*Journal of the Nigerian Association of Mathematical Physics*
*Volume35,(May, 2016)*, pp 57 – 64


# On Application of Soft Lattice to Distributed System

*A. M. Ibrahim[1] and A. O. Yusuf[2]*

[1]Department of Mathematics, Ahmadu Bello University, Zaria-Nigeria
[2]Department of Mathematical Sciences and Information TechnologyFederal University, Dutsin-Ma, Katsina, Nigeria

***Abstract***

***This paper crisply present the fundamentals of soft set theory to emphasize that soft set has enough developed basic supporting tools through which various algebraic structures in theoretical point of view could be developed. We introduced the concept of soft lattice theory and distributive soft lattice. Finally, we apply the soft lattice theory to distributed system.***

**Keywords:** Soft set, parameters, soft Lattice, soft sublattice and distributive soft lattice

## 1.0 Introduction

Let $U$ be a universal set and $E$ be the set of all possible parameters under consideration with respect to $U$. Let the power set of $U$ (i.e., the set of all subsets of $U$) be denoted by $P(U)$ and $A$ is a subset of the parameters, $E$ ($A\subseteq E$). The parameters are attributes, characteristics or properties associated with the objects in $U$. Then we have the followings which can be found in [1-12].

*Definition 1.1*

A pair $(F,E)$ is called a soft set over $U$ if and only if $F$ is a mapping of $E$ into the set of all subsets of the set $U$. That is, a soft set is a parametrized family of subsets of the set $U$. For all $e\in E$, $F(e)$ is considered as the set of $e-$approximate elements of the soft set $(F,E)$.

*Definition 1.2*

A soft set $(F,E)$ over a universe $U$ is said to be *null* or *empty* soft set denoted by$\widetilde{\emptyset}$, if $\forall e\in E, F(e)=\emptyset$.

*Definition 1.3*

A soft set $(F,A)$ over a universe $U$ is called *absolute* or *universal soft set* denoted by $\widetilde{(F,A)}$ or $\widetilde{U}$, if$\forall e\in E, F(e)=U$.

*Definition 1.4*

Let $E=\{e_1,e_2,e_3,\ \ldots,e_n\}$ be a set of parameters. The *not-set* of $E$ denoted by $\neg E$ is defined as $\neg E=\{\neg e_1,\ \neg e_2,\ \neg e_3,\ \ldots,\ \neg e_n\}$.

*Definition 1.5*

The *complement* of a soft set $(F,E)$, denoted by $(F,E)^c$, is defined as $(F,E)^c=(F^c,\neg E)$.

Where $F^c:\neg E\to P(U)$ is a mapping given by $F^c(\alpha)=U-F(\neg\alpha),\forall\alpha\in\neg E$. $F^c$ is called the soft complement function of $F$. Consequently, $(F^c)^c=F$ and $((F,E)^c)^c=(F,E)$

*Definition 1.6*

Let $(F,A)$ and $(G,B)$ be any two soft sets over a common universe $U$, $(F,A)$ is called a soft subset of $(G,B)$, denoted by $(F,A)\cong(G,B)$if ;

(i) $A\subset B$, and

(ii) $\forall e\in A, F(e)=G(e)$

$(F,A)$ is said to be a soft super set of $(G,B)$, if $(G,B)$ is a subset of $(F,A)$ and it is denoted by $(F,A)\supseteq(G,B)$.

*Definition*1.7

Two soft sets $(F,A)$ and $(G,B)$ over a common universe $U$ are said to be soft equal, denoted by $(F,A)=(G,B)$, if $(F,A)$ is a soft subset of $(G,B)$ and $(G,B)$ is a soft subset of $(F,A)$.

Corresponding author:A. M. Ibrahim, E-mail:amibrahim@abu.edu.ng, Tel.: +2348037032464& 7039057669(AOY)

*Definition 1.8*
If $(F,A)$ and $(G,B)$ are two soft sets then "$(F,A)\ AND\ (G,B)$" denoted by $(F,A)\wedge(G,B)$ is defined as$(F,A)\wedge(G,B)=(H,A\cap B)$, where $H(\alpha)=F(\alpha)\cap G(\alpha),\ \forall\alpha\in A\cap B$.
*Definition 1.9*
If $(F,A)$ and $(G,B)$ are two soft sets then "$(F,\ A)\ OR\ (G,B)$" denoted by $(F,A)\vee(G,B)$ is defined as $(F,\ A)\vee(G,B)=(P,\ A\cup B)$, where, $P(\alpha)=F(\alpha)\cup G(\alpha),\forall\alpha\in A\cup B$.
*Definition 1.10*
Let $(F,A)$ and $(G,B)$ be two soft sets over a common universe *U*. The union or extended union of $(F,A)$ and $(G,B)$, denoted by $(F,A)\cup(G,B)$ or $(F,A)\cup_E(G,B)$, is the soft set $(H,C)$ satisfying the following conditions:

(i) $C=A\cup B$, (ii) $\forall e\in C$, $\quad H(e)=\begin{cases}F(e) & \text{if } e\in A\backslash B\\ G(e) & \text{if } e\in B\backslash A\\ F(e)\cup G(e) & \text{if } e\in A\cap B\end{cases}$

*Definition 1.11*
The intersection of two soft sets $(F,A)$and $(G,B)$ over a common universe set $U$ is the soft set $(H,C)$, where $C=A\cap B$, and $\forall e\in C, H(e)=F(e)$or $G(e)$, we write $(F,A)\cap(G,B)=(H,C)$
Let $(F,A)$and$(G,B)$ be two soft sets over a common universe $U$ such that$A\cap B\neq\emptyset$ . The restricted difference of $(F,A)$and $(G,B)$ denoted by $(F,A)\sim_R(G,B)$, is defined as $(F,A)\sim_R(G,B)=(H,C)$, where $C=A\cap B$, and $\forall e\in C, H(e)=F(e)\setminus G(e)$.
Various properties of these operations and algebraic structures defined on soft sets could be found elsewhere [2, 4, 9, 10, 11]

## 2.0 Distributive Soft Lattices

There are several studied on Distributive lattices in standard or classical setting. Here we look at the concept of Distributive lattices based on soft set theory.
*Definition 2.1*
Let $(\Gamma,E)$ be a soft set. Let $A,B,C\subseteq E$ such that $(F,A),(G,B)$ and $(H,C)$ are all defined. Then $(\Gamma,E)$ together with the binary operations $\vee$ and $\wedge$ is called soft lattice if the following axioms are satisfied:

L1: (a) $(F,A)\vee(G,B)=(G,B)\vee(F,A)$
(b) $(F,A)\wedge(G,B)=(G,B)\wedge(F,A)$(Commutative laws)

L2: (a) $(F,A)\vee\big((G,B)\vee(H,C)\big)=\big((F,A)\vee(G,B)\big)\vee(H,C)$
(b) $(F,A)\vee\big((G,B)\vee(H,C)\big)=\big((F,A)\vee(G,B)\big)\vee(H,C)$ (Associative laws)

L3: (a) $(F,A)\vee(F,A)=(F,A)$
(b) $(F,A)\wedge(F,A)=(F,A)$ (Idempotent laws)

L4: (a) $(F,A)=(F,A)\vee\big((F,A)\wedge(G,B)\big)$
(b) $(F,A)=(F,A)\wedge\big((F,A)\vee(G,B)\big)$, (Absorption laws)

We denote the soft lattice $(\Gamma,E)$by $L(\Gamma,E)$. For convenience we simply write *L*. Where $\vee$ and $\wedge$ are as defined in *Definition 1.8* and *Definition 1.9.*
*Definition 2.2*
Let $(\Gamma,\mathrm{E})$ be a soft lattice. Let $A,B,C\subseteq E$ such that $(\mathrm{F},\mathrm{A})$, $(\mathrm{G},\mathrm{B})$,and $(\mathrm{H},\mathrm{C})$ are all defined. Then $(\Gamma,\mathrm{E})$ together with the binary operations $\vee$ and $\wedge$ is called distributive soft lattice if the following axioms are satisfied:

D1: $(\mathrm{F},\mathrm{A})\wedge\big((\mathrm{G},\mathrm{B})\vee(\mathrm{H},\mathrm{C})\big)=\big((\mathrm{F},\mathrm{A})\wedge(\mathrm{G},\mathrm{B})\big)\vee\big((\mathrm{F},\mathrm{A})\wedge(\mathrm{H},\mathrm{C})\big)$
D1: $(\mathrm{F},\mathrm{A})\vee\big((\mathrm{G},\mathrm{B})\wedge(\mathrm{H},\mathrm{C})\big)=\big((\mathrm{F},\mathrm{A})\vee(\mathrm{G},\mathrm{B})\big)\wedge\big((\mathrm{F},\mathrm{A})\vee(\mathrm{H},\mathrm{C})\big)$

*Definition 2.3*
If $(\Gamma,E)$ is a soft lattice and $(F,A)\neq\emptyset$ is a soft subset of $(\Gamma,E)$ such that $\forall\mathrm{F}(e_1),\mathrm{F}(e_2)\in(F,A)$ both $\mathrm{F}(e_1)\vee\mathrm{F}(e_2)$ and $\mathrm{F}(e_1)\wedge\mathrm{F}(e_2)$ are in $(F,A)$, where $\vee$ and $\wedge$ are the soft lattice operations of $(\Gamma,E)$. Then we say that $(F,A)$ with the same operations (restricted to $(F,A)$) is a soft sublattice of $(\Gamma,E)$.
*Definition 2.4*
A soft set $(\Gamma,\mathrm{E})$ is called an ordered soft set if the parameter E is ordered.
*Remark 2.1*
If $(\Gamma,E)$ is order soft set then for $A,B,C\subseteq E$, $(F,A),(G,B),(H,C)$ are all order soft sets.
*Definition 2.5*
A binary relation $\sqsubseteq$ defined on the set of parameters $E$ is a partial order on $E$ if for every $A,B,C\subseteq E,(F,A),(G,B),(H,C)$ are defined such that the following axioms are satisfied
(i)$(F,A)\ \widetilde{\sqsubseteq}\ (F,A)$ (Reflexivity)

(ii)$(F,A) \cong (G,B)$ and $(G,B) \cong (F,A) \Rightarrow (F,A) = (G,B)$ (Antisymmetry)
(iii)$(F,A) \cong (G,B)$ and $(G,B) \cong (H,C) \Rightarrow (F,A) \cong (H,C)$ (Transitivity)
If, in addition, for every $A, B \subseteq E$
(iv) )$(F,A) \cong (G,B)$ or $(G,B) \cong (F,A)$, then we say $\cong$ is a total order on $E$.
A non- empty soft set $(\Gamma, E)$ with a partial order on it is called *partially orderedsoft set* denoted as $\big((\Gamma,E),\cong\big)$. If the relation is total order then we say that $\big((\Gamma,E),\cong\big)$ is called *totally orderedsoft set.*

## 3.0 Soft lattice Distributed Computing

In this paper, we have surveyed various application of lattice theory to distributed computing system by different researchers[14-20]. If the element of the underline set has attributes, then the classical set theory will be unable to tackle such problems.
The following distributed computing problems can be solved by soft lattice theory:
(i) Detecting a predicate in a computation, that is determining whether there exists a *consistent cu*t of the computation satisfying the given predicate.
(ii) Computing the slice of a computation with respect to a predicate. A slice is a concise representation of all those global states of the computation that satisfy the given predicate.
(iii) Analyzing a partial order trace of a distributed program to determine whether it satisfies the given temporal logic.
(iv) Timestamping events and global states of a computation to capture the order relationship. We study how the results from soft lattice theory can be used in solving some of the above problems.

## 3.1 Detecting Global Predicate

**Definition 3.1.1**
Let $(\Gamma, E)$ be the soft set of events of a computation and → be the happened-before order on $(\Gamma, E)$. A soft subset $(\mathrm{G}, D)$ of $(\Gamma, E)$ is a consistent cut if whenever it contains an element $G(f)$ then it contains all elements $G(e)$ that happened -before $G(f)$.
This concept is identical to the notion of order ideal in the soft lattice theory.
**Definition 3.1.2**
A predicate is simply a soft Boolean function from the soft set of all *consistent cuts* to $\{0,1\}$. Equivalently, a predicate specifies a soft subset of consistent cuts in which the soft Boolean function evaluate to 1.
We now define various classes of predicates.
**Definition 3.1.3 Meet-closed predicates**
A predicate B is meet-closed if for all consistent cuts $(G,D)$, $(H,C)$:
$B\big((G,D)\big) \wedge B\big((H,C)\big) \Rightarrow B\big((G,D) \wedge (H,C)\big)$.
The classes of meet-closed predicate are useful because they allow us to compute the least consistent cut that satisfies a given predicate.
**Definition 3.1.4 Crucial Element**
For a consistent cut $(G,D) \subsetneq (\Gamma, E)$ and a predicate B, we define $G(d) \in (\Gamma, E) \sim (G,D)$ to be crucial for $(G,D)$ as:
Crucial$\big((G,D), G(d), B\big) = \forall (H,C) \supseteq (G,D)$: $B\big(G(d) \in (H,C)\big) \vee \neg B\big((H,C)\big)$.
**Definition 3.1.5Linear predicate**
A predicate B is linear if for all consistent cuts $(G,D) \subsetneq (\Gamma, E)$,
$\neg B\big((G,D)\big) \rightarrow \exists G(d) \in (\Gamma, E) \sim (G,D)$: Crucial$\big((G,D), G(d), B\big)$.
*Theorem 3.1.1*
A predicate B is linear if and only if it is meet-closed.
*Proof*
First, assume that B is not closed under meet. We show that B is not linear. Since B is not closed under meets, there exist two consistent cuts $(H,C)$, $(K,A)$ such that $B\big((H,C)\big)$ and $B\big((K,A)\big)$ but not $B\big((H,C) \cap (K,A)\big)$. Define $(G,D)$ to be $(H,C) \cap (K,A)$ is a strict soft subset of $(H,C) \subseteq (\Gamma, E)$ because $B\big((H,C)\big)$ but not $B\big((G,D)\big)$. Therefore, $(G,D)$ cannot be equal to $(\Gamma, E)$. We show that B is not linear by showing that there does not exist any crucial element $G(d)$ for $(G,D)$. A crucial element $G(d)$, if it exists, cannot be in $(\Gamma, E) \sim (G,D)$ because $(K,A)$ does not contain $G(d)$ and still $B\big((K,A)\big)$ holds.
Similarly, it cannot be in $(\mathrm{K}, A) \sim (G,D)$ because $(H,C)$ does not contain $G(d)$ and still $B\big((H,C)\big)$ holds. It also cannot be in $(\Gamma, E) \sim \big((H,C) \cup (K,A)\big)$ because of the same reason. We conclude that there does not exist any crucial event for $(G,D)$.
Now assume that B is not linear. This implies that there exists $(G,D) \subsetneq (\Gamma, E)$ such that $\neg B\big((G,D)\big)$ and none of the elements in $(\Gamma, E) \sim (G,D)$ is crucial. We first claim that $(\Gamma, E) \sim (G,D)$ cannot be a singleton. Assume if possible $(\Gamma, E) \sim (G,D)$ contains only one element $G(d)$. Then, any consistent cut $(H,C)$ that contain $(G,D)$ and does not contain $G(d)$ must

be equal to $(G,D)$ itself. This implies that $\neg B\big((H,C)\big)$ because we assumed $\neg B\big((G,D)\big)$. Therefore, $G(d)$ is crucial contradicting our assumption that none of the elements in $(\Gamma,E) \sim (G,D)$ is crucial.

Let $(\mathrm{W},T) = (\Gamma,E) \sim (G,D)$. For each $G(d) \in (\mathrm{W},T)$, we define $(\mathrm{H},C)_{G(d)}$ as the consistent cut that contains $(G,D)$, does not contain $G(d)$ and still satisfies B. It is easy to see that $(G,D)$ is the meet of all $(\mathrm{H},C)_{G(d)}$. Therefore, B is not meet-closed because all $(\mathrm{H},C)_{G(d)}$ satisfy B, but not their meets.

*Example 3.1.1*

Consider the soft Boolean lattice generated by all soft subset of $\{1,\dots,n\}$. Let the predicate B defined to be true on a consistent cut $(G,D)$ as if $(G,D)$ contains any odd $i < n$, then it also contains $i+1$. It is easy to verify that B is meet-closed. Given any $(G,D)$ for which B does not holds, the crucial elements consist of $\{i/i$ is even, $2 \le i \le n, i-1 \in (G,D), i \notin (G,D)\}$.

*Example 3.1.2*

Consider a distributed computation on two processes $P_1$ and $P_2$ and the predicate B to be true on a consistent cut if both the processes are in the critical section. Given any consistent cut $(G,D)$ for which B does not hold, either $P_1$ is not in the critical section, or $P_2$ is not in the critical section. In the former case, the next event of $P_1$ after $(G,D)$, entering the critical section is crucial and in the latter case the event of $P_2$ entering the critical section is crucial. This example can be easily generalized to any global Soft Boolean predicate that can be expressed as a conjunction of local predicates.

*Theorem 3.1.2*

If B is a linear predicate with the efficient advancement property, then there exists an efficient algorithm to determine the least consistent cut that satisfies B (if any).

*Proof*

An efficient algorithm to find the least cut in which B is true is given in *Figure 1*. We search for the least consistent cut starting from the empty consistent cut. If the predicate is false in the consistent cut, then we find the crucial element using the efficient advancement property and then repeat the procedure. If this is the least on the process, then we return false else we advance along the process that has the crucial event.

**Algorithm to detect a linear predicate**.

Soft Boolean function detect (B: Soft Boolean-Predicates, P: Poset).

Var

$(G,D)$; Consistent cut initially $(G,D) \coloneqq \{\}$;

while $\Big(\neg B\big((G,D)\big) \wedge \big((G,D) \neq P\big)\Big)$ do

Let $G(d)$ be such that crucial $\big((G,D),G(d),B\big)$ in $P$;

$(G,D) \coloneqq (G,D) \cup \{G(d)\}$.

End while;

If $B\big((G,D)\big)$ return true;

Else return false;

**Figure 1**: An efficient algorithm to detect a linear predicate.

Assuming that crucial $\big((G,D),G(d),B\big)$ can be evaluate efficiently for a given poset, we can determine the least consistent cut that satisfies B efficiently even though the number of consistent cuts may be exponentially larger than the size of the poset. If the predicate B is join-closed, then one can search for the largest consistent cut that satisfies B in a fashion analogous to finding the least consistent cut when it is meet-closed.

*Definition 3.1.6*(Regular predicate)

A predicate is regular if the soft set of consistent cuts that satisfy the predicate form a soft sublattice of the soft lattice of consistent cuts, Equivalently, a predicate B is regular with respect to P if it is closed under ∩ and ∪, i.e., for all consistent cuts $(G,D)$, $(H,C)$ of the poset P:

$B\big((G,D)\big) \wedge B\big((H,C)\big) \Rightarrow B\big((G,D) \cup (H,C)\big) \wedge B\big((G,D) \cap (H,C)\big)$.

The set of consistent cuts that satisfy a regular predicates forms a soft sublattice of the soft lattice of all consistent cuts.

*Example 3.1.3*

Consider the predicate B as "there is no outstanding message in the channel". We show that this predicate is regular. Observe that B holds on a consistent cut $(G,D)$ if and only if for all send events in $(G,D)$ the corresponding receive events are also in $(G,D)$. It is easy to see that if $B\big((G,D)\big)$ and $B\big((H,C)\big)$, then $B\big((G,D) \cup (H,C)\big)$. To see that it holds for $(G,D) \cap (H,C)$, let $G(d)$ be any send event in $(G,D) \cap (H,C)$, let $H(d)$ be the receive event corresponding to $G(d)$.

From $B\big((G,D)\big)$, we get that $H(d) \in (G,D)$ and from $B\big((H,C)\big)$, we get that $H(d) \in (H,C)$. Thus, $H(d) \in (G,D) \cap (H,C)$. Hence, $B\big((G,D) \cap (H,C)\big)$.

## 3.2 Slicing Distributed Computing

Suppose we are only interested in a soft subset of consistent cuts of a computation but not all consistent cuts of a computation, namely those that satisfy some property of interest to us expressed as a predicate mapping a consistent cut to a Boolean valued.

A soft sublattice of a distributive soft lattice is also a distributive soft lattice. Therefore, the soft sublattice generated by the consistent cuts satisfying the predicate is completely characterized by the joint-irreducible elements of the soft sublattice.

*Example 3.2.1*

The distributed computation shown in Figure **2** consists of two processes $P_1$ and $P_2$. Process $P_1$ executes events **a** and **b**, whereas process $P_2$ executes events **c** and **d**. On executing **b**, $P_1$ sends a message to $P_2$, which is received by $P_2$ at **d.** The set of consistent cuts of the computation are shown in Figure **2.**

Suppose we are interested in only those consistent cuts for which no messages are in transit, also known as *strongly consistent cuts*. They have been shaded in Figure **3** and are shown separately in Figure **4.** The set of strongly consistent cuts forms a soft sublattice and its join-irreducible element has been drawn with thick boundaries. The poset induced on the set of join-irreducible elements of the soft sublattice is shown in Figure **5**

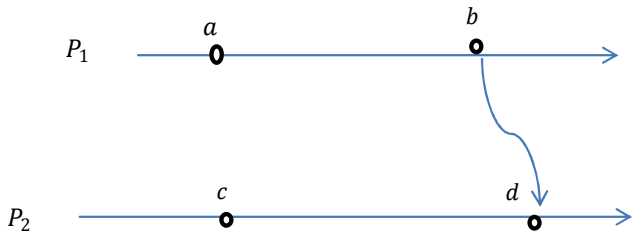


**Figure 2** A Distributed Computation

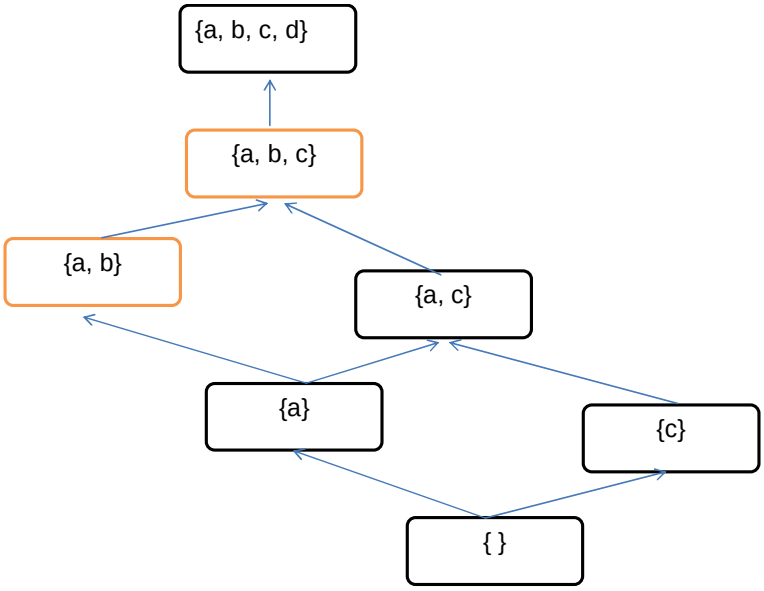


**Figure 3**The distributive soft lattice generated by its consistent cuts

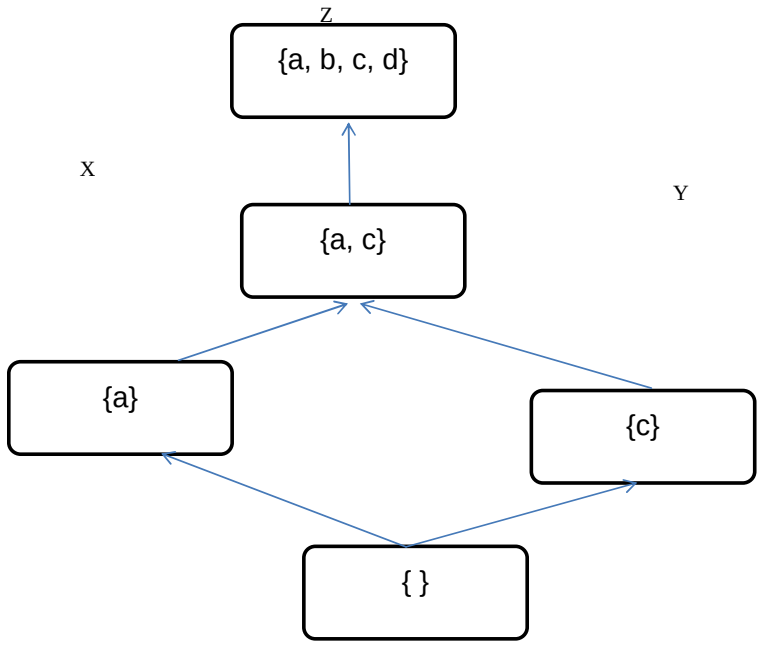


**Figure 4**The soft sublattice containing all consistent cuts for which no messages are in transit.

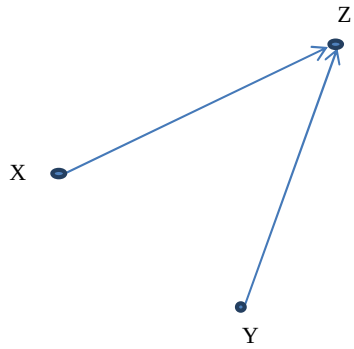


**Figure 5** The poset induced on the set of join-irreducible elements of the soft sublattice.

In case the set of consistent cuts that satisfy the predicate does not form a soft sublattice, we add one or more other consistent cuts, which do not satisfy the predicate, to complete the soft sublattice. The consistent cuts are added in such a way so as to minimize the total number of consistent cuts in the resulting soft sublattice. The soft sublattice is then represented using the set of its join-irreducible elements. This succinct representation of a possible large set of consistent cuts satisfying some property is referred to as a slice.

*Theorem 3.2.1*

The slice of a distributed computation is uniquely defined for all predicates.

*Proof*

Let D denote the set of all consistent cuts that satisfy the predicate. We show that the soft sublattice with the least number consistent cuts that satisfy D is uniquely defined. Assume the contrary. Let $X$ and $Y$ be two distinct soft Sublattices with the least number of consistent cuts such that:

(1) $Cardinality(X) = Cardinality(Y)$, and

(2) both $X$ and $Y$ contain $D$.

Consider $Z = X \cap Y$ . Clearly, $Z$ also contain $D$ . Also, since $X \neq Y$ , $Cardinality(Z) < Cardinality(X)$ and $Cardinality(Z) < Cardinality(Y)$. It can be proved that intersection of two soft sublattices is also a soft sublattice. This implies that $Z$ is a soft sublattice that contain $D$ and has fewer number of consistent cuts than either $X$ or $Y$, a contradiction.

*Remark 3.2.1*

The slice for a predicate may contain consistent cuts that do not satisfy the predicate, namely that are added to complete the soft sublattice. A slice is *lean* if it contains only those consistent cuts that satisfy the predicate. Clearly, the slice of a computation for a predicate is *lean* if and only if the predicate is regular.

Another way of looking at slice is that it specifies which events should be executed in an atomic fashion and the order in which they should be executed. For example, the slice shown in Figure **5** and redrawn in Figure **6** specifies that events **b** and **d** should be executed atomically after events **a** and **c** have been executed. This is expected because any consistent cut which includes the send event of a message but not its receive will have at least one message in transit.

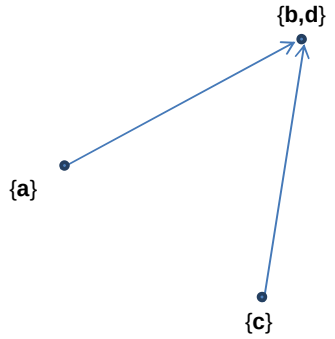


**Figure 6:** Aslice that depict the events that are to executed atomically

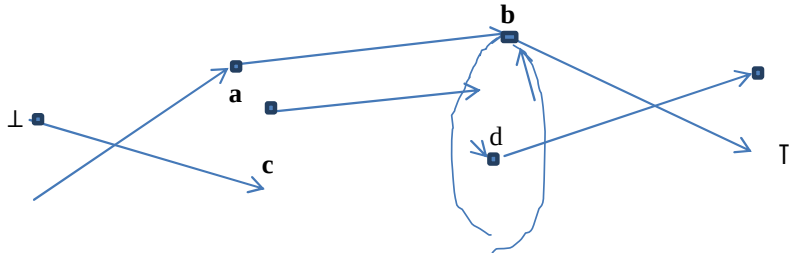


**Figure 7:** The graph representation of the slice in Figure 6

For algorithmic purposes, it is more convenient to represent a slice using a directed graph on events possibly containing cycles: all events that are to be executed atomically form a strongly connected component. The notion of consistent cut has to be extended appropriately.

We define a consistent cut (global state) on directed graph as a subset of vertices such that if the subset contains a vertex then it contains all its incoming neigbours. Observe that the empty ∅ and the set of all vertices are trivial consistent cuts.

We introduce a fictitious *global initial* and *global final* event, denoted by ⊥and ⊺**,** respectively.

The global initial event occurs before any other event on the processes and initializes the state of the processes. The global final event occurs after all other events on the processes. Any non-trivial consistent cut will contain the global initial event and not the global final event.

## 4.0 Conclusion

In this paper wepresent the fundamentals of soft set theory. We introduced the concept of soft lattice, soft sublattice theory and distributive soft lattice. Finally, we focused our discussion on the application of soft lattices to detecting a predicate in a computation and computing the slice of a computation with respect to a predicate.

## 5.0 References

[1] A. M. Ibrahim, M. K. Dauda and D. Singh. Composition of Soft Set Relations and Construction of Transitive Closure, *Journal of Mathematical Theory and Modeling*, Vol. 2, No. 7, pp. 98 – 107, 2012.

[2] A. Kharal and B. Ahmad. Mappings on Soft Classes, *Information Sciences, INS-D-08-1231 by ESS*, pp. 1 – 11, 2010.

[3] A. O. Atagun and A. Sezgin. Soft Substructures of Rings, Fields and Modules, *Computers and Mathematics with Applications*, Vol. 61, pp. 592 – 601, 2011.

[4] A. Sezgin and A. O. Atagun. On operation of Soft Sets, *Computers and Mathematics with Applications* Vol. 61, 1457 – 1467, 2011.
[5] D. A. Molodtsov. Soft Set Theory- First Results, *Computers and Mathematics with Applications* 37 (4/5), 19 – 31, 1999.
[6] S.V. Manemaran. On Fuzzy Soft Groups, *International Journal of ComputerApplications* Vol. 15 No. 7, 2011.
[7] D. K. Sut. An Application of Fuzzy Soft Relation in Decision Making Problems,*International Journal of Math. Trends and Tech.* Vol. 3 No. 2, 50 – 53, 2012.
[8] F. Feng, Y. B. Jun, and X. Zhao. Soft semirings, *Computers and Mathematics with Applications* Vol. 56, 2621 – 2628, 2008.
[9] H. Aktas and N. Cagman, Soft Sets and Soft Groups, *Information Sciences*, 177,2726 – 2735, 2007.
[10] K. Gong, Z. Xiao and X. Zgang. The Bijective Soft Set with its operations, *Computers and Mathematics with Applications* Vol. 60, 2270 – 2278, 2010.
[11] K. Qin and Z. Hong, on Soft Equality, *Journal of Computer and Applied Mathematics* 234,1347 – 1355 2010.
[12] K.V. Babitha and J. J. Sunil. Soft Sets Relations and Functions, *Computers and Mathematics with Applications* 60, 1840 – 1849 2010.
[13] A. M. Ibrahim and A. O Yusuf, On Soft Lattice Theory. Journal of the Nigerian Association of Mathematical Physics, Vol. 31, 263-270, 2015.
[14] Lamport. L. (1978). Time, Clocks and the ordering of Events in a Distributed system. *Communications of the ACM (CACM)*,21(7): 558-565.
[15] Chandy, K.M., and Lamport, L. (1985). Distributed Snapshots: Determining Global States of Distributed Systems. *ACM Transaction on computer systems* 3(1): 63-75.
[16] Mattern, F. (1989). Virtual Time and Global States of Distributed Systems. *In parallel and Distributed Algorithms (WDAG),* 215-226.
[17] Charron-Bost, B. (1991). Concerning the size Logical Clocks in Distributed systems, *Information processing Letters (IPL)*, 39: 11-16.
[18] Chase, C. and Garg, V. K. (1995). On Techniques and their Limitations for the Global Predicate Detection problem, *In proceeding of the Workshop on Distributed Algorithms (DAG)*, 303-317.
[19] Garg, V. K. and Skawrananond, C. (2001). String Realizers of posets with Applications to Distributed to computing. *In proceeding of the 20th ACM Symposium on principles of Distributed computing (PODD),* 72-80.
[20] Sen, A. and Garg, V. K. (2002). Automatic Generation of Computation Slice for Detecting Temporal Logic Predicates, *Technical Report TR-PDS 2002-001. The parallel and Distributed Systems, Laboratory, Department of Electrical and computer Engineering*, The University of Texas at Austin