Title	Proving Properties of Incremental Merkle Trees
Author(s)	Ogawa, Mizuhito; Horita, Eiichi; Ono, Satoshi
Citation	
Issue Date	2009-09-22
Type	Presentation
Text version	publisher
URL	http://hdl.handle.net/10119/8333
Rights	
Description	1st VERITE : JAIST/TRUST-AIST/CVS joint workshop on VERIfication TEchnologyでの発表資料, 開催：2005年9月21日～22日, 開催場所：金沢市文化ホール3F

Proving Properties of Incremental Merkle Trees

Mizuhito Ogawa, Eiichi Horita, Satoshi Ono

September 22nd, 2005

JAIST-AIST joint workshop, VERITE

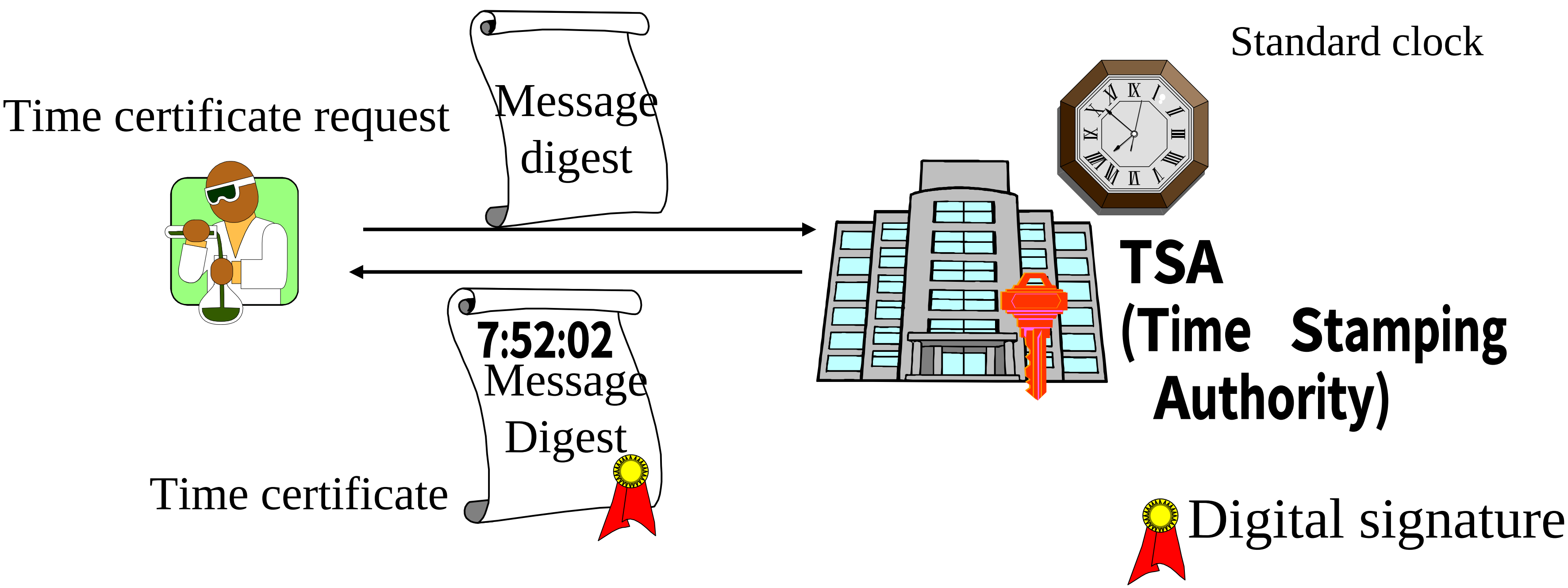
(presented at CADE-20, July 27th 2005)

What is temporal authentication ?

Certificate an occurrence of an transaction at “*time*”

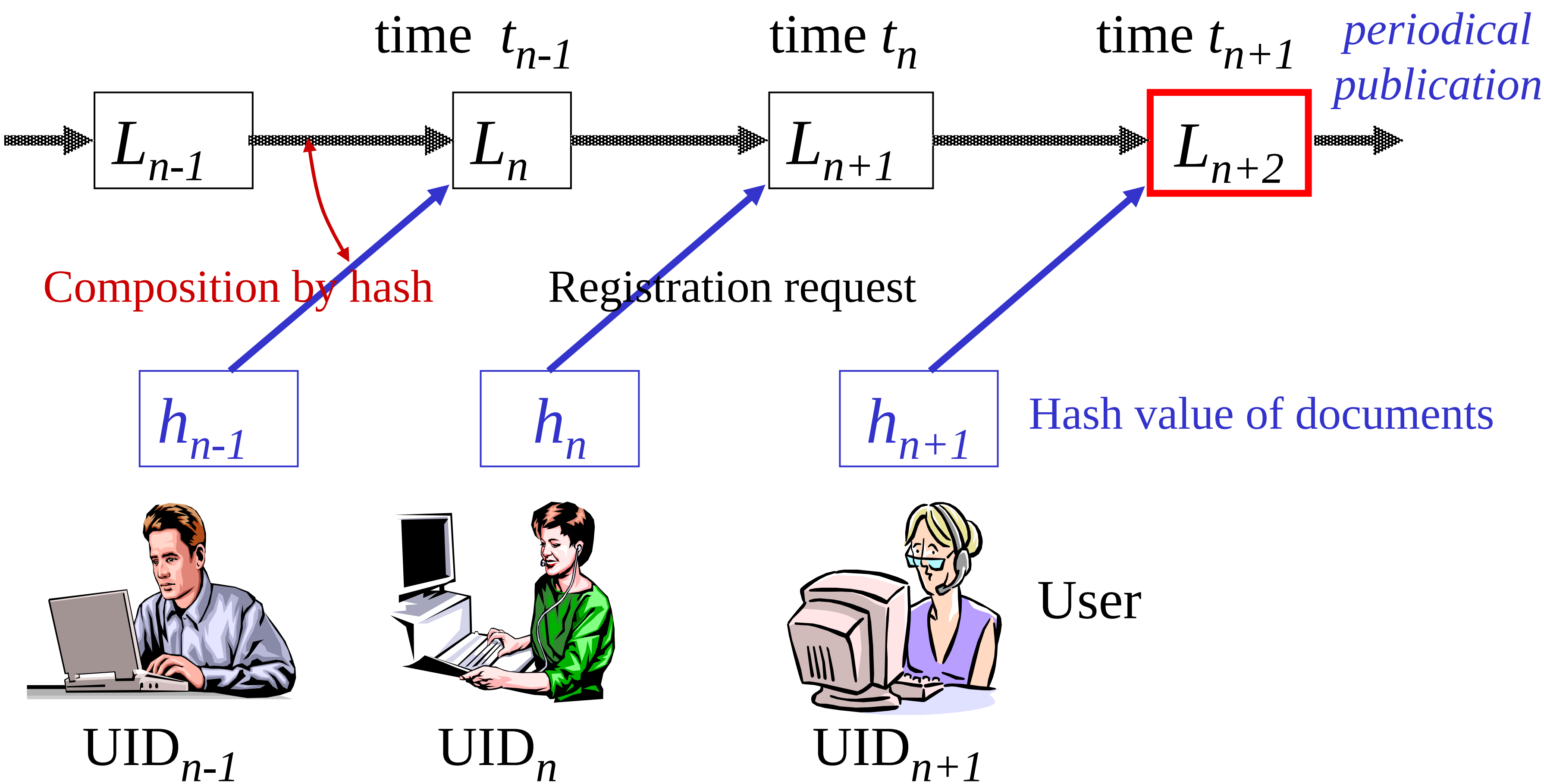
- Time stamp by digital signature (rfc-3161)
- Linking and publication by hash (ISO 18014-3)

Time stamp by digital signature (rfc-3161)



Linking and publication by hash function (ISO18014-3)

Compose past time stamps by a hash function



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

Time stamp by digital signature v.s. Linking by hash

Time stamp by digital signature

First certificate

Pros：

- Relatively safe for intra-dishonesty by Hardware Secure Module.
- Fine precision (< sec).

Cons:

- Contamination of crypto system invalidates *all* certificates.
- Relatively short life span : ～5years.

Linking and publication by hash function

Secondary certificate

Pros：

- Relies only on hardness of a hash function (e.g., SHA-1).
- Relatively long life span : ～30years.

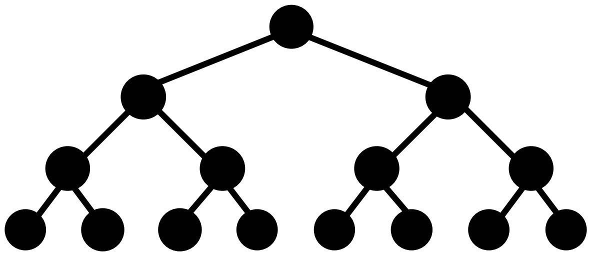
Cons:

- Hash function has no key; guarantee required for intra-dishonesty.
- During publication period, no auditor can check.

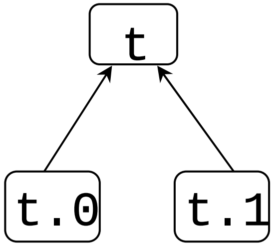
Merkle tree (Merkle 1979)

- *Merkle tree = Binary tree + hash function*

- Each node has its hash value, computed from a pair of hash values of its children.



+

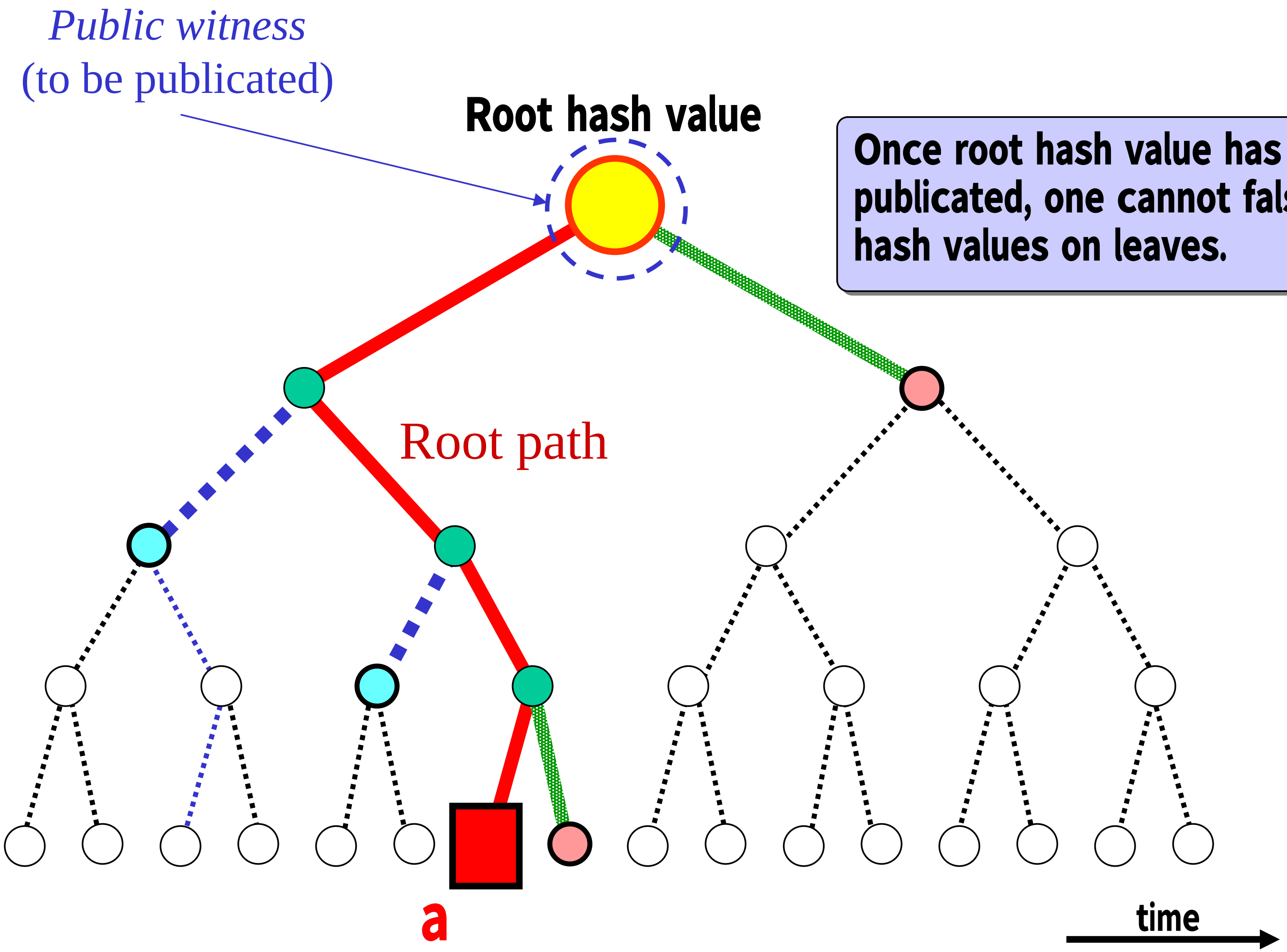


hash(v.0,v.1)

(v.0, v.1)

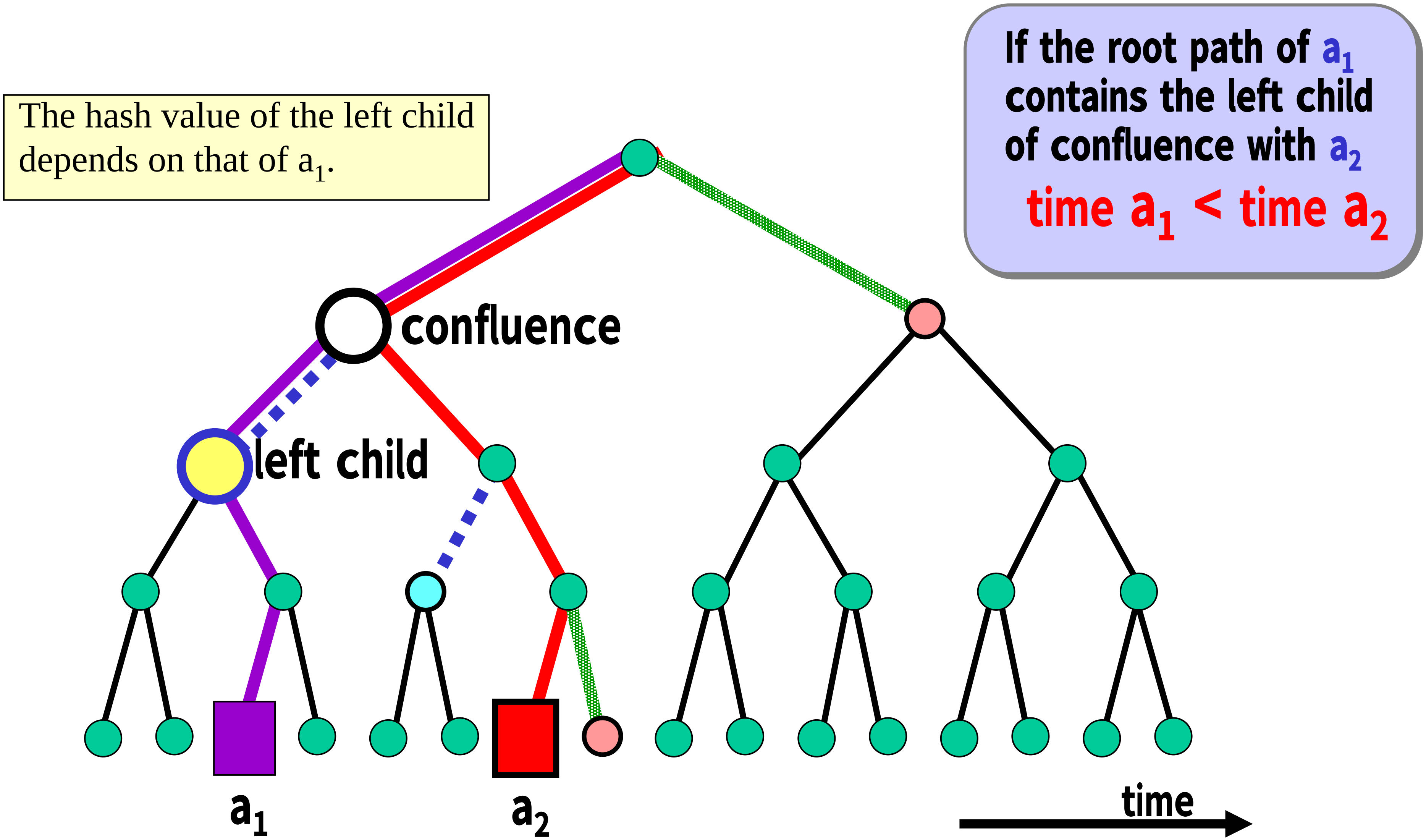
We assume collision-resistance, one-way hash function.

Basic idea : Merkle tree (1)



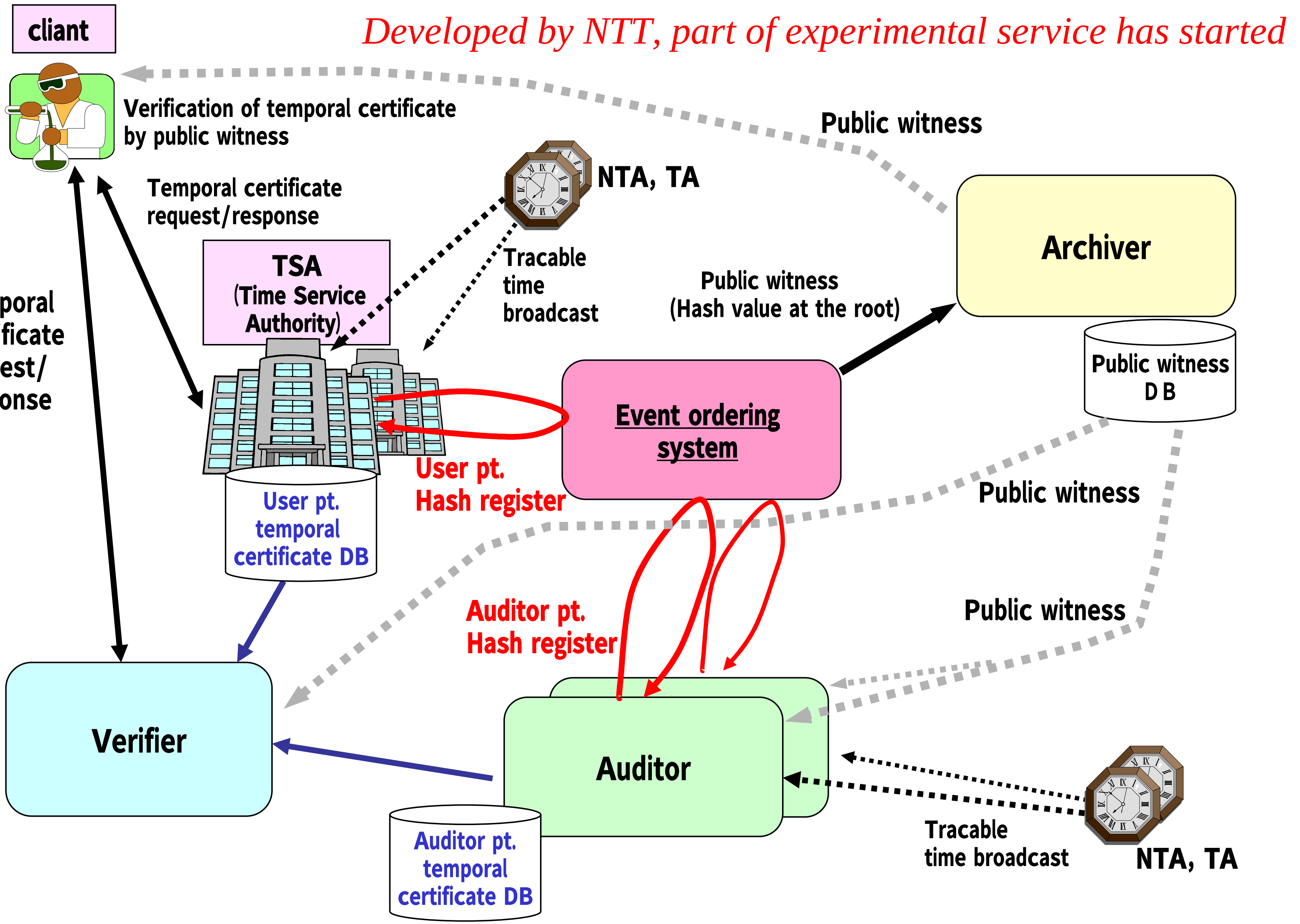
Once root hash value has been publicated, one cannot false hash values on leaves.

Basic idea: Merkle tree (2)

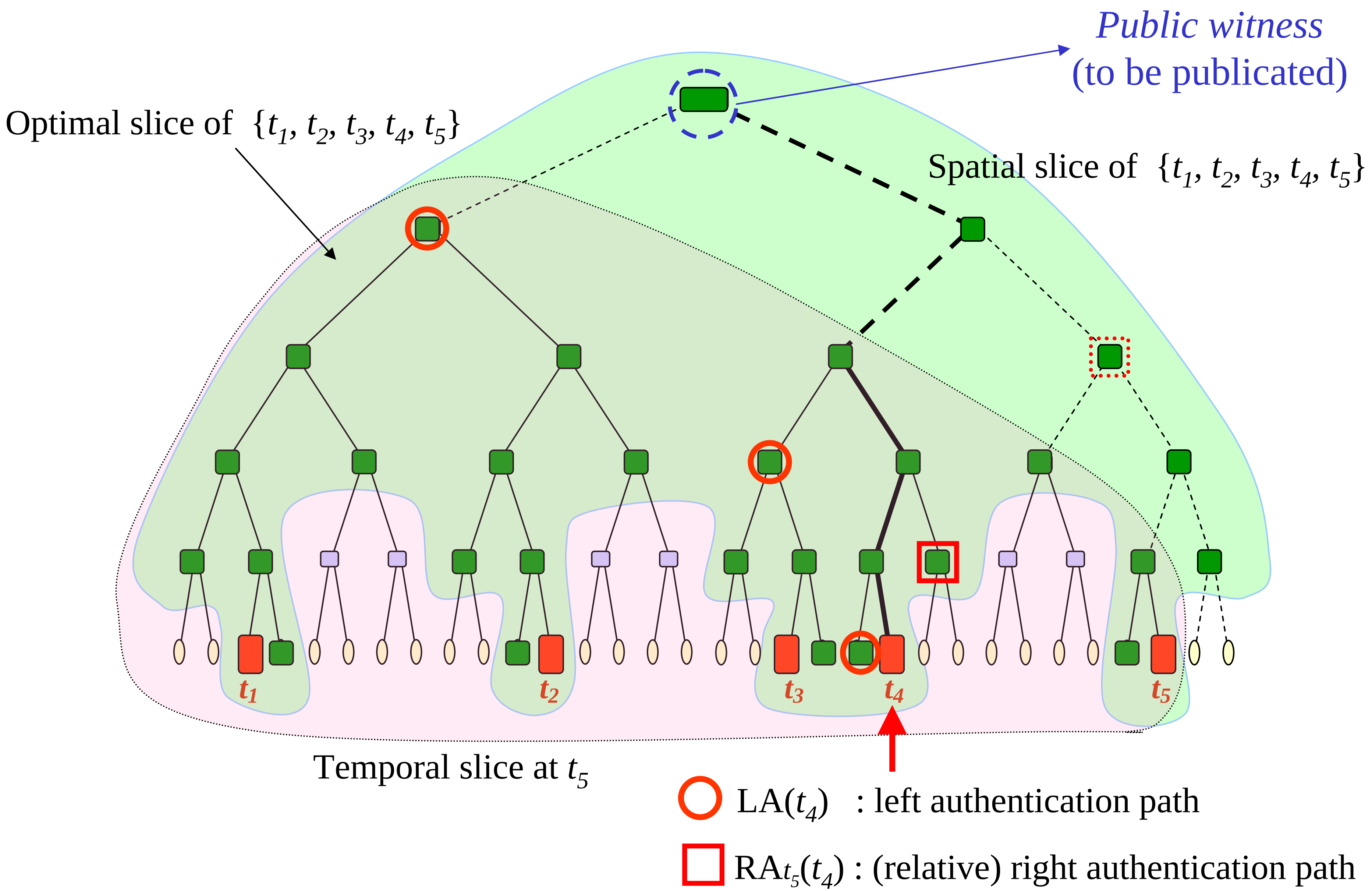


Reference model of event ordering system based on Incremental Merkle trees

Developed by NTT, part of experimental service has started



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



Protocol between users and an event-ordering system

- We assume that each user will register reasonably frequent.

- Assume a user registers at $t_1, t_2, \ldots, t_n$ 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 $0 < i \leqq n$.

Size $O(n)$

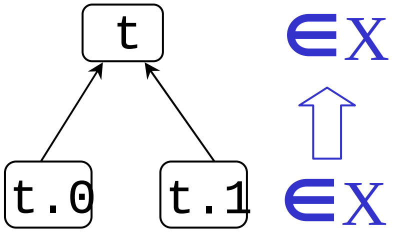
- We denote $LS(t_i) = LA(t_i) \cup \{t_i\}$

$$LSR_{t_{i+1}}(t_i) = LS(t_i) \cup RA_{t_{i+1}}(t_i)$$

Incremental scheme for Optimal Slice replication

- **Def.** Closure Cls(X) is the minimum set such that
  - X $\subseteq$ Cls(X)
  - t.0 (left child), t.1 (right child) $\in$ Cls(X)

    $\Rightarrow$ t $\in$ Cls(X)



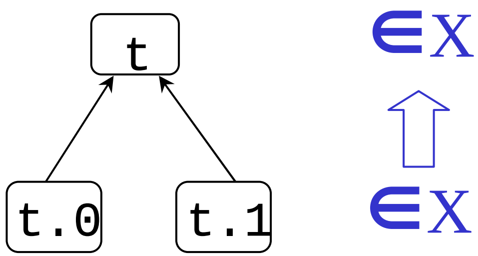
- **Th 1.** OptimalSlice($\{t_1, t_2, \ldots, t_n\}$)

  $= (\bigcup_{1 \leqq i < n} \mathrm{Cls}(\mathrm{LSR}_{t_{i+1}}(t_i))) \cup \mathrm{Cls}(\mathrm{LS}(t_n))$

  Described in WS2S

  (incomparable(A) & opt_slice(A,X) &
  LSRclosure_union(A,Y)) => X = Y;

MONA:WS2S satisfiablity checker




```
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
```

Trick 1 : Generalized Merkle tree

- MONA cannot describe that :

  "*a binary tree has the same depth*"

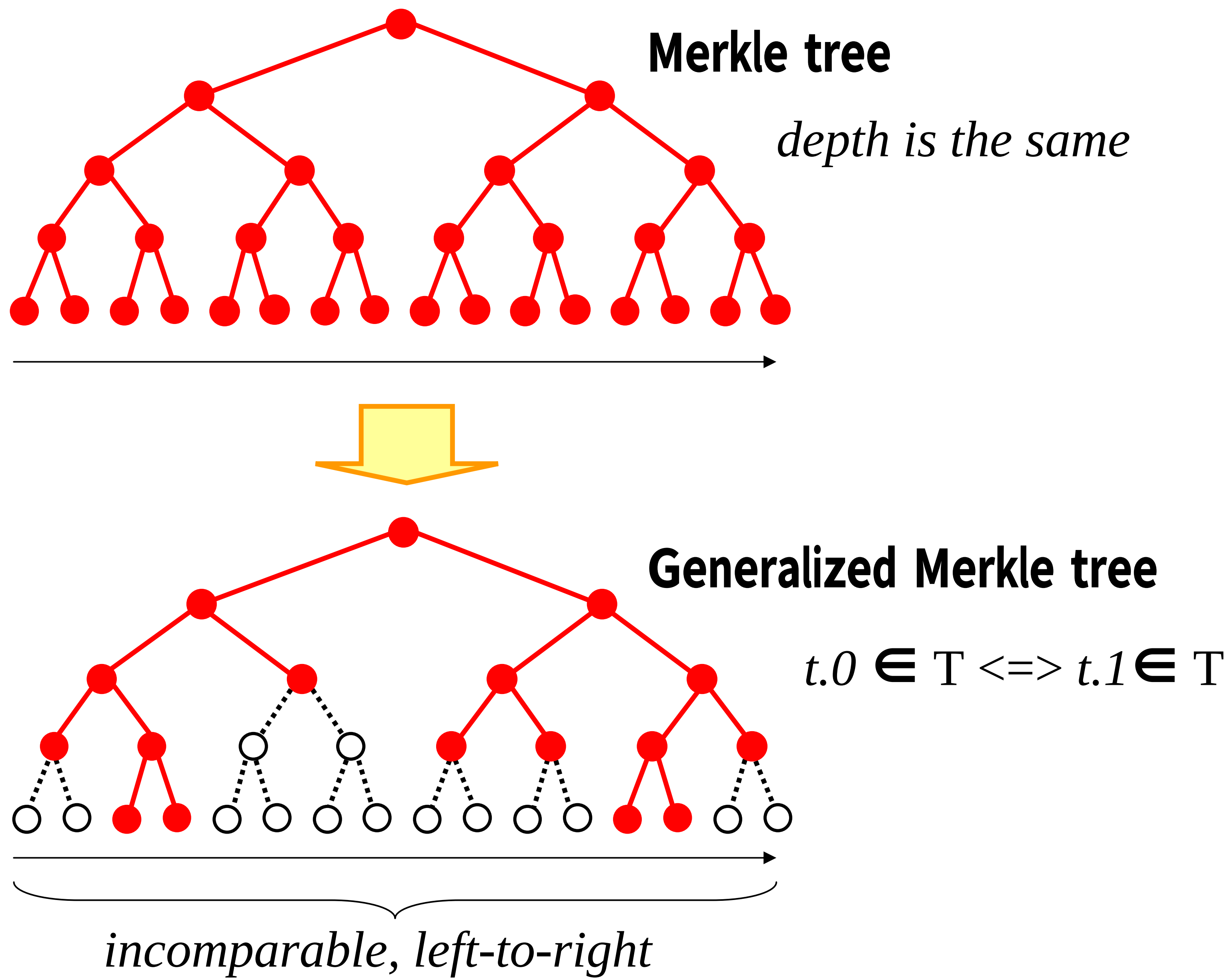
  (i.e., *each root path has the same length*)

- We have been implicitly assuming that :

  "*a Merkle tree has the same depth*"

- We relax "*the same depth*" to just "*don't care depth*".

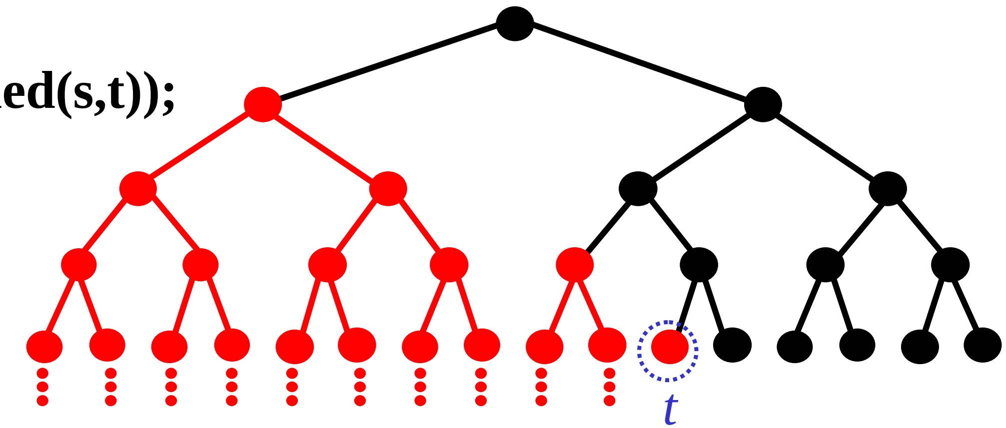
Generalized Merkle tree example



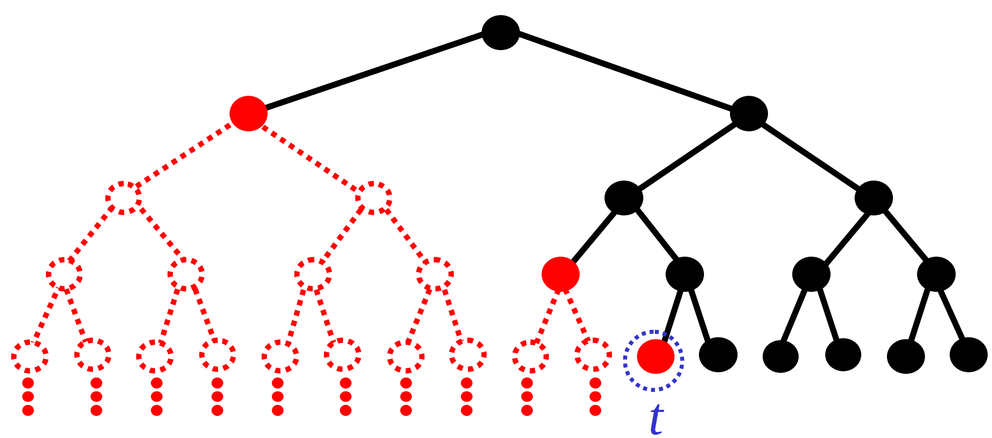
Trick 2: *Temporal slice* in WS2S

- First attempt : "*becomes an infinite set.*"

all1 s: (s in X <=> defined(s,t));

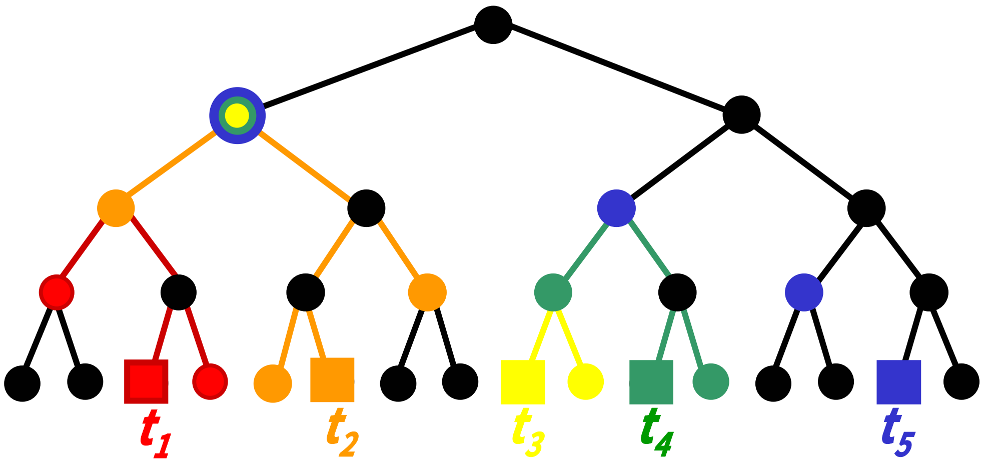


- Second attempt : "*temporal slice as its roots.*"



Second challenge: Sanity Check

- Hash values are computed from different $\mathrm{LSR}_{t_{i+1}}(t_i)$'s. If multiple computations at each node coincide (i.e., consistent), *it suggests no internal-failures*.



Consistency

- Let $(U_i,\ \alpha_i)$ such that
  U_i : a set of incomparable nodes,
  α_i : labeling function on U_i
  (extended $\alpha_i(\mathrm{t}) = \mathit{hash}(\alpha_i(t.0),\ \alpha_i(t.1))$ on $\mathrm{Cls}(U_i)$)

- **Def.** $\{(U_i,\ \alpha_i)\}$ is *weakly consistent* if
  $\alpha_i(t) = \alpha_j(t)$ for each $t \in \mathrm{Cls}(U_i) \cap \mathrm{Cls}(U_j)$

- **Def.** $\{(U_i,\ \alpha_i)\}$ is *consistent* if α is well-defined for

$$\alpha(t) = \begin{cases} \alpha_i(t) & \text{when } t \in U_i \\ \mathit{hash}(\alpha(t.0),\ \alpha(t.1)) & \text{when } \neg\, t \in \mathrm{leaves}(\bigcup U_i) \end{cases}$$

Correctness of incremental sanity check

Cannot be described in WS2S

- **Th 2.** *If* $\{ (\mathrm{LSR}_{t_{i+1}}(t_i), \boldsymbol{\alpha}_i), (\mathrm{LS}(t_{i+1}), \boldsymbol{\alpha}_{i+1}) \}$ *is weakly consistent for each i with* $1 \leqq i < n$, $\{(\mathrm{LSR}_{t_{i+1}}(t_i), \boldsymbol{\alpha}_\mathrm{i}) \mid 1 \leqq \mathrm{i} < \mathrm{n}\} \cup \{(\mathrm{LS}(t_n), \boldsymbol{\alpha}_\mathrm{n})\}$ *are consistent.*

Checked by large-scale experiment, but has not been proved !

- **Key Lemma**. Let $i+1 \leqq k \leqq j$. Then, $\mathrm{Cls}(\mathrm{LSR}_{t_{i+1}}(t_i)) \cap \mathrm{Cls}(\mathrm{LSR}_{t_{j+1}}(t_j)) \subseteq \mathrm{Cls}(\mathrm{LS}(t_k))$

Described in WS2S

(lefter(s,t) & (t = u | lefter(t,u)) & (u = v | lefter(u,v)) & (v = w | lefter(v,w)) & LSRclosure(s,t,X) & LSclosure(u,Y) & LSRclosure(v,w,Z)) => X inter Z sub Y;

Conclusion

- Case study of proving new properties of an event-ordering system developed by NTT, using MONA.
- Once clarified, they are not difficult; but when finding the *first* proofs (there are pitfalls), MONA assists very well.
- Found bug in “*where*”-sentence in WS2S mode of MONA:-)
- *Future work*: combine MONA & Isabelle/HOL, e.g., Th.2 (I have been saying this; but recent my focus is on Math…)

Thank you