Hardware-Software Codesign

7. Design Space Exploration

Lothar Thiele
System Design

specification -> system synthesis

system synthesis

SW-compilation <- intellectual prop. code

SW-compilation -> instruction set

instruction set ->dashed line -> intellectual prop. block

HW-synthesis

estimation

estimation

machine code

net lists
Optimization-Analysis Cycle

**Optimization Algorithm**

make decisions only by knowing (and comparing) $f$
Three Examples
Example 1: Remember ...

Definition: A specification graph is a graph $G_S=(V_S,E_S)$ consisting of a data flow graph $G_P$, an architecture graph $G_A$, and edges $E_M$. In particular, $V_S=V_P \cup V_A$, $E_S=E_P \cup E_A \cup E_M$.
Example 1: Remember …

**Definition:** Given a specification graph $G_S$ an **implementation** is a triple $(\alpha, \beta, \tau)$, where $\alpha$ is a feasible allocation, $\beta$ is a feasible binding, and $\tau$ is a schedule.
Example 1: Simple Mapping Model

**search algorithm**

EA

1. selection
2. recombination
3. mutation

“chromosome” = encoded allocation + binding

**solutions**

- allocation
- binding

**analysis of individual solutions**

- decode allocation
- decode binding
- scheduling

fitness evaluation

fitness

user constraints

binding $\beta$  scheduling $\tau$  allocation $\alpha$

design point (implementation)
Challenges of EAs in DSE

- encoding allocation+binding
  - **simple encoding**
    e.g., one bit per resource, one variable per binding
    - easy to implement
    - ... however, it may lead to (many) infeasible partitioning solutions
  - **encoding + repair**
    e.g. simple encoding and repair for allocation
    s.t. for each $v_p \in V_p$ there exists at least one $v_a \in \alpha$ with $(v_p, v_a) \in E_m$
    - reduces number of infeasible partitioning solutions

- (“smart”) generation of initial population

- (“smart”) neighborhood operations, e.g., mutation, crossover
Example 2: Network Processors - Definition

- Typically, network processors serve as bridge between the network and the source/sink audio/video device (or set of devices)

- **implementation**: high-performance, programmable devices optimized for (real-time) network packet processing

- **features**: complex packet processing capabilities at high line speeds (routing; forwarding; de-/encryption; de-/compression; …) and means to guarantee quality-of-service
Network Processor Architecture (*)

Network processor heterogeneous hardware/software architecture:

- available processing units
  - ... are described in resource set $R = \{\text{ARM9, PowerPC, DSP, MEngine, Classifier, Cipher, LookUp, CheckSum}\}$
  - ... have a relative implementation cost $\text{cost}(r) \geq 0, \; r \in R$
  - ... and are selected for a specific architecture during the allocation step
    - with $\text{alloc}(r) = 1$ if a resource is selected and 0 otherwise

application structure: set of streams \( s \in S \) and set of tasks \( t \in T \)

- each stream includes an ordered sequence of tasks \( v(s) = [t_0, \ldots, t_n] \)

example:
\[ S = \{ \text{RTSend}, \text{NRTDecrypt}, \text{NRTEncrypt}, \text{RTRecv}, \text{NRTForward} \} \]
Problem: Optimal Design of Network Processor

- mappings $M \subseteq T \times R$: all possible bindings of tasks
  - i.e., if $(t, r) \in M$, then task $t$ could be executed on resource $r$
- request $w(r, t) \geq 0$
  - i.e., execution of one packet in task $t$ would use $w$ computing units of resource $r$
- resource allocation cost $c(r) \geq 0$

binding $Z$ of tasks to resources $Z \subseteq M$ (leading to actual implementation)
  - subset of mappings $M$ s.t. every task $t \in T$ is bound to exactly one allocated resource $r \in R$ and $\text{alloc}(r) = 1$ and $r = \text{bind}(t)$
NP Design Constraints

The design of network processors typically faces conflicting goals:

- **delay constraints**
  - e.g., maximal time a packet is processed within NP

- **throughput maximization**
  - e.g., maximum throughput of NP (packets per second)

- **cost minimization**
  - implementation with small amount of resources (e.g., processing units, memory, and communication networks)

- and conflicting usage scenarios
  - usually, a packet processor is used in several different systems (e.g., router or consumer multimedia processing device) and might have different implementations with different throughput/delay requirements
NP Design Space Exploration

issues to be considered during system-level design (and synthesis):

- allocation
  - determine hardware components of the network processor

- binding
  - for each process of the software application choose an allocated hardware unit which executes it

- scheduling
  - for the set of tasks mapped onto a specific resource choose scheduling policy/parameters – from available run-time environment, e.g., a fixed priority for each stream $s$: $\text{prio}(s) > 0$
Design Space Exploration Flow

- Hardware Architecture Template
- Software Application
- Run-Time Environment
- Application Scenarios

**Allocation**

\[ \text{alloc}(r) = 0/1 \]

**Binding**

\[ r = \text{bind}(t) \]

**Scheduling**

\[ \text{prio}(s) > 0 \]

**Multiobjective Evolutionary Selection**

Constraints Opt. Criteria

**Performance Analysis**

**HW/SW Architecture Cost and Performance**

**VARIATION**

**SELECTION**
Tools and a Small Demo
... Some Results

- **performance of encryption/decryption**
- **performance of RT voice processing**

### Results

<table>
<thead>
<tr>
<th>Component</th>
<th>NRT</th>
<th>RT</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>LookUp</strong></td>
<td>15%</td>
<td>6%</td>
</tr>
<tr>
<td><strong>Classifier</strong></td>
<td>27%</td>
<td>11%</td>
</tr>
<tr>
<td><strong>Cipher</strong></td>
<td>71%</td>
<td>0%</td>
</tr>
<tr>
<td><strong>DSP</strong></td>
<td>64%</td>
<td>39%</td>
</tr>
<tr>
<td><strong>DSP</strong></td>
<td>35%</td>
<td>39%</td>
</tr>
</tbody>
</table>
Example 3: Wave Field Synthesis

What is wave field synthesis (WFS)?

- high quality spatial sound reproduction system for huge listening areas
- 32 sound sources and 300 loudspeakers for medium sized reproduction rooms
System Specification: WFS Application

Parallel application modeled as *Kahn Process Network*

**structure: XML**

**functionality: ANSI C & DOL(*) API**

![Diagram of Kahn Process Network]

**Algorithm 1 Process Model**

1: `procedure INIT(DOLProcess p)`  \(\triangleright\) initialization
2: initialize local data structures
3: `end procedure`
4: `procedure FIRE(DOLProcess p)`  \(\triangleright\) execution
5: `DOL_read(INPUT, size, buf)`  \(\triangleright\) blocking read
6: manipulate
7: `DOL_write(OUTPUT, size, buf)`  \(\triangleright\) blocking write
8: `end procedure`

System Specification: Architecture

- Architecture is modeled at abstract level in XML format
- Modeled elements:
  - processors, buses, memories
  - communication paths between these elements
  - … parameters are included in the model
Application-to-Architecture Mapping

parallel application

heterogeneous architecture

design space exploration
(performance analysis & mapping optimization)

software synthesis
Simple Analysis Model

- **max processor load**
  \[
  obj_1 = \max_{c \in C} \left\{ \sum_{\forall p \text{ mapped to } c} n(p) \cdot r(p, c) \right\}
  \]

- **communication link with worst load**
  \[
  obj_2 = \max_{g \in G} \left\{ \sum_{\forall s \text{ mapped onto } g} \frac{b(s)}{t(g)} \right\}
  \]

- **number of activations of process p**
- **runtime of process p on processor c**
- **max bus load**
Example:  SDF specification

<table>
<thead>
<tr>
<th>Process</th>
<th>P₁</th>
<th>P₂</th>
<th>P₃</th>
<th>Channel</th>
<th>P₁→P₂</th>
<th>P₂→P₃</th>
<th>P₃→P₁</th>
<th>P₁→P₃</th>
</tr>
</thead>
<tbody>
<tr>
<td>Runtime</td>
<td>1ms</td>
<td>3ms</td>
<td>2ms</td>
<td>Token</td>
<td>100 Byte</td>
<td>2048 Byte</td>
<td>2048 Byte</td>
<td>30k Byte</td>
</tr>
</tbody>
</table>

Happens:  P₁→CPU₁ ;  P₂→CPU₁ ;  P₃→CPU₂
\[ \text{obj}_1 = \max \{ \text{rate} \cdot (2.1 \text{ms} + 3 \text{ms}, 2.2 \text{ms}) \} = \text{rate} \cdot 5 \text{ms} \]

\[ \text{obj}_2 = \max \{ \text{rate} \cdot \frac{2 \cdot 204 \text{Byte} + 2 \cdot 304 \text{Byte} + 2004 \text{Byte}}{100 \text{ MByte/s}} \} = \text{rate} \cdot 3 \text{ms} \]

\[ \frac{\text{obj}_2}{\text{rate}} \]

Graph showing \( 3 \text{ms} \) and \( 5 \text{ms} \) on the x-axis and \( \text{obj}/\text{rate} \) on the y-axis.
Where Are Data Obtained From?

- **Static parameters**: bandwidth of buses $t(g)$
- **Functional simulation**: number of activations for each process $n(p)$, amount of data for each channel $b(s)$
- **Instruction-set simulation**: runtime of each process on different processors $r(p,c)$ by using benchmark mappings
Design Space Exploration Cycle – An Example

- **instruction-level simulation**
- **functional simulation**
- **designer’s data sheet**
- **annotated application XML**
- **annotated architecture XML**

**Evaluation**
- **analysis model**
- **performance numbers**
- **system description**

**EXPO**
- **EXPO application**
- **EXPO architecture**

**Mapping generation & variation** (mutation/crossover)
- **EXPO mapping**
- **mapping XML**

**PISA interface**

**Multi-objective optimization**
- **evolutionary algorithm**

---

**EXPO**

**Multi-objective optimization**

**Evolutionary algorithm**
Example 3: Exploration

- microphones
- convolution
- sum
- loudspeakers

max. bus load

max. processor load

single processor mapping

search direction

current population

ARM

current population

mAgic

AHB 0