PART II: Raven Assignments

GraBaTs 2008
LiVE Tool Contest

4th International Conference on Graph Transformation
Leicester (UK), September 12-13, 2008

Contest Chair: Pieter Van Gorp, University of Antwerp
Ranking Chair: Arend Rensink, University of Twente

http://www.fots.ua.ac.be/events/grabats2008/
(Domain independent) Challenge 1a: Instance Generation
http://alloy.mit.edu/ = example outside the GT community
ASSIGNMENT 2
Generate all possible instances...

... of **Presentation**, **Session**, **Person**, **Slot**

✔ with adaptable restrictions, like
  - there should be exactly 5 persons,
  - there should be exactly 3 session,
  - there should be exactly 6 slots,
  - there should be three papers per session,
  - at least one person is chair and presents two papers,
  - at least one person is presenting in 2 different sessions

✔ Visualization of the graph?
✔ Ability to isolate fragments?
(Domain independent) Challenge 1b: Property Verification

http://alloy.mit.edu/ = example outside the GT community

pred conflicting[p1,p2:Presentation] {
    p1!=p2 and ( 
        p1.presenter=p2.presenter or ( 
            p1.session!=p2.session and ( 
                p2.presenter = p1.session.chair or 
                p1.session.chair=p2.session.chair 
            )
        )
    )
}

fact scheduleWithoutConflicts { 
    no disj pres1, pres2: Presentation | 
    pres1.when=pres2.when and 
    conflicting[pres1, pres2] 
}

/* check that conflicting is OK */
assert conflictingReflexive { 
    all p1, p2: Presentation | 
    conflicting[p1,p2] 
    <=> conflicting[p2,p1] 
}

/* check that allocation is OK */
assert noSimultaneousPresentation { 
    no disj pres1, pres2: Presentation | 
    pres1.presenter=pres2.presenter and 
    pres1.when=pres2.when 
}
ASSIGNMENT 3
Property Verification

✔ Can you verify that the instances generated by your solution to ASSIGNMENT 2 do not violate... (see previous slide)?
  ➔ conflictingReflexive
  ➔ noSimultaneousPresentation

✔ What other properties can you verify?

✔ To what scale can you guarantee the properties in general?
  ✔ Independent of the example configuration
  ✔ Property of your transformation rules, tool, ...

  ➔ How many instances of Person, Presentation, Session, Slot, ...?
Case Study: Functional Extension

- **Reservable** (bookable room on a slot)
  - Parallel sessions: more than 1 Reservable per slot
- **Block** (unit of Reservables between coffee breaks)

✗ Do not generate double bookings!
ASSIGNMENT 4

Generate all possible instances...

... of Presentation, Session, Person, Slot, Reservable, Block with adaptable restrictions, like:

✔ (see ASSIGNMENT 2)
✔ over the complete conference
   there are never more than 2 rooms available,
✔ on the first two days, there are 2 rooms before noon, one room in the afternoon, ...
✔ (impress us!)

Again:
✔ Visualization of the graph?
✔ Ability to isolate fragments?
Challenge 2: Concrete Syntax / GUI / User Interaction / ...

How user-friendly is your solution? with regards to

✓ entering presentations, chairs, ...
✓ setting restrictions on room availability
✓ changing the schedule manually
   ✗ Without introducing conflicts?
✓ ...

Create 3 (up to 10) screenshots and a README.txt to illustrate:

- ✔ entering presentations, chairs, ...
- ✔ setting restrictions on room availability
- ✔ changing the schedule manually
- ❌ Without introducing conflicts?
- ✔ ...

ASSIGNMENT 5
Concrete Syntax / GUI / User Interaction / ...
(Domain independent) Challenge 3: Integration of “Independent” Modules

\[
= (\text{out-place}) \text{ MAPPING PROBLEM} \Rightarrow \text{AGG, declarative ATL, ...}
\]


### Presentation

**Pair of presentations that cannot be scheduled in parallel**

- 2 papers from the same presenter
- 2 papers in different sessions with the presenter from one paper being the chair of the other paper's session
- 2 papers in different sessions with the same chair
Graph Coloring

Assign a color to each node, such that all neighboring nodes have a different color.

! Reuse colors where possible...

=> Construct a graph that is k-colorable when the conference can be scheduled in k sequential slots

Use of http://www.cs.ualberta.ca/~joe/Coloring/index.html#Graph.Colorers allowed
Format/Examples: http://mat.gsia.cmu.edu/COLOR/instances.html
Can you integrate with 3GL (Java, ...) libraries to assign a minimal coloring?
Mapping Back

• Extra grammar:
  - Map Presentations corresponding to the colored Nodes to Slots
  - Nodes with same color => Presentations can be on same Slot
    » BUT: preserve Session principle consecutive slots!
  - Traceability links to keep track which Presentations are already scheduled so far
ASSIGNMENT 6
Out-place Mapping support / Interoperability

Demonstrate reuse/modularity by:

✔ Refactoring the conflict resolution into three steps
  (1) Mapping to GRAPH domain (graph grammar?),
  (2) Computing a Coloring,
  (3) Mapping to Timeslot domain (graph grammar?).
✔ Reusing a 3GL (Java, C++, C#, ...) library for computing a minimal coloring (step 2)

Integration:
✔ In-memory (JMI, EMF, ...)?
✔ Serialized (XMI, GXL, ...)?
✔ ... ?
APPENDIX: larger example

5 persons, 9 presentations, 3 sessions

valid schedule
APPENDIX: larger example (3/3)

valid schedule

= one of many valid allocations

Slots (0,1,2): Presentations (3,5,0),
Slots (3,4,5): Presentations (8,6,7) and (2,4,1)
Move \textit{conflictsWith} to the \textit{Colorability} domain?