HW/SW Codesign

Exercise 8: Design Space Exploration
(Pareto optimality and evolutionary algorithm)

23 November 2016

Rehan Ahmed
rehan.ahmed@tik.ee.ethz.ch
Plan

• Recap
• Question explanation
• Your time to solve (~75 mins)
• Solution explanation
• Projects @ TEC
Multi-criteria optimization / Pareto-points

A dominates B

cheaper

cost

faster

latency
Multi-criteria optimization / Pareto-points

A and C not comparable

A

C

cheaper

cost

faster

latency
Multi-criteria optimization / Pareto-points

Pareto-front

Points dominated by C

Pareto-Points. = Points, which are not dominated by others
Evolutionary Algorithm Cycle

- representation operator
- mating selection
- crossover
- initial/parent set
- children set
- environmental selection
- mutation operator
Representation

solutions encoded by vectors, matrices, trees, lists, ...

Issues:

- completeness (each solution has an encoding)
- uniformity (all solutions are represented equally often)
- feasibility (each encoding maps to a feasible solution)
- redundancy (a solution has multiple encodings)
Vector Mutation: Examples

**Bit vectors:**

1 0 1 1 1 0

Each bit is flipped with probability 1/6

1 0 0 1 1 0

**Permutations:**

1 2 3 4 5 6 → 1 4 3 2 5 6

Swap

1 2 3 4 5 6 → 1 3 4 2 5 6

Rearrange
Vector Crossover: Examples

**Bit vectors:**

\[
\begin{bmatrix}
1 & 1 & 0 & 0 & 1 & 0 \\
1 & 0 & 1 & 0 & 0 & 1
\end{bmatrix}
\rightarrow
\begin{bmatrix}
1 & 1 & 0 & 0 & 0 & 1
\end{bmatrix}
\]

**Permutations:**

\[
\begin{bmatrix}
1 & 2 & 3 & 4 & 5 & 6 \\
6 & 2 & 3 & 4 & 1 & 5
\end{bmatrix}
\rightarrow
\begin{bmatrix}
1 & 2 & 3 & 6 & 4 & 5
\end{bmatrix}
\text{child}
\]
Constraint Handling

**Constraint:** \( g(x) \geq 0 \)

solution in decision space

**Approaches:**

- representation is chosen such that decoding always yields a feasible solution
- construct initialization and neighborhood operators such that infeasible solutions are not generated
- add to children population only feasible solutions
- in environmental selection, preferably select feasible solutions
- calculate constraint violation \( g(x) \) and incorporate it into objective function using a penalty function: \( \text{penalty}(x) > 0 \) if \( g(x) < 0 \), \( \text{penalty}(x) = 0 \) if \( g(x) \geq 0 \). For example, add penalty function to every objective.
- include the constraints as new objectives
Exercise 8.1

- Four tasks T1 .. T4 can be executed on components as shown in Table 1.
- Each component executes tasks serially.
- Tasks are non-preemptive.

<table>
<thead>
<tr>
<th>component</th>
<th>number</th>
<th>cost</th>
<th>execution time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>T1</td>
</tr>
<tr>
<td>MIPS</td>
<td>1</td>
<td>200,-</td>
<td>5 ms</td>
</tr>
<tr>
<td>DSP</td>
<td>1</td>
<td>120,-</td>
<td>—</td>
</tr>
<tr>
<td>FPGA</td>
<td>1</td>
<td>240,-</td>
<td>—</td>
</tr>
<tr>
<td>ASIC</td>
<td>1</td>
<td>400,-</td>
<td>—</td>
</tr>
</tbody>
</table>

Table 1: Available components with cost and execution times for tasks T1 ... T4.
Exercise 8.1 (a)

Construct the design-space by listing all possible design points. Design point consists of allocation, binding, and scheduling.

- Find the total cost and execution time for each design point.

For example, one of the design points is:
(T1 $\rightarrow$ MIPS, T2 $\rightarrow$ DSP, T3 $\rightarrow$ DSP, T4 $\rightarrow$ DSP).

The total cost = MIPS + DSP = 200 + 120 = 320

Execution time: 5ms + 20ms + 18ms + 5ms = 48ms
Exercise 8.1 (b)

Draw the design points in a cost-time diagram. Which design points are Pareto points?
Exercise 8.1 (c)

Consider an allocation without resource constraints. That means we are given an arbitrarily high number of components of each type (MIPS, DSP, FPGA and ASIC).

- Are there new design points?
- Does the set of Pareto points change?
Exercise 8.1 (d)

How many design points exist at least if a task graph comprises \( n \) tasks, each task can be executed on at least three component types, and there are no resource constraints?
An individual is defined by:

- **Allocation** (which resources are used?)
- **Binding** (which task runs on which resource?)
- **Scheduling** (how are tasks scheduled on a single resource?)
  - Here, scheduling is also part of the chromosome
DSE With EA: Exercise 8.2 (2)

- 6 tasks with sequence graph:

- 3 resources with possible bindings between tasks and resources:
DSE With EA: Exercise 8.2 (2)

• Tasks:
  – Find suitable **encoding** for (i) allocation, (ii) binding, and (iii) scheduling
  – Prove **completeness**
  – Evaluate **uniformity, redundancy, and feasibility**
  – Specify **mutation** and **recombination** operators
    • How to handle possible unfeasible solutions?
Plan

• Recap
• Question explanation
• Your time to solve (~75 mins)
• Solution explanation
• Projects @ TIK
Plan

• Recap
• Question explanation
• Your time to solve
• Solution explanation
• Projects @ TEC
**Solution 8.1 (a)**

**Design points:**

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_1$</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
</tr>
<tr>
<td>$T_2$</td>
<td>DSP</td>
<td>DSP</td>
<td>DSP</td>
<td>DSP</td>
<td>DSP</td>
<td>DSP</td>
<td>DSP</td>
<td>DSP</td>
<td>DSP</td>
<td>DSP</td>
<td>DSP</td>
<td>DSP</td>
</tr>
<tr>
<td>$T_3$</td>
<td>DSP</td>
<td>DSP</td>
<td>FPGA</td>
<td>FPGA</td>
<td>FPGA</td>
<td>FPGA</td>
<td>FPGA</td>
<td>FPGA</td>
<td>FPGA</td>
<td>FPGA</td>
<td>ASIC</td>
<td>ASIC</td>
</tr>
<tr>
<td>$T_4$</td>
<td>DSP</td>
<td>DSP</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
</tr>
<tr>
<td>Exec.</td>
<td>48 ms</td>
<td>45 ms</td>
<td>30 ms</td>
<td>27 ms</td>
<td>27 ms</td>
<td>28 ms</td>
<td>25 ms</td>
<td>32 ms</td>
<td>29 ms</td>
<td>22 ms</td>
<td>19 ms</td>
<td></td>
</tr>
</tbody>
</table>

Considering 4 additional design points where $T_3$ executes before $T_2$ is also a valid solution.

Table 1: Available components with cost and execution times for tasks $T_1 \ldots T_4$.  

<table>
<thead>
<tr>
<th>component</th>
<th>number</th>
<th>cost</th>
<th>$T_1$</th>
<th>$T_2$</th>
<th>$T_3$</th>
<th>$T_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>MIPS</td>
<td>1</td>
<td>200.-</td>
<td>5 ms</td>
<td>—</td>
<td>—</td>
<td>2 ms</td>
</tr>
<tr>
<td>DSP</td>
<td>1</td>
<td>120.-</td>
<td>—</td>
<td>20 ms</td>
<td>18 ms</td>
<td>5 ms</td>
</tr>
<tr>
<td>FPGA</td>
<td>1</td>
<td>240.-</td>
<td>—</td>
<td>12 ms</td>
<td>10 ms</td>
<td>—</td>
</tr>
<tr>
<td>ASIC</td>
<td>1</td>
<td>400.-</td>
<td>—</td>
<td>—</td>
<td>800 µs</td>
<td>—</td>
</tr>
</tbody>
</table>
Solution 8.1 (b)

Pareto-points: 2, 8, 10, 12
Solution 8.1 (c)

New design points:

<table>
<thead>
<tr>
<th></th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_1$</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
</tr>
<tr>
<td>$T_2$</td>
<td>DSP1</td>
<td>DSP1</td>
<td>FPGA1</td>
<td>FPGA1</td>
</tr>
<tr>
<td>$T_3$</td>
<td>DSP2</td>
<td>DSP2</td>
<td>FPGA2</td>
<td>FPGA2</td>
</tr>
<tr>
<td>$T_4$</td>
<td>MIPS</td>
<td>DSP1/2</td>
<td>MIPS</td>
<td>DSP</td>
</tr>
</tbody>
</table>

Cost: 440.- 440.- 680.- 800.-
Execution time: 27 ms 30 ms 19 ms 22 ms

Original design points:

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_1$</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
</tr>
<tr>
<td>$T_2$</td>
<td>DSP</td>
<td>DSP</td>
<td>DSP</td>
<td>DSP</td>
<td>DSP</td>
<td>DSP</td>
<td>DSP</td>
<td>DSP</td>
<td>DSP</td>
<td>FPGA</td>
<td>FPGA</td>
<td>FPGA</td>
</tr>
<tr>
<td>$T_3$</td>
<td>DSP</td>
<td>DSP</td>
<td>FPGA</td>
<td>ASIC</td>
<td>ASIC</td>
<td>ASIC</td>
<td>ASIC</td>
<td>ASIC</td>
<td>ASIC</td>
<td>FPGA</td>
<td>FPGA</td>
<td>FPGA</td>
</tr>
<tr>
<td>$T_4$</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
<td>MIPS</td>
</tr>
</tbody>
</table>

Execution time: 48 ms 45 ms 30 ms 27 ms 30 ms 27 ms 28 ms 25 ms 29 ms 22 ms 19 ms

Pareto-points: 2, 8, 13, 15
Solution 8.1 (d)

Minimum number of design points: $3^n$
DSE With EA: Exercise 8.2 (2)

- 6 tasks with sequence graph:

- 3 resources with possible bindings between tasks and resources:
Exercise 8.2: Possible Solution 1

- Allocation:

<table>
<thead>
<tr>
<th>Res.</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alloc.</td>
<td>1</td>
<td>1</td>
<td>0/1</td>
</tr>
</tbody>
</table>

- Binding:

<table>
<thead>
<tr>
<th>Arrow</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bound</td>
<td>0/1</td>
<td>0/1</td>
<td>0/1</td>
<td>0/1</td>
<td>0/1</td>
<td>0/1</td>
<td>0/1</td>
<td>0/1</td>
<td>0/1</td>
<td>0/1</td>
</tr>
</tbody>
</table>

- Scheduling:

<table>
<thead>
<tr>
<th>Task</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
</table>

- Non-uniform, redundant, complete, leads to infeasible solutions

- Possible strategy to have feasible scheduling solutions with mutation/recombination: use a constraint matrix

<table>
<thead>
<tr>
<th>Task</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>5</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>6</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>
Exercise 8.2: Possible Solution 2

- **Allocation and Binding:**

<table>
<thead>
<tr>
<th>Task</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Res.</td>
<td>1</td>
<td>2</td>
<td>1/2</td>
<td>1/2</td>
<td>2/3</td>
<td>2/3</td>
</tr>
</tbody>
</table>

- **Scheduling:**

<table>
<thead>
<tr>
<th>Order</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task</td>
<td>1/2/3</td>
<td>1/2/3/4</td>
<td>1/2/3/4/5</td>
<td>2/3/4/5</td>
<td>3/4/5</td>
<td>6</td>
</tr>
</tbody>
</table>

- **Non-uniform, redundant, complete, leads to infeasible solutions**
Why Redundant??

Schedule decoding

1. Each resource (R1, R2, R3) constructs a set of tasks S which are eligible for execution.

2. If S is empty: the resource waits for a task to become eligible

3. If S has one task: that task is executed

4. If S has more than one tasks: the task with lowest order is executed

5. After completion of a task, steps 1-4 are repeated
Why Redundant??

Binding:

<table>
<thead>
<tr>
<th>Task</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Res.</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>2</td>
</tr>
</tbody>
</table>

Schedule 1

<table>
<thead>
<tr>
<th>Order</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>6</td>
</tr>
</tbody>
</table>

Schedule 1

<table>
<thead>
<tr>
<th>Order</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task</td>
<td>1</td>
<td>4</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>6</td>
</tr>
</tbody>
</table>

Only relative order of T2 and T3 important for this binding

For redundancy: list all possible schedule encodings and count the ones where T2 Order < T3 Order
Why Non-uniform

Different bindings will have different number of redundant schedule encodings.

Feel free to prove/disprove
Exercise 8.2: Possible Solution 2

- Mutation & Recombination

<table>
<thead>
<tr>
<th>Scheduling 1</th>
<th>Task</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Order</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td></td>
</tr>
</tbody>
</table>

Mutation/Swapping

<table>
<thead>
<tr>
<th>Mutated 1</th>
<th>Task</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Order</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td></td>
</tr>
</tbody>
</table>

Recombination/Crossover

<table>
<thead>
<tr>
<th>Scheduling 2</th>
<th>Task</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Order</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>6</td>
<td></td>
</tr>
</tbody>
</table>

Final

<table>
<thead>
<tr>
<th>Final</th>
<th>Task</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Order</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>6</td>
<td></td>
</tr>
</tbody>
</table>
Exercise 8.2: Possible Solution 2

- What if it goes wrong?
  
  ...Discard

  ...Change your randomization rule

  ...or keep based on fitness evaluation

  $^{2}$nd in general difficult.
Some interesting projects

Thermal fingerprinting

- Identify videos being played on a platform by reading temperature

Develop/propose classification/identification algorithms

Contact:
Philipp Miedl (philipp.miedl@tik.ee.ethz.ch)
Dr. Rehan Ahmed (rehan.ahmed@tik.ee.ethz.ch)
Some interesting projects

Prediction strategies for energy harvesting sources

Develop source specific prediction schemes

Contact:
Dr. Rehan Ahmed (rehan.ahmed@tik.ee.ethz.ch)
Stefan Draskovic (stefan.draskovic@tik.ee.ethz.ch)
Some interesting projects

UnCovert 4 - The Power Covert Channel

- Establish Covert Channel using CPU power consumption
- Model the covert channel & determine channel capacity
- Design a case study to evaluate the covert channel
- Determine maximum achievable transmission rates
- Study the covert channel robustness in different environments

- Skills: C/C++/Java, Data Analysis, Script Languages

⇒ Philipp Miedl: philipp.miedl@tik.ee.ethz.ch, ETZ G76
Some interesting projects

A Smart Attack using the Frequency Covert Channel

- Establish Covert Channel using operation frequency
- Develop smart src and snk applications
- Automatically establish covert communication
- Evaluate robustness and performance

Skills: C/C++/Java, Data Analysis, Script Languages

⇒ Philipp Miedl: philipp.miedl@tik.ee.ethz.ch, ETZ G76
⇒ Matthias Meyer: matthias.meyer@tik.ee.ethz.ch, ETZ G81