# Rolling Boles, Optimal XML Structure Integrity For Updating Operations


**Sebastian Graf**
Distributed Systems Group
University of Konstanz
sebastian.graf@uni-konstanz.de

**Sebastian Kay Belle**
Distributed Systems Group
University of Konstanz
sebastian.kay.belle@uni-konstanz.de

**Marcel Waldvogel**
Distributed Systems Group
University of Konstanz
marcel.waldvogel@uni-konstanz.de


## Motivation

### Idea

+ Recursive structure integrity for XML
+ Suitable for updating operations on the tree-structure
− Fast computation results in weaker hash operation
− Equality of isomorphic structures

### Use Cases

I. REST-access on integrity-guarded resources
II. DOM-based Ajax for presenting flexible content
III. Guarding of distributed SOAP-requests

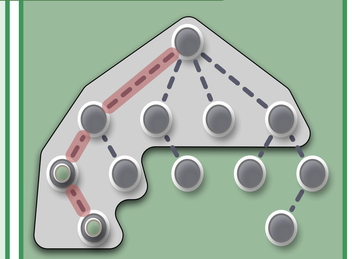
## Approach

### Rolling Boles

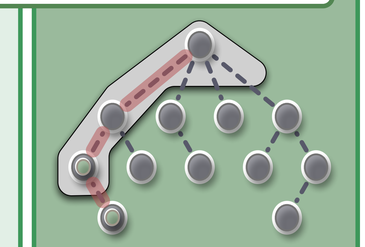
- Incremental Computation of recursive hash-values
- Only ancestors need to be read and written within single adaptions of the hash-structure
- No read of siblings related to adapted nodes necessary

### Insert Operation

DOMHash / Merkle-Hash



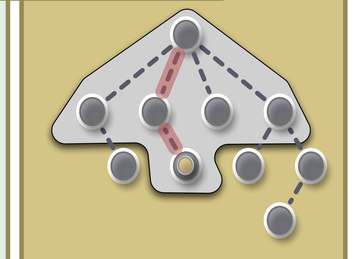
Rolling Boles / Incremental Hash



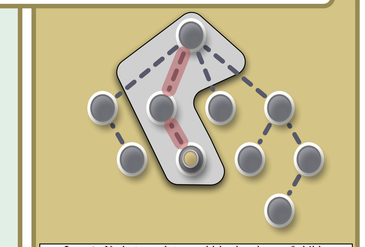
```
   Input: Node to insert i, parent node p to proceed i.
1  function:: rollingInsert (Node p, Node i) begin
2      oldHash ← p.check;
3      // update hash of p according to i
4      p.check ← p.check + i.check * PRIME;
5      newHash ← p.check;
6      // update ancestor(s), cf.  Alg.??.
7      update(p.parent, newHash, oldHash);
8      // attach the new node/sub-tree to p
9      p.attach(i);
```

### Update Operation

DOMHash / Merkle-Hash



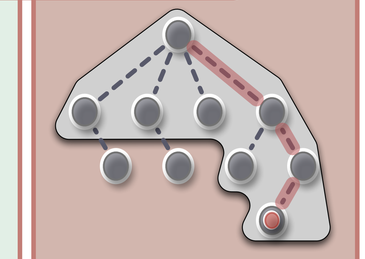
Rolling Boles / Incremental Hash



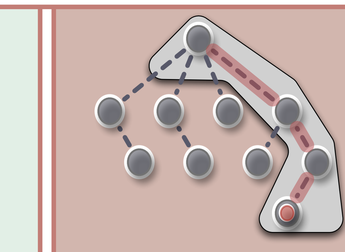
```
   Input: Node to update u, old hash value o of child, new
          hash value n of child.
1  function:: c (Node u, o ∈ ℤ, n ∈ ℤ) begin
2      if u ≠ NULL then
3          // compute the new check-sum according to
               the update information.
4          new ← u.check - o * PRIME;
5          new ← new + n * PRIME;
6          old ← u.check;
7          // update hash of u
8          u.check ← new;
9          // recursive invocation to update
               ancestor(s).
10         c(u.parent, new, old);
```

### Remove Operation

DOMHash / Merkle-Hash



Rolling Boles / Incremental Hash



```
   Input: Node to delete r, parent node p of r.
1  function:: rollingInsert (Node p, Node r) begin
2      newHash ← p.check - r.check * PRIME;
3      oldHash ← p.hash;
4      // update hash of p
5      p.hash ← newHash;
6      // update ancestor(s), cf.  Alg.??.
7      rollingUpdate(p.parent, newHash, oldHash);
8      // remove node/sub-tree from p
9      p.detach(r);
```

## Results

### Generating the Results

✓ Reference implementation in *Treetank* with DOM-like node-encoding
✓ Inserting of different instances and adaption of structure within all node insertions
✓ Comparing classic *Merkle–Hash* approaches and *Rolling Boles* based on time and hash collisions

### Collisions

#### Shakespeare XML

| Number | Name | % Collisions *Postorder* | % Collisions *Rolling Boles* |
|---|---|---|---|
| 1 | a_and_c | 21.89107 | 21.89107 |
| 2 | all_well | 21.73626 | 21.73626 |
| 3 | as_you | 21.20843 | 21.20843 |
| 4 | com_err | 22.27305 | 22.27305 |
| 5 | coriolan | 19.59637 | 19.59637 |
| 6 | cymbelin | 17.31237 | 17.31237 |
| 7 | dream | 17.31358 | 17.31358 |
| 8 | hamlet | 20.64356 | 20.64356 |
| 9 | hen_iv_1 | 18.01242 | 18.01242 |
| 10 | hen_iv_2 | 19.02951 | 19.02951 |
| 11 | hen_v | 16.57031 | 16.57031 |
| 12 | hen_vi_1 | 16.65624 | 16.65624 |
| 13 | hen_vi_2 | 17.28408 | 17.28408 |
| 14 | hen_vi_3 | 19.22263 | 19.22263 |
| 15 | hen_viii3 | 15.52216 | 15.52216 |
| 16 | j_caesar | 20.41749 | 20.41749 |
| 17 | john | 15.28960 | 15.28960 |
| 18 | lear | 21.35896 | 21.35896 |
| 19 | lll | 24.44469 | 24.44469 |
| 20 | m_for_m | 21.65987 | 21.65987 |
| 21 | m_wives | 24.78604 | 24.78604 |
| 22 | macbeth | 19.09541 | 19.09541 |
| 23 | merchant | 17.56916 | 17.56916 |
| 24 | much_ado | 23.60733 | 23.60733 |
| 25 | othello | 22.67203 | 22.67203 |
| 26 | pericles | 18.16571 | 18.16571 |
| 27 | r_and_j | 19.38435 | 19.38435 |
| 28 | rich_ii | 14.67171 | 14.67171 |
| 29 | rich_iii | 19.93816 | 19.93816 |
| 30 | shakespeare | 21.20786 | 21.20786 |
| 31 | t_night | 24.66138 | 24.66138 |
| 32 | taming | 21.76387 | 21.76387 |
| 33 | taming | 21.76387 | 21.76387 |
| 34 | tempest | 19.69146 | 19.69146 |
| 35 | timon | 20.66641 | 20.66641 |
| 36 | titus | 15.88324 | 15.88324 |
| 37 | troilus | 22.17166 | 22.17166 |
| 38 | two_gent | 24.63514 | 24.63514 |
| 39 | win_tale | 16.32369 | 16.32369 |

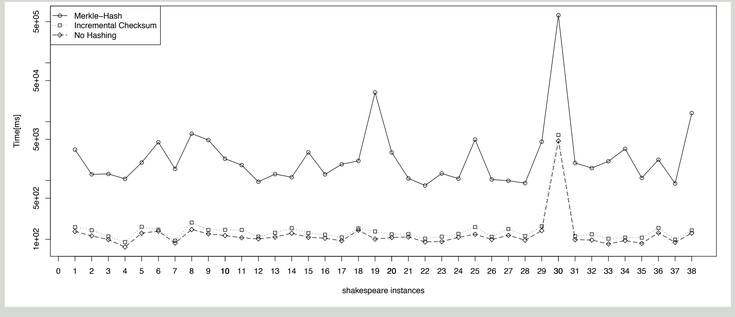
#### Music XML

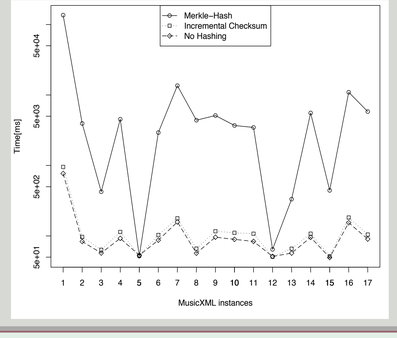
| Number | Name | % Collisions *Postorder* | % Collisions *Rolling Boles* |
|---|---|---|---|
| 1 | ActorPreludeSample | 92.32616 | 92.49873 |
| 2 | BeetAnGeSample | 83.65879 | 83.89860 |
| 3 | Binchois | 85.09955 | 85.80604 |
| 4 | BrookeWestSample | 89.80986 | 89.95715 |
| 5 | Chant | 71.37331 | 71.56673 |
| 6 | DebuMandSample | 84.51530 | 84.93678 |
| 7 | Dichterliebe01 | 88.94054 | 89.04707 |
| 8 | Echigo-Jishi | 82.42280 | 82.48219 |
| 9 | FaurReveSample | 88.20232 | 88.45713 |
| 10 | MahlFaGe4Sample | 85.45879 | 85.66096 |
| 11 | MozaChloSample | 86.08486 | 86.28868 |
| 12 | MozartPianoSonata | 87.28404 | 87.76780 |
| 13 | MozartTrio | 88.69742 | 89.10289 |
| 14 | MozaVeilSample | 85.67389 | 85.89381 |
| 15 | Saltarello | 90.71466 | 90.76682 |
| 16 | SchbAvMaSample | 89.71572 | 89.84950 |
| 17 | Telemann | 91.01628 | 91.17069 |

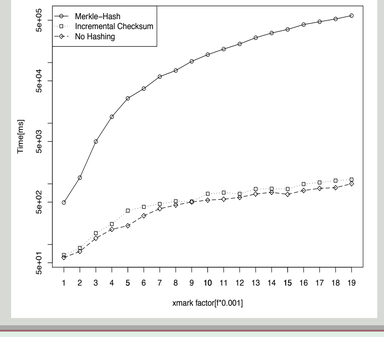
#### XMark

| f | % Collisions *Postorder* | % Collisions *Rolling Boles* |
|---|---|---|
| 0.001 | 19.36584 | 19.39993 |
| 0.002 | 23.44748 | 23.44748 |
| 0.003 | 26.42946 | 26.42946 |
| 0.004 | 25.59570 | 25.59570 |
| 0.005 | 26.87548 | 26.88239 |
| 0.006 | 26.27088 | 26.28290 |
| 0.007 | 26.77935 | 26.79440 |
| 0.008 | 27.55996 | 27.57756 |
| 0.009 | 27.77184 | 27.77580 |
| 0.010 | 29.41035 | 29.42065 |
| 0.011 | 30.07259 | 30.08550 |
| 0.012 | 29.30732 | 29.30732 |
| 0.013 | 29.55431 | 29.55704 |
| 0.014 | 30.14871 | 30.16368 |
| 0.015 | 30.46767 | 30.47471 |
| 0.016 | 29.94990 | 29.95853 |
| 0.017 | 29.22185 | 29.23830 |
| 0.018 | 30.08834 | 30.09822 |
| 0.019 | 30.02343 | 30.03449 |

### Time




Universität Konstanz