Object-Oriented KRR with Flora-2 – Access Control and Privacy Control Example

CSE 505 – Computing with Logic
Stony Brook University
http://www.cs.stonybrook.edu/~cse505
Knowledge Representation and Reasoning with Flora-2

- Example: Social networks have complex information access and privacy policies.
  - In this example, we model such a network and use it to create various views based on these policies for friends, public, private and groups
- A user has several properties with various access policies:

  ```
  User[]
  // String_Object, Gender_Object,
  // Location, and others are subclasses of
  // Access_Controlled and can have access
  // permissions
  first_name => String_Object,
  last_name => String_Object,
  profile => Profile_Object,
  ```
Knowledge Representation and Reasoning with Flora-2

cover => Cover_Object,
gender {0..1} => Gender_Object,
email => String_Object,
birthday {0..1} => Birthday_Object,
userid {1..1} => String_Object,
user_name => String_Object,
education_history => School_Attendance,
job_history => Job_Held,
relationship => Relationship,
location => Location,
content => Content,

// created objects: Page, Event, Group
creates => Created_Objects,
likes => LikeContent,
timeline => Timeline_Object
transaction => Transaction,
A class of all objects to which access can be controlled:

\[
\text{Access\_Controlled[ |} \\
\text{read(Access\_Entity)} \Rightarrow \text{boolean,} \\
\text{write(Access\_Entity)} \Rightarrow \text{boolean,} \\
\text{find(Access\_Entity, Access\_Entity)} \Rightarrow \text{boolean,}
\]

default values inherited by all objects unless overwritten:

\[
\text{read(?) } \Rightarrow \text{false,}
\text{write(?) } \Rightarrow \text{false,}
\text{find(?, ?) } \Rightarrow \text{false}
\]

\[\].
Knowledge Representation and Reasoning with Flora-2

- Entities that can have access control privacy control:
  
  \[
  \{\text{male, female}\} : \text{Gender\_Object}.
  \]

  \[
  \text{Gender\_Object} : : \text{Access\_Controlled}[| \\
  \quad \text{value} => \text{Gender\_Type}
  |
  |].
  \]

  \[
  \text{Birthday\_Object} : : \text{Access\_Controlled}[| \\
  \quad \text{value} => \text{date}
  |
  |].
  \]

  \[
  \{\text{spouse, friend, girlfriend, parent, child}\} : \text{Relationship}.
  \]

  \[
  \text{Relationship} : : \text{Access\_Controlled}[| \\
  \quad \text{type} => \text{Relationship\_Type}, \\
  \quad \text{person} => \text{User}
  |
  |].
  \]
Attendance::Access_Controlled[]
    institution {1..1} => \string,
    start {0..1} => \date,
    end {0..1} => \date,
    address => Address
].
{School_Attendance,Job_Held} :: Attendance.

School_Attendance[]
    status => \string,
    level => \string
].

Job_Held[]
    position => \string
].
Location::Access_Controlled[]
  country => string,
  region => string,
  city => string,
  latitude => decimal,
  longitude => decimal
].

// general address
Address :: Location[]
  street {0..1} => string,
  number {0..1} => string,
  apartment {0..1} => string,
  zipcode {0..1} => string
].
Knowledge Representation and Reasoning with Flora-2

\{Photo, Video, Comment, Note, Group, Event, Link, StatusPost, Message\} :: Content.

Content::Access_Controlled[|
   author {1..1} => User,
   creation_time {1..1} => \datetime,
   description => \string,
   comment => Comment,
   tags => Tag,
   audience => Audience
|].

Timeline[|
   content => Content
|].

\{Public, Friends, FriendsofFriends, OnlyMe\} : Audience.
Social networks also provide access to purchasing products and store information about login and transactions:

```
Product[]
    name    {1..1} => \string,
    owner    => \string,
    price    {1..1} => \decimal,
    description    => \string
|].
```

```
Transaction::Access_Controlled[]
    account => Account,
    time    => \datetime,
    product    => Product,
    amount    => \decimal
|].
```
Account[]
    bank_name {1..1} => string,
    account_number {1..1} => string,
    created {1..1} => date,
    owner {1..*} => User
].

Cookie[]
    device => string,
    browser => string,
    os => string,
    location => Location,
    IP_address => string,
    login => string,
    time => datetime
].
Knowledge Representation and Reasoning with Flora-2

```
paul:User[
    relationship -> paul_mary_relationship,
    relationship -> paul_john_relationship,
    relationship -> paul_mike_relationship,
    relationship -> paul_jack_relationship,
    timeline -> paul_timeline
].
paul_mary_relationship:Relationship[
    type -> spouse,
    person -> mary
].
paul_john_relationship:Relationship[
    type -> child,
    person -> john
].
```
paul_jack_relationship:Relationship[
    type -> friend,
    person -> jack
].

paul_timeline:Timeline[
    content -> post1,
    content -> photo1
].

post1:Post[
    value -> "I am in Berlin",
    audience -> Family
].

photo1:Photo[
    imageName -> "Berlin 1",
    audience -> Friends
].
mary: User[
    relationship -> mary_paul_relationship,
    relationship -> mary_john_relationship,
    relationship -> mary_mike_relationship,
    relationship -> mary_jack_relationship,
    relationship -> mary_jane_relationship,
    timeline -> mary_timeline
].

mary_paul_relationship: Relationship[
    type -> spouse,
    person -> paul
].

mary_john_relationship: Relationship[
    type -> child,
    person -> john
].
Knowledge Representation and Reasoning with Flora-2

mary_jack_relationship:Relationship[
    type -> friend,
    person -> jack
].

mary_jane_relationship:Relationship[
    type -> friend,
    person -> jane
].

mary_timeline:Timeline[
    content -> post2,
    content -> photo2
].
post2:Post[
    value -> "I am in Berlin",
    audience -> Family
].
photo2:Photo[
    imageName -> "Berlin 1",
    audience -> Friends
].
john:User[
    relationship -> john_paul_relationship,
    relationship -> john_mary_relationship,
    relationship -> john_mike_relationship,
    relationship -> john_jasmine_relationship,
    relationship -> john_sun_relationship,
    timeline -> john_timeline
].  ...
Knowledge Representation and Reasoning with Flora-2

// a user's timeline
timeline(?User, ?ListContent):-
    ?User:User[
        timeline -> ?UserTimeline
    ],
    ?ListContent = setof{ ?Content | 

• We will filter it for a viewer depending on the relationship with the current user.
Knowledge Representation and Reasoning with Flora-2

```prolog
filter_only_family(?User, ?User2, ?ListContent):-
    ?User:User[
        relationship -> ?Relationship
    ],
    (?Relationship[type->spouse, person->?User2];
    ?Relationship[type->child, person->?User2];
    ?Relationship[type->parent, person->?User2];
    ?Relationship[type->grandparent, person->?User2]
    ),
    ?ListContent = setof{ ?Content | 
        ?UserTimeline[ content -> ?Content[
            audience -> Family ] ] }.
```
filter_only_friends(?User,?User2,?ListContent):-
  ?User:User[
    relationship -> ?Relationship
  ],
  ?Relationship[type->friend,person->?User2],
  ?ListContent = setof{ ?Content |
Knowledge Representation and Reasoning with Flora-2

\texttt{filter\_only\_friends\_of\_friends} ...