Reliable Run-time Adaptation in Resource-constrained Embedded Systems

Twan Basten

Joint work with Marc Geilen, AmirHossein Ghamarian, Rob Hoes, Hamid Shojaei, Sander Stuijk, and others

Funding:
EC FP6 Betsy & WASP
EC FP7 MNEMEE

’Knowing is not understanding.’
Charles Kettering

Run-time Adaptation

Encoding qualities
Bandwidth requirements

Decoding streams of different quality
Computational effort required

Available bandwidth under given circumstances
Different levels of computation power
Different levels of power consumption

What is the optimal configuration?
- given communication bandwidth
- given battery status
- given user preferences

Requires run-time decision making
- multi-dimensional problem
- strict timing constraints
- limited resources

Pareto Algebra

Fundamenta Informaticae, 2007
5. **Approach: Pareto Algebra**

Goal: Compositional computation of trade-offs

- producer-consumer operation

6. **A Pareto Space**

- Pareto points: optimal trade-offs
- configurations
- dominance
- energy
- cost function
- optimal point
- better
- time
- ≤

7. **Pareto Algebra**

- The elements: sets of configurations
- (Relevant) operators:
  - **Minimization**
    - Gives the Pareto points (optimal trade-offs)
  - **Product**
    - Cartesian product of configurations
      e.g. application and platform configurations
  - **Constraint**
    - Selects solutions according to constraints
      e.g. all application configurations with some minimal quality
  - **Abstraction**
    - Discards information about solutions
      e.g. bandwidth usage in bandwidth × energy × quality configurations
  - **Derived metric**
    - Derive a new metric from other metrics
      e.g. total power from power of components

8. **Compound Operations**

- **producer-consumer operation**
  - <=

- **join operation (relational databases)**
  - >=
Run-time Adaptation

Pareto-Algebraic Characterization

... of run-time adaptation

- product - derive | constrain | abstract - minimize
- ... select and configure

- component trade-off spaces
- ... system trade-off space (at the time of a change)

- instances:
  - producer-consumer
  - join

Complexity

- n components with p Pareto points each

- O(p^2n)
- system trade-off space (at the time of a change)
Complexity

- \( n \) components with \( p \) Pareto points each
- take a product and on-the-fly derive | constrain | abstract - minimize
- \( O(p^2n) \)

Complexity Control

keep \( n \) and \( p \) small

- Compositional reasoning, among others
- Approximate Pareto sets

Run-time Application Management

- Applications starting/stopping over time
- 6 pros, slow (ck1) and fast clocks (ck2)
- Minimize energy under time constraints

Chip Multi-Processors
Multi-dimensional Multiple-choice Knapsack Problem

MMKP
- One optimization objective (value)
- Multiple resource dimensions with capacity constraints
- Multiple independent applications
- Multiple independent configurations per application
Pick one configuration per application optimizing value within constraints

NP hard

A Parameterized Compositional Heuristic

- Project all resource dimensions into one dimension
- Approximate Pareto set in 2-dimensional space

Compute product while on-the-fly
- Applying resource constraints
- Computing values, projecting all resource dimensions into one (taking simply the sum)
- Minimizing the configuration set in 2-dimensional space

project-constrain-derive-abstract-minimize
### 21 A Parameterized Compositional Heuristic

- Project all resource dimensions into one dimension
- Approximate Pareto set in 2-dimensional space

Parameter $p$: maintain (at most) $p$ Pareto points

Allows to budget analysis time

$$\text{analysis time} \leq c \cdot n \cdot p^2$$

(n number of applications; $c$ platform constant)

### 22 MMKP Benchmark

(www.laria.u-picardie.fr/hifi/OR-Benchmark/MMKP)

<table>
<thead>
<tr>
<th>Test case</th>
<th># of applications</th>
<th># of configurations</th>
<th># of resources</th>
</tr>
</thead>
<tbody>
<tr>
<td>I1</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>I2</td>
<td>10</td>
<td>10</td>
<td>5</td>
</tr>
<tr>
<td>I3</td>
<td>15</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>I4</td>
<td>20</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>I5</td>
<td>25</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>I6</td>
<td>30</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>I7</td>
<td>100</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>I8</td>
<td>150</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>I9</td>
<td>200</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>I10</td>
<td>250</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>I11</td>
<td>300</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>I12</td>
<td>350</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>I13</td>
<td>400</td>
<td>10</td>
<td>10</td>
</tr>
</tbody>
</table>

Tested on the SimIt-ARM simulator 206 Mhz StrongARM

### 23 Controlling Time Budgets

- Always within budget
- Good values
- Similar results to the IMEC, SOC 2006, non-compositional state-of-the-art heuristic

### 24 Compositionality

- Applications start at various times
- 5 at a time (a), 1 at a time (b)

Values always at least as good as IMEC heuristic
Wireless Sensor Networks

Configuring a Static WSN

Problem
How to configure the network?

Approach
Compositional computation following the routing-tree structure

Motivation
Pareto algebra

Adaptation
wsn

Conclusions

2700 configurations in 4D space

900 nodes
27 configurations per node
4 quality metrics
Routing tree to data sink

Less than 45 seconds …

… but then the sink moves …
Run-time Adaptation

- Repair the routing tree
- Compute QoS trade-offs
- Select and load new configuration

**Parameters**
- \( d \): deviation from minimally affected area
- \( n \): max number of children per node
- \( p \): max number of Pareto points

**Trade-off**
- Global reconfiguration: cost vs. quality
- Local reconfiguration

Repairing the Tree: QuickFix

Reverse the path from the former children of the sink to the new children

Quality Improvement: Controlled Flooding

Reconfigure up to some given deviation depth \( d \) from the affected area
**Computation at a Given Node**

Nodes that need to recompute Pareto sets:
- All nodes with a new parent or new children
- All their ascendants

**Conclusions**

- Run-time adaptation:
  
  product - derive | constrain | abstract - minimize
  
  ... select and configure

- Parameterized compositional method
  
  - Allows to trade off quality with analysis time
  - Allows to bound analysis time
  - Open question: components leaving the composition?

- Feasible for resource-constrained embedded systems

- Wide variety of applications

**Reconfiguration Time vs. Quality**

900 node network
Simulated in the OMNET++ simulator
Calibrated for TelosB sensor nodes
8 Mhz processor
250 kbps tranceiver bit rate

90 node network

- non-optimised
- local
- global
- semi-local
- par

Quality loss (%)

0 1 2 3 4

Reconfiguration time (s)

0 100 200 300 400

Good quality reconfiguration within one minute

**Thank you!**

Questions?

More info: [www.es.ele.tue.nl/~tbasten/](http://www.es.ele.tue.nl/~tbasten/)
[www.es.ele.tue.nl/pareto/](http://www.es.ele.tue.nl/pareto/)

‘An understanding of the natural world and what’s in it is a source of not only great curiosity but great fulfillment.’

David Attenborough