An Ontology for Construction and Reuse of Knowledge Discovery Workflows

Motivation

- Genesis of this work
  - FP6 SEVENPRO project: “semantic engineering environment”
  - Including relational data mining (mainly from CAD) with ontologies as background knowledge
  - Complex data mining workflows
  - Results of mining must be stored and searchable in a semantic repository

- **Primary objective**:

  enable the workflow planner to reason about which algorithms can be used to produce intermediary or final results required by a specified data mining task

- Other areas of use:
  - retrieval and reasoning about results of data mining
  - information about runtimes for different algorithms/workflows
  - management of available algorithms and their required I/O + parameters
  - management of results and info about experiments
  - annotation of manually created workflows for reuse and possibly meta learning

KD Ontology - Scope

- Focus on relational data mining with background knowledge
- Formalization of background knowledge
- Focus on predictive and descriptive rule learning

- Use cases:
  - SEVENPRO
    - relational rule learning through propositionalization
    - data: annotations of CAD command histories
  - Bioinformatics
    - Subgroup discovery in gene expression data using GO as background knowledge
  - Orange data mining toolkit
    - classical preprocessing, classification, association and visualization algorithms

Competency questions

- Can algorithm A be used on dataset D directly? If not, which preprocessing steps can be used?
- Which DM tasks have been performed on dataset D?
- Given dataset D and background knowledge B, which algorithms can be used to get a decision tree model?
- Retrieve all decision trees for dataset D
- Retrieve all rules for class M in dataset D with support > Y
- Which algorithms were used to classify dataset D? What were their runtimes and accuracy?
- Retrieve all workflows using algorithm A
- What is the influence of language bias on number of rules and efficiency?
- Compare different discretization methods for a particular dataset
- Retrieve all rules containing attribute F

Top level ontologies

- General top level ontologies
  - DOLCE
    - linguistic bias
    - ontology of information objects and plans not directly usable
  - BFO
    - most likely candidate
    - purpose: support domain ontologies developed for scientific research
    - used for bioinformatics investigations
- Workflow representation
  - OWL-S
  - WSMO

Related work

3 ontologies of data mining/knowledge discovery being developed:

OntoDM (Panov et al.,2009)

KDDOnto (Diamantini et al., 2009)

OntoDM (Hilario et al.,2009)

Related work - OntoDM

- attempts to capture minimum information for describing DM investigation
- heavyweight ontology, developed by top-down approach
- algorithm = information content entity, process, realizable entity -> separate specification, implementation, application
- uses an upper level ontology BFO (Basic Formal Ontology) for top-level classes and OBO Relational Ontology to define relations
- fully aligned with top-level structure of OBI
- concepts from draft version of IAO (Information Artifact Ontology) for dealing with information
- supports complex structured data
- process aspect of algorithm can be used for workflows

Related work - KDDOnto

- developed within KDDVM project (2005-present)
- based on DAMON ontology
- level separation: algorithm (described by ontology) - tool - service (described by eWSDL)
- top-level structure given by phases of the DM process
- main purpose: support automatic composition of DM workflows
- core concepts:
  - Algorithm uses Method specifies Task specifies-phase Phase
  - Data - Dataset, GenericParameter, Model
  - Performance
- taxonomy of models
  - focus on classical DM models (classification, regression, clustering)
  - no properties of models defined
- no workflow representation

Related work - DMOnto

- FP7 project e-LICO
- 2 ontologies being developed:
  - DMOnto
  - Planning ontology
- DMOnto
  - purpose: algorithm selection, meta-minig of experimentation records, provide a controlled vocabulary
  - core concepts: Task, Dataset, Method, Model
  - focus on classification
  - algorithm viewed as optimization of some parameters
- Planning ontology
  - core concepts: IObject, MetaData, Operators, GoalDescription
  - SWRL for preconditions and effects

KD Ontology

– main notions : **Knowledge**, **Algorithm, Task, Workflow**

– **Knowledge -** data, background knowledge, models and patterns

– representation language: OWL-DL
  – densely interlinked knowledge structures
  – highly optimized reasoners available (Pellet, RacerPro, Fact++, ...)

– for representation of workflows: OWL-S

key subclasses:

- **Dataset**

```
Knowledge and example some Example
```

- **LogicalKnowledge**

```
subclassOf Knowledge
and hasExpressivity some Expressivity
and hasFormat some KnowledgeFormat
```

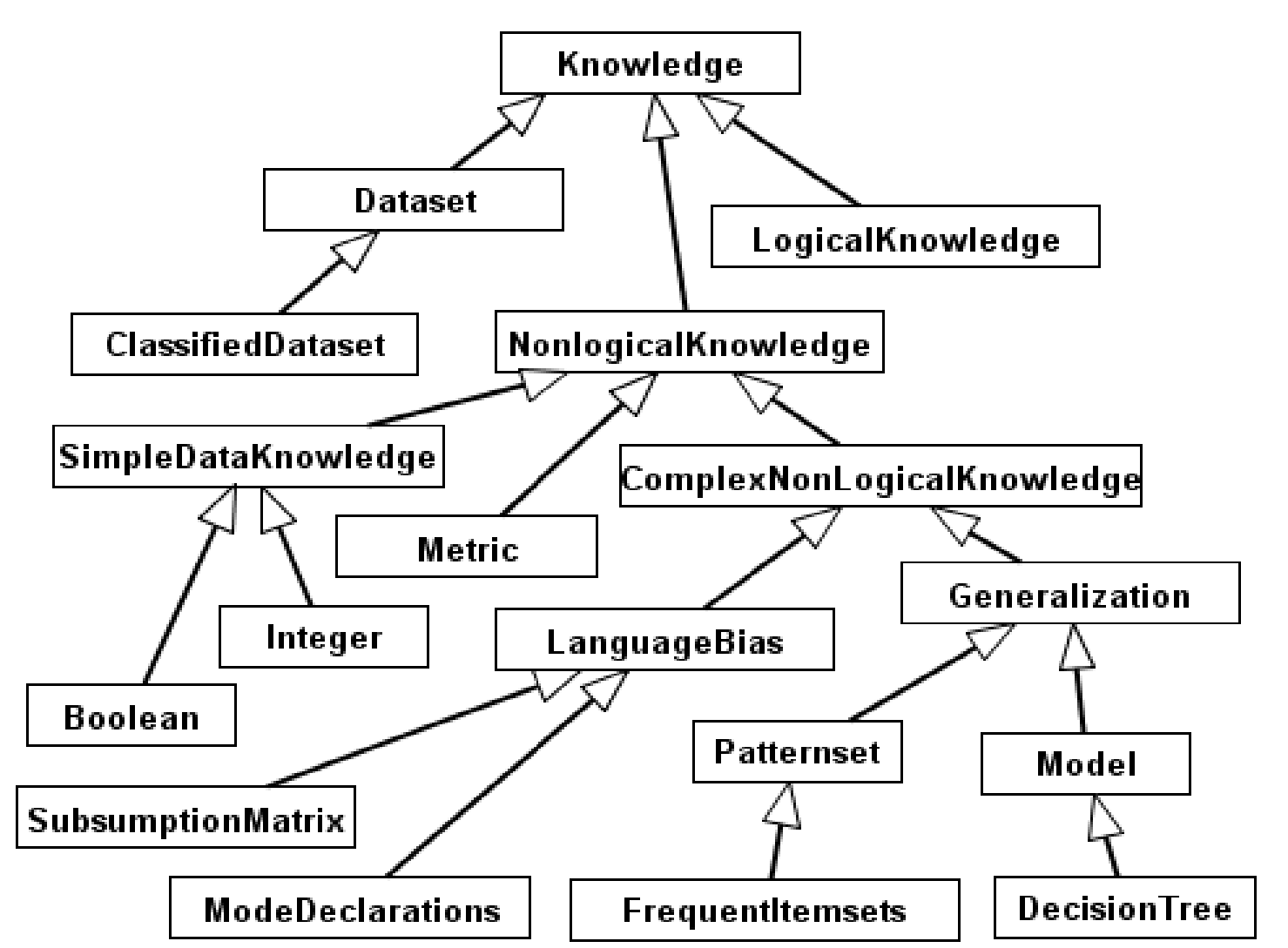
- **NonLogicalKnowledge**

```
Knowledge and not LogicalKnowledge
```

- **Generalization**
  - **Model, Patternset**

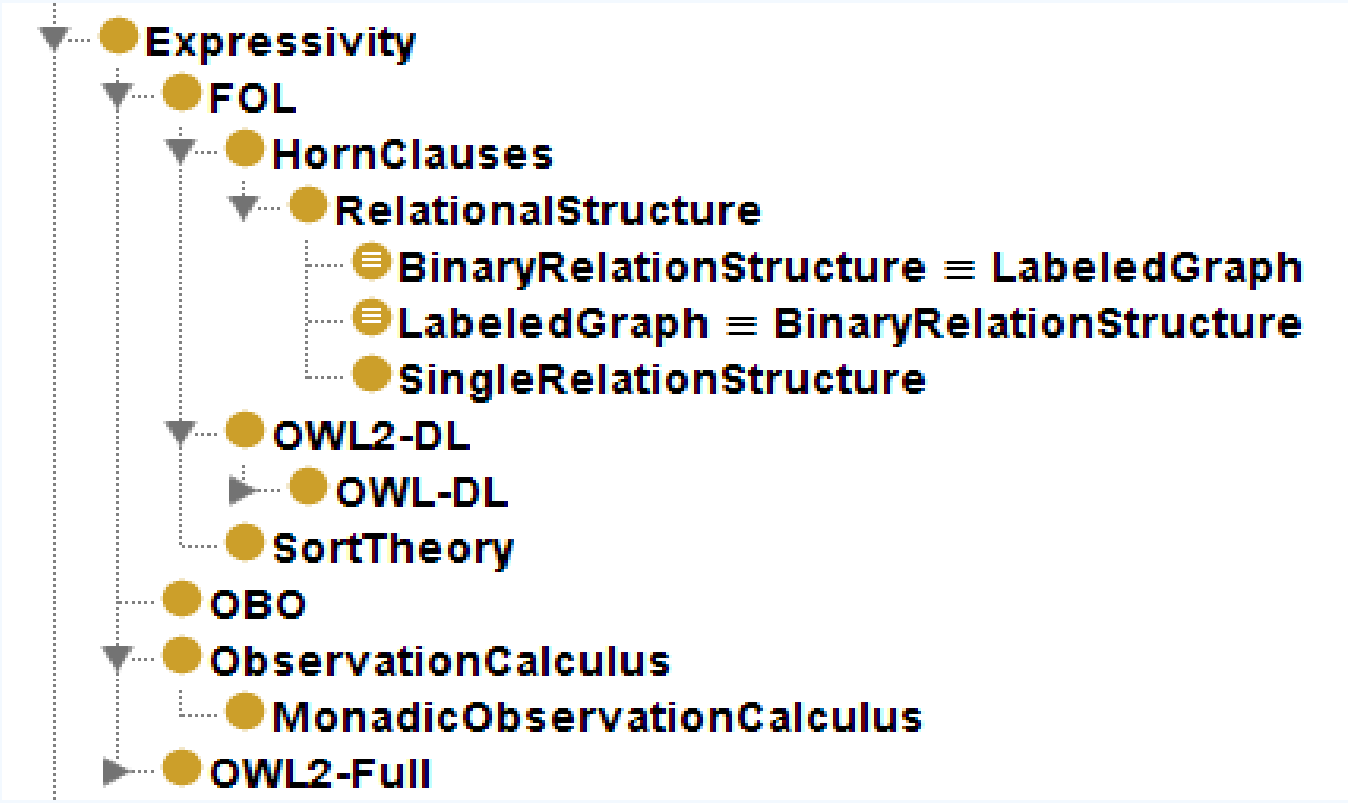
- multiple formats may be attached to each **Knowledge** class
  - each knowledge instance has a specified **KnowledgeFormat**

Knowledge


Knowledge
Dataset
LogicalKnowledge
ClassifiedDataset
NonlogicalKnowledge
SimpleDataKnowledge
ComplexNonLogicalKnowledge
Metric
Generalization
Integer
LanguageBias
Boolean
Patternset
Model
SubsumptionMatrix
ModeDeclarations
FrequentItemsets
DecisionTree

Logical Knowledge and Expressivity

Logical Knowledge ≡ Knowledge
and hasExpressivity some Expressivity
and hasFormat some KnowledgeFormat



Algorithm

- all executable routines that can be used in a knowledge discovery process, like inductive algorithms and knowledge format transformations
- defined by its input and output knowledge specifications and by its parameters
- **`Algorithm`** defined as an equivalent class of the OWL-S **`Process`**
- configuration of an executable algorithm is an instance of its subclass **`NamedAlgorithm`**
- **`Workflow`** is defined as an equivalent class of the OWL-S **`CompositeProcess`**

Algorithm annotation example

$$\{\text{Apriori}\} \sqsubseteq \text{NamedAlgorithm} \sqcap \exists \text{output} \cdot \{\text{Apriori-O-Rules}\} \sqcap \exists \text{input} \cdot \{\text{Apriori-I-Dataset}\} \sqcap \exists \text{input} \cdot \{\text{Apriori-I-MinSupport}\} \sqcap \exists \text{input} \cdot \{\text{Apriori-I-MinConfidence}\}$$

$$\{\text{Apriori-I-Dataset-Range}\} \equiv \text{isRangeOf} \cdot \{\text{Apriori-I-Dataset}\} \equiv \text{Dataset} \sqcap \forall \text{hasFormat} \cdot \{\text{TAB}\} \sqcap \forall \text{hasExpressivity} \cdot \text{SingleRelationStructure} \sqcap \forall \text{hasAttributesType} \cdot \{\text{dDiscrete}\}$$

$$\{\text{Apriori-O-Rules-Range}\} \equiv \text{isRangeOf} \cdot \{\text{Apriori-O-Rules}\} \equiv \text{Patternset} \sqcap \forall \text{contains} \cdot \text{AssociationRule}$$

Workflow Construction

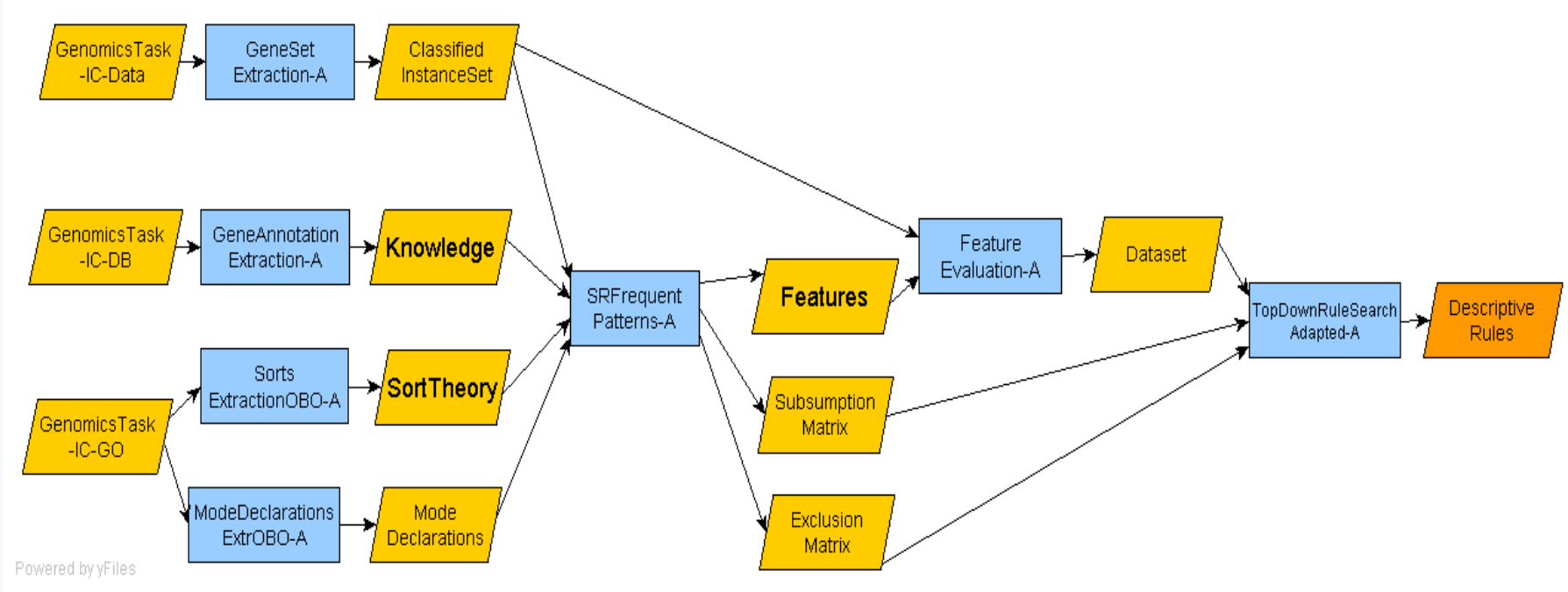
Automatic workflow construction

1. Classifying the KD ontology
2. Generating a plan using a planning algorithm
3. Storing the generated abstract workflow in form of semantic annotation
4. Instantiating the abstract workflow with specific algorithm configurations available in the KD ontology

Planning Algorithm

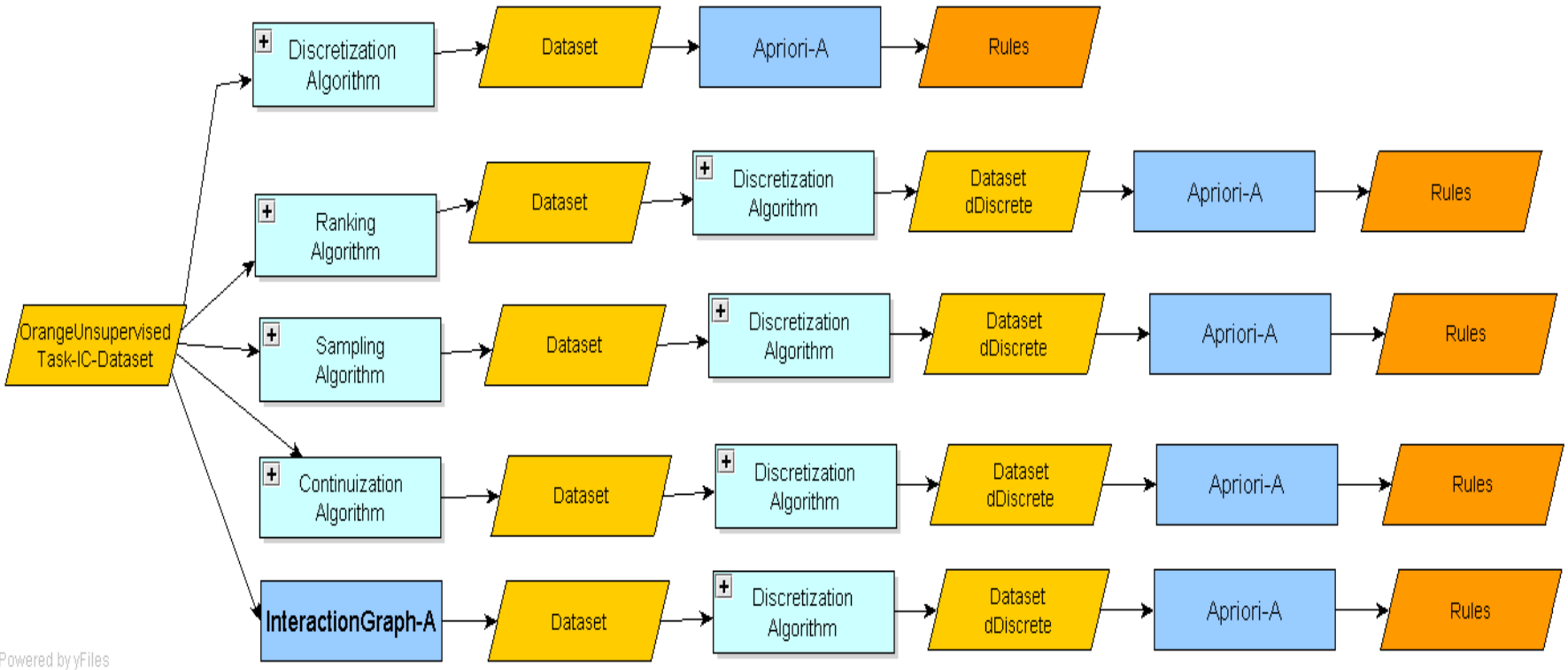
- based on Fast-Forward planning system (Hoffman, 2001)
- enforced hill climbing algorithm to perform forward state space search
- goal distances estimated using relaxed GRAPHPLAN
  - i.e. ignoring delete lists of the operators
- obtains possible next steps by querying the KD ontology using SPARQL
- version with and without exploiting algorithms hierarchy implemented

Workflow Example - Bioinformatics



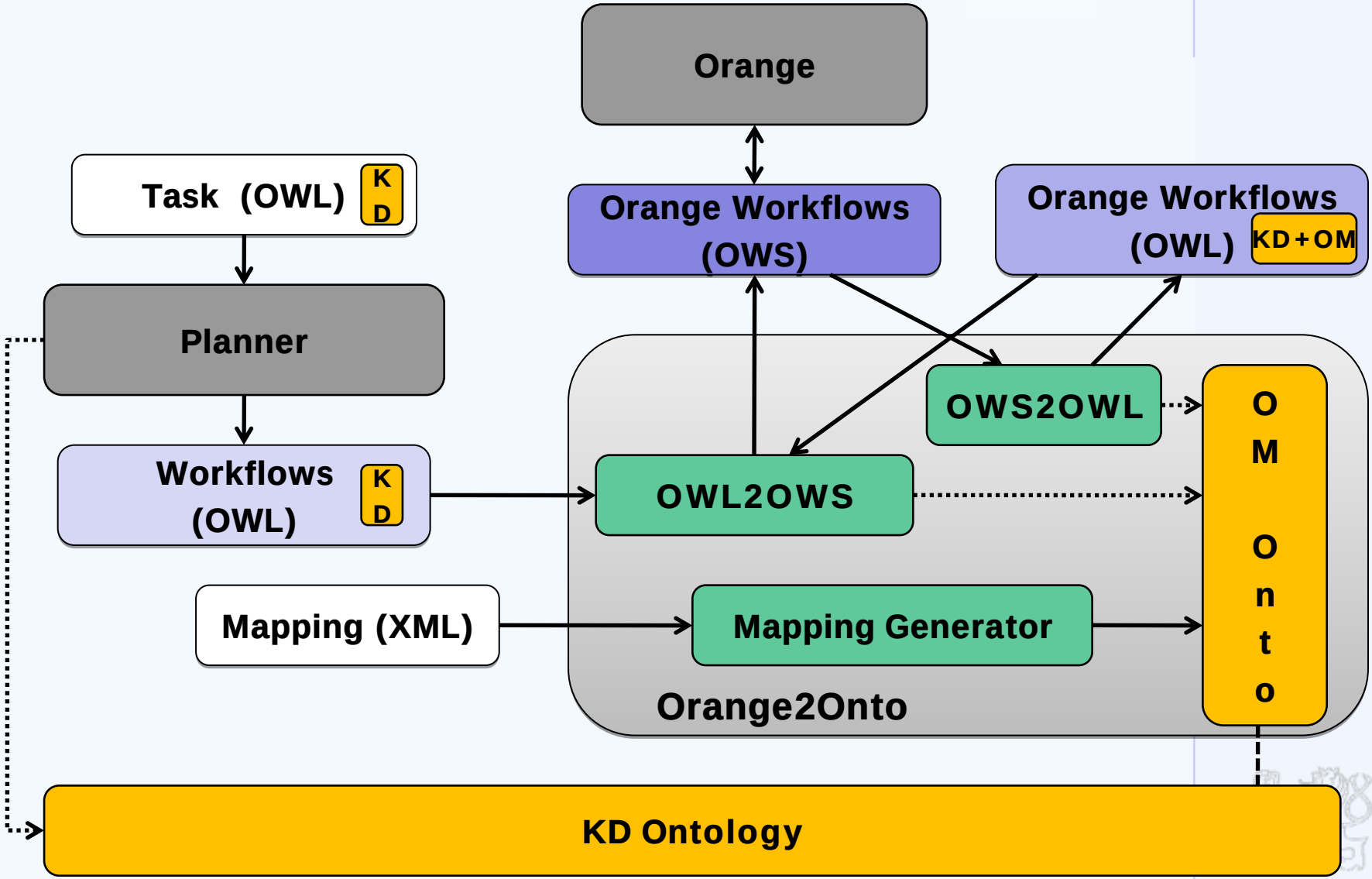
- Task: producing rules distinguishing between two types of leukemia
- Input:
  - gene expression data
  - annotations of genes using the GO ontology

Workflow Example - Orange



- Task: producing association rules from a propositional dataset
- Input: propositional dataset with both discrete and continuous attributes, without class attribute

Integration into Orange



Orange to KD Ontology Mapping

Intelligent Data Analysis
RESEARCH GROUP

Orange-Map (OM) ontology

- to capture formally the mapping between the internal Orange representation and the representation of algorithms using the KD ontology
- defines templates for mapping of algorithms, data and parameters

$$\begin{aligned} \texttt{OrangeRadioParamMapping} \sqsubseteq\ & \exists \texttt{parameter} \cdot \{\texttt{kd:AlgorithmParameter}\} \\ & \sqcap\ \exists \texttt{radioValue} \cdot \texttt{OrangeRadioValue} \\ & \sqcap\ \exists \texttt{orangeAlias} \cdot \texttt{string} \end{aligned}$$

$$\begin{aligned} \texttt{OrangeRadioValue} \sqsubseteq\ & \exists \texttt{paramURI} \cdot \{\texttt{anyURI}\} \\ & \sqcap\ \exists \texttt{rbNumber} \cdot \texttt{int} \end{aligned}$$

Annotation of a New Algorithm in Orange

1. create instances of AlgorithmParameter for all inputs and outputs
2. create an instance of NamedAlgorithm
3. for each instance of AlgorithmParameter create a class defining its range

   (if not yet defined, add the necessary subclasses of Knowledge - this should be required only when a new type of algorithm is added)
4. create an XML file defining a mapping between the algorithm representation in Orange and in the KD ontology
5. run a script for generating a mapping using the OM ontology

Experimental Results

– planner with and without exploiting the algorithms hierarchy
– 2 tasks:
  – discovering descriptive rules in the genomics domain
  – discovering association rules
– KD ontology: ~500 classes and ~500 individuals
  (including annotations of algorithms)

Task	No. of algorithms	Planner		HierarchyPlanner	
		Preproc.	Planning	Preproc.	Planning
GEN	71	72	0.85	115	0.56
GEN	99	104	1.12	155	0.57
ASSOC	71	94	27.29	125	25.15
ASSOC	99	98	107.55	153	24.35

Future Work

- Refining and extending KD ontology
- Explore possibilities of treating the individual KD algorithms as web services
- Incorporating more complex constraints and user preferences for more effective workflow ranking