b] = [a, (b+2)/2]$, iteration from $[0, 1]$ leads to $[0, 2]$, the correct solution, as an infinite limit.
(ii) For $B$ the function in Example 5.1, $L_A = L_B = [0, 1]$, so iteration from $[0, 1]$ again behaves correctly.
(iii) For $B$ the function of Example 5.2, $L_A$ is an upper bound for $\langle B'(0) \rangle$, but not a fixed point. Thus even though $\langle B'(L_A) \rangle$ converges to a fixed point, namely $[0, 2]$, this is not the limit of the sequence $\langle B'(0) \rangle$, which has no limit (i.e., $B$ is not continuous).
6. Conclusions

Our results on the correctness of restarting iteration as an incremental update technique for solving systems of equations have practical application in static program analysis, graph algorithms, and set equations. For large problems whose parameters change over time, incremental update algorithms are attractive and may be the only feasible solution procedure. We have presented two theoretical results. The first proves our four case classification of the fixed points of monotonic functions (i.e., Theorem 4.1). The second gives sufficient conditions guaranteeing the convergence of restarting iteration of a perturbed monotonic function at its old fixed point (or an old iterate) (i.e., Theorems 4.2 and 4.3). Our examples illustrated these results and indicate problems in incremental iterative algorithms in the literature.

Acknowledgments

We wish to thank the editor for his helpful and constructive suggestions which have greatly improved our presentation.

References