Genetic Programming of Process Decomposition Strategies for Evolvable Hardware
Outline

- Genetic Programming (GP)
- Evolvable Hardware (EH)
- Hardware Evolution of GP Trees
- Context Switching
- Implementation and Experiments
- Conclusion and Future Work
Genetic Programming (GP)

This is a program for a mobile robot

Sample GP Tree

+ Automatic evolution of computer programs
+ Tree-structured chromosomes
+ Expressive power
+ Generality
+ Easy to incorporate prior knowledge

– Time complexity (fitness evaluation)
– Space complexity
Evolvable Hardware (EH)

- Run-time reconfigurability
- Higher performance than general-purpose processors
- More flexible than ASICs
- On-line learnability
- Customization
Hardware Evolution of GP Trees

1. Define Primitive Symbols
2. Initialization of Trees
3. Conversion to Bitstreams
4. Apply Genetic Operators
5. Fitness Evaluation on EH
6. Termination Condition?
   - Yes: Resulting Program Tree
   - No: New Trees
Difficulty in Hardware Evolution of GP Trees

- On-chip representation of tree structures
- Routing problem
- Hardware resource utilization
First Approach: Linear Representation

- Comparison of resource utilization

<table>
<thead>
<tr>
<th></th>
<th>Tree Representation</th>
<th>Linear Representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Resource</td>
<td>80.9%</td>
<td>86.7%</td>
</tr>
<tr>
<td>Placement Resource</td>
<td>37.8%</td>
<td>49.2%</td>
</tr>
</tbody>
</table>
Second Approach: Context Switching

- Separate implementation of subtrees of a GP tree
- Reduction of redundancy
- Better utilization of chip resources
Context Switching: Process Decomposition

- A GP Tree is decomposed into a number of sub-trees.
- Based on the resource necessity of sub-trees.
- Number of nodes in the sub-tree
Context Switching: Hardware Library

- Library: storage of hardware configuration information for subtrees
- Nodes of a GP tree are interchangeable
- Drawback
  - Possibility of resource waste

Illustration of hardware library
Training of GP Trees with Context Switching

- A sub-tree is chosen depending on the input values.
- Then, this sub-tree is trained by varying and selecting fitter subtrees.
Test Bed: Evolving Controllers for Autonomous Robots

- Transportation of an object to the goal (light)
- Cooperation of two robots
- Khepera
- Xilinx XC6216
Setup of GP for Robot Control

- **Function nodes**
  - `IF_OBJ, IF_GOAL, IF_FORWARD, IF_OBS1~4`

- **Terminal nodes**
  - `MOVE_FORWARD, MOVE_FORWARD & TURN_LEFT, MOVE_FORWARD & TURN_RIGHT, MOVE_BACKWARD,TURN_LEFT, TURN_RIGHT, RANDOM`

- **Fitness function**
  
  (1) \( F_{\text{new}} = F_{\text{old}} + w_1 \times (#\text{collisions}) + w_2 \times (#\text{steps}) \)

  (2) \( F_{\text{new}} = F_{\text{old}} + w_1 \times (#\text{miss}) + w_2 \times (#\text{steps}) + \text{vision} \times w_3 \)
Experimental Results: Best Fitness
Experimental Results: Average Fitness
Experimental Results: Hits

![Graph showing hits vs number of evaluations]
Conclusion and Future Work

- Presented a method for evolving GP trees on EH
- Speed-up by reducing fitness evaluation time
- The larger the training set, the higher the speed-up factor
- Possibility of special-purpose GP hardware
- Appropriate for on-line hardware learning
- Further possibility of resource utilization in evolving GP trees on EH
Sources

H.-S. Seok, K.-J. Lee, B.-T. Zhang
Seoul National University
Seoul, Korea

D.-W. Lee and K.-B. Sim
Chung-Ang University
Seoul, Korea