

QUALITY MANAGEMENT
OF REAL-TIME DATA
IN CYBER-PHYSICAL SYSTEMS

WANG JIANTAO

DOCTOR OF PHILOSOPHY

CITY UNIVERSITY OF HONG KONG

JUNE 2013

Abstract

This thesis investigates how to manage the quality of real-time data in a cyber-physical system. We present intensive theoretical analysis and extensive performance evaluation of the designed algorithms. The major work is outlined as follows.

Temporal data consistency is very important in cyber-physical systems. A cyber-physical system is a system featuring a tight combination of, and coordination between, the system's computational and physical elements. In a cyber-physical system, the update tasks that aim at updating real-time data always have transmission delays or jitters from the sources to the destinations. In Chapter 2, we present how to maintain the temporal consistency of real-time data when different types of jitters are taken into consideration. We first present a novel algorithm to *guarantee* the temporal consistency of real-time data when the jitter of each update task has a limited bound. Based on the novel algorithm, an enhanced algorithm is presented to provide *statistical* temporal consistency of real-time data when the jitter of each update task is unbounded and unpredictable. Performance evaluation shows that both algorithms can provide good temporal data consistency when different kinds of jitters are considered.

Maintaining the temporal consistency of real-time data has a close relationship with meeting the deadlines of control tasks. Unfortunately, these two goals, maintaining temporal data consistency and meeting the control tasks' deadlines, conflict with each other and are difficult to be achieved at the same time. To address the conflict, in Chapter 3, we present a *fixed* priority co-scheduling algorithm. It uses periodic update tasks to maintain the temporal consistency of real-time data. It judiciously decides the priority order among all the update tasks and control tasks

so that the constructed co-scheduling can satisfy the deadline constraints of all the control tasks while maximizing the qualities of real-time data. However, the performance of the fixed priority co-scheduling algorithm decreases with the increasement of the system workload. In chapter 4, instead of using a fixed priority co-scheduling, a *dynamic* co-scheduling algorithm with the purposes to meet the deadlines of the control tasks and maximize the *quality of control* offered by the control tasks are presented. Extensive simulation results show that the dynamic co-scheduling algorithm can effectively improve the system schedulability and the temporal consistency of real-time data.

In Chapter 5, we present a novel multi-version B⁺-tree index structure for indexing multi-versions of data objects in an multi-version database over flash memory. The proposed multi-version index structure is divided into two levels. At the higher level, a multi-version index is maintained for keeping successive versions of each data object. These versions are allocated consecutively in a version block. Each version block consists of a pre-defined number of consecutive pages. At the lower level, a version array is used for searching a specific data version within a version block. Extensive simulation results show that the overhead for managing the index in processing update operations can be greatly reduced. At the same time, the number of accesses to the index in processing version-range queries is greatly reduced.

Contents

Abstract	i
Table of Contents	v
List of Figures	ix
List of Tables	xii
1 Introduction	1
2 Maintain Temporal Data Consistency in Distributed Sensing and Control Systems	5
2.1 Related Work	8
2.2 Backgrounds and Preliminaries	11
2.2.1 Temporal Consistency of Real-Time Data	11
2.2.2 Maintaining Temporal Consistency with Jitter	14
2.2.3 The More-Less Method and its Limitations	16
2.3 Jitter-Based More-Less	18
2.3.1 Theoretical Preliminaries	19
2.3.2 <i>JB-ML</i> Approach	21
2.3.3 Comparison of <i>JB-ML</i> with More-Less	23
2.4 Statistical Jitter-Based More-Less	26
2.4.1 Statistical Guarantee in Temporal Consistency	27
2.4.2 Principles of SJB-ML	28
2.4.3 Algorithm Details	29

2.5	Performance Evaluation	35
2.5.1	Simulation Model and Parameters	36
2.5.2	Experiment 1: Comparison between <i>JB-ML</i> and <i>ML</i>	38
2.5.3	Experiment 2: Performance of <i>SJB-ML</i>	42
2.5.4	Experiment 3: Co-Scheduling with User Transactions	49
2.6	Summary	56
3	Co-Scheduling with Static Priorities	57
3.1	System Model	59
3.1.1	Update Task Model and Assumptions	60
3.1.2	Control Task Model and Assumptions	61
3.1.3	Data Staleness	62
3.1.4	Quality of Data (QoD)	62
3.1.5	Quality of Control (QoC)	63
3.2	Fixed Priority Co-Scheduling Algorithm	64
3.2.1	The Baseline Algorithms	64
3.2.2	The Periodic Co-Scheduling Problem	65
3.2.3	The Design Principles of <i>PCS</i>	67
3.2.4	The Details of the <i>PCS</i> Algorithm	68
3.2.5	An Example of the <i>PCS</i>	72
3.3	Performance Evaluation	73
3.3.1	Simulation Model and Metrics	74
3.3.2	Experimental Results and Discussions	75
3.4	Summary	80
4	Co-Scheduling with Dynamic Priorities	83
4.1	Least Idle Slot First Co-Scheduling Algorithm	84
4.2	Performance Evaluation	87
4.2.1	Simulation Model and Parameters	88
4.2.2	Performance Results and Discussions	89
4.3	Summary	97

5	Blocked-based Multi-version B⁺-tree for Flash-based Database Systems	100
5.1	Related Work	103
5.2	Preliminaries	105
5.2.1	Flash Memory Storage Systems	105
5.2.2	Multi-version B ⁺ -Tree (MVBT)	107
5.2.3	Problems and Motivation	108
5.3	System Model and Operations	110
5.3.1	System Model and Assumptions	110
5.3.2	Write and Read Operations	111
5.4	The Block-based MVBT (BbMVBT)	113
5.4.1	Principles of BbMVBT	113
5.4.2	An Example of BbMVBT	116
5.4.3	Operations in BbMVBT	119
5.5	Purging & Version Block Size	121
5.5.1	Purging of Data Versions and Index Nodes	121
5.5.2	The Determination of Version Block Size	123
5.6	Analysis of BbMVBT	125
5.6.1	Run-time Analysis of Update Transactions	126
5.6.2	Run-time Analysis of User Queries	127
5.6.3	Space Analysis	129
5.7	Performance Evaluation	130
5.7.1	System Model and Parameters	130
5.7.2	Experimental Results	135
5.8	Summary	138
6	Conclusions	141
6.1	Summary	141
6.2	Future Work	142
6.2.1	Optimization of the Co-scheduling Problem	142

6.2.2	Multi-version R-tree for Spatial Temporal Data over Flash Memory	144
6.2.3	Implementation of the Complete System	144
	References	145
	List of Publications	155