Taxonomy of Effortless Creation of Algorithm Visualizations

Petri Ihantola, Ville Karavirta,
Ari Korhonen and Jussi Nikander

HELSINKI UNIVERSITY OF TECHNOLOGY
Department of Computer Science and Engineering
Laboratory of Information Processing Science

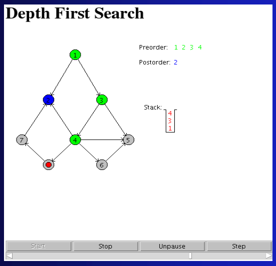
Outline

- What is Algorithm Visualization?
- Motivation & Objectives
- Taxonomy of Effortless Creation of AV
- Example Evaluation of 4 AV systems
- Conclusions

Software Visualization

- Visual = sight (lat.), **but**
- **Visualization** = "the power or process of forming a mental picture or vision of something not actually present to the sight"
- Research area in Software Engineering
- Algorithm Visualization is a subset of SV

Example: JAWAA



Areas of Interest

- Visualization Techniques
  - Pretty-printing, graph models, program visualization, algorithm animation, program auralization, specification styles
- Specialized Domains
  - Visualization of object-oriented programming, functional programming, knowledge based systems, concurrent programs, etc.
- Visualization for Software Engineering
  - Integrated Development Environments (IDE)
- **Visualization for Education & Evaluation**

Motivation

- SV research is technology driven
  - focus on new innovations such as
    - "backward and forward animation" or
    - "multiple views" or
    - "smooth animation"
- Missing connection to CS education research
  - the above are "nice to have", but do they **promote learning**?
- Need for communication channel between
  - SV developers (SV research) and
  - CS educators (CSE research)

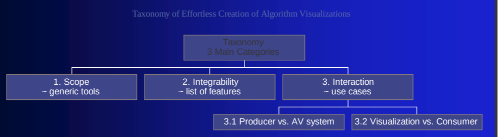
Objectives

1. Methods and tools to **analyse and evaluate** Software Visualizations (SV) (in Educational context)
2. Focus on the “burden of creating new visualizations”, i.e., the **time and effort** required to design, integrate and maintain the visualizations
3. **Taxonomy**: effortlessness in AV systems

Related work

- First evaluation of SV systems (2002) based on taxonomy of Price et al. (1993)
  - technical analysis, no link to CS education
- Questionnaire for CS educators (2004)
  - 22 answers (mostly from SV developers)
- Several other taxonomies and evaluations
  - e.g., Engagement taxonomy, Naps et al. (2003)
- The following taxonomy is a synthesis

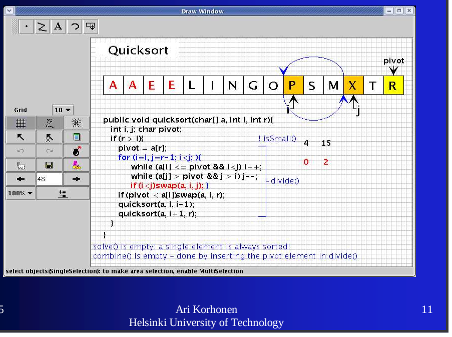
Taxonomy


Taxonomy of Effortless Creation of Algorithm Visualizations
Taxonomy
3 Main Categories
1. Scope
~ generic tools
2. Integrability
~ list of features
3. Interaction
~ use cases
3.1 Producer vs. AV system
3.2 Visualization vs. Consumer

Category 1: Scope

- The range or area the tool deals with
- **Generic tools** like Animal or JAWAA
  - one can produce (almost) any kind content
- vs. **non-generic** tools like MatrixPro and Jeliot 3
  - content (almost always) related to CS education
- More fine-grained classification in the paper

Example: Animal



```
public void quicksort(char[] a, int l, int r){
   int i, j; char pivot;
   if (r > l){
      pivot = a[r];
      for (i=l, j=r-1; i<j; ){
           while (a[i] <= pivot && i<j) i++;
           while (a[j] > pivot && j > i) j--;
           if (i<j)swap(a, i, j); }
      if (pivot < a[i])swap(a, i, r);
      quicksort(a, l, i-1);
      quicksort(a, i+1, r);
   }
}
```

solve() is empty: a single element is always sorted!
combine() is empty – done by inserting the pivot element in divide()

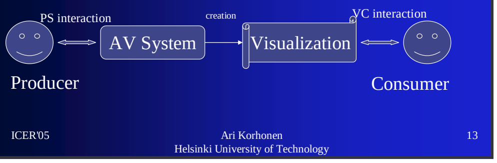
select objects(SingleSelection): to make area selection, enable MultiSelection

Category 2: Integrability

- Basically: a number of “features” that are “nice to have” in all SV systems including
  - easy installation and customization
  - platform independency
  - internationalization
  - documentation and tutorials
  - interactive prediction support
  - course management support
  - integration into a hypertext, etc.
- Bottom line: these are essential, but not sufficient

Category 3: Interaction

- Two kinds of interaction
  - Producer vs. System (PS)
    - resulting new visualization
  - Visualization vs. Consumer (VC)
    - use of the outcome



ICER'05 Ari Korhonen Helsinki University of Technology 13

Producer-System Interaction

- Producer can be, e.g,
  - teacher creating a new lecture demonstration
  - learner submitting a visualization to be graded
- Evaluation based on
  - number of use cases covered in terms of
    - no prior preparation at all
    - requires programming
    - requires programmin and annotation/instrumentation
  - time-on-task

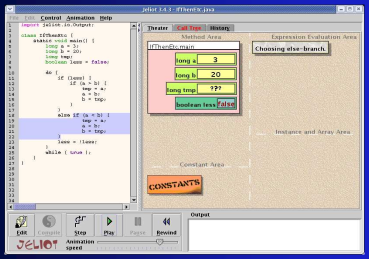
ICER'05 Ari Korhonen Helsinki University of Technology 14

Use Cases (Based on Survey 2004)

- Lecture
  - single lecture example (14)
  - answering student's questions (14)
  - preparing questions for a lecture (14)
- Teaching material production
  - on-line illustrations (12)
  - static (e.g., lecturer's notes) illustrations (12)
- Examination/summative evaluation (12)
- Practice session material
  - exercises (12)
  - demonstrations for tutor/close labs (9)
  - demonstrations for students/closed labs (7)
  - demonstrations for students/open labs (6)

ICER'05 Ari Korhonen Helsinki University of Technology 15

Example: Jeliot 3



```
import jeliot.io.Output;

class IfThenEtc {
    static void main() {
        long a = 3;
        long b = 20;
        long tmp;
        boolean less = false;

        do {
            if (less) {
                if (a > b) {
                    tmp = a;
                    a = b;
                    b = tmp;
                }
            }
            else if (a < b) {
                    tmp = a;
                    a = b;
                    b = tmp;
            }
            less = !less;
        }
        while ( true );
    }
}
```

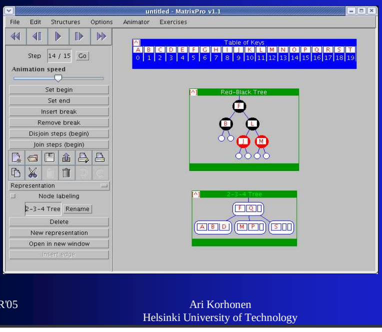
ICER'05 Ari Korhonen Helsinki University of Technology 16

Producer-System Interaction

- Producer can be, e.g,
  - teacher creating a new lecture demonstration
  - learner submitting a visualization to be graded
- Evaluation based on
  - number of use cases covered
  - **time-on-task**
- Especially **on-the-fly** use like in MatrixPro
  - vs. prior preparation

ICER'05 Ari Korhonen Helsinki University of Technology 17

Example: MatrixPro



Step 14 / 15

2-3-4 Tree

Table of Keys

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19

ICER'05 Ari Korhonen Helsinki University of Technology 18

Visualization-Consumer Interaction

- Also consumer can be teacher or learner
- Trivial case: consumer = producer
- In evaluation, consumer = learner
- Engagement taxonomy
  - viewing
  - responding
  - changing
  - constructing
  - representing

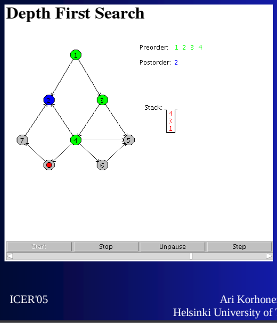
Example Evaluation of 4 Systems

- Systems visualizing concepts in Algorithms and Data Structures course
  - Animal
  - JAWAA 2
  - Jeliot 3
  - MatrixPro
- Disclaimer: some other systems could have been evaluated instead or as well (actually, we did!). However, these are enough to demonstrate the taxonomy in context of algorithms and data structures.

Evaluation

- Based on
  - journal and conference articles as well as subjective experiments (4 authors) with the systems
  - the latest available version
  - the most obvious way to use the system (i.e., how it is intended to be used by the developer)
  - majority of the use cases (i.e., there can be a small number of use cases in which the evaluation could end up to be different)

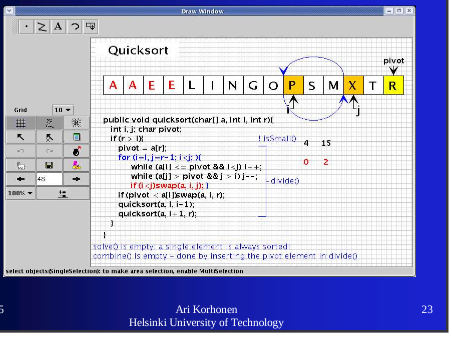
Example: JAWAA



JAWAA animation based on instrumenting code (interesting events)

Separate editor available

Example: Animal

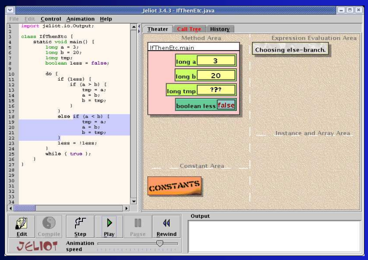


```
public void quicksort(char[] a, int l, int r){
   int i, j; char pivot;
   if (r > l){
      pivot = a[r];
      for (i=l, j=r-1; i<j; ){
           while (a[i] <= pivot && i<j) i++;
           while (a[j] > pivot && j > i) j--;
           if (i<j)swap(a, i, j); }
      if (pivot < a[i])swap(a, i, r);
      quicksort(a, l, i-1);
      quicksort(a, i+1, r);
   }
}
```

solve() is empty: a single element is always sorted!
combine() is empty – done by inserting the pivot element in divide()

select objects(SingleSelection): to make area selection, enable MultiSelection

Example: Jeliot 3



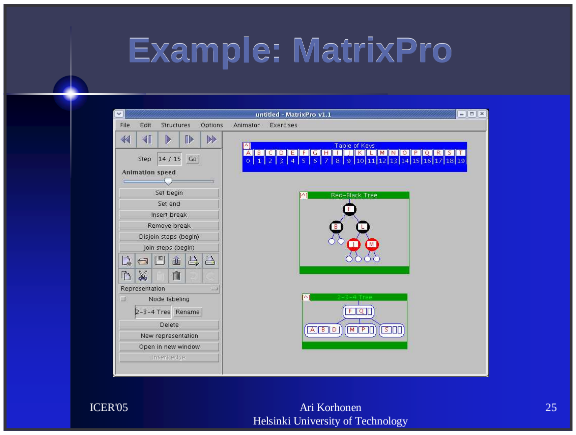
```
import jeliot.io.Output;

class IfThenEtc {
    static void main() {
        long a = 3;
        long b = 20;
        long tmp;
        boolean less = false;

        do {
            if (less) {
                if (a > b) {
                    tmp = a;
                    a = b;
                    b = tmp;
                }
            }
            else if (a < b) {
                    tmp = a;
                    a = b;
                    b = tmp;
            }
            less = !less;
        }
        while ( true );
    }
}
```

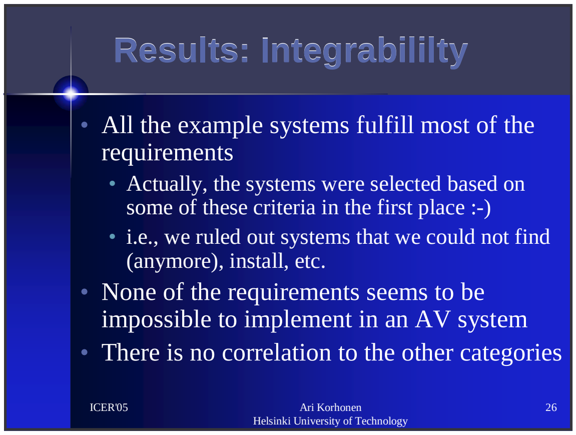
IfThenEtc.main
long a 3
long b 20
long tmp ???
boolean less false

Choosing else-branch.



Example: MatrixPro

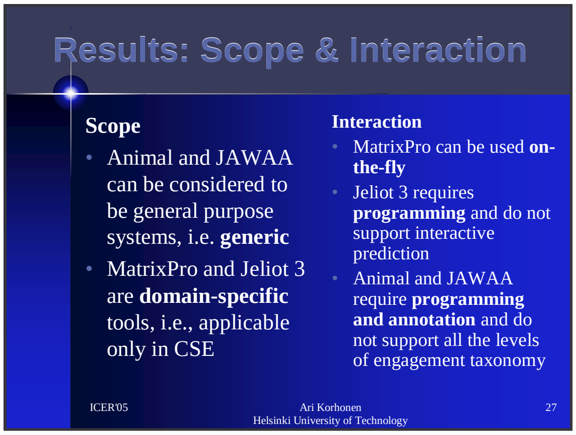
ICER'05 Ari Korhonen
Helsinki University of Technology 25



Results: Integrabililty

- All the example systems fulfill most of the requirements
  - Actually, the systems were selected based on some of these criteria in the first place :-)
  - i.e., we ruled out systems that we could not find (anymore), install, etc.
- None of the requirements seems to be impossible to implement in an AV system
- There is no correlation to the other categories

ICER'05 Ari Korhonen
Helsinki University of Technology 26



Results: Scope & Interaction

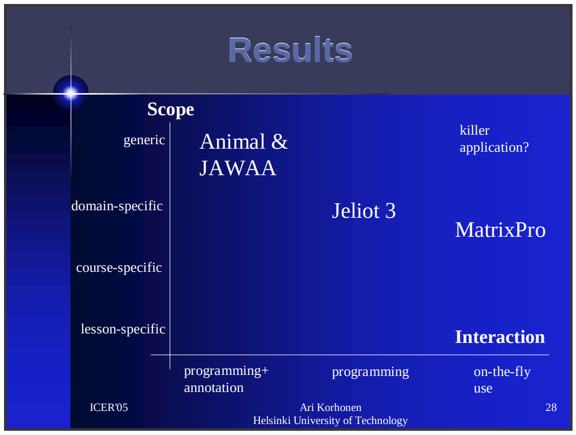
Scope

- Animal and JAWAA can be considered to be general purpose systems, i.e. **generic**
- MatrixPro and Jeliot 3 are **domain-specific** tools, i.e., applicable only in CSE

Interaction

- MatrixPro can be used **on-the-fly**
- Jeliot 3 requires **programming** and do not support interactive prediction
- Animal and JAWAA require **programming and annotation** and do not support all the levels of engagement taxonomy

ICER'05 Ari Korhonen
Helsinki University of Technology 27



Results

Scope

generic

domain-specific

course-specific

lesson-specific

Animal & JAWAA

killer application?

Jeliot 3

MatrixPro

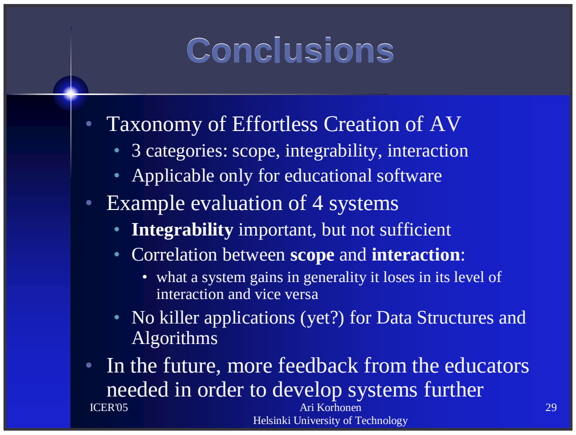
Interaction

programming+ annotation

programming

on-the-fly use

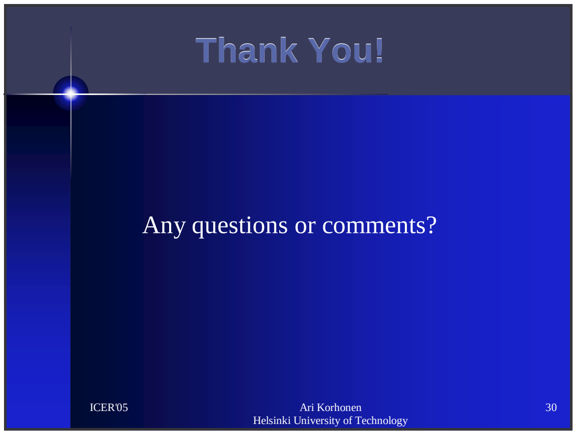
ICER'05 Ari Korhonen
Helsinki University of Technology 28



Conclusions

- Taxonomy of Effortless Creation of AV
  - 3 categories: scope, integrability, interaction
  - Applicable only for educational software
- Example evaluation of 4 systems
  - **Integrability** important, but not sufficient
  - Correlation between **scope** and **interaction**:
    - what a system gains in generality it loses in its level of interaction and vice versa
  - No killer applications (yet?) for Data Structures and Algorithms
- In the future, more feedback from the educators needed in order to develop systems further

ICER'05 Ari Korhonen
Helsinki University of Technology 29



Thank You!

Any questions or comments?

ICER'05 Ari Korhonen
Helsinki University of Technology 30