Computer Science Department Technical Report
University of California
Los Angeles, CA 90024-1596

COMPUTING LEAST FIXED POINTS BY ASYNCHRONOUS ITERATIONS AND RANDOM ITERATIONS

X. Wang
D. S. Parker

June 1992
CSD-920030

Computing Least Fixed Points by Asynchronous Iterations and Random Iterations*

Xin Wang D. Stott Parker
POP Laboratory
Computer Science Department
University of California, Los Angeles

Abstract

Asynchronous iterations and random iterations are two commonly-used iteration (or relaxation) schemas in multiprocess systems, which correspond to message-passing and shared-memory models, respectively. Asynchronous iterations (AI's) are iteration processes of a map on a product space that essentially allow, at each iteration step, application of an arbitrary subset of component maps of the map to a vector of any previous values of components, while random iterations (RI's), on the other hand, are iteration processes of a family of maps on a common space in which one and only one map out of the family is selected and applied successively at each iteration. These iterations have been extensively studied in the context of contracting and more general nonexpansive maps on metric and Banach spaces for establishing conditions that guarantee their convergence and uniqueness of convergence. In this report, we study these two kinds of iterations for maps on partially ordered sets. We show that asynchronous iterations can be used to compute least fixed points of continuous maps on some complete partially ordered product sets (e.g., product cpo's and lexicographic cpo's), and that any (α-) confluent or semi-confluent family of continuous maps on a common cpo have a non-empty set of common fixed points and random iterations can be used to compute the least common fixed point of the family.

*This research is supported by NSF grant IRI-8917907

1 Introduction

This report is an extension of [Wan89]. Here we are interested in two problems of computing least fixed points of continuous maps on complete partially ordered sets (cpo's): (i) given a cpo $(X, \sqsubseteq)$ where $X = \Pi_{i \in I} X_i$ is a product set and a continuous map $F: X \to X$, compute the least fixed point $\mu F \in X$ of F; and (ii) given a family of continuous maps $f_i: X \to X$, $i \in I$, on a common cpo X, compute the least common fixed point $\mu\{f_i \mid i \in I\}$ of the f_i's when it exists.

In an abstract sense, these two problems correspond to finding a least solution to a system $x = f(x)$ in message-passing and shared-memory parallel and distributed processing systems. Examples of these kinds of fixed-point computation include constraint satisfaction systems, program data-flow analysis, graph algorithms [BT89], rewriting systems [DJ90], and many others. Frequently used are iterative methods of solving such problems; that is, given an appropriate initial value for x, an iterative algorithm repeats the assignment $x \leftarrow f(x)$ until a fixed point is found. In fact, it is known that problem (i) can be solved as a system of equations by two classic methods: iteration and elimination. For example, Tarski showed in his fundamental fixed-point theorem [Tar55] that the least fixed point μF can be computed by iterative application of F to the least element $\perp$ of X and μF is equal to the supremum $\sqcup\{F^t(\perp)\}$ of the iterative sequence $\{F^t(\perp) \mid t = 0, 1, 2, ...\}$, where F^t is F iterated t times, and Bekič provided a "bisection lemma" in [Bek84] (also see [Mos89a]) which allows use of the elimination method for simultaneous equations to compute μF. As for problem (ii), a few results exist for the cases where family F of maps are commuting, $f_i \cdot f_j = f_j \cdot f_i$. For example, for a finite, commuting family of continuous maps, the least common fixed point exists [DeM64] and can be computed by iteratively applying any composite map $g = f_{i_1} f_{i_2} \cdots f_{i_n}$ ($i_j \neq i_k$ for $j \neq k$) to the bottom element $\perp$ [CC79].

In this report, we are interested in using asynchronous iterations (AI's) as given in [Bau78, BT89] and random iterations (RI's) as given in [Bru82, Tsi87] to compute least fixed points in problem (i) and (ii), respectively. Roughly speaking, an asynchronous iteration is a discrete iteration in which at each iterating step any subset of components of a point is updated by corresponding component maps and other components remain unchanged, while a random iteration is an iterative process in which one and only one map that is randomly chosen from a family of maps is applied at each iterating step.

The motivation for considering AI's and RI's is to account for parallel and distributed implementations of iterative methods (like Tarski's iteration) on a multiprocess system such as a message-passing or shared-memory system [BT89], in such a way as to reduce possible communication and synchronization overhead between cooperating processes. For asynchronous iterations, "this reduction is obtained not by forcing the processes (component maps) to follow a predetermined sequence of computation, but simply by allowing a process, when starting the evaluation of a new iterate, to choose dynamically not only the components to be evaluated but also the values of the previous iterates used in the evaluation" [Bau78], while for random iterations this reduction is achieved by choosing an arbitrary process (map) from a given family of processes and applying it to the result of previous applications at each iterative step.

These methods have been successfully applied in different situations. For instance, AI's have been used to solve linear equations with coefficient matrices of spectral radii less than 1 [CM69] and other numerical problems [BT89], and to compute the fixed point of general contracting maps on

Banach spaces [Bau78] or some variant spaces [BT89], while RI's have been taken as an effective mean of computing common fixed points of, most notably, a family of contraction or more general nonexpensive maps on metric and normed spaces [Bru82], and a family of maps on topological spaces for which Liapunov functions can be defined [Tsi87].

An instructive parallel drawn between the two very fundamental fixed-point theorems, Banach and Tarski theorems, also promotes the raising of problems (i) and (ii) in the present study. The Banach theorem asserts that a contracting map on a Banach space (complete normed space) has a unique fixed point, which can also be explicitly computed by an iterative process analogous to that in Tarski's theorem. In Banach's Theorem, the approximating sequence converges because the normed space is complete and its limit is a fixed point thanks to the continuity of the contracting map. In Tarski's theorem, the order-theoretic notions of completeness and continuity replace the topological ones in Banach's theorem. In fact, given a contracting map on a Banach space X, we may define an *ad hoc* ordering on X so that Tarski's theorem is applicable. Denote by x^* the unique fixed point of F on X. Then the ordering is given by the reflexive-transitive closure of the relation

$$x \sqsubset y \quad \text{if and only if} \quad \|x - x^*\| > \|y - x^*\|.$$

It is straightforward to check that, for any $x_0 \in X$, the upper set, $x_0\uparrow = \{x \mid x_0 \sqsubseteq x\}$, of x_0 is a cpo with bottom $\perp = x_0$ and F restricted to $x_0\uparrow$ is a continuous map with respect to the order $\sqsubseteq$. Now according to Tarski's theorem, F has a least fixed point, which turns out to be the unique fixed point x^* of F as given in Banach's theorem. In fact, very recently, it has been shown in [Bar91] that Banach's theorem is indeed a particular case of Tarski's theorem, by embedding a metric space (or similarly a norm space) into a partially ordered space, which is not dependent on any contractive functions on the metric space, so that any contractive function is a continuous map on the poset and it has the unique (least) fixed point. However, if in turn X is a cpo and F is an order-theoretically continuous map on X, it is usually hard to construct a norm $\|\cdot\|$ on X such that X equipped with $\|\cdot\|$ becomes a Banach space and F is a contracting map. Some examples of how to imposing quasi-metrics (differing from metrics in not requiring that $d(x, y) = 0$ implies $x = y$) and generalized metrics (not requiring symmetry of metrics and additive commutativity of metric values) on cpo's are given in [Smy91] and [JMP86], respectively. Even in these cases either the quasi-metrics or generalized metrics depend on specific cpo's, or they are suitable for general cpo's but fail to introduce the Scott topology (which will be defined later in Section 2) or topologies that are consistent with partial orders on cpo's [GHK+80] and hence have no guarantee on an order-continuous map being metric-continuous. In this sense, Tarski's theorem is more fundamental than Banach's theorem. Therefore, not only do problem (i) and (ii) complete the picture of using AI's and RI's to compute fixed points provided in both the theorems, but they have a more profound significance than their counterparts in metric or norm spaces.

A key property that asynchronous iterations and random iterations can possess is a kind of canonicity, i.e., that, starting with a point, every iteration converges to a unique fixed point. Very similar to rewriting systems, the canonicity is usually decomposed into two components: convergence, which, like termination in rewriting systems, ensures that every iteration approaches a fixed point, and confluence, which ensures that there can be at most one fixed point for iterations starting with a same initial point to converge to. A natural consequence of a canonical AI or RI system is that AI's or RI's compute the same fixed points as do simple, common iterations, so that they asynchronize or randomize the usual iterations of computing fixed points.

The purpose of this report is to provide some results on the canonicity of AI's and RI's of continuous

maps on complete partially ordered sets. Our results show that the least fixed point computed by a continuous map on a product order and alternative extension orders on the product of a family of complete partially ordered sets can be computed by arbitrary asynchronous iterations, and that the least common fixed point of an α-semi-confluent family of continuous maps on a common cpo can be computed by any fair α-random iterations, which generalizes a classic result on computing the least fixed point of a commuting family of continuous maps by a composite map.

The rest of the report is organized as follows. In Section 2 we recall some basic concepts and notations on partially ordered sets. We give the definition of AI's in Section 3. In Section 4 we prove that the least fixed point of an (ω-) continuous map over a product cpo or a lexicographic cpo, that is computed by the Tarski's (parallel) iterative approximation, can be computed by any AI defined in Section 3. We then define RI's formally and discuss their relation with term rewriting systems in Section 5, and show that RI's can be used to to compute the least common fixed point of any confluent family of continuous maps in problem (ii). We finally conclude in Section 6 with some possible implications of our results for the semantics of programming languages and concurrency.

2 Preliminaries

Here we give some notations and concepts that will be used in the rest of the report. For a general discussion of partially ordered sets (posets), we refer to [Bir69, DG82, DP90, NR85, Sco72].

A *partially ordered set* (poset) is a pair $(X, \sqsubseteq)$, where X is a non-empty set and $\sqsubseteq$ is a binary relation on X , called a partial order, that has reflexive, antisymmetric and transitive properties. Given two partial orders $\sqsubseteq_1$, $\sqsubseteq_2$ on X, $\sqsubseteq_2$ is an extension of $\sqsubseteq_1$ if, whenever $x \sqsubseteq_1 y$, $x \sqsubseteq_2 y$.

Let $(X, \sqsubseteq)$ be a poset and S be a nonempty subset of X. S is *directed* if for any $x, y \in S$, there exists some $z \in S$ such that $x \sqsubseteq z$ and $y \sqsubseteq z$. A special case of a directed subset is a *countable ascending chain*, which is a sequence $\{x(t) \in X \mid t = 0, 1, 2, ...\}$ with $x(0) \sqsubseteq x(1) \sqsubseteq \cdots \sqsubseteq x(t) \sqsubseteq \cdots$. An *upper* (*lower*) *bound* of S is some $z \in X$ with $z \sqsubseteq x (x \sqsubseteq z)$ for every $x \in S$. The *supremum* $\sqcup S$ (*infimum* $\sqcap S$) of S, if it exists, is the least (greatest) such z. The least point of the whole set X, if it exists, is denoted by $\bot$. A *pointed* poset is a poset with least point $\bot$. A *complete partially ordered set* (cpo) is a pointed poset $(X, \sqsubseteq)$ in which every *directed* subset S has supremum $\sqcup S$.

Let $(X, \sqsubseteq, \bot)$ be a cpo. A subset $U \subseteq X$ is *open* if (i) for any $x, y \in X$, $x \in U$ and $x \sqsubseteq y$ implies $y \in U$; (ii) for any directed subset $S \subseteq X$, $\sqcup S \in U$ implies $S \cap U \neq \emptyset$. The family of all such open subsets of X induces a topology on X called the *Scott topology* ([Sco72]). A sequence of points in X, $\{x(t) \mid t = 0, 1, 2, ...\}$, *converges to* $x \in X$, written as $\lim_{t\to\infty} x(t) = x$, if whenever U is open and $x \in U$, then there exists some $T \geq 0$ such that $x(t) \in U$ for all $t \geq T$. $\{x(t)\}$ *converges finitely* if there exists some T such that for any $t' > T$, $x(t') = \lim_{t\to\infty} x(t)$. Notice that in the Scott topology any directed sequence S converges to its supremum $\sqcup S$ and, typically, any infinite ascending chain $\{x(t)\}$ converges to $\sqcup\{x(t)\}$.

A map $F: X \to X$ is *monotone* if for every $x, y \in X$, $x \sqsubseteq y$ implies $F(x) \sqsubseteq F(y)$. F is *ω-continuous* if for any countable ascending chain $x(0) \sqsubseteq x(1) \sqsubseteq x(2) \sqsubseteq \cdots$ in X, $F(\sqcup\{x(t)\}) = \sqcup\{F(x(t))\}$. F is *continuous* if for any directed set $S \subseteq X$, $F(\sqcup S) = \sqcup\{F(x) \mid x \in S\}$. Note that every continuous

map is ω-continuous, and every ω-continuous map must also be monotone, since $x \sqsubseteq y$ implies $F(x) = F(\sqcup\{x,y\}) = \sqcup F(\{x,y\}) = F(x) \sqcup F(y) \sqsubseteq F(y)$. It is known that this definition of continuity of a map coincides with the continuity of a map defined in terms of the Scott topology in the usual topological sense.

Let I be a non-empty set and $X_i (i \in I)$ be a family of non-empty sets. Denote by $X = \prod_{i\in I} X_i$ the product set of X_i, consisting of elements of form $x = [x_i]_{i\in I}$ with $x_i \in X_i$. When $(X_i, \sqsubseteq_i) (i \in I)$ are a family of cpo's with with bottom elements $\perp_i$'s, denote by $(X, \sqsubseteq)$ the product poset of $(X_i, \sqsubseteq_i)$, namely, $X = \prod_{i\in I} X_i$ and, for any $x = [x_i]_{i\in I}, y = [y_i]_{i\in I} \in X$, $x \sqsubseteq y$ if and only if $x_i \sqsubseteq_i y_i$ for all $i \in I$. It is easy to check that product poset $(X, \sqsubseteq)$ is a cpo with bottom element $\perp = [\perp_i]_{i\in I}$. However, the Scott topology on the product cpo X is in general not a product topology of the Scott topologies on cpo's X_i. But, for any sequence $\{x(t)\}$ in X, $\lim_{t\to\infty} x(t) = x$ if and only if $\lim_{t\to\infty} x_i(t) = x_i$ for all $i \in I$. Also, if X, $X_i (i \in I)$ are cpo's and $F: \Pi_{i\in I} X_i \to X$, then F is monotone (ω-continuous, continuous) if and only if f_i is monotone (ω-continuous, continuous) for every $i \in I$ ([Sco72]).

There is another way to impose an order on a product set $X = \prod_{i\in I} X_i$ of posets $(X_i, \sqsubseteq_i)$. Assume that I is linearly ordered (i.e., for any i, j in I, one and only one of cases $i < j$, $i = j$ and $j < i$ occurs). The *lexicographic order* on X (with respect to the order on I) is the reflexive closure of the relation $\sqsubset$ defined by $[x_i] \sqsubset [y_i]$ if and only if there exists some j in I such that $x_j \sqsubset y_j$ and, for all $k < j$, $x_k =_i y_k$. Clearly, the lexicographic order is an extension of the product order.

Given a family of non-empty sets $X_i, i \in I$ and a family of maps f_i from product set $X = \Pi_{i\in I} X_i$ into X_i, we denote, for any $I_0 \subset I$, a map $F_{I_0}: X \to X$ as follows. For any $x = [x_i]_{i\in I}, y = [y_i]_{i\in I} \in X$, $y = F_{I_0}(x)$ if and only if, for all $i \in I$,

$$y_i = \begin{cases} f_i(x) & \text{if } i \in I_0 \\ x_i & \text{if } i \notin I_0 \end{cases}$$

That is, $y = F_{I_0}(x)$ results from changing the i-th components of x to $f_i(x)$ for all $i \in I_0$. In the cases that $I_0 = I$ and $I_0 = \{i\}$, F_{I_0} is also written as F and $F_i (= f_i)$, respectively. Notice that F is the product of the maps f_i, that is, $F = [f_i]_{i\in I}$.

3 Asynchronous Iterations

Definition 1 Given a non-empty (index) set I, an *asynchronous iteration scheme* is any pair $(\mathcal{I}, \mathcal{S})$, where

- $\mathcal{I} = \{I(t) \subseteq I \mid t = 1, 2, ...\}$ is a sequence of nonempty subsets of I,
- $\mathcal{S} = \{ \{[s_j^i(t)]_{j\in I} \mid i \in I(t)\} \mid t = 1, 2, ...\}$ is a sequence of sets of functions of I into N (the set of natural numbers); that is, for every $t \geq 1$ and every $i \in I(t)$, $s_j^i(t) \in N$ for each $j \in I$,

satisfying conditions:

(a) $\forall i \in I$, i occurs infinitely often in the sets $I(t), t = 1, 2, ...$; i.e., $\{t \mid i \in I(t)\}$ is infinite for all $i \in I$;

(b) $\forall t \geq 1$ and $\forall i \in I(t)$, $s_j^i(t) \sqsubseteq t - 1$, for each $j \in I$;

(c) $\lim_{t \to \infty} \min\{s_j^i(t) \mid j \in I, i \in I(t)\} = \infty$.

∎

Definition 2 Let $X = \prod_{i \in I} X_i$ be the product of X_i's, $f_i: X \to X_i$ a family of maps, $x \in X$ a point, and $(\mathcal{I}, \mathcal{S})$ an asynchronous iteration scheme. An *asynchronous iteration* or *trajectory* of F starting at x in $(\mathcal{I}, \mathcal{S})$ is a sequence $\{x(t) \mid t = 0, 1, 2, \ldots\}$ of points of X defined recursively by $x(0) = x$, and for $t > 0$, $i \in I$,

$$x_i(t) = \begin{cases} f_i([x_j(s_j^i(t))]_{j \in I}) & \text{if } i \in I(t) \\ x_i(t-1) & \text{if } i \notin I(t) \end{cases}$$

which is also denoted in a compact form by $x(t) = F_{I(t)}(x(s(t)))$. ∎

The definitions given above can be interpreted together as follows: At each time instant $t \geq 1$, $I(t)$ indicates those components $i \in I$ of point $x(t-1)$ that need to be updated, with the remaining components left unchanged. The updating of $x(t-1)$ into $x(t)$ is achieved by applying f_i $(i \in I(t))$ to the values $x_j(s_j^i(t))$ of all components known at previous times, namely the values of all components x_j at times $s_j^i(t) < t$. Thus, the asynchronization is reflected by the freedom to update any subset of components at each time and the freedom to use any previous values of components. Conditions (a), (b) and (c) on asynchronous iteration schemes are fairly reasonable restrictions to rule out some exceptional cases: condition (a) guarantees that no component will be abandoned forever without being updated, condition (b) states the fact that only previous values of components can be used in the current updating, and condition (c) requires that, eventually, the values at early times will not be used any further in updating, and more recent values of the components will be used instead.

In terms of concurrent programming, asynchronous iterations characterize behavior of a multiprocess, message-passing distributed system. Consider the index set I as a collection of (identifications of) processes in the system. If the maps f_i are thought of as functions implemented on individual processors, then $\mathcal{I}$ can be regarded as activation sequences of the processors and $\mathcal{S}$ as communicating sequences between the processors. For instance, first-in-first-out (FIFO) communicating channels can be characterized as: if $t_1 < t_2$ and $i \in I(t_1) \cap I(t_2)$, then $s_j^i(t_1) \sqsubseteq s_j^i(t_2)$ for all $j \in I$; *i.e.*, only the most recently available values of components are used in the further updating though they could be delayed. Now, conditions (a) and (c) become liveness properties of the processors and communication channels between the processes, respectively.

Classical iteration methods in numerical computation [Blu72] like point Jacobi, block Jacobi, Gauss-Seidel methods, as well as others introduced more recently such as chaotic relaxation [CM69], random sequential iteration [Hop82], execution sequences [Tsi87], to name a few, can all be regarded as special asynchronous iterations with different schemes $(\mathcal{I}, \mathcal{S})$. We list some of them as follows. For the details of these special cases, see [Bau78] or the related references.

(a) **Parallel iteration.** This is the most common iteration, where $I(t) = I$ and $s_j^i(t) = t - 1$, for all $i, j \in I$ and $t = 1, 2, ...$;

(b) **Sequential (point Jacobi) iteration.** This applies only to cases when I is finite and is based on the strategy of using the most recently updated components: $I(t) = \{1 + (t-1) \pmod n\}$ and $s_j^i(t) = n\lfloor (t-1)/n \rfloor, j \in I(t), i \in I(t), t = 1, 2, \ldots$.

Note that there are $n!$ different sequential iterations, corresponding to the $n!$ possible permutations of $\{1, ..., n\}$, which change components only one at a time in the order prescribed by the permutations. All sequential iterations have the same fixed points as those of the parallel iteration [Rob86] – we will generalize this result later to any asynchronous iteration. But they may introduce different limit cycles (explain);

(c) **Block sequential (block Jacobi) iteration.** The *block sequential iteration* associated to the ordered partition $\{I_k\}$ $(k = 1, 2, ..., n)$ of set I is defined in [GCP85] by

$$\forall k \in \{1, ..., n\}, \forall i \in I_k, \; x_i(t+1) = f_i(y^k(t))$$

where

$$y^1(t) = x(t)$$
$$y_j^k(t) = \begin{cases} x_j(t+1) & \text{if } j \in I_1 \cup ... \cup I_{k-1} \\ x_j(t) & \text{otherwise.} \end{cases} \quad \text{for } k \in \{2, ..., n\}.$$

This corresponds to our definition as $I(t) = I_k$ when $k = 1 + ((t-1) \bmod n)$ and $s_j^i(t) = n\lfloor (t-1)/n \rfloor$ for all $j \in I$ and $i \in I(t)$, $t = 1, 2, ...$;

(d) **Random sequential iteration.** Each time only one component is chosen randomly with equal probability, $I(t) = \{i\}$ and $s_j^i(t) = t - 1$;

(e) **Chaotic relaxation.** This kind of iteration is very similar to asynchronous iteration, except it requires that $t - s_j^i(t)$ be uniformly bounded by some fixed positive integer s for all $i \in I(t), j \in I$ and $t = 1, 2, ...$;

(f) **Execution sequence.** This is the case where all $s_j^i(t) = t - 1$. Then a trajectory is expressed by

$$x(t) = F_{I(t)}(x(t-1)).$$

If each $I(t)$ is represented by its set-theoretically characteristic function $\sigma(t) = [\sigma_j(t)]_{j \in I}$, where $\sigma_j(t) = 1$ or 0 depends on whether $i \in I$ or not, then the trajectory can also be defined in a closed vector form

$$x(t) = \sigma(t) \cdot F(x(t-1)) + (1 - \sigma(t)) \cdot x(t-1)$$

where the dot product is interpreted in an obvious way.

The definition of asynchronous iterations given above is similar to the one given by Bertsekas and Tsitsiklis in [BT89] together with their assumption 1.1 (like conditions (a) and (c) above), which is a generalization of Baudet's in [Bau78] (which requires that $s_j^i(t) = s_j(t)$ for all $i \in I(t)$) and the original *chaotic iterations* by Chazan and Miranker in [CM69]. However, unlike their definitions, we make no assumption that index set I have to be finite. Therefore, our definition makes asynchronous iterations applicable even in the cases like $I = \mathbf{N}^k$ (k-dimensional discrete grid of processors) and

$I = \mathbf{R}^k$ (k-dimensional continuous grid of processors) with $k \geq 1$, where each $I(t)$ ($t = 1, 2, \cdots$) could be taken, for example, as $\{0, 1, 2, ..., t\}^k$ and $[-t, t]^k$, respectively.

Definition 3 A point $x \in X$ is said to be a *fixed point* of an asynchronous iteration of F in $(\mathcal{I}, \mathcal{S})$, if, starting at x, the trajectory $x(t) = x$ for every $t \geq 0$. ∎

In [Par87, Rob86], it has been shown that any fixed point of a *sequential* iteration of F is a fixed point of the parallel iteration of F and vice versa. The following theorem generalizes this to any asynchronous iteration.

Theorem 4 *For any $x \in X$, x is a fixed point of an asynchronous iteration of F in $(\mathcal{I}, \mathcal{S})$ if and only if x is a fixed point of F.*

Proof: The if part can be easily shown by induction on t. The only-if part proceeds as follows: if $x \neq F(x)$, then for some $i \in I$, $x_i \neq f_i(x)$, which leads to $x_i(t) \neq f_i(x(s(t_0))) = f_i(x)$, $i \in I(t_0)$ for some t_0. This contradicts $x(t_0) = x$. ∎

Two major issues regarding asynchronous iterations are their *convergence* and *determinacy*. Convergence means that any asynchronous trajectory converges (in a certain sense) to some fixed point of map F, and determinacy requires further that, starting at a given point, asynchronous iterations in different asynchronous iteration schemes, when converge, will converge to a unique point. Clearly, convergence and determinacy together ensure that starting a point, a unique fixed point will be reached by any asynchronous iterations. In [CM69], a necessary and sufficient condition for the convergence and determinacy of asynchronous iterations was obtained for linear operators on the n-dimensional real space $\mathbf{R}^n$, provided the spectral radii of the operators are less than 1. Later in [Bau78], a sufficient condition was given for contracting maps on $\mathbf{R}^n$ (or in general any Banach space). A quite general result for finite product sets was provided in [BT89] which assumes only a notion of convergence on structures of the product sets but needs the so-called "Synchronous Convergence Condition" and "Box Condition" that are equivalent to identifying Liapunov functions in the stability analysis of nonlinear dynamical systems [LaS86]. In the next section we will establish some results on the convergence and determinacy of asynchronous iterations for continuous maps on complete partially ordered sets.

4 Computing Least Fixed Points

Through this section, we will fix X_i, $i \in I$, as a family of cpo's, $X = \Pi_{i \in I} X_i$ as a product set (which may be partially ordered in many different ways like the product and lexicographic orders), F as a map from X into itself, and $(\mathcal{I}, \mathcal{S})$ as an asynchronous iteration scheme. Clearly, any result on convergence and determinacy of asynchronous iterations depends on an order imposed on product set X. We first give the result for the product order, and then generalize it to other partial orders with the lexicographic order in particular that are complete extensions of the product order. Finally, we provide a more practical formulation of asynchronous iterations and give a result on convergence and determinacy for this new formulation.

We first consider X as the product cpo of X_i's. The following two lemmas establish a comparison between "growth rates" of an asynchronous iteration and the parallel iteration when map F is monotone.

Lemma 5 *Suppose that F is monotone and $x \in X$ is such that $x \sqsubseteq F(x)$. Then trajectory $\{x(t)\}$ of F starting at x in $(\mathcal{I}, \mathcal{S})$ satisfies*

$$x(t) \sqsubseteq F^t(x),\ t = 0, 1, 2, \ldots.$$

Proof: First notice that the assumption $x \sqsubseteq F(x)$ implies that $x = F^0(x) \sqsubseteq F(x) \sqsubseteq F^2(x) \sqsubseteq \cdots$ is a countable ascending chain.

The proof that $x(t) \sqsubseteq F^t(x)$ proceeds by induction on t. When $t = 0$, $x(0) = x = F^0(x)$. Suppose that $x(t') \sqsubseteq F^{t'}(x)$ for all $0 = t' < t$. Now consider the case for t. For $i \notin I(t)$, $x_i(t) = x_i(t-1) \sqsubseteq_i (F^{t-1}(x))_i \sqsubseteq_i (F^t(x))_i$. For $i \in I(t)$, $s_j^i(t) \sqsubseteq t-1$ for all $j \in I$ (by condition (b) on $\mathcal{S}$), and $x_j(s_j^i(t)) \sqsubseteq_j (F^{t-1}(x))_j$, which implies that $[x_j(s_j^i(t))]_{j \in I} \sqsubseteq F^{t-1}(x)$ and $x_i(t) = f_i([x_j(s_j^i(t))]_{j \in I}) \sqsubseteq f_i(F^{t-1}(x)) = (F^t(x))_i$. Hence $x(t) \sqsubseteq F^t(x)$ for all $t = 1, 2, \ldots.$ ∎

Lemma 6 *Suppose that F is monotone and $x \in X$ is such that $x \sqsubseteq F(x)$. Then, for trajectory $\{x(t)\}$ of F starting at x in $(\mathcal{I}, \mathcal{S})$, there exists a sequence of integers, $t_0 < t_1 < t_2 < \cdots$, such that, for $p = 0, 1, 2, \ldots$,*

$$(*) \qquad F^p(x) \sqsubseteq x(t),\ \textit{for all}\ t \geq t_p.$$

Proof: Again notice that $x \sqsubseteq F(x)$ implies that $x = F^0(x) \sqsubseteq F(x) \sqsubseteq F^2(x) \sqsubseteq \cdots$ is a countable ascending chain. We proceed with the proof by induction on p.

Base case: $p = 0$. Take $t_0 = 0$. An induction on $t \geq 0$ shows that $(*)$ holds for $t_0 = 0$: when $t = 0$, $F^0(x) = x = x(0)$, and when $t > 0$, by the condition (b) on $\mathcal{S}$, $s_j^i(t) \sqsubseteq t-1, j \in I$ if $i \in I(t)$, and, by the induction hypothesis, $x \sqsubseteq x(s(t))$. Thus,

$$x \sqsubseteq F_{I(t)}(x) \sqsubseteq F_{I(t)}(x(s(t))) = x(t), \quad \text{for all}\ \ t \geq t_0 = 0,$$

which completes the base case.

Induction Hypothesis: For $p > 0$, there are integers $t_0 < t_1 < \cdots < t_{p-1}$ such that $(*)$ holds for all q, $0 \sqsubseteq q < p$. Assume that t_q has been found and $(*)$ holds that for all $q = 0, 1, \ldots, p-1$.

Induction Step: First, define r_p by

$$r_p = \min\{k \mid \forall t \geq k, s_j^i(t) \geq t_{p-1}, \text{for}\ \ i \in I(t), j \in I\}.$$

Such r_p is well-defined due to condition (c) on $\mathcal{S}$. Further, $r_p > t_{p-1}$, by condition (b) on $\mathcal{S}$. Hence,

$$F^{p-1}(x) \sqsubseteq x(r_p).$$

Take any $t \geq r_p$ and consider an arbitrary i-th component $x_i(t)$ of $x(t)$. If $i \in I(t)$, let $z^i = [x_j(s_j^i(t))]_{j \in I}$. By the choice of r_p, $s_j^i(t) \geq t_{p-1}$ for all $j \in I$ and $F^{p-1}(x) \sqsubseteq z^i$ by the induction hypothesis. This shows that

$$(F^p(x))_i = f_i(F^{p-1}(x)) \sqsubseteq f_i(z^i) = x_i(t).$$

On the other hand, if $i \notin I(t)$, the ith component does not change, $x_i(t) = x_{i-1}(t)$. Therefore, as long as the i-th component is updated between times r and t, $(F^p(x))_i \sqsubseteq x_i(t)$.

Now define t_p as

$$t_p = \min\{t \mid t \geq r, \text{and } I(r) \cup \cdots \cup I(t) = I\}.$$

Such t_p is well-defined due to condition (a) on $\mathcal{I}$. Then, for any $t \geq t_p$, every component of $x(t)$ is updated at least once between times r and t, and therefore $F^p(x) \sqsubseteq x(t)$. This shows that $(*)$ holds for p, which completes the induction and the proof of the lemma. ∎

Here is the main result of this section.

Theorem 7 *Suppose that $X = \Pi_{i \in I} X_i$ is the product cpo of cpo's $X_i (i \in I)$ and $F: X \to X$ is continuous. Then trajectory $\{x(t)\}$ of F starting at $\perp$ in any asynchronous iteration scheme $(\mathcal{I}, \mathcal{S})$ is directed and converges to the least fixed point μF of F.*

Proof: According to Tarski's theorem, F has the least fixed point μF in X and $\mu F = \sqcup\{F^t(\perp) \mid t = 0, 1, 2, ...\}$. Since the continuity of F implies its monotonicity, the conclusions of Lemmas 5 and 6 hold for $x = \perp$. For any $x(t_1), x(t_2) \in \{x(t)\}$, let $p = \max\{t_1, t_2\}$. Then, from lemma 6, there is some t_p such that

$$x(t_1) \sqsubseteq F^{t_1}(\perp) \sqsubseteq F^p(\perp) \sqsubseteq x(t_p), \text{ and } x(t_2) \sqsubseteq F^{t_2}(\perp) \sqsubseteq F^p(\perp) \sqsubseteq x(t_p),$$

which shows that $\{x(t)\}$ is directed. As X is a cpo, $\{x(t)\}$ has supremum $\sqcup\{x(t)\}$ in X. By Lemma 5,

$$\sqcup\{x(t)\} \sqsubseteq \sqcup\{F^t(\perp)\} = \mu F;$$

and, again by Lemma 6,

$$\mu F \sqsubseteq \sqcup\{x(t)\}.$$

Hence $\lim_{t \to \infty}\{x(t)\} = \sqcup\{x(t)\} = \mu F$. ∎

Corollary 8 *Under the same condition of Theorem 7, $\{x(t)\}$ converges finitely in any asynchronous iteration scheme if and only if $\{F^t(\perp)\}$ converges finitely.*

We now consider two generalizations of Theorem 7. The first one concerns different complete partial orders on X that extend the product order. In some cases (see [Bos85] for many examples), the product order may not serve a right purpose. Instead, other complete partial orders like lexicographic orders need to be introduced. The next theorem states that, as long as these orders are extensions of the product order, the conclusion of Theorem 7 remains true.

Theorem 9 *Let $X = \Pi_{i\in I} X_i$ be the product set of sets $X_i (i \in I)$, $(X, \sqsubseteq)$ a cpo and $F: X \to X$ a continuous map with respect to $\sqsubseteq$. If the order $\sqsubseteq$ is an extension of the product order on X, then the trajectory $\{x(t)\}$ of F starting at $\perp$ in any asynchronous iteration scheme $(\mathcal{I}, \mathcal{S})$ is directed and converges to the least fixed point μF of F.*

Proof: Since F is continuous, $\mu F = \lim_{n\to\infty} F^n(\perp)$. According to Lemma 5, 6 and the assumption that $\sqsubseteq$ is an extension of the product order, the trajectory $\{x(t)\}$ is directed and converges to the limit of $F^n(\perp)$, that is, μF. ∎

Since any lexicographic order is an extension of the product order on a product of partially ordered sets, we have

Corollary 10 *If F is continuous with respect to a lexicographic order on X, then the conclusion of the theorem holds.*

The second generalization of Theorem 7 involves generalizing the definition of asynchronous iterations in their spatial dimension, in order to adapt a practical modeling of iterative computations in distributed processing systems of message-passing type. Consider index set I as a set of (virtual) spatial sites of individual processors within a multiprocessor network system and each X_i as a local state space at site i. In many practical situations, due to restrictions associated with the network model, topology, etc., sites may not all share directly a common global state space as a product set of all local state spaces (like X in the previous section), or they may not be able to distinguish local values coming from different local sites (as in shared memory models). Rather, each site has its own local conception of a global state space, namely, its own input state space Y_i, and local map f_i is defined from Y_i into X_i. The conversion from local state spaces X_j at all sites $j \in I$ to a local input space Y_i at site i is a (virtual) map $\phi_i: \prod_{j\in I} X_j \to Y_i$, which is determined or carried out by the network communication protocol. This formulation results in the following *spatially-extended* asynchronous iteration

$$x_i(t) = \begin{cases} f_i(\phi_i([x_j(s_j^i(t))]_{j\in I})) & \text{if } i \in I(t) \\ x_i(t-1) & \text{if } i \notin I(t) \end{cases}$$

It is clear that when all $Y_i = \prod_{j\in I} X_j$ and ϕ_i are identity maps, spatially-extended AI's reduce to the AI's on the product $\prod_{j\in I} X_j$. However, the conclusion of Theorem 7 still holds for this kind of extension of AI's.

Theorem 11 *Suppose that X_i and Y_i are all cpo's, and $f_i: Y_i \to X_i$, $\phi_i: \prod_i X_i \to Y_i$ are all continuous with respect to respective cpo's. Then a spatially-extended asynchronous iteration $\{x(t)\}$ of $F = [f_i]$ starting at $\perp$ in any asynchronous iteration scheme $(\mathcal{I}, \mathcal{S})$ is directed and converges to the least fixed point μF of F.*

5 Random Iterations

Definition 12 Given an ordinal α greater than or equal to the first limit ordinal ω [Sho67] and

given a non-empty (index) set I, an *α-random iteration scheme* is a map r from set $\{\beta \mid \beta \in \alpha\}$ of ordinals less than α into set I. An α-random iteration scheme r is *fair* if, for each $i \in I$, cardinality $\mathbf{Card}(r^{-1}(i))$ of set $r^{-1}(i) \subseteq \alpha$ is $\mathbf{Card}(\alpha)/\mathbf{Card}(\mathbf{I})$. ∎

Definition 13 Let X be a topological space, where a notion of limit of a (transfinite) sequence makes sense, and let $F = \{f_i : X \to X \mid i \in I\}$ be a family of maps. An *α-random iteration* or *α-trajectory* starting at a point $x_0 \in X$ according to an α-random iteration scheme r, denoted by x^r, is a (transfinite) sequence $\{x^r(\beta) \in X \mid \beta \in \alpha\}$, recursively defined as follows: $x^r(0) = x_0$, and, for any β, $0 \in \beta \in \alpha$,

$$x^r(\beta) = \begin{cases} f_{r(\beta-1)}x^r(\beta-1) & \text{if } \beta \text{ is a sucessor ordinal} \\ \lim_{\gamma \in \beta} x^r(\gamma) & \text{if } \beta \text{ is a limit ordinal.} \end{cases}$$

We often omit prefix α when $\alpha = \omega$ and simply speak of random iterations schemes and random iterations. ∎

In the case $\alpha = \omega$, the definition of α-random iterations reduced to the ones of execution sequences in [Tsi87] and random products in [Bru82].

A point x in X is a *common fixed point* (or *irreducible*, as used in the rewriting system theory [Hue80]) of a family F of maps on X if $f(x) = x$ for all f in F. Denote by $Fix(F)$ the set of all common fixed points of family F and by μF the least common fixed point of F when it exists. Clearly, a point x is a common fixed point of F if and only if any fair α-trajectory of x is a transfinite sequence of x itself. A point y is called an α-limit point (or α-normal form) of x if there is an α-random iteration x^r with $x^r(0) = x$ and $x^r(\alpha) = y$.

As for asynchronous iterations, convergence and determinacy are also two properties of major interest. A family F of maps on a topological space X is *convergent* if any random iteration of any x in X has a limit point which is a common fixed point, and F is *deterministic* if any point has at most a unique limit point that is a common fixed point. identical.

Random iterations find application in concurrent systems of shared-memory type, rewrite systems and rule-based deduction systems. The basic computing mechanism in all these systems is that one and only one process or rule out of a pool or a set of them is activated or applied at each computation step, resulting in a common piece of memory or a term being updated or rewritten successively. Consider rewriting systems [DJ90] as an example. Recall that a rewriting relation is a binary relation $\to$ on $T = T_\Sigma(X)$, the set of all terms on a signature Σ using variables from a countable set X, satisfying that, if $s \to t$ with $Var(s) \subseteq Var(t)$, where $Var(s)$ is the set of all free variables of s, then $u[s\sigma]_p \to u[t\sigma]_p$, for all terms s, t and u, all positions p in u, and all substitutions σ. A rewrite system is a set of rewriting rules of the form $u \to v$, where u, v are terms in T with $Var(u) \subseteq Var(v)$. Associated with a given rewrite system Γ is a family $F(\Gamma)$ of maps on terms constructed as follows, which represent the intensional meaning of the rewriting system. For any rewriting rule $u \to v \in \Gamma$, any substitution σ and any position p, define a map $f : T_X(\Sigma) \to T_X(\Sigma)$ by

$$f(s) = \begin{cases} t & \text{if } \exists w, s = w[u\sigma]_p, t = w[v\sigma]_p \\ s & \text{otherwise.} \end{cases}$$

and let $F(\Gamma)$ be the set of all such formed maps. It is clear that each map f is well defined and that, if Γ is countable, $F(\Gamma)$ is also countable. Also, a rewriting sequence in Γ corresponds to a random iteration of $F(\Gamma)$. As well known, the theory of rewriting systems is a theory of normal forms. Important properties of a rewriting system are (finite) termination, which guarantees the existence of normal forms, and confluence (or the Church-Rosser property), which ensures the uniqueness of normal forms. It can be easily seen that these two properties correspond to the properties of convergence and determinacy of random iterations, if we do not insist on finite terminations, and that a normal form in a rewriting system Γ is a common fixed-point of corresponding family $F(\Gamma)$. Thus, studying random iterations of $F(\Gamma)$ is equivalent to studying rewritings in Γ.

Also because random iterations have a wide range of applications, many fundamental concepts and results in various applicative areas help to pursue a general study of RI's. For instance, a concept of confluent RI's can be formulated in the same spirit of rewriting systems. A family $F = \{f_i\}$ of maps is α-confluent if $\stackrel{\alpha}{\leftarrow} \circ \stackrel{\alpha}{\rightarrow} \subseteq \stackrel{\alpha}{\rightarrow} \circ \stackrel{\alpha}{\leftarrow}$, where $\stackrel{\alpha}{\rightarrow}$ is the α-transfinite reflexive-transitive closure of the union of graphs $\{(x, f_i(x))\}$ of all f_i's, $\stackrel{\alpha}{\leftarrow}$ is the inverse of $\stackrel{\alpha}{\rightarrow}$, and $\circ$ is the composition of binary relations.

Not surprisingly, we can have the following result, which corresponds to a similar fundamental result on existence and uniqueness of normal forms in rewriting systems.

Theorem 14 *If F is α-convergent and α-confluent, then each point has a unique α-normal form.*

Proof: Since F is α-convergent, any point has an α-normal form. If a point x has two normal forms, say y, z, then $y \rightarrow^{\alpha} w$, $z \rightarrow^{\alpha} w$ for some w, which is only possible when $y = w = z$ as y, z are normal forms. ∎

6 Computing Least Common Fixed Points

In this section, we are interested in how to use random iterations to compute the least common fixed point μF of a family F of continuous maps on a cpo X. A classic result related to this is the following cpo-and-continuous-map version of the corresponding theorems occurred in [Tar55, DeM64, CC79] for monotone maps on lattices.

Theorem 15 *Let F be a commuting family of continuous maps from a cpo X into itself, i.e., $f(g(x)) = g(f(x))$ for all $f, g \in F$ and all $x \in X$. Then F has a non-empty set of common fixed points $Fix(F)$ in which the least element μF exists. Moreover, when the family is finite, $F = \{f_i \mid i = 1, 2, ..., n\}$, and g is a composite map of F, i.e., $g = f_{i_1} f_{i_2} \cdots f_{i_n}$, $Fix(F) = Fix(g)$.*

Since common fixed points of a finite, commuting family F of monotone maps are identical to fixed points of any composite map g, the least common fixed point μF is the least fixed point μg. If, in addition, maps in F are all continuous, then μF can be simply computed by iterating g starting with the bottom element; that is,

$$\mu F = \mu g = \sqcup_{n \rightarrow \infty} g^n(\perp).$$

We realize that iterating a composite map is a random iteration scheme and the above theorem can be indeed generalized to any random iteration scheme.

Theorem 16 *The least common fixed point μF of a finite, commuting family F of continuous maps from a cpo X into itself can be computed by any fair random iteration x^r of the family starting with bottom element $\perp$, $\mu F = \sqcup_{n\geq 0} x^r(n)$ with $x^r(0) = \perp$.*

Proof: Assume that $F = \{f_1, ..., f_m\}$ and fix a composite map g of F, say $g = f_1 \cdots f_m$. Now let r be a fair random iteration scheme and x^r be the trajectory starting with $\perp$. Consider an increasing sequence of integers, $n_0 < n_1 < ... < n_i < ...$ such that, for any $i \geq 0$, $x^r(n_i) = h_i(g^i(\perp))$ for some h_i (which is a composite of some maps in F) and $\{x^r(n_i)\}$ is a nondecreasing sequence. Clearly, such sequence $\{x^r(n_i)\}$ exists thanks to r being fair and each h_i is continuous. Notice that

$$x^r(n_i) = h_i(g^i(\perp)) = g^i(h_i(\perp)) \sqsupseteq g^i(\perp).$$

Then

$$\mu F = \sqcup_{i\geq 0} g^i(\perp) \sqsubseteq \sqcup_{i\geq 0} x^r(n_i) \sqsubseteq \sqcup_{n\geq 0} x^r(n) \sqsubseteq \mu F,$$

which shows $\sqcup_{n\geq 0} x^r(n) = \mu F$. ∎

A simple case where a family F is commuting is that F consists of a finite number of iterates of a single map f, that is, $F = \{f^{i_1}, ..., f^{i_m}\}$. The previous two theorems say that $Fix(F) = Fix(f)$ and $\mu f = \mu F$ can be computed by any random iterations of F. This could be useful to speed up convergence rate of iterating f in a multiprocess environment in which each constituting process computes an iterate f^{i_j} of f.

Certainly, F being a commuting family is a strong condition. We now relax it to a semi-confluent family.

Definition 17 Given a family F of maps on a set X, denote by F^* the set of all possible finite composite maps out from F. F is said to be *strong semi-confluent* if for any $x \in X$, $u, v \in F^*$, $u(v(x)) \sqsupseteq v(u'(x))$ for some $u' \in F^*$. F is said to be *weak semi-confluent* if for any $x \in X$, $f \in F$, $u \in F^*$, $u(f(x)) \sqsupseteq f(u'(x))$ for some $u' \in F^*$. ∎

Clearly, a strong semi-confluent family is weak semi-confluent, and a commuting family is strong semi-confluent. Generally, a weak semi-confluent family is not necessarily strong.

Another example of strong semi-confluent families comes from a restricted class of asynchronous iterations. According to the definition, AI's are always associated with product spaces and allow using any previous values at each iterative step, while RI's live on any set but are restricted to only using most recent values. When AI's of a map F on a product set $X = \prod_{i\in I} X_i$ are restricted to execution sequences (see Section 3) which use only the most recent values, they are equivalent to RI's of a family of maps $f_{I_0}: X \to X$, $I_0 \subseteq I$, defined by

$$(f_{I_0}(x))_j = \begin{cases} x_j & j \notin I_0 \\ (F(x))_j & j \in I_0 \end{cases} \quad \text{for every } j \in I,$$

and the least common fixed point of f_{I_0}'s exists and is equal to the least fixed point of F. From Lemmas 5, 6 in Section 4, the resulting family $\{F_{I_0}\}$ is strong semi-confluent.

Theorem 18 *Let X be a ω-complete poset and F a weak semi-confluent family of monotone maps on X with a non-empty set of common fixed points, $Fix(F) \neq \emptyset$. Then*

(i) there exists a least common fixed point μF in $Fix(F)$;

(ii) for any $x \in X$, if $f(x) \sqsubseteq x$ for all $f \in F$, then $\mu F \sqsubseteq x$; and

(iii) $Fix(F)$ is ω-complete with respect to the induced order.

Proof: (i) Let

$$A = \{x \in X \mid x \sqsubseteq f(x) \text{ for all } f \in F \text{ and } x \sqsubseteq y \text{ for all } y \in Fix(F)\}.$$

Clearly, $\perp \in A$. If $g \in F$, $x \in A$, $y \in Fix(F)$, $g(x) \sqsubseteq g(y) = y$. If in addition $f \in F$, then there exists some $u \in F^*$ such that $f(g(x)) \sqsupseteq g(u(x)) \sqsupseteq g(x)$, because of the weak semi-confluent property and $x \in A$. Thus, $g(A) \subseteq A$. It is easy to see that A is ω-complete. For any fair random iteration x^r starting with an element $x^r(0)$ in A, let $f(x) = \sqcup_{n=0}^{\infty} x^r(n)$. Clearly, $x \sqsupseteq f(x)$ for all x in A. Then f has a fixed point $a \in A$, i.e., $x^r(n) = a$ as long as $x^r(0) = a$. Thus $a = \mu F$.

(ii) Let $x\downarrow = \{z \in X \mid z \sqsubseteq x\}$. $x\downarrow$ is ω-complete and $f(x\downarrow) \subseteq x\downarrow$ for all $f \in F$. By (i), $x\downarrow \cap Fix(F) \neq \emptyset$. Thus, $\mu F \sqsubseteq x$.

(iii) Let $\{y_n \mid n \geq 0\}$ is an ω-chain in $Fix(F)$ and let $y = \sqcup_X y_n$, where $\sqcup_X$ mean that $\sqcup$ is taken upon X. The upper set $y\uparrow$ is ω-complete. It is easy to verify that $f(y\uparrow) \subseteq y\uparrow$ for all $f \in F$. By (i), there is a least fixed point z in $Fix(F) \cap y\uparrow$. Clearly, $z = \sqcup_{Fix(F)} y_n$, where $\sqcup_{Fix(F)}$ mean that $\sqcup$ is taken upon $Fix(F)$. ∎

In that case that F is further a strong semi-confluent family, the theorem implies F is deterministic. More precisely,

Corollary 19 *Let X be an ω-complete poset and F a strong semi-confluent family of monotone maps on X. If a fair α-random iteration x^r starting with bottom element $\perp$ converges to a fixed point, then the fixed point is μF.*

Proof: The proof is similar to the one for (i) of the theorem. First of all, the theorem guarantees the existence of μF. Let

$$A = \{x \in X \mid x \sqsubseteq u(x) \text{ for all } u \in F^* \text{ and } x \sqsubseteq \mu F\}.$$

Clearly, $\perp \in A$. If $v \in F^*$ and $x \in A$, $v(x) \sqsubseteq v(\mu F) = \mu F$. If in addition $u \in F^*$, then there exists some $u' \in F^*$ such that $u(v(x)) \sqsupseteq v(u'(x)) \sqsupseteq v(x)$, because of the strong semi-confluent property and $x \in A$. Thus, $v(A) \subseteq A$. Also any α-random iteration x^r starting with an element in A is nondecreasing and $x^r(\beta) \sqsubseteq \mu F$. Hence if $y = x^r(\alpha)$ is a fixed point, then $y = \mu F$. ∎

7 Conclusion

We have studied two kinds of general iterations: asynchronous iterations of a map on a product set and random iterations of a family of maps on a common set. Our results show that the least fixed point computed by a continuous map on a product order and alternative extension orders on the product of a family of complete partially ordered sets can be computed by arbitrary asynchronous iterations, and that the least common fixed point of an α-semi-confluent family of continuous maps on a common cpo can be computed by any fair α-random iterations, which generalizes a classic result on computing the least fixed point of a commuting family of continuous maps by a composite map.

Least fixed points of continuous maps on various cpo's play a very important role in (theoretical) computer science. One of the most significant application is to provide a rigorous semantics definition of programming languages ranging from deterministically sequential to nondeterministically distributive ones [NR85]. Since also every recursive function is the least fixed point of certain continuous functionals on some complete partially ordered set [Kle52], we expect that the results obtained here may soon find some applications particularly in recursive theory [Mos89a, Mos89b] and computer science, *e.g.*, program semantics [NR85, Par87, Sco76], verification [dB80, Man74], and synthesis and transformation [BW82, MGMW79].

References

[Bar91] A. Baranga. The contraction principle as a particular case of kleene's fixed point theorem. *Discrete Mathematics*, 98:75–79, 1991.

[Bau78] G. M. Baudet. Asynchronous iterative methods for multiprocessors. *Journal of the Association for Computing Machinery*, 25:226–244, 1978.

[Bek84] H. Bekič. Programming languages and their definitions. volume 177 of *Lecture Notes in Computer Science*. Springer-Verlag, 1984.

[Bir69] G. Birkhoff. *Lattice Theory*, volume 25. AMS Colloquium Publications, third edition, 1969.

[Blu72] E. K. Blum. *Numerical Analysis and Computation: Theory and Practice*. Addison-Wesley, 1972.

[Bos85] N. K. Bose, editor. *Recent Trends in Multidimensional Systems Theory*. Reidel Publication Co., 1985.

[Bru82] R. E. Bruck. Random products of contractions in metric and banach spaces. *Journal of Mathematical Analysis and Applications*, 88(2):319–332, 1982.

[BT89] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Prentice Hall, 1989.

[BW82] F. L. Bauer and H. Wosser. *Algorithmic languages and Program Development*. Springer, Berlin, 1982.

[CC79] P. Cousot and R. Cousot. Constructive versions of tarski's fixed point theorems. *Pacific Journal of Mathematics*, 82:43–57, 1979.

[CM69] L. V. Chazan and W. Miranker. Chaotic relaxation. *Linear Algebra and Applications*, 2:199–222, 1969.

[dB80] J. de Bakker. *Mathematical Theory of Program Correctness.* Prentice-Hall, New York, 1980.

[DeM64] R. DeMarr. Common fixed points for isotone mappings. *Colloquium Mathematics*, XIII:45–48, 1964.

[DG82] J. Dugundji and A. Granas. *Fixed Point Theory.* Warszawa, 1982.

[DJ90] N. Dershowitz and J.-P. Jonannaud. Rewriting systems. In A. Meyer, M. Nivat, M. Paterson, and D. Perrin, editors, *Handbook of Theoretical Computer Science.* North-Holland Publishing Co., Amsterdam, 1990.

[DP90] B. A. Davey and H. A. Priestley. *Introduction to lattices and orders.* Cambridge University Press, 1990.

[GCP85] E. Goles-Chacc and D. Pellegrin. Decreasing energy functions as a tool for studying threshold networks. *Discrete Applied Mathematics*, 12:261–277, 1985.

[GHK+80] G. Gierz, K. H. Hofmann, K. Keimel, J. D. Lawson, M. Mislove, and D. S. Scott. *A Compendium of Continuous Lattices.* Springer-Verlag, Berlin, 1980.

[Hop82] J. Hopfield. Neural networks and physical systems with emergent computational abilities. *Proc. Nat. Acad. Sci. USA*, 79:2554–2558, 1982.

[Hue80] G. Huet. Confluent reduction: Abstract properties and applications to term rewriting systems. *JACM*, 27(4):797–821, 1980.

[JMP86] E. M. Jawhari, D. Misane, and M. Pouzet. Retracts: Graphs and ordered sets from the metric point of view. In I. Rival, editor, *Combinatorics and Ordered Sets*, volume 57 of *Contemporary Mathematics*, pages 175–226. American Mathematical Society, Providence, RI, 1986.

[Kle52] S. C. Kleene. *Introduction to Metamathematics.* Van Nostrand, New York, 1952.

[LaS86] J. P. LaSalle. *The Stability and Control of Discrete Processes*, volume 62 of *Applied Mathematical Sciences.* Springer-Verlag, 1986.

[Man74] Z. Mannar. *Mathematical Theory of Computation.* McGraw-Hill, New York, 1974.

[MGMW79] M. M. Gordon, A. Milner, and C. Wadsworth. *Edinburgh LCF.* Springer, Berlin, 1979.

[Mos89a] Y. N. Moschovakis. The formal language of recursion. *Mathematics Department, UCLA*, 1989.

[Mos89b] Y. N. Moschovakis. A mathematical modeling of pure, recursive algorithms. *Mathematics Department, UCLA*, 1989.

[NR85] M. Nivat and J. C. Reynolds, editors. *Algebraic Methods in Semantics.* Cambridge University Press, 1985.

[Par87] D. S. Parker. Partial order programming. *Technical Report CSD-870067, Computer Science Department, UCLA*, 1987.

[Rob86] F. Robert. *Discrete Iterations.* Springer, Berlin, 1986.

[Sco72] D. Scott. *Continuous Lattices*, volume 274 of *Lecture Notes in Mathematics.* Springer-Verlag, 1972.

[Sco76] D. Scott. Data types as lattices. *SIAM Journal of Computing*, 5:522–587, 1976.

[Sho67] J. R. Shoenfield. *Mathematical Logic.* Addison-Wesley, Reading, MA, 1967.

[Smy91] M. B. Smyth. Totally bounded spaces and compact ordered spaces as domains of computation. In G. M. Reed, A. W. Roscoe, and R. F. Wachter, editors, *Topology and Category Theory in Computer Science*, pages 207–229. Clarendon Press, Oxford, 1991.

[Tar55] A. Tarski. A lattice-theoretical fixpoint theorem and its applications. *Pacific Journal of Mathematics*, 5:285–309, 1955.

[Tsi87] J. N. Tsitsiklis. On the stability of asynchronous iterative processes. *Mathematical System Theory*, 20:137–153, 1987.

[Wan89] Xin Wang. Computing least fixed points by asynchronous iterations. *Research Manuscript, Computer Science Department, UCLA*, 1989.