Optimality of Heuristic Schedulers in Utility Accrual Real-time Scheduling Environments

Veena Basavaraj

Thesis submitted to the faculty of the Virginia Polytechnic Institute and State University in partial fulfillment of the requirements for the degree of

Master of Science

In

Computer Science

Godmar V. Back, Chair
E. Douglas Jensen
Kirk W. Cameron

July 6, 2006
Blacksburg, Virginia

Keywords: real-time, time-utility functions, utility accrual, optimality

Copyright @ 2006, Veena Basavaraj
Optimality of Heuristic Schedulers in Utility Accrual Real-time Scheduling Environments

Veena Basavaraj

(ABSTRACT)

Scheduling decisions in soft real-time environments are based on a utility function. The goal of such schedulers is to use a best-effort approach to maximize the utility function and ensure graceful degradation at overloads. Utility Accrual (UA) schedulers use heuristics to maximize the accrued utility. Heuristic-based scheduling do not always yield the optimal schedule even if there exists one because they do not explore the entire search space of task orderings. In distributed systems, local UA schedulers use the same heuristics along with deadline decomposition for task segments.

At present, there has been no evaluation and analysis of the degree to which these polynomial-time, heuristic algorithms succeed in maximizing the total utility accrued. We implemented a preemptive, off-line static scheduling algorithm that performs an exhaustive search of all the possible task orderings to yield the optimal schedules. We simulated two important online dynamic UA schedulers, DASA-ND and LBESA for different system loads, task models, utility and load distribution patterns, and compared their performance with their corresponding optimal schedules.

Our experimental analysis indicates that for most scenarios, both DASA-ND and LBESA create optimal schedules. When task utilities are equal or form a geometric sequence with an order of magnitude difference in their utility values, UA schedulers show more than 90% probability of being optimal for single-node workloads. Even though deadline decomposition substantially improves the optimality of both DASA-ND and LBESA under different scenarios for distributed workloads, it can adversely affect the scheduling decisions for some task sets we considered.
To My Parents,

Thank you for your strong support and encouragement.
Acknowledgements

I express my sincere gratitude to Dr. Godmar Back for his continued support and invaluable guidance. I admire his persistence and discipline that motivated me to work under him. I am fortunate to have had this opportunity and will treasure the things learnt while working with him. I express my deepest appreciation and thank him for the dedication and timely co-operation he has shown towards my thesis work.

I acknowledge the effort of my committee members E. Douglas Jensen & Dr. Kirk Cameron for their co-operation and involvement in the final stages of my work.

My sincere thanks to all the members of CAAPC, Virginia Tech, whose support and encouragement has immensely helped me complete this work.

Special thanks to all my friends, co-graduate students, from whom I derived inspiration to take up work that seemed to be beyond my reach in the initial stages.
Table of Contents

ACKNOWLEDGEMENTS ................................................................................................................ IV

TABLE OF CONTENTS .................................................................................................................. V

LIST OF ABBREVIATIONS ........................................................................................................ VIII

LIST OF FIGURES ................................................................................................................... VIII

LIST OF TABLES ..................................................................................................................... XVIII

CHAPTER 1  INTRODUCTION AND OVERVIEW ........................................................................ 1

1.1  THESIS SUMMARY .............................................................................................................. 1

1.2  THESIS CONTRIBUTION .................................................................................................... 2

1.3  THESIS ORGANIZATION ..................................................................................................... 3

CHAPTER 2  BACKGROUND ........................................................................................................ 5

2.1  SYSTEM MODEL ................................................................................................................ 5

2.1.1  Task Models .................................................................................................................. 6

2.1.1.1  PERIODIC TASK MODEL ........................................................................................................... 6

2.1.1.2  APERIODIC TASK MODEL ......................................................................................................... 7

2.2  DYNAMIC TASK SCHEDULING ON UNIPROCESSORS ......................................................... 8

2.2.1  Deadline-based Scheduling ............................................................................................. 8

2.2.2  Utility Accrual Scheduling ............................................................................................... 9

2.2.3  Types of TUF .................................................................................................................... 9

2.2.4  Heuristic Scheduling Algorithms ..................................................................................... 10

2.2.4.1  LBESA (α = 1 | β = indep | γ = Umax) ......................................................................................... 10

2.2.4.2  DASA-ND (α = 1 | β = prec | γ = Umax) .................................................................................... 11

2.3  DYNAMIC TASK SCHEDULING IN DISTRIBUTED SYSTEMS .............................................. 12

2.3.1  Decomposition Techniques ............................................................................................. 12

2.3.1.1  DEADLINE SLICING ............................................................................................................... 13

2.3.1.2  TUF SCALING .......................................................................................................................... 14

CHAPTER 3  OPTIMALITY OF UTILITY ACCRUAL SCHEDULERS ............................................. 16

CHAPTER 4  SCHEDULING ALGORITHM FOR OPTIMAL SCHEDULES .................................. 20

4.1  DESIGN RATIONALE FOR THE OPTIMAL SCHEDULER .................................................. 20

4.2  OPTIMAL SCHEDULER: A HIGH-LEVEL DESCRIPTION .................................................. 20
List of Abbreviations

UA: Utility Accrual  
TUFs: Time/Utility Functions  
DT: Distributed Task  
DASA-ND: Dependent Activity Scheduling Algorithm with No Resource Dependencies  
LBESA: Locke’s Best Effort Scheduling Algorithm  
PUD: Potential Utility Density  
PVD: Potential Value Density  
EDF: Earliest Deadline First  
SLEQF: Slicing based on Equal Flexibility  
SCEQF: Scaling based on Equal Flexibility  
STEPS: Step Downward Function Scaling  
C.D.F: Cumulative Distribution Function  
P.D.F: Probability Distribution Function

List of Figures

Figure 2-1: DISTRIBUTED TASKS & THEIR CHARACTERISTICS ALONG WITH THEIR REMOTE CALL PATTERN ............... 6  
Figure 2-2: PERIODIC TASKS REPRESENTATION ........................................................................................................... 7  
Figure 2-3: APERIODIC TASK REPRESENTATION........................................................................................................... 7  
Figure 2-4: EXAMPLE TUFs ...........................................................................................................................................10  
Figure 2-5: DEADLINE DECOMPOSITION AIDS UA SCHEDULERS ................................................................. 14  
Figure 2-6: STEPS, TUF SCALING TECHNIQUE ............................................................................................................. 15  
Figure 4-1: SEARCH TREE REPRESENTATION .............................................................................................................22  
Figure 4-2: TASKS VS RUN-TIME OF OPTIMAL-ALGORITHM, PERIODIC TASKS, HIGH LOAD ........................................26  
Figure 4-3: TASKS VS # OF SEARCH NODES/SEARCH PATHS FOR PERIODIC TASKS, HIGH LOAD ..........................26  
Figure 4-4: TASKS VS RUN-TIME OF OPTIMAL ALGORITHM, APERIODIC TASKS, LOW LOAD...............................27  
Figure 4-5: TASKS VS RUN-TIME OF OPTIMAL ALGORITHM, APERIODIC TASKS, HIGH LOAD ...........................28  
Figure 4-6: TASK-SEGMENTS VS RUN-TIME OF OPTIMAL ALGORITHM, DISTRIBUTED APERIODIC TASKS ..........28  
Figure 4-7: TASKS VS SEARCH NODES, NODES = 3 FOR PERIODIC DISTRIBUTED WORKLOAD .........................29  
Figure 4-8: TASKS VS TREE BREADTH FOR DISTRIBUTED APERIODIC TASKS .....................................................30  
Figure 4-9: TASK VS TREE DEPTH FOR DISTRIBUTED APERIODIC TASKS ............................................................30  
Figure 6-1: LOAD VS DSR, OPTIMAL & DASA-ND FOR W1, EQUAL UTILITIES ONLY .............................................43
FIGURE 6-64: SCENARIO 14-W A10, DASA-ND& LBESA FAIL TO PRODUCE OPTIMAL RESULTS AT A LOAD = 150%.

FIGURE 7-1: BALANCED LOAD DISTRIBUTION, HERE 33% MULTIPLIED BY THE NUMBER OF NODES IS THE ACTUAL LOAD ON EACH NODE; HENCE, THE LOAD ON THESE NODES IS 100%. THIS IS CALLED THE SYSTEM LOAD. ........110

FIGURE 7-2: UNBALANCED (BIASED) LOAD DISTRIBUTION, THE LOAD ON NODE 1 IS 150%, WHICH HAPPENS TO BE THE MAXIMUM SYSTEM LOAD. ........................................................................................................................................110

FIGURE 7-3: LOAD VS DELTA AUR, DASA-ND, DW-1(TABLE 7-1), EQUAL UTILITIES, EQUAL DEADLINES AND EQUAL COST SEGMENTS FOR ALL TASKS ..................................................................................................................114

FIGURE 7-4: LOAD VS DELTA AUR, LBESA, DW-1(TABLE 7-1), EQUAL UTILITIES, EQUAL DEADLINES AND EQUAL COST SEGMENTS FOR ALL TASKS ........................................................................................................................................115

FIGURE 7-5: SCENARIO D1 - LBESA IS OPTIMAL, DASA-ND PRODUCES SUB-OPTIMAL RESULTS FOR DW-1, EQUAL UTILITIES, EQUAL DEADLINES AND EQUAL COST SEGMENTS FOR ALL TASKS ........................................................................................................................................116

FIGURE 7-6: C.D.F AND HISTOGRAM FOR DW-1, BALANCED LOAD, EQUAL UTILITIES, EQUAL DEADLINES AND EQUAL COST SEGMENTS FOR ALL TASKS ........................................................................................................................................117

FIGURE 7-7: C.D.F AND HISTOGRAM FOR DW-1, UNBALANCED LOAD, EQUAL UTILITIES, EQUAL DEADLINES AND EQUAL COST SEGMENTS FOR ALL TASKS ........................................................................................................................................117

FIGURE 7-8: C.D.F AND HISTOGRAM FOR DW-1, SLEQF, AND BALANCED LOAD, EQUAL UTILITIES, EQUAL DEADLINES AND EQUAL COST SEGMENTS FOR ALL TASKS ........................................................................................................................................118

FIGURE 7-9: C.D.F AND HISTOGRAM FOR DW-1, SLEQF, AND UNBALANCED LOAD, EQUAL UTILITIES, EQUAL DEADLINES AND EQUAL COST SEGMENTS FOR ALL TASKS ........................................................................................................................................118

FIGURE 7-10: LOAD VS DELTA AUR, DASA-ND DW-2, EQUAL UTILITIES, EQUAL DEADLINES AND EQUAL COST SEGMENTS FOR ALL TASKS, VARYING TASK SEGMENTS AMONG TASKS ........................................................................................................................................120

FIGURE 7-11: LOAD VS DELTA AUR, LBESA, DW-2, EQUAL UTILITIES, EQUAL DEADLINES AND EQUAL COST SEGMENTS FOR ALL TASKS, VARYING TASK SEGMENTS AMONG TASKS ........................................................................................................................................120

FIGURE 7-12: C.D.F AND HISTOGRAM FOR DASA-ND DW-2, EQUAL UTILITIES, EQUAL DEADLINES AND EQUAL COST SEGMENTS FOR ALL TASKS, VARYING TASK SEGMENTS AMONG TASKS ........................................................................................................................................121

FIGURE 7-13: C.D.F AND HISTOGRAM FOR LBESA DW-2, EQUAL UTILITIES, EQUAL DEADLINES AND EQUAL COST SEGMENTS FOR ALL TASKS, VARYING TASK SEGMENTS AMONG TASKS ........................................................................................................................................121

FIGURE 7-14: C.D.F AND HISTOGRAM FOR DASA-ND WITH SLEQF, LBESA DW-2, EQUAL UTILITIES, EQUAL DEADLINES AND EQUAL COST SEGMENTS FOR ALL TASKS, VARYING TASK SEGMENTS AMONG TASKS ........................................................................................................................................123

FIGURE 7-15: C.D.F AND HISTOGRAM FOR LBESA WITH SLEQF, LBESA DW-2, EQUAL UTILITIES, EQUAL DEADLINES AND EQUAL COST SEGMENTS FOR ALL TASKS, VARYING TASK SEGMENTS AMONG TASKS ........................................................................................................................................123

FIGURE 7-16: SCENARIO D2 - DASA-ND AND LBESA PRODUCE SUB-OPTIMAL RESULTS FOR DW-2, EQUAL UTILITIES, EQUAL DEADLINES, EQUAL COST SEGMENTS, AND VARYING NUMBER SEGMENTS AMONG THE TASKS ........................................................................................................................................124

FIGURE 7-17: SCENARIO D2 - LBESA COMPARED WITH LBESA WITH SLEQF ........................................................................................................................................125
Figure 7-18: Load vs Delta AUR, DASA-ND DW-3 (Table 7-2), Geometric Utilities, CR = 2, Equal deadlines and Equal cost segments for all tasks, Equal task segments among tasks

Figure 7-19: Load vs Delta AUR, LBESA DW-3 (Table 7-2), Geometric Utilities, CR = 2, Equal deadlines and Equal cost segments for all tasks, Equal task segments among tasks

Figure 7-20: Scenario - D3, Both DASA-ND, and LBEA are sub-optimal for workload DW-3, Geometric Utilities, CR = 2, Equal deadlines and Equal cost segments for all tasks, Equal task segments among tasks

Figure 7-21: CDF and Histogram for DASA-ND, Balanced Load, DW-3 (Table 7-2), Geometric Utilities, CR = 2, Equal deadlines and Equal cost segments for all tasks, Equal task segments among tasks

Figure 7-22: CDF and Histogram for DASA-ND, Unbalanced Load DW-3 (Table 7-2), Geometric Utilities, CR = 2, Equal deadlines and Equal cost segments for all tasks, Equal task segments among tasks

Figure 7-23: CDF and Histogram for LBESA, Balanced Load DW-3, Geometric Utilities, CR = 2, Equal deadlines and Equal cost segments for all tasks, Equal task segments among tasks

Figure 7-24: CDF and Histogram for LBESA, Unbalanced Load DW-3, Geometric Utilities, CR = 2, Equal deadlines and Equal cost segments for all tasks, Equal task segments among tasks

Figure 7-25: CDF and Histogram for DASA-ND with SLEQF, DW-3, Geometric Utilities, CR = 2 Equal deadlines and Equal cost segments for all tasks, Equal task segments among tasks

Figure 7-26: CDF and Histogram for LBESA with SLEQF, DW-3, Geometric Utilities, CR = 2, Equal deadlines and Equal cost segments for all tasks, Equal task segments among tasks

Figure 7-27: Load vs Delta AUR, DASA-ND DW-4 (Table 7-2), Geometric Utilities, CR = 10, Equal deadlines and Equal cost segments for all tasks, Unequal task segments among tasks

Figure 7-28: Load vs Delta AUR, LBESA DW-4 (Table 2), Geometric Utilities, CR = 10, Equal deadlines and Equal cost segments for all tasks, Unequal task segments among tasks

Figure 7-29: CDF and Histogram for DASA-ND, Balanced Load, and DW-4, Geometric Utilities, CR = 10, Equal deadlines and Equal cost segments for all tasks, Unequal task segments among tasks

Figure 7-30: CDF and Histogram for DASA-ND, Unbalanced Load, and DW-4, Geometric Utilities, CR = 10, Equal deadlines and Equal cost segments for all tasks, Unequal task segments among tasks

Figure 7-31: CDF and Histogram for LBESA, Balanced Load, and DW-4, Geometric Utilities, CR = 10, Equal deadlines and Equal cost segments for all tasks, Equal task segments among tasks

Figure 7-32: CDF and Histogram for LBESA, Unbalanced Load, and DW-4, Geometric Utilities, CR = 10, Equal deadlines and Equal cost segments for all tasks, Equal task segments among tasks
Figure 7-33: C.D.F and histogram with SLEQF, Balanced Load, and DW-4, Geometric Utilities, CR = 10, Equal deadlines and Equal cost segments for all tasks, Equal task segments among tasks

Figure 7-34: C.D.F and histogram with SLEQF, Unbalanced Load, and DW-4, Geometric Utilities, CR = 10, Equal deadlines and Equal cost segments for all tasks, Equal Unequal task segments among tasks

Figure 7-35: Load vs Delta AUR, DASA-ND DW-5, Geometric Utilities, CR = 2, Equal deadlines and Equal cost segments for all tasks, Unequal task segments among tasks, Small No of Segs $\rightarrow$ LU and Large No of Segs $\rightarrow$ HU

Figure 7-36: Load vs Delta AUR, LBESA DW-5, Geometric Utilities, CR = 2, Equal deadlines and Equal cost segments for all tasks, Unequal task segments among tasks, Small No of Segs $\rightarrow$ LU and Large No of Segs $\rightarrow$ HU

Figure 7-37: Scenario D4 - DASA-ND and LBESA are optimal for DW-5, Geometric Utilities, CR = 2, Equal deadlines and Equal cost segments for all tasks, Unequal task segments among tasks, Small No of Segs $\rightarrow$ LU and Large No of Segs $\rightarrow$ HU

Figure 7-38: Scenario D4 - DASA-ND and LBESA are sub-optimal for DW-6, Geometric Utilities, CR = 2, Equal deadlines and Equal cost segments for all tasks, Unequal task segments among tasks, Small No of Segs $\rightarrow$ HU and Large No of Segs $\rightarrow$ LU

Figure 7-39: C.D.F and histogram, Balanced Load for DW-5, Geometric Utilities, CR = 2, Equal deadlines and Equal cost segments for all tasks, Unequal task segments among tasks, Small No of Segs $\rightarrow$ LU and Large No of Segs $\rightarrow$ HU

Figure 7-40: C.D.F and histogram, Unbalanced Load for DW-5, Geometric Utilities, CR = 2, Equal deadlines and Equal cost segments for all tasks, Unequal task segments among tasks, Small No of Segs $\rightarrow$ LU and Large No of Segs $\rightarrow$ HU

Figure 7-41: C.D.F and histogram, with SLEQF for Balanced Load for DW-5, Geometric Utilities, CR = 2, Equal deadlines and Equal cost segments for all tasks, Unequal task segments among tasks, Small No of Segs $\rightarrow$ LU and Large No of Segs $\rightarrow$ HU

Figure 7-42: C.D.F and histogram, with SLEQF for Unbalanced Load for DW-5, Geometric Utilities, CR = 2, Equal deadlines and Equal cost segments for all tasks, Unequal task segments among tasks, Small No of Segs $\rightarrow$ LU and Large No of Segs $\rightarrow$ HU

Figure 7-43: Load vs Delta AUR, DASA-ND DW-6, Geometric Utilities, CR = 2, Equal deadlines and Equal cost segments for all tasks, Unequal task segments among tasks, Small No of Segs $\rightarrow$ HU and Large No of Segs $\rightarrow$ LU

Figure 7-44: Load vs Delta AUR, LBESA DW-6, Geometric Utilities, CR = 2, Equal deadlines and Equal cost segments for all tasks, Unequal task segments among tasks, Small No of Segs $\rightarrow$ HU and Large No of Segs $\rightarrow$ LU
FIGURE 7-45: C.D.F AND HISTOGRAM, BALANCED LOAD FOR DW-6, GEOMETRIC UTILITIES, CR = 2, EQUAL
deadlines and Equal cost segments for all tasks, Unequal task segments among tasks, Small No
of Segs $\rightarrow$ HU and Large No of Segs $\rightarrow$ LU .......................................................................................................................... 144
FIGURE 7-46: C.D.F AND HISTOGRAM, UNBALANCED LOAD FOR DW-6, GEOMETRIC UTILITIES, CR = 2, EQUAL
deadlines and Equal cost segments for all tasks, Unequal task segments among tasks, Small No
of Segs $\rightarrow$ HU and Large No of Segs $\rightarrow$ LU .......................................................................................................................... 145
FIGURE 7-47: C.D.F AND HISTOGRAM, WITH SLEQF FOR BALANCED LOAD FOR DW-6, GEOMETRIC UTILITIES,
CR = 2, Equal deadlines and Equal cost segments for all tasks, Unequal task segments among
tasks, Small No of Segs $\rightarrow$ HU and Large No of Segs $\rightarrow$ LU .......................................................................................................................... 145
FIGURE 7-48: C.D.F AND HISTOGRAM, WITH SLEQF FOR UNBALANCED LOAD FOR DW-6, GEOMETRIC UTILITIES,
CR = 2, Equal deadlines and Equal cost segments for all tasks, Unequal task segments among
tasks, Small No of Segs $\rightarrow$ HU and Large No of Segs $\rightarrow$ LU .......................................................................................................................... 146
FIGURE 7-49: LOAD VS DELTA AUR, DASA-ND, DW-7 (TABLE 7-3), AND ARITHMETIC UTILITIES, CD =100, EQUAL
deadlines and Equal cost segments for all tasks, Equal segments among tasks ......................... 148
FIGURE 7-50: LOAD VS DELTA AUR, LBESA, DW-7 (TABLE 7-3), ARITHMETIC UTILITIES, CD=100, EQUAL
deadlines and Equal cost segments for all tasks, Equal segments among tasks ......................... 148
FIGURE 7-51: SCENARIO D5- BOTH DASA-ND AND LBESA ARE SUB-OPTIMAL FOR DW-7, ARITHMETIC UTILITIES,
Equal deadlines and Equal cost segments for all tasks,.................................................................... 150
FIGURE 7-52: C.D.F AND HISTOGRAM FOR DW-7, DASA-ND, AND BALANCED LOAD, ARITHMETIC UTILITIES,
Equal deadlines and Equal cost segments for all tasks,.................................................................... 151
FIGURE 7-53: C.D.F AND HISTOGRAM FOR DW-7, DASA-ND, UNBALANCED LOAD , ARITHMETIC UTILITIES,
Equal deadlines and Equal cost segments for all tasks,....................................................................... 152
FIGURE 7-54: C.D.F AND HISTOGRAM FOR DW-7, LBESA BALANCED LOAD, ARITHMETIC UTILITIES, EQUAL
deadlines and Equal cost segments for all tasks,............................................................................. 152
FIGURE 7-55: C.D.F AND HISTOGRAM FOR DW-7, LBESA UNBALANCED LOAD, ARITHMETIC UTILITIES, EQUAL
deadlines and Equal cost segments for all tasks,............................................................................. 152
FIGURE 7-56: C.D.F AND HISTOGRAM FOR DW-7 WITH SLEQF AND BALANCED LOAD, ARITHMETIC UTILITIES,
Equal deadlines and Equal cost segments for all tasks,.................................................................... 153
FIGURE 7-57: C.D.F AND HISTOGRAM FOR DW-7 WITH SLEQF AND UNBALANCED LOAD, ARITHMETIC UTILITIES,
Equal deadlines and Equal cost segments for all tasks,.................................................................... 153
FIGURE 7-58: LOAD VS DELTA AUR, DASA-ND, DW-8, ARITHMETIC UTILITIES, Equal deadlines and Equal
cost segments for all tasks, varying task segments among tasks, Small No of Segs $\rightarrow$ LU and
Large No of Segs $\rightarrow$ HU .......................................................................................................................... 154
FIGURE 7-59: LOAD VS DELTA AUR, LBESA DW-8, ARITHMETIC UTILITIES, Equal deadlines and Equal cost
segments for all tasks, varying task segments among tasks, Small No of Segs $\rightarrow$ LU and Large No
of Segs $\rightarrow$ HU .......................................................................................................................... 155
FIGURE 7-60: C.D.F AND HISTOGRAM FOR DW-8, DASA-ND, BALANCED LOAD, ARITHMETIC UTILITIES, EQUAL
deadlines and Equal cost segments for all tasks, varying task segments among tasks, Small No
of Segs → LU and Large No of Segs → HU ........................................................................................................ 155
FIGURE 7-61: C.D.F AND HISTOGRAM FOR DW-8, DASA-ND, AND UNBALANCED LOAD, ARITHMETIC UTILITIES,
Equal deadlines and Equal cost segments for all tasks, varying task segments among tasks,
Small No of Segs → LU and Large No of Segs → HU ................................................................................. 156
FIGURE 7-62: C.D.F AND HISTOGRAM FOR DW-8, LBESA BALANCED LOAD, ARITHMETIC UTILITIES, EQUAL
deadlines and Equal cost segments for all tasks, varying task segments among tasks, Small No
of Segs → LU and Large No of Segs → HU ........................................................................................................ 156
FIGURE 7-63: C.D.F AND HISTOGRAM FOR DW-8 LBESA UNBALANCED LOAD, ARITHMETIC UTILITIES, EQUAL
deadlines and Equal cost segments for all tasks, varying task segments among tasks, Small No
of Segs → LU and Large No of Segs → HU ........................................................................................................ 156
FIGURE 7-64: C.D.F AND HISTOGRAM FOR DW-8 WITH SLEQF AND BALANCED LOAD, ARITHMETIC UTILITIES,
Equal deadlines and Equal cost segments for all tasks, varying task segments among tasks,
Small No of Segs → LU and Large No of Segs → HU .................................................................................. 157
FIGURE 7-65: C.D.F AND HISTOGRAM FOR DW-8 WITH SLEQF AND UNBALANCED LOAD, ARITHMETIC UTILITIES,
Equal deadlines and Equal cost segments for all tasks, varying task segments among............. 158
FIGURE 7-66: LOAD VS DELTA AUR, BALANCED LOAD, LBESA, WITH SLEQF DW-8, ARITHMETIC UTILITIES,
Equal deadlines and Equal cost segments for all tasks, varying task segments among tasks,
Small No of Segs → LU and Large No of Segs → HU ............................................................................................................................. 160
FIGURE 7-67: LOAD VS DELTA AUR, UNBALANCED LOAD, LBESA, WITH SLEQF DW-8, ARITHMETIC
Utilities, Equal deadlines and Equal cost segments for all tasks, varying task segments among
tasks, Small No of Segs → LU and Large No of Segs → HU ........................................................................ 160
FIGURE 7-68: SCENARIO D6 - LBESA IS HIGHLY SUB-OPTIMAL FOR DW-8, ARITHMETIC UTILITIES, EQUAL
deadlines and Equal cost segments for all tasks, varying task segments among tasks, Small No
of Segs → LU and Large No of Segs → HU ........................................................................................................ 161
FIGURE 7-69: SCENARIO D6: LBESA COMPARED WITH LBESA WITH SLEQF ......................................................... 162
FIGURE 7-70: LOAD VS DELTA AUR, DASA-ND DW-9, ARITHMETIC UTILITIES, Equal deadlines and Equal
cost segments for all tasks, varying task segments among tasks, Small No of Segs → HU and
Large No of Segs → LU ........................................................................................................................................ 163
FIGURE 7-71: LOAD VS DELTA AUR, LBESA DW-9, ARITHMETIC UTILITIES, Equal deadlines and Equal cost
segments for all tasks, varying task segments among tasks, Small No of Segs → HU and Large No
of Segs → LU ........................................................................................................................................ 163
FIGURE 7-72: C.D.F AND HISTOGRAM, DASA-ND, BALANCED LOAD, DW-9, ARITHMETIC UTILITIES, EQUAL
deadlines and Equal cost segments for all tasks, varying task segments among tasks, Small No
of Segs → HU and Large No of Segs → LU ........................................................................................................ 164
List of Tables

TABLE 6-1: WORKLOADS WITH EQUAL TASK UTILITIES ................................................................. 42
TABLE 6-2: WORKLOADS WITH TASK UTILITIES IN GEOMETRIC SEQUENCE .............................. 46
TABLE 6-3: AVERAGE AUR VALUES FOR W5, GEOMETRIC UTILITY, CR=2 .................................. 47
TABLE 6-4: WORKLOAD WITH GEOMETRIC SEQUENCE TASK UTILITIES AND ORDERED TASK DEADLINES .......... 55
TABLE 6-5: WORKLOAD W6 WITH DIFFERENT TASK DEADLINES .................................................. 60
TABLE 6-6: WORKLOADS WITH TASK UTILITIES IN ARITHMETIC SEQUENCE ............................. 66
TABLE 6-7: AVERAGE AUR VALUES FOR W9, ARITHMETIC UTILITY, CD= 1 ................................. 67
TABLE 6-8: WORKLOAD WITH DIFFERENT TASK DEADLINES FOR ARITHMETIC SEQUENCE TASK UTILITIES ........... 79
TABLE 6-9 : WORKLOADS WITH STOCHASTIC EXECUTION COSTS ...................................................... 85
TABLE 6-10 : AVERAGE AUR VALUES FOR W-S6-1 ......................................................................... 86
TABLE 6-11 : AVERAGE AUR VALUES FOR W-S 10-1 ...................................................................... 87
TABLE 6-12: WORKLOADS FOR APERIODIC TASK MODEL ............................................................. 90
TABLE 6-13 : SUMMARY OF RESULTS FOR SINGLE-NODE .................................................................. 105
TABLE 7-1 : DISTRIBUTED PERIODIC TASKS WITH EQUAL UTILITIES, EQUAL DEADLINES AND CONTROLLED LOAD DISTRIBUTION, EITHER BALANCED OR UNBALANCED .................................................. 112
TABLE 7-2: DISTRIBUTED PERIODIC TASKS WITH GEOMETRIC SEQUENCE UTILITIES AND CONTROLLED LOAD
DISTRIBUTION WITH EITHER BALANCED OR UNBALANCED........................................................................................................126

TABLE 7-3: DISTRIBUTED PERIODIC TASKS WITH ARITHMETIC SEQUENCE UTILITIES AND CONTROLLED LOAD
DISTRIBUTION WITH EITHER BALANCED OR UNBALANCED........................................................................................................147

TABLE 7-4: DISTRIBUTED PERIODIC TASKS WITH EQUAL UTILITIES, UNIFORM DISTRIBUTION SYSTEM LOAD, AND
FIXED LOCAL LOAD.............................................................................................................................................................................167

TABLE 7-5: DISTRIBUTED PERIODIC TASKS WITH GEOMETRIC UTILITIES, RANDOM SYSTEM LOAD, FIXED LOCAL
LOAD.................................................................................................................................................................................................................174

TABLE 7-6: DISTRIBUTED PERIODIC TASKS WITH ARITHMETIC UTILITIES, RANDOM SYSTEM LOAD, AND FIXED LOCAL
LOAD.................................................................................................................................................................................................................181

TABLE 7-7: DISTRIBUTED APERIODIC TASKS WITH DIFFERENT UTILITY PATTERNS ........................................................................186

TABLE 7-8: SUMMARY OF RESULTS FOR DISTRIBUTED WORKLOADS ............................................................................................203

TABLE 7-9: SUMMARY OF RESULTS FOR DISTRIBUTED WORKLOADS WITH < 0.1 DIFFERENCE IN DELTA AUR ...........204

TABLE 7-10: LEGEND FOR TABLE 7-8 AND TABLE 7-9 ........................................................................................................................................205
Chapter 1 Introduction and Overview

1.1 Thesis Summary

Scheduling algorithms in soft real-time systems use a best-effort approach to schedule tasks based on a utility function. Utility is a measure of the benefit gained from completing a task. Utility Accrual (UA) scheduling is one such scheduling paradigm defined for soft real-time tasks on uniprocessors (single-node). UA schedulers use time/utility functions (TUFs) [Jensen et al., 1985] to model the timing constraints of real-time tasks. Unlike optimal algorithms that have super-exponential-time complexity, UA scheduling algorithms complete in polynomial-time and do not always find the optimal schedule for any given task set, but instead use heuristics to maximize the utility accrued. The utility function aims to finish the most important tasks for a given task set and accrue the maximum possible utility to the system. DASA-ND (Dependent Activity Scheduling Algorithm with no dependencies) [Clark, 1990] and LBESA (Locke’s Best Effort Scheduling Algorithm) [Locke, 1986] are two important UA scheduling algorithms. These uniprocessor scheduling algorithms are also applied in distributed environments, where tasks traverse multiple nodes before they finish.

Unlike distributed scheduling mechanisms where schedulers on different nodes exchange and synchronize system and task information among them, the UA schedulers operate only on locally available task information. As in uniprocessors, they use the same heuristics to maximize the local utility accrued on each node. Such scheduling schemes are less complex than the ones used in distributed scheduling and complete in polynomial-time due to the use of heuristics. Though they complete in polynomial time, scheduling decisions for task segments made in isolation at each local node do not guarantee the end-end completion of distributed task segments. The likelihood of completing the high utility distributed tasks is lower than in uniprocessor environments. Recent work shows that decomposing the end-end task deadline and TUF into sub-segment deadlines and sub-segment TUFs improves the performance of such UA schedulers in distributed systems [Wu et al., 2005].
Nevertheless, there exists no formal evaluation of the degree of optimality of these UA schedulers under different system loads in both uniprocessor and distributed environments. A formal evaluation of the optimality of these schedulers for different workloads would allow us to identify the specific system parameters, task properties and load patterns that are most congenial for such heuristic schedulers in both uniprocessor and distributed systems. It would also reveal the type of workloads for which such schedulers have a higher likelihood of producing sub-optimal results in real world applications.

We have designed and built an optimal scheduling algorithm that aids the formal evaluation of optimality of DASA-ND and LBESA. This scheduling algorithm outputs the optimal schedules for any given task set with step TUFs in both single and distributed UA scheduling environments. We have also created representative workloads using unbiased methods and explored interesting variations in system and task properties in order to answer the research questions at hand.

Our research evaluates and analyzes the probability that DASA-ND and LBESA are optimal at higher system loads on uniprocessors for different utility patterns and task models. We investigate workloads that have defined relationship between task utilities and costs. It further answers how the node local schedulers contribute to the global (system-wide) optimality in the distributed environments. We also evaluate the effect of different load distribution patterns on the UA schedulers in a distributed real-time environment. We finally assess how different decomposition techniques improve the optimality of UA schedulers for distributed workloads.

1.2 Thesis Contribution

The contributions of this thesis include,

1) Implementing an off-line static scheduling algorithm that creates optimal schedules for a given task set in both single-node and distributed UA scheduling environments.

2) Creating synthetic and representative workloads for different task models and utility patterns, both in uniprocessor and distributed environments to evaluate the degree of optimality of DASA-ND and LBESA.
3) Using systematic and unbiased workload generation methods, statistical procedures to generate task schedules for both, DASA-ND and LBESA in the OMNET++ simulation environment.

4) Finally, using the reference optimal scheduler, to generate the corresponding optimal schedules, evaluate, and analyze the optimality of DASA-ND and LBESA for these workloads.

Our experimental analysis indicates that in both the single-node and distributed case DASA-ND and LBESA show a better degree of optimality for periodic workloads than the aperiodic workloads. In the case for periodic task sets on a single-node, both DASA-ND and LBESA show a 100% probability of being optimal for workloads with equal utilities for all tasks. For one of the utility patterns we experimented with, both DASA-ND and LBESA show 100% probability of being optimal with < 0.01 difference in the accrued utility ratios. For most periodic workloads, DASA-ND performed marginally better than LBESA. For aperiodic workloads both, LBESA and DASA-ND show less than 0.5 probability of being optimal for a specific utility pattern we explored. LBESA is marginally better than DASA-ND for such workloads. Unbalanced load distribution among the system nodes degrades the optimality of UA schedulers for most of the utility patterns. Even though the deadline decomposition technique substantially improves the optimality of both DASA-ND and LBESA for periodic distributed workloads, it can adversely affect certain distributed workloads, irrespective of the utility pattern.

1.3 Thesis Organization

The rest of the thesis is organized as follows. Chapter 2 provides the required background information and introduces the terms and definitions used in the domain of soft real-time scheduling environments that apply to our work. Chapter 3 explains the research area, problem statement and our research goals in detail. Chapter 4 explains the design rationale and implementation details of the off-line scheduling algorithm implemented to evaluate the optimality of the DASA-ND and LBESA. Chapter 5 explains the workload generation methods, evaluation metrics, scheduling models, and the techniques used in computing and evaluating the optimality of heuristic-based UA schedulers, DASA-ND and LBESA. Chapter 6 reports on simulation results and the inferences derived from the experiments done on a single-node. In
Chapter 7, we discuss the results and inferences derived from experiments done in a distributed environment. Chapter 8 provides a summary of results and concludes with the important findings from our experimental analysis. It also discusses the future directions this work can take.
Chapter 2 Background

This chapter introduces the terms and definitions used in the domain of soft real-time scheduling environments that are applicable to our work. We initially discuss the task properties and the primary objectives of dynamic schedulers in soft real-time systems. We further motivate the objective of UA schedulers in gracefully handling system overloads. Finally, we explain the rationale behind the UA schedulers DASA-ND and LBESA, and their applicability in single-node and distributed real-time environments.

2.1 System Model

Real-time systems are characterized by their task models. A task is defined as an execution unit that is either running on the available CPU or waiting to execute in the future. A real-time task, has stringent timing requirements, termed as the “deadline,” and should complete within this time to achieve the desired output. In a pre-emptive real-time system, any executing task can be interrupted to allow other important tasks to continue executing. A real-time task that executes on a uniprocessor is completely defined by the following parameters;

i. arrival time, the time the task becomes available for execution
ii. computation time or the execution cost, the time required to complete the task execution
iii. laxity or slack, the time by which the task can be delayed before starting the executing
iv. utility, a value that signifies the relative significance of completing this task
v. deadline, defines the timing constraint of the task
vi. precedence constraints, defines the dependencies on other tasks
vii. resource constraints, defines the resources required by this task

The above description represents a simple task model applicable to a uniprocessor system. Extending this simple model to a distributed system, we define a Distributed Task (DT) as a task that transparently extends/retracts one or more system nodes based on its remote procedure call (RPC) pattern [Li et al., 2004]. In addition to the characteristics mentioned for the tasks on a single-node, DTs have one or more sub segments mapped to a specific node. Usually these sub
segments are scheduled using the end-end DT constraints; however, in some scenarios, the end-end DT constraint such as the DT deadline can be decomposed into segment deadlines relative to the segment costs to achieve better performance. Figure 2-1 represents two distributed tasks, DT1 and DT2, with their characteristics. DT1, has 3 segments (represented by SEG) and extends from Node 1 (N1) to Node 3 (N3) and finally completes on N2. DT2 also has 3 segments, but extends from Node 2 to Node 3 and retracts to Node 2.

Figure 2-1: Distributed tasks & their characteristics along with their remote call pattern

2.1.1 Task Models

This subsection describes the two important task arrival models we consider.

2.1.1.1 PERIODIC TASK MODEL

Periodic tasks arrive at regular intervals of time and thus consist of identical execution units, called instances. The time interval between the arrivals of successive instances is called the period or the cycle. The instances are identical only if the system uses a deterministic cost model. If not, these periodic instances can have varying costs that follow a distribution with a given expected value. Most often, we assume that the task deadlines are same as their period, but it is not mandatory. A typical example of periodic tasks in real world is the sensor and system monitoring systems that periodically activate instances to acquire and process data from their surroundings. Figure 2-2 shows the representation of periodic tasks with variations in task
properties. The horizontal line represents progression of time. \( \uparrow \) denotes the task arrival. \( \downarrow \) denotes the task deadline. The shaded rectangle represents the task execution time. Type 1 and Type 2 have identical units across different cycles. In Type 3, periodic instances have varying costs.

**Figure 2-2: Periodic Tasks representation**

### 2.1.1.2 APERIODIC TASK MODEL

Aperiodic tasks, represented in Figure 2-3 consist of a sequence of execution units activated at irregular time intervals. The inter-arrival times vary according to a probability distribution. Similarly, the execution costs of the aperiodic instances are random variables that follow a particular probability distribution. A typical example of aperiodic tasks includes the system interrupts that arise from interactions with the external environment.

**Figure 2-3: Aperiodic Task representation**
2.2 **Dynamic Task Scheduling on Uniprocessors**

Task scheduling is the process of finding an ordering of tasks on one or more processors while satisfying the timing, resource, and precedence constraints specified by the task properties.

### 2.2.1 Deadline-based Scheduling

Dynamic scheduling algorithms such as Earliest Deadline First (EDF) are considered optimal for scheduling independent and preemptive task sets [Buttazzo, 1997]. They assign priorities to tasks in the increasing order of their deadlines\(^1\). EDF ensures that all deadlines are met in a schedulable set but its performance degrades under system overloads. In system overloads, the total cost of all the tasks exceeds the available CPU time. System overloads may be transient with either bursty aperiodic task arrivals or cost overruns due to component failures. EDF degrades at overloads and suffers from the domino effect [Buttazzo, 1997]. The domino effect is a phenomenon in which the arrival of a new task with an earlier deadline can cause all the previously guaranteed tasks to miss their deadlines.

It is important to ensure graceful degradation at overloads by creating a feasible set of tasks that maximizes the scheduling criterion. Tasks are associated with utility values (importance) and these values represent the relative significance of the tasks. The objective of best-effort schedulers is to create a feasible set of tasks that maximizes the total utility gained. This optimization problem aims to find the best ordering that maximizes the total utility gained. A scheduling algorithm is said to be optimal if it is able to find a feasible ordering of tasks for any schedulable task set [Buttazzo, 1997]. In addition, if it is not possible to complete all tasks within their deadlines, the optimal scheduler maximizes the total usefulness to the system by scheduling the high utility tasks. According to [Garey and Johnson, 1979], creating optimal schedules for any given task set at different system loads with precedence & resource constraints using the scheduling criteria of maximizing the total utility is known to be NP-complete\(^2\). Such NP-

---

1 Assuming that priority based task scheduling is used in the underlying operating system.

2 A problem is classified as NP-complete if it can be shown that it is both NP-Hard and verifiable in polynomial time.
complete problems that require optimal solutions are addressed using exhaustive search algorithms that have $O(n!)$ i.e. super-exponential-time complexity. In dynamic scheduling environments, task arrival times are not known a priori and scheduling decisions are made during every task arrival and completion point. Exhaustive search algorithms are not practical in dynamic environments.

### 2.2.2 Utility Accrual Scheduling

Tasks’ utility (importance) is often not directly related to their urgency (deadline) i.e., the most urgent activity (i.e. one having an earlier deadline) might not have the highest utility to the system. Second, tasks might produce different utility to the system depending on the time they finish. TUFs [Jensen et al., 1985] model the relationship between the importance and urgency of the soft-real time tasks. UA Scheduling uses TUFs to model the timing constraints of tasks. UA schedulers follow EDF order under system underload. In system overload, these schedulers first select a feasible subset, which is then scheduled in EDF order. Maximizing the total utility accrued is the goal of these best-effort UA schedulers. Since searching through all possible task orderings to find the maximum total utility takes super-super-exponential time, UA schedulers employ heuristics. Heuristic-based schedulers complete in polynomial time making them suitable for dynamic scheduling environments. One such heuristic is the Potential Value Density (PVD). It is the ratio of the task utility gained if the task is run to completion to its computation time [Locke, 1986]. Tasks with higher PVD are preferred to the lower ones. We next illustrate a few simple TUFs.

### 2.2.3 Types of TUF

Figure 2-4 illustrates some of the example TUFs assigned to tasks. TUFs [Jensen et al., 1985] define the utility gained from completing a task as a function of its completion time. The simplest is the (a) STEP TUF that has a constant value utility represented by $\text{max}U$, from the time a task arrives to the time within which it completes. (b) SOFT-STEP TUF is a variation of STEP TUF, where after a certain point represented by $t_c$, the utility of the task linearly decreases. If the task completes exactly at its deadline, it accrues no utility to the system. (c) LINEAR TUF shows that the earlier the completion time, the higher the utility of the task. It accumulates no
utility if the task completes at its deadline. Our work is restricted to tasks with only STEP TUFs. Details on other types of TUFs and their application in real world scenarios are described in [Li et al., 2004].

\[\text{at: Arrival Time } \text{dt: deadline Time } \text{te: critical time}\]

![Example TUFs](image)

(a) STEP  
(b) SOFT-STEP  
(c) LINEAR

**2.2.4 Heuristic Scheduling Algorithms**

Scheduling algorithms can be classified based on the following three characteristics;

1) The number of machines supported by the algorithm, denoted by “α”
2) The task characteristics and resource dependencies it accommodates, denoted by “β” &
3) The optimality criterion used, denoted by “γ”

This systematic notation was proposed in [Graham et al., 1979]. We classify and compare the two heuristic UA schedulers, DASA-ND and LBESA using this systematic notation. Both DASA-ND and LBESA are uniprocessor UA scheduling algorithms, but differ in the task model and the type of TUFs they support. Maximizing the system-wide utility is the optimality criterion (\(U_{\text{max}}\)) used in both DASA-ND and LBESA.

**2.2.4.1 LBESA (α = 1 | β = indep | γ = \(U_{\text{max}}\))**

- LBESA handles tasks with arbitrarily shaped TUFs
- It supports both synchronous task activations and arbitrary task arrivals.
• LBESA handles varying task execution times derived from a probability distribution.
• LBESA only schedules independent tasks (indep) and cannot handle precedence constraints arising due to resource dependencies among tasks.
• It uses EDF ordering in case of underload.
• It detects system overload by comparing the available CPU time and the time required in completing the current tasks.
• At overloads, it uses a task rejection policy that removes tasks in non-decreasing order of their Potential Value Density from the deadline ordered tentative feasible set, starting with the smallest PVD task.
• Finally, the feasible set is a deadline ordered set of higher PVD tasks, having rejected the lower PVD tasks that would have made the set infeasible.
• LBESA does not specify any order to break deadline ties between tasks in the feasible set [Locke, 1986].

2.2.4.2 DASA-ND ($\alpha = 1 \mid \beta = \text{prec} \mid \gamma = U_{\text{max}}$)

• Unlike LBESA, DASA-ND handles tasks with only STEP TUFs.
• It supports both synchronous task activations and arbitrary task arrivals.
• Unlike LBESA, DASA-ND can handle dependent tasks and precedence constraints (prec) and thus tasks can share resources among them.
• It only supports deterministic task execution times.
• It also uses EDF ordering in case of system underload.
• At overloads, it considers the ready tasks in decreasing order of the Potential Utility Density (PUD) ³ of tasks and its dependents, starting with the highest PUD task.
• It then constructs the deadline ordered tentative feasible set, by inserting tasks in the decreasing PUD order.
• The tasks in the tentative schedule are ordered by deadline before the feasibility check is made. If inserting a new task renders the tentative schedule infeasible, DASA-ND rejects

³ PUD heuristic has the same definition as the PVD heuristic introduced earlier in 2.2.2
this task and continues with the next task in decreasing order of PUD, until all tasks are exhausted.

- Since the available tasks are ordered in PUD order, deadline ties are broken in PUD order [Clark, 1990].

### 2.3 Dynamic Task Scheduling in Distributed Systems

Uniprocessor scheduling algorithms such as DASA-ND and LBESA can also be applied to distributed real-time environments. Unlike distributed scheduling mechanisms, which exchange and synchronize task and load information across the system nodes, UA schedulers operate in isolation on local nodes; they order locally available tasks using the same heuristics used in uniprocessor environments [Li et al., 2004]. As explained in Figure 2-1, the task model consists of DTs that traverse multiple node boundaries and carry their execution context along with them. The execution context for the DT segments on the local nodes consists of the end-end DT properties such as DT deadline and DT TUF. End-end deadlines for segments do not correctly capture the amount of delay that can be introduced in scheduling the different segments at the local nodes. There are situations where the local nodes have additional local tasks that compete with the distributed segments. This global-local contention hinders the completion of the end-end DT since DT segments may be delayed at each local node scheduling local tasks [Wu et al., 2005]. As a solution to this problem, the end-end DT properties can be decomposed to create segment properties, such that the segments have a more equitable chance of completing on their respective local nodes.

#### 2.3.1 Decomposition Techniques

Two of the important decomposition techniques specified in [Wu et al., 2005] include deadline slicing and TUF scaling. Slicing based on Equal Flexibility (SLEQF) and Scaling based on Equal Flexibility (SCEQF) both use proportional deadline slicing. SCEQF not only slices the deadline, but also scales the height of the segment TUFs relative to the segment deadlines. According to [Wu et al., 2005] TUF scaling is applicable only with non STEP-TUFs, where sub-segments have different utility values depending on their relative position in the end-end DT. Step Downward Function Scaling (STEPS) is a TUF scaling technique mentioned in [Wu et al., 2005] that
converts any arbitrary TUF to a STEP-TUF and scales the height of TUF by a factor relative to
the height of local tasks. Each segment is assigned the DTs TUF. The segment deadlines remain
the same as the DTs end-end deadline. We next explain both techniques with an example.

2.3.1.1 DEADLINE SLICING

Deadline slicing decomposes the end-end DT deadline into segment deadlines. Without
decomposition, the earlier segments tend to use all the end-end DT slack and deprive the later
segments of any slack at their respective local nodes. Hence, deadline slicing proportionally
distributes the end-end slack to the segments based on the execution time. A longer segment now
has more slack than a shorter segment. In addition, the segment deadlines are earlier than the
ones that would be imposed by the end-end DT deadline. However, because of these earlier
deadlines, it is also possible that some DTs are forced to become infeasible earlier than before.
However, when there are more competing task segments on the same node, making one of the
end-end DT infeasible may increase the likelihood of end–end completion of the remaining task
segments. This results in improved performance for the heuristic schedulers, as shown in the
example below.

In Figure 2-5 there are three distributed tasks, DT1, DT2, and DT3. Let us assume the local UA
schedulers use LBESA and all tasks have equal utilities. The top segment describes the DT
properties. The left top segment describes the segment properties with their end-end deadline and
the right top segment describes the segments properties with after deadline decomposition. In the
left bottom segment, we show the segment ordering on the 3 nodes, N1, N2, and N3 without
deadline decomposition. The right bottom segment shows the ordering with deadline
decomposition. The first segment of all three tasks executes on Node 1. However, all 3 segments
are not feasible together with deadline decomposition since all have earlier segment deadlines.
Hence, DT2 becomes infeasible at an earlier stage, since it is the lowest PUD task, due to its high
cost. This allows DT3 to complete and accrue higher utility to the system. Without deadline
decomposition, local scheduler on N1 constructs a feasible schedule of all the 3 segments. It
delays DT3 because of its longer end-end deadline. At N2, DT1 is preempted to accommodate a
higher PUD task DT3. Eventually on N3, DT3, a higher PUD task takes priority over DT2, since
it has partially completed before DT3 arrives. Overall, only 1 of the 3 tasks complete without
deadline decomposition. With deadline decomposition, the longer task DT3 is eliminated, earlier allowing for end-end completion of 2 of the 3 tasks.

![Figure 2-5: Deadline Decomposition aids UA schedulers](image)

### 2.3.1.2 TUF SCALING

TUF scaling decomposes the end-end TUF by scaling the sub-TUFs by a factor. This factor depends on the height of the local tasks’ TUF. By doing so, we scale the utility of the distributed segments and improve their chances of being scheduled at the local nodes. In Figure 2-6, we show the TUFs of the local task and the DT. The DT has 3 segments. Before TUF scaling, the DT has a STEP-TUF with a height of 20 and the local task has a STEP-TUF with a height of 15. After applying the TUF scaling the height of the DT is scaled by a factor relative to the height of the local tasks’ height. This factor is a constant determined based on the range of the utility values considered for local tasks. If we assume the factor to be 10 times the ratio of height of the local task’s TUF to that of the DTs, it amounts to 7.5. The final utility value for the DT is 7.5 times the initial height, i.e. 150.
Figure 2-6: STEPS, TUF Scaling technique
Chapter 3  Optimality of Utility Accrual Schedulers

UA schedulers use heuristics based on one or more task characteristics to schedule tasks in soft real-time environments. Such schedulers aim at meeting some defined optimality criterion in a predictable manner. Minimizing the number of late tasks, minimizing the maximum lateness, minimizing the sum of finishing times are some of the commonly used optimality criteria. In the case of UA schedulers, the optimality criterion is to maximize the total accrued utility. Another goal of such schedulers is to realize this objective in polynomial time to make them suitable for use in on-line environments. Heuristics assist the UA schedulers to complete in polynomial-time. Thus, heuristic schedulers do not exhaustively search all possible task orderings to fulfill the optimality criterion. They do not guarantee optimal schedules.

Nevertheless, they find applicability in a wide range of critical dynamic environments such as military systems [Li et al., 2004], robotics, telecommunications, industrial automation, and virtual reality. These systems model the timing constraints of various tasks using TUFs. These systems are not standalone but have many sub-systems that operate in a distributed environment. Tasks originating on one sub-system cross system boundaries and execute on one or more sub-systems to achieve the desired output. UA schedulers make scheduling decisions in these sub-systems, as per the heuristics, using the locally available task information. We know that the local heuristic decisions are not guaranteed to be optimal. The cumulative effect of such sub-optimal results in each of the local sub-systems may worsen the end-end system-wide (global) optimality of UA schedulers in distributed environments. However, to our knowledge, there has being no effort so far to formally evaluate the degree of optimality of the UA schedulers in single-node and distributed environments. Our research is an effort to find convincing inferences on the optimality of two UA schedulers, DASA-ND and LBESA for different workloads, which will eventually aid real world applications that use UA scheduling to choose the most favorable system and task parameters.
We next discuss the scenarios we intend to explore as part of the experimental analysis that work against the heuristic decisions of DASA-ND and LBESA. As explained before, DASA-ND uses the PUD heuristic to schedule tasks on uniprocessors.

Consider a simple task model, where tasks do not share any resources among them. At system underload, it is possible to order all task segments in the order of their increasing deadlines and achieve optimal results. During system overload, it is not possible to construct a feasible schedule of all the task segments. DASA-ND examines tasks in the decreasing order of their PUD and constructs the feasible set in the deadline order, by adding as many ready tasks as possible while still ensuring the feasibility of the task ordering. If the higher PUD tasks happen to have longer deadlines than the lower PUD task in the feasible set, lower PUD tasks take priority over the higher PUD tasks in the scheduling order. Delaying the high PUD tasks is a risk DASA-ND takes since it is greedy to complete more tasks in deadline order. Depending on the task properties, this risk may either aid DASA-ND to reach optimal results or adversely affect the outcome. Since DASA-ND is a dynamic on-line scheduler, it is unaware of the future task arrivals. If the future arrivals have higher PUD and shorter deadline than the delayed task, they take precedence over the delayed task. If the delayed task takes longer time to complete, there is a chance of this task becoming eventually infeasible. Thus, DASA-ND loses a high utility task because of an initial greed to complete more tasks. Other combinations of task properties can defeat the heuristic scheduler as well. For instance, if the high PUD of a task is a result of low cost rather than high utility, a low utility, high PUD task may take precedence over high utility task. A mere ratio might be a misleading heuristic in some situations.

In a distributed system, other factors also work against the heuristic-based local scheduling decisions. One such factor is the load distribution on multiple nodes. A single bottleneck node can nullify the optimal decisions made on other lightly loaded nodes. Local schedulers are neither aware of the end-end path taken by the local segments, nor the number of successive segments that need to finish, for a distributed task to accrue utility. Using the end-end timing constraints for sub segments of the DT has shown dismal results. Thus, decomposition techniques have been used to slice the end-end slack proportionally among the segments and scale the height of the segments’ TUF relative to the sliced deadlines. This creates earlier
deadlines for segments and forces the local schedulers to complete shorter segments earlier, providing more room for the longer ones. However, in some situations earlier deadlines for the competing segments on a local node can make them infeasible at an early stage. Hence, by prematurely eliminating a few tasks, the nodes may idle at the later stages, resulting in overall lower utility to the system. Thus, we illustrate how UA schedulers produce sub-optimal results in different scenarios. However, at this stage we cannot prove the degree or probability to which such scenarios occur in typical workloads. Our primary goal is to implement an off-line scheduling algorithm that yields the optimal schedules for any given task set and use the results as a reference for analyzing the optimality of DASA-ND and LBESA. Subsequently we create comprehensive set of workloads using systematic and unbiased methods and evaluate the degree of optimality of DASA-ND at different system loads for different combinations of task parameters.

**Research Goals:** We simulate DASA-ND and LBESA along with the decomposition techniques to analyze their degree of optimality for different workloads. We intend to answer the following questions as part of the experimental analysis.

- What is the probability that DASA-ND and LBESA are optimal for different utility patterns on a single-node?
- Is DASA-ND better than LBESA for all types of workloads? If not, which workloads favor LBESA more than DASA-ND and vice-versa?
- What is the probability that DASA-ND and LBESA are optimal for workloads with equal task utilities in a distributed system?
- Does a specific combination of task deadlines and utility pattern show a higher probability of optimal results for DASA-ND and LBESA?
- How do different load distribution patterns affect the optimality of DASA-ND and LBESA in a distributed system?
- Do both DASA-ND and LBESA show similar optimality for both periodic and aperiodic tasks?
- What is the likelihood that DASA-ND and LBESA along with deadline slicing and TUF scaling are optimal for distributed workloads?
• Are they any task utility patterns that show optimal results in both the single-node and the distributed UA scheduling environments?
Chapter 4  Scheduling Algorithm for Optimal Schedules

This chapter explains the design rationale behind the scheduling algorithm implemented to evaluate the optimality of the UA schedulers both in single-node and distributed systems. It also discusses the high-level details of the algorithm and quantitatively analyzes the complexity of the scheduling algorithm.

4.1 Design Rationale for the Optimal Scheduler

Our research goal is to implement a scheduling algorithm that can generate optimal task orderings in UA scheduling environments for both single-node and distributed systems. Our implementation exhaustively searches through all possible feasible orderings of task segments and selects the one that yields the best overall utility to the system. Hence, it produces the optimal task schedules for any given task set. This is an off-line algorithm; it operates on a predefined set of tasks before the task activation, unlike the on-line UA schedulers that take scheduling decisions at run-time. It supports a preemptive task model. It is a static algorithm since the task parameters remain fixed during the scheduling process. Clearly, the algorithm that outputs the optimal schedules has super-exponential-time complexity due to the exhaustive search and it becomes computationally intractable\(^4\) to calculate the optimal schedules task sets beyond a certain size. The optimal algorithm only serves as a reference to assess the performance of real schedulers for different system loads and task properties for small task sets.

4.2 Optimal Scheduler: A High-level description

This section provides a high-level description of the scheduling algorithm used to generate the optimal schedules in UA scheduling environments. The optimal scheduling algorithm exhaustively explores the search space for a given task set. This search space can be represented

\(^4\) a computational problem for which the best possible algorithm has a super-exponential run-time.
as a tree diagram as shown in Figure 4-1. The root of the tree represents an empty schedule of tasks. Any intermediate vertex in the tree represents a partial or incomplete schedule for a given task set. A leaf vertex in the search tree represents a complete schedule. A complete schedule might be either an optimal or a sub-optimal ordering based on the total utility accrued. The goal is to select that ordering of task segments that produces the maximum utility to the system.

We use a branch and bound algorithmic technique to find the best solution. Each branch explores a unique task schedule. We keep the best schedule found so far and compare subsequent schedules with the current best schedule. If the subsequent schedule does not perform better than the current best, it is discarded. If there are multiple best schedules, only one is kept. This technique is implemented as a backtracking algorithm.

Backtracking is used to retract from one search path and branch into another using a depth first search (DFS) of the tree. The search tree is constructed dynamically by adding a new feasible task to the partial schedule in the search path at each step of the DFS. A search path is discarded if it completes with a sub-optimal result. The total depth of the search path and the branching factor of the tree at each level are determined by the eligibility and feasibility of the ready task segments for a given task set. A task segment is eligible to be scheduled if its predecessor task segment is already part of the partial schedule. A task segment is feasible to be scheduled if it can complete its execution on its respective node before its deadline, from the time it is scheduled. The algorithm determines the task segments that become available in future on a particular node and schedules the current task only to that point. Hence, a task segment can be scheduled only partially in one of the search paths giving preference to the other ready tasks. In another search path, this task will be given the preference compared to the other ready tasks and scheduled completely. Thus, determining the tree branching equivalent to the scheduling events in on-line schedulers occurs at both the arrival of new task and the completion of the current task. This reflects the fact that the optimal scheduler supports a preemptive task model. A search path terminates when there are no more eligible and feasible task segments to be scheduled. An optimized bounding technique would prune the tree if it could be determined that all remaining feasible and eligible tasks cannot yield better utility than the best, found so far. At the end of each search path i.e. before backtracking to the next search path in the DFS order, the optimality
of the current completed schedule is determined. If the current task ordering yields a better utility than the best so far, this schedule replaces the best ordering so far. The output of this algorithm is one of the task orderings that yields the best overall utility to the system.

![Search Tree Representation](image)

**Figure 4-1 : Search Tree Representation**

### 4.3 Optimal Scheduler: Pseudocode

This section provides details on the implementation modules of the optimal scheduling algorithm. **Algorithm 1** provides an overview of the optimal algorithm. The input to the algorithm is a task set of n segments mapped to m nodes. The output is an optimal task ordering. **Module 1** provides the details of the recursive DFS search of the tree representing the task schedules. **Module 2** explains the algorithm details for determining the tree branching at each level of the DFS search. **Module 3** explains how the eligibility of task segments is modified after the addition
of the task to partial schedule in the current search. The “backtrack search path” module explains the steps in branching to a new search path after the end of the current search path.

<table>
<thead>
<tr>
<th>ALGORITHM : OVERVIEW OF OPTIMAL SCHEDULING ALGORITHM</th>
</tr>
</thead>
<tbody>
<tr>
<td>INPUT : TASK SET WITH N TASK SEGMENTS, M NODES</td>
</tr>
<tr>
<td>OUTPUT : TASK ORDERING WITH MAXIMUM UTILITY</td>
</tr>
</tbody>
</table>

1. TaskSet = READ INPUT FILE FOR TASK PROPERTIES
2. BestUtility = 0; BestSchedule = null; currentNodeTime[1..m] = 0;
3. **DFS SEARCH** /* explores all possible search paths from the task set*/
4. PRINT OPTIMAL SCHEDULE

Algorithm 1: Overview of Optimal Scheduling Algorithm

<table>
<thead>
<tr>
<th>MODULE : DFS SEARCH</th>
</tr>
</thead>
<tbody>
<tr>
<td>INPUT : EMPTY TASK SCHEDULE</td>
</tr>
<tr>
<td>OUTPUT : SEARCH PATH THAT REPRESENTS OPTIMAL SCHEDULE</td>
</tr>
</tbody>
</table>

1. BreadthCount = **CONSTRUCT TREE BRANCING** AT LEVEL I /* Each vertex is a partial schedule*/
2. IF BreadthCount > 0
3. For each BranchingVertex at level i,
   3.1 ADD BranchingVertex to CurrentSchedule
   3.2 **PROCESS CurrentSearchPath**
   3.3 DFS SEARCH /* recursive search*/
   3.4 BACKTRACK from CurrentSearchPath
4. ELSE /* end of current search path, completed schedule*/
   4.1 Current Utility = CALCULATE UTILITY OF COMPLETE SCHEDULE
   4.2 IF CurrentUtility > BestUtility
      4.2.1 BestUtility = CurrentUtility.
      4.2.2 Best Schedule = Current Schedule.
   4.3 ELSE
      4.3.1 CurrentSchedule = null
      4.3.2 RETURN FROM FUNCTION

Module 1 : DFS Search
**MODULE : CONSTRUCT TREE BRANCHING**

**INPUT**: TASKSET, N TASK SEGMENTS, M NODES  
**OUTPUT**: TREE BRANCHING FACTOR

1. IF FEASIBLE AND ELIGIBLE TASK SEGMENTS ARE AVAILABLE FOR ANY NODE  
   1.1 For each feasible and eligible task segment in TaskSet  
      1.1.1 preemptTime = DETERMINE PREEMPT TIME  
      1.1.2 IF preemptTime > 0  
         1.1.2.1 Insert task into tree branch with preemptTime &  
         1.1.2.2 Insert task again into tree branch with totalExecTime  
         1.1.2.3 BreadthCount++  
      1.1.3 Insert task into tree branch only once with totalExecTime  
      1.1.4 BreadthCount++  
   2. ELSE  
      2.1 RETURN 0 /* end of current search path*/  
3. RETURN BreadthCount

---

**MODULE : PROCESS SEARCH PATH**

**INPUT**: TASK SEG on NODE M  
**OUTPUT**: MODIFY ELIGIBILITY of Successive TASKSEG in CurrentSearchPath

1. ADD TASKSEG TO CurrentSearchPath  
2. IF totalRemainingTime of TASK I = 0  
   2.1 make TASK I, SEG J+1 ELIGIBLE  
   2.2 make TASK I, SEG J INELIGIBLE  
3. ELSE  
   3.1 Do Nothing  
4. Increment CurrentNodeTime[m] by totalExecTime of TASK I SEG J

---

**MODULE : BACKTRACK SEARCH PATH**

**INPUT**: TASKSEG on NODE M  
**OUTPUT**: MODIFY ELIGIBILITY of TASK SEG in Successive SEARCH PATH

1. DELETE TASKSEG FROM CurrentSearchPath  
2. IF ! SEG J-1 exits in CurrentSchedule  
   2.1 make TASK I, SEG J ELIGIBLE  
3. Decrement CurrentNodeTime[m] by totalExecTime of TASK I SEG J

---

Module 2 : Construct Tree Branch  
Module 3 : Process and Backtrack Search Path
4.4 Complexity Analysis of Optimal Scheduler

This section discusses the run-time complexity of the optimal algorithm. The complexity of the exhaustive search is non-polynomial and grows super-exponentially as a function of the input task and system parameters. The total number of task segments in the task set determines the maximum depth of the search tree if no task segments are preempted i.e. every segment is run to completion at every point. However, with preemption, the maximum depth might exceed the total number of segments. The maximum branching factor of the search tree depends on a combination of task parameters. The total task segments that arrive at time unit zero decide the top most level branching factor. The branching factor for subsequent levels depends on the arrival rate, feasibility, and eligibility of the ready tasks. The upper bound on the number of leaves (or search paths decided by the branching at each level) is $n!$ where, $n$ is the number of task segments. Thus, $n!$ paths each of depth $> n$ (i.e. in preemptive task model) are searched in the worst case, i.e. when all the tasks can be accommodated in each of the search paths.

For aperiodic tasks with varying costs in the single-node case, the branching factor is higher at low loads than at higher loads. At low loads, more tasks are feasible and hence more search paths are explored. For periodic tasks, due to fixed arrivals and deterministic costs, even at high loads the average run-time of the optimal algorithm is lower than for aperiodic tasks. Figure 4-2 shows the plot of tasks vs the average run-time taken by the optimal algorithm for periodic tasks (workload description shown above the plot) at high loads. The time taken increases as the number of task segments increase. For task segments $> 11$, the time taken to complete grows and becomes computationally intractable. Figure 4-3 shows the plot of tasks vs the search nodes and search paths explored for the same periodic task set at high system loads (load = 1.5). Figure 4-4 & Figure 4-5 show the plots of tasks vs average run-time of optimal algorithm for aperiodic tasks (workload description shown above the plot) at low and high loads. As explained, at low loads, more search paths are explored and hence the average run-time is higher than at high loads for the same workload.
# OF TASKS vs RUN-TIME OF OPTIMAL ALGORITHM, SINGLE-NODE, PERIODIC TASKS

![Graph showing run-time vs number of tasks](image)

Figure 4-2: Tasks vs Run-time of optimal-algorithm, Periodic tasks, high load

# OF TASKS vs SEARCH NODES and SEARCH PATHS OF OPTIMAL ALGORITHM, SINGLE-NODE, PERIODIC TASKS

![Graph showing search nodes vs number of tasks](image)

Figure 4-3: Tasks vs # of Search Nodes/Search Paths for periodic tasks, high load

<table>
<thead>
<tr>
<th>#of tasks</th>
<th>Periods</th>
<th>Utility Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 to 11</td>
<td>[4, 4, 4, 4]</td>
<td>[100, 200, 300, 400, 500]</td>
</tr>
</tbody>
</table>
The average run-time of the optimal algorithm for distributed workloads not only depends on the number of task segments, load and arrival rate, but also on the total number of nodes and the segment to node mappings. Figure 4-6 shows a plot of task segments vs average run-time of optimal algorithm for aperiodic distributed workloads (workload description shown above the plot) different system loads. For fewer than 8 task segments, the run-time is almost zero. It increases drastically for higher number of task-segments. Figure 4-7 shows the plot of tasks vs search nodes for the optimal algorithm for the same periodic distributed workload.

<table>
<thead>
<tr>
<th>#of tasks</th>
<th>Arrival Rate</th>
<th>Utility Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>8 to 18</td>
<td>AR = 8.33</td>
<td>[100, 200, 300, 400, 500]</td>
</tr>
</tbody>
</table>

Figure 4-4 : Tasks vs Run-Time of optimal algorithm, aperiodic tasks, low load
Figure 4-5: Tasks vs Run-Time of optimal algorithm, aperiodic tasks, High load

<table>
<thead>
<tr>
<th>Period Values for tasks</th>
<th>Utility Values</th>
<th>Segments</th>
</tr>
</thead>
<tbody>
<tr>
<td>[4, 4, 4]</td>
<td>DTs: [100, 200, 300]</td>
<td>[2, 4, 2]</td>
</tr>
<tr>
<td>[4, 4, 4]</td>
<td>Local-tasks: [60, 60, 60]</td>
<td>Local-tasks: [1, 1, 1]</td>
</tr>
</tbody>
</table>

Figure 4-6: Task-segments vs Run-time of optimal algorithm, Distributed aperiodic tasks
Figure 4-7: Tasks vs Search nodes, nodes = 3 for periodic distributed workload

<table>
<thead>
<tr>
<th>Aperiodic</th>
<th>Utility Values</th>
<th>Segments</th>
</tr>
</thead>
<tbody>
<tr>
<td>$AR = 8.33$</td>
<td>$DTs: [100, 200, 300]$</td>
<td>$[2, 2, 3]$</td>
</tr>
<tr>
<td></td>
<td>$Local-tasks: [60, 60, 60]$</td>
<td>$[1, 1, 1]$</td>
</tr>
</tbody>
</table>
Figure 4-8: Tasks vs Tree Breadth for Distributed Aperiodic Tasks

Figure 4-9: Task vs Tree Depth for Distributed Aperiodic Tasks
Figure 4-8 shows the plot of task segments vs the max breadth for the search tree for one of the aperiodic distributed tasks sets (workload description shown above the plot) for different nodes. Figure 4-9 shows the corresponding plot of task segments vs the max depth for the search tree. We observe that for the same task set with more system nodes, the max breadth of the search tree decreases. However, as the number of task segments increases, irrespective of the nodes, the breadth and depth of the tree increase at the same rate.
Chapter 5 Experimental Methods

This chapter explains the workload generation methods, evaluation metrics, and the techniques used in computing and evaluating the optimality of heuristic-based UA schedulers, DASA-ND and LBESA.

5.1 Design Methodology

In order to assess the optimality of UA schedulers; we need representative and unbiased workloads, a systematic approach to generate accurate results and finally relevant and quantifiable measures to evaluate the results. The workload generation methods and evaluation techniques must ensure repeatability. We next discuss the workload convention and methods used to generate unbiased task sets for both single-node and distributed system. We elaborate on the statistical procedures used to generate results and define the metrics relevant for our experiments.

5.1.1 Workload Description

Workloads in our experiments are represented using a systematic notation. The notation specifies both the system and task parameters and is used in the subsequent sections and chapters. Each workload is a vector of size 10. Parameters starting with “S” denote a system parameter and that starting with “T” denote a task parameter. We also define a range of valid values for each parameter in our context.

5.1.1.1 SYSTEM PARAMETERS

The first parameter “Sn” is a scalar quantity and represents the number of system nodes. We experiment only with one node or 3 nodes. Hence, a value of one denotes that tasks are activated only on a single node and a value of 3 represents a distributed system of nodes. The second parameter “St” is also a scalar and represents the total number of tasks activated in the system. We specify an upper limit of 8 for St. The third parameter “SSL” is a vector of size “Sn” representing the system load on each node in the system. We denote this as the system load
vector. In the distributed case, this vector is indicative of the load distribution on each node. It shows either a balanced or unbalanced load distribution among the system nodes. The fourth parameter “$S_{LL}$” is a scalar representing the local-load on each node. This is only relevant in the distributed case. In our experiments, we assume that all nodes in the distributed system have the same local load. The fifth parameter denoted by “$S_{\Lambda}$” is a scalar representing the scheduler used in the experiment. It can either represent the DASA-ND, LBESA or the optimal scheduler. The sixth parameter denoted by “$S_d$” is a scalar representing the decomposition method used by the algorithm. It can represent either the SLEQF or the STEPS decomposition technique. The optimal scheduler has a system-wide view of the task characteristics and hence does not require any decomposition methods such as deadline slicing or TUF scaling. Node-local DASA-ND and LBESA schedulers use these decomposition techniques.

### 5.1.1.2 TASK PARAMETERS

The seventh parameter denoted by “$T_{period}$” is a vector of size “$S_t$” representing the arrival rate of each task defined in the system. For periodic tasks, each element in the vector can take any value in the range [1-4]. For aperiodic tasks, this is a scalar value representing the arrival rate modeled using the Poisson distribution function. The eighth parameter denoted by “$T_{utility}$” is again a vector of size “$S_t$” and represents the utility values for each of the task. Since our experiments are limited to using step-downward TUFs with positive utility, the elements in the vector can take any value $> 0$. The ninth parameter denoted by “$T_{cost}$” is a vector representing the expected execution costs of each task. The vector data can take any value $> 0$. We impose another constraint that the “$T_{cost}$” values are less than or equal to the corresponding “$T_{period}$” values for each task in the periodic task model. The actual interpretation of the value depends on the execution-cost model used. Tasks can either have deterministic or stochastic execution costs for different instances of their activation. In the case of stochastic execution model, the vector values denote the expected value of the task execution-cost. The last parameter denoted by “$T_{segs}$” is a vector representing the number of segments for each task. The maximum value possible for this parameter is 4. A value of one denotes local tasks that do not traverse multiple node boundaries.
5.1.2 Workload Selection

In order to evaluate the optimality of DASA-ND and LBESA, we need to generate workloads that are representative of the system and task properties of the UA scheduling environments. We consider all parameters that affect the performance of schedulers in a real-time environment. Since it is beyond the scope of this work to consider real workloads in both the single-node and distributed case, we synthesize certain combinations of system and task parameters that cover a wide range of variations in these values. We analyze workloads with a defined relationship between the task parameters. For instance, utility values for a given set of tasks form one of the task parameters. We not only assess how different utility patterns influence the optimality of DASA-ND and LBESA, but also determine how a task set with a defined relationship between task deadlines and utility pattern together influence the optimality of the heuristic schedulers.

Of the 10 different workload parameters identified, we need to identify the most important parameters, whose variation heavily influences the outcome. Hence, we carefully choose the fixed parameters and the varying factor for different sets of experiments.

In the case of the single-node scheduling experiments, system load acts as an important factor. We conduct experiments by varying the load at defined values for each workload. We execute 100 simulation runs for each of the system load value. In each of the 100 runs, we vary the task costs and keep the remaining workload parameters fixed. Across each workload, we vary utility patterns, the cost model, and finally the task model. We model three different types of utility patterns, discussed in detail in Chapter 6. We experiment with two kinds of task cost models, deterministic and stochastic. Depending on the pattern of task arrival, we experiment with periodic and aperiodic task arrivals. In the case of the distributed system, we not only vary the system load, but also the segment to node mappings in each of the 100 runs for every workload. Across the workloads, we vary the number of task divisions, utility pattern, segment costs, load distribution, and the task model.
5.1.3 Workload Generation

5.1.3.1 SINGLE-NODE

We consider system load as a factor and simulate experiments at varying loads. We use statistical techniques, i.e. random sampling and average the results for 100 simulation runs at different system load for each of the workload selected. The task properties such as number of tasks, tasks’ cost, arrival rate, or periodicity, specified in the workload directly determine the overall system load. For a periodic task model with \( n \) tasks, system load is given by,

\[
\sum_{i=1}^{n} (T_{\text{cost}}[i] + T_{\text{period}}[i])
\]

where, \( T_{\text{cost}} \) represents cost of each task, & \( T_{\text{period}} \) represents their corresponding periods. Simulation of system overload is done by either adding more tasks to the system or increasing the execution cost of a fixed set of tasks relative to the load factor. We choose the latter and collect results for a particular workload at both low and high system loads.

For a particular system load, we are required to generate \( S_t \) random values, corresponding to the cost of the individual local tasks in each of the 100 simulation runs. Task periods are fixed and are a multiple of 2. One of the approaches to generate the vector \( T_{\text{cost}} \) is to derive each value from a distribution in the range \([0, T_{\text{period}}]\) and thereby calculate the system load. However, we want to generate many combinations of the vector values for a given system load. UUniFast [Bini and Buttazzo, 2005] is an \( O(n) \) algorithm that samples these combinations such that the sum of these “\( S_t \)” random values is equal to the system load on a single-node for periodic tasks. The “\( S_t \)” random values are derived from a uniform distribution in the range \([0, 1]\). The exact psuedocode for UUniFast is provided in [Bini and Buttazzo, 2005]. Task deadlines are equal to their corresponding task periods. For aperiodic task arrival, the arrival-rate \( \lambda \) is modeled as a random variable from the Poisson distribution function with an average value \( \lambda_{\text{avg}} = (S_t * T_{\text{cost}}) / S_{\text{SSL}} \). The slack is computed as a random variable from a uniform distribution within an interval of \([0.5-2.5]\) and task deadline is the sum of task cost and the slack.

Task utilities follow a defined pattern. It is practically infeasible to explore all combinations of utilities; we choose to experiment only with the following three sets of utility patterns. The first case is the simplest, where the utilities of all the tasks are the same. The second case uses a
utility pattern where the ratio between the consecutive task utilities is constant. The task utilities follow a geometric sequence. Two sub-cases, one for low common ratio (CR) (e.g. [2, 4, 8, 16, 32]) and another for higher common ratio (e.g. [1, 8, 64, 512, 4096]) are simulated. The third case uses a utility pattern where the difference between the two consecutive task utility values is constant. The task utilities follow an arithmetic sequence. Again, the value of the common difference is a factor. We chose to experiment with two sub cases, one for a low common difference (CD) and other for high common difference. For instance, one set has a utility vector of [2, 3, 4, 5, 6] for a set of 5 tasks with a low constant value. The other set has a utility vector of [100, 200, 300, 400, 500] with a higher constant.

5.1.3.2 DISTRIBUTED SYSTEM

We do not fix the system load across the 100 runs, but rather vary it in the set of experiments for distributed workloads. Hence, we derive the vector $T_{cost}$ from a lognormal distribution in the range $[0, T_{period}]$ and thereby calculate the total load. For a distributed system, load distribution is an important aspect. In some workloads, we control the load on each node, by assigning segment to nodes based on a random value. This random variable is derived from a uniform probability distribution in the range $[0, S_n]$. Task utilities follow a defined pattern as in the single-node experiments.

5.2 Evaluation Procedure

Once the workloads are selected, the next step is to design procedures to generate accurate results. In our context, the results represent the task schedules on one or more nodes in the system. A simulation model [Wu et al., 2005] built using OMNET++ [A.Varga, 2004] computes the corresponding heuristic-based UA schedules for DASA-ND and LBESA. OMNET++ is a discrete event simulation environment used to model the DASA-ND, LBESA and the corresponding deadline and TUF decomposition techniques, SLEQF and STEPS [Wu et al., 2005]. The OMNET++ simulation environment specifies mechanisms for seed selection and supports different algorithms for random number generation used in different probability distribution functions. We did modifications to the existing OMNET++ simulation model for the UA schedulers to add workload generation methods described above for both the single-node
and distributed environments. The off-line scheduling model described in Chapter 4 computes the optimal (best possible) schedules for defined workloads.

As discussed, we simulate 100 runs for each variation of the workload parameters for DASA-ND, LBESA and optimal scheduler. Each individual observation yields representative results for the corresponding workload. We calculate the average and variance from these 100 observations to infer how a given task set influences the optimality of DASA-ND and LBESA. Our statistical procedure use random sampling technique and probability distribution functions to model different random variables representative of the task and system parameters in the real-time environments.

### 5.3 Evaluation Metrics

Metrics refers to the criterion used to measure and hence compare the performance (optimality) of a particular system with another. In our context, “system” refers to the UA schedulers. Metrics are defined as a function of one or more of the system task properties. We define and discuss the motivation behind the use of the three important metrics in our experimental analysis.

#### 5.3.1.1 Deadline Satisfaction Ratio

Most often, the evaluation metrics for soft real-time schedulers use the deadline of the task. The Deadline Satisfaction Ratio (DSR) is defined as the ratio of task deadlines met to the total number of triggered tasks in the system [Li et al., 2004]. If the objective of the best-effort schedulers is to minimize the total number of missed deadlines, this metric is used.

In the case of the UA schedulers, the optimality criterion is to maximize the total task utility gained. Hence, a metric defined as a function of the total task utility seems the apt measure to compare their optimality.
5.3.1.2 ACCRUED UTILITY RATIO

The Accrued Utility Ratio (AUR) is defined as the ratio of utility accrued to the total maximum possible utility from all the triggered tasks [Li et al., 2004] in the system.

The results from the 100 random samples provide the average AUR and DSR values for each of the workloads. Since the sample mean of the AUR values might not completely characterize the optimality of DASA-ND\(^5\) and LBESA at different loads, we also consider histograms\(^6\) and the cumulative distribution function (c.d.f) of the delta AUR sample values at a particular system load to describe their optimality. Delta AUR sample values represent the difference between AUR values of optimal and heuristic schedulers. Any point on the c.d.f curve represents the proportion of the samples that have delta AUR less than or equal to the value on the horizontal axis. Histograms show, how the density of these delta AUR values is distributed, or what proportion of the sample observations lie within a given range.

\(^5\) DASA hereafter refers to DASA-ND, i.e. DASA without resource dependencies

\(^6\) Since we have used only 100 samples, the frequency represented on the y-axis of the Histograms can be equated to the probability representation in the Probability Density Function (P.D.F). Thus, a frequency of 60 in the histogram is equivalent to a 0.6 probability on the P.D.F. We refer to the histogram as the probability density function in the remaining sections.
Chapter 6  Results and Optimality Analysis: Single-Node

Accurate experimental analysis complements a well-modeled evaluation design. Chapter 6 and Chapter 7 provide a detailed experimental analysis of the degree of optimality in the UA schedulers, DASA-ND and LBESA, backed by the experimental results obtained from the actual simulation runs. Representative workloads are generated using different system and task parameters. In our context, the experimental results represent the task schedules generated by the scheduling algorithms. These task schedules are measured using the evaluation metrics defined in the chapter 5.

One of the goals of our experimental analysis is to evaluate the performance (optimality) of DASA-ND and LBESA in a single node environment with local tasks. Local tasks do not traverse multiple node-boundaries. They finish on the same node they originate. The task model simplifies in such a scenario. Tasks do not have segments and thus decomposing task properties such as deadline and utility does not apply to such scenarios. Hence, the Tseg and Sd parameters described in Chapter 5 are not applicable to these workloads.

As described in chapter 5, every experiment is composed of 100 simulation runs each executed for a predefined time to generate 100 different cost samples. This time is equal to the hyper-period of the task set for periodic workloads. Of the ten parameters that define the workload, only one of them varies across the simulation runs. The parameter that varies is called as the “factor.” For instance, for a given fixed set tasks, nodes, utility and period values, computation times varies across the 100 simulation runs. Similarly for a given workload combination, system load acts as a factor.
6.1 Research Questions for Single-Node Workloads

Our experimental analysis aims to answer the degree of optimality of DASA-ND and LBESA for different workloads in single-node UA scheduling environments. We provide answers to the following research questions on a single-node.

- **What is the probability that DASA-ND and LBESA are optimal for different task models and cost models on single-node?**
  
  Primarily, we experiment with the periodic and aperiodic task models. We also consider how deterministic and stochastic costs affect the degree of optimality of DASA-ND and LBESA in periodic workloads.

- **What is the probability that DASA-ND and LBESA are optimal for different workloads at higher system loads?**
  
  The role of heuristics in maximizing the total system utility accrued is more evident at system overloads than system underload. Therefore, it is necessary to analyze the optimality of UA schedulers not only when the system operates below its capacity and to its full capacity but also beyond its capacity. Hence, our experiments consider system load as an important factor in generating the workload parameters.

- **What is the probability that DASA-ND and LBESA are optimal for different utility patterns on a single-node?**
  
  The heuristics used in DASA-ND and LBESA depend on the task utilities. Hence, we create workloads that explore different utility patterns explained in Chapter 5.1.3.

- **Does a specific combination of task deadlines and utility pattern show a higher probability of optimal results for DASA-ND and LBESA?**
We also know that both DASA-ND and LBESA construct deadline ordered feasible sets to decide the total ordering of the ready tasks. Hence, we determine the combined effect of specific deadline and utility relationships among the tasks.

- **Is DASA-ND better than LBESA for all types of workloads? If not, which workloads are far more congenial for LBESA than for DASA-ND and vice-versa?**

### 6.2 Periodic Task Model, Deterministic Cost Model

Our initial set of experiments use workloads with only three periodic tasks. We know that both DASA-ND and LBESA schedulers use heuristics that are a function of task utility and execution cost. Task deadlines also influence the order in which tasks are scheduled by these schedulers. In our experiments, we assume that the task deadlines are equal to their periods. Our aim is to create workloads that will allow us to determine how different task utilities and task deadlines (periods) together influence the optimality of both the schedulers. We have not included any resource dependencies for DASA nor experimented with non-rectangular TUFs for LBESA at this point.

#### 6.2.1 EQUAL TASK UTILITIES

Table 6-1 summarizes the workload properties used in the first set of experiments. We perform the simulation experiments for both DASA-ND and LBESA. 100 independent simulation runs per workload explore different combinations of task execution times and slack. The periods and utilities of the tasks are fixed and do not vary in the simulation runs. Workload W1 and W2 has three tasks with equal utilities. W1 determines the effect of equal utility tasks on the optimality of DASA-ND and LBESA. It is sufficient to measure the DSR of DASA-ND/LBESA and optimal for such workloads. If both the optimal and UA schedulers have the same average DSR, it indicates that both meet the same of task deadlines. If the DSRs of both the heuristic and
optimal are the same, it implies that the AUR’s will be equal as well since all tasks have same utilities. In workload W2, we use equal task deadlines.

<table>
<thead>
<tr>
<th>W</th>
<th>S_t</th>
<th>T_period</th>
<th>DT</th>
<th>DV</th>
<th>T_utility</th>
<th>UT</th>
<th>T_cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>W1</td>
<td>3</td>
<td>[2, 4, 2]</td>
<td>UE D=P</td>
<td>2,4</td>
<td>[100, 100, 100]</td>
<td>E</td>
<td>D</td>
</tr>
<tr>
<td>W2</td>
<td>3</td>
<td>[4, 4, 4]</td>
<td>E D=P</td>
<td>4</td>
<td>[100, 100, 100]</td>
<td>E</td>
<td>D</td>
</tr>
<tr>
<td>W3</td>
<td>5</td>
<td>[4, 4, 4, 4]</td>
<td>E D=P</td>
<td>4</td>
<td>[100, 100, 100, 100]</td>
<td>E</td>
<td>D</td>
</tr>
<tr>
<td>W4</td>
<td>8</td>
<td>[4, 4, 4, 4, 4]</td>
<td>E D=P</td>
<td>4</td>
<td>[100, 100, 100, 100, 100, 100, 100, 100]</td>
<td>E</td>
<td>D</td>
</tr>
</tbody>
</table>

Table 6-1: Workloads with equal task utilities

Figure 6-1 & Figure 6-2 shows the graph of system load vs average DSR for workload W1 for DASA-ND and LBESA respectively. Each plot represents the average DSR obtained from a set of 100 independent simulations for different system loads. The plot represented in blue and diamond shaped icon [●] indicates the average and variance in DSR values for DASA-ND at different system loads. The plot represented in magenta and triangle shaped icon [▲] indicates the average and variance in DSR values for LBESA scheduler at different system loads. The plot represented in red line and square shaped icon [■] indicates the average and variance in DSR values for the best (optimal) case. The results show that both DASA-ND and LBESA produce optimal schedules for workload W1.
Figure 6-1: Load vs DSR, OPTIMAL & DASA-ND for W1, Equal Utilities only

Figure 6-2: Load vs DSR, OPTIMAL & LBESA for W1, Equal Utilities only

Figure 6-3 & Figure 6-4 shows the graph of system load vs average DSR for workload W2 for DASA-ND and LBESA respectively. Workload W2 does not have tasks with a mix of short and long deadlines as in W1, but, all the three tasks have same deadlines (periods). The results show that there is an overall decrease in the average DSR values obtained for system loads > 1.0. Fewer tasks complete due to equal task deadlines. However, the optimality of DASA-ND and LBESA is not affected. Both complete equal number of tasks as the optimal scheduler. As a result, DASA-ND and LBESA remain optimal by accumulating the theoretically maximum possible utility for task sets with equal utilities.
Next, we investigate whether increasing the total number of tasks influences the optimality of DASA-ND and LBESA. Hence, workloads W3 and W4 have five or more tasks with equal utilities and deadlines. Figure 6-5 & Figure 6-6 shows the graph of system load vs average DSR for workloads W3 and W4 respectively.
Figure 6-5: Load vs DSR for W3 with Equal Deadlines and Utilities & 5 tasks

Figure 6-6: Load vs DSR for W4 with Equal Deadlines and Utilities & 8 tasks

Investigating a few example runs, shows that even though the task schedules differ between the optimal and UA schedulers, both DASA-ND and LBESA meet the same number of deadlines as the optimal scheduler. Since, every task has the same utility, differences in task selections does not affect the optimality of DASA-ND and LBESA. Hence, we conclude that both DASA-ND & LBESA remain optimal for periodic workloads with equal utilities irrespective of task deadline values and total number of tasks in a single-node.
6.2.2 VARYING TASK UTILITIES

The next set of workloads use different utilities for 3 tasks and hence AUR is used as the metric to compare the results of heuristic schedulers with the optimal. For system loads, > 1.0 both DASA-ND and LBESA do not produce optimal schedules. Since there are only three tasks, there is not much variation in the results. This means that for a given set of utilities, UA schedulers may reject one, reject two, or accept all when the system load is high. Thus, AUR values fluctuate only between three levels. Results for such workloads show high standard deviation (variance) at higher system loads. Thus, the average AUR values are not the true indicators of the performance of schedulers. Hence, we here forth use workloads with more than 3 tasks having different utility patterns.

6.2.2.1 GEOMETRIC TASK UTILITIES

<table>
<thead>
<tr>
<th>W</th>
<th>Sₜ</th>
<th>T_period</th>
<th>DT</th>
<th>DV</th>
<th>T_utility</th>
<th>UT</th>
<th>T_cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>W5</td>
<td>5</td>
<td>[2, 4, 2, 4, 4]</td>
<td>UE (D=P)</td>
<td>2,4</td>
<td>[2, 4, 8, 16, 32]</td>
<td>G(2)</td>
<td>D</td>
</tr>
<tr>
<td>W6</td>
<td>5</td>
<td>[2, 4, 2, 4, 4]</td>
<td>UE (D=P)</td>
<td>2,4</td>
<td>[1, 8, 64, 512, 4096]</td>
<td>G(8)</td>
<td>D</td>
</tr>
<tr>
<td>W7</td>
<td>5</td>
<td>[2, 4, 2, 4, 4]</td>
<td>UE (D=P)</td>
<td>2,4</td>
<td>[1, 10, 100, 1000, 10000]</td>
<td>G(10)</td>
<td>D</td>
</tr>
</tbody>
</table>

D: Deterministic Cost, G: Geometric Sequence

Table 6-2: Workloads with Task Utilities in Geometric Sequence

Table 6-2 shows the details of the workloads that have task utilities in geometric sequence. We choose the task deadlines randomly. Thus, these workloads have a mix of short and long deadline tasks. Both the utility and deadline values remain fixed for each workload across the 100 simulation runs. Workload W5 has a lower CR of 2. W6 and W7 have higher constant ratios. Table 6-3 shows the average and standard deviation values for optimal, DASA-ND and LBESA for W5 at different loads. Figure 6-7 shows the graph of system load vs AUR for DASA-ND and LBESA in comparison with optimal schedules for W5. Each plot in the graph represents the

7 The primary y-axis shows the average values. The secondary y-axis for such plots represent the ratio of achieved AUR from DASA-ND/LBESA to the optimal AUR.
average AUR obtained from a set of 100 independent simulations for different system loads. The plot represented by the blue line and the filled ellipse shaped icon [●] indicates the average and variance in AUR values for DASA-ND at different system loads. The plot represented by the magenta line and triangle (filled) shaped icon [▲] indicates the average and variance in AUR values for LBESA at different system loads. The plot represented by the red line and square (filled) shaped icon [☐] indicates the average and variance in AUR values for the optimal (best) schedules. We see that both DASA-ND and LBESA are optimal at system under-loads. This is expected since, DASA-ND and LBESA use EDF (earliest deadline first) ordering at underload. Moreover, periodic task set ensures that all task properties follow the constraint that \( \sum (\frac{Ci}{Ti}) < 1 \) [Buttazzo, 1997]. At higher system loads, both DASA-ND and LBESA degrade. At a system load of 1.5, optimal scheduler has an average AUR of 0.866, DASA-ND has around 0.837 and LBESA is little lower with 0.829. Figure 6-8 shows the AUR distribution of optimal, DASA-ND and LBESA for workload W5 across the 100 runs at a load of 1.5. The red square represents optimal AUR, blue circle represents DASA-ND, and green triangles represent LBESA. Overlapping AUR values imply that one or more of these schedulers produce the same result. We observe that LBESA shows more cases where the AUR obtained in different runs is < 0.75\(^8\). Even though DASA-ND is sub-optimal, in most cases the AUR for different runs remains > 0.75. We investigate the special cases where DASA-ND and LBESA fail and explain the reasons for their sub-optimal results.

<table>
<thead>
<tr>
<th>% LOAD</th>
<th>AVG AUR/OPTIMAL</th>
<th>AVG AUR-DASA-ND</th>
<th>AVG AUR-LBESA</th>
<th>STD DEV OPTIMAL</th>
<th>STD DEV DASA-ND</th>
<th>STD DEV LBESA</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>80</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>100</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>101</td>
<td>0.970278</td>
<td>0.954167</td>
<td>0.956111</td>
<td>0.007087</td>
<td>0.028845</td>
<td>0.022742</td>
</tr>
<tr>
<td>110</td>
<td>0.956666</td>
<td>0.941111</td>
<td>0.941111</td>
<td>0.018139</td>
<td>0.057524</td>
<td>0.057524</td>
</tr>
<tr>
<td>130</td>
<td>0.914444</td>
<td>0.885278</td>
<td>0.885556</td>
<td>0.055788</td>
<td>0.094645</td>
<td>0.094532</td>
</tr>
<tr>
<td>150</td>
<td>0.86691</td>
<td>0.837778</td>
<td>0.829722</td>
<td>0.084877</td>
<td>0.119996</td>
<td>0.130627</td>
</tr>
<tr>
<td>180</td>
<td>0.812778</td>
<td>0.791111</td>
<td>0.791667</td>
<td>0.106546</td>
<td>0.132747</td>
<td>0.133976</td>
</tr>
<tr>
<td>200</td>
<td>0.777778</td>
<td>0.753611</td>
<td>0.755278</td>
<td>0.120121</td>
<td>0.153708</td>
<td>0.151411</td>
</tr>
</tbody>
</table>

Table 6-3: Average AUR values for W5, Geometric Utility, CR=2

\(^8\) An AUR defined (as the ratio of obtained to the total utility) of 1.0 means that UA schedulers accumulate the expected utility
There are 2 other plots in the same graph, one represented in blue with unfilled ellipse icon [ ] for DASA-ND and other represented in magenta with unfilled triangle [ ] for LBESA. These plots represent how DASA-ND and LBESA perform relative to the optimal as the system load increases. A slope of zero for these plots indicates that optimality of DASA-ND and LBESA does not vary with system load. A positive slope (increasing line) is a good sign indicating that these schedulers are closer to optimal for higher loads. A decreasing line (with negative slope) shows that these schedulers deviate further from optimal schedules with increasing load. Ideally, we expect to have a zero slope. However, as shown in the graph below, both DASA-NDs and LBESA’s degree of optimality fluctuates at different loads. Particularly, LBESA performs badly at a load of 1.5 and improves at 1.8 and 2.0. We attribute this to the fact that, for a few task cost and deadline combinations LBESA is worse than the rest at a load of 1.5. On the other hand, DASA-ND is more consistent in its performance at different high loads.
Figure 6-8: AUR distribution of OPTIMAL, DASA-ND and LBESA in W5 (Table 6-2) for 100 runs

**Scenario 1:**

Consider an example scenario shown in Figure 6-11 for workload W5. The top segment describes the tasks’ properties. There are 7 periodic tasks within a hyper-period of 4. The figure indicates task arrival, utility value, execution cost and deadline. A horizontal rectangular bar represents each task, with a vertical green line indicating its arrival, and a vertical red line indicating its deadline. The green line is visible in the actual task orderings created by Optimal, DASA-ND and Lbesa schedulers. The shaded (colored) region inside the rectangular bar represents the worst case execution cost of a task. To the left of the horizontal bars, we also indicate the task number and its utility value. For instance, we represent task1 (T1) as shown below in Figure 6-9. It has a utility value of 2 represented by \([U = 2]\). It arrives at time 01:00 (HR: MIN). Its execution cost is 13 sec and its deadline is 1 minute from its arrival.

---

**Figure 6-9: Task Description**
The bottom segments indicate the task orderings for OPTIMAL, DASA-ND and LBESA for each of the 7 tasks. **Figure 6-10** shows a partial schedule representation. An empty horizontal bar for a task under DASA-ND indicates that DASA-ND does not schedule this task for this workload. A filled rectangle within an interval indicates that the task finished under that particular scheduler. A partially filled rectangle indicates that the task aborts in favor of other tasks or fails to complete within its deadline under that particular scheduler.

In **Figure 6-11**, we see that DASA-ND fails to schedule the highest utility task T5, thus accumulating a total utility ratio of only 0.5 as compared to OPTIMAL and LBESA’s 0.81. DASA-ND considers tasks in PUD order and constructs the feasible set in deadline order. For this example task set, DASA-ND creates a feasible set of [T4/4.0, T1/2.0, T5/4.0, T3/2.0]. It initially schedules tasks T1 and T3 that have shorter deadlines than T4 and T5, even though T4 and T5 have higher PUD values. At this point, if no future tasks arrived, all tasks in the feasible set would complete before their deadlines. However, at time [01:02], T6 and T7 arrive, and T5 is halfway through completion. T6 turns out to be a higher PUD task compared to T5 and thus take precedence over T5. This is a case where a longer deadline task is initially pushed further ahead in the task order in favor of shorter deadline tasks, and later loses to newly arriving higher PUD tasks. Eventually, even before T6 finishes, T5 aborts since it becomes infeasible. Thus, DASA-ND rejects the highest utility task. Even, LBESA considers tasks in deadline order, and thus schedules T3 and T1 from the feasible set of [T3/2.0, T1/2.0, T5/4.0,T4/1.0]. Interestingly,

---

9Feasible set is a vector of one or more tasks that complete within their deadline. Feasible set uses [X/Y] notation. X represents task number and Y represents the corresponding task deadline. A value of 4.0 for Y means 4 units of time.
LBESA does not specify any ordering in case of deadline ties between two tasks. T3 is scheduled before T1 and similarly T5 is scheduled before T4. This gives T5 more room to finish than in DASA-ND. Hence, the newly arriving task T6 and T7 have a lower PUD of 16 and 8 as compared to T5’s 32. Both T6 and T7 are eliminated in favor of T5, while constructing the feasible set. Hence, LBESA completes the highest utility task and accumulates a utility ratio of 0.81.

**Scenario 2:**
Consider another scenario as shown Figure 6-12. The top segment describes the task properties. There are 7 periodic tasks within a hyper-period of 4. Both the higher utility tasks (T4 and T5) have high costs and equal deadlines. T2 has the highest utility among the tasks in the feasible set. As per the heuristic, T2 is higher than T5 and T5 is marginally higher in PUD than T4. DASA-ND breaks deadline ties using PUD heuristic [Clark, 1990]. However, if the actual implementation does not use a stable sorting algorithm to order tasks by deadline, we cannot ensure that the highest PUD task in among the deadline tied tasks is scheduled first [Clark, 1990]. The OMNET++ DASA-ND implementation does not use a stable sorting algorithm. Hence, DASA-ND picks T5 in favor of T2 or T4, the higher in the deadline order. LBESA does not use any heuristic such as PUD to break deadline ties. LBESA can choose T2, T4, or T5. Here, T4 is scheduled first and this causes T5 (PUD = 18) to abort eventually, since it loses to T2 (PUD = 26) after the highest PUD task T7 (PUD = 82) finishes. It then becomes infeasible. In DASA-ND, on the other hand, orders, T5 is scheduled prior to T4. Even though, both are sub-optimal, DASA-ND performs better than LBESA in this case. If LBESA used PUD value to break deadline ties and order tasks, it would perform as well as DASA-ND in such scenarios.
LBESA is optimal with AUR of 0.81. DASA-ND is sub-optimal with a total AUR of 0.5.

Figure 6-11: Scenario 1 - DASA-ND produces sub-optimal results for W5, Geometric Utility, CR=2 (Table 6-2) at a Load of 1.5.
Both DASA-ND and LBESA are sub-optimal. DASA-ND fares better than LBESA with an AUR of 0.75

Figure 6-12: Scenario 2 - LBESA and DASA-ND produce sub-optimal results for W5, Geometric Utility, CR=2 at a Load of 1.5
Figure 6-13 shows the graph of c.d.f and histogram for differences (delta values) in AUR values of the optimal and DASA-ND/LBESA for the 100 simulation runs at a system load of 1.5. It shows a 0.77 probability that DASA-ND is optimal and a 0.72 probability that LBESA is optimal. The corresponding histogram shows that around 20 of the 100 cases, DASA-ND is sub-optimal with a difference in AUR (delta AUR) greater than 0.01 and less than 0.35. LBESA performs worse and has a higher frequency of cases that produce sub-optimal results in the same range.

Since, it is clear from the explanations in Scenario 1 and Scenario 2 that task costs, deadlines and utility values, together influence the optimality of heuristic schedulers; we experiment with another workload, with similar task utilities as W5, but with a defined relation between task deadlines and utility values. This workload has a mix of short, medium, and long tasks and hence more variation in the costs for tasks generated within the hyper-period. Instead of using random task periods, we clearly define a relationship between the task utilities and deadlines. W8 shown in Table 6-4 has 5 tasks, with shorter deadline tasks having low utility and longer deadline tasks having higher utility values. We explore all possible task costs combinations in the 100 simulation runs and compute the average AUR for each of the 3 schedulers for both high and low loads.
The results in Figure 6-14 for W8 show that there is not much deviation from the results obtained for W5. There are more tasks in this workload as compared to W5. As a result, there is an overall drop in actual AUR values of all schedulers. The likelihood of DASA-ND and LBESA remaining optimal remains the same, at 0.7 for DASA-ND and 0.72 for LBESA. We discuss a specific example scenario for this workload, where both DASA-ND and LBESA produce sub-optimal results.

**Scenario 3:**
Consider the scenario shown in Figure 6-16 for W8. Workload W8 has more number of tasks than in workload W5. Since the system load is 2.0, tasks generally have higher costs. Task 4 and Task 5 have high execution costs. Even, the optimal schedule is able to schedule only one of these two tasks and chooses to schedule T5, due its higher utility value. Both, DASA-ND and LBESA, abort the partially completed T5 favoring short tasks that have higher PUD. Thus, DASA-ND
initially delays longer deadline, higher PUD in favor or shorter deadline tasks in the feasible set. DASA-ND later aborts these tasks favoring other shorter, higher PUD tasks that arrive in future. Hence, for workloads with a mix of periodic tasks having longer deadlines & higher utility values along with tasks having shorter deadline & lower utility values does not favor the heuristic schedulers.

Further, we experiment with workloads that have higher constant ratios (W6 and W7). Figure 6-15 and Figure 6-17 show the graph of system load vs AUR for DASA-ND and LBESA in comparison with the optimal schedules for W6 & W7 (Table 6-2) respectively. Workload W6 shows higher sub-optimal results only at a system load of 1.5. For other loads, difference in AUR values of the UA schedulers and the optimal is still lower. Further investigation shows that for one particular combination of task costs and deadlines, both DASA-ND and LBESA fail to schedule the high utility task and thus result in a 0.7 delta AUR. This is shown in the histogram of DASA-ND and LBESA for W6 in Figure 6-18. The c.d.f for DASA-ND and LBESA shows a 0.89 probability that DASA-ND is exactly optimal and 0.79 probability that LBESA is exactly optimal. Thus, DASA-ND fares better than LBESA.

![Figure 6-15](image-url)
Both DASA-ND and LBESA are sub-optimal with 0.38 AUR compared to an AUR of 0.65 for the optimal.

Figure 6-16: Scenario 3 - Both DASA-ND and LBESA produce sub-optimal results for W8, Geometric Utility, CR=2 at Load of 2.0.
For workload, W7, both DASA-ND and LBESA are optimal. On an average, there is a less than 0.1 difference in the AUR’s of UA schedulers and optimal scheduler. Figure 6-19 shows the
corresponding C.D.F and HISTOGRAM at a system load of 1.5. It shows a 0.93 probability that DASA-ND is optimal and a 0.82 probability that LBESA is optimal. Interestingly, it also indicates that at a system load of 1.5 both have a 100% probability that they are optimal with the delta AUR being <= 0.01. This indicates that for major proportion of the individual samples in the 100 runs, DASA-ND and LBESA achieve the maximum possible utility. For a minor proportion of cases, they make decisions that result in 0.01 difference of utility ratio. This shows that DASA-ND and LBESA have a 100% chance of completing all the higher utility tasks, if not accommodating even the lower utility ones together.

As the constant ratio increases and there is wider variation in the task utility values, both DASA-ND and LBESA are closer to the optimal. Both DASA-ND and LBESA perform consistently with respect to the optimal with increasing system loads. If we closely examine the cases in W5, where DASA-ND and LBESA produce sub-optimal results, a clear pattern emerges. *Most often periodic tasks with longer deadline, high cost and higher utility tasks lose to short tasks that happen to have better PUD than it does. The high PUD value of short task is due to their short completion times rather than due to their higher utility value. Hence, low utility, but higher PUD tasks are favored more. This is no longer the case with workloads such as W6 and W7.*

Figure 6-19 : C.D.F and HISTOGRAM for W7, Geometric Utility, CR=10 at Load = 1.5
6.2.2.2 DEFINED RELATIONSHIP BETWEEN GEOMETRIC UTILITIES AND TASK DEADLINES

We have already seen that certain combinations of task deadlines and task utility patterns influence the optimality of heuristic schedulers more than others do. In order to determine how such different task deadlines affect the utility values in workload W6, we create three interesting task deadline and utility combinations. W6-1 has a mix of tasks with shorter deadline tasks having lower utility values and longer deadline tasks with higher utility values. W6-2 has tasks with a reverse relationship between task deadlines and utilities. W6-3 has equal task deadlines. All the 3 workloads have same utility values and thus a CR value of 8.

<table>
<thead>
<tr>
<th>$W$</th>
<th>$S_t$</th>
<th>$T_{period}$</th>
<th>$DT$</th>
<th>$DV$</th>
<th>$T_{utility}$</th>
<th>$UT_{(CR)}$</th>
<th>Utility $\rightarrow$ Deadline</th>
<th>$T_{cost}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$W6$-1</td>
<td>5</td>
<td>[1, 2, 2, 4, 4]</td>
<td>UE ($D=P$)</td>
<td>2, 4</td>
<td>[1, 8, 64, 512, 4096]</td>
<td>$G(8)$</td>
<td>Short Deadline $\rightarrow$ Low Utility, Longer Deadline $\rightarrow$ Higher Utility</td>
<td>D</td>
</tr>
<tr>
<td>$W6$-2</td>
<td>5</td>
<td>[4, 4, 2, 2, 1]</td>
<td>UE ($D=P$)</td>
<td>2, 4</td>
<td>[1, 8, 64, 512, 4096]</td>
<td>$G(8)$</td>
<td>Short Deadline $\rightarrow$ High Utility, Longer Deadline $\rightarrow$ Low Utility</td>
<td>D</td>
</tr>
<tr>
<td>$W6$-3</td>
<td>5</td>
<td>[4, 4, 4, 4, 4]</td>
<td>$E$ ($D=P$)</td>
<td>2, 4</td>
<td>[1, 8, 64, 512, 4096]</td>
<td>$G(8)$</td>
<td>N/A</td>
<td>D</td>
</tr>
</tbody>
</table>

Table 6-5: Workload W6 with different Task Deadlines

Figure 6-20 shows the results for W6-1. They indicate a slight degradation in the performance of DASA-ND and LBESA compared to that of W6. Figure 6-21 shows the c.d.f and histogram for DASA-ND and LBESA at a system load of 1.5 for W6-1. It indicates a 0.59 probability that LBESA is optimal and 0.7 probability that DASA-ND is optimal. LBESA has a higher frequency of sub-optimal results compared to DASA-ND. Figure 6-22 and Figure 6-23 compares the variation in AUR values of OPTIMAL, DASA-ND & LBESA for W6 and W6-1 at 150% system load for the 100 simulation runs. The red square represents optimal AUR, blue circle represents DASA-ND, and green triangles represent LBESA. Overlapping AUR values imply that one or more of these produced the same result. Clearly, Figure 6-23 shows that LBESA (green triangles) creates more sub-optimal results for workload W6-1 as compared to W6 shown in Figure 6-22.
Figure 6-20: Load vs AUR values for W 6-1 (Table 6-5), CR=8 compared with Figure 6-47 (Stochastic cost)

Figure 6-21: C.D.F and HISTOGRAM for W 6-1, CR=8 (Table 6-4) at system load of 1.5
For workloads, W6-2 and W6-3, both DASA-ND and LBESA are optimal. Figure 6-24 & Figure 6-25 clearly shows that both DASA-ND and LBESA are not affected by these specific deadline patterns and produce optimal results for workloads with task utilities in geometric sequence (and higher CR) with a high probability. With increasing constant ratio, the higher utility tasks
maintain a higher PUD, since they have very high utility values compared to the lower utility tasks and thus outperform the short tasks in terms of the heuristic.

**Figure 6-24**: Load vs AUR for W 6-2 (Table 6-5), CR=8

**Figure 6-25**: Load vs AUR for W 6-3 (Table 6-5), CR=8 compared with Figure 6-49 (Stochastic cost)
It is apt to conclude that both, DASA-ND and LBESA are optimal for task sets with utility values forming a geometric sequence and high constant ratios. The results show a 100% probability that DASA-ND and LBESA are optimal with less than 0.01 differences in the delta AUR values of optimal and UA schedulers at high system loads for such workloads (Figure 6-19). There are specifically two cases where DASA-ND and LBESA are sub-optimal with less than 80% probability of being optimal. The first case is where the constant ratio is low. Figure 6-26 and Figure 6-27 compare the performance of DASA-ND and LBESA respectively for increasing constant ratios in workloads that have task utilities in a geometric sequence. It clearly shows that DASA-ND is closer to optimal than LBESA for low constant ratios. Both move closer to optimal for higher constant ratios. We also have shown that most often the reason for this is that LBESA does not use any heuristic in breaking ties between equal deadline tasks. The second case, where DASA-ND and LBESA degrade is when there is a wider variation in task deadlines within a given set. Shorter tasks that have low utility values take precedence over longer tasks that have high utility, since short execution costs increases the PUD of low utility tasks.

Figure 6-26 : Performance of DASA-ND with different CR in geometric sequence utilities
To summarize our analysis so far, we conclude that for periodic tasks in single node case,

- At system under loads both DASA-ND and LBESA are optimal irrespective of task utility and deadline patterns.
- Both LBESA and DASA-ND are optimal for workloads that have equal task utilities with deterministic costs irrespective of task deadlines.
- DASA-ND overall performs better than LBESA for workloads that have task utilities in a geometric sequence and deterministic costs.
- DASA-ND and LBESA are optimal for task sets with utility values in geometric sequence and high constant ratios and deterministic costs.
- DASA-ND and LBESA fail to produce optimal results for workloads that have low constant ratio for geometric sequence task utilities with a mix of short and long tasks and added constraint that shorter deadline tasks have lower utility and longer deadline tasks have higher utility. With equal task deadlines, both DASA-ND and LBESA are again optimal in such cases.
6.2.2.3 ARITHMETIC TASK UTILITIES

We continue to experiment with another set of utility patterns. We create utility patterns in which the common difference between the two consecutive task utility values is a constant. This typically means that the task utilities follow an arithmetic sequence. Table 6-6 shows the details of the workloads that have task utilities in arithmetic sequence. We use random task deadlines. Thus, these workloads have a mix of short and long deadline tasks. Both the utility and deadline values remain fixed for each workload across the 100 simulation runs. Workload W9 has a lower Constant Difference (CD) of 1. W10 and W11 have higher constant differences.

<table>
<thead>
<tr>
<th>W</th>
<th>S_i</th>
<th>T_period</th>
<th>DT</th>
<th>DV</th>
<th>T_utility</th>
<th>UT(CD)</th>
<th>T_cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>W9</td>
<td>5</td>
<td>[2, 4, 2, 4, 4]</td>
<td>UE(D=P)</td>
<td>2,4</td>
<td>[2, 3, 4, 5, 6]</td>
<td>A(1)</td>
<td>D</td>
</tr>
<tr>
<td>W10</td>
<td>5</td>
<td>[2, 4, 2, 4, 4]</td>
<td>UE(D=P)</td>
<td>2,4</td>
<td>[100, 200, 300, 400, 500]</td>
<td>A(100)</td>
<td>D</td>
</tr>
<tr>
<td>W11</td>
<td>5</td>
<td>[2, 4, 2, 4, 4]</td>
<td>UE(D=P)</td>
<td>2,4</td>
<td>[1000, 2000, 3000, 4000, 5000]</td>
<td>A(1000)</td>
<td>D</td>
</tr>
</tbody>
</table>

D: Deterministic Cost, G: Geometric Sequence
A: Arithmetic Sequence, D: Deadline

Table 6-6: Workloads with Task Utilities in Arithmetic sequence

Table 6-7 shows the average and standard deviation values for optimal, DASA-ND and LBESA for W9 at different loads. Figure 6-28 shows the graph of system load vs AUR for DASA-ND and LBESA in comparison with optimal schedules for the workload W9. Each plot in the graph represents the average AUR obtained from a set of 100 independent simulations for different system loads. We see that both DASA-ND and LBESA are optimal at system under-loads. At higher system loads, both DASA-ND and LBESA degrade. The graph shows that both DASA-ND’s and LBESA’s degree of optimality changes at different loads. Particularly, DASA-ND’s AUR drops to 0.86 as compared to Optimal’s 0.9 at a load of 1.1 and improves for loads > 1.5. The C.D.F and HISTOGRAM shown in Figure 6-30 & Figure 6-31 for DASA-ND and LBESA at loads 1.1 and 1.5 respectively indicate this pattern more clearly. C.D.F in Figure 6-30 shows a 0.59 probability that DASA-ND is optimal and 0.69 probability that LBESA is optimal. HISTOGRAM for DASA-ND shows a higher frequency of cases where it loses to the optimal by more than 0.1 difference in AUR. In Figure 6-31, the C.D.F shows that DASA-ND’s optimality
improves at a system load of 1.5. Both LBESA and DASA-ND show around 0.8 probability that they are optimal. Overall, there are cases that > 0.1 and < 0.3 delta AUR for DASA-ND at a load of 1.1. It is slightly better for LBESA. Figure 6-29 shows the distribution of AUR for Optimal (red squares), DASA-ND (blue circles) and LBESA (green triangles) across the 100 simulation runs at a system load of 110% for W9. It clearly shows that DASA-ND falls below optimal values for more cases compared to LBESA. Further, we investigate the reasons for sub-optimal results for the workload W9 for both DASA-ND and LBESA. We also investigate specific cases, where DASA-ND’s performance is worse than LBESA for such workloads.

<table>
<thead>
<tr>
<th>%LOAD</th>
<th>AVG AUR</th>
<th>OPTIMAL</th>
<th>AVG AUR</th>
<th>LBESA</th>
<th>STD DEV</th>
<th>OPTIMAL</th>
<th>STD DEV</th>
<th>LBESA</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>80</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>100</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>101</td>
<td>0.920385</td>
<td>0.871923</td>
<td>0.886154</td>
<td>0.009813</td>
<td>0.052734</td>
<td>0.038624</td>
<td></td>
<td></td>
</tr>
<tr>
<td>110</td>
<td>0.899231</td>
<td>0.866923</td>
<td>0.881154</td>
<td>0.027077</td>
<td>0.054233</td>
<td>0.042024</td>
<td></td>
<td></td>
</tr>
<tr>
<td>130</td>
<td>0.845769</td>
<td>0.830769</td>
<td>0.828846</td>
<td>0.041955</td>
<td>0.058112</td>
<td>0.05691</td>
<td></td>
<td></td>
</tr>
<tr>
<td>150</td>
<td>0.805385</td>
<td>0.793462</td>
<td>0.794231</td>
<td>0.056217</td>
<td>0.06864</td>
<td>0.067702</td>
<td></td>
<td></td>
</tr>
<tr>
<td>180</td>
<td>0.753846</td>
<td>0.742692</td>
<td>0.745769</td>
<td>0.072365</td>
<td>0.08621</td>
<td>0.082345</td>
<td></td>
<td></td>
</tr>
<tr>
<td>200</td>
<td>0.726154</td>
<td>0.718462</td>
<td>0.718846</td>
<td>0.080177</td>
<td>0.093555</td>
<td>0.086728</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 6-7: Average AUR values for W9, Arithmetic Utility, CD= 1

Figure 6-28: Load vs AUR for W9 (Table 6-6), Arithmetic Utility, CD = 1
Figure 6-29: AUR distribution for Optimal, DASA-ND, and Lbesa in W9, CD=1 at Load = 1.1

Figure 6-30: C.D.F and HISTOGRAM for W9, CD =1 and Load = 1.1
DASA-ND performs marginally worse than LBESA at system loads in the range of 1.0 to 1.3. Scenario 4 explains the reason why DASA-ND performs worse than LBESA at a system load of 1.1.

**Scenario 4:**
Consider an example scenario shown in Figure 6-32. The optimal scheduler schedules the high utility tasks, T5 and T4 along with T7 and T3 and thus gains an overall AUR of 0.88. On the other hand, both DASA-ND and LBESA fail to accommodate all the high utility tasks together using the PUD heuristic and deadline ordering. LBESA succeeds to a good extent by scheduling the highest utility task, T5 (with longer execution cost) well ahead of T4 (shorter execution cost) and accumulating a utility of 0.85. LBESA does not impose an ordering in case of deadline ties. By random selection, LBESA schedules the highest utility task T5 even though it has a lower PUD than T4, since both have equal deadlines. DASA-ND, initially delays the scheduling the highest utility task T5 (with longer execution cost), in favor of other shorter deadline, shorter length, low PUD tasks that are part of the feasible set. Unable to predict the future influx of low
cost, high PUD tasks, that could abort T5, DASA-ND is greedy to complete all the tasks in the current feasible set in the deadline order. This potentially forces DASA-ND to produce sub-optimal results with a total AUR of 0.77. DASA-ND loses by picking tasks in the order of highest PUD heuristic. It schedules T4 ahead of T5. T5 being a longer task, loses to higher PUD, shorter tasks T6 and T7.

**Scenario 5:**
Consider another example scenario shown in Figure 6-33 for the same workload W9 at a higher system load of 1.5. Both, DASA-ND and LBESA fail to create the optimal schedule. Even though there is marginal difference, in the performance of UA schedulers and the optimal, this example shows that, heuristic algorithms, in the process of completing high utility tasks first, might end up idling in the latter part of the task ordering. Optimal scheduler is able to schedule all high utility tasks and meet more deadlines than DASA-ND or LBESA. As shown in the Figure 6-33, optimal scheduler accommodates, T1, a low utility task in the end-end schedule and still manages to complete the high utility tasks within their deadline. DASA-ND and LBESA eliminate the low utility tasks, since they are the lowest PUD tasks.

We further discuss two interesting scenarios for workload W9 at a high system load of 1.8. Even though in Scenario 6, DASA-ND’s heuristic manages to create optimal schedules even under high system load, in Scenario 7, both LBESA and DASA-ND, misjudge and loose to the optimal scheduler.
Both DASA-ND and LBESA are sub-optimal. LBESA fares better than DASA-ND for W9 at 110% Load.

Figure 6-32: Scenario 4 - Both DASA-ND and LBESA produce sub-optimal results for W9.
Both DASA-ND and LBESA are sub-optimal with an AUR of 0.73 as compared to Optimal AUR of 0.77 for W9 at 150% Load.

Figure 6-33: Scenario 5 - Both DASA-ND and LBESA produce sub-optimal results for W9 at a Load = 150%
On the other hand, LBESA chooses T4 prior to T5 in the deadline order. This choice is random due to deadline ties between T4 and T5. When T6 and T7 arrive, LBESA has completed a major portion of T4. Hence, it still wins against the T6 based on the PUD heuristic and continues to schedule T4. Overall, it meets lesser number of task deadlines, as compared to the DASA-ND and hence produces a sub-optimal result.

**Scenario 7:**
In our experiments, we maintain a constant number of tasks in the workload for different system loads, but instead increase the task costs to reflect the increasing system load. At high system loads, task costs are higher and it becomes infeasible to accommodate all tasks. In Figure 6-35, at a system load of 1.8, both T4 and T5 tend to have high costs and higher utilities. Even, the optimal scheduler is able to accommodate only T4 and rejects T5 in order to meet more tasks with shorter lengths in the lower utility range. Hence, the optimal scheduler judiciously accumulates a total AUR of around 0.70. Both, DASA-ND and LBESA use the PUD heuristic. T4 though a higher utility task, is categorized as a low PUD task due to its high cost. DASA-ND schedules T4 after T1, T2 and T3 complete. T4 is preempted on the arrival of T6 and T7, since T6 and T7 have higher PUD due to low costs. By the time, T6 and T7 finish, T4 becomes infeasible. Hence, both DASA-ND and LBESA are unable to schedule a high utility task, due to its low PUD, even though it could be accommodated amidst other shorter, high PUD tasks. This is inherent to the UA schedulers due to their following shortcomings. One, like other dynamic schedulers, they cannot predict the future influx of short tasks with high PUD. Second, they cannot determine if the high PUD of a task is due to its low cost or more so due to its high utility and vice versa. Third, it cannot assess the time for which a task can be delayed to still complete within its deadline. In all these cases, they rely only on the PUD heuristic and deadline ordering, which defeats them for particular combinations of task properties.
Both DASA-ND and LBESA are sub-optimal. DASA-ND fares better than LBESA for W9 at 180% Load.

Figure 6-34: Scenario 6 - DASA-ND and LBESA produce sub-optimal results for W9
Both DASA-ND and LBESA are sub-optimal with an AUR of 0.58 as compared to Optimal AUR of 0.65 at 180% Load.

Figure 6-35: Scenario 7 - DASA-ND and LBESA produce sub-optimal results for W9
Further, we experiment with workloads that have higher constant differences (W10 and W11). Figure 6-36 & Figure 6-37 show the graph of system load vs AUR for DASA-ND and LBESA in comparison with the optimal schedules for W10 and W11 (Table 6-6) respectively. The results for both DASA-ND and LBESA remain consistent with the results obtained for workload W9. As in W9, DASA-ND performs worse than LBESA at system load of 1.1. Unlike in the W9, higher difference in the utility values of the shorter length and longer tasks should have increased the chances of longer, high utility tasks being scheduled in favor of shorter, low utility tasks by the heuristic schedulers. However, certain task costs combinations still defeat the optimality of heuristic schedulers, forcing them to schedule, shorter tasks that have higher PUD more so due to their low execution costs and frequent periodic arrivals. We illustrate this with an example scenario for W11 in Figure 6-38.

**Scenario 8:**

Figure 6-38 shows a task set, with task utilities in arithmetic sequence and a constant difference of 1000. There are 5 tasks with task periods defined by the vector [2, 4, 2, 4, 4]. T1 and T3 have 2 periodic instances in this set. T2, T4, and T5 have one instance each in a hyper-period 4. T5 is the longest and the highest utility task. Based on the heuristics used by UA schedulers pick shorter deadline tasks (T1 and T3) in the feasible set prior to T2, T4, and T5. The remaining 3
tasks, each with a deadline of 4 units, form the subsequent feasible set. In the order of PUD, DASA-ND schedules T2, T4, and T5. T2 and T4 complete in the order they are scheduled. T5 is aborted in favor of the higher PUD tasks, T6 and T7. Hence, DASA-ND does not schedule the highest utility task T5. Whereas, LBESA does not use any heuristic to break equal deadline ties; it picks T5 at random LBESA finishes T5 before T2 and T4. Even though, T5 is preempted once in favor of higher PUD T6, T5 still completes before its deadline since it started early. At this point, T2, T4, and T7, all have equal deadlines, but together cannot form the feasible set. Hence, LBESA eliminates, the lowest PUD task, T7 (relatively higher cost) from the set and schedules, T4 and T2, in the order. LBESA wins, for the sole reason, that T5 is scheduled earlier. Both T4 and T2 have equal chance of being scheduled with T5, due to deadline ties. If, either T4 or T2 was scheduled prior to T5, LBESA would be sub-optimal similar to that of DASA-ND. Hence, it is evident that PUD heuristic fails to schedule tasks with high utility values for such workloads. If, LBESA addresses deadline ties, by scheduling tasks in decreasing order of PUD, it degenerates to DASA-ND in such scenarios.

Figure 6-37 : Load vs AUR for W11 (Table 6), CD = 1000

Overall, both DASA-ND and LBESA show less than 80% probability of being optimal at different system loads. Hence, we conclude that a higher common difference do not alter the performance of DASA-ND and LBESA. Both DASA-ND and LBESA degrade and produce sub-optimal results for arithmetic sequence utilities.
DASA-ND creates sub-optimal schedules with an AUR of 0.74 as compared to OPTIMAL AUR of 0.84.

Figure 6-38: Scenario 8 - DASA-ND produce sub-optimal results for W11 (Table 6-6) at a Load of 110%
6.2.2.4 DEFINED RELATIONSHIP BETWEEN ARITHMETIC UTILITIES AND TASK DEADLINES

We next determine how different task deadlines affect the arithmetic sequence utility values in workload W9 and W10. We create similar deadline and utility combinations as in the Table 6-5 for the arithmetic sequence utilities. W12 and W10-1 in Table 6-8 has a mix of tasks with shorter deadline tasks having lower utility values and longer deadline tasks with higher utility values. W10-2 has tasks with a reverse relation between task deadlines and utilities. W10-3 has equal task deadlines.

<table>
<thead>
<tr>
<th>W</th>
<th>S_5</th>
<th>T_period</th>
<th>DT</th>
<th>DV</th>
<th>utility</th>
<th>UT(CD)</th>
<th>Utility → Deadline</th>
<th>T_cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>W12</td>
<td>5</td>
<td>[1, 2, 2, 4, 4]</td>
<td>UE (D=P)</td>
<td>1,2,4</td>
<td>[2, 3, 4, 5, 6]</td>
<td>A(1)</td>
<td>Short Deadline→Low Utility</td>
<td>D</td>
</tr>
<tr>
<td>W10-1</td>
<td>5</td>
<td>[1, 2, 2, 4, 4]</td>
<td>UE (D=P)</td>
<td>1,2,4</td>
<td>[100, 200, 300, 400, 500]</td>
<td>A(100)</td>
<td>Short Deadline→Low Utility</td>
<td>D</td>
</tr>
<tr>
<td>W10-2</td>
<td>5</td>
<td>[4, 4, 2, 2, 1]</td>
<td>UE (D=P)</td>
<td>1,2,4</td>
<td>[100, 200, 300, 400, 500]</td>
<td>A(100)</td>
<td>Short Deadline→High Utility</td>
<td>D</td>
</tr>
<tr>
<td>W10-3</td>
<td>5</td>
<td>[4, 4, 4, 4, 4]</td>
<td>E (D=P)</td>
<td>4</td>
<td>[100, 200, 300, 400, 500]</td>
<td>A(100)</td>
<td>N/A</td>
<td>D</td>
</tr>
</tbody>
</table>

Table 6-8: Workload with different Task Deadlines for Arithmetic sequence Task Utilities

Figure 6-39 & Figure 6-40 show the results for W12 and W10-1. W12 and W10-1 differ only in their constant difference. There is no deviation from the results obtained for workloads W9 to W11 (Table 6-6), except that DASA-ND improves slightly. Figure 6-41 & Figure 6-42 show the corresponding C.D.F and HISTOGRAM for W12 at a system load of 1.1 and 1.5 respectively. At a system load of 1.1, C.D.F indicates a 0.55 probability that DASA-ND is optimal and a 0.65 probability that LBESA is optimal. DASA-ND shows a higher frequency of cases where the difference in AUR between itself and optimal is > 0.01 & < 0.1.
Hence, we conclude that irrespective of constant difference in the arithmetic sequence, task sets having a combination of tasks with shorter deadline & low utility values along with tasks having longer deadline & higher utility values yield sub-optimal results for both DASA-ND and LBESA.

Figure 6-39: Load vs AUR for W12, CD = 1 (Table 6-8)

Figure 6-40: Load vs AUR for W10-1, CD = 100 (Table 6-8) compared with Figure 6-48 (Stochastic cost)
Figure 6-41: C.D.F and HISTOGRAM for W12, CD = 1 (Table 6-8) at Load = 110%

Figure 6-42: C.D.F and HISTOGRAM for W12, CD = 1 (Table 6-8) at Load = 150%
Figure 6-43 shows the results for workload W10-2. W10-2 has similar utility values as W 10-1, but shorter deadline tasks have higher utility values and longer deadline tasks have lower utility values. Interestingly, for such workloads, even with arithmetic sequence utilities, both DASA-ND and LBESA show near optimal results. This is expected, since the shorter tasks, favored against the longer tasks have higher PUD values, due to their higher utility values and lower costs. However, in the previous workload W10, the same tasks had higher PUD, not because of their high utility values, but only because of their low cost values. LBESA and DASA-ND show close to 0.9 probability of being optimal.

Figure 6-43: Load vs AUR for W 10-2, CD = 100 (Table 6-8)

Figure 6-44 shows the graph of system load vs average AUR values for DASA-ND, LBESA in comparison with the OPTIMAL in W 10-3 (Table 6-8). W 10-3 has tasks with equal deadlines and task utilities in an arithmetic sequence. Both, DASA-ND and LBESA show similar performance under different system loads and are sub-optimal. Hence, unlike in geometric sequence utilities, equal task deadlines for arithmetic sequence utilities do not improve the optimality of UA schedulers. The influencing factor for such workloads is the utility pattern rather than mere task deadlines.
It is apt to conclude that both DASA-ND and LBESA are sub-optimal for task sets with utility values in arithmetic sequence and random task deadlines, irrespective of the common difference factor. Moreover, both the UA schedulers, show sub-optimal results even with equal task deadlines and have less than 0.8 probability of being optimal. For a few system loads, LBESA is marginally better than DASA-ND for such workloads, attributed merely to the way deadline ties between higher utility tasks are handled. LBESA does not use any defined order, unlike DASA-ND that uses PUD heuristic to break ties [Clark, 1990]. *The only case, both the UA schedulers, reach near optimal results for arithmetic task utilities are when shorter deadline tasks have higher utility values than the longer deadline tasks.*

*Figure 6-45 & Figure 6-46* compares the performance of DASA-ND and LBESA respectively for workloads that have task utilities in an arithmetic sequence and different task deadline to utility relation. It clearly shows that both DASA-ND and LBESA are near optimal when shorter tasks have higher utility values than longer tasks. LBESA performs worse at particular system loads when tasks have equal deadlines.
Figure 6-45: Performance of DASA-ND with arithmetic sequence utilities and different deadlines

Figure 6-46: Performance of LBESA with arithmetic sequence utilities and different deadlines
6.3 **Periodic Task Model, Stochastic Cost Model**

Until now, we have used deterministic execution costs for different instances of the periodic tasks. Most often, real workloads do not show deterministic cost pattern across different instances. Unexpected execution cost overruns for a particular task can force the system to reject important tasks. This aspect of variation in task costs across the instances in represented using a stochastic model for task execution costs. In the next set of experiments, we do not use deterministic cost but rather determine execution costs for different instances using a probability distribution function such as the lognormal probability distribution function. The expected value for the distribution function is derived as earlier using the UUniFast algorithm described in [Bini and Buttazzo, 2005]. We also assume that task deadlines are equal to their corresponding periods. We also ensure that the task costs derived from the distribution are less than or equal to their corresponding task deadlines. Hence, a periodic instance of a task either completes or aborts before the arrival of the subsequent one.

<table>
<thead>
<tr>
<th>W</th>
<th>S</th>
<th>$T_{period}$</th>
<th>$DT$</th>
<th>$DV$</th>
<th>$T_{utility}$</th>
<th>$U_{T(CR/CD)}$</th>
<th>Utility $\rightarrow$ Deadline</th>
<th>$T_{cost}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>W-S6-1</td>
<td>5</td>
<td>[1, 2, 2, 4, 4]</td>
<td>UE(D)P</td>
<td>1,2,4</td>
<td>[1, 8, 64, 512, 4096]</td>
<td>G(8)</td>
<td>SD -----&gt; LU</td>
<td>S</td>
</tr>
<tr>
<td>W-S10-1</td>
<td>5</td>
<td>[1, 2, 2, 4, 4]</td>
<td>UE(DL)P</td>
<td>1,2,4</td>
<td>[100, 200, 300, 400, 500]</td>
<td>A(100)</td>
<td>SD -----&gt; LU</td>
<td>S</td>
</tr>
<tr>
<td>W-S6-3</td>
<td>5</td>
<td>[4, 4, 4, 4, 4]</td>
<td>E(DL)P</td>
<td>4</td>
<td>[1, 8, 64, 512, 4096]</td>
<td>G(8)</td>
<td>N/A</td>
<td>S</td>
</tr>
<tr>
<td>W-S10-3</td>
<td>5</td>
<td>[4, 4, 4, 4, 4]</td>
<td>E(DL)P</td>
<td>4</td>
<td>[100, 200, 300, 400, 500]</td>
<td>A(100)</td>
<td>N/A</td>
<td>S</td>
</tr>
</tbody>
</table>


Table 6-9: Workloads with Stochastic Execution Costs

We have used the same workloads used in the previous experiments and compared the results from the deterministic cost model with that of the stochastic cost model. Workload W-S6-3 has equal task deadlines and utility pattern similar to that of workloads in W6-3 (geometric sequence utilities). Workload W-S10-3 has equal task deadlines and utility pattern similar to that of workloads in W10-3 (arithmetic sequence utilities). Workloads W-S 6-1 and W-S10-1 have task sets with a combination of shorter deadline tasks having low utility values and longer deadline tasks having high utility values.
Stochastic cost model shows higher overall AUR values for all schedules compared to the deterministic model. This is because of the varying execution times across different periodic cycles. Hence, the actual value of $\sum Ci / Ti$ i.e. system load may be much less or greater than the predicted value. There is also an improvement in the overall performance of DASA-ND and LBESA in the stochastic model. Such results are dependent on the probability distributions used in generating the task execution times.

We have previously concluded that for workload W 6-1, both DASA-ND and LBESA are optimal. Table 6-10 shows the average and standard deviation in AUR values for OPTIMAL, DASA-ND and LBESA for W-S 6-1. Comparing Figure 6-20 & Figure 6-47 for W6-1 (deterministic cost model) and W-S6-1 (stochastic cost model) respectively, we see no difference in the degree of optimality of DASA-ND and LBESA. However, there is an increase in the average AUR values, mainly due to stochastic task costs. For similar reasons explained before in earlier scenarios, both DASA-ND and LBESA perform worse for a few specific task cost combinations at a system load of 150%.

<table>
<thead>
<tr>
<th>%LOAD</th>
<th>AVG AUR-OPTIMAL</th>
<th>AVG AUR-DASA-ND</th>
<th>AVG AUR-LBESA</th>
<th>STD DEV OPTIMAL</th>
<th>STD DEV DASA-ND</th>
<th>STD DEV LBESA</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>0.999952</td>
<td>0.999828</td>
<td>0.999825</td>
<td>0.000</td>
<td>0.001</td>
<td>0.001</td>
</tr>
<tr>
<td>80</td>
<td>0.993427</td>
<td>0.99156</td>
<td>0.991529</td>
<td>0.025</td>
<td>0.026</td>
<td>0.026</td>
</tr>
<tr>
<td>100</td>
<td>0.99291</td>
<td>0.990416</td>
<td>0.990616</td>
<td>0.021</td>
<td>0.025</td>
<td>0.024</td>
</tr>
<tr>
<td>101</td>
<td>0.992767</td>
<td>0.99036</td>
<td>0.990566</td>
<td>0.021</td>
<td>0.025</td>
<td>0.024</td>
</tr>
<tr>
<td>110</td>
<td>0.988259</td>
<td>0.986194</td>
<td>0.985206</td>
<td>0.029</td>
<td>0.030</td>
<td>0.032</td>
</tr>
<tr>
<td>130</td>
<td>0.981676</td>
<td>0.977553</td>
<td>0.977534</td>
<td>0.035</td>
<td>0.037</td>
<td>0.037</td>
</tr>
<tr>
<td>150</td>
<td>0.974224</td>
<td>0.948959</td>
<td>0.949296</td>
<td>0.043</td>
<td>0.148</td>
<td>0.149</td>
</tr>
<tr>
<td>180</td>
<td>0.975557</td>
<td>0.970856</td>
<td>0.970086</td>
<td>0.039</td>
<td>0.041</td>
<td>0.042</td>
</tr>
<tr>
<td>200</td>
<td>0.969628</td>
<td>0.965208</td>
<td>0.964798</td>
<td>0.045</td>
<td>0.045</td>
<td>0.046</td>
</tr>
</tbody>
</table>

Table 6-10 : Average AUR values for W-S6-1

We have also concluded that for workload W 10-1, both DASA-ND and LBESA are sub-optimal with a less than 0.8 probability of being optimal. Table 6-10 shows the average and standard deviation in AUR values for OPTIMAL, DASA-ND and LBESA for W-S10-1. Comparing Figure 6-40 for W10-1 (with deterministic cost model) and Figure 6-48 for W-S10-1 (with stochastic cost model), we see no difference in the degree of optimality of DASA-ND and LBESA. However, there is a decrease in the average AUR values at underload, mainly due to stochastic task costs.
Figure 6-47: Load vs AUR for W-S 6-1 (Table 6-9) compared with Figure 6-20 (Deterministic cost)

<table>
<thead>
<tr>
<th>% LOAD</th>
<th>AVG AUR-OPTIMAL</th>
<th>AVG AUR-DASA-ND</th>
<th>AVG AUR-LBESA</th>
<th>STD DEV OPTIMAL</th>
<th>STD DEV DASA-ND</th>
<th>STD DEV LBESA</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>0.996957</td>
<td>0.995652</td>
<td>0.995652</td>
<td>0.018</td>
<td>0.025</td>
<td>0.025</td>
</tr>
<tr>
<td>80</td>
<td>0.942609</td>
<td>0.93087</td>
<td>0.929565</td>
<td>0.071</td>
<td>0.085</td>
<td>0.086</td>
</tr>
<tr>
<td>100</td>
<td>0.909565</td>
<td>0.885652</td>
<td>0.886957</td>
<td>0.081</td>
<td>0.100</td>
<td>0.097</td>
</tr>
<tr>
<td>101</td>
<td>0.908696</td>
<td>0.884783</td>
<td>0.885652</td>
<td>0.082</td>
<td>0.100</td>
<td>0.097</td>
</tr>
<tr>
<td>110</td>
<td>0.886956</td>
<td>0.862174</td>
<td>0.863478</td>
<td>0.084</td>
<td>0.099</td>
<td>0.101</td>
</tr>
<tr>
<td>130</td>
<td>0.85</td>
<td>0.827391</td>
<td>0.827391</td>
<td>0.093</td>
<td>0.109</td>
<td>0.107</td>
</tr>
<tr>
<td>150</td>
<td>0.832174</td>
<td>0.80913</td>
<td>0.810435</td>
<td>0.106</td>
<td>0.114</td>
<td>0.115</td>
</tr>
<tr>
<td>180</td>
<td>0.806522</td>
<td>0.781304</td>
<td>0.777391</td>
<td>0.116</td>
<td>0.127</td>
<td>0.128</td>
</tr>
<tr>
<td>200</td>
<td>0.790435</td>
<td>0.76</td>
<td>0.762174</td>
<td>0.109</td>
<td>0.122</td>
<td>0.121</td>
</tr>
</tbody>
</table>

Table 6-11: Average AUR values for W-S 10-1

We have shown that for workload W 6-3, both DASA-ND and LBESA are near optimal. Comparing Figure 6-25: Load vs AUR for W 6-3 (Table 6-5) & Figure 6-49: Load vs AUR: W-S 6-3 (Table 6-9) for W6-3 (deterministic cost model) and W-S6-3 (stochastic cost model) respectively, we see that the results are almost similar. Except that, both DASA-ND and LBESA show a little more degradation at a system load of 1.5 for a few task costs and deadline combinations for such workloads. Similarly, comparing W10-3 (with deterministic cost model) & W-S 10-3 (with stochastic cost model) shows no difference. Hence, we conclude that the results for all periodic workloads are similar in both deterministic and stochastic cost models.
Figure 6-48: Load vs AUR W-S 10-1 (Table 6-9) compared with Figure 6-40 (Deterministic cost)

Figure 6-49: Load vs AUR W-S 6-3 (Table 6-9) compared with Figure 6-25 (Deterministic cost)
The next section discusses the results for workloads that use aperiodic task model. We have experimented similar task utility and deadline patterns as used in the periodic task model.

### 6.4 Aperiodic Task Model

After determining the optimality of DASA-ND and LBESA for periodic task model in single-node environment, we next determine the performance of DASA-ND and LBESA for the aperiodic task model. Execution times “t-c” are chosen as a random variable from the lognormal probability distribution. The arrival-rate $\lambda$ is modeled as a random variable from the Poisson distribution with an average value $\lambda_{\text{avg}} = (S_t \times t-c) / \rho$, where “$\rho$” represents system load. The slack is computed as a random variable from a uniform distribution within an interval of [0.5-2.5]. The task deadline is the sum of the task execution time and the slack. We have used the same workloads used in the previous experiments and compared the results from the periodic task model with that of the aperiodic task model.
We have shown that for periodic task model with task sets having equal task utilities such as in W3, both DASA-ND and LBESA are optimal. Comparing Figure 6-5 & Figure 6-51 for W3 (periodic task model) and W-A3 (aperiodic task model), we see that both DASA-ND and LBESA are close to optimal. Similarly, the c.d.f and histogram show 100% probability of being optimal with less than 0.01 differences in the delta AUR values.

Figure 6-52 shows the C.D.F and HISTOGRAM for both DASA-ND and LBESA for W-A3 at a system load of 1.5. C.D.F indicates a 0.95 probability that both DASA-ND and LBESA are optimal. They fail in 3 of the 100 random samples with a difference of more than 0.1 in the AUR values for similar reasons explained for the periodic task model.
We have shown that for periodic task model with task sets having utilities in geometric sequence such as in W6, both DASA-ND and LBESA are close to optimal. Comparing Figure 6-15 & Figure 6-53 for W6 (periodic task model) and W-A6 (aperiodic task model), we see that such task sets do not result in optimal results for both DASA-ND and LBESA. We have compared the simulation results of DASA-ND and LBESA with the optimal schedules for a short simulation time with less than 12 tasks, since the optimal scheduler grows super-exponentially with increasing tasks. Under this constraint, DASA-ND and LBESA show wider deviation from the optimal results.

Figure 6-53 shows that on an average there is a wide difference in the average AUR values of DASA-ND and optimal scheduler. Interestingly, LBESA perform slightly better than DASA-ND.

Figure 6-54 Figure 6-52 shows the C.D.F and HISTOGRAM for both DASA-ND and LBESA for W-A6 at a system load of 1.5. C.D.F indicates a 0.42 probability that DASA-ND is optimal and a
0.43 probability that LBESA is optimal. The difference is more evident in the histogram. LBESA shows a higher frequency of cases, where the difference in the AUR values is in the interval of 0.0 and 0.1 DASA-ND shows higher frequency in the interval of 0.4 and 0.45. Figure 6-55 and Figure 6-56, more closely identify the exact runs that resulted in sub-optimal results for LBESA and DASA-ND. They show the wide variation that exists in the performance of UA schedulers and optimal. Figure 6-55 compares LBESA with optimal schedules. Figure 6-56 compares DASA-ND with optimal schedules. We also highlight the specific runs where DASA-ND is worse than LBESA.

We further analyze these special cases, where LBESA and DASA-ND produce sub-optimal results. We consider 3 scenarios at a system load of 1.5. Scenario 9 and Scenario 10 show how DASA-ND fails to create optimal schedules for particular task sets. Scenario 11 and Scenario 12 depict cases where both DASA-ND and LBESA fail to create optimal task orderings.

**Scenario 9:**
Consider the example case shown in Figure 6-57, with 6 aperiodic tasks for W-A6. The top segment describes the task properties, such as their arrival time, utility value, execution cost and deadline. The bottom segment shows the OPTIMAL, LBESA and DASA-ND schedules for this task set for 8 units of time. Optimal scheduler picks T3, which has almost zero cost and the highest utility task T5. It idles later, since no other task is feasible within a stipulated time of 8 units. Both LBESA and DASA-ND, pick T3 since it has the highest PUD and earliest deadline. LBESA then creates a feasible set from the tasks, T1, T2, T5 and T4. LBESA considers tasks in deadline order. The 4 tasks in LOW-HIGH deadline order are [T2, T1, T5, T4] and LOW-HIGH PUD order are [T1, T2, T4, T5]. LBESA finds T2, T1, T5 and T4 infeasible and thus eliminates the lowest PUD task, which happens to be T1. Again, it finds T2, T5 and T4 together infeasible, and eliminates lowest PUD task T2. The remaining 2 tasks in the feasible set are T4 and T5. Since T5 has earlier deadline than T4, T5 is scheduled prior to T4. LBESA accumulates an overall AUR of 0.47 and remains optimal in this case. DASA-ND, on the other hand, creates a feasible set, by selecting tasks in the PUD order. The HIGH-LOW PUD order for the 4 tasks is [T5, T4, T2, T1]. It inserts tasks into the feasible set in deadline order. DASA-ND creates a deadline ordered feasible set of [T1/16.78, T5/20.4, and T4/20.7]. Hence, DASA-ND delays the highest utility task and schedules the lowest utility task T1 prior to T5 only with a hope of
eventually completing all the tasks in the current feasible set. Since, we have considered simulation results for a short length of 8 units of time, DASA-ND can only schedule T1 [U=1] and thus accumulates only a meager part of the total utility possible.

**Scenario 10:**
Consider the example case shown in Figure 6-58 with 8 aperiodic tasks for W-A6. Optimal scheduler picks T5 and T6, which are the highest utility tasks. Within the 8 units of time, it also schedules the lowest utility task T1 and accumulates an overall 0.84 AUR. LBESA eliminates the lowest PUD tasks while creating the feasible set, and schedules T5 and T6, one short of optimal. Even then, by rejecting T1 with an utility of 1, LBESA accumulates a total AUR of around 0.84. DASA-ND does not reject the low utility task, since it is part of the feasible set created by DASA-ND. T1 has an earlier deadline than T5, and hence T5 is delayed by the amount of T1’s execution time. DASA-ND later pays for this delay, since it is unable to complete T5, favoring higher PUD task T6 that arrives in future. It finally accumulates an AUR of 0.42 as opposed to an OPTIMAL AUR of 0.84.

**Scenario 11 & 12:**
The example cases shown in Figure 6-59 & Figure 6-60 have both DASA-ND and LBESA producing sub-optimal results for W-A6 at a load of 1.5. Optimal scheduler picks the highest utility tasks.
Figure 6-53: Load vs AUR for W A6 compared with Figure 6-15 (periodic task model)

Figure 6-54: C.D.F and HISTOGRAM for W-A6 (Table 12) at Load = 1.5
**Figure 6-55:** AUR Distribution for Optimal and LBESA W-A 6 at 150% Load

**Figure 6-56:** AUR Distribution for Optimal and DASA-ND W-A 6 at 150% Load
Figure 6-57: Scenario 9 – W-A6, DASA-ND fails to produce OPTIMAL results.
Figure 6-58: Scenario 10 - W A6 DASA-ND fails to produce OPTIMAL results at a Load of 150%. 
Figure 6-59: Scenario 11-W A6, DASA-ND, & LBESA produce sub-optimal results at a Load of 150%.

Figure 6-60: Scenario 12-W A6, DASA-ND, & LBESA produce sub-optimal results at a Load of 150%
The heuristics used by UA schedulers, tend to delay the higher utility and highest PUD tasks if there are other lower utility (not lowest) tasks with shorter deadline in the feasible set. Figure 6-60 shows an interesting case. T4, T5, T6 are all high utility tasks. They all arrive within a short difference of time. They have shorter deadlines and high PUD values. Based on the heuristics used by DASA-ND and LBESA, they all have high chances of completing. However, in reality, UA schedulers can select the best of low utility and high utility tasks. When there are 3 tasks, with 2 of them having the highest utility and other a slightly lower value, DASA-ND and LBESA are unable to pick the two highest utility tasks. They are greedy in trying to complete all higher utility tasks and eventually loose one of the highest utility tasks. In this example, T4 [U=512] and T5 [U=4096] are chosen initially. By the time, T6 [U=4096] arrives, T4 has completed, and T5 partially completed. T5 with a higher PUD is chosen as against the highest utility task T6. T6 becomes infeasible by the time T5 completes. Optimal scheduler on the other hand, is able to find the best ordering by initially scheduling T5 and then accommodating T6. It rejects T4.

Previously, we have shown that for periodic task sets having utilities in arithmetic sequence, both DASA-ND and LBESA produce sub-optimal results. We also had shown cases where DASA-ND performs marginally lower than LBESA for such task sets. Comparing Figure 6-36 & Figure 6-61 for W10 (periodic task model) and W-A10 (aperiodic task model), we see similar results in the aperiodic task model as well.

Figure 6-61 shows that on an average there is a smaller difference in the AUR values of DASA-ND and optimal scheduler. LBESA performs slightly better than DASA-ND for particular system loads. On an average LBESA is better then DASA-ND by a slight margin. Figure 6-62 shows the AUR distribution of OPTIMAL, DASA-ND and LBESA for W-A10 at a load of 1.5. Both DASA-ND and LBESA produce sub-optimal results and show < 0.7 probability of being optimal under different loads. The figure also highlights the specific runs, where DASA-ND underperforms both optimal, and LBESA. We discuss two of these special cases and explain the reasons for sub-optimal results of DASA-ND and LBESA.

Scenario 13:
Consider the example case shown in Figure 6-63, with 5 aperiodic tasks for W-A10. The top segment describes the task properties, such as their arrival time, utility value, execution cost and deadline.
The bottom segment shows the OPTIMAL, LBESA and DASA-ND schedules for this task set for 8 units of time. Optimal scheduler, picks the highest utility task T4 and later T5. It idles, since no other task is feasible within a stipulated time of 8 units. Both, DASA-ND and LBESA initially schedule the higher utility task T4 [U = 400], which also happens to be the highest PUD task with the shortest deadline. For, LBESA, the next step is to create a feasible set of the remaining tasks, T1, T2, T3 and T4. LBESA considers tasks in deadline order. The 4 tasks in LOW-HIGH deadline order are [T3, T1, T5, T2] and LOW-HIGH PUD order are [T1, T2, T3, T5]. LBESA finds T3, T1, T5 and T2 infeasible and thus eliminates the lowest PUD task, which happens to be T1. Again, it finds T3, T5 and T2 infeasible, and eliminates T2. The remaining 2 tasks in the feasible set are T3 and T5. LBESA finds both are infeasible together and eliminates the lower PUD of the two, T3. Hence, by law of LBESA’s heuristics, the highest PUD, and the highest utility task is completed within its deadline. LBESA accumulates an overall AUR of 0.60 and remains optimal in this case. DASA-ND, on the other hand, creates a feasible set, by selecting tasks in the PUD order. The HIGH-LOW PUD order for the 4 tasks is [T5, T3, T2, T1]. It inserts tasks into the feasible set in deadline order. DASA-ND creates a deadline ordered feasible set of [T1/ 6.78, T5/ 9.12, and T2/ 16.81]. Hence, DASA-ND delays the highest utility task and schedules the lowest utility task T1 prior to T5 with the hope of eventually completing all the tasks in the current feasible set. Since, we have considered simulation results for a short length of 8 units of time; DASA-ND fails to complete T5 and accumulates a low AUR of 0.33.

**Scenario 14:**
Consider the example case shown in Figure 6-64, with 10 aperiodic tasks for W-A10 at a load of 1.5. The top segment describes the task properties, such as their arrival time, utility value, execution cost and deadline. The bottom segment shows the OPTIMAL, LBESA and DASA-ND schedules for this task set for 8 units of time. The optimal scheduler, schedules, T3 [U= 300], T4 [U=400] and T9 [U = 500], thus accumulating an overall AUR of 0.38. Both, DASA-ND and LBESA initially schedule the higher utility task T4, which also happens to be the highest PUD task with the shortest deadline. In the subsequent steps, DASA-ND schedules a low utility task prior to higher PUD ones for similar reasons explained in Scenario 13. LBESA, though initially schedules T3, as in the optimal case, midway preempts T3 when T10 arrives. T10 has a shorter deadline and is part of the feasible set. Eventually, within the stipulated time of 8 units, neither T3 nor T10
complete. Thus, LBESA accumulates an AUR of 0.28 only. Both DASA-ND and LBESA produce sub-optimal results.

![Graph showing Load vs AUR for W-A10 compared with Figure 6-36 (periodic task model)](image)

Figure 6-61: Load vs AUR for W-A10 compared with Figure 6-36 (periodic task model)

![Graph showing AUR Distribution for W-A10 at 150% Load](image)

Figure 6-62: AUR Distribution for W-A10 at 150% Load
Figure 6-63: Scenario 13 – W-A10, DASA-ND fails to produce OPTIMAL results at a Load of 150%.
Figure 6-64: Scenario 14-W A10, DASA-ND & LBESA fail to produce optimal results at a Load = 150%.
6.5 Summary of Results for Single-node

To summarize our analysis so far, we conclude that for workloads on a single node (see Table 6-13: Summary of Results for Single-node),

- For periodic tasks, both DASA-ND and LBESA show 100% probability of being optimal with $\leq 0.01$ difference in AUR values, for geometric task utilities at high constant ratios for both deterministic and stochastic cost models. They are sub-optimal with a less than 80% likelihood of being optimal for similar task sets with lower constant ratios. DASA-ND is marginally better than LBESA at lower constant ratios.

- For periodic tasks, both DASA-ND and LBESA show 100% probability of optimal results for equal utility tasks with both deterministic and stochastic costs.

- For periodic tasks, DASA-ND and LBESA are optimal with less than 0.8 probability for task sets with utility values in arithmetic sequence, irrespective of the common difference factor, for both deterministic and stochastic cost models. LBESA is marginally better than DASA-ND for a particular system loads. Both DASA-ND and LBESA vary in their optimality over different system loads, which are not seen in workloads with geometric sequence utilities. They improve and show almost a 0.9 probability of being optimal for such workloads, when there is defined relation between the task costs and utility values. They perform best for combinations of tasks having high cost and low utility value along with tasks having low cost, high utility value.

- For aperiodic tasks, apart from workloads that have equal utilities, other utility patterns adversely affect the optimality of DASA-ND and LBESA. DASA-ND shows around 0.42 probability of being optimal for geometric sequence utilities and LBESA shows 0.43 probability of being optimal.
<table>
<thead>
<tr>
<th>COST MODEL</th>
<th>UTILITY TYPE</th>
<th>U→D</th>
<th>DASA-ND</th>
<th>LBESA</th>
<th>DASA-ND vs LBESA</th>
<th>COMMENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>DETERMINISTIC</td>
<td>EQUAL</td>
<td>EQUAL</td>
<td>OPTIMAL</td>
<td>OPTIMAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DETERMINISTIC</td>
<td>GEOMETRIC(2)</td>
<td>SD----&gt;LU</td>
<td>SUB-OPTIMAL</td>
<td>SUB-OPTIMAL</td>
<td>DASA-ND is marginally better</td>
<td>CR =2</td>
</tr>
<tr>
<td>DETERMINISTIC</td>
<td>GEOMETRIC(10)</td>
<td>LD----&gt;HU</td>
<td>OPTIMAL</td>
<td>OPTIMAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DETERMINISTIC</td>
<td>ARITHMETIC</td>
<td>SD----&gt;LU</td>
<td>SUB-OPTIMAL</td>
<td>SUB-OPTIMAL</td>
<td>LBESA is marginally better for few loads</td>
<td>CD=1,100,1000</td>
</tr>
<tr>
<td>DETERMINISTIC</td>
<td>ARITHMETIC</td>
<td>SD----&gt;HU</td>
<td>OPTIMAL</td>
<td>OPTIMAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DETERMINISTIC</td>
<td>GEOMETRIC</td>
<td>SD----&gt;LU</td>
<td>SUB-OPTIMAL</td>
<td>SUB-OPTIMAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DETERMINISTIC</td>
<td>ARITHMETIC</td>
<td>SD----&gt;LU</td>
<td>SUB-OPTIMAL</td>
<td>SUB-OPTIMAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STOCHASTIC</td>
<td>GEOMETRIC</td>
<td>SD----&gt;LU</td>
<td>NEAR OPTIMAL</td>
<td>NEAR OPTIMAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STOCHASTIC</td>
<td>ARITHMETIC</td>
<td>SD----&gt;LU</td>
<td>SUB-OPTIMAL</td>
<td>SUB-OPTIMAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STOCHASTIC</td>
<td>GEOMETRIC</td>
<td>SD----&gt;LU</td>
<td>NEAR OPTIMAL</td>
<td>NEAR OPTIMAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STOCHASTIC</td>
<td>ARITHMETIC</td>
<td>SD----&gt;LU</td>
<td>SUB-OPTIMAL</td>
<td>SUB-OPTIMAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STOCHASTIC</td>
<td>EQUAL</td>
<td>-----</td>
<td>NEAR OPTIMAL</td>
<td>NEAR OPTIMAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STOCHASTIC</td>
<td>GEOMETRIC</td>
<td>-----</td>
<td>SUB-OPTIMAL</td>
<td>SUB-OPTIMAL</td>
<td></td>
<td>LBESA is marginally better</td>
</tr>
<tr>
<td>STOCHASTIC</td>
<td>ARITHMETIC</td>
<td>-----</td>
<td>SUB-OPTIMAL</td>
<td>SUB-OPTIMAL</td>
<td></td>
<td>LBESA is marginally better</td>
</tr>
</tbody>
</table>

Table 6-13: Summary of Results for Single-node

10 NEAR OPTIMAL: PROBABILITY THAT IT IS OPTIMAL is >= 0.89

11 SUB-OPTIMAL: PROBABILITY THAT IT IS OPTIMAL is < 0.80
Chapter 7  Results and Optimality Analysis: Distributed System

A distributed real-time system consists of an interconnection of \( n \) nodes with \( m \) tasks mapped to these nodes. Each node schedules the locally available tasks. Such systems have two types of tasks. The first one is the local tasks. As explained before, local tasks do not traverse multiple node-boundaries. They finish on the same node they originate. These tasks constitute the local load on their respective nodes. The second type is the distributed tasks, which traverse more than one node boundary. Distributed tasks can be either periodic or aperiodic and have different numbers of sub-segments, either equal or unequal in length. A distributed task contributes to the overall system utility only when all the segments complete on their respective nodes before the task deadline. Distributed tasks, together with the local tasks determine the total load in the system.

We define the distributed system load as the sum of the total loads on each of the \( n \) nodes divided by the number of nodes. The total system load is defined by all the distributed tasks that execute on each of these nodes. If there is an equal load distribution on all the \( n \) nodes, system load equals the load on any of the \( n \) nodes. Unequal load distribution results in one of the \( n \) nodes having high load either due to more tasks and/or due to longer tasks. The remaining nodes are either idle or execute below their capacity.

Usually in a multi-node system, there are two stages to scheduling. The initial stage consists of allocating the available tasks to system nodes and the second stage finds an efficient ordering of local tasks within each node.

In a multi-node system, where each task can execute on any available node, an efficient global assignment ensures to balance the load on all the nodes. However, in the case of distributed real-time systems, tasks properties themselves specify the segment to node mappings. This is based on the remote procedure call (RPC) inherent to the tasks’ execution. Hence, we cannot guarantee at all times that the system load is equally
distributed among the available nodes. In our simulation, we statically create the segment to node mapping. Our workloads comprehensively cover different patterns of task to node assignments, which in turn decide whether the load distribution is balanced or unbalanced among individual nodes in the system.

In UA scheduling environments, local schedulers on each node attempt to make locally optimal scheduling decisions to maximize the overall system utility accrued. In doing so, local schedulers use heuristics, which are a function of the end-end task constraints such as utility, remaining execution cost and task deadline to schedule local task segments. Further, decomposing the end-end task properties using techniques such as deadline slicing and TUF scaling to create segment deadlines and segment TUFs has shown an improved performance in the UA schedulers [Wu et al., 2005]. We create workloads that explore combinations of local and distributed task patterns in order to evaluate how decomposition of task properties influences the global (end-end) optimality of DASA-ND and LBESA in different scenarios.

7.1 Research Questions for Distributed Workloads

Our experimental analysis for distributed workloads, primarily answers the following two research questions.

- How do different load distribution patterns influence the global optimality of DASA-ND & LBESA for periodic tasks in a distributed system?

The first set of periodic workloads determines the effect of load distribution on the optimality of DASA-ND and LBESA. We simplify the task model with equal task segments and same periods for the 4 tasks. We only experiment with the deterministic cost model. Task costs are derived from a lognormal probability distribution function with a certain expected value and variance. Since we intend
to explore a wide range of system loads, we use a high variance. We experiment with equal task utilities, arithmetic sequence and geometric sequence utilities as in the single-node case. We also assume that task deadlines are equal to their periods.

Deadline decomposition for distributed tasks proportionately distributes the total slack to all the segments of the distributed task, thus giving a fair chance for all the segments to complete within the task deadline. Such methods can increase the likelihood of completing end-end distributed tasks, thus improving the system-wide utility accrued. We use SLEQF (see Chapter 2.3.1 for details), a deadline decomposition technique and determine how it improves the global optimality of DASA-ND and LBESA for different workloads in this section.

We explore three different task sets. One set, where all tasks have equal number of task segments. The other two sets have varying number of task segments. Of them, one has tasks with less number of segments having lower utility values than the tasks with more number of segments. The other has tasks with more number of segments having lower utility values than the tasks with less number of segments.

- **How does additional local-load influence the global optimality of DASA-ND & LBESA for periodic tasks? Can decomposition techniques improve the optimality for such workloads?**

The second set of distributed workloads creates a combination of local and distributed tasks in the system. For simplicity, we create a fixed local load of 0.4 on each node. All local tasks have the same cost value. Distributed tasks contribute to the system load. Only the distributed task costs are varied across the simulation runs. Each local task has a lower utility than the end-end distributed tasks. It has being shown in [Wu et al., 2005] that in situations where there is additional local-load on system nodes, applying decomposition techniques to
distributed task properties, improves the chance of completing the distributed tasks, thus allowing UA schedulers to accumulate a higher system (global) utility.

In presence of additional local load on the system nodes, the local UA scheduler tend to favor the local tasks than the segments of the distributed tasks, since local tasks have earlier deadlines than the end-end distributed tasks, even if they have lower PUD values. TUF scaling increases the height of the TUF of the distributed tasks by a factor relative to that of local tasks and thus aids distributed tasks to complete as against the local tasks. We determine if STEPS (see Chapter 2.3.1 for details), a TUF scaling technique improves the performance of DASA-ND and LBESA in completing the distributed tasks in the presence of local-load. Deadline slicing for distributed tasks proportionately distributes the total slack to all the segments of the distributed task, thus giving a fair chance for all the segments to complete within the task deadline. We determine if SLEQF (see Chapter 2.3.1 for details), a deadline decomposition technique improves the global optimality of DASA-ND and LBESA for different workloads in this section.

For every workload, we experiment initially without using the corresponding decomposition technique. Subsequently, we repeat the same experiment with SLEQF, and STEPS for each of the workloads in the following sections.

7.2 Load Distribution

We mainly consider two important load distribution patterns in our experiments for the periodic workloads. In the balanced case, we distribute load equally across all the 3 nodes. In unbalanced or biased case, we create a single bottleneck node, with heavy load, and the remaining nodes are lightly loaded. Figure 7-1 & Figure 7-2 depict the actual load

---

12 Without deadline decomposition, uses end-end task deadline. Our experiments use the Ultimate TUF strategy. We do not accept this as the appropriate method, but rather the end-end deadline for sub-segments should be derived as depicted in the example scribed in Chapter 2.3.1.1. The OMNET++ simulation model supports only Ultimate TUF.
values we have created on the 3 nodes for the balanced and unbalanced cases respectively. The system load in the balanced case is in the range [0.0 to 0.33] and [0.0 to 0.5] in case of unbalanced distribution.

Figure 7-1 : Balanced Load Distribution, Here 33% multiplied by the number of nodes is the actual load on each node; hence, the load on these nodes is 100%. This is called the system load.

Figure 7-2 : Unbalanced (Biased) Load Distribution, The load on Node 1 is 150%, which happens to be the maximum system load.

The total task segments, their costs and their frequency of arrival on a particular node, determine the load on that node. To simplify things, we assume that all periodic task segments in the first set of experiments have equal costs and task periods. Thus, in order to create a balanced load scenario, we assign an equal number of tasks segments to each node; unequal number of task segments to each node creates unbalanced load distribution. Even though assigning a predefined set of task segments to each node
determines the load distribution pattern, the total ordering of all the segments on different
nodes in every segment-node mapping significantly influences the end outcome. For
instance, consider a system of 3 nodes and 3 tasks, each with 3 segments. Even though
each node has being assigned 3 segments of the total 9, if the 3 segments on Node 1 are
the initial segments of each of the 3 tasks, chances of completing all the 3 DTs are
meager. Instead, if the 3 initial segments of each of the 3 tasks are assigned to 3 different
nodes, there is a higher likelihood of completing all the 3. Hence, in order to explore
different mappings, we randomly vary the segment-node mapping in each of the 100
simulation runs.

In order to simplify the system setup, we perform experiments with only 3 nodes and 4
tasks with different segment patterns for both the periodic and aperiodic task model.
Unlike in the single-node case, here, we emphasize exploring different orderings of
segment-node mappings to create balanced and unbalanced load distribution patterns.
Every experiment is composed of 100 simulation runs, each executed for a predefined
time. This time is equal to the hyper-period of the task set in case of periodic workloads.
Additional parameters such as segment-node mapping, overall node distribution, and
system load vector (see Chapter 5 for definition) characterize a distributed system
workload. We use probability distribution functions that sample distributed and local task
costs to produce a wide range of system loads from 0.2 to 1.0. In the case of balanced
load distribution, the maximum system load is 1.0, but can have a value > 1.0 on one of
the nodes in case of unbalanced load distribution. Across the 100 simulation runs for a
workload, we not only vary the task costs (as in the single-node case) but also vary the
segment to node assignments (see Chapter 7.2 for details). We have not included any
resource dependencies nor experimented with non-rectangular utility functions for tasks
at this point.
7.3 Effect of Load Distribution on Optimality

7.3.1 Periodic Task Model

Our aim is to create workloads that will allow us to not only to determine how different task utilities and task deadlines affect the optimality of UA schedulers in a distributed system, but in addition determine how different load distribution (balanced and unbalanced) patterns across the system nodes influence the overall global (end-end system-wide) optimality of DASA-ND and LBESA. This section uses the periodic task model to evaluate the optimality of DASA-ND and LBESA.

7.3.1.1 EQUAL TASK UTILITIES, EQUAL NUMBER OF SEGMENTS AMONG TASKS

Table 7-1 summarizes the task properties for distributed system workloads with equal task utilities. We simulate each workload with balanced and unbalanced load distribution among the 3 nodes and compare the results for both DASA-ND and LBESA.

<table>
<thead>
<tr>
<th>W</th>
<th>$T_{period}$</th>
<th>$T_{utility}$</th>
<th>UT</th>
<th>$T_{seg}$</th>
<th>$S_{SL}$</th>
<th>LD</th>
</tr>
</thead>
<tbody>
<tr>
<td>DW-1</td>
<td>[4, 4, 4, 4]</td>
<td>[100, 100, 100, 100]</td>
<td>E</td>
<td>[3, 3, 3]</td>
<td>[0.33, 0.33, 0.33]</td>
<td>B</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>[0.25, 0.50, 0.25]</td>
<td>UB</td>
</tr>
<tr>
<td>DW-2</td>
<td>[4, 4, 4, 4]</td>
<td>[100, 100, 100, 100]</td>
<td>E</td>
<td>[2, 2, 4, 4]</td>
<td>[0.33, 0.33, 0.33]</td>
<td>B</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>[0.25, 0.50, 0.25]</td>
<td>UB</td>
</tr>
</tbody>
</table>

B: Balanced Load, UB: Unbalanced Load, LU: Low Utility, HU: High Utility, LD: Load Distribution

Table 7-1: Distributed Periodic Tasks with Equal Utilities, Equal deadlines and controlled Load Distribution, either balanced or unbalanced

We vary the system load across the 100 runs and determine how DASA-ND and LBESA perform at different system loads and different segment to node mappings. We present the AUR distribution of DASA-ND and LBESA at varying system loads for different segment to node mappings in comparison with the optimal scheduler. We consider the difference in the AUR values of the optimal and UA schedulers, DASA-ND and LBESA. We term this difference as the “Delta AUR”. We plot the delta values against the
different system loads. A delta AUR of zero implies UA schedulers are optimal. A higher difference implies sub-optimality of DASA-ND and LBESA.

Figure 7-3 & Figure 7-4 show the delta AUR distribution for different systems loads for DASA-ND and LBESA respectively for DW-1. The filled ellipse shaped icon [●] indicates the delta AUR values for DASA-ND at different system loads for balanced load distribution. The unfilled ellipse shaped icon [○] indicates the delta AUR values for DASA-ND at different system loads for unbalanced load distribution. The filled triangle shaped icon [▲] indicates the delta AUR values for LBESA at different system loads for balanced load distribution. The unfilled triangle shaped icon [▲] indicates the delta AUR values for LBESA at different system loads for unbalanced load distribution. We compare the performance of both DASA-ND and LBESA for balanced and unbalanced load distributions.

For system underload in the range [0.0 to 0.6], both DASA-ND and LBESA are optimal. Unlike in the single-node scenario, both DASA-ND and LBESA produce sub-optimal schedules for distributed tasks sets with equal task utilities at higher system loads. Beyond a system load of 0.68, DASA-ND degrades more than LBESA for periodic workloads containing equal length task segments, deadlines, and equal number of task divisions. In the case of unbalanced load distribution, the degradation happens much earlier at a system load of 0.65. For system loads in the range 1.2 to 1.4, DASA-ND shows a huge difference of 0.75 in the delta AUR values. Since, there are 4 tasks with equal utilities, [100, 100, 100, 100], a delta AUR of 0.75 indicates that only one of the 4 tasks is completed, whereas the optimal scheduler accommodates all the 4 tasks. LBESA comparatively shows better performance than DASA-ND for such workloads.
Figure 7-3: Load vs Delta AUR, DASA-ND, DW-1 (Table 7-1), Equal Utilities, Equal deadlines and Equal cost segments for all tasks

Scenario D1:
We investigate a specific case, where DASA-ND creates sub-optimal schedules for distributed workload DW-1 as shown in Figure 7-5. The top segment describes the tasks’ properties. There are 4 periodic tasks within a hyper-period of 4. A DT is represented by one or more horizontal rectangular bars, each representing a segment of the task. To the left of the horizontal bars, we indicate in order the task number, segment number and the node on which this segment executes. A separate legend to the right of the tasks’ properties segment indicates the utility values for each task. The other conventions remain the same as in the Chapter 6.

All tasks in DW-1 have equal deadlines, equal costs and utilities. Hence, at each local node, all arriving tasks have same PUD. The deciding factor is the order in which the equal deadlines, equal PUD task segments are selected to form the feasible set at each local node. In this specific case, for DASA-ND, N1 (Node 1) selects [T3, S1] as against [T4, S1], since it is higher in the feasible set ordering. DASA-ND is unaware that the

---

13 This is a pathological case, system operators in distributed UA scheduling environments must avoid using equal cost, equal utility, and equal deadline scenarios
successive segment of T3 is next assigned to N2, which has an influx of other task segments around the same time [T3, S1] completes on N1.

Figure 7-4: Load vs Delta AUR, LBESA, DW-1(Table 7-1), Equal Utilities, Equal deadlines and Equal cost segments for all tasks

On the other hand, LBESA considers tasks T4 and T3 both are feasible and chooses to schedule T4 prior to T3. Fortunately, the segment-node mapping for successive segment of T4 helps LBESA to complete T4. Hence, for such workloads, segment-node mapping play an important role than the actual heuristics used in the node-local schedulers. In most cases, UA schedulers are bound to lose to the optimal scheduler, which has a system-wide view of the task characteristics including the all the segment-node mappings.

As in the single-node case, in order to provide an accurate estimate of DASA-ND’s and LBESA’s optimality we consider the cumulative distribution function (c.d.f) and probability density functions (also the histogram) of all the delta AUR values for both the balanced and biased load distribution scenarios.
Figure 7-5: Scenario D1 - LBESA is optimal, DASA-ND produces sub-optimal results for DW-1, Equal Utilities, Equal deadlines and Equal cost segments for all tasks
Figure 7-6: C.D.F and HISTOGRAM for DW-1, Balanced Load, Equal Utilities, Equal deadlines and Equal cost segments for all tasks

Figure 7-7: C.D.F and HISTOGRAM for DW-1, Unbalanced Load, Equal Utilities, Equal deadlines and Equal cost segments for all tasks
Figure 7-8: C.D.F and HISTOGRAM for DW-1, SLEQF, and **Balanced Load**, Equal Utilities, Equal deadlines and Equal cost segments for all tasks

Figure 7-9: C.D.F and HISTOGRAM for DW-1, SLEQF, and **Unbalanced Load**, Equal Utilities, Equal deadlines and Equal cost segments for all tasks
Figure 7-6 shows the C.D.F and HISTOGRAM for DASA-ND and LBESA in the balanced case for DW-1. For DASA-ND and LBESA, there is 0.23 and 0.21 likelihood that they are optimal with balanced load distribution across the 3 nodes. With equal task deadlines, equal number of segments in each task, equal task costs and utility values, the factor that creates the sub-optimal results for DASA-ND and LBESA is the segment to node mappings. DASA-ND produces sub-optimal schedules with a higher delta AUR than LBESA. LBESA, at minimum completes 2 of the 4 tasks, for all high system loads. Around 60% of the times, LBESA completes 3 of the 4 tasks, showing only a 0.25 delta AUR. Figure 7-7 shows the C.D.F and HISTOGRAM for DASA-ND and LBESA with unequal load distribution among the 3 nodes for DW-1. For DASA-ND and LBESA, it shows 0.14 and 0.12 likelihood they are optimal. Figure 7-8 & Figure 7-9 show the C.D.F and HISTOGRAM for DASA-ND and LBESA with SLEQF decomposition technique. As expected, the proportional deadline slicing substantially improves the performance of both DASA-ND and LBESA. There are a 0.5 and 0.6 probability that DASA-ND and LBESA are optimal respectively with SLEQF.

7.3.1.2 EQUAL TASK UTILITIES, VARYING NUMBER OF SEGMENTS AMONG TASKS

We next, vary the number of segments in each task. Some tasks have less number of segments than the remaining tasks. This should increase the probability of completing more tasks, since segments moving from one node to another less often.

Figure 7-10 & Figure 7-11 show the delta AUR distribution for different systems loads for DASA-ND and LBESA respectively for workload DW-2. Each plot compares the performance of DASA-ND and LBESA for balanced and unbalanced load distribution among the three nodes. As expected, both DASA-ND and LBESA show a higher probability of being optimal under a balanced distribution. However, LBESA degrades under a biased load distribution.

Figure 7-12 & Figure 7-13 show the C.D.F and HISTOGRAM for DASA-ND and LBESA respectively under both balanced and unbalanced load for workload DW-2. For DASA-
ND, there is a 0.54 probability that it is optimal with balanced load distribution and 0.34 probability that it is optimal with unbalanced load distribution. LBESA has a higher likelihood of 0.68 under balanced load distribution. It degrades by a huge margin, with only 0.35 probability of being optimal.

Figure 7-10: Load vs Delta AUR, DASA-ND DW-2, Equal Utilities, Equal deadlines and Equal cost segments for all tasks, Varying task segments among tasks

Figure 7-11: Load vs Delta AUR, LBESA, DW-2, Equal Utilities, Equal deadlines and Equal cost segments for all tasks, Varying task segments among tasks
Figure 7-12: C.D.F and HISTOGRAM for DASA-ND DW-2, Equal Utilities, Equal deadlines and Equal cost segments for all tasks, Varying task segments among tasks

Figure 7-13: C.D.F and HISTOGRAM for LBESA DW-2, Equal Utilities, Equal deadlines and Equal cost segments for all tasks, Varying task segments among tasks
We next use the deadline-slicing technique SLEQF (see Chapter 2.3.1 for details) along with LBESA to determine its effect on the global optimality of LBESA. The motive behind deadline slicing is to finish short segments earlier than the longer segments of other competing tasks on a local node. However, for workload DW-2, all segments have equal length and slack. Hence, the deadline decomposition results in segments closely competing to be part of the feasible set, since not all segments can be accommodated together in the local schedulers. Even then, deadline decomposition improves the optimality of LBESA. Figure 7-14 & Figure 7-15 show a 0.75 probability that DASA-ND and LBESA (under unbalanced load distribution) are optimal with deadline decomposition. We illustrate how decomposition aids LBESA with the following example scenario.

**Scenario D2:**

Figure 7-16 & Figure 7-17 shows a distributed workload with unbalanced load distribution and varying number of segments between tasks. N3 has higher load than N1 and N2. T1 and T2 have 2 segments each. T3 and T4 have 4 segments. All segments have equal costs. Figure 7-16 illustrates the optimal, DASA-ND and LBESA schedules for this workload. Both DASA-ND and LBESA fail to produce optimal results. DASA-ND at least completes 2 of the 4 tasks. In the case of LBESA, N2 prefers to schedule T4 against T2, not knowing that the odds of completing the remaining segments of T4 on N3 are meager (N3 is heavily loaded). Since both T4 and T2 have equal deadlines and are part of the feasible set, the order of breaking deadlines plays a major role. Figure 7-17 compares LBESA and LBESA with SLEQF. Due to deadline decomposition, both T4 and T2 have earlier deadlines. On N2, either T2 or T4 can be scheduled, but not both. LBESA has to eliminate one of the tasks between T4 and T2. Since both have equal utilities and costs, T4 being higher in the order (may be the task T4 arrived on N2 prior to T2) is removed from the feasible set. The end-end LBESA schedule now accumulates a higher overall utility of 50%.

*We conclude that for workloads with equal costs and equal deadlines it is highly difficult to predict the probability of DASA-ND and LBESA being optimal, since the order in*
which the deadline ties are broken primarily influence the optimality of the UA schedulers. This result applies to workloads with equal and varying number of segments among the tasks.

Figure 7-14: C.D.F and HISTOGRAM for DASA-ND with SLEQF, LBESA DW-2, Equal Utilities, Equal deadlines and Equal cost segments for all tasks, Varying task segments among tasks

Figure 7-15: C.D.F and HISTOGRAM for LBESA with SLEQF, LBESA DW-2, Equal Utilities, Equal deadlines and Equal cost segments for all tasks, Varying task segments among tasks
Figure 7-16: Scenario D2 - DASA-ND and LBESA produces sub-optimal results for DW-2, Equal Utilities, Equal deadlines, Equal cost segments, and Varying number segments among the tasks.
Figure 7-17: Scenario D2 - LBESA compared with LBESA with SLEQF
7.3.1.3 **GEOMETRIC TASK UTILITIES, EQUAL NUMBER OF SEGMENTS**

Table 7-2 shows the details of the workloads that have task utilities in geometric sequence. We previously concluded in the single-node case (see Chapter 6) that a high CR geometric utility sequence tends to show optimal results, whereas low CR utility values do not. We experiment with both types of sequences, to determine their effect on the optimality of UA schedulers for distributed task sets. We use equal task costs and deadlines as in the previous workloads. Both the utility and deadline values remain fixed for each workload across the 100 simulation runs. Workload DW-3 and DW-4 has an equal number of segments for all tasks. Workload DW-5 and DW-6 have varying number of segments. DW-5 has tasks with less number of segments having lower utility values than tasks with more number of segments. DW-6 has a reverse relationship between the task segments and utility values of the tasks.

<table>
<thead>
<tr>
<th>W</th>
<th>Tperiod</th>
<th>Tutility</th>
<th>UT</th>
<th>Utility → Segments</th>
<th>Tseg</th>
<th>S_SL</th>
<th>LD</th>
</tr>
</thead>
<tbody>
<tr>
<td>DW-3</td>
<td>[4, 4, 4, 4]</td>
<td>[2, 4, 8, 16]</td>
<td>G(2)</td>
<td>N/A</td>
<td>[3, 3, 3, 3]</td>
<td>[0.33,0.33,0.33]</td>
<td>[0.25,0.50,0.25]</td>
</tr>
<tr>
<td>DW-4</td>
<td>[4, 4, 4, 4]</td>
<td>[1, 10, 100, 1000]</td>
<td>G(10)</td>
<td>N/A</td>
<td>[3, 3, 3, 3]</td>
<td>[0.33,0.33,0.33]</td>
<td>[0.25,0.50,0.25]</td>
</tr>
<tr>
<td>DW-5</td>
<td>[4, 4, 4, 4]</td>
<td>[2, 4, 8, 16]</td>
<td>G(2)</td>
<td>Small No of Segs → LU</td>
<td>Large No of Segs → HU</td>
<td>[2, 2, 4, 4]</td>
<td>[0.33,0.33,0.33]</td>
</tr>
<tr>
<td>DW-6</td>
<td>[4, 4, 4, 4]</td>
<td>[2, 4, 8, 16]</td>
<td>G(2)</td>
<td>Small No of Segs → HU</td>
<td>Large No of Segs → LU</td>
<td>[4, 4, 2, 2]</td>
<td>[0.33,0.33,0.33]</td>
</tr>
</tbody>
</table>

Figure 7-18 & Figure 7-19 shows the delta AUR distribution for different systems loads for DASA-ND and LBESA respectively for DW-3. Each plot compares the performance of DASA-ND and LBESA with balanced and unbalanced load distribution among the three.
nodes. For system loads in the range [0.0 to 0.6], both DASA-ND and LBESA are optimal. As in the single-node scenario, both DASA-ND and LBESA produce sub-optimal schedules for distributed tasks sets with low CR geometrics sequence utilities at higher system loads. As expected, unbalanced load distribution degrades the performance of both DASA-ND and LBESA more than in the balanced case. With equal deadlines, task segments, and equal task costs, utility value is the main factor in deciding the ordering of tasks at the local schedulers.

DASA-ND fares better than LBESA by at least completing both the higher utility tasks at higher system loads. The maximum delta AUR for DASA-ND is 0.33 indicating that at minimum DASA-ND gains 22 of the total expected utility of 30 for DW-3. For LBESA the maximum delta AUR reaches as high as 0.53 indicating that at a minimum LBESA can ensure to accumulate 16 of the expected global utility of 30 for DW-3.

Figure 7-18: Load vs Delta AUR, DASA-ND DW-3 (Table 7-2), Geometric Utilities, CR = 2, Equal deadlines and Equal cost segments for all tasks, Equal task segments among tasks

Scenario D3:

We explain a scenario for DW-3, which forces both DASA-ND and LBESA to produce sub-optimal results with biased load distribution. N3 is heavily loaded compared to N1 and N2. The optimal scheduler accumulates a total AUR of 93% by scheduling 3 of the 4
tasks with higher utilities. Both, DASA-ND and LBESA schedule based on the PUD and deadline heuristic and accumulate 87% and 69% respectively. The node-local schedulers use end-end task deadlines for the segments. Since all segments have equal deadlines and costs, DASA-ND orders the high utility tasks ahead of the lower utility ones. DASA-ND ensures to complete tasks in the order of their importance. However, LBESA does not use the PUD order to break ties between equal deadline tasks in the feasible set and loses to DASA-ND for such workloads. The local UA schedulers have no global knowledge and do their best as per the heuristics. The earlier task segments have more slack than the later segments. Hence, the local schedulers find the earlier segments of different tasks feasible from their perspective. They tend to delay lower utility tasks and complete them late in the total order.

Figure 7-19: Load vs Delta AUR, LBESA DW-3(Table 7-2), Geometric Utilities, CR = 2, Equal deadlines and Equal cost segments for all tasks, Equal task segments among tasks
Figure 7-20: Scenario - D3, Both DASA-ND, and LBEA are sub-optimal for workload DW-3, Geometric Utilities, CR = 2, Equal deadlines and Equal cost segments for all tasks, Equal task segments among tasks
Figure 7-21 & Figure 7-22 show the C.D.F and HISTOGRAM for DASA-ND’s delta AUR values in the balanced and unbalanced load scenarios for DW-3. For DASA-ND, there is a 0.63 probability that it is optimal with balanced load distribution and 0.61 probability that it is optimal with unbalanced load distribution. Since both LBESA and DASA-ND ensure to accumulate at least 16 of the expected 30 for DW-3, it is evident that the highest utility task has a 100% likelihood of completion. Figure 7-23 & Figure 7-24 show the C.D.F and HISTOGRAM for LBESA’s delta AUR values in balanced and unbalanced load scenarios for DW-3. Even though, LBESA shows 0.63 probability of being optimal, it shows cases of higher delta AUR than DASA-ND.

Figure 7-25 & Figure 7-26 shows the C.D.F and HISTOGRAM for DASA-ND & LBESA’s delta AUR values for DW-3 with the SLEQF deadline decomposition technique. LBESA shows a marked improvement in its global optimality. They indicate >0.7 probability that LBESA is optimal both under balanced and biased load distribution.
Figure 7-23: C.D.F and HISTOGRAM for LBESA, Balanced Load DW-3, Geometric Utilities, CR = 2,
Equal deadlines and Equal cost segments for all tasks, Equal task segments among tasks

Figure 7-24: C.D.F and HISTOGRAM for LBESA, Unbalanced Load, DW-3, Geometric Utilities, CR = 2,
Equal deadlines and Equal cost segments for all tasks, Equal task segments among tasks

Figure 7-25: C.D.F and HISTOGRAM for DASA-ND with SLEQF, DW-3, Geometric Utilities, CR = 2
Equal deadlines and Equal cost segments for all tasks, Equal task segments among tasks
The next workload DW-4 has a higher CR than DW-3. Figure 7-27 & Figure 7-28 shows the delta AUR distribution for different systems loads for DASA-ND and LBESA respectively for DW-4. The corresponding c.d.f and histogram for DASA-ND in Figure 7-29 & Figure 7-30 indicate 1.0 probability that DASA-ND is optimal with less than 0.15 difference in the AUR values of optimal and DASA-ND. Figure 7-31 & Figure 7-32 show the corresponding c.d.f and histogram for LBESA. For one particular instance, LBESA shows a huge delta AUR of 0.9. Apart from this deviation, LBESA still shows a 1.0 probability that it is optimal with less than 0.15 differences in AUR. We conclude that both DASA-ND and LBESA remain near optimal for periodic distributed tasks with geometric sequence utilities and higher constant ratio under SLEQF.
Figure 7-27: Load vs Delta AUR, DASA-ND DW-4 (Table 7-2), Geometric Utilities, CR =10, Equal deadlines and Equal cost segments for all tasks, Unequal task segments among tasks.

Figure 7-28: Load vs Delta AUR, LBESA DW-4 (Table 2), Geometric Utilities, CR =10, Equal deadlines and Equal cost segments for all tasks, Unequal task segments among tasks.
Figure 7-29: C.D.F and HISTOGRAM for DASA-ND, Balanced Load, and DW-4, Geometric Utilities, CR =10, Equal deadlines and Equal cost segments for all tasks, Unequal task segments among tasks

Figure 7-30: C.D.F and HISTOGRAM for DASA-ND, Unbalanced Load, and DW-4, Geometric Utilities, CR =10, Equal deadlines and Equal cost segments for all tasks, Unequal task segments among tasks

Figure 7-31: C.D.F and HISTOGRAM for LBESA, Balanced Load, and DW-4, Geometric Utilities, CR =10, Equal deadlines and Equal cost segments for all tasks, Equal task segments among tasks
Figure 7-32: C.D.F and HISTOGRAM for LBESA, Unbalanced Load, and DW-4, Geometric Utilities, CR = 10, Equal deadlines and Equal cost segments for all tasks, Equal task segments among tasks

Figure 7-33: C.D.F and HISTOGRAM with SLEQF, Balanced Load, and DW-4, Geometric Utilities, CR = 10, Equal deadlines and Equal cost segments for all tasks, Equal task segments among tasks
The remaining two workloads in this sub-section have task sets that have a defined relationship between the task utility values and number of segments in each task. Both of these workloads have tasks with varying number of segments.

**Figure 7-35 & Figure 7-36** show the delta AUR distribution for different systems loads for DASA-ND and LBESA for DW-5. **Figure 7-43 & Figure 7-44** show the delta AUR distribution for different systems loads for DASA-ND and LBESA for DW-6.

Comparing the results from DASA-ND and LBESA for the two workloads, it shows that both the UA schedulers have a higher probability of being optimal for DW-5 than DW-6. This is true with both balanced and unbalanced load distribution. Hence, both UA schedulers are closer to optimal when tasks with smaller number of segments have low
utility values than the tasks with more number of segments. The reason is that the UA schedulers have no knowledge of the number of segments a task has. The heuristics do not intend to complete tasks with small or large number of segments. They only aim to complete as many as high utility tasks as possible. Hence, in both DW-5 and DW-6, the local schedulers finish high utility segments prior to the low utility segments. Thus, the low utility segments are delayed on each local node. The striking difference between the two workloads is that DW-6 has more segments for tasks with low utility. Having already delayed the initial segments of tasks in DW-6, on different nodes, the latter segments become infeasible. However, in DW-5, the low utility tasks have less number of segments. Even though they are delayed, due to fewer segments, low utility tasks in DW-5 have a higher likelihood of completing.

Scenario D4:
Consider the example case shown in Figure 7-37 & Figure 7-38. They compare optimal and UA schedules for DW-5 and DW-6 for the same task properties, except that in DW-5, low utility tasks (T1, T2) have smaller number of segments than the same tasks in DW-6. In Figure 7-37, both DASA-ND and LBESA are optimal, complete 3 out of 4 tasks, and accumulate 73% AUR. Even though there is an equal load distribution among the 3 nodes, high utility tasks T3 and T4 compete to finish around the same time on N2. Eventually, T4 being the higher PUD task, is scheduled and thus completes. T3 becomes infeasible. In Figure 7-38, we see that higher utility tasks have small number of segments and take less time to complete. Hence, the optimal scheduler is able to accumulate a higher AUR of 87%. It is even able to accommodate one of the lower utility tasks. However, DASA-ND and LBESA delay the low utility tasks, initially. Though they complete the higher utility tasks, they fail to accommodate the lower utility ones at a later stage.
Figure 7-35: Load vs Delta AUR, DASA-ND DW-5, Geometric Utilities, CR = 2, Equal deadlines and Equal cost segments for all tasks, Unequal task segments among tasks. Small No of Segs \( \rightarrow \) LU and Large No of Segs \( \rightarrow \) HU

Figure 7-36: Load vs Delta AUR, LBESA DW-5, Geometric Utilities, CR = 2, Equal deadlines and Equal cost segments for all tasks, Unequal task segments among tasks, Small No of Segs \( \rightarrow \) LU and Large No of Segs \( \rightarrow \) HU

138
Figure 7-37: Scenario D4 - DASA-ND and LBESA are optimal for DW-5, Geometric Utilities, CR = 2, Equal deadlines and Equal cost segments for all tasks, Unequal task segments among tasks, Small No of Segs \(\rightarrow\) LU and Large No of Segs \(\rightarrow\) HU
Figure 7-38: Scenario D4 - DASA-ND and LBESA are sub-optimal for DW-6, Geometric Utilities, CR = 2, Equal deadlines and Equal cost segments for all tasks, Unequal task segments among tasks, Small No of Segs → HU and Large No of Segs → LU
The c.d.f for DASA-ND and LBESA for DW-5 and DW-6 provide a more accurate estimate of their optimality. Figure 7-39 & Figure 7-40 indicate that DASA-ND and LBESA have a 0.87 and 0.72 probability that their optimal with balanced load distribution for workload DW-5. Figure 7-45 & Figure 7-46 indicate that DASA-ND and LBESA have a 0.41 probability that they are optimal with balanced load distribution for workload DW-6. DASA-ND fares better than LBESA in DW-5. However, with SLEQF, both DASA-ND and LBESA improve their optimality and show a 0.62 probability of being optimal for DW-6 respectively (see Figure 7-41 & Figure 7-47).

![Figure 7-39: C.D.F and HISTOGRAM, Balanced Load for DW-5, Geometric Utilities, CR=2, Equal deadlines and Equal cost segments for all tasks, Unequal task segments among tasks, Small No of Segs \(\Rightarrow\) LU and Large No of Segs \(\Rightarrow\) HU](image-url)
Figure 7-40: C.D.F and HISTOGRAM, Unbalanced Load for DW-5, Geometric Utilities, CR=2, Equal deadlines and Equal cost segments for all tasks, Unequal task segments among tasks, Small No of Segs Æ LU and Large No of Segs Æ HU

Figure 7-41: C.D.F and HISTOGRAM, with SLEQF for Balanced Load for DW-5, Geometric Utilities, CR=2, Equal deadlines and Equal cost segments for all tasks, Unequal task segments among tasks, Small No of Segs Æ LU and Large No of Segs Æ HU
Figure 7-42: C.D.F and HISTOGRAM, with SLEQF for Unbalanced Load for DW-5, Geometric Utilities, CR=2, Equal deadlines and Equal cost segments for all tasks, Unequal task segments among tasks, Small No of Segs $\rightarrow$ LU and Large No of Segs $\rightarrow$ HU

Figure 7-43: Load vs Delta AUR, DASA-ND DW-6, Geometric Utilities, CR = 2, Equal deadlines and Equal cost segments for all tasks, Unequal task segments among tasks, Small No of Segs $\rightarrow$ HU and Large No of Segs $\rightarrow$ LU
Figure 7-44: Load vs Delta AUR, LBESA DW-6, Geometric Utilities, CR = 2, Equal deadlines and Equal cost segments for all tasks, Unequal task segments among tasks, Small No of Segs $\rightarrow$ HU and Large No of Segs $\rightarrow$ LU

Figure 7-45: C.D.F and HISTOGRAM, Balanced Load for DW-6, Geometric Utilities, CR = 2, Equal deadlines and Equal cost segments for all tasks, Unequal task segments among tasks, Small No of Segs $\rightarrow$ HU and Large No of Segs $\rightarrow$ LU
Figure 7-46: C.D.F and HISTOGRAM, Unbalanced Load for DW-6, Geometric Utilities, CR = 2, Equal deadlines and Equal cost segments for all tasks, Unequal task segments among tasks, Small No of Segs $\Rightarrow$ HU and Large No of Segs $\Rightarrow$ LU

Figure 7-47: C.D.F and HISTOGRAM, with SLEQF for Balanced Load for DW-6, Geometric Utilities, CR = 2, Equal deadlines and Equal cost segments for all tasks, Unequal task segments among tasks, Small No of Segs $\Rightarrow$ HU and Large No of Segs $\Rightarrow$ LU
Figure 7-48: C.D.F and HISTOGRAM, with SLEQF for Unbalanced Load for DW-6, Geometric Utilities, CR = 2, Equal deadlines and Equal cost segments for all tasks, Unequal task segments among tasks, Small No of Segs → HU and Large No of Segs → LU

7.3.1.5 ARITHMETIC TASK UTILITIES, EQUAL NUMBER OF SEGMENTS

Table 7-3 shows the details of the workloads that have task utilities in arithmetic sequence. We previously concluded in the single-node case (Chapter 6) that different CD (constant difference) arithmetic utility sequence shows no variation in the end outcome. Hence, we only experiment with a CD of 100. We use equal task costs and deadlines as in the previous workloads in this sub-section. Both the utility and deadline values remain fixed for each workload across the 100 simulation runs. Workload DW-7 has an equal number of segments for all tasks. Workload DW-8 and DW-9 have varying number of segments. DW-8 has tasks with less number of segments having lower utility values than tasks with more number of segments. DW-9 has a reverse relationship between the task segments and utility values of the tasks.
Figure 7-49 & Figure 7-50 show the delta AUR distribution for different systems loads for DASA-ND and LBESA respectively for DW-7. Each plot compares the performance of DASA-ND and LBESA with balanced and biased load distribution among the three nodes.

<table>
<thead>
<tr>
<th>$W$</th>
<th>$T_{period}$</th>
<th>$T_{utility}$</th>
<th>$UT$</th>
<th>Utility $\rightarrow$ Segments</th>
<th>$T_{seg}$</th>
<th>$S_{SL}$</th>
<th>$LD$</th>
</tr>
</thead>
<tbody>
<tr>
<td>DW-7</td>
<td>[4, 4, 4, 4]</td>
<td>[100, 200, 300, 400]</td>
<td>A(100)</td>
<td>N/A</td>
<td>[3, 3, 3]</td>
<td>[0.33, 0.33, 0.33]</td>
<td>B</td>
</tr>
<tr>
<td>DW-8</td>
<td>[4, 4, 4, 4]</td>
<td>[100, 200, 300, 400]</td>
<td>A(100)</td>
<td>Small No of Segs $\rightarrow$ LU  Large No of Segs $\rightarrow$ HU</td>
<td>[2, 2, 4, 4]</td>
<td>[0.33, 0.33, 0.33]</td>
<td>B</td>
</tr>
<tr>
<td>DW-9</td>
<td>[4, 4, 4, 4]</td>
<td>[100, 200, 300, 400]</td>
<td>A(100)</td>
<td>Small No of Segs $\rightarrow$ HU  Large No of Segs $\rightarrow$ LU</td>
<td>[4, 4, 2, 2]</td>
<td>[0.33, 0.33, 0.33]</td>
<td>B</td>
</tr>
</tbody>
</table>

B: Balanced Load, UB: Unbalanced Load, LU: Low Utility, HU: High Utility, LD: Load Distribution

Table 7-3: Distributed Periodic Tasks with Arithmetic Sequence Utilities and controlled Load Distribution with either balanced or unbalanced

For system loads in the range [0.0 to 0.6], both DASA-ND and LBESA are optimal. As in the single-node scenario, both DASA-ND and LBESA produce sub-optimal schedules for distributed tasks sets with arithmetic sequence utilities at higher system loads. As expected, unbalanced load distribution degrades the performance of both DASA-ND and LBESA far more than in the balanced case. DASA-ND fares better than LBESA with a higher likelihood of completing both the higher utility tasks at higher system loads, both under balanced and unbalanced load distribution. The maximum delta AUR for both DASA-ND and LBESA is 0.4 indicating that at minimum, both DASA-ND and LBESA accumulate a utility of 600 out of the expected 1000 for workload DW-7.
Figure 7-49: Load vs Delta AUR, DASA-ND, DW-7 (Table 7-3), and Arithmetic Utilities, CD = 100, Equal deadlines and Equal cost segments for all tasks, Equal segments among tasks

Figure 7-50: Load vs Delta AUR, LBESA, DW-7 (Table 7-3), Arithmetic Utilities, CD = 100, Equal deadlines and Equal cost segments for all tasks, Equal segments among tasks
Scenario D5:
We discuss the scenario shown in Figure 7-51, which results in sub-optimal results for both DASA-ND and LBESA for workload DW-7. All the tasks have same number of segments with equal costs and deadlines. As discussed in the geometric task utility workloads, DASA-ND schedules locally available segments in the order of their PUD and delays the low utility segments. Similarly, LBESA creates deadline ordered feasible set with as many high PUD tasks as possible. With these heuristics, both DASA-ND and LBESA, meet fewer deadlines than the optimal for this workload. In this example, completing 3 of the lower utility tasks yields higher AUR than just finishing only one of the highest utility tasks. This happens because the tasks in DW-7 have utilities in an arithmetic sequence [100, 200, 300, and 400]. This situation could not happen for geometric sequence utilities such as [2, 4, 8, and 16], because the total utility from T1 to T3 is 14, which is less than the highest utility task T4 with a utility of 16. Local schedulers use only the locally available task information. Each node-local scheduler produces the best possible local ordering unlike the optimal scheduler that aims at the best possible global ordering.
Figure 7-51: Scenario D5: Both DASA-ND and LBESA are sub-optimal for DW-7, Arithmetic Utilities, Equal deadlines and Equal cost segments for all tasks,
Figure 7-52 & Figure 7-53 show the C.D.F and HISTOGRAM for DASA-ND’s delta AUR values in the balanced and unbalanced load scenarios for DW-7. For DASA-ND there is a 0.57 probability that it is optimal with balanced load distribution and 0.53 probability that it is optimal with unbalanced load distribution. With equal deadlines, task segments, and equal task costs, utility value is the main factor in deciding the ordering of tasks at the local scheduler. Since local DASA-ND schedulers are unaware of the end-end path taken by a task to complete, they only schedule tasks based on the locally available task deadlines and PUD. For close to 50 cases out of the 100, such locally optimal decisions ensure globally optimal results. Figure 7-54 & Figure 7-55 show the C.D.F and HISTOGRAM for LBESA’s delta AUR values in balanced and unbalanced load scenarios for DW-7. As in DASA-ND, LBESA shows a 0.57 probability that it is optimal with balanced load distribution and 0.53 probability that it is optimal with unbalanced load distribution. In addition, LBESA shows an instance where the delta AUR reaches as high as 0.4 under balanced load distribution among the 3 nodes.

Figure 7-56 & Figure 7-57 shows further improvement in the optimality of DASA-ND and LBESA respectively for DW-7 with the use of SLEQF technique irrespective of load distribution. DASA-ND and LBESA show a 0.65 probability that they are optimal with balanced load distribution and 0.63 probability that they are optimal with unbalanced load distribution.

Figure 7-52 : C.D.F and HISTOGRAM for DW-7, DASA-ND, And Balanced Load, Arithmetic Utilities, Equal deadlines and Equal cost segments for all tasks,
Figure 7-53: C.D.F and HISTOGRAM for DW-7, DASA-ND, Unbalanced Load, Arithmetic Utilities, Equal deadlines and Equal cost segments for all tasks.

Figure 7-54: C.D.F and HISTOGRAM for DW-7, LBESA Balanced Load, Arithmetic Utilities, Equal deadlines and Equal cost segments for all tasks.

Figure 7-55: C.D.F and HISTOGRAM for DW-7, LBESA Unbalanced Load, Arithmetic Utilities, Equal deadlines and Equal cost segments for all tasks.
Figure 7-56: C.D.F and HISTOGRAM for DW-7 with SLEQF and Balanced Load, Arithmetic Utilities, Equal deadlines and Equal cost segments for all tasks,

Figure 7-57: C.D.F and HISTOGRAM for DW-7 with SLEQF and Unbalanced Load, Arithmetic Utilities, Equal deadlines and Equal cost segments for all tasks,
7.3.1.6 ARITHMETIC TASK UTILITIES, VARYING NUMBER OF SEGMENTS

The remaining two workloads in this sub-section have task sets that have a defined relationship between the arithmetic utility values and the number of segments in each task. Figure 7-58 & Figure 7-59 show the delta AUR distribution for different systems loads for DASA-ND and LBESA for DW-8. Figure 7-70 & Figure 7-71 show the delta AUR distribution for different systems loads for DASA-ND and LBESA for DW-9.

Comparing the results for DASA-ND and LBESA for DW-8 and DW-9 it is clear that both the UA schedulers have a higher probability of being optimal for DW-8 than DW-9. Hence, irrespective of the utility type, UA schedulers perform better for workloads that have tasks with less number of segments having low utility than the tasks with more number of segments (All the tasks have equal deadlines).

LBESA performs worst with unequal load distribution among the nodes for both workloads DW-8 and DW-9. We further analyze the reasons for such degradation in the performance of LBESA.

Figure 7-58 : Load vs Delta AUR, DASA-ND, DW-8, Arithmetic Utilities, Equal deadlines and Equal cost segments for all tasks, varying task segments among tasks, Small No of Segs $\rightarrow$ LU and Large No of Segs $\rightarrow$ HU
Figure 7-59: Load vs Delta AUR, LBESA DW-8, Arithmetic Utilities, Equal deadlines and Equal cost segments for all tasks, varying task segments among tasks, Small No of Segs $\rightarrow$ LU and Large No of Segs $\rightarrow$ HU

Figure 7-60: C.D.F and HISTOGRAM for DW-8, DASA-ND, balanced Load, Arithmetic Utilities, Equal deadlines and Equal cost segments for all tasks, varying task segments among tasks, Small No of Segs $\rightarrow$ LU and Large No of Segs $\rightarrow$ HU
Figure 7-61: C.D.F and HISTOGRAM for DW-8, DASA-ND, and Unbalanced Load, Arithmetic Utilities, Equal deadlines and Equal cost segments for all tasks, varying task segments among tasks, Small No of Segs → LU and Large No of Segs → HU

Figure 7-62: C.D.F and HISTOGRAM for DW-8, LBESA Balanced Load, Arithmetic Utilities, Equal deadlines and Equal cost segments for all tasks, varying task segments among tasks, Small No of Segs → LU and Large No of Segs → HU

Figure 7-63: C.D.F and HISTOGRAM for DW-8 LBESA Unbalanced Load, Arithmetic Utilities, Equal deadlines and Equal cost segments for all tasks, varying task segments among tasks, Small No of Segs → LU and Large No of Segs → HU
The c.d.f for DASA-ND and LBESA for DW-5 and DW-6 provide a more complete analysis of their optimality. Figure 7-60 & Figure 7-62 indicate that DASA-ND and LBESA have a 0.87 and 0.72 probability that they are optimal under balanced load distribution for workload DW-8. Figure 7-72 & Figure 7-74 indicate that both DASA-ND and LBESA have a 0.40 probability that they are optimal under balanced load distribution for workload DW-9. Again, DASA-ND fares better than LBESA for DW-8 without any decomposition. However, with SLEQF, both DASA-ND and LBESA improve their performance and show a 0.72 and 0.63 probability of being optimal for DW-8 and DW-9 respectively (see Figure 7-64 & Figure 7-76).

Figure 7-64: C.D.F and HISTOGRAM for DW-8 with SLEQF and Balanced Load, Arithmetic Utilities, Equal deadlines and Equal cost segments for all tasks, varying task segments among tasks, Small No of Segs $\rightarrow$ LU and Large No of Segs $\rightarrow$ HU.
Scenario D6:
This example shows that DASA-ND is better than LBESA for workloads that have equal deadline tasks. Local DASA-ND schedulers always break ties among equal deadline tasks in the feasible set, in the order of the PUD. This method of breaking deadline ties is particularly important in a distributed task-set, since this ordering ensures that at least higher utility tasks are given more room and are more likely to complete in their end-end path. Local LBESA schedulers do not enforce this criterion among the tasks in the feasible set. From the perspective of the local node, all tasks are feasible and it does not matter, if either of them are scheduled earlier, since eventually it can complete all the tasks in the feasible set. This decision affects the global ordering since there are other factors, the local scheduler is unable to foresee. One such factor is the load on the remaining nodes, where the subsequent segments of the local tasks will execute in future.

In Figure 7-68, we describe a scenario that leads to highly sub-optimal results for LBESA. Consider the situation on N3. After scheduling T1, N3 remains idle, since there are no
other tasks to execute. When [T4, S2] arrives on N3, there are no other segments competing on that node. LBESA schedules [T4, S2]. Though we ensure that all tasks have equal costs, [T4, S2] is slightly longer than other tasks. Even before, [T4, S2] completes, [T3, S3] arrives. T4 has the highest PUD, due to its very low remaining execution time. Both T4 and T3 have equal deadlines in the feasible set. DASA-ND chooses [T4, S2]. LBESA does not enforce PUD heuristic to break deadline ties and chooses [T3, S2]. Eventually neither T3 nor T4 complete in LBESA. This ordering forces LBESA to accumulate an overall AUR of only 0.1.

We next determine whether deadline decomposition can improve the performance of LBESA for such workloads. Figure 7-66 & Figure 7-67 compare LBESA with and without the deadline decomposition in both the balanced and unbalanced load distribution. The filled triangle shaped icon [▲] indicates the delta AUR values for LBESA at different system loads. The cross-shaped icon [★] indicates the delta AUR values for LBESA at different system loads with the deadline decomposition technique SLEQF. Both show that SLEQF improves the optimality of LBESA by a huge margin. Figure 7-69 also illustrates this fact. In this example, LBESA is optimal after applying deadline decomposition to the task set. Deadline decomposition not only divides the total slack proportionally, but also renders some tasks infeasible at an early stage. In some cases, this might decrease the overall performance of LBESA, but in other cases, it might aid LBESA. If a task rendered infeasible at any earlier stage happens to be a longer task with a comparatively lower probability of completing, it can make room for many other shorter tasks, thus improving the chances of completing more number of tasks and the overall global utility accrued. In this case, T3, a high utility, but a low PUD task is eliminated on N2, since both T4 and T3 are not feasible together. This early elimination of T3 on N2 makes room for T2 to complete on N3 without contention with T3.
Figure 7-66: Load vs Delta AUR, BALANCED LOAD, LBESA, with SLEQF DW-8, Arithmetic Utilities, Equal deadlines and Equal cost segments for all tasks, varying task segments among tasks, *Small No of Segs → LU* and *Large No of Segs → HU*

Figure 7-67: Load vs Delta AUR, UNBALANCED LOAD, LBESA, with SLEQF DW-8, Arithmetic Utilities, Equal deadlines and Equal cost segments for all tasks, varying task segments among tasks, *Small No of Segs → LU* and *Large No of Segs → HU*
Figure 7-68: Scenario D6 - LBESA is highly sub-optimal for DW-8, Arithmetic Utilities, Equal deadlines and Equal cost segments for all tasks, varying task segments among tasks, Small No of Segs $\Rightarrow$ LU and Large No of Segs $\Rightarrow$ HU
Figure 7-69: Scenario D6: LBESA compared with LBESA with SLEQF
Figure 7-70: Load vs Delta AUR, DASA-ND DW-9, Arithmetic Utilities, Equal deadlines and Equal cost segments for all tasks, varying task segments among tasks, Small No of Segs ➔ HU and Large No of Segs ➔ LU

Figure 7-71: Load vs Delta AUR, LBESA DW-9, Arithmetic Utilities, Equal deadlines and Equal cost segments for all tasks, varying task segments among tasks, Small No of Segs ➔ HU and Large No of Segs ➔ LU
Figure 7-72: C.D.F and HISTOGRAM, DASA-ND, Balanced Load, DW-9, Arithmetic Utilities, Equal deadlines and Equal cost segments for all tasks, varying task segments among tasks, Small No of Segs ➔ HU and Large No of Segs ➔ LU

Figure 7-73: C.D.F and HISTOGRAM, DASA-ND, Unbalanced Load, DW-9, Arithmetic Utilities, Equal deadlines and Equal cost segments for all tasks, varying task segments among tasks, Small No of Segs ➔ HU and Large No of Segs ➔ LU

Figure 7-74: C.D.F and HISTOGRAM, LBESA Balanced Load, DW-9, Arithmetic Utilities, Equal deadlines and Equal cost segments for all tasks, varying task segments among tasks, Small No of Segs ➔ HU and Large No of Segs ➔ LU
Figure 7-75: C.D.F and HISTOGRAM, LBESA, Unbalanced Load, DW-9, Arithmetic Utilities, Equal deadlines and Equal cost segments for all tasks, varying task segments among tasks, Small No of Segs $\Rightarrow$ HU and Large No of Segs $\Rightarrow$ LU

Figure 7-76: C.D.F and HISTOGRAM, with SLEQF Balanced Load, DW-9, Arithmetic Utilities, Equal deadlines and Equal cost segments for all tasks, varying task segments among tasks, Small No of Segs $\Rightarrow$ HU and Large No of Segs $\Rightarrow$ LU
7.4 Effect of Local-Load on Optimality

Until now, we have used equal costs for all tasks in the system. The following workloads use varying task and segment costs. We still use equal task deadlines and experiment with equal, geometric, and arithmetic sequence utility pattern. To simplify the system model, we do not control the load distribution across nodes; instead assume that system load to vary between 0.0 & 1.0 \(^{14}\) (refer Chapter 5.1 for details). We create additional local load on each of the 3 nodes.

---

\(^{14}\) System load does not include the additional local-load on each node. The x-axis value in the plots indicates only the system load. The x-axis value of 0.6 on one node represents a load of 0.6* 3 = 180% total load contributed by the distributed tasks,
### 7.4.1 Periodic Task Model

#### 7.4.1.1 EQUAL TASK UTILITIES

Table 7-4 shows the workload properties for DW-10, which has equal utilities and varying segment costs for the 3 distributed tasks. We create an additional local-load of 40% on each of the 3 nodes.

<table>
<thead>
<tr>
<th>W</th>
<th>Sd</th>
<th>T\text{period}</th>
<th>T\text{utility}</th>
<th>UT</th>
<th>T\text{seg}</th>
<th>S\text{LL}</th>
</tr>
</thead>
<tbody>
<tr>
<td>DW-10</td>
<td>None/SLEQF/STEPS</td>
<td>[4, 4, 4]</td>
<td>[100, 100, 100, 100]</td>
<td>E</td>
<td>[2, 4, 2]</td>
<td>0.4</td>
</tr>
</tbody>
</table>

Table 7-4: Distributed Periodic Tasks with Equal Utilities, Uniform Distribution System Load, and Fixed Local Load

There is a slight change in the convention used in the plots. Each plot now compares UA scheduler’s performance, without decomposition along with the scheduler’s performance with the corresponding decomposition techniques. Figure 7-78 & Figure 7-79 show the delta AUR distribution for different systems loads for DASA-ND and LBESA respectively for DW-10. The filled ellipse shaped icon [●] indicates the delta AUR values for DASA-ND at different system loads without decomposition. The unfilled ellipse shaped icon [○] indicates the delta AUR values for DASA-ND at different system loads with STEPS decomposition. The filled triangle shaped icon [▲] indicates the delta AUR values for LBESA at different system loads without decomposition. The unfilled triangle shaped icon [▲] indicates the delta AUR values for LBESA at different system loads with SLEQF decomposition.

Both DASA-ND and LBESA’s degree of optimality improves with their corresponding decomposition techniques for the workload DW-10. The improvement is marginal for DASA-ND, but substantial for LBESA. Figure 7-80 show the C.D.F and HISTOGRAM for DASA-ND and LBESA without the decomposition techniques for workload DW-10. Figure 7-81 & Figure 7-82 show the C.D.F and HISTOGRAM for DASA-ND and LBESA respectively with the decomposition techniques STEPS and SLEQF. For DASA-ND, there is a 0.70 probability that it is optimal without decomposition and 0.72 probability that it is optimal with STEPS decomposition in the presence of additional local load. The
optimality is best with using deadline slicing. DASA-ND with SLEQF has the highest, a 0.8 probability of being optimal for DW-10. For LBESA there is a 0.20 probability that it is optimal without decomposition and 0.80 probability that it is optimal with SLEQF decomposition in the presence of additional local load. It performs worst with STEPS decomposition showing only a 0.2 probability of being optimal. (See Figure 7-82)

Figure 7-78: Load vs Delta AUR, DASA-ND, 40% Local Load, DW-10 (Table 7-4), equal utilities, unequal costs and task-node mappings derived from an uniform distribution

Figure 7-79: Load vs Delta AUR, LBESA, 40% Local Load, DW-10, equal utilities, unequal costs and task-node mappings derived from an uniform distribution
Figure 7-80: C.D.F and HISTOGRAM, DASA-ND and LBESA without Decomposition, 40% Local Load DW-10, equal utilities, unequal costs and task-node mappings derived from an uniform distribution.

Figure 7-81: C.D.F and HISTOGRAM, DASA-ND with DECOMPOSITION, 40% Local Load DW-10, equal utilities, Unequal costs and task-node mappings derived from a uniform distribution.
Scenario D7:

Figure 7-83 & Figure 7-84 discuss a specific example for workload DW-10 where deadline decomposition improves the optimality of LBESA substantially. The local tasks contribute to the local load and have lower utility than the distributed tasks. T4 on N1, T5 on N2, and T6 on N3 contribute to a local-load of 40%. There are 3 distributed tasks, with varying costs and segments. Without decomposition, the node-local LBESA schedules local tasks earlier than the distributed task segments, since all tasks have equal deadlines. The delay in scheduling earlier segments of the distributed task eventually makes them infeasible on the remaining nodes. DASA-ND schedules the distributed task segments, since they have higher PUD than the local-tasks in the equal deadline task sets. Hence, DASA-ND accumulates a overall AUR of 0.67 as compared to LBESA’s 0.58.

Figure 7-84 compares the LBESA schedules without and with decomposition techniques. On proportionally decomposing the distributed task deadline into segment deadlines, local LBESA schedulers, now order the distributed task segments prior to the local-tasks. For instance, on N2, LBESA schedules, low PUD, longer deadline task T5 is pushed further back in the order in preference to the distributed segments T2 and T3. Eventually
T5 is discarded in favor of a higher PUD, shorter deadline task T1. Since, distributed tasks have higher utility than the local-tasks, the system-wide AUR increases to 0.75, near to the optimal schedule of 0.79. 15

Hence, we conclude that deadline decomposition aids DASA-ND & LBESA and substantially improves their performance. STEPS (TUF scaling technique) only marginally improve the performance of DASA-ND for tasks with step-downward TUFs. It degrades LBESA’s performance. With deadline decomposition SLEQF, LBESA and DASA-ND have 0.8 probability that it is optimal for distributed workloads with equal task utilities and task deadlines even in the presence of additional local load.

15 The optimal scheduler does not distinguish between local and distributed tasks. It accumulates the best possible utility from all the available tasks.
Figure 7-83: Scenario D7 - Unequal task costs, Equal Utilities, DASA-ND & LBESA are sub-optimal for DW-10, equal utilities, Unequal costs and task-node mappings derived from an uniform distribution
Figure 7-84: Scenario D7 - LBESA compared with LBESA+SLEQF for DW-10
7.4.1.2 GEOMETRIC TASK UTILITIES

Table 7-5 shows the workload properties for DW-11, which has geometric sequence utilities and varying segment costs for the 3 distributed tasks. We create an additional local-load of 40% on each of the 3 nodes. All the local tasks have a fixed utility of 3 and fixed costs.

<table>
<thead>
<tr>
<th>W</th>
<th>S_d</th>
<th>T_period</th>
<th>T_utility</th>
<th>UT</th>
<th>T_seg</th>
<th>S_LL</th>
</tr>
</thead>
<tbody>
<tr>
<td>DW-11 None/SLEQ/STEPS</td>
<td>[4, 4, 4]</td>
<td>[2, 4, 8, 16]</td>
<td>G(2)</td>
<td>[2, 4, 2]</td>
<td>0.4</td>
<td></td>
</tr>
</tbody>
</table>

Table 7-5 : Distributed Periodic Tasks with Geometric Utilities, random System Load, fixed Local Load

Figure 7-85 & Figure 7-86 show the delta AUR distribution for different systems loads for DASA-ND and LBESA respectively with and without their corresponding decomposition techniques for DW-11. LBESA’s performance improves with SLEQF decomposition techniques for the workload DW-11. DASA-ND does not show any improvement; rather it degrades with STEPS decomposition for this particular workload. We discuss one of the example scenarios & explain the reason for sub-optimal results for DW-11. We also show how LBESA benefits with SLEQF decomposition and becomes as optimal as DASA-ND in this example.

Figure 7-85: Load vs Delta AUR, DASA-ND, 40% Local Load, DW-11 (Table 5), geometric utilities, Unequal costs and task-node mappings derived from an uniform distribution
Scenario D8:

Figure 7-87 shows the task properties for the workload DW-11. The DTs task utilities from a geometric sequence. The local tasks contribute to the local load and have a utility of 3. T4 on N1, T5 on N2, and T6 on N3 contribute to a local-load of 40%. Without decomposition, the node-local LBESA, schedules local tasks earlier than the distributed task segments in the feasible set, since all tasks have equal deadlines. DASA-ND breaks deadline ties in PUD order and still favors the distributed task segments, since they have higher PUD than the local-tasks. Even thought the ordering of the tasks in the local nodes differs between LBESA and DASA-ND, both manage to complete the same number of tasks and accumulate an overall AUR of 0.74 out of an optimal possible of 0.83. Figure 7-88 show that STEPS does not favor DASA-ND for such workloads. STEPS for step-downward TUF tasks, does not alter the segment deadlines, but scales (increases the height of) the TUF of distributed tasks by an order of magnitude relative to the height of the local tasks. The TUF scaling increases the PUD of distributed tasks, thus favoring distributed tasks over the local-tasks. Since there is no marked difference in utility values of local and distributed-tasks in this particular workload, using TUF scaling hinders the optimality for such workloads.
Figure 7-87: Scenario D8 - Both LBESA and DASA-ND are sub-optimal for DW 11, geometric utilities, unequal costs and task-node mappings derived from an uniform distribution.
Figure 7-88: Scenario D8 - DASA-ND compared with DASA-ND + STEPS for DW 11, geometric utilities, unequal costs and task-node mappings derived from an uniform distribution.
Figure 7-89: Scenario D8 - LBESA compared with LBESA+ SLEQF for DW 11, geometric utilities, unequal costs and task-node mappings derived from an uniform distribution.
Figure 7-89 shows that LBESA benefits from the deadline decomposition. With proportional decomposition of the distributed task deadline into segment deadlines, local LBESA schedulers, now schedule the distributed task segments prior to the local-tasks due to their earlier deadlines. For instance, on N2 and N3, LBESA completes the distributed segments T2 and T3 as against the local tasks T5 and T6. Overall, LBESA accumulates a high AUR of 0.78 with deadline decomposition.

Figure 7-90 & Figure 7-91 show the C.D.F and HISTOGRAM for DASA-ND and LBESA without and with their corresponding decomposition techniques for workload DW-10. For DASA-ND, there is a 0.66 probability that it is optimal. For LBESA there is a 0.35 probability that it is optimal without decomposition. However, both show a 0.66 probability that they are optimal with SLEQF decomposition in presence of additional local load (see Figure 7-91 & Figure 7-92). Hence, deadline decomposition benefits LBESA. TUF scaling alone does not improve DASA-ND and LBESA, but rather degrades it.

Figure 7-90 : C.D.F and HISTOGRAM, DASA-ND and LBESA, 40% Local Load DW- 11, geometric utilities, unequal costs and task-node mappings derived from an uniform distribution
Figure 7-91: C.D.F and HISTOGRAM, DASA-ND with Decomposition, 40% Local Load DW-11, geometric utilities, unequal costs and task-node mappings derived from an uniform distribution.

Figure 7-92: C.D.F and HISTOGRAM, LBESA with Decomposition, 40% Local Load DW-11, geometric utilities, unequal costs and task-node mappings derived from an uniform distribution.
We conclude that deadline decomposition aids LBESA and DASA-ND and substantially improves their performance. TUF scaling degrades both DASA-ND and LBESA’s performance for distributed workloads with geometric sequence utilities and equal deadlines in the presence of local load. This result is dependent on the utility values chosen for the local-tasks. The utility of the local task is more than the utility of one of the DT in this workload. Hence, TUF scaling does not aid DASA-ND and LBESA for such combinations of task utilities. With deadline decomposition, LBESA and DASA-ND have a 0.66 probability that they are optimal for distributed workloads with geometric sequence utilities and equal task deadlines even in the presence of local load on system nodes.

7.4.1.3 ARITHMETIC TASK UTILITIES

Table 7-6 shows the workload properties for DW-12, which has arithmetic sequence utilities and varying segment costs for the 3 distributed tasks. We create an additional local-load of 40% on each of the 3 nodes.

<table>
<thead>
<tr>
<th>W</th>
<th>Sd</th>
<th>Tperiod</th>
<th>Tutility</th>
<th>UT</th>
<th>Tseg</th>
<th>SLL</th>
</tr>
</thead>
<tbody>
<tr>
<td>DW-12</td>
<td>None/SLEQF/STEPS</td>
<td>[4, 4, 4]</td>
<td>[100, 200, 300]</td>
<td>A(10)</td>
<td>{2, 4, 2}</td>
<td>0.4</td>
</tr>
</tbody>
</table>

Table 7-6 : Distributed Periodic Tasks with Arithmetic Utilities, random System Load, and Fixed Local Load

Figure 7-93 & Figure 7-94 show the delta AUR distribution for different systems loads for DASA-ND and LBESA with and without their corresponding decomposition techniques for DW-12. As in DW-11, LBESA’s optimality substantially improves with SLEQF decomposition techniques for the workload DW-12. DASA-ND does not show much improvement. As in DW-10, there is a marginal increase in the optimality of DASA-ND with TUF scaling.

Figure 7-95 & Figure 7-96 show the C.D.F and HISTOGRAM for DASA-ND and LBESA without and with their corresponding decomposition techniques for workload DW-12.
For DASA-ND, there is a 0.77 probability that it is optimal without TUF scaling and a 0.8 probability that it is optimal with TUF scaling in the presence of additional local-load. There is an additional improvement with deadline slicing technique. **DASA-ND shows a 0.89 probability of being optimal with SLEQF.** For LBESA there is a 0.12 probability that it is optimal without decomposition and 0.89 probability that it is optimal with SLEQF decomposition in the presence of additional local load. TUF scaling alone does not aid LBESA. (See Figure 7-96 & Figure 7-97)

![Graph](image)

**Figure 7-93**: Load vs Delta AUR, DASA-ND, 40% Local Load, DW-12, Arithmetic utilities, unequal costs, and task-node mappings derived from an uniform distribution
Hence, we conclude that deadline decomposition aids LBESA and DASA-ND and substantially improves their optimality. Both have a 0.89 probability that they are optimal for distributed workloads with arithmetic sequence utilities and equal task deadlines even in the presence of additional local load. TUF scaling (PUD scaling) only aids DASA-ND in increasing its optimality for workloads with arithmetic sequence utilities and equal task deadlines, since LBESA does not break deadline ties in PUD order.
Figure 7-95: C.D.F and HISTOGRAM, DASA-ND and LBESA, 40% Local Load DW-12, Arithmetic utilities, unequal costs and task-node mappings derived from a uniform distribution.

Figure 7-96: C.D.F and HISTOGRAM, DASA-ND with Decomposition, 40% Local Load DW-12, Arithmetic utilities, unequal costs and task-node mappings derived from an uniform distribution.
7.4.2 Aperiodic Task Model

We next experiment with the aperiodic task model for the UA schedulers in a distributed system. Execution times “T_{cost}” for each task are modeled as random variables from an super-exponential probability distribution function with an expected value of 3.0. The arrival-rate \( \lambda \) is modeled as a random variable from a Poisson distribution with an average value \( \lambda_{avg} = (S_t \times t-c) / \rho \), where “\( \rho \)” represents system load. The slack is computed as a random variable from a uniform distribution within an interval of [0.5-2.5]. The task deadline is the sum of the task execution time and the slack. We only experiment with a stochastic cost model. We have used the same workloads that were used in the previous experiments and compared the results from the periodic task model with that of the aperiodic task model.

Table 7-7 shows the workload properties for aperiodic distributed tasks, which have different utility patterns, and varying segment costs for the 3 distributed tasks. We create an additional local-load on the three nodes. The costs for the local costs are derived with an expected value equal to that of the distributed task costs.
<table>
<thead>
<tr>
<th>W</th>
<th>$S_d$</th>
<th>$T_{period}$</th>
<th>$T_{utility}$</th>
<th>UT</th>
<th>$T_{seg}$</th>
<th>Utility$\rightarrow$Segments</th>
<th>SUL</th>
</tr>
</thead>
<tbody>
<tr>
<td>DWA-10</td>
<td>None/SLEQF/STEPS</td>
<td>AR = 8.33</td>
<td>[100, 100, 100][60]</td>
<td>E</td>
<td>[2, 2, 3]</td>
<td>N/A</td>
<td>0.4</td>
</tr>
<tr>
<td>DW-A11</td>
<td>None/SLEQF/STEPS</td>
<td>AR = 8.33</td>
<td>[2, 4, 8][3]</td>
<td>G(2)</td>
<td>[2, 2, 3]</td>
<td>Small No of Segs $\rightarrow$ LU Large No of Segs $\rightarrow$ HU</td>
<td>0.4</td>
</tr>
<tr>
<td>DW-A12</td>
<td>None/SLEQF/STEPS</td>
<td>AR = 8.33</td>
<td>[100, 200, 300][60]</td>
<td>A(100)</td>
<td>[2, 2, 3]</td>
<td>Small No of Segs $\rightarrow$ LU Large No of Segs $\rightarrow$ HU</td>
<td>0.4</td>
</tr>
</tbody>
</table>

Table 7-7: Distributed Aperiodic Tasks with different utility patterns

### 7.4.2.1 EQUAL TASK UTILITIES

Figure 7-98 & Figure 7-99 show the delta AUR distribution for DASA-ND and LBESA for the workload DW-A 10. We have shown that for periodic distributed workloads that have tasks with equal utilities and equal deadlines, DASA-ND and LBESA have a 0.80 probability that they are optimal with decomposition. For aperiodic workloads with equal task utilities, both DASA-ND and LBESA show a 0.7 probability that they are optimal (see Figure 7-100). Applying TUF scaling to both does not alter their optimality, but deadline decomposition reduces the optimality of both DASA-ND and LBESA (see Figure 7-101 & Figure 7-102)
Figure 7-98: Load vs Delta AUR, DASA-ND DW-A-10, Equal utilities, Unequal costs and task-node mappings derived from an uniform distribution

Figure 7-99: Load vs Delta AUR, LBESA DW-A-10 (Table 7), Equal utilities, Unequal costs and task-node mappings derived from a uniform distribution
Figure 7-100: C.D.F and HISTOGRAM for DW-A 10, Equal utilities, Unequal costs, and task-node mappings derived from an uniform distribution.

Figure 7-101: C.D.F and HISTOGRAM for DW-A 10, DASA-ND with decomposition, Equal utilities, Unequal costs and task-node mappings derived from an uniform distribution.

188
Scenario D9:

We investigate the reasons for decline in the performance of LBESA under deadline decomposition. **Figure 7-103** shows an example aperiodic task set for workload DW-A-10, with equal utilities. T4, T5, and T6 contribute to the additional local-load on N1, N2, and N3. Both DASA-ND and LBESA are optimal with an AUR of 0.38. The rightmost schedule shows that ordering of tasks in LBESA with deadline decomposition. The downside of the deadline slicing is that segments and hence end-end tasks have higher chances of being rejected earlier in the ordering if they have a lower PUD compared to other competing segments in the local nodes. A segment that would be normally delayed on a local node is now categorized infeasible, thus affecting the end-end completion of the task. In this specific example, deadline decomposition for distributed tasks, T2 and T3 forces them to finish earlier, making more room for the subsequent segments. However, both T2 and T3 start on the same node. Not all the competing tasks (T2, T3, and local
task T5) can be accommodated together before their deadlines. Hence, LBESA removes the lowest PUD task T2 and schedules T3. T3 being a longer segment, eventually fails to complete, since the second segment of T3 loses to the shorter, higher PUD local-task T5. Hence, deadline decomposition adversely affects the ordering of LBESA node-local schedulers.
Figure 7-103: Scenario D9: LBESA and DASA-ND optimal, LBESA results in sub-optimal result with SLEQF for DW-A-10, Equal Utilities
7.4.2.2 GEOMETRIC TASK UTILITIES

We next compare the performance of DASA-ND and LBESA for aperiodic distributed workloads that have task utilities in geometric sequence with that of their periodic counterpart. Figure 7-104 & Figure 7-105 show the delta AUR distribution for DASA-ND and LBESA for the workload DW-A 11. We have shown that for periodic distributed workloads that have tasks with geometric sequence utilities and equal deadlines, both DASA-ND and LBESA (with deadline decomposition) have a 0.66 probability that they are optimal. For aperiodic workloads with geometric sequence task utilities, both DASA-ND and LBESA show a 0.61 probability that they are optimal. Applying TUF scaling to them does not alter their optimality rate (See Figure 7-107 & Figure 7-108), but deadline decomposition reduces the optimality of both DASA-ND and LBESA for similar reasons explained in Scenario D9.

Figure 7-104: Load vs Delta AUR, DASA-ND DW-A-11, Geometric utilities, unequal costs and task-node mappings derived from an uniform distribution
Figure 7-105: Load vs Delta AUR, LBESA DW-A-11, Geometric utilities, unequal costs and task-node mappings derived from an uniform distribution.

Figure 7-106: C.D.F and HISTOGRAM for DW-A11, Geometric utilities, unequal costs and task-node mappings derived from an uniform distribution.
Figure 7-107: C.D.F and HISTOGRAM for DW-A 11, DASA-ND with decomposition, Geometric utilities, unequal costs and task-node mappings derived from an uniform distribution

Figure 7-108: C.D.F and HISTOGRAM for DW-A 11, LBESA with decomposition, Geometric utilities, unequal costs and task-node mappings derived from an uniform distribution
7.4.2.3 ARITHMETIC TASK UTILITIES

Finally, we compare the performance of DASA-ND and LBESA for aperiodic workloads that have task utilities in arithmetic sequence with their corresponding periodic task set. Figure 7-109 & Figure 7-110 show the delta AUR distribution for DASA-ND and LBESA for the workload DW-A12.

We have already shown that for periodic distributed workloads with tasks utilities in arithmetic sequence and equal deadlines, DASA-ND and LBESA have a 0.89 probability that they are optimal with deadline decomposition. For aperiodic workloads with arithmetic sequence utilities, both DASA-ND and LBESA show a 0.56 probability that they are optimal. Applying TUF scaling to both of them does not alter their optimality, but deadline decomposition reduces their optimality showing a probability of just 0.41 that they are optimal (see Figure 7-112 & Figure 7-113).

We investigate an example case to determine the reason for the degradation of LBESA with deadline decomposition for aperiodic task sets. Figure 7-114 illustrates an example task set for the workload DW-A12. Figure 7-115 compares LBESA with and without the SLEQF deadline decomposition technique. When the local tasks are shorter, they end-up with higher PUD than the distributed tasks, and thus, even the deadline decomposition does not aid LBESA. In this case, T5, a local task has the earliest deadline and is scheduled earlier to the distributed segments. Deadline decomposition forces LBESA to eliminate the lower PUD task T4 from the feasible set and schedule T5. Eventually, T5, a longer task with more segments than T4, fails to complete before its deadline. Thus, deadline decomposition ends up accumulating only an AUR of 0.26 leading to further degradation in the performance of LBESA.
Figure 7-109: Load vs Delta AUR, DASA-ND DW-A-12, Arithmetic utilities, unequal costs and task-node mappings derived from an uniform distribution

Figure 7-110: Load vs Delta AUR, LBESA, DW-A-12 (Table 7-7), Arithmetic utilities, unequal costs, and task-node mappings derived from an uniform distribution
Figure 7-111: C.D.F and HISTOGRAM for DW-A 12, Arithmetic utilities, unequal costs and task-node mappings derived from an uniform distribution.

Figure 7-112: C.D.F and HISTOGRAM for DW-A 12, DASA-ND with decomposition, Arithmetic utilities, unequal costs and task-node mappings derived from an uniform distribution.
Figure 7-113: C.D.F and HISTOGRAM for DW-A 12, LBESA with decomposition, (Arithmetic utilities, unequal costs, and task-node mappings derived from a uniform distribution
Figure 7-114: Scenario D10 - Both LBESA and DASA-ND are optimal for DW-A -12, Arithmetic utilities, Unequal costs, and task-node mappings derived from an uniform distribution.
Figure 7-115: Scenario D10 - LBESA compared with LBESA with SLEQF for DW-A-12 (Table 7-7)
7.5 Summary of Results for Distributed Workloads

Table 7-8 & Table 7-9 provide a summary of all the workloads discussed so far, with their task properties and results for the optimality of DASA-ND and LBESA. Table 7-10 shows the description of acronyms used in Table 7-8 & Table 7-9. Table 7-8 shows the probability that both DASA-ND and LBESA are completely optimal for different workloads. Table 7-9 shows the probability that both DASA-ND and LBESA are near optimal for different workloads with a less than 0.1 difference in the delta AUR values.

To summarize our analysis for distributed workloads, we conclude that,

- For all periodic workloads, at system loads in the range 0.0 to 0.6, both DASA-ND and LBESA are optimal irrespective of load distribution and utility pattern.

- For periodic workloads, at high system loads i.e. > 0.6 with equal load distribution, both DASA-ND and LBESA with SLEQF decomposition are near optimal with a high probability of 0.925 for task sets with equal number of task segments, equal task deadlines, and geometric task utilities (high CR). Both DASA-ND and LBESA with SLEQF decomposition have a 0.55 probability of being optimal for workloads with equal task segments, deadlines, and equal task utilities. Both DASA-ND and LBESA with SLEQF decomposition have a 0.73 probability of being optimal for workloads with equal task segments, deadlines, and geometric task utilities with low CR. Both DASA-ND and LBESA with SLEQF decomposition have a 0.65 probability of being optimal for workloads with equal task segments, deadlines, and arithmetic task utilities.

- For periodic workloads, at high system loads, both DASA-ND and LBESA show a higher likelihood of being optimal for workloads with a mix of small and large number of task segments. DASA-ND shows at least a 0.87 probability that it is optimal and LBESA shows at least a 0.72 probability that it is optimal without any decomposition. They favor task sets with smaller number of segments having lower utility values than the ones with larger number of segments. Actually, deadline decomposition slightly degrades the results for DASA-ND. This result does not depend on the task utility pattern.
• For periodic workloads, irrespective of the utility type, both DASA-ND and LBESA degrade in performance with unequal load distribution compared to equal load distribution. However, the degree of degradation is high for task sets with equal and arithmetic utilities.

• For periodic workloads, in the presence of local load with equal task deadlines, deadline decomposition SLEQF substantially improves the optimality of both DASA-ND and LBESA. On the other hand, TUF scaling alone with STEPS shows no improvement for DASA-ND and degrades the optimality of LBESA, irrespective of utility patterns.

• Deadline decomposition, SLEQF degrades DASA-ND and LBESA’s optimality for aperiodic workloads in the presence of local load, irrespective of the utility type. TUF scaling, STEPS makes no difference irrespective of utility patterns.
<table>
<thead>
<tr>
<th>TM</th>
<th>LD</th>
<th>LL</th>
<th>CT</th>
<th>DT</th>
<th>ST</th>
<th>UTIL</th>
<th>S→u</th>
<th>DASA-ND</th>
<th>D+SLEQF</th>
<th>D+STEPS</th>
<th>LBESA</th>
<th>L+SLEQF</th>
<th>L+STEPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>P</td>
<td>E / UE</td>
<td>NO</td>
<td>E</td>
<td>E</td>
<td>E</td>
<td>E</td>
<td>0.23</td>
<td>0.14</td>
<td>0.55</td>
<td>0.48</td>
<td>0.21</td>
<td>0.12</td>
<td>0.55</td>
</tr>
<tr>
<td>P</td>
<td>E / UE</td>
<td>NO</td>
<td>E</td>
<td>E</td>
<td>E</td>
<td>UE</td>
<td>0.54</td>
<td>0.34</td>
<td>0.75</td>
<td>0.50</td>
<td>0.68</td>
<td>0.35</td>
<td>0.75</td>
</tr>
<tr>
<td>P</td>
<td>E / UE</td>
<td>NO</td>
<td>E</td>
<td>E</td>
<td>E</td>
<td>G(2)</td>
<td>0.63</td>
<td>0.61</td>
<td>0.73</td>
<td>0.71</td>
<td>0.63</td>
<td>0.61</td>
<td>0.73</td>
</tr>
<tr>
<td>P</td>
<td>E / UE</td>
<td>NO</td>
<td>E</td>
<td>E</td>
<td>E</td>
<td>G(10)</td>
<td>0.65</td>
<td>0.60</td>
<td>0.72</td>
<td>0.71</td>
<td>0.65</td>
<td>0.60</td>
<td>0.72</td>
</tr>
<tr>
<td>P</td>
<td>E / UE</td>
<td>NO</td>
<td>E</td>
<td>E</td>
<td>E</td>
<td>UE</td>
<td>G(2)</td>
<td>0.87</td>
<td>0.38</td>
<td>0.72</td>
<td>0.54</td>
<td>0.72</td>
<td>0.48</td>
</tr>
<tr>
<td>P</td>
<td>E / UE</td>
<td>NO</td>
<td>E</td>
<td>E</td>
<td>E</td>
<td>UE</td>
<td>G(2)</td>
<td>0.41</td>
<td>0.41</td>
<td>0.62</td>
<td>0.67</td>
<td>0.41</td>
<td>0.34</td>
</tr>
<tr>
<td>P</td>
<td>E / UE</td>
<td>NO</td>
<td>E</td>
<td>E</td>
<td>E</td>
<td>E</td>
<td>A(100)</td>
<td>0.57</td>
<td>0.53</td>
<td>0.65</td>
<td>0.63</td>
<td>0.57</td>
<td>0.53</td>
</tr>
<tr>
<td>P</td>
<td>E / UE</td>
<td>NO</td>
<td>E</td>
<td>E</td>
<td>E</td>
<td>E</td>
<td>A(100)</td>
<td>0.87</td>
<td>0.45</td>
<td>0.72</td>
<td>0.50</td>
<td>0.72</td>
<td>0.36</td>
</tr>
<tr>
<td>P</td>
<td>E / UE</td>
<td>NO</td>
<td>E</td>
<td>E</td>
<td>E</td>
<td>E</td>
<td>A(100)</td>
<td>0.40</td>
<td>0.44</td>
<td>0.63</td>
<td>0.67</td>
<td>0.40</td>
<td>0.34</td>
</tr>
<tr>
<td>P</td>
<td>UE</td>
<td>YES</td>
<td>UE</td>
<td>E</td>
<td>UE</td>
<td>E</td>
<td>0.70</td>
<td>0.8</td>
<td>0.72</td>
<td>0.20</td>
<td>0.80</td>
<td>0.2</td>
<td></td>
</tr>
<tr>
<td>P</td>
<td>UE</td>
<td>YES</td>
<td>UE</td>
<td>E</td>
<td>UE</td>
<td>G(2)</td>
<td>0.66</td>
<td>0.66</td>
<td>0.45</td>
<td>0.35</td>
<td>0.66</td>
<td>0.33</td>
<td></td>
</tr>
<tr>
<td>P</td>
<td>UE</td>
<td>YES</td>
<td>UE</td>
<td>E</td>
<td>UE</td>
<td>A(100)</td>
<td>0.77</td>
<td>0.89</td>
<td>0.80</td>
<td>0.12</td>
<td>0.89</td>
<td>0.12</td>
<td></td>
</tr>
<tr>
<td>AP</td>
<td>UE</td>
<td>YES</td>
<td>UE</td>
<td>AR</td>
<td>UE</td>
<td>E</td>
<td>0.7</td>
<td>0.52</td>
<td>0.7</td>
<td>0.7</td>
<td>0.52</td>
<td>0.7</td>
<td></td>
</tr>
<tr>
<td>AP</td>
<td>UE</td>
<td>YES</td>
<td>UE</td>
<td>AR</td>
<td>UE</td>
<td>G(2)</td>
<td>0.61</td>
<td>0.46</td>
<td>0.61</td>
<td>0.61</td>
<td>0.47</td>
<td>0.61</td>
<td></td>
</tr>
<tr>
<td>AP</td>
<td>UE</td>
<td>YES</td>
<td>UE</td>
<td>AR</td>
<td>UE</td>
<td>A(100)</td>
<td>0.56</td>
<td>0.4</td>
<td>0.56</td>
<td>0.56</td>
<td>0.4</td>
<td>0.4</td>
<td></td>
</tr>
</tbody>
</table>

Table 7-8: Summary of results for distributed workloads
<table>
<thead>
<tr>
<th>TM</th>
<th>LD</th>
<th>LL</th>
<th>CT</th>
<th>DT</th>
<th>ST</th>
<th>UTIL</th>
<th>S→u</th>
<th>D &lt; 0.1</th>
<th>D+SLEOF</th>
<th>D+STEPS</th>
<th>L &lt; 0.1</th>
<th>L+SLEOF</th>
<th>L+STEPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>P</td>
<td>E/UE</td>
<td>NO</td>
<td>E</td>
<td>E</td>
<td>E</td>
<td>E</td>
<td></td>
<td>0.23</td>
<td>0.14</td>
<td>0.55</td>
<td>0.48</td>
<td>0.21</td>
<td>0.12</td>
</tr>
<tr>
<td>P</td>
<td>E/UE</td>
<td>NO</td>
<td>E</td>
<td>E</td>
<td>UE</td>
<td>E</td>
<td></td>
<td>0.5</td>
<td>0.34</td>
<td>0.75</td>
<td>0.50</td>
<td>0.68</td>
<td>0.35</td>
</tr>
<tr>
<td>P</td>
<td>E/UE</td>
<td>NO</td>
<td>E</td>
<td>E</td>
<td>E</td>
<td>G(2)</td>
<td></td>
<td>0.81</td>
<td>0.825</td>
<td>0.81</td>
<td>0.83</td>
<td>0.78</td>
<td>0.825</td>
</tr>
<tr>
<td>P</td>
<td>E/UE</td>
<td>NO</td>
<td>E</td>
<td>E</td>
<td>E</td>
<td>G(10)</td>
<td></td>
<td>0.925</td>
<td>0.92</td>
<td>0.92</td>
<td>0.91</td>
<td>0.92</td>
<td>0.92</td>
</tr>
<tr>
<td>P</td>
<td>E/UE</td>
<td>NO</td>
<td>E</td>
<td>E</td>
<td>UE</td>
<td>G(2)</td>
<td>S S→LU LS→HU</td>
<td>0.92</td>
<td>0.73</td>
<td>0.77</td>
<td>0.69</td>
<td>0.76/</td>
<td>0.59</td>
</tr>
<tr>
<td>P</td>
<td>E/UE</td>
<td>NO</td>
<td>E</td>
<td>E</td>
<td>UE</td>
<td>G(2)</td>
<td>SS→HU LS→LU</td>
<td>0.71</td>
<td>0.71</td>
<td>0.91</td>
<td>0.90</td>
<td>0.69</td>
<td>0.64</td>
</tr>
<tr>
<td>P</td>
<td>E/UE</td>
<td>NO</td>
<td>E</td>
<td>E</td>
<td>E</td>
<td>A(100)</td>
<td></td>
<td>0.77</td>
<td>0.82</td>
<td>0.79</td>
<td>0.82</td>
<td>0.75</td>
<td>0.81</td>
</tr>
<tr>
<td>P</td>
<td>E/UE</td>
<td>NO</td>
<td>E</td>
<td>E</td>
<td>UE</td>
<td>A(100)</td>
<td>SS→LU LS→HU</td>
<td>0.91</td>
<td>0.7</td>
<td>0.76</td>
<td>0.69</td>
<td>0.76</td>
<td>0.6</td>
</tr>
<tr>
<td>P</td>
<td>E/UE</td>
<td>NO</td>
<td>E</td>
<td>E</td>
<td>UE</td>
<td>A(100)</td>
<td>SS→HU LS→LU</td>
<td>0.58</td>
<td>0.60</td>
<td>0.90</td>
<td>0.87</td>
<td>0.57</td>
<td>0.50</td>
</tr>
<tr>
<td>P</td>
<td>UE</td>
<td>YES</td>
<td>UE</td>
<td>E</td>
<td>UE</td>
<td>E</td>
<td></td>
<td>0.72</td>
<td>0.89</td>
<td>0.77</td>
<td>0.50</td>
<td>0.89</td>
<td>0.55</td>
</tr>
<tr>
<td>P</td>
<td>UE</td>
<td>YES</td>
<td>UE</td>
<td>E</td>
<td>UE</td>
<td>G(2)</td>
<td></td>
<td>0.83</td>
<td>0.92</td>
<td>0.75</td>
<td>0.67</td>
<td>0.88</td>
<td>0.67</td>
</tr>
<tr>
<td>P</td>
<td>UE</td>
<td>YES</td>
<td>UE</td>
<td>E</td>
<td>UE</td>
<td>A(100)</td>
<td></td>
<td>0.81</td>
<td>0.92</td>
<td>0.81</td>
<td>0.30</td>
<td>0.92</td>
<td>0.30</td>
</tr>
<tr>
<td>AP</td>
<td>UE</td>
<td>YES</td>
<td>UE</td>
<td>AR</td>
<td>UE</td>
<td>E</td>
<td></td>
<td>0.77</td>
<td>0.58</td>
<td>0.77</td>
<td>0.77</td>
<td>0.58</td>
<td>0.77</td>
</tr>
<tr>
<td>AP</td>
<td>UE</td>
<td>YES</td>
<td>UE</td>
<td>AR</td>
<td>UE</td>
<td>G(2)</td>
<td></td>
<td>0.61</td>
<td>0.47</td>
<td>0.62</td>
<td>0.62</td>
<td>0.47</td>
<td>0.62</td>
</tr>
<tr>
<td>AP</td>
<td>UE</td>
<td>YES</td>
<td>UE</td>
<td>AR</td>
<td>UE</td>
<td>A(100)</td>
<td></td>
<td>0.58</td>
<td>0.46</td>
<td>0.58</td>
<td>0.58</td>
<td>0.46</td>
<td>0.58</td>
</tr>
</tbody>
</table>

Table 7-9 : Summary of results for distributed workloads with < 0.1 difference in delta AUR
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TM</td>
<td>TASK MODEL</td>
</tr>
<tr>
<td>LD</td>
<td>LOAD DISTRIBUTION</td>
</tr>
<tr>
<td>LL</td>
<td>LOCAL LOAD</td>
</tr>
<tr>
<td>CT</td>
<td>COST TYPE</td>
</tr>
<tr>
<td>DT</td>
<td>DEADLINE TYPE</td>
</tr>
<tr>
<td>ST</td>
<td>SEGMENT TYPE</td>
</tr>
<tr>
<td>S-U</td>
<td>SEGMENT TO UTILITY RELATION</td>
</tr>
<tr>
<td>P</td>
<td>PERIODIC TASK MODEL</td>
</tr>
<tr>
<td>LS</td>
<td>LARGE NUMBER OF SEGMENTS</td>
</tr>
<tr>
<td>SS</td>
<td>SMALL NUMBER OF SEGMENTS</td>
</tr>
<tr>
<td>UE</td>
<td>UNEQUAL</td>
</tr>
<tr>
<td>E</td>
<td>EQUAL</td>
</tr>
<tr>
<td>AR</td>
<td>ARRIVAL RATE</td>
</tr>
<tr>
<td>B</td>
<td>BALANCED LOAD</td>
</tr>
<tr>
<td>UB</td>
<td>UNBALANCED LOAD</td>
</tr>
<tr>
<td>D</td>
<td>DASA-ND</td>
</tr>
<tr>
<td>L</td>
<td>LBESA</td>
</tr>
<tr>
<td>AP</td>
<td>APERIODIC TASK MODEL</td>
</tr>
<tr>
<td>G</td>
<td>GEOMETRIC SEQUENCE UTILITIES</td>
</tr>
<tr>
<td>A</td>
<td>ARITHMETIC SEQUENCE UTILITIES</td>
</tr>
<tr>
<td>HU</td>
<td>HIGH UTILITY VALUE</td>
</tr>
<tr>
<td>LU</td>
<td>LOW UTILITY VALUE</td>
</tr>
</tbody>
</table>

Table 7-10: Legend for Table 7-8 and Table 7-9
Chapter 8 Conclusion and Future Work

We have implemented a scheduling algorithm that finds optimal schedules for any given task set with step TUFs in both single-node and distributed UA scheduling environments. The scheduler has super-exponential-time complexity for large task sets. Hence, it can only serve as a reference to evaluate and analyze UA schedulers for small, representative periodic and aperiodic task sets.

We have performed an exhaustive experimental analysis for different combinations of task and system parameters. These workloads are generated using unbiased methods. The results indicate that for most scenarios, the UA schedulers create optimal schedules. We conclude that DASA-ND and LBESA show highest probability of being optimal in 3 important cases.

1) For periodic tasks in both single-node and distributed environments, if the assigned utilities form a geometric sequence with an order of magnitude difference in their utility values.

2) For periodic tasks in single-node with equal utilities among the tasks.

3) For distributed workloads, there exists a high probability that deadline decomposition aids UA schedulers in creating near optimal schedules.

We have also identified specific combinations of task and system parameters that result in sub-optimal results for these UA schedulers.

To summarize our analysis, we conclude that LBESA shows sub-optimal results for workloads that have deadline ties among tasks. Delaying a high PUD task in the equal deadline order on local nodes decreases the probability of end-end completion of tasks in both single-node and distributed environments. In distributed systems, in addition to the task characteristics such as periodicity, and task utility pattern, the load on individual nodes determined by the segment-node mapping influence the optimality of local UA
Schedulers. The ratio (PUD) alone as a heuristic in single-node and distributed cases is insufficient for the UA schedulers. The high PUD of a task can be a result of low cost of the task. Thus completing more than one shorter task, with high PUD but lower utility may not accrue as much utility as completing one highest utility, high cost task. DASA attempts to complete as many tasks as possible by creating a deadline-ordered feasible set. It can delay a high PUD task with longer deadline and schedule an earlier deadline, low PUD task in the feasible set. This delay reduces the probability of completing the high PUD task depending on the future arrival of tasks.

Our results and analysis aid the system operators using UA scheduling for soft real-time applications in making optimal decisions in assigning utility values to local and distributed tasks.

**Future Work**

There are mainly two directions in which the current work can be extended. With the existing setup, other important UA schedulers that support step TUFs such as $D_{\text{over}}$ and GUS (Generic Utility Scheduling) could be evaluated for their optimality. So far, we have considered only workloads with either periodic or aperiodic task sets. Future experiments might consider better techniques to model workloads with a combination of periodic and aperiodic task sets to determine the degree to which the stochastic nature of the arrival processes influences the heuristic decisions. Our work used a naïve load-assignment strategy with equal task costs to create different load distribution patterns across the system nodes. A more advanced technique would be able to analyze different variations in the task properties for the same load distribution patterns.

The existing implementation could be optimized using a bounding technique as explained in Chapter 4, to enable the analysis of large task sets. The existing implementation of the optimal scheduling algorithm only supports tasks with step TUFs. Future extensions can support task sets with arbitrary TUFs. In case of arbitrary TUFs, the time at which the
task completes decides the actual utility accrued to the system. Hence, the optimal scheduler should try to finish such tasks at their optimal time in order to accrue the maximum task utility.

With this extension, workloads with a combination of step and arbitrary TUFs can be explored and it could aid in analyzing how arbitrary TUFs influence the optimality of UA schedulers at both system underload and overloads. DASA and GUS support resource sharing among the tasks. Another important extension to the optimal scheduler is to model the resource vector for each task. Apart from the feasibility and eligibility constraints, tasks will also have precedence constraints that show how task segments are dependent on completion of other tasks to acquire the required shared resources. It is important to analyze how DASA performs with resource dependencies since real workloads exhibit this characteristic. In addition, other decomposition techniques discussed in [Wu et al., 2005] could be used for distributed workloads along with the different UA schedulers in order to estimate how they influence the global optimality of UA schedulers.
References

Vita

Veena Basavaraj was born in Mysore, a city close to the silicon hub of India, Bangalore. In the year 2002, she got her Bachelor of Science degree in Computer Science and Engineering from National Institute of Engineering (NIE), Mysore, India. Her academic interests are in the area of distributed computing, computer networking, and middleware systems. She then worked as a software engineer in the information systems division of a reputed Banking Company until July 2004. The responsibilities included implementing enterprise systems for the consumer finance division of the ANZ Bank, Melbourne, Australia. She likes to travel and volunteer for non-profit organizations in her leisure.