Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Nikolaus Binder and Alexander Keller, June, 2016
Efficient Hierarchy Traversal
Pruning/postponing nodes and backtracking
Efficient Hierarchy Traversal
Pruning/postponing nodes and backtracking

postponed nodes
Efficient Hierarchy Traversal
Pruning/postponing nodes and backtracking
Efficient Hierarchy Traversal

Pruning/postponing nodes and backtracking
Efficient Hierarchy Traversal
Pruning/postponing nodes and backtracking
Efficient Hierarchy Traversal

Pruning/postponing nodes and backtracking
Efficient Hierarchy Traversal
Pruning/postponing nodes and backtracking
Efficient Hierarchy Traversal
Comparing previous backtracking strategies
Efficient Hierarchy Traversal
Comparing previous backtracking strategies

Stack

Bit Trail

addr(3)
addr(5)

1
1
0
0
Efficient Hierarchy Traversal
Comparing previous backtracking strategies

Stack
- addr(3)
- addr(5)

Bit Trail
1
0
0
0
Efficient Hierarchy Traversal
Comparing previous backtracking strategies

Bit Trail

Stack
addr(3)
 addr(5)

Bit Trail

Bit Trail

Efficient Hierarchy Traversal
Comparing previous backtracking strategies

<table>
<thead>
<tr>
<th>State for book keeping (per ray)</th>
<th>Backtracking effort</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stack</td>
<td>$O(h(\text{tree}))$</td>
</tr>
<tr>
<td>Stackless, Backtracking from root</td>
<td>$O(1)$</td>
</tr>
<tr>
<td>Stackless, Backtracking with parents/siblings</td>
<td></td>
</tr>
</tbody>
</table>

- Stack: $O(h(\text{tree}))$ for state keeping, $O(1)$ for backtracking effort.
- Stackless, Backtracking from root: $O(1)$ for state keeping, $O(h(\text{tree}))$ for backtracking effort.
- Stackless, Backtracking with parents/siblings: $O(1)$ for both state keeping and backtracking effort.
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Building block 1: Using a bit trail, go to \( n^{th} \) uncle in constant time
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Building block 1: Using a bit trail, go to \( n^{th} \) uncle in constant time
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Building block 1: Using a bit trail, go to $n^{th}$ uncle in constant time
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Building block 1: Using a bit trail, go to n^{th} uncle in constant time
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Building block 1: Using a bit trail, go to $n^{th}$ uncle in constant time

Perfect Hash Map $h$: node key $k \mapsto$ node address $\text{addr}(k)$

- properties
  - no collisions
  - no need to store keys
  - lookup in constant time
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Building block 2: Two level hashing using an additional displacement table $D$

$$k \mapsto (k \mod |T| + D[k \mod |D|]) \mod |T|$$

[Tarjan, Yao 1979]
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Building block 2: Two level hashing using an additional displacement table $D$

$k \mapsto (k \mod |T| + D[k \mod |D|]) \mod |T|$

[Tarjan, Yao 1979]

$$S = \{1, 2, 3, 4, 5, 8, 9, 18, 19, 38, 39\}$$

$|S| = 11$

$|T| = 11 = |S| \Rightarrow$ minimal perfect hash table

$|D| = 8$
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Building block 2: Two level hashing using an additional displacement table $D$

$$k \mapsto (k \mod |T| + D[k \mod |D|]) \mod |T|$$  
[Tarjan, Yao 1979]

Greedy resolution in decreasing number of dependencies  
[Fox, Heath, Chen, and Daoud 1992]

$$S = \{1, 2, 3, 4, 5, 8, 9, 18, 19, 38, 39\}$$  
$|S| = 11$

$$|T| = 11 = |S| \Rightarrow \text{minimal perfect hash table}$$  
$|D| = 8$
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Building block 2: Two level hashing using an additional displacement table $D$

$k \mapsto (k \mod |T| + D[k \mod |D|]) \mod |T|$

Greedy resolution in decreasing number of dependencies

$S = \{1, 2, 3, 4, 5, 8, 9, 18, 19, 38, 39\}$

$|S| = 11$

$|T| = 11 = |S| \Rightarrow$ minimal perfect hash table

$|D| = 8$

[Tarjan, Yao 1979]

[Fox, Heath, Chen, and Daoud 1992]
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Building block 2: Two level hashing using an additional displacement table $D$

$k \mapsto (k \mod |T| + D[k \mod |D|]) \mod |T|$

Greedy resolution in decreasing number of dependencies

$S = \{1, 2, 3, 4, 5, 8, 9, 18, 19, 38, 39\}$

$|S| = 11$

$|T| = 11 = |S| \Rightarrow \text{minimal perfect hash table}$

$|D| = 8$

[Tarjan, Yao 1979]

[Fox, Heath, Chen, and Daoud 1992]
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Building block 2: Two level hashing using an additional displacement table $D$

$$k \mapsto (k \mod |T| + D[k \mod |D|]) \mod |T|$$

[Tarjan, Yao 1979]

Greedy resolution in decreasing number of dependencies

[Fox, Heath, Chen, and Daoud 1992]

<table>
<thead>
<tr>
<th>#</th>
<th>$D$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>+0</td>
</tr>
<tr>
<td>2</td>
<td>+0</td>
</tr>
<tr>
<td>3</td>
<td>+0</td>
</tr>
<tr>
<td>4</td>
<td>+0</td>
</tr>
<tr>
<td>5</td>
<td>+0</td>
</tr>
<tr>
<td>6</td>
<td>+0</td>
</tr>
<tr>
<td>7</td>
<td>+0</td>
</tr>
<tr>
<td>8</td>
<td>+0</td>
</tr>
<tr>
<td>9</td>
<td>+0</td>
</tr>
</tbody>
</table>

$S = \{1, 2, 3, 4, 5, 8, 9, 18, 19, 38, 39\}$

$|S| = 11$

$|T| = 11 = |S| \Rightarrow$ minimal perfect hash table

$|D| = 8$
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Building block 2: Two level hashing using an additional displacement table $D$

$k \mapsto (k \mod |T| + D[k \mod |D|]) \mod |T|$  \hspace{1cm} [Tarjan, Yao 1979]

Greedy resolution in decreasing number of dependencies  \hspace{1cm} [Fox, Heath, Chen, and Daoud 1992]

\[
\begin{array}{c|c|c}
\text{k mod } |T| & \text{k mod } |D| & S = \{1, 2, 3, 4, 5, 8, 9, 18, 19, 38, 39\} \\
\hline
1 & 1 & 8 \\
2 & 2 & 9 \\
3 & 18 & 8 \\
4 & 19 & 9 \\
5 & & \\
6 & & \\
7 & & \\
8 & & \\
9 & & \\
10 & & \\
11 & & \\
\end{array}
\]

\[
\begin{array}{c|c|c}
\text{D} & \text{S} & \text{T} \\
\hline
1 & 1 & 18 \\
2 & 2 & 19 \\
3 & 3 & \\
4 & 4 & \\
5 & 5 & \\
6 & 6 & \\
7 & 7 & \\
8 & 8 & \\
9 & 9 & \\
10 & 10 & \\
11 & 11 & \\
\end{array}
\]

$|S| = 11$

$|T| = 11 = |S| \Rightarrow \text{minimal perfect hash table}$

$|D| = 8$
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Building block 2: Two level hashing using an additional displacement table D

\[ k \mapsto (k \mod |T| + D[k \mod |D|]) \mod |T| \]

[Tarjan, Yao 1979]

Greedy resolution in decreasing number of dependencies

[Fox, Heath, Chen, and Daoud 1992]

\[ S = \{1, 2, 3, 4, 5, 8, 9, 18, 19, 38, 39\} \]

|\( |S| = 11 \) |
|\( |T| = 11 = |S| \Rightarrow \) minimal perfect hash table |
|\( |D| = 8 \) |
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Building block 2: Two level hashing using an additional displacement table $D$

$$k \mapsto (k \mod |T| + D[k \mod |D|]) \mod |T|$$

[Tarjan, Yao 1979]

Greedy resolution in decreasing number of dependencies

[Fox, Heath, Chen, and Daoud 1992]

$k \rightarrow (k \mod |T| + D[k \mod |D|]) \mod |T|$

<table>
<thead>
<tr>
<th>#</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>+2</td>
</tr>
<tr>
<td>2</td>
<td>+0</td>
</tr>
<tr>
<td>2</td>
<td>+0</td>
</tr>
<tr>
<td>2</td>
<td>+0</td>
</tr>
<tr>
<td>1</td>
<td>+0</td>
</tr>
<tr>
<td>1</td>
<td>+0</td>
</tr>
<tr>
<td>1</td>
<td>+0</td>
</tr>
<tr>
<td>1</td>
<td>+0</td>
</tr>
</tbody>
</table>

$S = \{1, 2, 3, 4, 5, 8, 9, 18, 19, 38, 39\}$

$|S| = 11$

$|T| = 11 = |S| \Rightarrow$ minimal perfect hash table

$|D| = 8$

$k \mod |T|$

$S = \{1, 2, 3, 4, 5, 8, 9, 18, 19, 38, 39\}$

$|S| = 11$

$|T| = 11 = |S| \Rightarrow$ minimal perfect hash table

$|D| = 8$
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Building block 2: Two level hashing using an additional displacement table $D$

$$k \mapsto (k \mod |T| + D[k \mod |D|]) \mod |T|$$

[Tarjan, Yao 1979]

Greedy resolution in decreasing number of dependencies

[Fox, Heath, Chen, and Daoud 1992]

$$S = \{1, 2, 3, 4, 5, 8, 9, 18, 19, 38, 39\}$$

$|S| = 11$

$|T| = 11 = |S| \Rightarrow \text{minimal perfect hash table}$

$|D| = 8$
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Building block 2: Two level hashing using an additional displacement table $D$

$$k \mapsto (k \mod |T| + D[k \mod |D|]) \mod |T|$$  \hfill [Tarjan, Yao 1979]

Greedy resolution in decreasing number of dependencies  \hfill [Fox, Heath, Chen, and Daoud 1992]

$k \mapsto (k \mod |T| + D[k \mod |D|]) \mod |T|$

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

$S = \{1, 2, 3, 4, 5, 8, 9, 18, 19, 38, 39\}$

$|S| = 11$

$|T| = 11 = |S| \Rightarrow$ minimal perfect hash table

$|D| = 8$
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Building block 2: Two level hashing using an additional displacement table D

\[ k \mapsto (k \mod |T| + D[k \mod |D|]) \mod |T| \]

Greedy resolution in decreasing number of dependencies

\[ |S| = 11 \]
\[ |T| = 11 = |S| \Rightarrow \text{minimal perfect hash table} \]
\[ |D| = 8 \]
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Building block 2: Two level hashing using an additional displacement table $D$

$$k \mapsto (k \mod |T| + D[k \mod |D|]) \mod |T|$$

[Tarjan, Yao 1979]

Greedy resolution in decreasing number of dependencies

[Fox, Heath, Chen, and Daoud 1992]

$$S = \{1, 2, 3, 4, 5, 8, 9, 18, 19, 38, 39\}$$

$$|S| = 11$$

$$|T| = 11 = |S| \Rightarrow \text{minimal perfect hash table}$$

$$|D| = 8$$
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Building block 2: Two level hashing using an additional displacement table $D$

$$k \mapsto (k \mod |T| + D[k \mod |D|]) \mod |T|$$

Greedy resolution in decreasing number of dependencies

[Tarjan, Yao 1979]

[Fox, Heath, Chen, and Daoud 1992]
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Building block 2: Two level hashing using an additional displacement table $D$

$$k \mapsto (k \mod |T| + D[k \mod |D|]) \mod |T|$$

[Tarjan, Yao 1979]

Greedy resolution in decreasing number of dependencies

[Fox, Heath, Chen, and Daoud 1992]

$S = \{1, 2, 3, 4, 5, 8, 9, 18, 19, 38, 39\}$

$|S| = 11$

$|T| = 11 = |S| \Rightarrow$ minimal perfect hash table

$|D| = 8$
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Building block 2: Two level hashing using an additional displacement table $D$

$$k \mapsto (k \mod |T| + D[k \mod |D|]) \mod |T|$$  
[Tarjan, Yao 1979]

Greedy resolution in decreasing number of dependencies  
[Fox, Heath, Chen, and Daoud 1992]

$$S = \{1, 2, 3, 4, 5, 8, 9, 18, 19, 38, 39\}$$
$$|S| = 11$$
$$|T| = 11 = |S| \Rightarrow \text{minimal perfect hash table}$$
$$|D| = 8$$
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Building block 2: Two level hashing using an additional displacement table $D$

$$k \mapsto (k \mod |T| + D[k \mod |D|]) \mod |T|$$

Greedy resolution in decreasing number of dependencies

$[\text{Tarjan, Yao 1979}]$

$[\text{Fox, Heath, Chen, and Daoud 1992}]$

$S = \{1, 2, 3, 4, 5, 8, 9, 18, 19, 38, 39\}$

$|S| = 11$

$|T| = 11 = |S| \Rightarrow \text{minimal perfect hash table}$

$|D| = 8$
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Building block 2: Two level hashing using an additional displacement table \( D \)

\[
k \mapsto (k \mod |T| + D[k \mod |D|]) \mod |T|
\]

[Tarjan, Yao 1979]

Greedy resolution in decreasing number of dependencies

[Fox, Heath, Chen, and Daoud 1992]

\[
S = \{1, 2, 3, 4, 5, 8, 9, 18, 19, 38, 39\}
\]

\(|S| = 11\)

\(|T| = 11 = |S| \Rightarrow \text{minimal perfect hash table}\)

\(|D| = 8\)
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Building block 2: Two level hashing using an additional displacement table \( D \)

\[ k \mapsto (k \mod |T| + D[k \mod |D|]) \mod |T| \]

Greedy resolution in decreasing number of dependencies

[Tarjan, Yao 1979]

\[ |S| = 11 \]

\[ |T| = 11 = |S| \Rightarrow \text{minimal perfect hash table} \]

\[ |D| = 8 \]

\[ S = \{1, 2, 3, 4, 5, 8, 9, 18, 19, 38, 39\} \]
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Building block 3: Reducing the number of hash lookups

- backtracking statistics
  - to sibling: 27%
  - to uncle: 15%
  - to grand uncle: 15%

\[
\begin{align*}
\text{around 57% altogether}
\end{align*}
\]
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Building block 3: Reducing the number of hash lookups

- backtracking statistics
  - to sibling: 27%
  - to uncle: 15%
  - to grand uncle: 15%

- store references to uncle and grand uncle in node
  - in unused padding space
  - data loaded anyway

around 57% altogether
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Building block 3: Reducing the number of hash lookups

- backtracking statistics
  - to sibling: 27%
  - to uncle: 15%
  - to grand uncle: 15%

  \[
  \text{around 57\% altogether}
  \]

- store references to uncle and grand uncle in node
  - in unused padding space
  - data loaded anyway

- store most recently postponed node in a register
  - always used for transitions to siblings
  - similar to a short stack, but more powerful
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Building block 4: Avoid pointless backtracking

- subtrees behind intersection may not always be culled
  - due to overlapping bounding boxes
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Building block 4: Avoid pointless backtracking

- subtrees behind intersection may not always be culled
  - due to overlapping bounding boxes
- discard levels with disjoint $t$-intervals
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Building block 4: Avoid pointless backtracking

- subtrees behind intersection may not always be culled
  - due to overlapping bounding boxes

- discard levels with disjoint $t$-intervals
  - cheap
    - no $t_0$ values stored
    - mask with one bit per level
    - bit set to one if overlapping, zero if disjoint
    - bitwise \texttt{and} with bit trail after intersection has been found
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

**Building block 4: Avoid pointless backtracking**

- subtrees behind intersection may not always be culled
  - due to overlapping bounding boxes

- discard levels with disjoint \( t \)-intervals
  - cheap
    - no \( t_0 \) values stored
    - mask with one bit per level
    - bit set to one if overlapping, zero if disjoint
    - bitwise and with bit trail after intersection has been found
  - compromise
    - cannot account for intersections outside overlap
Building block 5: Resuming traversal in last node node instead of starting at the root

- **pause:** state (key and bit trail) must be stored

- **resume:** start in last node, set bit trail to
  - previous bit trail if same ray origin and direction
    - transparent/translucent object, cut outs
  - 1 for all levels above current level if ray origin or direction has changed
    - tracing paths
    - refraction
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Summary

- optimized stackless traversal
  - backtracking in constant time by perfect hashing
  - reduced number of hash lookups
  - store references to uncles and grand uncles in nodes
  - store most recently postponed node in a register

- additional building blocks currently not used in software (e.g. due to register pressure)
  - discard unreachable postponed nodes
  - pause and resume traversal in current node

- exhaustive tests
  - many different and freely available scenes
  - various practical camera positions
  - different ray types
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Summary

- optimized stackless traversal
  - backtracking in constant time by perfect hashing
  - reduced number of hash lookups
    - store references to uncles and grand uncles in nodes
    - store most recently postponed node in a register
- additional building blocks currently not used in software (e.g. due to register pressure)
  - discard unreachable postponed nodes
  - pause and resume traversal in current node

exhaustive tests
- many different and freely available scenes
- various practical camera positions
- different ray types
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Summary

- optimized stackless traversal
  - backtracking in constant time by perfect hashing
  - reduced number of hash lookups
    - store references to uncles and grand uncles in nodes
    - store most recently postponed node in a register

- additional building blocks currently not used in software (e.g. due to register pressure)
  - discard unreachable postponed nodes
  - pause and resume traversal in current node

- exhaustive tests
  - many different and freely available scenes
  - various practical camera positions
  - different ray types
Efficient Stackless Hierarchy Traversal with Backtracking in Constant Time

Results: Performance in M rays/s, NVIDIA Titan X, for Primary/Shadow/Diffuse Rays

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Primary</td>
<td>Shadow</td>
<td>Diffuse</td>
</tr>
<tr>
<td>Armadillo</td>
<td>837</td>
<td>236</td>
<td>214</td>
</tr>
<tr>
<td>Conference</td>
<td>786</td>
<td>399</td>
<td>253</td>
</tr>
<tr>
<td>Dragon</td>
<td>743</td>
<td>212</td>
<td>194</td>
</tr>
<tr>
<td>Emily</td>
<td>676</td>
<td>254</td>
<td>234</td>
</tr>
<tr>
<td>Buddha</td>
<td>1237</td>
<td>210</td>
<td>185</td>
</tr>
<tr>
<td>Hairball</td>
<td>190</td>
<td>77</td>
<td>65</td>
</tr>
<tr>
<td>Enchanted Forest</td>
<td>237</td>
<td>81</td>
<td>64</td>
</tr>
<tr>
<td>San-Miguel</td>
<td>246</td>
<td>149</td>
<td>81</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
We are hiring.

akeller@nvidia.com