Gödel
Hashing

matt.might.net
@mattmight
Disclaimer
“simple, fun idea”
“simple, fun idea”

“works well in practice,”
“simple, fun idea”

“works well in practice,”

“but theory says it will not.”
An old problem
An older solution
A big impact
An old problem
“CFA is slow!”
An older solution
Gödel hashing

- functional
- monotonic
- compact
- dynamic
- incremental
- perfect

*Inspired by a true theorem.*
Word-level parallelism!
Great cache behavior!
A big impact
Minutes of work
2x
Motivation
(f x)
\( f(x) \)
What is \( f \)?
Why not run the program?
$e \rightarrow O \rightarrow O$
What is $f$, here?
What is $f$, here?
Problem
\hat{S}_1 \subseteq \hat{S}_2
\hat{s}_1 = (e, \hat{\rho}, \hat{\sigma}, \hat{\kappa})
\[ \hat{\xi}_1 = (e, \hat{\rho}, \hat{\sigma}, \hat{\kappa}) \]
\[ \hat{\text{\(s_1\)}} = (e, \hat{\rho}, \hat{\sigma}, \hat{\kappa}) \]
\[ \hat{\sigma} : \overbrace{Addr} \rightarrow \mathcal{P}(\overbrace{Value}) \]
<table>
<thead>
<tr>
<th>Addr</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>010000101</td>
</tr>
<tr>
<td></td>
<td>000001100</td>
</tr>
<tr>
<td></td>
<td>000101000</td>
</tr>
<tr>
<td></td>
<td>010000100</td>
</tr>
<tr>
<td></td>
<td>100000000</td>
</tr>
<tr>
<td></td>
<td>000100100</td>
</tr>
<tr>
<td></td>
<td>000001111</td>
</tr>
<tr>
<td></td>
<td>001000100</td>
</tr>
</tbody>
</table>
First: Hash sets
Prime decomposition
\( p_3 \quad p_4 \)
$p_3 \times p_4$
$A \subseteq B$
\[ [B] \mod [A] = 0 \]
\( A \cap B \)
\text{gcd}(\llbracket A \rrbracket, \llbracket B \rrbracket)
lcm(\langle A \rangle, \langle B \rangle)
A \cup \ U \ B
A \ - \ B
\[ A ] / \gcd([A],[B]) \]
\[ n = p_1^{m_1} p_2^{m_2} p_3^{m_3} \ldots \]
$n = \bigcup \{ \text{\color{green}{\Large \bullet}}, \text{\color{blue}{\Large \bullet}}, \text{\color{red}{\Large \bullet}} \}$
x \sqsubseteq y
lcm(\[ x \], \[ y \])
\[ x \subseteq y \]
\[ \lfloor y \rfloor \mod \lfloor x \rfloor = 0 \]
x ∩ y
\( \text{gcd}(\lfloor x \rfloor, \lfloor y \rfloor) \)
But, does it work for CFA?
\( \hat{\sigma} : \overline{Addr} \rightarrow \mathcal{P}(\overline{Value}) \)
\{ \hat{a}_1, \hat{a}_2 \} \\
\{ \hat{v}_1, \hat{v}_2 \}
\[
\begin{align*}
\hat{a}_1 & \mapsto \{ \hat{v}_2 \} \\
\hat{a}_2 & \mapsto \{ \hat{v}_1 \} \\
\hat{a}_2 & \mapsto \{ \hat{v}_2 \} \\
\hat{a}_1 & \mapsto \{ \hat{v}_1 \}
\end{align*}
\]
$L_1$ has a prime basis.

$L_2$ has a prime basis.
$L_1 \times L_2$ has a prime basis.
$L_1 + L_2$ has a prime basis.
$X \rightarrow L_2$ has a prime basis.
What else?
\[ \left[ \{ a^n, b^m \} \right]\]
\[(a^n) \cdot (b^m)\]
A \subseteq B
$[B] \ mod \ [A] = 0$
A \cup B
\[ [A] \times [B] \]
\[ [a] \quad [b] \quad [c] \]
\[ p_1 \quad p_2 \quad p_3 \]
Wait a minute...
gcd is $O(n^2)$
mod is $O(n^2)$
How is this more efficient?
Flow sets are sparse.
99% of flow sets: < 5 values
Median flow set: 2 values
Primes are dense.
$U \ln U$
1,000,000 abstract values?
23 bit prime
Most flow sets fit in a word.
Most of the time, $n = 1$. 
If not, great locality.
4-6%
100x
Programming is about making choices.
3 E’s
Elegance
Efficiency
Efficacy
Programmers:
Pick any two
Functional Programmers:
Pick any three
Questions?
Algebraic data types?
deriving (Hashable)
$p_1 \ p_2 \ p_3 \ p_4 \ p_5$