

Time-Conceptual Foundations of Programming

Masters Thesis of

Wendsomde Yameogo, Dipl.-Inform.(FH)

January 21, 2003

Contents

- Goals
- Preliminaries
- Motivations
- Time-Conceptual Foundations of Programming
- Conclusion – Outlook
- Difficulties/Challenges

Goals

- Applying Temporal Concept Analysis to Programming
 - Identify potential applications of TCA in Programming
 - Investigate technical issues
- First steps towards a Time-Conceptual Theory of Programming

Genesis & Scientific Context

- Formal Concept Analysis (FCA)
 - Wille, beginning of the 80's
 - Mathematization of the concept of concept
 - Based on lattice theory
- Temporal Concept Analysis (TCA)
 - Wolff, last 2 years
 - Application of FCA to temporal modelling
- Time-Conceptual Foundations of Programming
 - Yameogo, this Master's Thesis
 - Apply TCA to Programming

Formal Concept Analysis(1)

Formal Context

Attributes

Objects

	machine-oriented	Problem-oriented	Object-oriented	procedural	functional
Java		X	X		
C++		X	X	X	
Assembler	X				
Pascal		X		X	
Lisp		X			X

Goals - Preliminaries - Motivations - TCFP - Conclusion - Difficulties

Formal Concept Analysis(2)

Formal Concept

	machine-oriented	Problem-oriented	Object-oriented	procedural	functional
Java		X	X		
C++		X	X	X	
Assembler	X				
Pascal		X		X	
Lisp		X			X

Goals - Preliminaries - Motivations - TCFP - Conclusion - Difficulties

Formal Concept Analysis(3)

Implications

	machine-oriented	Problem-oriented	Object-oriented	procedural	functional
Java		X	X		
C++		X	X	X	
Assembler	X				
Pascal		X		X	
Lisp		X			X

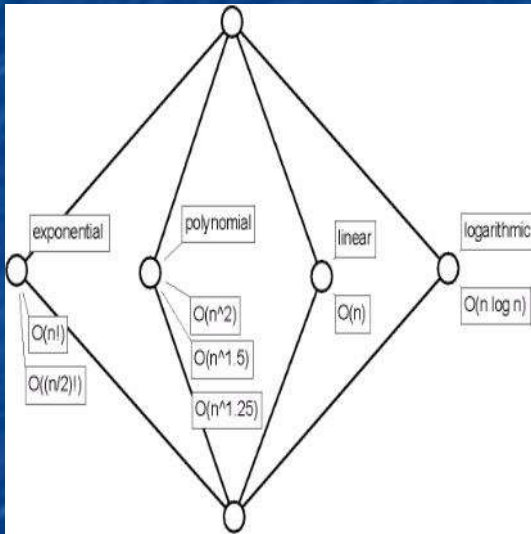
Formal Concept Analysis(5)

Many-Valued Contexts

Algorithm	Worst-Case	Average-Case	Best-Case	Stable
Bubble Sort	$O(n^2)$	$O(n^2)$	$O(n)$	Yes
Heap Sort	$O(n \log n)$	$O(n \log n)$	$O(n \log n)$	No
Insertion Sort	$O(n^2)$	$O(n^2)$	$O(n)$	Yes
Merge Sort	$O(n \log n)$	$O(n \log n)$	$O(n \log n)$	Yes
Quick Sort	$O(n^2)$	$O(n \log n)$	$O(n \log n)$	No
Selection Sort	$O(n^2)$	$O(n^2)$	$O(n^2)$	Yes
Shell Sort	$O(n^{1.5})$	$O(n^{1.25})$	$O(n \log n)$	No
Bogo Sort	$O(n!)$	$O((n/2)!)$	$O(n)$	No

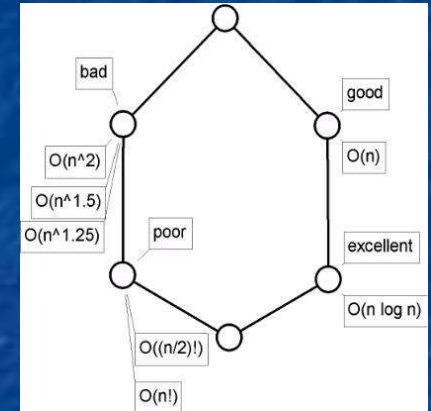
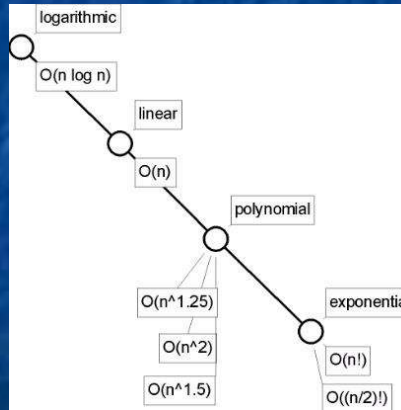
Formal Concept Analysis(6)

Conceptual Scaling



Nominal scale

Ordinal scale



Biordinal scale


Formal Concept Analysis(7)

Derived Formal Context

Algorithm	Worst-Case, poor	Worst-Case, bad	Worst-Case, good	Worst-Case, excellent
Bubble Sort		X		
Heap Sort			X	X
Insertion Sort		X		
Merge Sort			X	X
Quick Sort		X		
Selection Sort		X		
Shell Sort		X		
Bogo Sort	X	X		

Temporal Concept Analysis(1)

Conceptual Time System

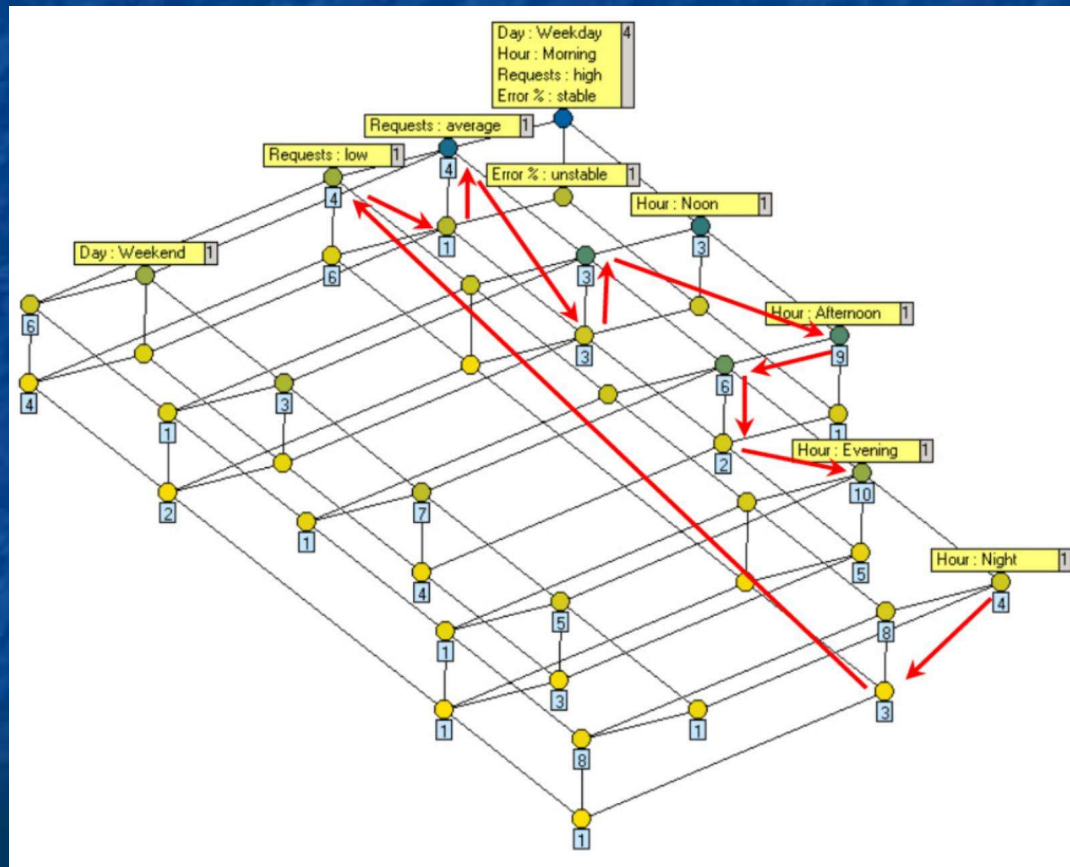


Date	Day	Hour	Requests	Errors %
2200	Fri	00	901	1
2201	Fri	01	603	3
2202	Fri	02	606	0
2203	Fri	03	467	2
2204	Fri	04	494	3
2205	Fri	05	497	1
2206	Fri	06	492	1

Goals - Preliminaries - Motivations - TCFP - Conclusion - Difficulties

Temporal Concept Analysis(2)

Life-track



Goals - Preliminaries - Motivations - TCFP - Conclusion - Difficulties

Why a Time-Conceptual Approach to Programming?

- Sound Mathematical Basis
 - consistency
 - reliability
- Graphical Representation Tool
 - Existing possibilities are limited
- Knowledge-Level Processing
 - Inference over models
 - E.g. Correctness Proofs or Code Generation

TCFP

Application areas

- Program Modelling & Analysis
 - Building Time-Conceptual Model of concrete Programs
 - Analyzing them with respect to
 - Complexity
 - Correctness
 - Semantics
- Reality Modelling and Program Synthesis
 - Building TCA-Model of reality
 - Supporting program creation with this model

TCFP - Program Modelling(1)

Time-Conceptual Interpretation of Turing Machines

- Turing Machine
 - Mathematical Model for the Universal Computer
- acceptor-CTS
 - „Each string accepted by a given Turing Machine defines a Conceptual Time System, the so-called Acceptor-CTS“
 - Event-Part:current state of TM, Time-Part:current-tape symbols

TCFP – Program Modelling(2)

Example Acceptor-CTS

	0	1	b
q0	(q0,0, R)	(q1,1,R)	(q0,0,R)
q1	(qf,0,R)	(qf,0,R)	(qf,0,R)
qf	/	/	/

	q	T1
1	q0	1
2	q1	0
3	qf	b

Example Turing Machine M

Conclusion:

Acceptor-CTS clearly defines a life-track in a situation space. This situation space is a subset of the so-called Machine-situation space.

Acceptor-CTS for „10“ in M

TCFP – Program Modelling()

Application to programs

- A program
 - defines a situation space
- Every program run
 - defines a life-track in this space

Program Modelling(4)

Describing Programs with Implications

- Programs as set of life-tracks
- How to describe set of life-tracks/CTSs?
 - Enumeration (impracticable)
 - With laws(implications)
- We demonstrated how to
 - derive CTS-attributes from a program
 - State implications over those attributes
 - That describe a class of CTSs
 - Such that every CTS models a concrete program run

TCFP-Reality Modelling(1)

- Supporting Reality Modelling Paradigms
 - Like Object-Oriented
 - Or Agent-Oriented
- Need of Relational-Temporal modelling
- Need of a clear definition of Objects
- Introduction of CTSATs

TCFP – Reality Modelling

CTSATs

- Relational and Temporal Modelling
- Introduction of actual actors
- Introduction of aspects
- Introduction of objects as sets of aspects
- Introduction of life-tracks of the objects
- CTSATs can support
 - Agent-discovery in agent-oriented Paradigm
 - Object-discovery in object-oriented Paradigm

Conclusion(1)

- In this work we could
 - Establish first connections between TCA and Programming
 - Identify possible application areas
 - Discuss some possibilities for technical realization
- Goal reached!
- But Work might continue

Conclusion(2)

- Future Work might
 - Generalize the methods introduced
 - Introduce inference logic for CTSATs
 - Investigate special application possibilities
 - Introduce new Temporal Relational structures
 - Formally precise some of the ideas

Difficulties/Challenges

- Complexity of the Research Area
- Methodology of Research-oriented scientific work
- Mathematics
- Last-minute topic choice