Accelerating the Evolution of Evolvable Hardware-based Packet Classifiers

Harald Widiger, Ralf Salomon, Andreas Tockhorn
University of Rostock
Institute of Applied Microelectronics and Computer Engineering
Outline

- Packet-Classification Problem
- Adaptive Hashing in Hardware
- Classifier System
- Fitness Evaluation Acceleration
  - Early Termination
  - Parallel Fitness Evaluation
  - Memory Interleaving
- Conclusion
Classification Problem
Classification Problem

<table>
<thead>
<tr>
<th>Key</th>
<th>Information (Rule)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>N</td>
</tr>
</tbody>
</table>

Memory Access

Mapping Function (Memory Search)

Key (Header)

Information

Header Parser

Functional Element

Frame In

Frame Out

03.04.2007 University of Rostock
Adaptive Hashing in Hardware

- Collision: $X \neq Y ; H(X) = H(Y)$
- Resolution
  - Rehashing $H(H(X))$
  - Linear $H(X) + \text{Prime}$
- Time Complexity: $O(1)$
- Memory Space: $O(N)$
Adaptive Hashing in Hardware

Genome: $M \cdot 2 \cdot \log_2(N)$
Classifier System

- Frame In ➔ Key Parser ➔ Buffer ➔ Frame Out

- Key

- Switch

- Hash Func0 ➔ MemIF0

- External SRAM ➔ Arbiter

- SRAM

- Hash Func1 ➔ MemIF1

- Hardware Evolution ➔ Keys

- Classification Rule

03.04.2007  University of Rostock
Genetic Algorithm – Implementation in Hardware

- μ Individuals
- λ Offspring
- Mutation Rate: 1/N
- Survivor Selection: μ new parents out of λ offspring and fittest old parent; (λ,μ)-elitist
Fitness Evaluation

- Fitness Value ↔ Number of Collisions
  - Fitness Reversed → 0 = perfekt
  - Hash all existing keys
  - $H(\text{Key})$ used in the memory?
    - $H(\text{Key}) = H(\text{Key}) + \text{prime}$
    - $F++$
Performance – Random Data

(1+6)-GA with Random Keys

Fitness

0 200 400 600 800 1000

Number of Generations
Performance – Real World Data

(1+6)-GA with Real World Data: Blow up

Fitness vs. Number of Generations
Fitness Evaluation

\[ T_i = \sum_{j=1}^{k} \tau \cdot (1 + \text{coll}_j) \]

\[ T = \sum_{i=1}^{\lambda} T_i + T_{\text{reconf}} \]
Fitness Evaluation - Speed

Size: 5416 slices

Time: 251 sec
Early Termination (ET)

\[ T = \sum_{i=0}^{\mu} T_i + T_{reconf} + (\lambda - \mu) \cdot T_\mu \]
Early Termination - Speed

(4,12)-GA with Early Termination

- Size:
  - 5592 slices
  - +3.35%

- Time:
  - 200 sec
  - +25%
Parallel Fitness Evaluation (PFE)

\[ \sum_{i=1}^{k} \sum_{j=1}^{\lambda} \text{MAX}(T_{i,j}) + T_{\text{Reconf}} \]
Parallel Fitness Evaluation (PFE)

- **Size (4x):**
  - 7203 slices
  - +32.99%

- **Time (4x):**
  - 219 sec
  - +15%
Memory Interleaving (MI)

\[ T_i^n = \sum_{j=1}^{k} \tau \cdot \left( 1 + \frac{\text{coll}_j}{n} \right) \]
Memory Interleaving (MI)

(4,12)-GA with Memory Interleaving

- Size (4x):
  - 5572
  - +2.88%

- Time (4x):
  - 108 sec
  - +132%
Combination

(4,12)-GA with Combined Acceleration Approaches

- Size
  - 7473 slices
  - 37.98%

- Time
  - 85 sec
  - 230%
## Synopsys

<table>
<thead>
<tr>
<th>Module</th>
<th>Slices</th>
<th>Increase</th>
<th>Evolution Time [sec]</th>
<th>Speed Gain [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original</td>
<td>5416</td>
<td>-</td>
<td>251</td>
<td>-</td>
</tr>
<tr>
<td>ET</td>
<td>5592</td>
<td>3.25</td>
<td>200</td>
<td>25</td>
</tr>
<tr>
<td>2xPFE</td>
<td>5945</td>
<td>9.77</td>
<td>223</td>
<td>12</td>
</tr>
<tr>
<td>4xPFE</td>
<td>7203</td>
<td>32.99</td>
<td>219</td>
<td>15</td>
</tr>
<tr>
<td>2xMI</td>
<td>5531</td>
<td>2.12</td>
<td>152</td>
<td>67</td>
</tr>
<tr>
<td>4xMI</td>
<td>5572</td>
<td><strong>2.88</strong></td>
<td>108</td>
<td><strong>132</strong></td>
</tr>
<tr>
<td>Comb</td>
<td>7473</td>
<td>37.98</td>
<td>85</td>
<td>230</td>
</tr>
</tbody>
</table>
Conclusion

- Working implementation of an evolvable Packet Classifier (7400 Slices, 125 MHz) with a real world performance of 10 million classifications per second

- Increase of evolution speed
  - Early Termination → 25%
  - Parallel Fitness Evaluation → 15%
  - Memory Interleaving → 132%
  - Combination → 230%

- Future Research:
  - Alternative architectures for hash functions
  - Adaptive fitness evaluation, and mutation rate
  - Cashing functionalities in the data path