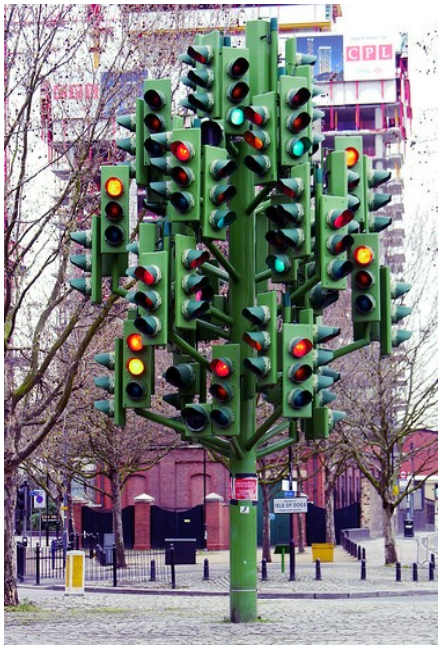
MSBD project presentation

Ákos Nagy

10. Alloy (analysis by bounded exploration): link with Traffic example



MSBD project presentation

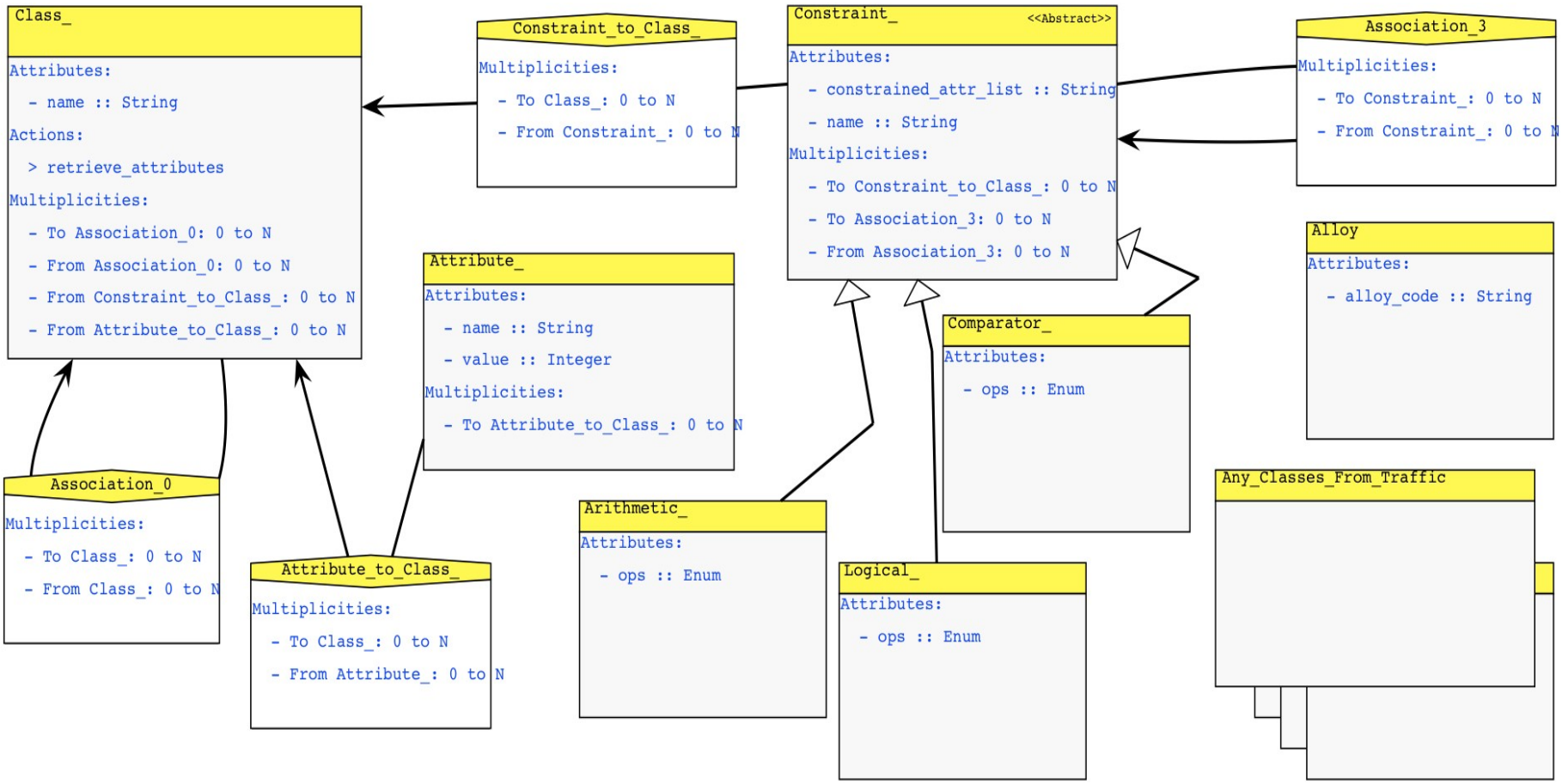
- Problem explanation:
  - meta-model explicitly class and its constraint
  - take the meta-model and transform somehow to Alloy
- AtoM^3 is the working environment
- Why?
  - enable analysis in Alloy
  - reason about models created in “Traffic” formalism

Class and constraint

- Meta-model in Class-Diagram Formalism
- “Class_” named class, “Constraint_” named class and “Attribute_”
- “Constraint_” contains list: operands, different kinds of operators may be applied on them

  a + b < 20 OR True
- “Class_” provides methods to retrieve attribute values which the constraint is applied on

Meta-model


Class_
Attributes:
- name :: String
Actions:
> retrieve_attributes
Multiplicities:
- To Association_0: 0 to N
- From Association_0: 0 to N
- From Constraint_to_Class_: 0 to N
- From Attribute_to_Class_: 0 to N
Constraint_to_Class_
Multiplicities:
- To Class_: 0 to N
- From Constraint_: 0 to N
Constraint_
<<Abstract>>
Attributes:
- constrained_attr_list :: String
- name :: String
Multiplicities:
- To Constraint_to_Class_: 0 to N
- To Association_3: 0 to N
- From Association_3: 0 to N
Association_3
Multiplicities:
- To Constraint_: 0 to N
- From Constraint_: 0 to N
Attribute_
Attributes:
- name :: String
- value :: Integer
Multiplicities:
- To Attribute_to_Class_: 0 to N
Alloy
Attributes:
- alloy_code :: String
Comparator_
Attributes:
- ops :: Enum
Association_0
Multiplicities:
- To Class_: 0 to N
- From Class_: 0 to N
Attribute_to_Class_
Multiplicities:
- To Class_: 0 to N
- From Attribute_: 0 to N
Arithmetic_
Attributes:
- ops :: Enum
Logical_
Attributes:
- ops :: Enum
Any_Classes_From_Traffic

Mapping to Alloy

- Existing tool is: UML2Alloy
  - Here not the case
- Application of Graph Grammar rules

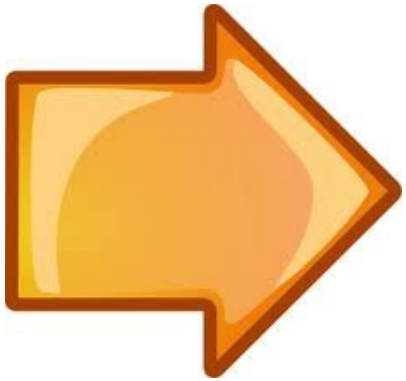
- Match a “Class_” and its associated “Constraint_” on LHS
- On RHS the matched pattern remains the same but in “Action” textual Alloy annotation is created
- Corresponding signature for “Class_” and fact for its constraint

Python code snippet

```
new_code = "sig " + class_node_name + "{"
new_code += attr_names[0]+":Int,"
new_code += attr_names[1]+":Int"
new_code += "}"
```

Result:

```
sig any_name {
    max_capacity:Int,
    current_cars:Int
}
```

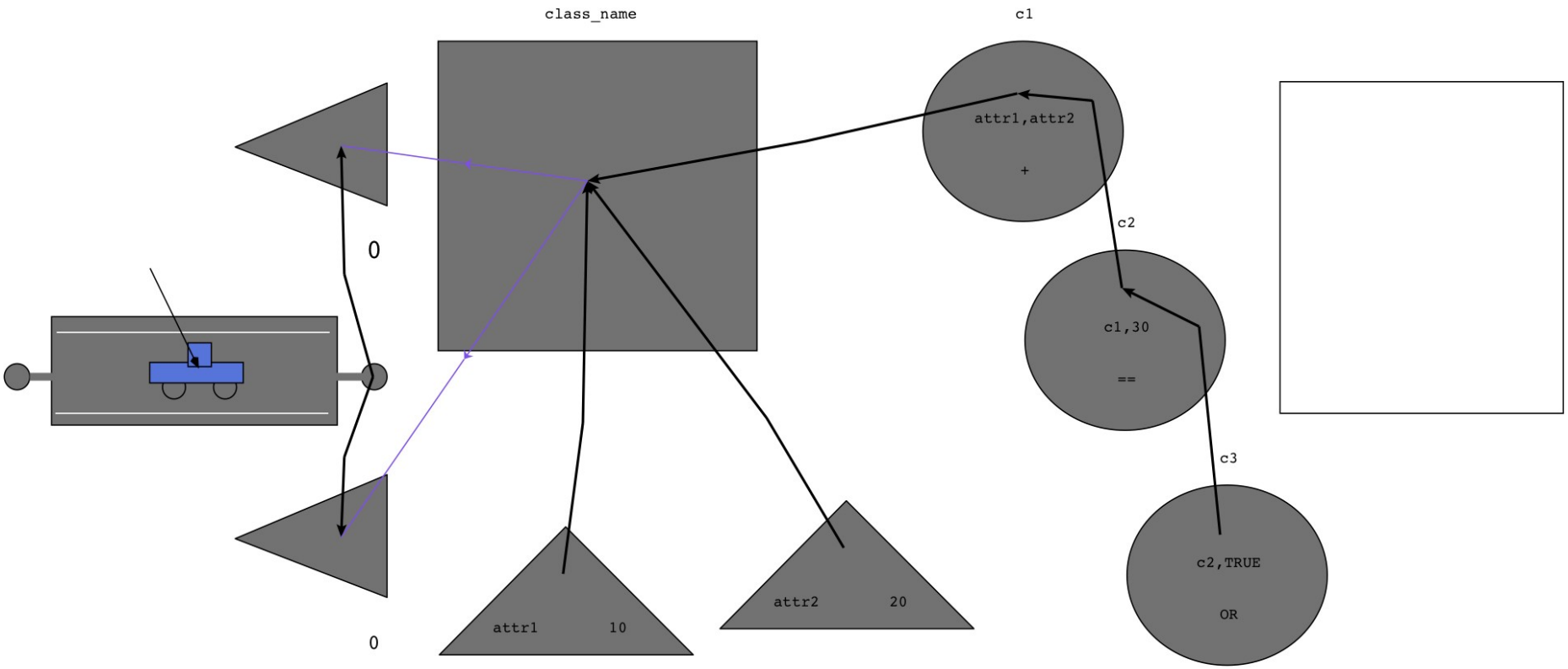


Combine with “Traffic”

- Particle of a vehicle traffic network
- “Class_”, “Attribute_” are added
- “Constraint_” chain is created

- For example: 2 sinks as parking lots
  - if one of them is full, the car has to drive to an other one
  - generate Alloy traces to add several cars

Model in multi-formalism



Alloy code snippet

```
sig cl_name {
   max_capacity:Int,
   car_counter:Int
}

fact con_name {
   all c:cl_name |
   c.max_capacity >= c.car_counter
}

pred show{} run show for 1
```

Future work

- Offer more options to express constraints
- Improve on method to retrieve attribute values
- Extend meta-model with MOF compliant entities
- Consider an other type of mapping to Alloy
- Save output in file
- ...

Thank you for you attention