8.1 Design Space, Pareto Points

The task graph in Figure 1 shows a system specification with four tasks, T1 . . . T4. The tasks can be executed on different components. Table 1 displays the execution times for the tasks on the different components as well as the component cost. For example, the MIPS processor costs 200 units and can run tasks T1 in 5 ms and task T4 in 2 ms. Table 1 shows also the number of components available for each component type (MIPS, DSP, FPGA and ASIC). All components execute tasks sequentially – at any given time a component executes at most one task. Task execution is non-preemptive – once a task is started, it runs to completion.

![Task graph](image)

Table 1: Available components with cost and execution times for tasks T1 . . . T4.

<table>
<thead>
<tr>
<th>component</th>
<th>number</th>
<th>cost</th>
<th>execution time</th>
</tr>
</thead>
<tbody>
<tr>
<td>MIPS</td>
<td>1</td>
<td>200.│ 5 ms — — 2 ms</td>
<td></td>
</tr>
<tr>
<td>DSP</td>
<td>1</td>
<td>120.│ — 20 ms 18 ms 5 ms</td>
<td></td>
</tr>
<tr>
<td>FPGA</td>
<td>1</td>
<td>240.│ — 12 ms 10 ms —</td>
<td></td>
</tr>
<tr>
<td>ASIC</td>
<td>1</td>
<td>400.│ — — 800 μs —</td>
<td></td>
</tr>
</tbody>
</table>

(a) Construct the design space by listing all possible design points. A design point consists of an allocation (selection of components), a binding (assignment of tasks to selected components) and a schedule (execution order for the tasks). Determine the total cost and execution time for each design point.

**Solution:** See Figure 2 and Table 2. If T2 and T3 are binded to the same resource, then the execution of T2 takes place prior to T3. If T2 and T3 are binded to different resources, then T2 and T3 executes in parallel.
Table 2: 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>FPGA</td>
<td>FPGA</td>
<td>ASIC</td>
<td>ASIC</td>
<td>ASIC</td>
<td>ASIC</td>
<td>ASIC</td>
<td>ASIC</td>
<td>ASIC</td>
<td>ASIC</td>
<td>ASIC</td>
</tr>
<tr>
<td>$T_4$</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>
<td>MIPS</td>
</tr>
</tbody>
</table>

Execution time: 48 ms 45 ms 30 ms 27 ms 27 ms 28 ms 25 ms 32 ms 29 ms 22 ms 19 ms

Figure 2: Design points in cost-time graph.

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

**Solution:** Points 2, 8, 10, 12 are Pareto-optimal.

(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?

**Solution:** See Figure 3 and Table 3. Points 2, 8, 13, 15 are now Pareto-optimal.

(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?

**Solution:** $3^n$


8.2 Design Space Exploration with Evolutionary Algorithms

In this exercise, we consider a design space exploration problem with an evolutionary algorithm as depicted in Fig. 4.

Figure 3: Additional design points in cost-time graph

An individual is defined by:
(a) **Allocation** (which resources are used?)
(b) **Binding** (which task runs on which resource?)
(c) **Scheduling** (how are the tasks scheduled on a single resource?)

*Note: In this exercise the scheduling is also part of the chromosome. Usually, only the allocation and the binding is encoded, whereas the scheduling is computed and optimized in an extra step after the decoding.*

A simple problem specification is given in Fig. 5 and Fig. 6. In this example, we do not consider the communication between resources.

**Tasks:**
8.2.a) Encoding of Solutions

- Find a suitable encoding for the given problem specification for: (i) the allocation, (ii) the binding, and (iii) the scheduling.
- Proof the completeness of your encoding.
- Evaluate your encoding regarding the uniformity, the redundancy, and the feasibility.

8.2.b) Mutation and Recombination

- Specify a mutation and a recombination operator for the given problem.
- Do the operations lead to infeasible solutions? If yes, what do you suggest for the handling of such solutions?

Solution:

*Many solutions are possible with different trade-offs.*