Multi-Objective Hardware/Software Co-Exploration

Cristina Silvano
silvano@elet.polimi.it
Politecnico di Milano

Co-Exploration

- Multi-Objective Co-Exploration of Source Code Transformation and Design Space Architectures for Low-Power Embedded Systems
- The System-Level Design Space is defined as:
  \[ S_{SL} = S_T \times S_A \]
  where \( S_T \) is the Source Code Transformation Design Space and \( S_A \) is the Architectural Design Space
Co-Exploration Frameworks

- Analysis of the influence of compiler on system power:
  - *Impact of source code transformation* on system power [Brandolese et al. JCSC-03]
  - *Influence of compiler optimizations* on system power [Kandemir et al. TVLSI-01]
- *Architecture/Compiler CoExploration* for ASIPs [Fisher et al. CASES-02]
- Architectural and Compiler techniques for energy reduction [Bellas et al. TVLSI-00]

Examples of source code transformations

- **Loop Unrolling**: Replicates the body of a loop
  - Advantages: ↑ Parallelism ↓ Loop overhead
  - Disadvantages: ↑ Code size, I-cache miss
- **Loop Tiling**: Brakes the accesses to matrices into accesses to sub-matrices
  - Advantages: ↓ D-cache miss
  - Disadvantages: ↑ Code size, branching
- **Function Inlining**: Replace the most frequently invoked function with the body of the function
  - Advantages: ↑ Spatial locality ↓ Number of function calls
  - Disadvantages: ↑ Code size
Multi-Objective Optimization Problem: Solutions Comparison

- Is A-Curve better than B-Curve? How much?
- Different techniques are used to compare different solutions of Multi-Objective Optimisation (MOO) problem. A critical survey is presented by Okabe et al. in IEEE Congress on Evolutionary Computation
Two-Set Coverage Metric

- To compare Pareto curves, we used the TWO-SET COVERAGE metric (C). C maps the ordered pair \((X',X'')\) to the \([0,1]\) interval. \(X'\) and \(X''\) are two different set of Pareto points

\[
C(X',X'') = \frac{|\left\{ \alpha'' \in X'' ; \exists \alpha' \in X' : \alpha' \succeq a'' \right\}|}{|X''|}
\]

where \(\alpha' \succeq a''\) iff \(\bar{f}(\alpha') \leq \bar{f}(\alpha'')\)

- Note that \(C(X',X'')\) can be different from \(C(X'',X')\)

Two-Set Coverage: Examples

1. \(C(A,B)=1 ; C(B,A)=0\)
2. \(C(A,B)<1 ; C(B,A)<1\)
3. \(C(A,B)<1 ; C(B,A)=0\)
**Experimental Results**

- Results obtained by applying the *Pareto Simulated Annealing Flow* (PSA) [PATMOS03] to the target system architecture based on a processor with configurable parameters and *Loop Unrolling* transformation.
- Results compared three different exploration strategies by evaluating the same number of simulation point:
  - **Two-Phase:**
    - $S_A + S_T$ Exploration
    - $S_T + S_A$ Exploration
  - **Co-Exploration** $S_T \times S_A$
- The chosen set of benchmarks is composed by a set of DCT transforms and FIR filters as well as other numerical algorithms written in C language.
- Architectural design space composed of more than 190000 system configurations for each benchmark.

**Experimental Results: LUEQ**

<table>
<thead>
<tr>
<th></th>
<th>$S_A + S_T$</th>
<th>$S_T + S_A$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C(C_0,TP)$</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$C(TP,C_0)$</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Experimental Results: FIR-1

\[ S_1 + S_2 \quad S_1 + S_2 \]

<table>
<thead>
<tr>
<th>C(C_0,TP)</th>
<th>C(TP,C_0)</th>
<th>C(C_0,TP)</th>
<th>C(TP,C_0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.96</td>
<td>0</td>
<td>0.82</td>
<td>0</td>
</tr>
</tbody>
</table>

Experimental Results: DCT-IDCT

\[ S_1 + S_2 \quad S_1 + S_2 \]

<table>
<thead>
<tr>
<th>C(C_0,TP)</th>
<th>C(TP,C_0)</th>
<th>C(C_0,TP)</th>
<th>C(TP,C_0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.07</td>
<td>0.7</td>
<td>0.09</td>
<td>0.8</td>
</tr>
</tbody>
</table>
Conclusions and On-Going Works

- The work presents a Co-Exploration methodology supporting the design of Microprocessor-based systems
- Experimental results have shown that on average better energy-delay Pareto points can be reached by the Co-Exploration with respect to the Two-Phase separate explorations
- On-going works:
  - Application of evolutionary algorithms for the multi-objective exploration to the framework
  - Definition of efficient pruning techniques to reduce the search effort
  - Implementation and analysis of other source code transformations