Evolutionary Algorithm for Embedded System Topology Optimization

Supervisor:
Prof. Dr. Martin Radetzki

Author:
Haowei Wang
Agenda

- Introduction to the problem
- Principle of evolutionary algorithm
- Model specification
- Algorithm to fit the problem into EA
The Problem

- Find an optimized solution for topology optimization in networked Embedded System.

- Modern ES:
  - heterogeneous techniques
  - distributed over the system.

- Common architectures:
  - Computational nodes;
  - Communication networks;
Design Space Exploration

- First on SoC, now also on networked embedded system;

- Usually work on system design level;

- Evolutionary Algorithm is suitable for finding optimal solution for
  - Huge amount of searching effort (NP-complete)
  - Multiple objective which are often conflict;
Basic Task

- Basic Task in designing networked ES is to decide:
  - Devices;
  - Implementation;

- Many needs to be considered when designing:
  - Monetary Cost;
  - Performance;
  - Network traffic;
  - Etc…

---

Resource (devices, buses…)

Constrains (PROC1 can only on RISC)

Needs (less money, more performance)

Design Space Exploration (Modeling, evaluation, optimization)

Optimized Solution
Evolutionary algorithm

- Heuristic optimization algorithm;
- Random search principle;
- Iteration improvement;
- Based on a model which specify the system;
Chromosome

- A set of genes;
- Predefined meaning;
- An instance of a model;
- Searching space: holds every possible solution for a model;

\[ Y = f(x_1, x_2, x_3, \ldots, x_n) \]
Population, evolution & evaluation

- **Population:**
  - Set of chromosomes;
  - Iteratively improve itself in evolution;

- **Evolution:**
  - Defines the way how population improves;
  - Usually combines of several tasks: Selection, Crossover, Mutation;

- **Evaluation:**
  - Calculate the fitness;
  - Used as the criteria of selection;
Multiobjective EA - SPEA

- **Concept:**
  - Pareto-optimal:
    - Gene can not be better without another gene to be worse;
  - Pareto-dominance:
    - A dominant B: every gene in A not worse than B;
  - Nondominated Set;

- **Fitness Calculation:**
  - Every nondominated individuals has a strength;
  - Fitness is the sum of the strength of the nondominated individuals which dominate this individual;

[from Eckart99]
Model

Constrains 1:
- task 1 can only be implemented in RISC

Specification Graph
Specification Graph 1 – Program Graph

- Program Graph:
  - Applications to be realized;
  - Demand defined on edges;

- Message Type:
  - Different communication protocols;
Specification Graph 2 – Architecture Graph

- Architecture Graph:
  - Architecture template;
  - Capacity defined on edges;
  - For each connection, one message type, one protocol;
Specification Graph 3 – Mapping Edge

- Mapping Edge:
  - possible implementation of a process on the resources;

- Specification Graph:
  - PG + AG + ME + MT;
  - Missing mapping edges mean constrains on mapping;
Implementation 1 –
An extreme example

- Implementation is a possible solution;
- An extreme example:
  - Extreme low cost;
  - Also low performance;
Implementation 2 – Basic Concept

- **Activation:**
  - a function that assigns each nodes or edges value 0 or 1.

- **Allocation:**
  - subset of activation with only nodes and edges in PG & AG;

- **Process binding:**
  - subset of activation with only mapping edges;

- **Demand binding:**
  - bind the communication edges;
Implementation 3 – Feasible allocation

- Feasible process binding:
  - Guarantees the functions and communication connections;

- Feasible demand binding:
  - Fulfill the capacity restriction;

- Feasible allocation:
  - Allow at least one feasible process binding and corresponding feasible demand binding;
Implementation 4 – Scheduling

- Scheduling:
  - Calculate the system’s performance;

- Implementation:
  - Feasible allocation/process binding/ demand binding + scheduling (optional);
Implementation 4 – Scheduling

- Scheduling:
  - Calculate the system’s performance;

- Implementation:
  - Feasible allocation + scheduling (optional);
Structure of topology optimization algorithm
Chromosome code

Chromosome

Allocation

Allocation list

| 1 | 0 | 0 | 0 | 0 | 1 | ... | 0 | 1 |

Repair Allocation Priority List

node2 node1 ... node4

Process Binding

Process Biding Priority

| P1 | P4 | P2 | P5 | ... | P3 |

Mapping order

| P1: node2 node6 node4 |
| P2: node4 node1 |
| ...... |

Demand binding

Problem Graph Edge Binding Priority

| edge1 | edge3 | edge7 | edge2 | edge4 | edge5 | ...... | edge6 |
Decoding process 1 – Allocation

- Allocation is simple;
- But, it’s necessary to do repairing! (Huge amount of infeasible solutions)
Decoding process 2 – Process Binding

- Decide the “real” mapping edges.
Decoding process 3 – Demand Binding

- Bind the communication edges in PG to the connection edges in AG;
- Check the limit of the capacities;

Problem Graph Edge Binding Priority

<table>
<thead>
<tr>
<th>edge1</th>
<th>edge3</th>
<th>edge7</th>
<th>edge2</th>
<th>edge4</th>
<th>edge5</th>
<th>......</th>
</tr>
</thead>
</table>

Diagram showing the binding process with nodes PROC1, PROC2, PROC3, RISC1, RISC2, CAN, GW, FlexRay, HW1, HW2.
Fitness evaluation

- Similar as the evaluation process of SPEA;

- Criteria like cost, performance, network traffic are common to be concerned;

- Constrains checking should be considered;
  - Use a constrain checker;
  - Invalid solution will be given the worst fitness;
Conclusion

- Embedded system design optimization depends on multiple objectives;
- Evolutionary algorithm can find heuristic optimized solution for multiobjective problems efficiently;
- Specification Graph is used to model the system;
- Implementation is a feasible solution;
- Decoding procedure transfers code to implementation;
- This general methodology is also suitable for many other optimization problems!
Thank you!