Modeling and Analyzing Distributed Computation in Monotone Spaces with Structural Map

Susmit Bagchi

Department of Aerospace and Software Engineering (Informatics), Gyeongsang National University, Jinju, Republic of Korea 660-701
susmitbagchi@yahoo.co.uk

Abstract

Distributed computation follows the models of discrete structures in combinatorial forms. In higher-dimensions, the simplex structures of topological spaces as well as homology are employed to model and analyze distributed asynchronous computations. However, the monotone spaces are the general forms of topological spaces and can be effectively employed to analyze distributed computation. This paper proposes an analytical model of distributed computation in monotone spaces. It is illustrated that, the modeling of distributed computation in monotone spaces helps in determining consistent cuts under closure and convergence of computation. Furthermore, a connective mapping between the simplexes and monotone is constructed.

Keywords: Distributed computing, shared memory, topology, monotone spaces, consistency

1. Introduction

The distributed computing systems are comprised of interacting processes executing at distributed nodes, where the nodes are connected by network. The network-topology of the interconnection graph in the systems involving distributed nodes can be static or dynamic depending on architecture. In general, the interactive distributed processes compute following the iterated shared memory model [6]. According to this model, distributed processes access and update a sequence of shared objects asynchronously following a deterministic sequence. The asynchronous and concurrent accesses to shared memory locations by multiple distribute processes invite the issues related to race condition and concurrency control. The control of concurrency becomes highly complex in distributed computing systems due to the availability of partial information about global states of computations. Thus, the designing of distributed computational model and algorithms often assume that the consistent global information about computation in a distributed system is not available at all the nodes. However, the employment of algebraic topological theory in computation architecture enables the handling of concurrency control in concurrently executing programs [2, 5]. This is because, the algebraic topology helps in partitioning of set of structures of objects in distributed computation by recognizing consistent structural forms from the set of non-conformal forms. Researchers have indicated that, the distributed computing systems have a closer interplay with algebraic topology [1]. On the other hand, the monotone spaces are the more generalized form of topology [3]. This paper argues that, the structural dynamics of monotone spaces have direct applications in modeling and analyzing distributed computing systems. The monotone property of topological spaces may be used to determine consistent cuts and to analyze the convergence of distributed computing in monotone topological spaces. In this paper, the concept of boundary elements within the state-space of distributed processes is introduced. The set of boundary elements partitions the state-space of distributed
computation by isolating regions of convergences, which helps in analyzing the consistency and termination in a distributed computation. The monotone spaces of distributed computation are partitioned into two sub-spaces and, and connectedness of the monotone topology is analyzed. A set of analytical properties is presented following the model of distributed computing in monotone topological spaces. The closure property of monotone spaces to maintain consistent cuts in a distributed computation is constructed. Furthermore, it is illustrated that there exists a connective mapping between simplexes and monotone spaces.

1.1. Motivation

The distributed computing systems involve large set of concurrent as well as distributed processes. The distributed processes communicate and exchanges messages or data in order to coordinate the distributed computations. The concurrent communication and coordinated executions at distributed nodes require shared objects placed in shared memory locations in the distributed systems. The maintenance of atomicity property and consistency of distributed shared objects are highly complex. The concurrency control in parallel and distributed asynchronous systems has similarities to algebraic topology and homotopy [11]. Moreover, it is proposed that, the algebraic topological spaces have a close resemblance to the iterated shared memory model of distributed computation [1]. Thus, an algebraic topological approach can be applied to model and analyze distributed computation in the presence of concurrency. However, the iterated model considers that all distributed processes accesses shared objects in deterministic sequences and only once [6]. These assumptions reduce the complexity of concurrency control however, these are comparatively rigid assumptions about the distributed computing models. It is proposed that the concurrency in a program representing a local process can be controlled by employing the directed homotopy [2]. However, the algebraic homotopy concepts are required to be adapted to concurrent distributed computing structures in order to model distributed consistency and convergence. The monotone topological spaces consider the connectedness in the weaker spaces and provide ending property [3]. Hence, the consistency and convergence properties of a distributed computing system can be modeled as the monotone spaces, which are generalized topological spaces. This paper proposes the modeling and analysis of distributed computation in monotone topological spaces. It is illustrated that, the global consistency of a distributed computation can be maintained in the presence of closure in the monotone spaces. The convergence property indicates the termination of a distributed computation in consistent state following the boundary elements. The main contributions of this paper are as follows.

- A model of distributed computing in monotone topological spaces is constructed.
- The sets of boundary elements are identified to maintain convergence of computation.
- A set of analytical properties is formulated representing consistency and termination of distributed computation.
- The closure property of monotone topological spaces is formulated based on consistency of distributed computations.
- The connective mapping is constructed between simplexes and monotone spaces.

Rest of the paper is organized as follows. Section 2 represents related work. Section 3 and 4 represent preliminary concepts, definitions and model. Section 5 illustrates a set of analytical properties. Section 6 describes comparative analysis and formation of connective map between spaces. Finally, Section 7 concludes the paper.
2. Related Work

The applications of concepts of algebraic and combinatorial topology into the domain of distributed computing have gained research attentions. The algebraic topology is about higher dimensional geometrical shapes of objects [13, 20]. The topological structure of higher dimensional automata can model the properties of concurrent processes and distributed computation [14]. The persistent homology is employed as a formal mechanism to construct the topological characteristics of data and to model distributed computation [16, 19]. The combinatorial and algebraic topological methods are applicable to model and analyze distributed computing systems following asynchronous iterated shared memory model [1, 6]. However, the issues related to concurrency control in distributed asynchronous systems are highly complex. Interestingly, the homotopy theory along with topology can be applied to solve the mutual exclusion issues related to concurrent programs [2, 12]. It is shown that stability properties of distributed concurrent processes can be proved by considering homotopy [11]. The semaphore objects can be formed by using partially ordered topological spaces and, can be extended to analyze deadlock as well as serializability properties of concurrent processes [10]. However, the general homotopy does not prevent time-reversal and thus, the directed homotopy is required in model construction.

The structural forms of algebraic topology can be equally applied to the synchronous as well as asynchronous distributed computing systems [7, 17]. The topological objects are generalizations of graphs with higher dimensions and, the connectivity property of topological objects can be employed to analyze the computability problems of processes in distributed computing [7, 9]. The simulation methods utilizing topological objects are proposed by researchers to prove impossibility results in distributed computing systems [9]. Furthermore, the algebraic topological concepts are employed in deriving the time complexity bounds to compute approximate agreement for the iterated immediate snapshot model [8]. In this approach, the time complexities of the distributed processes are mapped into the degree of sub-division of input complex in order to generate output complex. In general, these algebraic topological objects form simplical complexes to model distributed processes and protocols, which are having very complex and rigid structural geometries [4, 17]. The combinatorial topological model of asynchronous processes and wait-free computation involve defining combinatorial relations in topological spaces [4]. However, these relations are static and do not consider operational mode of interleaving computations. In case of immediate snapshot model of distributed computation, the simplical complexes can be reduced to manifolds [9]. This method reduces the complexity of structures of simplical complexes. Following an alternative approach, the characterizations of object-based distributed systems are realized by using topological algebraic structures [15]. The formal model of characterizations tries to cover the distributed systems, in general. The characterizations of liveness and safety properties of concurrent distributed systems are formulated by using topological structures [18]. These algebraic topological approaches do not consider monotone property of topological spaces [3]. The extension of monotone topological spaces into distributed computing systems can determine the convergence and closure of consistent distributed computation.

3. Preliminary Concepts

In this section, a set of topological and computational concepts are presented, which are employed in constructing the model. In general, a distributed computation can be formulated as a simple graph $G = (V, L)$, where $V$ is the set of vertices and $L$ is set of edges such that, $L \subseteq \{(v_a, v_b) : v_a, v_b \in V\}$. If $V$ signifies a set of processes $P$ in the distributed systems, then $L$ signifies set of communication channels between distributed processes. Otherwise, if $V$ signifies set of states in $P$ then, $L$ signifies the set of all compatible states of the corresponding processes. The simple graph $G$ is
complete and connected if \( L = V^2 \setminus \{ (v_a, v_b); \forall v_a, v_b \in V \}. \) On the other hand, if \( W \) is a point set and, \( u \subseteq \Omega(W) \) where \( \Omega(.) \) is power set then, the topological space is \( (W, u) \) having properties: \( W, \phi \) are open; \( \forall u_a, u_b \in u, u_a \cap u_b \) and \( u_a \cup u_b \) are open. The simplicial complexes are the higher-dimensional analogues of the graphs representing the distributed processes \([1, 2, 4]\). Considering that, \( \forall v_j \in V \) represents a computational state of a process \( pj \in P \), \( n \)-simplex contains set \( \{ v_a : a = n+1, v_a \in V \} \) representing compatible states of \( n+1 \) distributed processes in a system. Following the concepts of topological spaces, simplicial complexes \( W_c = \{ w_n : n > 0, n \in \mathbb{Z}^+ \} \) where, \( w_n \) is simplex and \( W_c \) is finite having properties: \( \forall w_m, w_m \in W_c, w_n \cap w_m \in W_c \) and, \( w_n \cup w_m \in W_c \). A simplicial map between simplexes \( (W_{c1} \text{ and } W_{c2}) \) is given by, \( \partial : W_{c1} \rightarrow W_{c2} \). This is a linear piece-wise mapping between vertex to vertex between two simplexes. An \( n \)-dimensional simplex is called a manifold if every \( (n-1) \) simplex contain in two \( n \)-simplexes and, the boundary of the manifold is generated by sub-complex formed by every \( (n-1) \)-dimensional external simplexes. The monotone spaces are the more generalized form of topological spaces \([3]\). If \( g \) is considered to be an operating function such that, \( g(\phi) = \phi \); \( \forall A \in \Omega(W), A \subset g(A) \) and, \( \forall A, F \in \Omega(W), A \subset F \Rightarrow g(A) \subset g(F) \) then, \( (W, g) \) is a monotone space. Any arbitrary intersections of closed sets in monotone spaces are closed.

4. Definitions and Model

In this section, the basic model is formulated and a set of definitions is constructed in order to formulate the model. Let a distributed computation be represented as \( D \) comprised of a set of distributed processes given by \( P = \{ pj : 1 \leq j \leq N, j \in \mathbb{Z}^+ \} \). All the distributed processes are considered as finite state machines and a \( pj \) has a set of deterministic execution states denoted by \( S_j \), where \( \phi \notin S_j \). Hence, \( D \) can be identified by,

\[
S(D) = \bigcup_{j=1}^{N} S_j
\]  

(1)

The distributed processes execute as state machines with in-process communications and as a result state transitions occur in a process, which are governed by a function given by,

\[
f : S(D) \rightarrow S(D)
\]

\[
f(s_j \in S_j) \in S_j \setminus \{ s_j \}
\]

(2)

The distributed computation is considered to be deterministic if it converges to a space and such converging space of \( D \) is defined as,

\[
S(D) = \bigcap_{j=1}^{N} S_j
\]

(3)

The set of all possible consistent cuts on \( D \) is denoted by \( C(D) \subset \Omega(S(D)) \).

A process \( pj \) in \( P \) maintains the state of the channels \( Q_j \subset P^2 \) where, \( Q_j = \{ (pj, pk) : pk \in P \} \). The process \( pj \) maintains the number of messages in transit within its channels by following a channel function at a computing state given by, \( \chi : S_j \times \mathbb{Z}^+ \rightarrow \mathbb{Z}^+ \). It is important to note that, the consistent cut of a distributed computation is highly dependent on the values of \( \chi(.) \) at every node.
4.1. Definition: Boundary Elements

The set of boundary elements $B_j$ of a process $p_j$ is defined as, $\forall p_j \in P$,

\[ f (B_j \subset S_j) \in S(D) \quad (4) \]

The concept of boundary elements represents the recognition of regions of convergences in state-space of a distributed computation.

4.2. Definition: Computation in $D$

If $B_j \subset S_j$ then a computation in $D$ is defined as,

\[ \tilde{D} = S(D) \setminus f \left( \bigcup_{j=1}^n B_j \right) \quad (5) \]

This indicates that, the executions in a distributed computation consider global states excluding the convergent region. However, it includes the boundary state-space.

4.3. Definition: Monotone Distributed Computing

A monotone space over $\tilde{D}$ is $(\tilde{D}, g)$ having following properties:

\[
g : \Omega(\tilde{D}) \rightarrow \Omega(\tilde{D}) \\
g(\phi) = \phi \\
\forall A \in \Omega(\tilde{D}), A \subset g(A) \\
\forall A, B \in \Omega(\tilde{D}), A \subset B \Rightarrow g(A) \subset g(B) \quad (6)\]

This definition represents the monotone property of a distributed computation and the mapping of structural forms within the state-space of distributed computation under consideration.

4.4. Definition: Connected Monotone

A monotone topological space $(\tilde{D}, g)$ of distributed computation $D$ is connected if following properties hold:

\[
\forall sj \in \tilde{D}, n \geq 1 \\
A = \Omega(f^{-1}(sj)) \setminus \{\phi\} \quad (7) \\
g(A) \setminus A \subset (\Omega(\tilde{D}) \setminus \bigcup_{j=1}^n \Omega(S_j))
\]

The connectedness of monotone space of a distributed computation represents that, the processes execute as a set of finite state machines having interconnection channels. Moreover, the set of structural forms of a distributed computation consider the instantaneous states of all processes existing in the system. It is assumed that, each process existing in a distributed computing system contains a distinct initial state and the corresponding final state. Moreover, a process can have any arbitrarily large but countable and finite set of intermediate states.
4.5. Definition: Convergent Monotone

In a connected monotone topological space \((\tilde{D}, g)\), \(\forall pj \in P, \forall sj \in \tilde{D}\), if \(\exists H \in g\) such that, \(Bj \subset H\) then \((\tilde{D}, g)\) is a convergent monotone of distributed computation \(D\). This definition indicates that, determination of eventual convergence of a distributed computation requires dynamics of state-space of every processes of the respective distributed computation.

4.6. Definition: Consistent Monotone

A monotone topological space \((\tilde{D}, g)\) of a distributed computation \(D\) is called consistent when the following holds:

\[
\forall A \in \Omega(\tilde{D}),
A \in C(\tilde{D}) \Rightarrow g(A) \subset C(\tilde{D})
\]

(8)

The monotonic consistency represents that, the consistent cuts of a distributed computation form a monotone topological space. In other words, the monotone topological space can recognize a set of consistent cuts of a distributed computation.

5. Analytical Properties

A distributed computing system in a monotone topological space has dynamic structures. The convergence and consistency properties of distributed computing vary in topological spaces over time depending on the executions of distributed processes. Moreover, the termination of a distributed computation can be validated in the monotone topological spaces if a set of properties is maintained. In this section, a set of analytical properties of the distributed computing in monotone topological spaces is formulated.

5.1. Theorem 1: In a convergent distributed computation \(D\), \(S(D) \neq \phi\).

Proof: The proof is by contradiction. Let \(S(D) = \phi\) in \(D\). This indicates that, \(\bigcup_{j=1}^{n} Bj = \phi\) in the distributed computation \(D\). However, \((\tilde{D}, g)\) is a convergent monotone.

Let \(\exists sj \in \tilde{D}\) such that, \(A = \{ sj^* : sj^* \in Sj, sj^* = f^*(sj), n \geq 1 \}\) and, \(E \in g(A)\). Thus, \(\exists x \in E\) where \(f(x) \in S(D)\). However, \(f(x) \neq \phi\) which is a contradiction. Hence, \(S(D) \neq \phi\).

Descriptions: The space of convergence cannot be empty in a convergent distributed computation in monotone topological spaces.

5.2. Theorem 2: If \(D\) is in convergent monotone topological spaces then \(\forall pj \in P, |Bj| \geq 1\).

Proof: The proof is by contradiction. Let \(\exists pj \in P\) such that, \(\forall sj \in Sj, f(sj) \notin S(D)\).

Now, \((\tilde{D}, g)\) is a connected monotone if \(D\) is convergent. However, \(\exists A \in \Omega(\tilde{D}) \setminus \bigcup_{j=1}^{n} \Omega(Sj)\), where \(\exists x \in A, f(x) \notin S(D)\). This further indicates that, \(\exists E \in g(A)\) such that, \(\exists x \in E, f(x) \notin S(D)\). It concludes that \((\tilde{D}, g)\) is not convergent, which is a contradiction. Hence, \(\forall pj \in P, Bj \neq \phi\) in a convergent \(D\) in monotone spaces.
Descriptions: There are no distributed processes without boundary states if the corresponding distributed computation is convergent.

5.3. Theorem 3: A consistent monotone topological space is closed.

Proof: The proof is by induction. Let \((D, g)\) be a consistent monotone over distributed computation \(D\) and, \(C(D) \subseteq \Omega(D)\). Let \(A \in C(D)\). Thus, if \(E \subseteq A\) then, \(g(E) \subseteq g(A)\) following monotone topological property. However, \(g(A) \subseteq C(D)\). Hence, \((D, g)\) is closed under consistency.

Descriptions: In a consistent distributed computation, the consistency property is deterministic within the domain of all consistent cuts at any time.

5.4. Theorem 4: A distributed computation will terminate if the monotone topological space is consistent and convergent.

Proof: The proof is by induction. Let \(\exists p_j \in P\) such that, \(|B_j| \geq 1\) and, \((D, g)\) is a consistent as well as convergent monotone. Let \(A \subseteq \Omega(D)\) such that, \(A \subseteq C(D)\). Thus, \(\exists H_j \in g(A)\) where, \(B_j \cap H_j \neq \emptyset\) and, \(f(B_j \cap H_j) \in S(D)\). As \((D, g)\) is consistent and convergent, hence, \(\forall p_j \in P\), \(\exists H_j 
\in g(A)\) such that, \(f(B_j \cap H_j) \in S(D)\). This completes the proof.

Descriptions: The convergent and consistent distributed computation in monotone topological spaces eventually terminates following boundary elements in the states of processes.

5.5. Theorem 5: In a distributed computation, if \(A \subseteq \Omega(D)\) and, \(A \subseteq C(D)\) then, \(\forall a \in A, \forall x \in A, \exists z_x \in Z^*\) such that, \(\sum_{x \in A} \chi(x, z_x) = 0\).

Proof: Let \(A \subseteq C(D) \subseteq \Omega(D)\) be in a distributed computation. If \(\exists a \in A\), such that, \(\exists x \in a\) and, \(\exists z_x \in Z^*\), \(\chi(x, z_x) > 0\) then, \(A \subseteq C(D)\) by the definition of consistent cut. Moreover, in a distributed computation \(\forall a \in A, a \in \Omega(D)\) and, \(\chi(a \times Z^*) \geq 0\). Thus, \(\forall x \in a, \exists z_x \in Z^*\) such that, \(\chi(x, z_x) = 0\) if \(a \in A\). Hence, \(\forall a \in A\), if \(A \subseteq C(D)\) then, \(\sum_{x \in A} \chi(x, z_x) = 0\).

Lemma 5.1: If \(\sum_{x \in A} \chi(x, z_x) = 0\) where, \(a \in A\) and, \(A \subseteq \Omega(D) \setminus C(D)\) then, \(A\) is a set of consistent sub-cuts iff \(|a| < N\).

Descriptions: The instantaneous states of channels cumulatively affect the determination of consistent cuts. The consistent sub-cuts can be determined with a partition of distributed processes if the respective channels are empty. However, in all cases except the consistent cuts and sub-cuts, at least one channel is not empty.
5.6. **Theorem 6**: If \( A \subset \Omega(D) \) and \( C(D) \) then, \( \sum_{x \in A} \chi(x, z_x) \geq 1 \) iff \( \forall a \in A, |a| = N. \)

**Proof**: Let \( \forall a \in A \) where it holds that, \( A \subset \Omega(D) \) and \( C(D) \) and, \( |a| = N. \) Thus, \( \exists x \in a \) and \( \exists x \in Z^* \) such that, \( \chi(x, z_x) \geq 1. \) If \( \forall y \in a \setminus \{x\} \) in a distributed computation such that, \( \sum_{y \in a \setminus \{x\}} \chi(y, z_y) = 0 \) then, \( \chi(a \times Z^*) \geq 1. \) On the other hand, in a distributed computation if \( \forall x \in a, \chi(x, z_x) \geq 1 \) then, \( \chi(a \times Z^*) > 1. \) Hence, in the lower bound, \( \sum_{x \in a} \chi(x, z_x) \geq 1 \) iff \( \forall a \in A, |a| = N. \)

**Descriptions**: The instantaneous states of channels in accumulation cannot be empty if the process states are not in consistent cut.

6. **Comparative Analysis**

The traditional approach for modeling and analyzing asynchronous distributed computation in topological spaces relies on the formation of structures of simplicial complexes [4]. Let, \( X \subset \Omega(S(D)) \) and, \( \forall A \in X, F \subset A \Rightarrow F \in X \) where, \( F \neq \phi. \) Hence, \( A \) is a simplex of \( X. \) On the other hand, within the monotone spaces if \( F = g(A) \) then, \( A \subset F \) and, \( g(A) \subset g(F). \) However, in the both cases, if \( M = \Omega(A) \setminus \{\phi\} \) then, \( X \cap M \neq \phi. \) Furthermore, the monotone functional operation can generate a simplical complex if it is constructed while maintaining certain properties. Let \( K = g(A), A \in \Omega(S(D)). \) If \( K \) has the property that, \( \forall E \in K, F \subset E \Rightarrow F \in X \) and, \( F \neq \phi \) then, \( E \) is a simplex of \( K \) generated by monotone operation \( g(.) \) over \( A. \) Moreover, \( A \subset K \) and, if \( Y \subset A \) then, \( Y \in K. \)

6.1. **Inter-spaces Map**

The interrelation and mapping between monotone spaces and simplexes can be formed by considering general topological spaces. Let \( X = \Omega(A \in X) \) be represented in simplex structure in topological spaces, where \( X \) is a basis set (point-set). Thus, \( F \subset A \Rightarrow F \in X \) and, \( F \neq \phi. \) However, if \( X_m = g(A \in X) \) in monotone spaces, then \( A \subset X_m. \) This indicates, \( X_m \subset X. \) Again, by following monotone property it can be said that, \( F \subset A \Rightarrow g(F) \subset g(A). \) On the other hand, \( F \subset A \Rightarrow X_{mF} \subset X_m \) where, \( X_{mF} = g(F). \) Hence, considering the base case of \( F \subset A, \) the mapping can be established between simplex and monotone space such that, \( F \in X \subset X_{mF} \subset X_m. \) Thus, the monotone spaces can induce simplexes within topological spaces while constructing the model of distributed computation. There exists a connective mapping between the spaces in distributed computation.

7. **Conclusions**

The modeling and analyzing of properties of distributed computing by following the views of topological spaces are possible alternatives to the traditional approaches. The simplex structures of topological spaces in higher-dimensions are generally employed to model distributed computing. In general, the simplicial complexes are static structures in nature. However, monotone spaces are the general form of topological spaces. The concepts of monotone spaces can be effectively employed to determine consistency and convergence of a distributed computation. The interrelation and mapping between the simplex and monotone spaces can be formed. The monotone spaces facilitate the determination of consistent cuts of a distributed computation under closure property.
References


Author

Susmit Bagchi. He has received B.Sc.(Honours) from Calcutta University in 1993, B. E. (Electronics Engg.) from Nagpur University in 1997 and, M. E. (Electronics & Telecomm. Engg.) from IIEST (formerly, Bengal Engineering and Science University) in 1999. He has obtained Ph.D. (Engineering) in Information Technology from IIEST in 2008. Currently, he is Associate Professor in Department of Aerospace and Software Engineering, Gyeongsang National University, Republic of Korea. His main research interests are in Distributed Computing and Operating Systems.