Hybrid Information Extraction Systems

Class 3: Rule-Based IE

Pablo Ariel Duboue, PhD

30va Escuela de Ciencias Informaticas (ECI)
Facultad de Cs. Exactas y Naturales UBA

Why Rules?

- Understandable
  - Debuggable
- Continuous Quality Improvement

Key: Rule Language

- Two approaches:
  1. Regular Expressions
  2. Wrapper-induction

Example

```
// his father was
Rule: relative_parent2
(
  ({Person})+ :person_self
  ({Token.string == "father"} | {Token.string == "mother"})
  ({Token.string == ","})*
  ({Person})+ :person_parent
  ({Token.string == ","})*
  {Token.string == "was"}
)
-->
{
    progenie.L.addFact("c-relative",
       new Object[]{"person_self",
         progenie.L.getClassStrings(inputAS,bindings,"person_self")},
       new Object[]{"person_parent",
         progenie.L.getClassStrings(inputAS,bindings,"person_parent")});
}
```

Rule Drawbacks

- Costly
- Evaluation still requires annotation
- Tempting of not doing any annotation
    - Big mistake

Regular Expression Rules for IE

- REs are quintessential CS
- REs over types seems the right tool for the job

R.E. Rules Example

```
Phase:  ProGenIE
Input:  THEME TARGET LOCATION

Rule:  r1
(
  ({THEME})  :person
  {TARGET.string == "born"}
  ({LOCATION})  :loc
)
-->
{
    System.out.println("Born␣event");
    gate.AnnotationSet person = (gate.AnnotationSet)bindings.get("person");
    gate.AnnotationSet loc = (gate.AnnotationSet)bindings.get("loc");
    System.out.println("Person:"+
      ((Annotation)person.iterator().next()).getFeatures().get("string"));
    System.out.println("Loc:"+
      ((Annotation)loc.iterator().next()).getFeatures().get("string"));
}
```

JAPE

- The Java Annotations Pattern Engine
- Regular Expressions (finite state transduction) over GATE annotations.
- DSL with rules:
  - LHS with patterns over annotations and their features
  - RHS is Java code (compiled on-the-fly using tools.jar in the JDK distribution)

https://gate.ac.uk/sale/thakker-jape-tutorial/index.html

Example

```
Phase:  Event
Input:  Token  Lookup  Location  Organization  Date  JobTitle  Person
Options:  control=appelt

Rule:  e_r1
(  ({Token.category == "DT"})?
  (({Organization})+)?
  (  ({Token.category == "JJ"})*
    ({Token.string == "chemical"}{Token.string == "weapons"})
    ({Token.string == "actions"}|{Token.string == "action"}|
     {Token.string == "attacks"}|{Token.string == "attack"})
  )
  {Token.string == "against"}
  ({Token.category == "DT"})?
  (({Location})+|
   ({Organization})+|
   (({Lookup.majorType == citizenship})+
    ({Token.string == "forces"}|{Token.string == "force"}|{Token.string == "army
    {Token.string == "troops"}|{Token.string == "soldiers"}|{Token.string == "n
    {Token.string == "interests"}|{Token.string == "citizens"}|{Token.string ==
    {Token.string == "embassies"}|{Token.string == "consulate"}|{Token.string ==
    {Token.string == "diplomat"}|{Token.string == "diplomats"}))|
   (({Location})+ {Token.string == "'s"}
    {Token.string == "secret"}{Token.string == "service"}{Token.string == "head
   ):target
  ( {Token.string == "in"} (({Location})+) :place )?
  ( ({Token.string == "in"}|{Token.string == "during"}) (({Date})+) :date )?
):event --> { /* ... */ }
```

RefO

- Regular Expressions for Objects
  - Developed in Cordoba by Machinalis
  - Python
  - The base of one of the IEPy subsystems

https://github.com/machinalis/refo/

RefO Example

```
class W(Predicate):
    def __init__(self, token=".*", pos=".*"):
        self.token = re.compile(token + "$")
        self.pos = re.compile(pos + "$")
        super(W, self).__init__(self.match)

    def match(self, word):
        m1 = self.token.match(word.token)
        m2 = self.pos.match(word.pos)
        return m1 and m2
rules = [
    Rule(condition=W(pos="NUMBER") + W(pos="UNITS_FEET"),
         action=feet_to_mt),
    Rule(condition=Plus(W(token="[^A-Z].*", pos="NNP")),
         action=capitalize_name),
]

sentence = "My|PRP␣friend|NN␣john|NNP␣smith|NNP␣is|VBZ␣2|NUMBER␣" +\
           "feet|UNITS_FEET␣taller|JJR␣than|IN␣mary|NNP␣Jane|NNP"
sentence = [Word(*x.split("|")) for x in sentence.split()]

for rule in rules:
  rule.apply(sentence)
```

IEPy

- A framework for building Information Extraction systems using two approaches:
  - Active Learning (annotate and learn, no coding necessary)
  - Rule-based (using RefO)
- Developed by Machinalis in collaboration with the NLP Group at UNC-FaMAF

https://github.com/machinalis/iepy

IEPy Example

```
@rule(True)
def born_date_in_parenthesis(Subject, Object):
    """
    Ex: Gary Sykes (Born 13 February 1984) is a British super featherweight boxer.
    """
    anything = Star(Any())
    born = Star(Pos(":")) + Question(Token("Born") | Token("born")) + \
      Question(Token("c."))
    entity_leftover = Star(Pos("NNP"))
    return Subject + entity_leftover + Pos("-LRB-") + born + Object + Pos("-RRB-") + \
           anything

@rule(True)
def born_two_dates_in_parenthesis(Subject, Object):
    """
    Ex: James Cunningham (born 1973 or 1974) is a Canadian stand-up comedian and
        TV host.
    """
    anything = Star(Any())
    born = Question(Token("Born") | Token("born"))
    entity_leftover = Star(Pos("NNP"))
    subject = Subject + entity_leftover
    or_object = (Object + Token("or") + Pos("CD") |
                 Pos("CD") + Token("or") + Object)
    return subject + Pos("-LRB-") + born + or_object + Pos("-RRB-") + anything
```

Drawbacks

- Corner cases will kill your program
- They work, and there are competing approaches

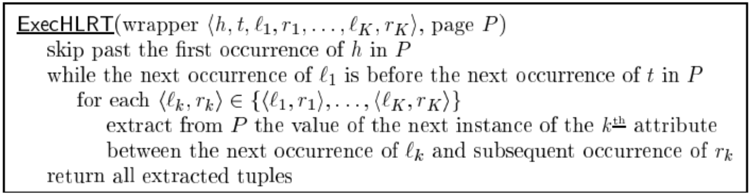
Anchor-based Rules for IE

- Alternative to transducing the whole sequence of characters / tokens / annotations
- Identify key elements that are rare enough and reliable as a start for processing
  - Anchors
- Structure the processing around further conditions around these elements
  - The context around the anchor can then be modeled using REs

Wrapper Induction

- Anchor-based IE is rooted in Nicholas Kushmeric thesis from 1997 UW
  - Wrapper Induction for Information Extraction
  - The wrapper induction uses inductive learning
- Learning to extract information from Web pages
  - Kushmeric uses "Information Extraction" to refer to what we now call Web scraping

Example



```
ExecHLRT(wrapper $\langle h, t, \ell_1, r_1, \ldots, \ell_K, r_K \rangle$, page $P$)
    skip past the first occurrence of $h$ in $P$
    while the next occurrence of $\ell_1$ is before the next occurrence of $t$ in $P$
        for each $\langle \ell_k, r_k \rangle \in \{\langle \ell_1, r_1 \rangle, \ldots, \langle \ell_K, r_K \rangle\}$
            extract from $P$ the value of the next instance of the $k^{\underline{\text{th}}}$ attribute
            between the next occurrence of $\ell_k$ and subsequent occurrence of $r_k$
    return all extracted tuples
```

from Kushmeric (1997), Figure 4.1 (a)

RuTA at Length

- Slides from GSCL 2013 by Peter Klügl
  - https://uima.apache.org/downloads/gscl2013/2013-GSCL-Ruta.pdf

Rule Induction

- WHISK
- LP^2

Generalities

- The key aspect of rule-learning algorithms is the rule-language used
- Many algorithms start from a training instance and generalize it following different criteria
- They then evaluate the different generalizations
    - Keeping a good balance of generalization power and correctness

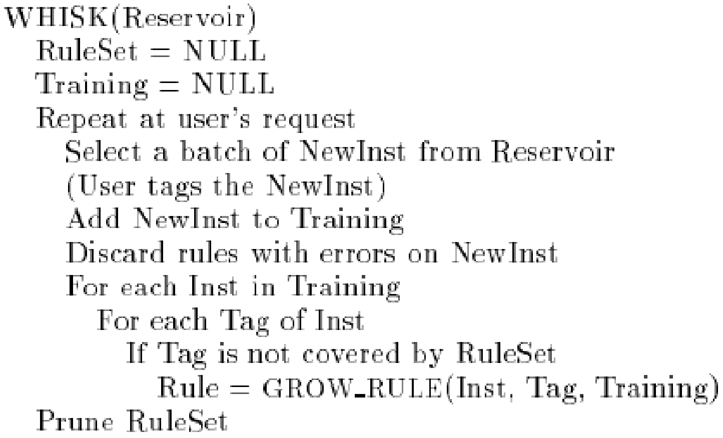
WHISK

- *"Learning Information Extraction Rules for Semi-structured and Free Text"* by Stephen Soderland (Machine Learning, 1999)
- Simple "regular expression like" language (bounded * operator)
- Applicable to both limited IE and Web scraping
- 1245 citations
- Algorithm selects which instances to ask for annotation

WHISK Rule Language

- Wilcard
- Words or lists of words
- Capture groups
  - Words
  - Digits
  - Numbers
- * (Digit) 'BR' * '$' (Number)
  - Renovated Apt 2BR available imm $2500 OBO

WHISK Algorithm



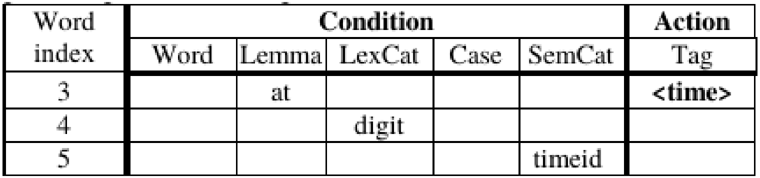
```
WHISK(Reservoir)
  RuleSet = NULL
  Training = NULL
  Repeat at user's request
    Select a batch of NewInst from Reservoir
    (User tags the NewInst)
    Add NewInst to Training
    Discard rules with errors on NewInst
    For each Inst in Training
      For each Tag of Inst
        If Tag is not covered by RuleSet
          Rule = GROW_RULE(Inst, Tag, Training)
  Prune RuleSet
```

From Soderland (1999), Figure 12

LP^2

- *"Adaptive Information Extraction from Text by Rule Induction and Generalisation"* by Fabio Ciravegna (IJCAI, 2001)
- Learn to insert tags by generalizing examples
  - Generalizing using linguistic information
    - Part-of-speech
    - Lists of terms (e.g., titles)
  - Open and close tags by separate rules
- Keep only high precision rules
  - Improve over less precise rules by keeping context

LP^2 Rule Language

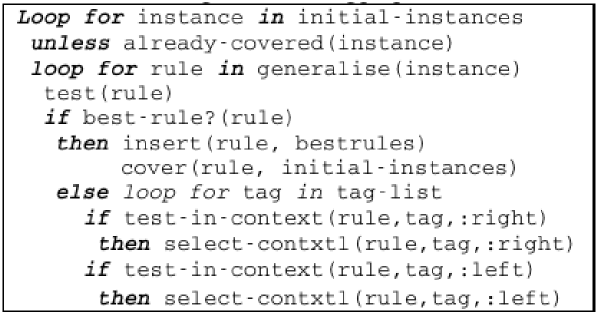


Word index	Condition					Action
	Word	Lemma	LexCat	Case	SemCat	Tag
3		at				**<time>**
4			digit			
5					timeid	

From Ciravegna (2001), Table 2

- Plus correction rules
  - Similar to the base rules but change an existing tag

LP^2 Algorithm



```
Loop for instance in initial-instances
 unless already-covered(instance)
 loop for rule in generalise(instance)
  test(rule)
  if best-rule?(rule)
   then insert(rule, bestrules)
        cover(rule, initial-instances)
   else loop for tag in tag-list
     if test-in-context(rule,tag,:right)
      then select-contxtl(rule,tag,:right)
     if test-in-context(rule,tag,:left)
      then select-contxtl(rule,tag,:left)
```

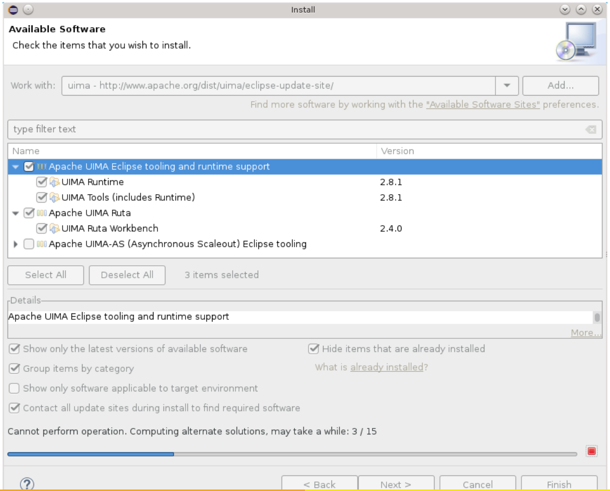
From Ciravegna (2001), Figure 3

RuTA in the End-to-End Case Study

- Installing RuTA Workbench:
  - Old version of Eclipse (Mars)
  - Set up a sandbox to use with the tooling
    - The sandbox project depends on the main (Maven) project
  - Copy the rules scripts when satisfied

Installation

- Eclipse > Help > Install New Software...



Available Software

Check the items that you wish to install.

Work with: uima - http://www.apache.org/dist/uima/eclipse-update-site/

Find more software by working with the "Available Software Sites" preferences.

Name	Version
☑ Apache UIMA Eclipse tooling and runtime support	
☑ UIMA Runtime	2.8.1
☑ UIMA Tools (includes Runtime)	2.8.1
☑ Apache UIMA Ruta	
☑ UIMA Ruta Workbench	2.4.0
☐ Apache UIMA-AS (Asynchronous Scaleout) Eclipse tooling	

3 items selected

Details

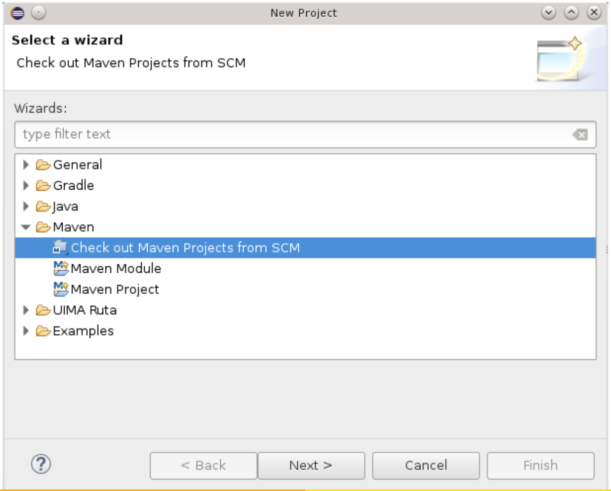
Apache UIMA Eclipse tooling and runtime support

☑ Show only the latest versions of available software
☑ Hide items that are already installed
☑ Group items by category
What is already installed?
☐ Show only software applicable to target environment
☑ Contact all update sites during install to find required software

Cannot perform operation. Computing alternate solutions, may take a while: 3 / 15

Installation (cont.)

- File > New > Project...


New Project

Select a wizard

Check out Maven Projects from SCM

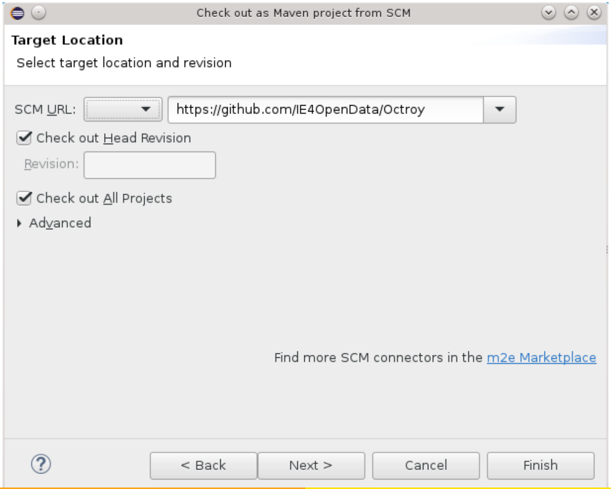
Wizards:

type filter text

- General
- Gradle
- Java
- Maven
  - Check out Maven Projects from SCM
  - Maven Module
  - Maven Project
- UIMA Ruta
- Examples

Installation (cont.)

- (might need to install egit from m2e marketplace)


Check out as Maven project from SCM

Target Location

Select target location and revision

SCM URL: https://github.com/IE4OpenData/Octroy

☑ Check out Head Revision

Revision:

☑ Check out All Projects

Advanced

Find more SCM connectors in the m2e Marketplace

Installation (cont.)

- On checkout, Eclipse might complain the generated clases are not found
- Right click on the project and select Maven in the contextual menu
  - Choose "Update project..."
  - Use default options
- Switch to RuTA Workbench perspective:
  - Window > Perspective > Open Perspective > Other... (select UIMA RuTA)
- Create sandbox project:
  - File > New > UIMA RuTA Project...
  - Set name to OctroySandbox
- Set the dependency:
  - Select the sandbox project and go to Project > Preferences > Project References
  - Check octroy

Sandbox Project

- Right click on script folder > New > UIMA RuTA Package
  - org.ie4opendata.octroy
- Open the package and right click on octroy > New > UIMA RuTA File
  - Reason.ruta
- Copy the dev100 into input

Types being Extracted

- Implement the baseline reason rules in RuTA

Some Rules

- CW+ @CompanyTrailing{->MARK(Company,1,2)};
- "la" "firme" W+{-CONTAINS(CompanyTrailing)} CompanyTrailing{->MARK(Company,3,4)};
- Company "concernant" ANY[2,50]{-REGEXP("[.,;]")->MARK(Reason)};

Evaluation

- RuTA Workbench has an evaluation environment
- Needs output XMI files
  - Executes the script and compare them to the annotated files
  - XMI files can be annotated using the UIMA Eclipse tooling