

# Formal proofs related to incremental Merkle trees

Mizuhito Ogawa

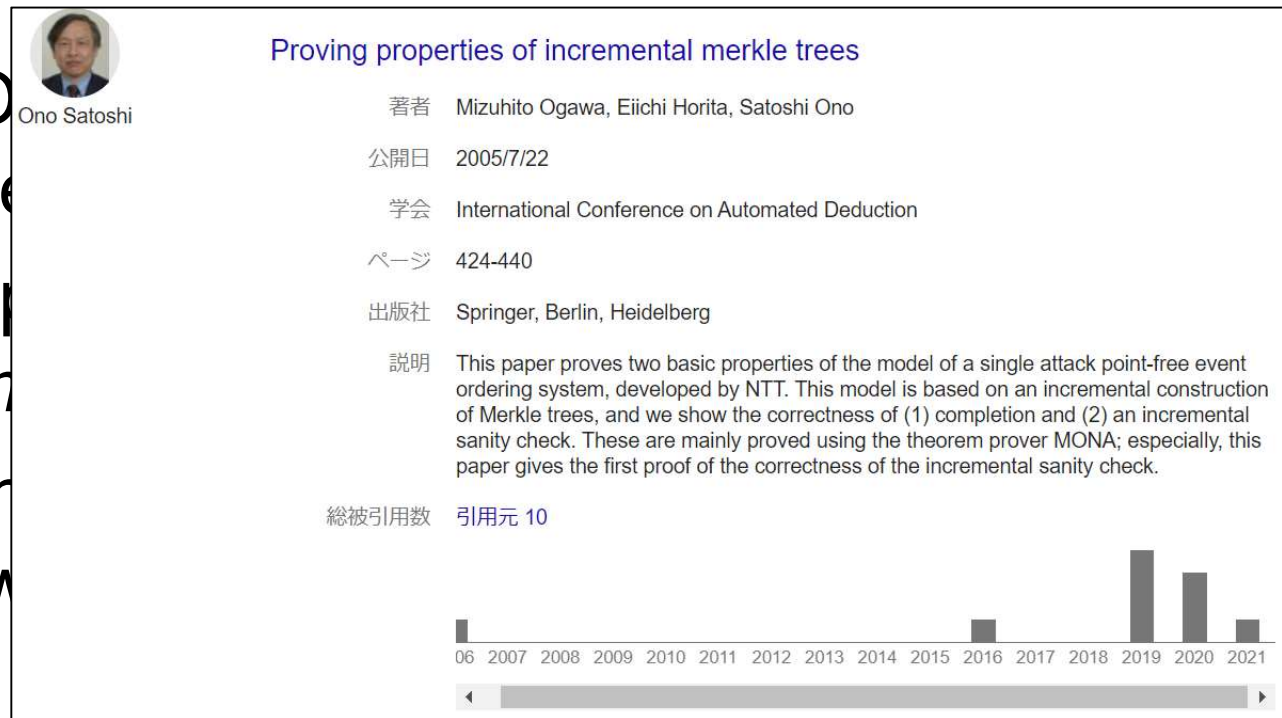
[www.jaist.ac.jp/~mizuhito](http://www.jaist.ac.jp/~mizuhito)

TPP2021 北見工大

(originally in CADE 2005,  
*“Proving properties of incremental Merkle trees”*)

# What we have shown in CADE 2005?

- M Ogawa, S Ono  
incremental Merkle trees
  - ✓ NTT developed  
like *block chain*
  - ✓ Proved “Sanity check”  
by MONA (with



Proving properties of incremental merkle trees

著者 Mizuhito Ogawa, Eiichi Horita, Satoshi Ono

公開日 2005/7/22

学会 International Conference on Automated Deduction

ページ 424-440

出版社 Springer, Berlin, Heidelberg

説明 This paper proves two basic properties of the model of a single attack point-free event ordering system, developed by NTT. This model is based on an incremental construction of Merkle trees, and we show the correctness of (1) completion and (2) an incremental sanity check. These are mainly proved using the theorem prover MONA; especially, this paper gives the first proof of the correctness of the incremental sanity check.

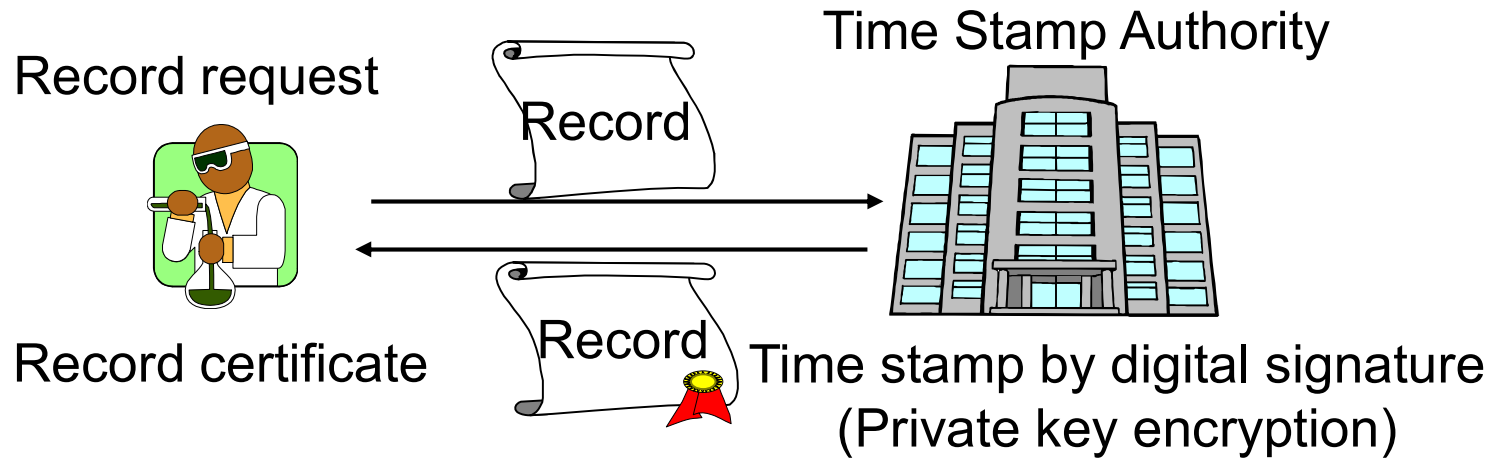
総被引用数 引用元 10

Year	Citations
2006	1
2007	0
2008	0
2009	0
2010	0
2011	0
2012	0
2013	0
2014	0
2015	0
2016	1
2017	0
2018	0
2019	4
2020	3
2021	1

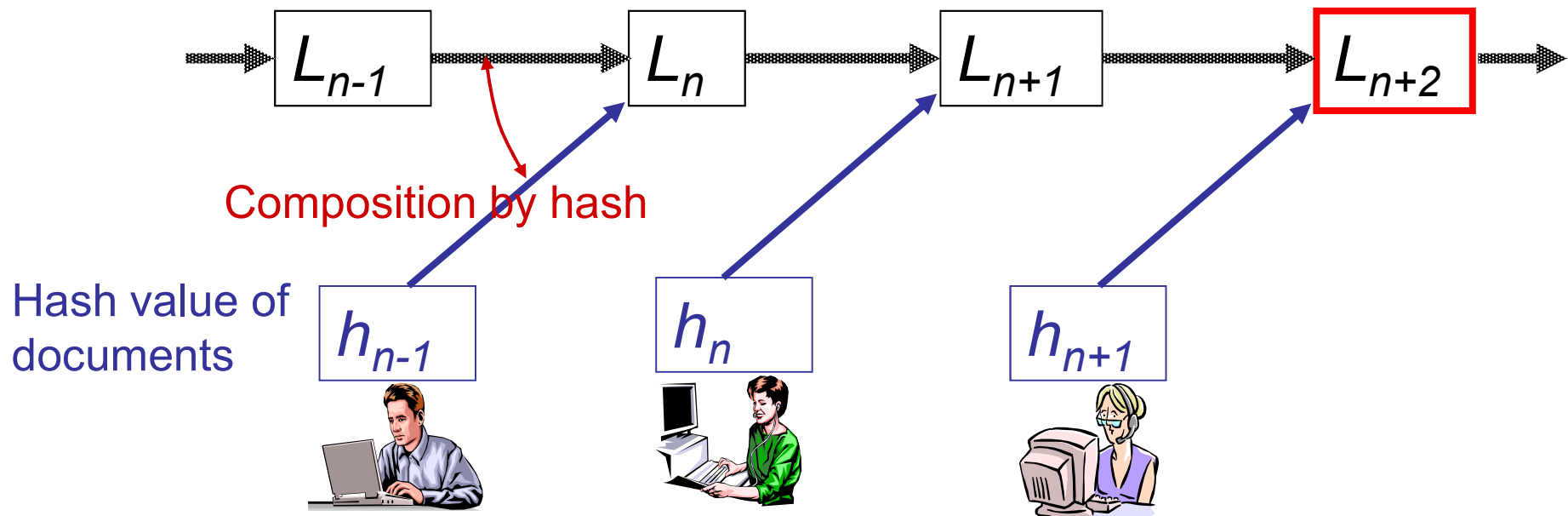
- Considering recent citation in formal proofs,  
(ESORICS 2016, ITP 2019/2020, ArXiv 2021, ...)
  - ✓ Complete formal proof in *Isabelle sledgehammer*?
  - ✓ How to formalize properties of *Hash function*?
  - ✓ How to formalize consensus? (e.g., majority)

# Two methods for event-ordering certificate

## Time stamp by digital signature (rfc-3161)

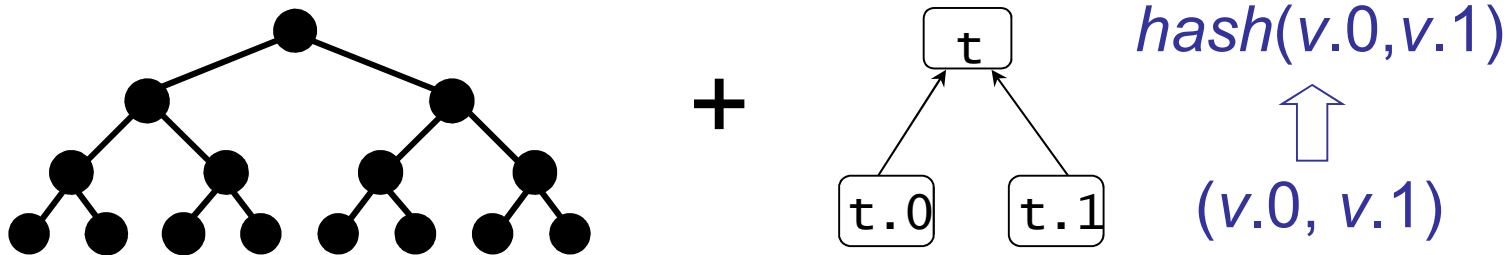


## Linking and publication by hash function (ISO18014-3)



# Merkle tree (Merkle 1979)

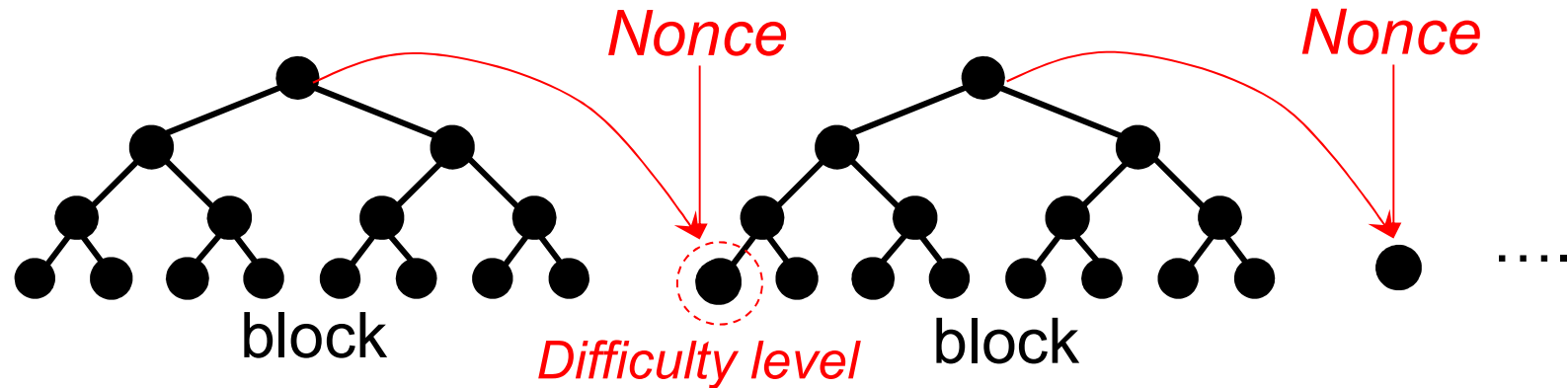
- *Merkle tree = Binary tree + hash function*
- Each node has its hash value, computed from a pair of hash values of its children.



We assume collision-resistance, one-way hash function.

Block chain = IML + *finding nonce* + DLT

- Incremental Merkle Tree (IML)



- Finding *nonce* (mining)

- ✓ Nonce such that its hash value holds *difficulty level*.

- DLT = distributed ledger technology

- ✓ Majority decision.

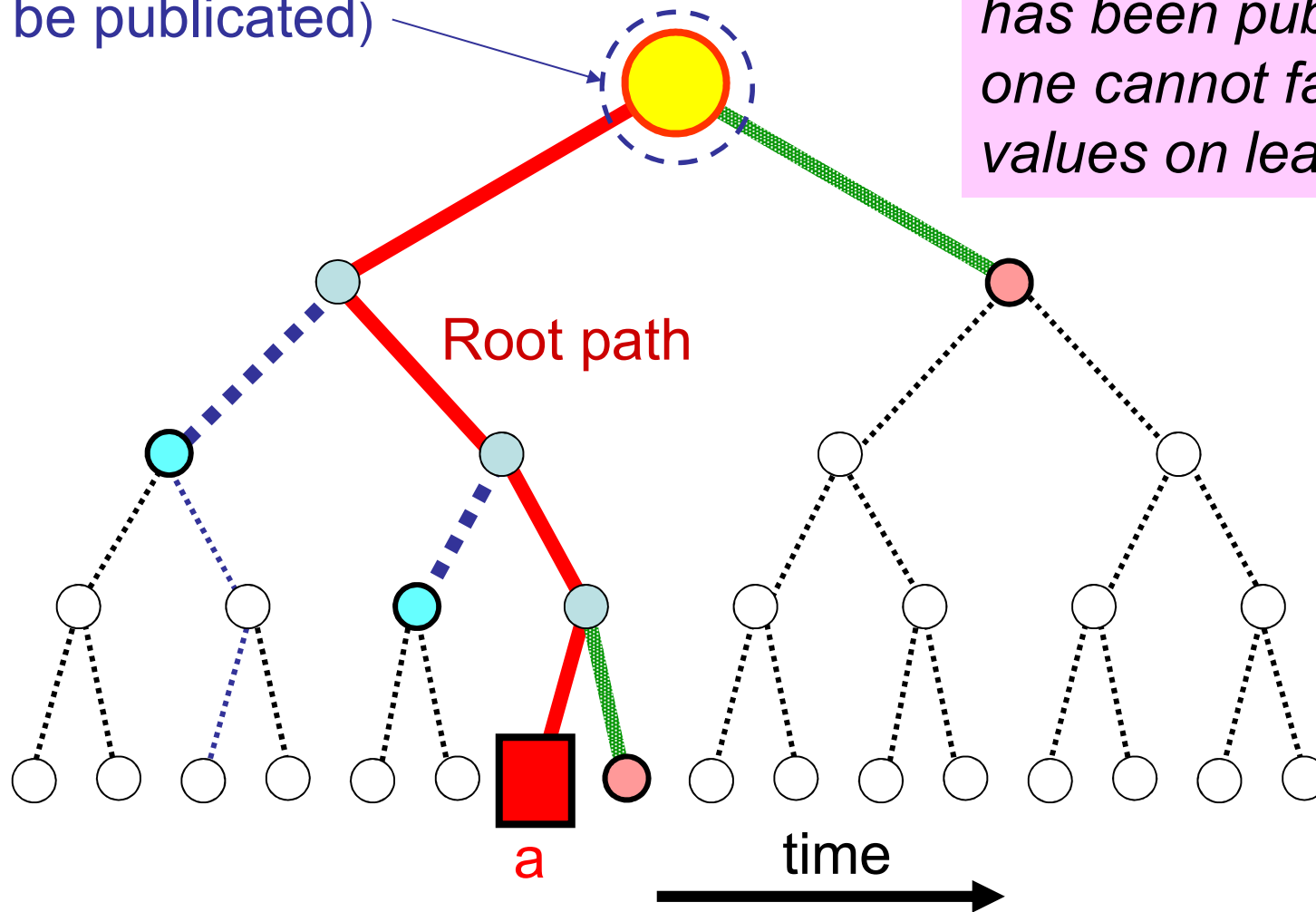
- ✓ Longer block chain is “*more*” valid.

# Basic idea : Merkle tree (1)

*Public witness*  
(to be publicated)

**Root hash value**

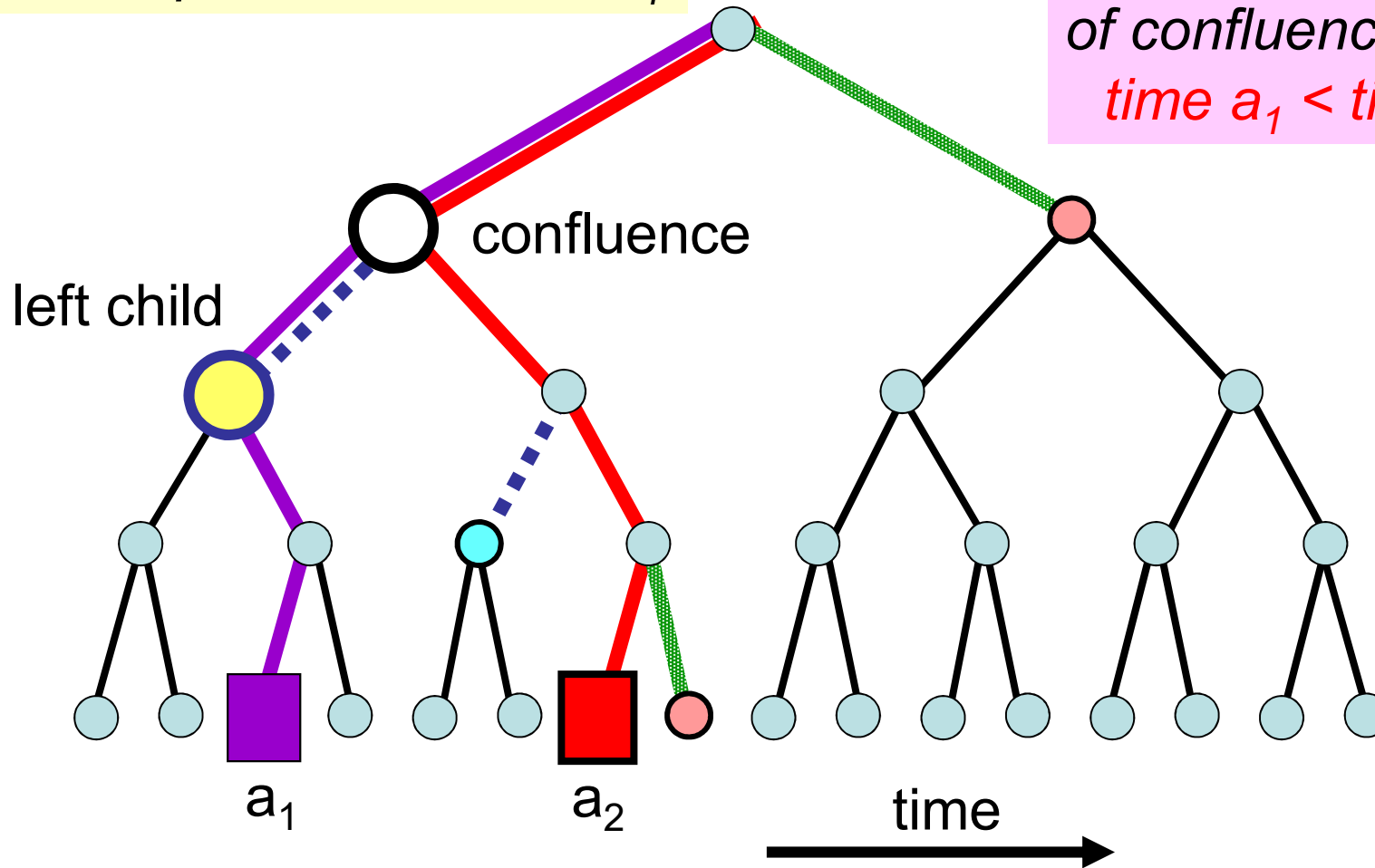
*Once root hash value has been publicated, one cannot falsify hash values on leaves.*



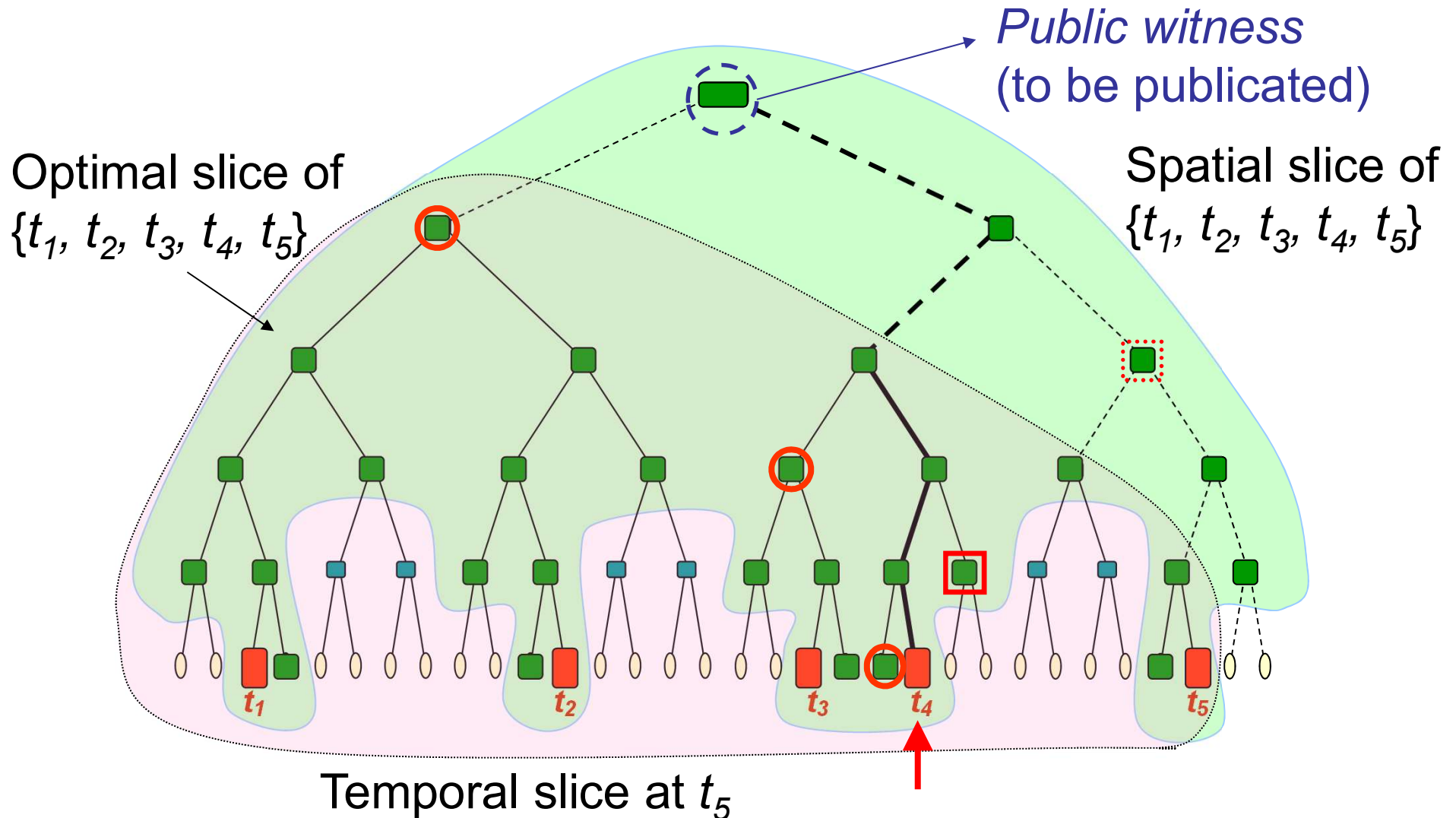
# Basic idea: Merkle tree (2)

The hash value of the left child depends on that of  $a_1$ .

If the root path of  $a_1$  contains the left child of confluence with  $a_2$   
time  $a_1 < \text{time } a_2$



# Incremental Merkle trees construction for registration requests at $t_1, t_2, t_3, t_4, t_5$



Temporal slice at  $t_5$

○  $LA(t_4)$  : left authentication path at  $t_4$

□  $RA_{t_5}(t_4)$  : (relative) right authentication path at  $t_4$  until  $t_5$



# Protocol for event registrations

- Assume a user registers at  $t_1, t_2, \dots, t_n$  and receives:
  - ✓  $(\phi, LA(t_1) \cup \{t_1\})$  at  $t_1$ .
  - ✓  $(RA_{t_i}(t_{i-1}), LA(t_i) \cup \{t_i\})$  at  $t_i$  with  $0 < i \leq n$ .

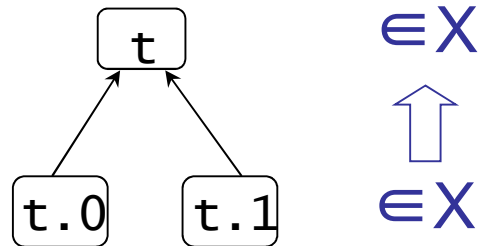
where  $LA(t)$  is the set of left authentications, and  $RA(t)$  is the set of right authentications.

*Described in WS2S*

**(incomparable(A) & opt\_slice(A,X) & LSRclosure\_union(A,Y)) => X = Y;**

- **Th.**  $OptimalSlice(\{t_1, t_2, \dots, t_n\})$   
 $= (\bigcup_{1 \leq i < n} Cls(LSR_{t_{i+1}}(t_i))) \cup Cls(LS(t_n))$   
 where  $LS(t_i) = LA(t_i) \cup \{t_i\}$ ,  $LSR_{t_{i+1}}(t_i) = LS(t_i) \cup RA_{t_{i+1}}(t_i)$ ,  
*Closure*  $Cls(X)$  is the minimum set with  $X \subseteq Cls(X)$  and  
 ✓ t.0 (left child), t.1 (right child)  $\in Cls(X) \Rightarrow t \in Cls(X)$

# Recall: MONA example on closure



```
MeadowNT.exe@CALIBAN
Buffers Files Tools Edit Search Mule Help
ws2s;

var2 X,Y,Z;
var1 s,t,u;

pred preclosure(var2 X,Y) = X sub Y &
  (all1 t: ((t.0 in Y & t.1 in Y) => t in Y));

pred closure(var2 X,Y) = preclosure(X,Y) &
  (all2 Z : preclosure(X,Z) => Y sub Z);

(closure(X,Y) & closure(Y,Z)) => closure(X,Z);
[--]S:** example.mona (MONA Encoded-kbd)-
```

```
~/papers/ono2/mona
$ mona example.mona
MONA v1.4-5 for WS1S/WS2S
Copyright (C) 1997-2002 BRICS

PARSING
Time: 00:00:00.05

CODE GENERATION
DAG hits: 34, nodes: 34
Time: 00:00:00.03

REDUCTION
Projections removed: 0 (of 4)
Products removed: 1 (of 16)
Other nodes removed: 0 (of 13)
DAG nodes after reduction: 32
Time: 00:00:00.02

AUTOMATON CONSTRUCTION
100% completed

Time: 00:00:00.22

Automaton has 12 states and 36 BDD nodes

ANALYSIS
Formula is valid
```

# Definition of closure

```
Isabelle/Isar Proof General: IndSetDefExample.thy
File Edit Options Buffers Tools Isabelle/Isar Proof-General Help
datatype bit = Zero ("\<zero>") | One ("\<one>")
-
consts bitval :: "bit => int"
primrec "bitval \<zero> = 0"
       "bitval \<one> = 1"
-
consts CIs :: "(bit list) set => (bit list) set"
inductive "CIs bvset"
intros
CIs base: "bv : bvset ==> bv : CIs bvset"
CIs step: "[| (bv @ (Zero # [])) : CIs bvset; (bv @ (One # [])) : CIs bvset |]
           ==> bv : CIs bvset"
--\** IndSetDefExample.thy (Isar script Scripting)--L46--28%-----
GC #0.0.0.1.18.1090: (0 ms)
Proofs for inductive set(s) "CIs"
  Proving monotonicity ...□
-S:-- *isabelle/isar-response* (response)--L3--All-----
```

# Example lemma: closure is idempotent

```
Isabelle/Isar Proof General: IndSetDefExample.thy
File Edit Options Buffers Tools Isabelle/Isar Proof-General Help

lemma Cls_idempotent_left :
  "Cls(Cls(X)) <= Cls(X)"
  apply(clarify) apply(erule_tac bvset = "Cls X" in Cls.induct) apply(simp)
  apply(drule_tac bvset = "X" in Cls_step) apply(auto)
done

lemma Cls_idempotent :
  "Cls(Cls(X)) = Cls(X)"
  apply(insert Cls_idempotent_left)
  apply(insert Cls_base) apply(auto)
done

--\** IndSetDefExample.thy (Isar script Scripting)--L50--42%-----
proof (prove): step 2

fixed variables: X

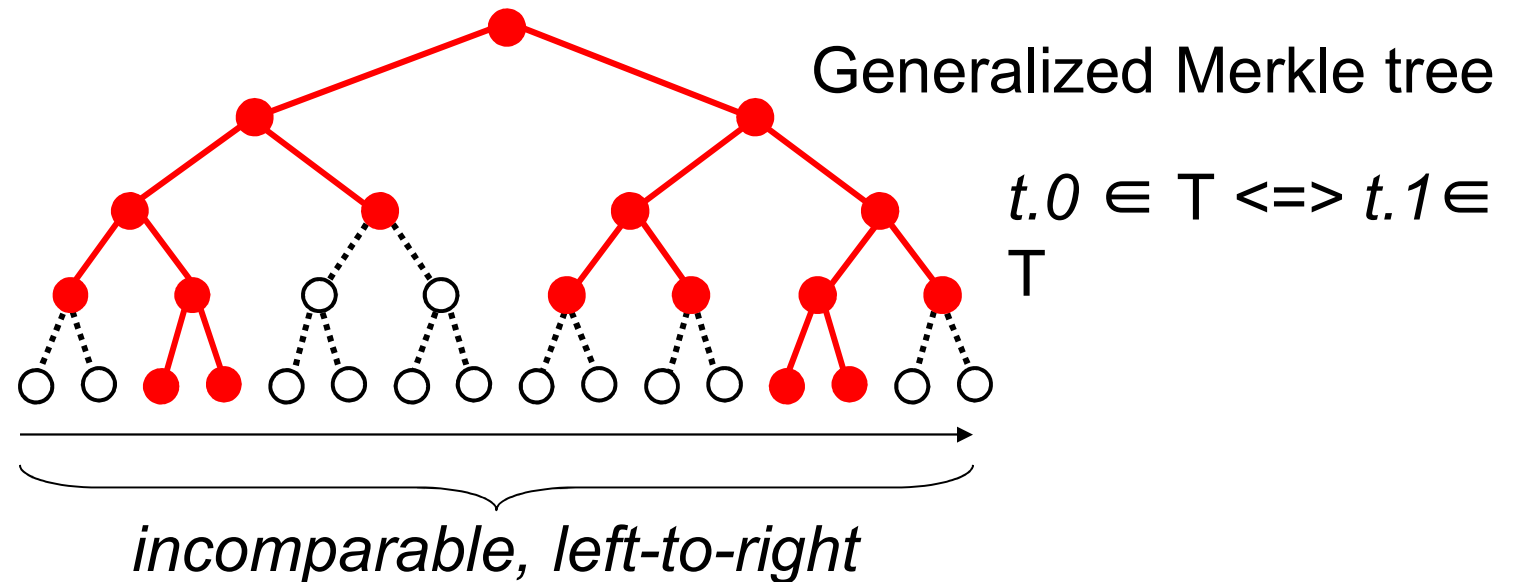
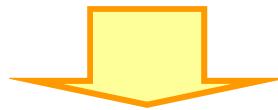
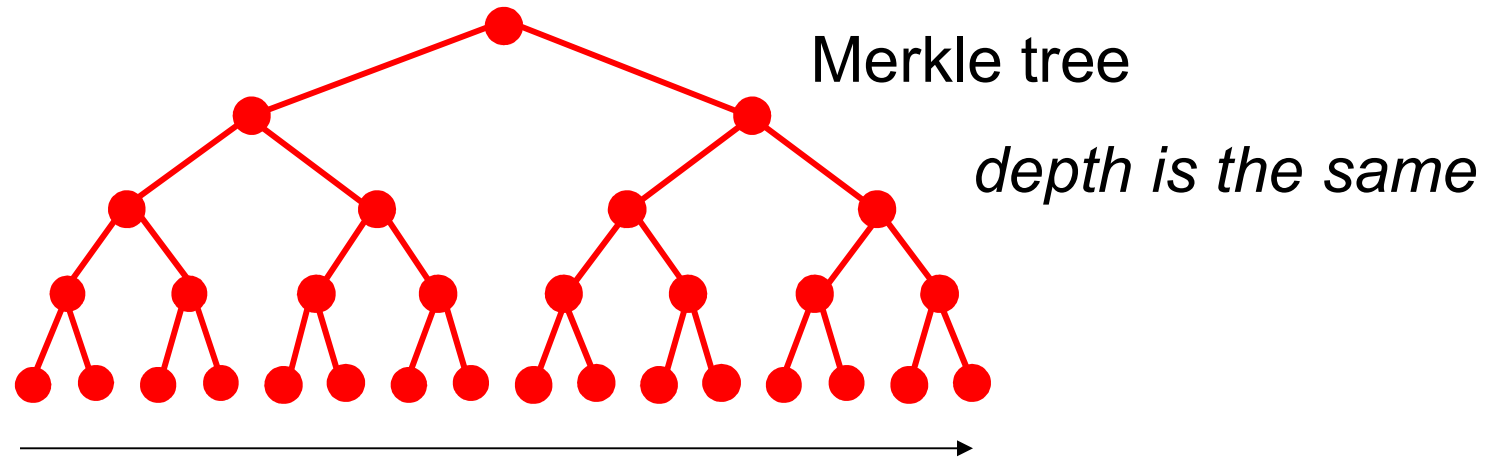
goal (lemma (Cls_idempotent_left), 2 subgoals):
  1. !!x bv. bv : Cls X ==> bv : Cls X
  2. !!x bv.
      [| bv @ [\<zero>] : Cls (Cls X); bv @ [\<zero>] : Cls X;
        bv @ [\<one>] : Cls (Cls X); bv @ [\<one>] : Cls X |]
      ==> bv : Cls X

--:-- *isabelle/isar-goals* (proofstate)--L1--All-----
```

# MONA Trick 1 : Generalized Merkle tree

- MONA **cannot** describe that :  
    *“a binary tree has the same depth”*  
    (i.e., *each root path has the same length*)
- We have been implicitly assuming that :  
    *“a Merkle tree has the same depth”*
- We relax *“the same depth”* to just *“don’t care depth”*.

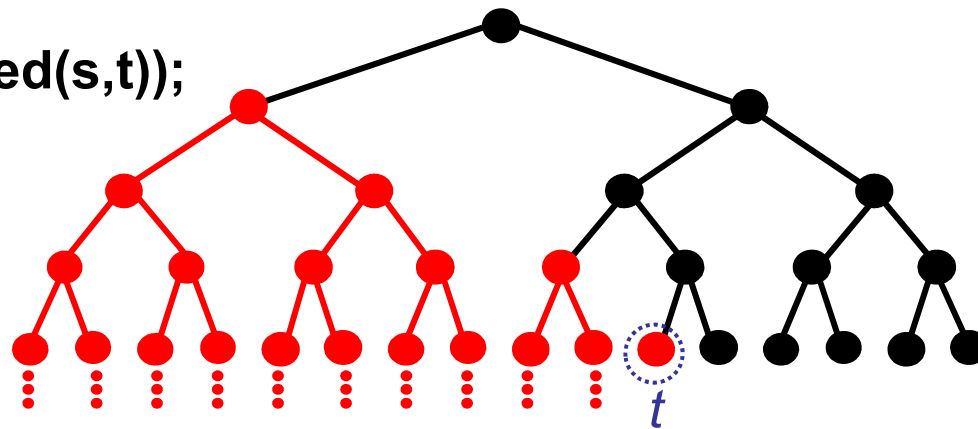
# Generalized Merkle tree example



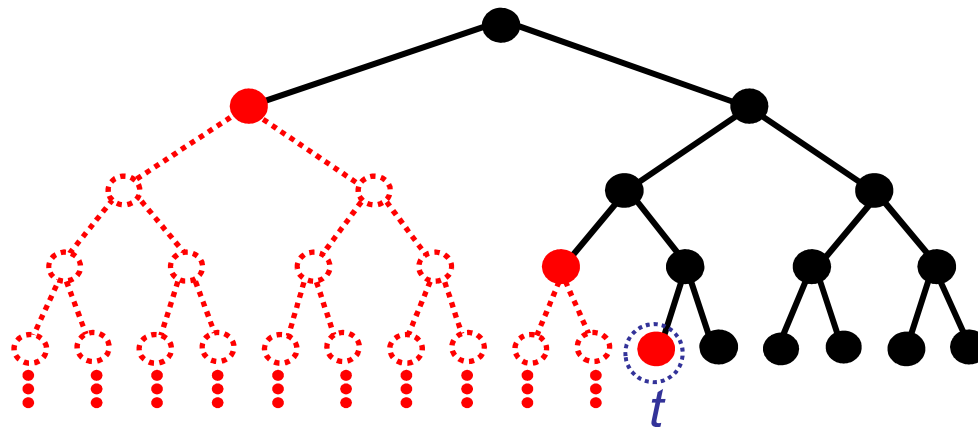
## MONA Trick 2: *Temporal slice* in WS2S

- First attempt : “Subtrees can grow infinitely.” (S2S)

all1 s: (s in X  $\iff$  defined(s,t));

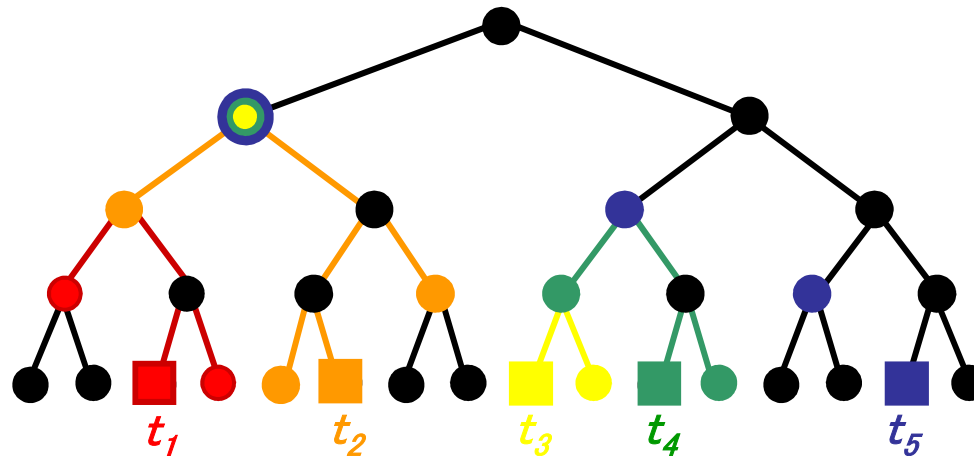


- Second attempt : “*temporal slice as its roots.*”



# Sanity Check

- Hash values are computed from different  $LSR_{t_{i+1}}(t_j)$ 's (i.e., different users compute hash values individually).
  - ✓ If multiple computations at each node coincide (i.e., *consistent*), it suggests *no internal-failures*.



- **Key Lemma.** Let  $i+1 \leq k \leq j$ . Then,
 
$$\text{Cls}(LSR_{t_{i+1}}(t_j)) \cap \text{Cls}(LSR_{t_{j+1}}(t_j)) \subseteq \text{Cls}(LS(t_k))$$

*Described in WS2S*

( $\text{left}(s,t) \ \& \ (t = u \mid \text{left}(t,u)) \ \& \ (u = v \mid \text{left}(u,v)) \ \& \ (v = w \mid \text{left}(v,w))$ )  
 &  $\text{LSRclosure}(s,t,X) \ \& \ \text{LSclosure}(u,Y) \ \& \ \text{LSRclosure}(v,w,Z)$   
 $\Rightarrow X \text{ inter } Z \text{ sub } Y;$



# Consistency

Beyond  
WS2S  
(MONA)

- Let  $(U_i, \alpha_i)$  such that
  - ✓  $U_i$  : a set of incomparable nodes,
  - ✓  $\alpha_i$  : labeling function on  $U_i$   
(extended  $\alpha_i(t) = \text{hash}(\alpha_i(t.0), \alpha_i(t.1))$  on  $\text{Cls}(U_i)$ )
- **Def.**  $\{(U_i, \alpha_i)\}$  is *weakly consistent* if  $\alpha_i(t) = \alpha_j(t)$  for each  $t \in \text{Cls}(U_i) \cap \text{Cls}(U_j)$
- **Def.**  $\{(U_i, \alpha_i)\}$  is *consistent* if  $\alpha$  holds
$$\alpha(t) = \begin{cases} \alpha_i(t) & \text{when } t \in U_i \\ \text{hash}(\alpha(t.0), \alpha(t.1)) & \text{when } t \notin \text{leaves}(\cup U_i) \end{cases}$$

# Incremental Sanity Check

- **Lemma 1.**  $\{(LSR_{t_{i+1}}(t_j), \alpha_j) \mid 1 \leq i < n\} \cup \{(LS(t_n), \alpha_n)\}$  is *weakly consistent*, if  $\{(LSR_{t_{i+1}}(t_j), \alpha_j), (LS(t_{i+1}), \alpha_{i+1})\}$  is *weakly consistent* for each  $1 \leq i < n$ .

- **Lemma 2.**  $\{(LSR_{t_{i+1}}(t_j), \alpha_j) \mid 1 \leq i < n\} \cup \{(LS(t_n), \alpha_n)\}$  is *consistent*, if it is *weakly consistent*.

**Proof.** Let  $X$  be the optimal slice of  $\{t_1, t_2, \dots, t_n\}$ .

By induction on  $|X \cap T_t|$  for  $T_t = \{s \mid t \leq s\}$ .

Beyond  
WS2S  
(MONA)

- **Cor.**  $\{(LSR_{t_{i+1}}(t_j), \alpha_j) \mid 1 \leq i < n\} \cup \{(LS(t_n), \alpha_n)\}$  is *consistent*, if  $\{(LSR_{t_{i+1}}(t_j), \alpha_j), (LS(t_{i+1}), \alpha_{i+1})\}$  is *weakly consistent* for each  $i$  with  $1 \leq i < n$ . *Each user correctly computes*

# Discussion

- Early formal proof example on incremental Merkle tree.
  - ✓ Mainly for “*Sanity check*”
- Isabelle sledhammer
  - ✓ FOL provers, SMT solvers, but not for MONA.
  - ✓ How to generate Isabelle proof for MONA validity?
- More properties
  - ✓ DLT neglected.
  - ✓ *Almost everywhere, consensus* ( $\Rightarrow$  Belief?)