International Conference on Information Theoretic Security (ICITS)
Madrid, Spain, May 25-28, 2007

Optimising SD and LSD in presence of non-uniform probabilities of revocation

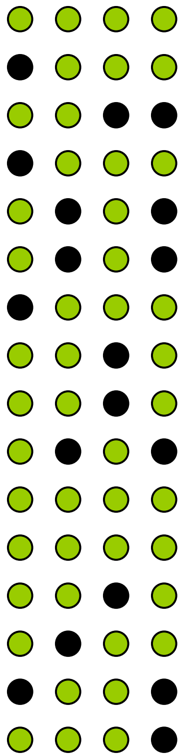
By

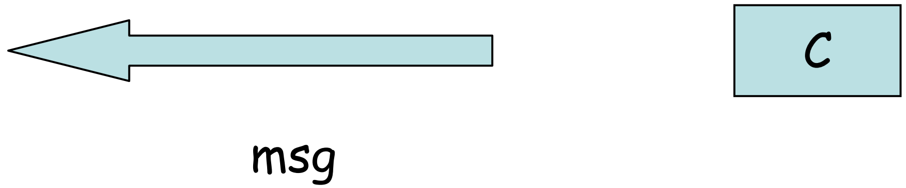
Paolo D'Arco and Alfredo De Santis

UNIVERSITÀ DI SALERNO

The Broadcast Encryption Problem
[Ber91, FN94]

- A center C broadcast a msg to a set N of receivers
- A subset R of them are revoked and should not be able to decrypt the msg
  - R changes from time to time





● revoked
● non-revoked

Applications
Key protection in media

- Content is distributed on CD, DVD, memory-card...
  - content is encrypted

- Players/Recorders are the receivers
  - typically are stateless
  - Receivers are given decryption keys at manufacturing

Goal:

- Revoke non-compliant players
  - revoked player cannot decode *future* content
- Trace the identity of a "cloned"/"hacked" player
  - black-box tracing

- Example: CPRM

Desiderata

- Low **bandwidth**: Small message expansion - E(content) not much longer than original message.
- Amount of **storage** at the users - I_u - small
  - Also at the center
- **Resiliency** to large coalitions of users who collude and share their resources

Contents of this talk

- Subset Cover framework: SD and LSD
  - Stateless receivers
- Revocation: non-uniform case
- Optimisation problems: coding theory
  - LKH schemes: Huffman codes
  - SD and LSD: Algorithms for Campbell's penalties
- Conclusions

Subset Cover Framework
[NNL01]

Framework encapsulates many previous schemes

- Idea: to revoke a subset R, partition the *remaining users* into subsets from some *predetermined* collection.
- Encrypt for each subset separately

An algorithm in the framework:

Underlying collection of subsets (of users/devices)

$S_1, S_2, \ldots, S_w$ $\quad\quad$ $S_j \subseteq N$.

- Each subset S_j is associated with a *long-lived* key L_j
  - A device $u \in S_j$ should be able to deduce L_j from its secret information I_u

The Broadcast Algorithm

- Choose a session key K
- Given *R*, **find** a partition of *N* \ *R* into disjoint sets

$$S_{i_1}, S_{i_2}, \ldots, S_{i_m}$$

$$N \setminus R = \bigcup S_{i_j}$$

with associated keys $L_{i_1}, L_{i_2}, \ldots, L_{i_m}$

- Encrypt message M

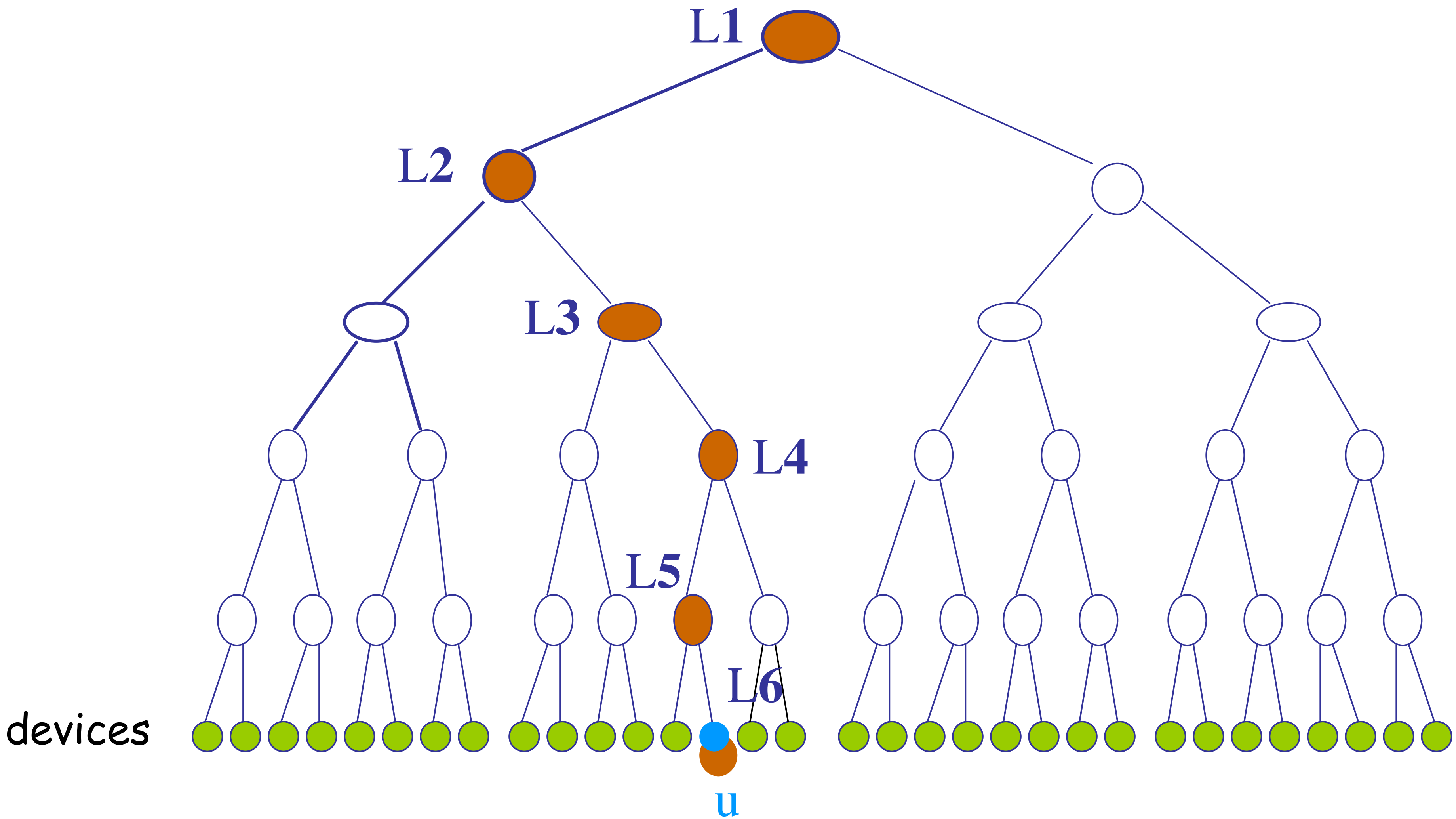
HEADER	Body
$[i_1, i_2, \ldots, i_m]$, $C_1 = E_{L_{i_1}}(K)$, … , $C_m = E_{L_{i_m}}(K)$	$F_K(M)$

Decryption (user u)

[$i_1, i_2, \ldots, i_m$], $C_1 = E_{L_{i_1}}(K)$, … , $C_m = E_{L_{i_m}}(K)$	$F_K(M)$
HEADER	Body

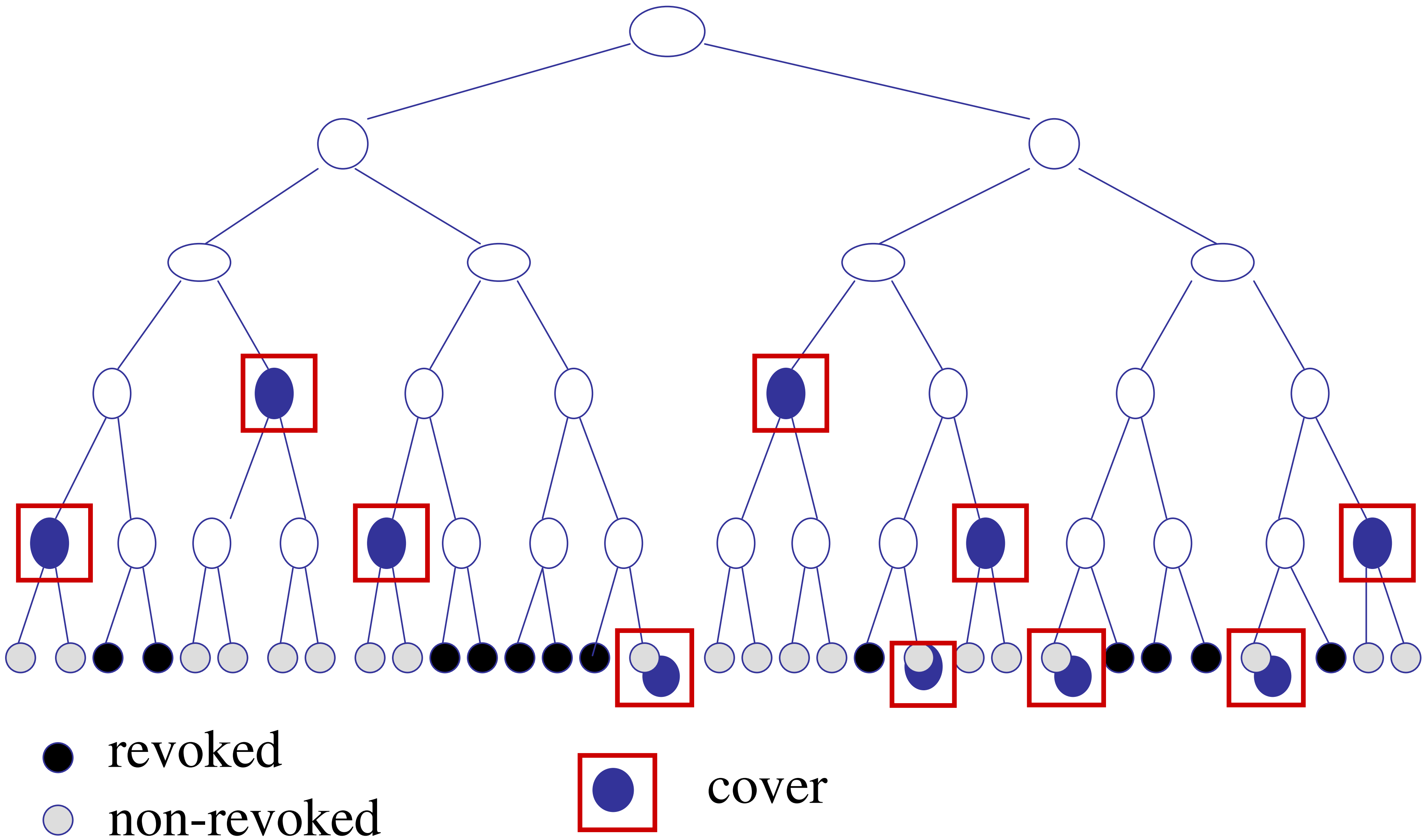
- Either
  - Find the subset i_j such that $u \in S_{i_j}$, or
  - null if $u \in R$ — u *is revoked!*
- Obtain L_{i_j} from the private information I_u
- Compute $D_{L_{i_j}}(C_j)$ to obtain K
- Decrypt $F_K(M)$ with K to obtain the message.

Complete Subtree

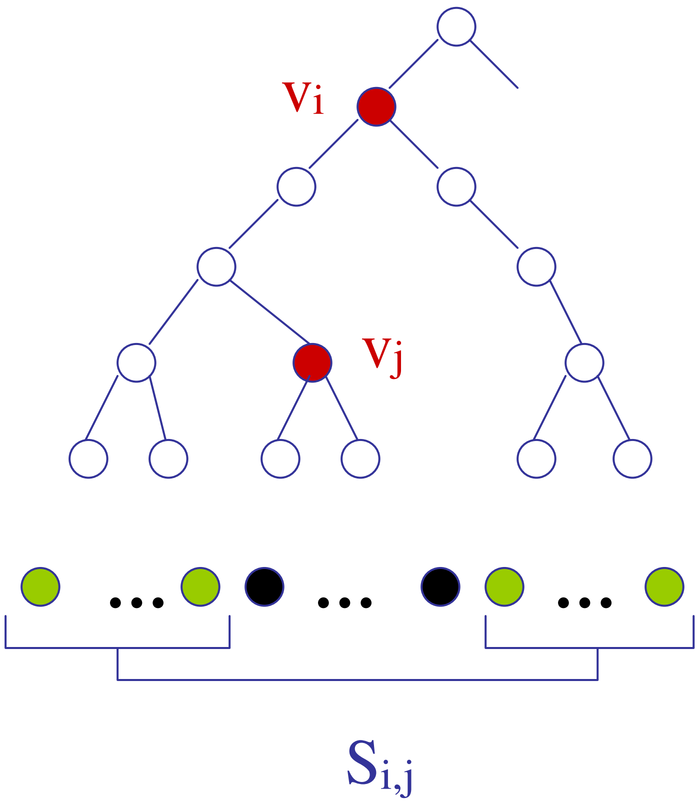


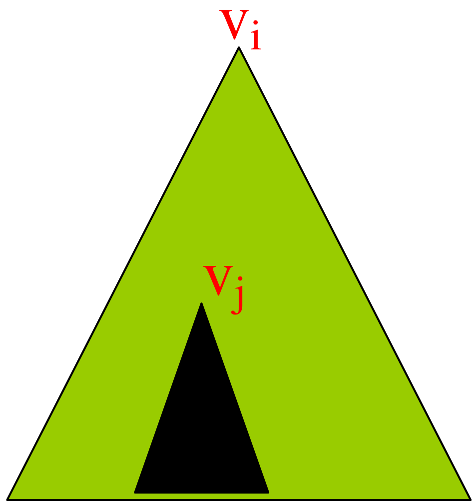
$I_u = \{ L_1 , L_2 , L_3 , L_4 , L_5 , L_6 \}$

Subset Cover of non-revoked devices
Complete Subtree Method



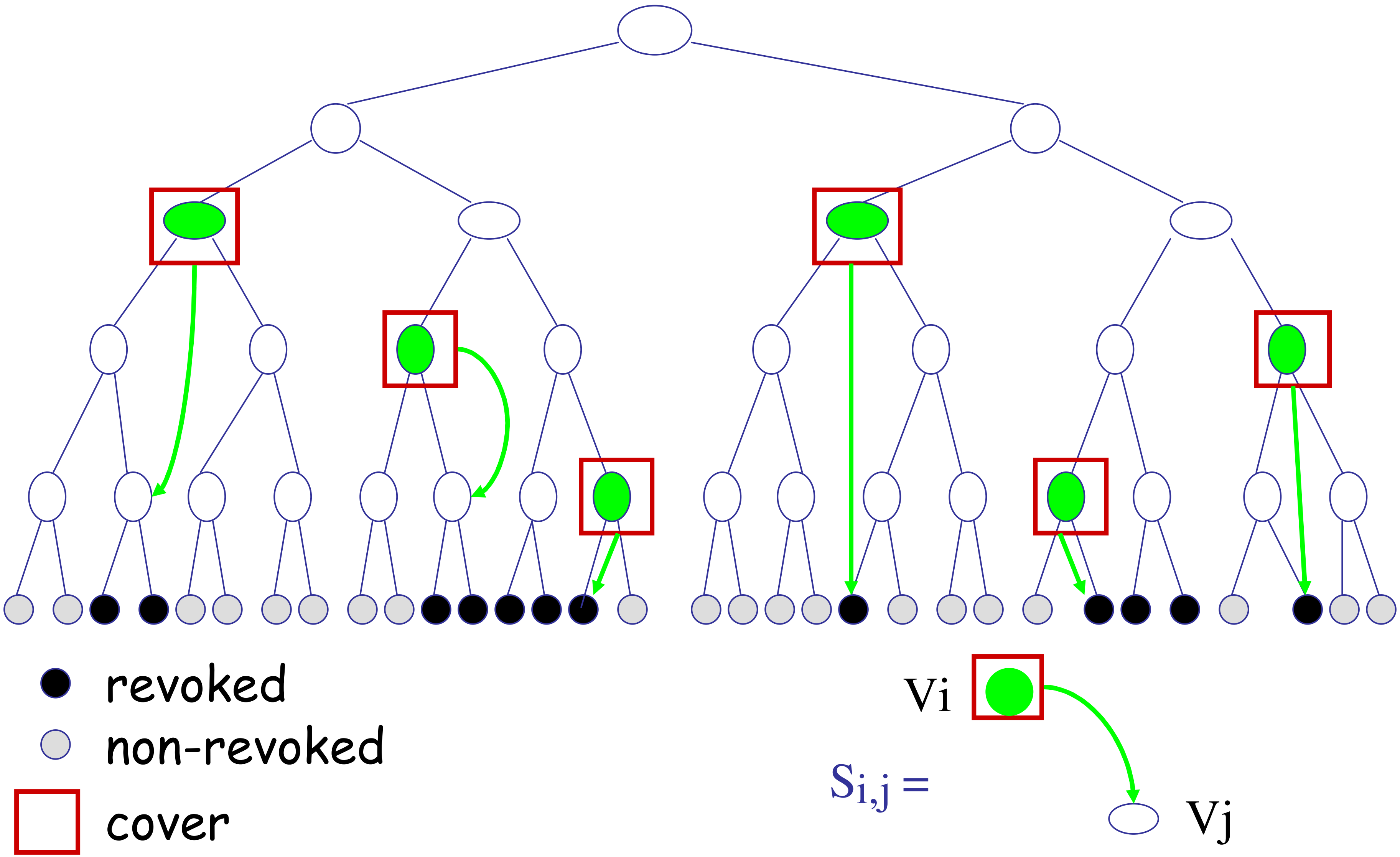
Subset Difference





$S_{i,j}$ = Set of all leaves in the subtree of V_i but <u>not</u> in V_j

Subset Cover of non-Revoked Devices
Subset-Difference Method



Key-Assignment
Subset-Difference Method

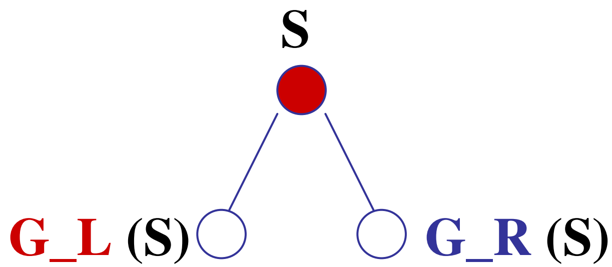
- Naive approach to the key assignment:
  - assign a key $L_{i,j}$ to every pair $[v_i, v_j]$ in the tree
  - impractical: each device must store O(n) keys…

- Use G, a pseudo-random sequence generator that ***triples*** the input length ($k \rightarrow 3k$) à la GGM

Key-Assignment
Subset-Difference Method

- Use G to derive a labeling process
  - S – label at node,
  - $G_L(S)$ – label at left child, $G_R(S)$ – label at right child
  - $G_M(S)$ – key at node.

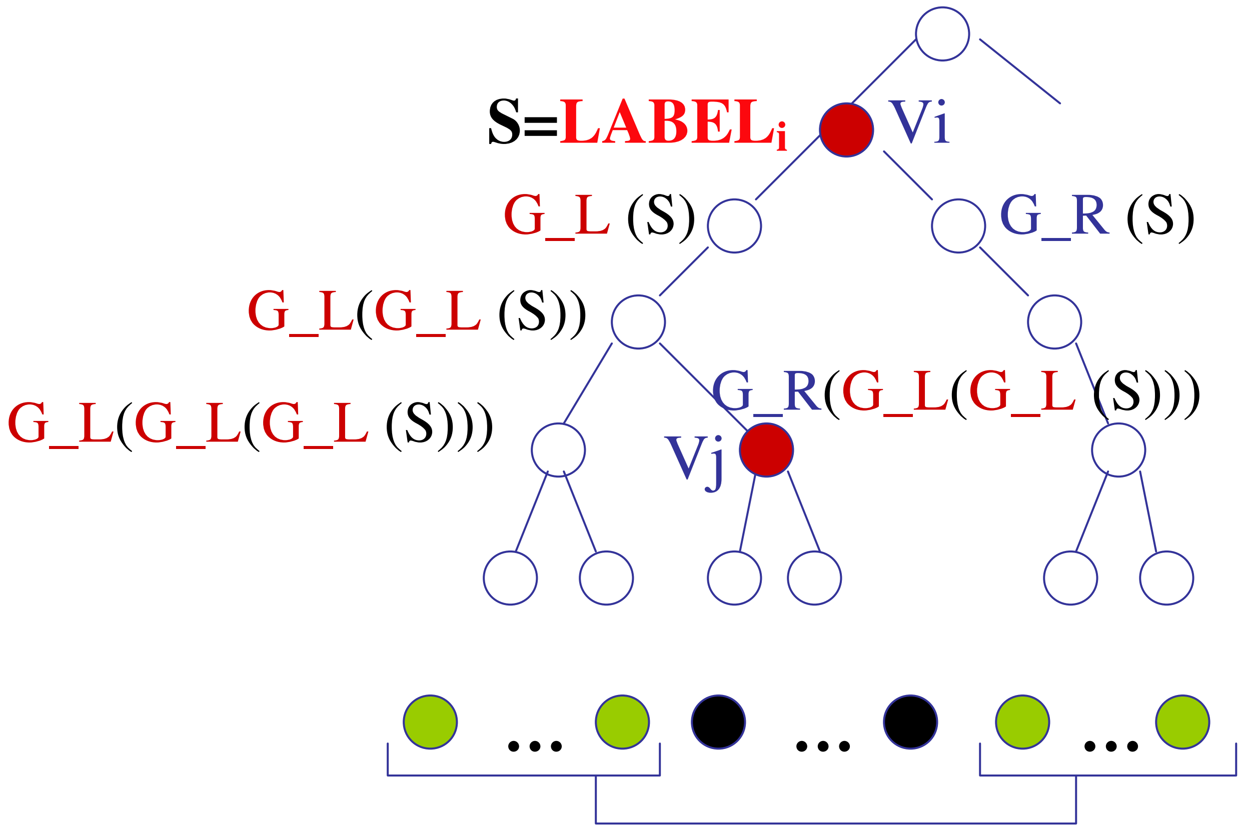
G (S) = | G_L (S) | G_M (S) | G_R (S) |



Assign to each node V_i a label **$LABEL_i$**

The key $L_{i,j}$ = G_M of the label **$LABEL_{i,j}$** at node V_j derived from **$LABEL_i$** down towards V_j

Key-Assignment
Subset-Difference Method

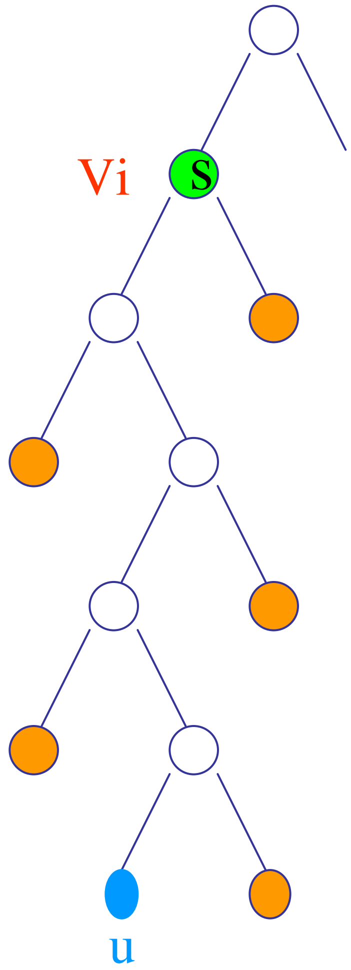


$$LABEL_{i,j} = G\_R(G\_L(G\_L\ (L_i)))$$

$$L_{i,j} = G\_M\ (LABEL_{i,j}\ )$$

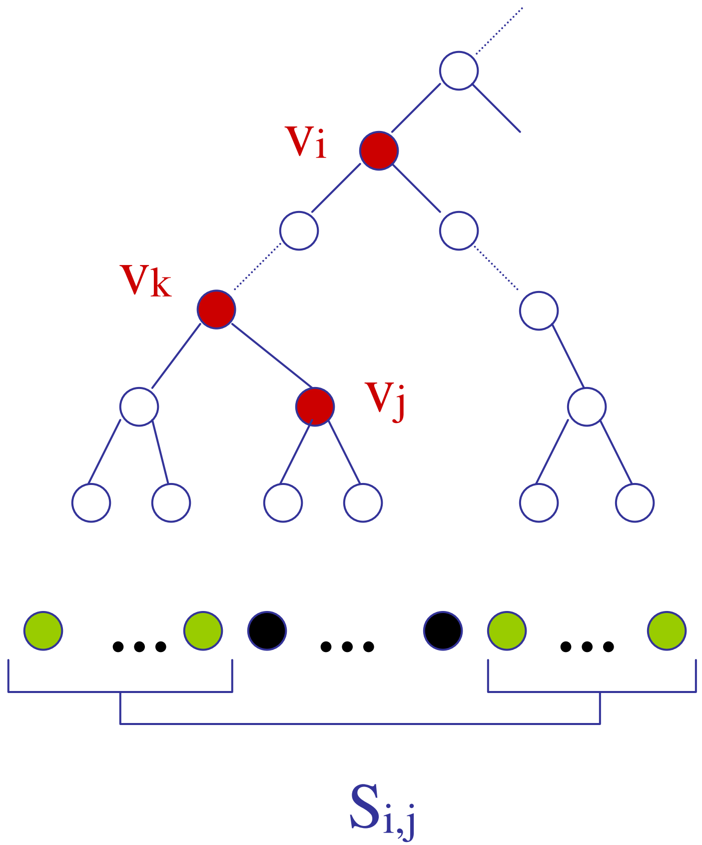
Providing Keys to Devices

- A device corresponds to a leaf u in the tree
- For every V_i ancestor of u whose label is S
  - u receives all labels at nodes that are **hanging off the path** from V_i to u. These labels are *all derived from S*.
- u can compute all keys of the sets it belongs to rooted at V_i , and *only* them.



Layered Subset Difference

[HS02]



Idea: A small collection of $S_{i,j}$

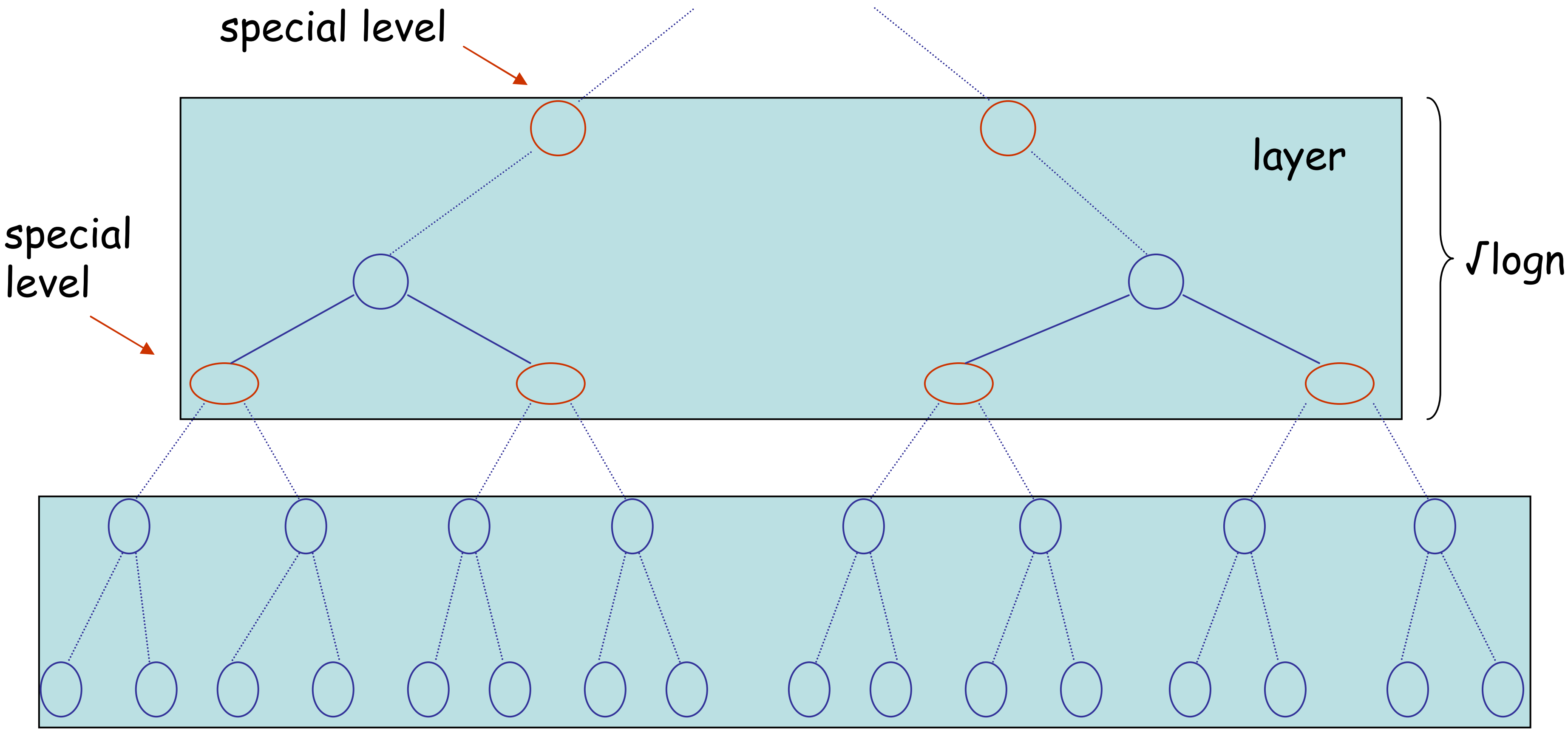
v_i, v_k, v_j nodes along root-to-leaf path

$\Rightarrow$ $S_{i,j} = S_{i,k} \cup S_{k,j}$

The tree has √logn special levels.
Levels between two special levels form a layer.

$S_{i,j}$: v_i and v_j at the same layer
or
v_i is at a special level

Layered Subset Difference



Performance

- CS
  - User storage: logn +1
  - Broadcast size: rlog(n/r)
- SD
  - User storage: $O(\log^2 n)$
  - Broadcast size: O(r)
- LSD
  - User storage : $O(\log^{3/2} n)$
  - Broadcast size: O(r)

n = # users
r = # revoked users

Non-Uniform Probabilities

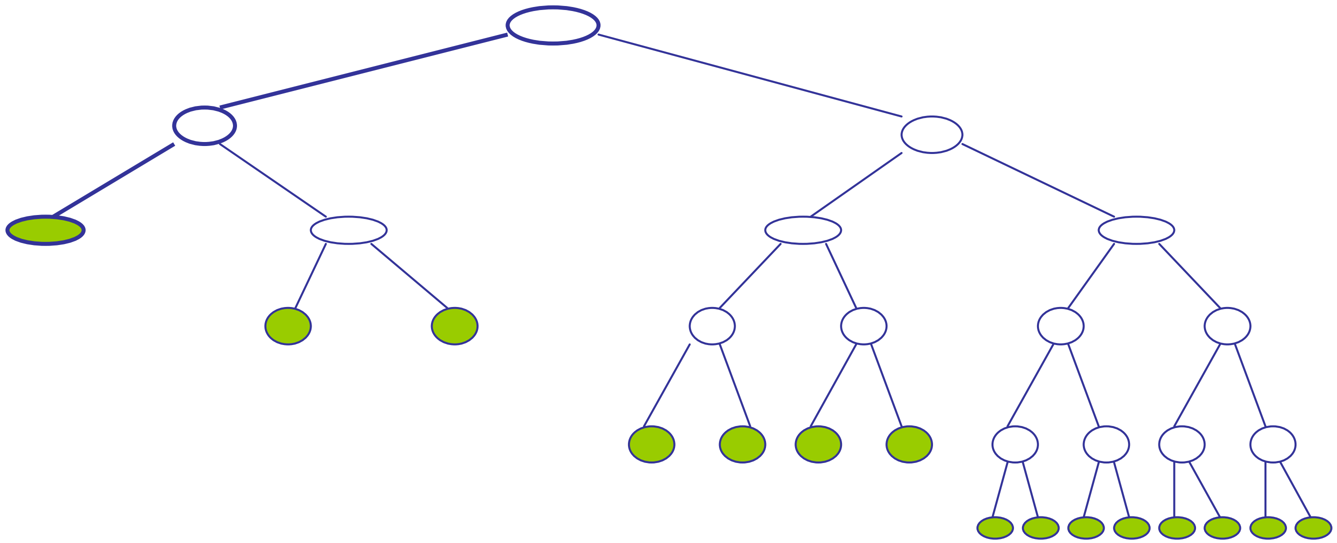
- Due to historical or legal reasons some geographic areas show different *adversarial behaviours*

- We would like to give less keys to devices held by malicious users, more to thustworthy ones

- User revocation: A probability distribution is available

- CS, SD, and LSD: the binary tree structure changes

Non-Uniform Probabilities



How to construct binary trees satisfying some optimality criteria

Revocation strategy?

It is easy to verify that the subset cover strategy

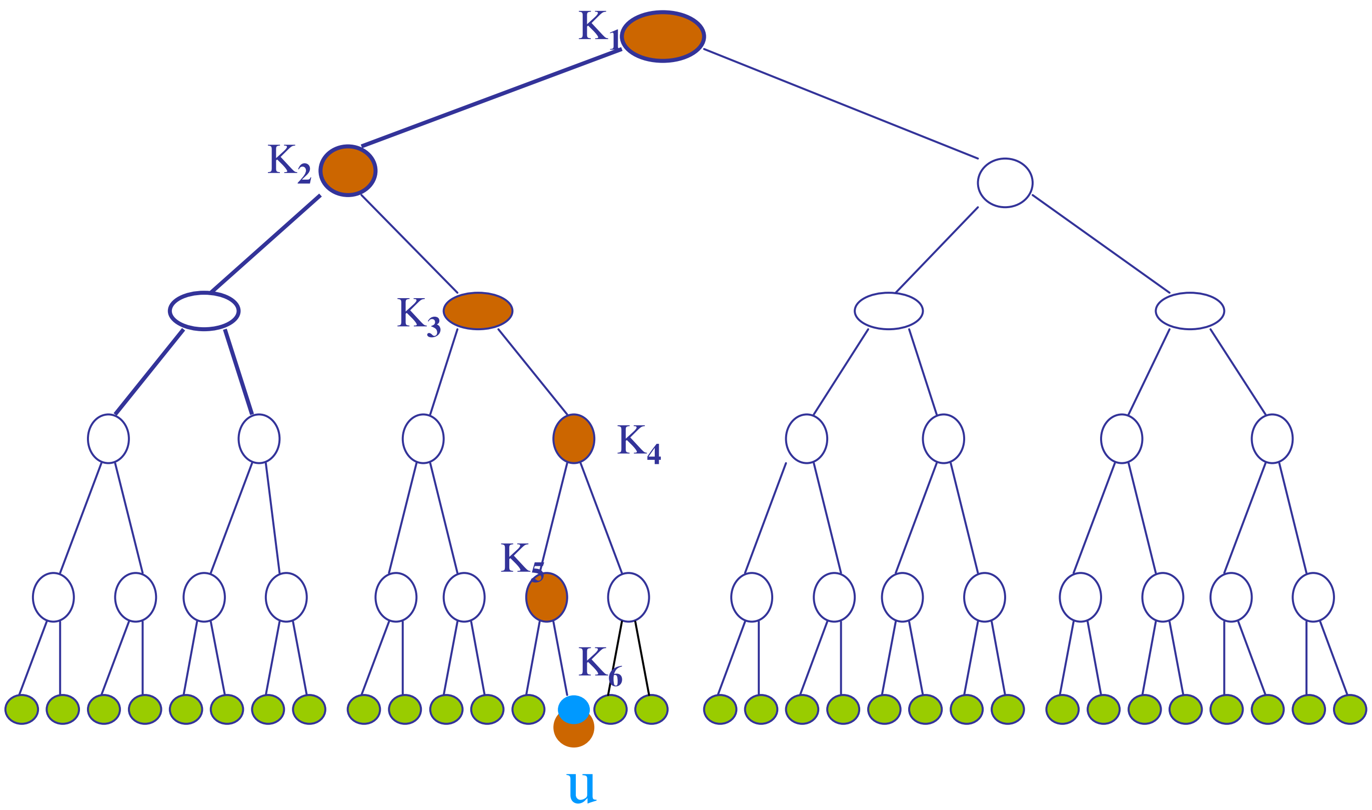
• at every iteration increases the number of covering subsets $S_{i,j}$ by **at most two**, and reduces **by one** the number of revoked leaves (Lemma 3 of NNL01)

• the property is **independent of** the structure of the tree, i.e., it holds even if the tree is not a full binary tree

• hence, the revocation strategy has the **same costs** of SD and LSD (i.e., O(r) broadcast msg size)

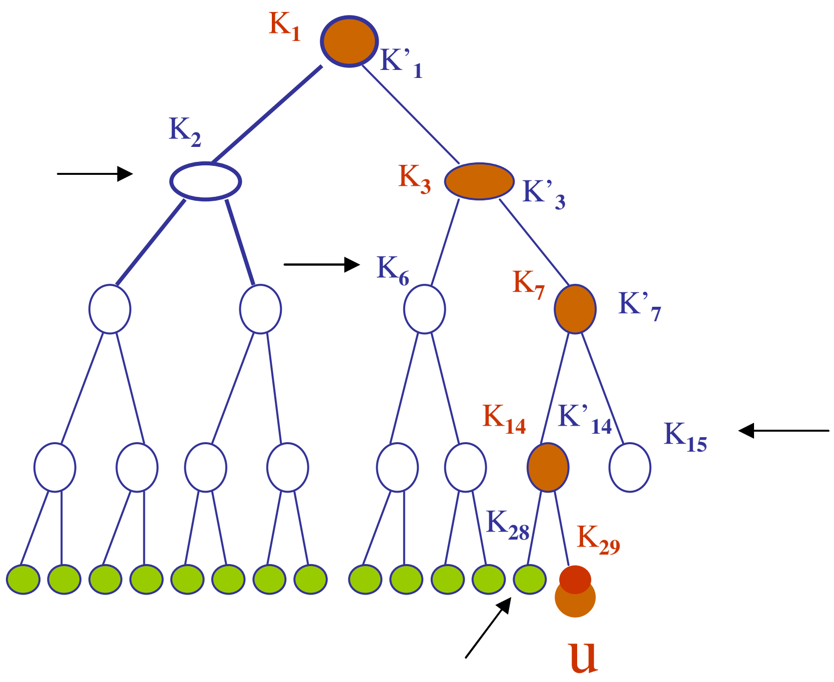
Previous work

Multicast:
Tree-based schemes



Users are associated to leaves. Receive keys assigned to nodes along the root-to-leaf path. The scheme is statefull.

Multicast:
Join and Revoke



GC

- delete K1, K3,K7,K14, K29
- generate new keys K'1, K'3,K'7,K'14
- send encrypted mgs

  E K_{28}(K'14),
  E K'_{14}(K'7), E K_{15}(K'7),
  E K_6(K'3), E K'_7(K'3),
  E K_2(K'1), E K'_3(K'1),

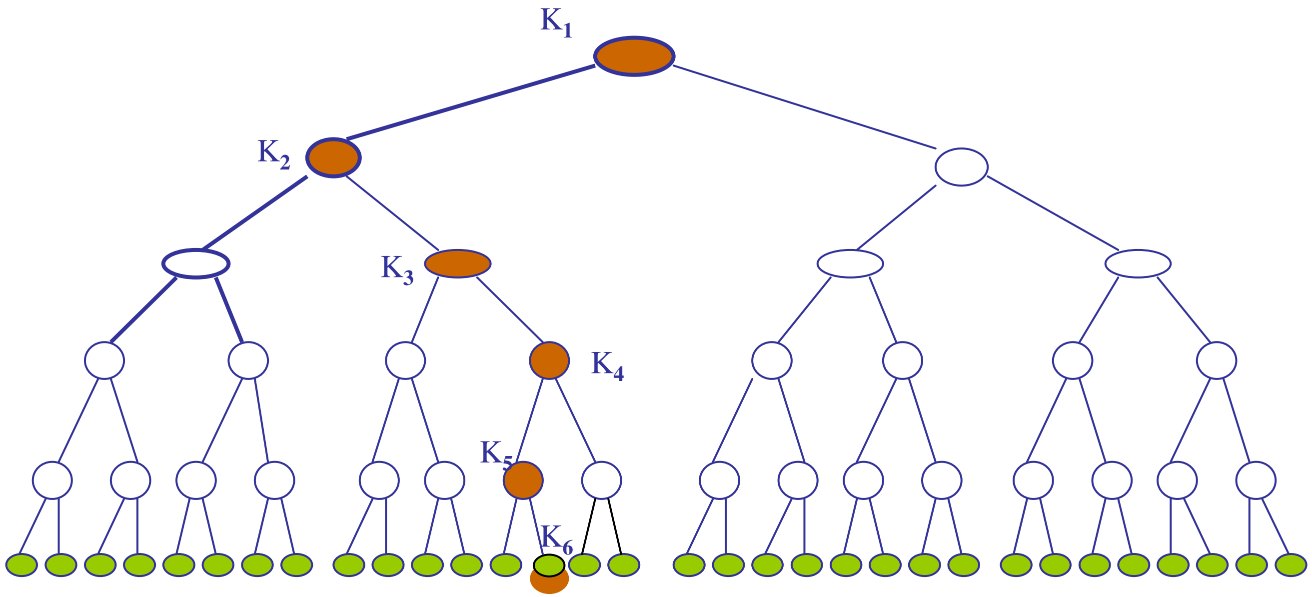
Revoke u. All keys along the root-to-leaf path need to be updated. The new keys are communicated to the users

LKH vs CS

- Same key assignment but different use
- CS – keys never change (stateless)
- LKH – keys updated due to join/revoke

Optimising user key storage in CS and LKH in presence of non-uniform prob. distribution is the **same**. Studied in [PB01].

LKH schemes
Key assignment [PB01]



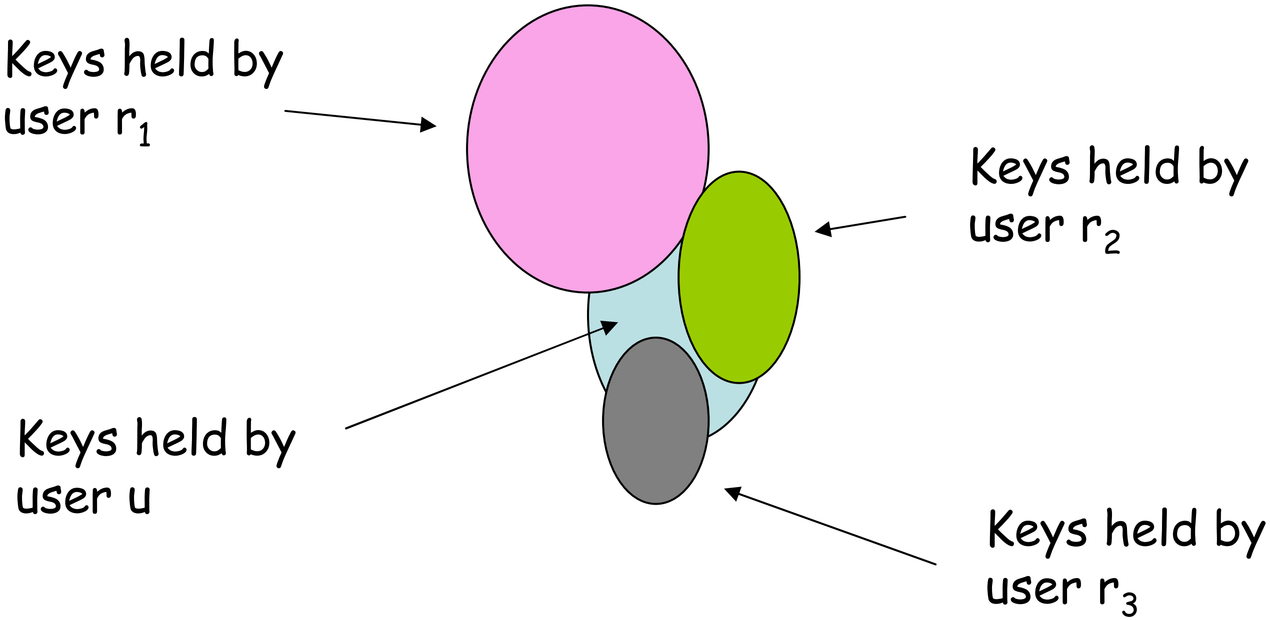
Problem 1

Which properties the keys assigned to the nodes of the key-tree have to satisfy to get a "secure" scheme?

Secure Scheme

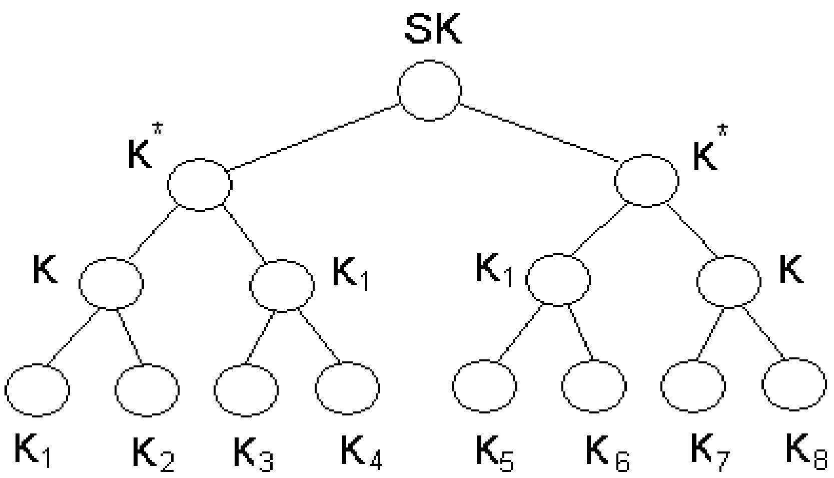
• The GC can always securely communicate with a non-revoked user

• Keys held by revoked users do not cover the subset of keys held by a non revoked user



Key Index (KID)

- KID_i = string obtained concatenating of all keys along the path
- KID_i unique w.r.t. permutation of concatenated keys



Distinct keys to leaves and prefix-free KIDs → secure assignment

Does not hold: the keys of U_1 are covered by the keys of U_5 and U_7

$KID_1 = K_1 \mid K \mid K^* \mid SK$

$KID_5 = K_5 \mid K_1 \mid K^* \mid SK$

$KID_7 = K_7 \mid K \mid K^* \mid SK$

A new characterization

For each leaf node u:

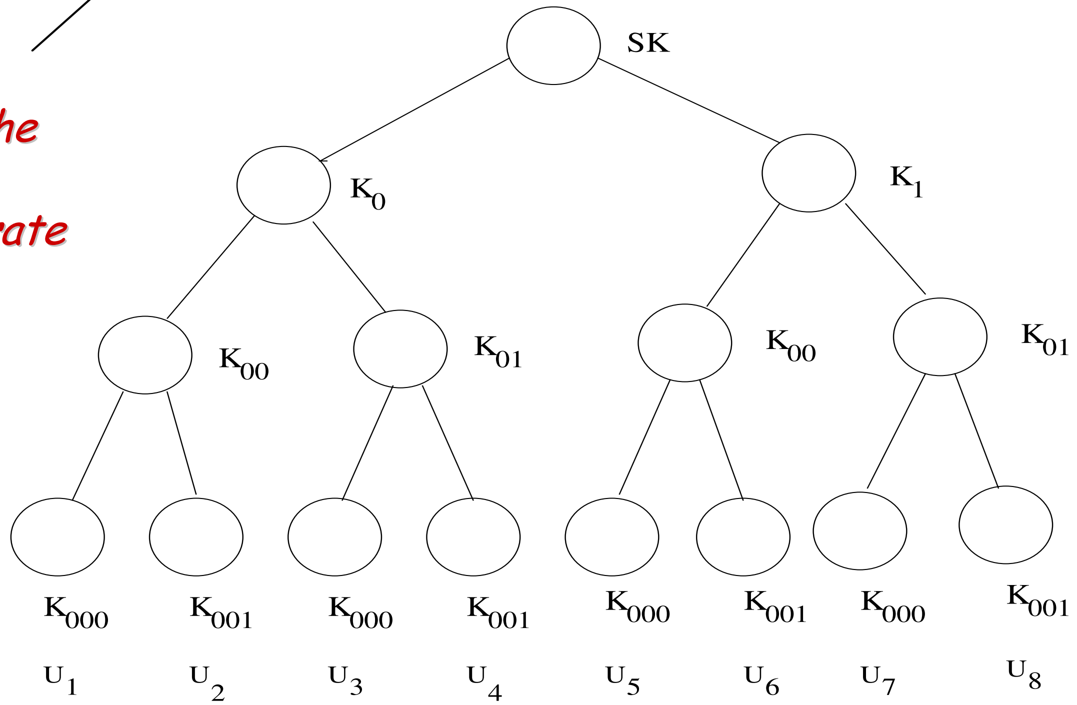
- $Kpath_u$ = multiset of keys along root-to-leaf path
- $Hpath_u$ = multiset of keys of nodes at distance 1 from the path

Key-tree secure w.r.t a single revoke operation if and only if

1. Keys in $Kpath_u$ all distinct
2. Keys in $Hpath_u$ all distinct
3. $Kpath_u \cap Hpath_u \neq 0$

Optimal key assignment [CEK+99, CWSP98]

Optimal w.r.t. the number of keys GC has to generate



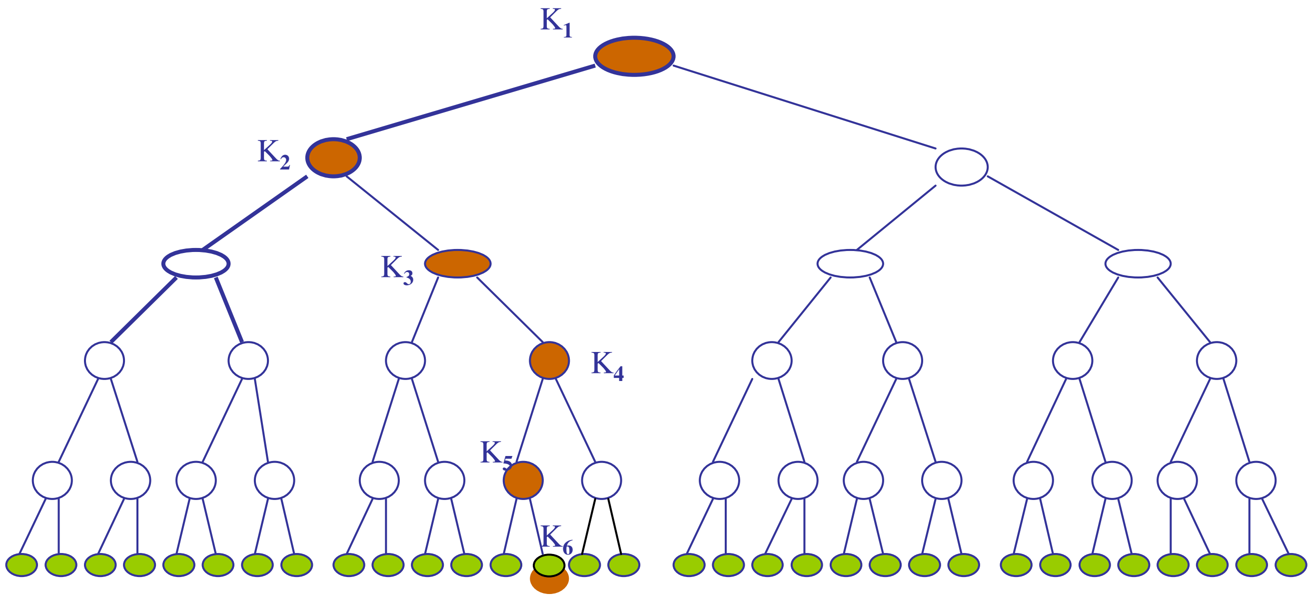
In any 1- secure key-tree the number of distinct keys is at least 2logn +1

LKH schemes
Key assignment

Sequence of key-trees: T_0, T_1 , T_2 , …

Theorem. A LKH scheme is secure w.r.t revoked users if in T_0 all keys associate to nodes **are distinct** and the join and revoke operations maintain such an invariant, i.e., at session j, for any j=1,2, …, all keys of T_j **are distinct among them and from all the previously used and deleted ones.**

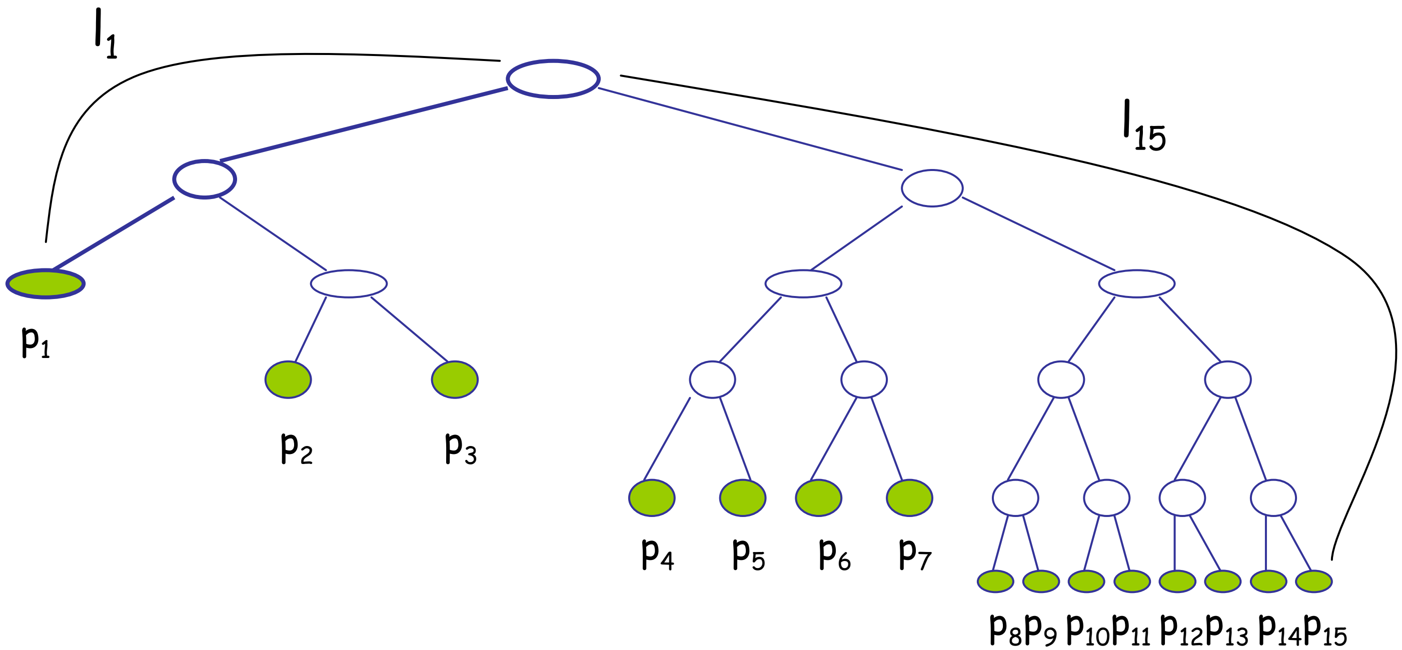
LKH schemes
Key assignment [PB01]



Problem 2

In presence of a non uniform probability distribution of user revocation, how to construct a tree which minimises the average numbers of keys a user has to store?

Minimising Average Lenght

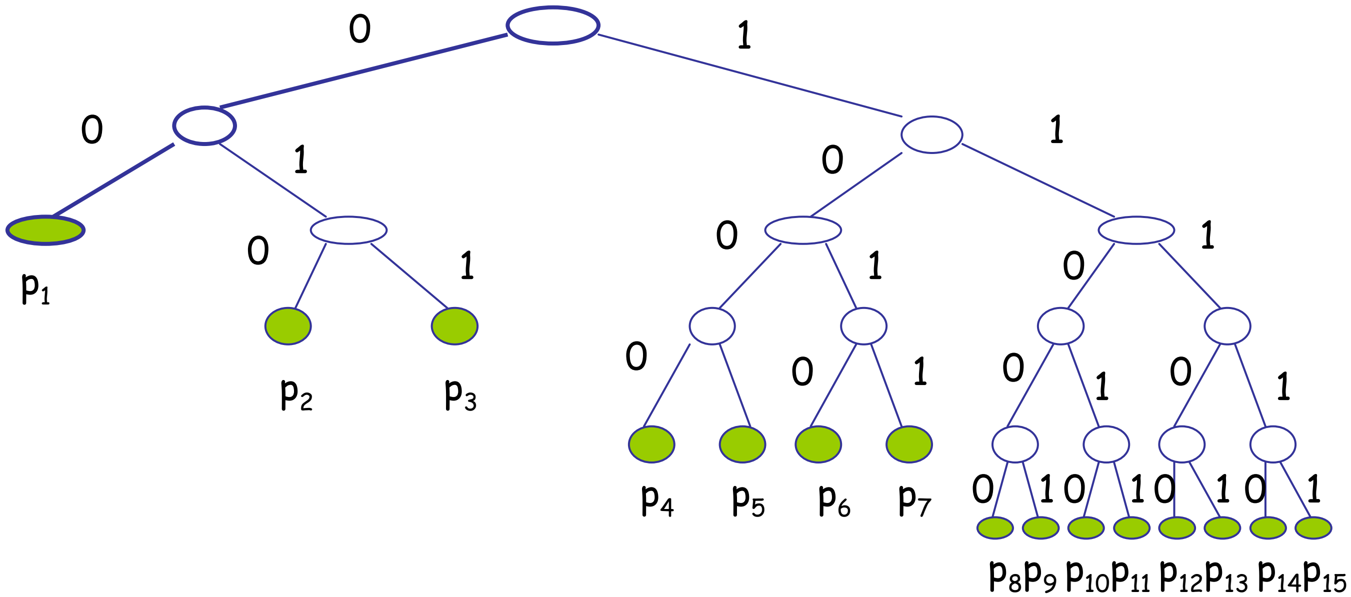


p_i = probability of revocation of user i

l_i = lenght of the path from the root to the leaf associated to user i

Problem: find lengths minimising $\sum_{i=1}^{n} p_i l_i$

Coding theory



Each path can be seen as a binary codeword associated to a leaf. Due to the structure of the tree, the set of codewords is prefix-free and the lengths satisfy Kraft's inequality

→ Huffman algorithm solves the problem!

SD and LSD

Optimization Criteria

$$\sum_{i=1}^{n} p_i l_i^2$$

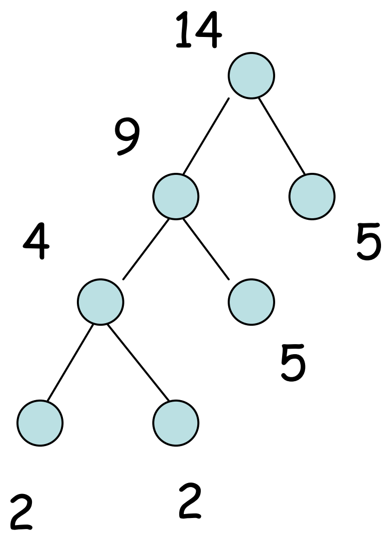
Subset Difference

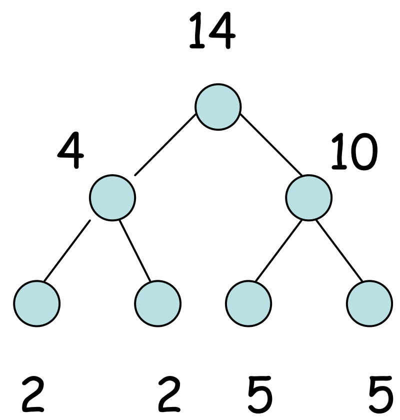
$$\sum_{i=1}^{n} p_i \frac{l_i^2}{\sqrt{\log n}}$$

Layered Subset Difference

Unfortunately, Huffman algorithm does not minimise the above measures

Huffman Algorithm





$\sum_{i=1}^{n} p_i l_i$ 2*3+2*3+5*2+5*1=21 2*2+2*2+5*2+5*2=24

$\sum_{i=1}^{n} p_i l_i^2$ $2*3^2+2*3^2+5*2^2+5*1^2=61$ $2*2^2+2*2^2+5*2^2+5*2^2=56$

Campbell's Penalties [Cam66]

Given a continuous (strictly) monotonic increasing cost function

$\varphi(l): R_+ \rightarrow R_+$ the value to minimise is

$$L(p,l,\varphi) = \varphi^{-1}\left(\sum_i p_i \varphi(l_i)\right)$$

The value L(p,l, φ) is the mean lenght for the cost function φ.
For brevity it is called "the penalty".

[Lar89] gave an algorithm for $\varphi(x)=\alpha x + \beta x^2$, with α and β non negative (time and space complexity $O(n^3)$).

Quasiarithmetic penalties [Bae06]

Definition 2: Let $f(l,p) : \mathbb{R}_+ \times [0,1] \rightarrow \mathbb{R}_+ \cup \{\infty\}$ be a function nondecreasing in l. Then

$$\tilde{L}(\boldsymbol{p}, \boldsymbol{l}, f) \triangleq \sum_{i \in \mathcal{X}} f(l_i, p_i) \tag{3}$$

is called a *generalized quasiarithmetic* penalty. Further, if f is convex in l, it is called a *generalized quasiarithmetic convex* penalty.

Generalisation of Campbell's problem (i.e., $f(l_i,p_i)=p_i\ \varphi(l_i)$)

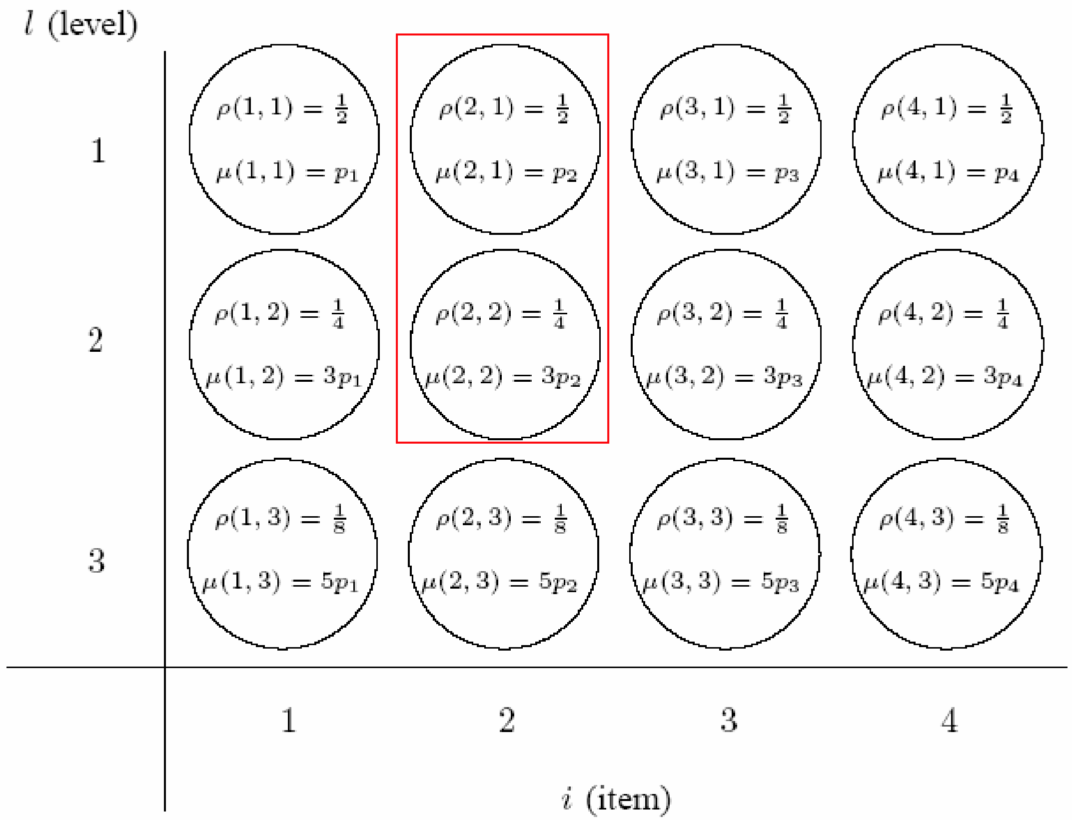
Algorithms for finding minimum penalties codes

- Nodeset Notation (alternative to tree notation)
- Each node (i,l) represents both the share of the penalties L(**p**,**l**,f) (weight) and the share of the Kraft sum k(l) (width).

Nodeset ass. to item i → First l_i nodes of column i

Nodeset associated to lenght distribution l → Union of nodesets ass. to the n items

To nodeset ass. to item i corresponds codeword c_i with lenght l_i



Coin Collector's Problem

Let 2^Z denote the set of all integer powers of two. The CC problem considers m coins with width $\rho_i \in 2^Z$ and weigth $\mu_i \in R$. The final problem parameter is t, the total width

$$\begin{array}{ll} \text{Minimize}_{\{B \subseteq \{1,\ldots,m\}\}} & \sum_{i \in B} \mu_i \\ \text{subject to} & \sum_{i \in B} \rho_i = t. \end{array}$$

We thus wish to choose coins with total width t such that the total weight is as small as possible.

The Package-Merge algorithm [Lar90] solves efficiently the problem

Reduction

Any optimal solution to the Coin Collector's Problem, represented by a subset of coins N, where the parameters are

- total width $t=n-1$
- width $\rho_i(i,l) = 2^{-l}$
- weight $\mu_i(i,l) = f(l,p_i) - f(l-1,p_i)$

is a nodeset for an optimal solution to the coding problem

Conclusions

- ✓ Analysis of LKH assignment scheme
- ✓ Characterization for key assignment
- ✓ Non-uniform probabilities of revocation: key-trees in SD and LSD and coding theory
- ✓ Efficient solutions available

Main References

[Bae06] M. Baer, *Source Coding for Campbell's Penalties*, IEEE Transactions on Information Theory, vol. 52, n. 10, 4380—4393, 2006.

[PB01] R. Poovendran and J. S. Baras. *An information theoretic analysis of rooted-tree based secure multicast key distribution schemes*. IEEE Transactions on Information Theory, 47(7):2824–2834, November 2001. Preliminary version in *Advances in Cryptology: Crypto '99*, Vol. 1666, pp. 624–638, Springer-Verlag, 1999.

[NNL01] D. Naor, M. Naor, and J. Lotspiech. *Revocation and tracing schemes for stateless receivers*. In Advances in Cryptology: Crypto'01, volume 2139 of *LNCS*, pages 41–62. Springer-Verlag, 2001.

[HS02] D. Halevy and A. Shamir. *The LSD broadcast encryption scheme*. In Advances in Cryptology— Crypto '02, volume 2442 of *LNCS*, pages 47–60, 2002.

Some of the above slides are from Moni Naor's presentation of [NNL01], available at http://www.wisdom.weizmann.ac.il/~naor/