Generalized strictly periodic scheduling analysis, resource optimization, and implementation of adaptive streaming applications

Niknam, S.

Generalized Strictly Periodic Scheduling Analysis, Resource Optimization, and Implementation of Adaptive Streaming Applications

Sobhan Niknam

Generalized Strictly Periodic Scheduling Analysis, Resource Optimization, and Implementation of Adaptive Streaming Applications

PROEFSCHRIFT

ter verkrijging van
de graad van Doctor aan de Universiteit Leiden,
op gezag van Rector Magnificus Prof.mr. C.J.J.M. Stolker,
volgens besluit van het College voor Promoties
te verdedigen op dinsdag 25 augustus 2020
klokke 15:00 uur

door

Sobhan Niknam
geboren te Tehran, Iran
in 1990

Promotor:	Dr. Todor Stefanov	Universiteit Leiden
Second-Promotor:	Prof. dr. Harry Wijshoff	Universiteit Leiden
Promotion Committee:	Prof. dr. Akash Kumar	TU Dresden
	Prof. dr. Jeroen Voeten	TU Eindhoven
	Prof. dr. Paul Havinga	Universiteit Twente
	Prof. dr. Frank de Boer	Universiteit Leiden
	Prof. dr. Aske Plaat	Universiteit Leiden
	Prof. dr. Marcello Bonsangue	Universiteit Leiden

The research was supported by NWO under project number 12695 (CPS-3).

Generalized Strictly Periodic Scheduling Analysis, Resource Optimization, and Implementation of Adaptive Streaming Applications
Sobhan Niknam. -
Dissertation Universiteit Leiden. - With ref. - With summary in Dutch.

This dissertation was typeset using LaTeX.

ISBN: 978-90-9033402-8
Printed by Ipskamp Printing, Enschede.

To my family

Contents

Table of Contents **vii**

List of Figures **xi**

List of Tables **xv**

List of Abbreviations **xvii**

1 Introduction **1**

1.1 Design Requirements for Embedded Streaming Systems . . . 2
1.2 Trends in Embedded Streaming Systems Design 4
1.2.1 Multi-Processor System-on-Chip (MPSoC) 4
1.2.2 Model-based Design 6
1.3 Two Important Design Challenges 8
1.4 Research Questions 9
1.4.1 Phase 1: Analysis 10
1.4.2 Phase 2: Resource Optimization 11
1.4.3 Phase 3: Implementation 12
1.5 Research Contributions 13
1.5.1 Generalized Strictly Periodic Scheduling Framework . 13
1.5.2 Algorithm to Find an Alternative Application Task Graph for Efficient Utilization of Processors 13
1.5.3 Energy-Efficient Periodic Scheduling Approach 14
1.5.4 MADF Implementation and Execution Approach . . . 14
1.6 Thesis Outline 15

2 Background **17**

2.1 Dataflow Models of Computation 18
2.1.1 Cyclo-Static/Synchronous Data Flow (CSDF/SDF) . . 18
2.1.2 Mode-Aware Data Flow (MADF) 20

2.2 Real-Time Scheduling Theory 23
2.2.1 System Model 23
2.2.2 Real-Time Periodic Task Model 23
2.2.3 Real-Time Scheduling Algorithms 24
2.3 HRT Scheduling of Acyclic CSDF Graphs 28
2.4 HRT Scheduling of MADF Graphs 30

3 Hard Real-Time Scheduling of Cyclic CSDF Graphs 35
3.1 Problem Statement 35
3.2 Contributions 36
3.3 Related Work 37
3.4 Motivational Example 38
3.5 Our Proposed Framework 40
3.5.1 Existence of a Strictly Periodic Schedule 41
3.5.2 Deriving Period, Earliest Start Time, and Deadline of Tasks 45
3.6 Experimental Evaluation 46
3.7 Conclusions 49

4 Exploiting Parallelism in Applications to Efficiently Utilize Processors 51
4.1 Problem Statement 52
4.2 Contributions 53
4.3 Related Work 54
4.4 Background 57
4.4.1 Unfolding Transformation of SDF Graphs 57
4.4.2 System Model 58
4.5 Motivational Example 59
4.6 Proposed Algorithm 63
4.7 Experimental Evaluation 67
4.7.1 Homogeneous platform 70
4.7.2 Heterogeneous platform 73
4.8 Conclusions 76

5 Energy-Efficient Scheduling of Streaming Applications 77
5.1 Problem Statement 77
5.2 Contributions 78
5.3 Related Work 79
5.4 Background 80
5.4.1 System Model 81
5.4.2 Power Model 81

5.5 Motivational Example 81
5.5.1 Applying VFS Similar to Related Works 82
5.5.2 Our Proposed Scheduling Approach 84
5.6 Proposed Scheduling Approach 87
5.6.1 Determining Operating Modes 91
5.6.2 Switching Costs $o_{HL}, o_{LH}, e_{HL}, e_{LH}$ 92
5.6.3 Computing Q_H and Q_L 95
5.6.4 Memory Overhead 96
5.7 Experimental Evaluation 98
5.7.1 Experimental Setup 98
5.7.2 Experimental Results 99
5.8 Conclusions 102

6 Implementation and Execution of Adaptive Streaming Applications 103
6.1 Problem Statement 104
6.2 Contributions 104
6.3 Related Work 105
6.4 K-Periodic Schedules (K-PS) 106
6.5 Extension of the MOO Transition Protocol 107
6.6 Implementation and Execution Approach for MADF 110
6.6.1 Generic Parallel Implementation and Execution Approach 110
6.6.2 Demonstration of Our Approach on LITMUS^{RT} 112
6.7 Case Studies 115
6.7.1 Case Study 1 116
6.7.2 Case Study 2 119
6.8 Conclusions 122

7 Summary and Conclusions 123

Bibliography 127

Summary 137

Samenvatting 139

List of Publications 141

Curriculum Vitae 143

Acknowledgments 145

List of Figures

1.1 Samsung Exynos 5422 MPSoC [70]. 6
1.2 Overview of the research questions and contributions in this thesis using a design flow. 10

2.1 Example of an MADF graph (G_1). 20
2.2 Two modes of the MADF graph in Figure 2.1. 20
2.3 Execution of two iterations of both modes SI^1 and SI^2. (a) Mode SI^1 in Figure 2.2(a). (b) Mode SI^2 in Figure 2.2(b). 22
2.4 Execution of graph G_1 with two mode transitions under the MOO protocol. 22
2.5 Execution of graph G_1 with a mode transition from mode SI^2 to mode SI^1 under the MOO protocol and the SPS framework. . . . 31
2.6 Execution of graph G_1 with a mode transition from mode SI^2 to mode SI^1 under the MOO protocol and the SPS framework with task allocation on two processors. 33

3.1 A **cyclic** CSDF graph G. The backward edge E_5 in G has 2 initial tokens that are represented with black dots. 39
3.2 The SPS of the CSDF graph G in Figure 3.1 without considering the backward edge E_5. Up arrows are job releases and down arrows job deadlines. 39
3.3 The $GSPS$ of the CSDF graph G in Figure 3.1. 40
3.4 Production and consumption curves on edge $E_u = (A_i, A_j)$. . . 41

4.1 An SDF graph G. 58
4.2 Equivalent CSDF graphs of the SDF graph G in Figure 4.1 obtained by (a) replicating actor A_5 by factor 2 and (b) replicating actors A_3 and A_4 by factor 2. 58

4.3 A strictly periodic execution of tasks corresponding to the actors in: (a) the SDF graph G in Figure 4.1 and (b) the CSDF graph G' in Figure 4.2(a). The x-axis represents the time. 60
4.4 Memory and latency reduction of our algorithm compared to the related approach with the same number of processors. . . . 71
4.5 Total number of task replications needed by FFD-EP and our proposed algorithm. 72
4.6 Memory and latency reduction of our algorithm compared to EDF-sh [92] for real-life applications on different heterogeneous platforms. 74

5.1 An SDF graph G. 82
5.2 The (a) SPS and (b) scaled SPS of the (C)SDF graph G in Figure 5.1. Up arrows represent job releases, down arrows represent job deadlines. Dotted rectangles show the increase of the tasks execution time when using the VFS mechanism. 83
5.3 Our proposed periodic schedule of graph G in Figure 5.1. In this schedule, graph G periodically executes according to schedules of operating mode SI^1 and operating mode SI^2 in Figure 5.2(a) and Figure 5.2(b), respectively. Note that this schedule repeats periodically. $o_{12} = 5$ and $o_{21} = 0$. 86
5.4 Normalized energy consumption of the scaled scheduling and our proposed scheduling of the graph G in Figure 5.1 for a wide range of throughput requirements. 87
5.5 (a) Switching scheme, (b) Associated energy consumption of the switching scheme and (c) Token production function Z(t). 88
5.6 Input and Output buffers. 90
5.7 Token consumption function $Z'(t)$. Note that, $o_{HL} + o_{LH} = o'_{HL} + o'_{LH} = \delta^{H\to L} + \delta^{L\to H}$. 97
5.8 Normalized energy consumption vs. throughput requirements. 100
5.9 Total buffer sizes needed in our scheduling approach for different applications. Note that the y axis has a logarithmic scale. . . 101

6.1 (a) An MADF graph G_1 (taken from Section 2.1.2). (b) The allocation of actors in graph G_1 on four processors. 108
6.2 Two modes of graph G_1 in Figure 2.1 (taken from Section 2.1.2 with modified WCET of the actors). 108
6.3 Execution of both modes SI^1 and SI^2 under a K-PS. 109

6.4 Execution of G_1 with two mode transitions under (a) the MOO protocol, and (b) the extended MOO protocol with the allocation shown in Figure 6.1(b). 109
6.5 Mode transition of G_1 from mode SI^2 to mode SI^1 (from (a) to (f)). The control actor and the control edges are omitted in figures (b) to (f) to avoid cluttering. 111
6.6 MADF graph of the Vocoder application. 117
6.7 The execution time of control actor A_c for applications with different numbers of actors. 119
6.8 CSDF graph of MJPEG encoder. 120
6.9 (a) The video frame production of the MJPEG encoder application over time for the throughput requirement of 5.2 frames/second. (b) Normalized energy consumption of the application for different throughput requirements. 121

List of Tables

2.1 Summary of mathematical notations. 17

3.1 Benchmarks used for evaluation. 47
3.2 Comparison of different scheduling frameworks. 48

4.1 Throughput $\mathcal{R}$ (1/time units), latency $\mathcal{L}$ (time units), memory requirements $\mathcal{M}$ (bytes), and number of processors m for G under different scheduling/allocation approaches. 63
4.2 Benchmarks used for evaluation taken from [23]. 68
4.3 Comparison of different scheduling/allocation approaches. . 69
4.4 Runtime (in seconds) comparison of different scheduling/allocation approaches. 73

5.1 Operating modes for graph G 85
5.2 Benchmarks used for evaluation. 99

6.1 Performance results of each individual mode of Vocoder. . . . 116
6.2 Performance results for all mode transitions of Vocoder (in ms). 118
6.3 The specification of modes SI^1 and SI^2 in MJPEG encoder application 121

List of Abbreviations

BFD	Best-Fit Decreasing
CDP	Constrained-Deadline Periodic
CSDF	Cyclo-Static Data Flow
DSE	Design Space Exploration
DVFS	Dynamic VFS
EDF	Earliest Deadline First
EE	Energy Efficient
FFD	First-Fit Decreasing
FFID-EDF	First-Fit Increasing Deadlines EDF
FIFO	First-In First-Out
GSPS	Generalized Strictly Periodic Scheduling
HRT	Hard Real-Time
IDP	Implicit-Deadline Periodic
MADF	Mode-Aware Data Flow
MCR	Mode Change Request
MoC	Model of Computation
MOO	Maximum-Overlap Offset
MPSoC	Multi-Processor System-on-Chip

PE	Performance Efficient
RM	Rate Monotonic
RTOS	Real-time Operating System
SDF	Synchronous Data Flow
SPS	Strictly Periodic Scheduling
SRT	Soft Real-Time
TDP	Thermal Design Power
VFS	Voltage-Frequency Scaling
WCET	Worst-Case Execution Time
WFD	Worst-Fit Decreasing