Web ngữ nghĩa

Soạn bởi: Nguyễn Bá Ngọc

Chương 5

Hà Nội-2021

Chương 5.

OWL & OWL2

Nội dung

5.1. Cú pháp OWL
5.2. Ngữ nghĩa trực quan OWL
5.3. Ví dụ tổng hợp: Thiên nhiên Châu Phi
5.4. Các tầng OWL
5.5. OWL 2

Nội dung

5.1. Cú pháp OWL

5.2. Ngữ nghĩa trực quan OWL

5.3. Ví dụ tổng hợp: Thiên nhiên Châu Phi

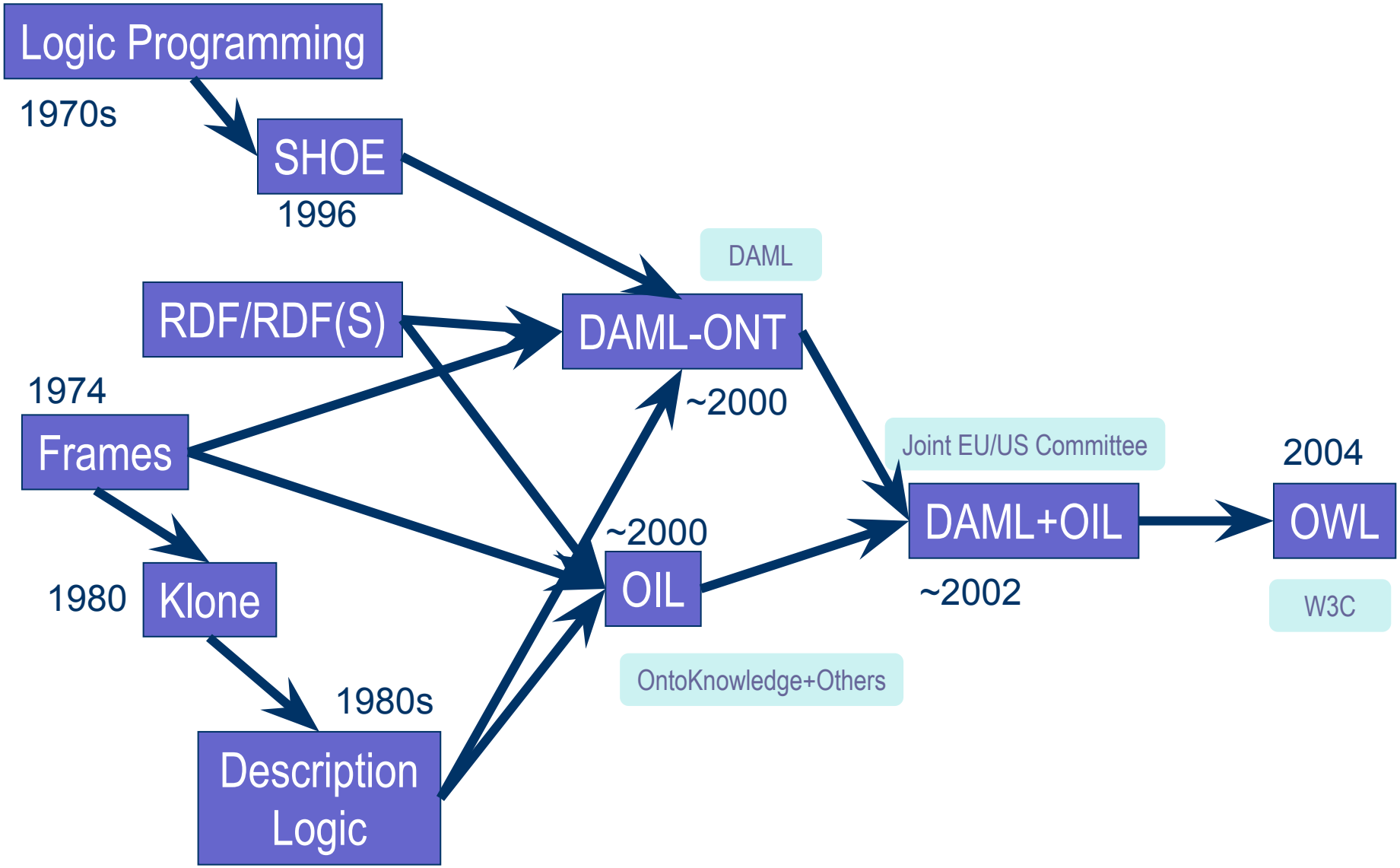
5.4. Các tầng OWL

5.5. OWL 2

Tiến trình phát triển OWL

- OWL được xây dựng trên cơ sở RDF để "*biểu diễn những tri thức đa dạng và phức tạp về các đối tượng, các lớp, và các mối quan hệ*"
- Được phát triển qua nhiều thập kỷ và kế thừa những thành tựu trong biểu diễn tri thức và suy diễn.
- OWL được phát triển từ DAML+OIL 2001 và trở thành quy chuẩn W3C năm 2004.
  - Phiên bản tiếp theo là OWL 2 (2012)
- OWL sử dụng cú pháp RDF và có ngữ nghĩa hình thức được xây dựng trên cơ sở lô-gic bậc nhất.
- Nhiều công cụ phần mềm chất lượng.

Tiến trình phát triển OWL



Ngôn ngữ Ontology: Các yêu cầu cơ bản

- Ngôn ngữ Ontology được sử dụng để mô tả hình thức mô hình lĩnh vực, vì vậy cần đáp ứng các yêu cầu sau:
  - Khả năng diễn đạt đủ dùng
  - Ngữ nghĩa hình thức
  - Khả năng suy diễn hiệu quả
  - Cú pháp đơn giản
  - Dễ hiểu

Khả năng diễn đạt vs. Suy diễn hiệu quả

- Luôn có sự bù trừ giữa khả năng diễn đạt và khả năng suy diễn
- Ngôn ngữ có khả năng diễn đạt càng mạnh thì càng khó xây dựng công cụ suy diễn hiệu quả
- Vấn đề suy diễn có thể không khả quyết hoặc bán khả quyết và nếu khả quyết thì có thể có độ phức tạp lũy thừa
- Chúng ta cần cân đối giữa:
  - Khả năng suy diễn hiệu quả và
  - Khả năng mô tả tri thức phức tạp

Một số vấn đề suy diễn

- Thành phần của lớp
  - Nếu x là 1 phần tử của lớp C và C là lớp con của D, thì chúng ta có thể kết luận x là phần tử của lớp D.
  - Dựa trên các cặp thuộc tính-giá trị có thể xác định 1 đối tượng là phần tử của lớp.
- Sự tương đương
  - Nếu lớp A tương đương với lớp B, và lớp B tương đương với lớp C, thì A cũng tương đương với C.
- Tính nhất quán
  - X là phần tử của lớp A và lớp B, A và B không giao nhau =>Mâu thuẫn => Khả năng lỗi trong Ontology hoặc dữ liệu

Ứng dụng suy diễn

- Khả năng suy diễn là điều kiện quan trọng để
  - Phát hiện các mối quan hệ mới và các thuộc tính mới
  - Tự động xác định lớp của đối tượng
  - Kiểm tra tính nhất quán của cơ sở tri thức
  - Kiểm tra mối quan hệ giữa các lớp
- Khả năng suy diễn là cần thiết để
  - Xây dựng cơ sở tri thức lớn với nhiều người cùng tham gia biên soạn
  - Tích hợp và chia sẻ các cơ sở tri thức từ nhiều nguồn

Suy diễn với OWL

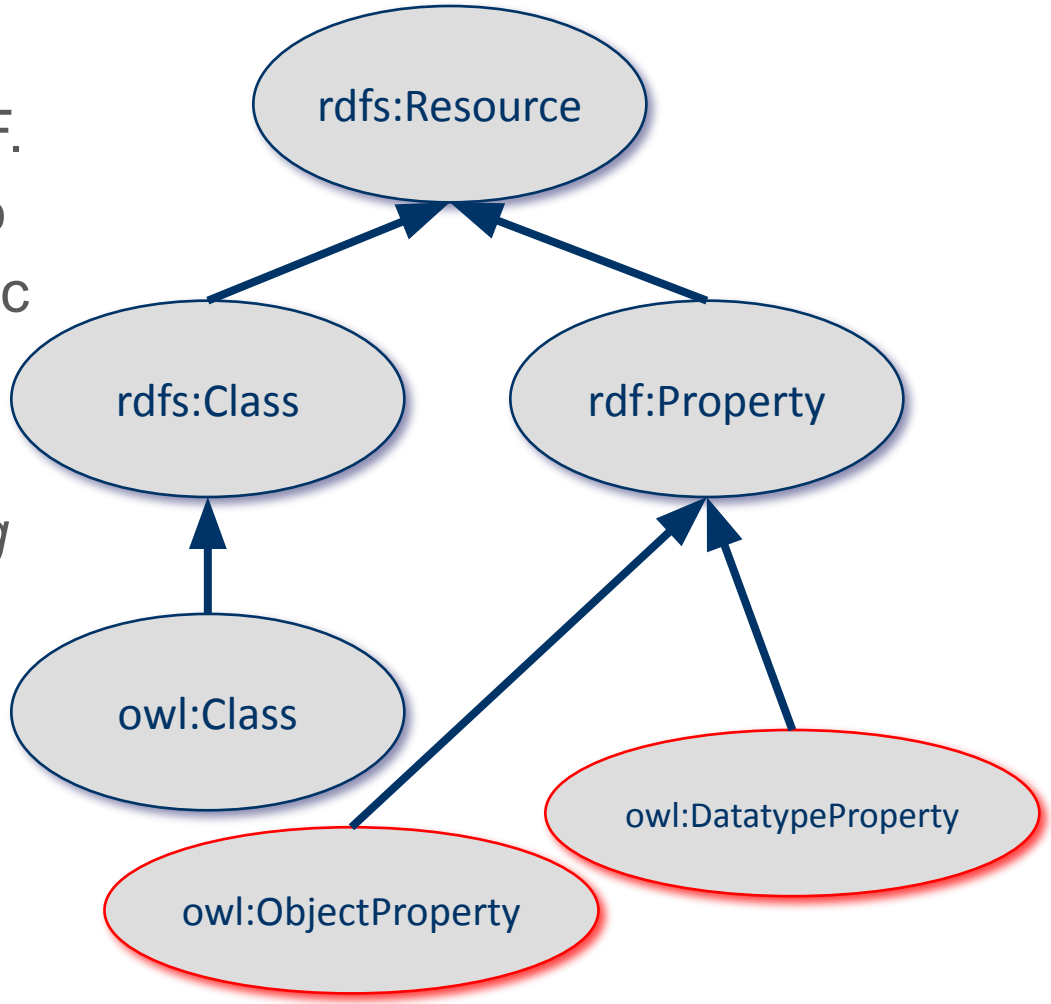
- Ngữ nghĩa là yếu tố cần thiết để có thể suy diễn:
  - Thiết lập ngữ nghĩa hình thức và suy diễn có thể được thực hiện bằng cách
  - Quy chiếu ngôn ngữ ontology sang 1 nền tảng lô-gic đã có và sử dụng các mô-đun suy diễn tự động đã tồn tại trong những nền tảng đó
- OWL (1 phần) được quy chiếu sang lô-gic mô tả
  - DL (Description Logic) là 1 tập con của lô-gic bậc nhất trong đó có thể thực hiện suy diễn hiệu quả.

OWL và RDFS

- Trong trường hợp lý tưởng, OWL cần mở rộng RDFS, nhất quán với kiến trúc phân tầng của Web ngữ nghĩa.
- Nhưng thuần túy mở rộng RDFS đi ngược với mục tiêu cân bằng khả năng diễn đạt và suy diễn hiệu quả
  - Kết hợp RDFS với lô-gic dẫn đến các vấn đề tính toán không kiểm soát được.
- OWL sử dụng RDF và 1 phần lớn của RDFS.

Tính tương thích của OWL và RDF(S)

- Tất cả các phiên bản của OWL sử dụng cú pháp RDF.
- Các phần tử được khai báo như trong RDF, sử dụng các mô tả RDF
- *Các cấu trúc OWL chi tiết hóa các cấu trúc tương ứng trong RDF(S)*
- *Các lớp và thuộc tính OWL có nhiều ràng buộc hơn so với RDF(S)*



Vì sao owl cần có owl:Class?

Mô tơ suy diễn *đúng đắn* và *đầy đủ*

- Một mô-đun suy diễn *đúng đắn* chỉ đưa ra các kết luận đúng có thể suy ra được từ đầu vào theo lô-gic
  - Sound reasoner.
  - Chúng ta thường *yêu cầu* các mô-đun suy diễn đúng đắn.
- Một mô-đun suy diễn *đầy đủ* có thể đưa ra tất cả các kết luận có thể suy ra được từ đầu vào theo lô-gic
  - Complete reasoner
  - Đưa ra tất cả các kết luận đúng và có thể cả những kết luận không đúng.
  - Không có mô-đun suy diễn đầy đủ cho toàn bộ FOL và nhiều tập con của nó
  - … và tương ứng mô-tơ suy diễn đầy đủ không được xây dựng cho OWL Full.

Các cú pháp OWL

- OWL được phát triển trên cơ sở RDF
  - Có thể viết các ontologies OWL bằng các định dạng RDF
- Bên cạnh đó các định dạng khác cho OWL cũng đã được phát triển
  - Ví dụ, cú pháp Manchester được sử dụng trong 1 số môi trường chuyên dụng.
  - Cú pháp hàm (Functional).
  - Có thể ngắn gọn, dễ đọc, dễ viết hơn so với cú pháp dựa trên RDF.

Ví dụ 5.1. Cú pháp hàm

```
SubClassOf(
   :ChildlessPerson
   ObjectIntersectionOf(
     :Person
     ObjectComplementOf(
       ObjectSomeValuesFrom(
         ObjectInverseOf( :hasParent )
         owl:Thing))))
 DisjointClasses(:Mother :Father :YoungChild)
 DisjointObjectProperties(:hasSon :hasDaughter)
 SubObjectPropertyOf(:hasFather :hasParent)
```

Thử chuyển sang cú pháp Turtle?

Ví dụ 5.1. Cú pháp hàm(2)

```
SubClassOf(
  :ChildlessPerson
  ObjectIntersectionOf(
    :Person
    ObjectComplementOf(
      ObjectSomeValuesFrom(
        ObjectInverseOf( :hasParent )
        owl:Thing))))
DisjointClasses(
  :Mother   :Father   :YoungChild)
DisjointObjectProperties(
  :hasSon   :hasDaughter)
SubObjectPropertyOf(
  :hasFather   :hasParent)
```

:ChildlessPerson là những người không nằm trong tập các giá trị của thuộc tính :hasParent.
:Mother :Father :YoungChild là các lớp tách biệt.
:hasSon :hasDaughter là các thuộc tính tách biệt.
:hasFather là thuộc tính con của :hasParent.

Thử chuyển sang cú pháp Turtle?

Ví dụ 5.1. Cú pháp hàm(3)

```
SubClassOf(
  :ChildlessPerson
  ObjectIntersectionOf(
    :Person
    ObjectComplementOf(
      ObjectSomeValuesFrom(
        ObjectInverseOf( :hasParent )
        owl:Thing))))
DisjointClasses(
  :Mother   :Father   :YoungChild)
DisjointObjectProperties(
  :hasSon   :hasDaughter)
SubObjectPropertyOf(
  :hasFather   :hasParent)
```

:ChildlessPerson là những người không nằm trong tập các giá trị của thuộc tính :hasParent.
owl:inverseOf, owl:complementOf, owl:interSectionOf, rdfs:subClassOf
:Mother :Father :YoungChild là các lớp tách biệt.
owl:AllDisjointClasses
:hasSon :hasDaughter là các thuộc tính tách biệt.
owl:propertyDisjointWith
:hasFather là thuộc tính con của :hasParent.
rdfs:subPropertyOf

Thử chuyển sang cú pháp Turtle?

Các không gian tên

@prefix owl: <http://www.w3.org/2002/07/owl#> .

@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

@prefix xsd: <http://www.w3.org/2001/XLMSchema#> .

- Các văn bản OWL là các văn bản RDF
- Bắt đầu với các khai báo không gian tên
- Quy chuẩn OWL của W3C có không gian tên http://www.w3.org/2002/07/owl#

Nội dung

5.1. Cú pháp OWL

5.2. Ngữ nghĩa trực quan OWL

5.3. Ví dụ tổng hợp: Thiên nhiên Châu Phi

5.4. Các tầng OWL

5.5. OWL 2

owl:Ontology

<https://semweb.edu.vn/uni-ns> a owl:Ontology;
    rdfs:comment "Example OWL ontology";
    owl:priorVersion <https://semweb.edu.vn/uni-ns-old>;
    owl:imports <https://semweb.edu.vn/persons> ;
    rdfs:label "University Ontology" .

- owl:Ontology - Lớp của các ontologies.
- owl:imports - Thuộc tính bắc cầu, lệnh chèn 1 tài liệu.
- owl:priorVersion - Phiên bản liền trước của tài liệu này

Định nghĩa thuộc tính

Phân loại thuộc tính trong OWL

- OWL phân biệt 2 loại thuộc tính
- Thuộc tính đối tượng có giá trị là tài nguyên
  - owl:ObjectProperty, ví dụ, isTaughtBy, supervises
- Thuộc tính dữ liệu có giá trị là hằng
  - owl:DatatypeProperty, ví dụ, SĐT, tên, tuổi, v.v…
- Được tách biệt để tạo thuận lợi cho việc xây dựng các mô-đun suy diễn đúng đắn và đầy đủ.

Các thuộc tính dữ liệu

Có giá trị là hằng có thể được định kiểu, ví dụ sử dụng kiểu dữ liệu trong lược đồ XML.

```
:age a owl:DatatypeProperty;
    rdfs:domain foaf:Person;
    rdfs:range xsd:nonNegativeInteger .
```

Các thuộc tính đối tượng

Có giá trị là tài nguyên

```
:isTaughtBy a owl:ObjectProperty;
    rdfs:domain :Course;
    rdfs:range :AcademicStaffMember;
    rdfs:subPropertyOf :involves.
```

Các cặp thuộc tính nghịch đảo

owl:inverseOf, ví dụ:

```
:teaches a owl:ObjectProperty;
    rdfs:range :Course;
    rdfs:domain :AcademicStaffMember;
    owl:inverseOf  :isTaughtBy.
```

Hoặc chỉ đơn giản là:

```
:teaches owl:InverseOf :isTaughtBy .
```

- Các bộ-3 tiên đề:
  owl:inverseOf rdfs:domain owl:ObjectProperty;
  rdfs:range owl:ObjectProperty;
  a owl:SymmetricProperty.
- Suy diễn:

(?P owl:inverseOf ?Q), (?S ?P ?O) -> (?O ?Q ?S)

(?P owl:inverseOf ?Q), (?P rdfs:domain ?C) -> (?Q rdfs:range ?C)

Các thuộc tính tương đương

owl:equivalentProperty, ví dụ:

:lectures owl:equivalentProperty :teaches .

- Các thuộc tính tương đương có cùng ***mở rộng***
  - Tập tất cả các cặp subject-object
- Suy diễn

  (?A rdfs:subPropertyOf ?B), (?B rdfs:subPropertyOf ?A)

  -> (?A owl:equivalentProperty ?B)

  (?A owl:equivalentProperty ?B)

  -> (?A rdfs:subPropertyOf ?B), (?B rdfs:subPropertyOf ?A)
- Thuộc tính nghịch đảo và thuộc tính tương đương

  (?A owl:inverseOf ?C), (?B owl:inverseOf ?C)

  -> (?A owl:equivalentProperty ?B)

Các tính chất của thuộc tính

- **owl:TransitiveProperty**
  - Lớp thuộc tính bắc cầu
  - (?P rdf:type owl:TransitiveProperty), (?A ?P ?B), (?B ?P ?C) => (?A ?P ?C)
  - Ví dụ: :subRegion a owl:TransitiveProperty .
- **owl:SymmetricProperty**
  - Lớp thuộc tính đối xứng
  - (?P rdf:type owl:SymmetricProperty), (?A ?P ?B) => (?B ?P ?A)
  - Ví dụ: :friendOf a owl:SymmetricProperty.

Các tính chất của thuộc tính(2)

- **owl:FunctionalProperty**
  - Lớp thuộc tính hàm
  - (?P rdf:type owl:FunctionalProperty), (?A ?P ?B), (?A ?P ?C) -> (?B owl:sameAs ?C)
  - Ví dụ :hasFather a owl:FunctionalProperty .
- **owl:InverseFunctionalProperty**
  - Lớp thuộc tính hàm ngược
  - (?P rdf:type owl:InverseFunctionalProperty), (?A ?P ?C), (?B ?P ?C) -> (?A owl:sameAs ?B)
  - Ví dụ :motherOf a owl:InverseFunctionalProperty .
- Thuộc tính hàm và thuộc tính hàm ngược
  - (?P owl:inverseOf ?Q), (?P rdf:type owl:FunctionalProperty) -> (?Q rdf:type owl:InverseFunctionalProperty)

Định nghĩa lớp

Các lớp trong OWL

- owl:Class là lớp của các lớp trong OWL, owl:Class là phần tử và đồng thời là lớp con của rdfs:Class
  - Ví dụ, :AssociateProfessor a owl:Class.
  - => :AssociateProfessor a rdfs:Class.
- owl:Thing là lớp khái quát nhất, chứa tất cả mọi thứ
  - Tất cả các lớp owl đều là lớp con của owl:Thing
- owl:Nothing là lớp rỗng
  - owl:Nothing là lớp con của tất cả các lớp owl

Các lớp không giao nhau

- Quan hệ không giao nhau được mô tả bằng thuộc tính **owl:disjointWith**
  - Hai lớp không giao nhau không có phần tử chung
  - Ví dụ, :Man owl:disjointWith :Woman.
- Các bộ-3 tiên đề:
  - owl:disjointWith a rdf:Property ;
    rdfs:domain owl:Class ;
    rdfs:range owl:Class .
- Suy diễn:
  - (?C owl:disjointWith ?D), (?X rdf:type ?C), (?Y rdf:type ?D) -> (?X owl:differentFrom ?Y)

Ví dụ 5.1. Các lớp không giao nhau

:Man rdfs:subClassOf foaf:Person .

:Woman rdfs:subClassOf foaf:Person .

:Man owl:disjointWith :Woman .

Các câu hỏi:

- :Man là owl:Class hay chỉ là rdfs:Class?
- Vì sao không cần kiểm tra :Man là 1 kiểu lớp?
- Chúng ta có cần kiểm tra tính chất giao nhau theo cả 2 chiều hay không?
- Điều gì sẽ xảy ra nếu chúng ta cho rằng :abc a :Man; a :Woman?

Các lớp tương đương

- Quan hệ tương đương giữa các lớp được mô tả bằng **owl:equivalentClass**
  - Các lớp tương đương có tập phần tử giống nhau
  - Ví dụ, :Faculty owl:equivalentClass :AcademicStaffMember.
- Suy diễn
  - (?P owl:equivalentClass ?Q) -> (?P rdfs:subClassOf ?Q), (?Q rdfs:subClassOf ?P)
  - (?P rdfs:subClassOf ?Q), (?Q rdfs:subClassOf ?P) -> (?P owl:equivalentClass ?Q)

Kết hợp các lớp

- Chúng ta có thể định nghĩa lớp mới bằng cách kết hợp các lớp đang có với toán tử Boolean (hợp, giao, phần bù)
- Có thể đặt tên cho kết quả của biểu thức Boolean với owl:equivalentClass

Đại số Boolean: Phần bù

- **owl:complementOf** - Diễn đạt phủ định, phần bù của
- Ví dụ: Những người không thuộc lớp :StaffMembers không phải là cán bộ

:NoneStaffMember owl:complementOf :StaffMember.

Đại số Boolean: Phép hợp

- **owl:unionOf** - Hợp của
- Ví dụ, người trong trường đại học bao gồm cán bộ và sinh viên

:PeopleAtUni owl:equivalentClass

[owl:unionOf (:StaffMember :Student)].

Hoặc

:PeopleAtUni owl:unionOf (:StaffMember :Student).

Đại số Boolean: Phép giao

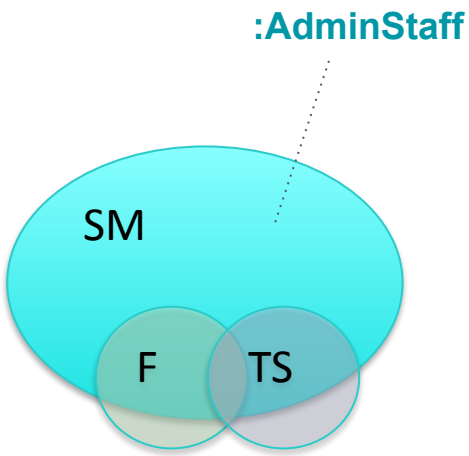
- **owl:intersectionOf** - Giao của
- Ví dụ, Lớp :Mother là giao của :Parent và :Woman

:Mother owl:equivalentClass
[owl:intersectionOf (:Parent :Woman)].

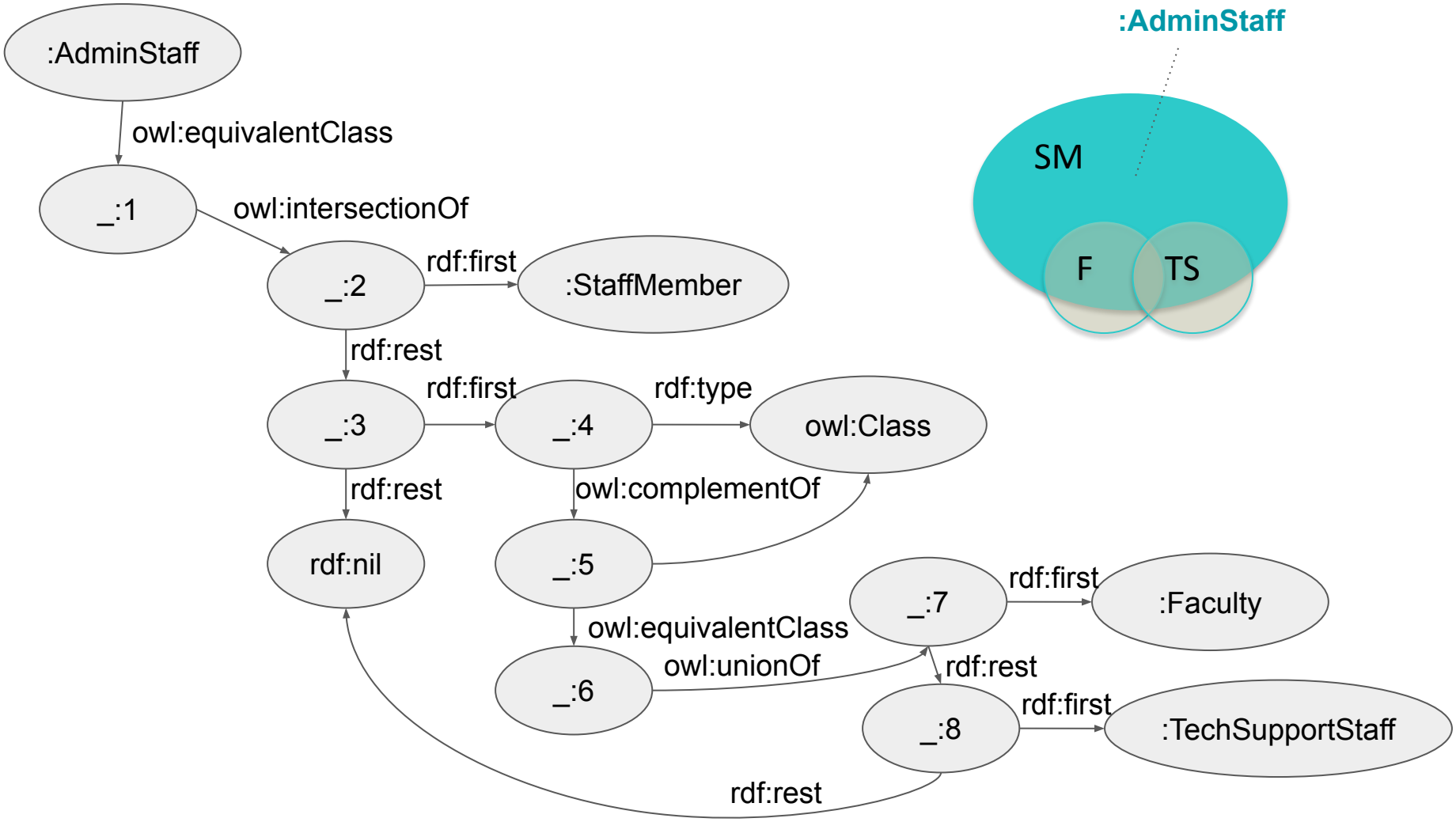
:Mother owl:intersectionOf (:Parent :Woman).

Ví dụ 5.2. Biểu thức Boolean phức tạp

```
:AdminStaff owl:equivalentClass
  [owl:intersectionOf
    (:StaffMember
      [a owl:Class;
        owl:complementOf
        [a owl:Class;
          owl:equivalentClass
            [owl:unionOf (:Faculty :TechSupportStaff)]
        ]
      ]
    )
  ] .
```



Ví dụ 5.2. Biểu thức Boolean phức tạp(2)



Liệt kê phần tử thuộc lớp với owl:oneOf

- **owl:oneOf** - Một của.
  - Được sử dụng để định nghĩa lớp bằng cách liệt kê các phần tử của nó
- Ví dụ lớp ngày trong tuần: Monday, Tuesday, …

```
[a owl:Class;
 owl:oneOf (:Monday
            :Tuesday
            :Wednesday
            :Thursday
            :Friday
            :Saturday
            :Sunday) ]
```

Giới hạn thuộc tính

- Chúng ta có thể định nghĩa lớp bằng cách chọn lọc các phần tử của nó theo giá trị thuộc tính
- Phong cách diễn đạt: *[C là] lớp của các đối tượng có giá trị của thuộc tính P thỏa mãn các điều kiện ràng buộc (định danh C không bắt buộc, lớp có thể là ẩn danh).*
- Ví dụ lớp không tên: Lớp những người có ít nhất 1 con.

```
[owl:intersectionOf (:Person
    [a owl:Restriction;
     owl:onProperty :hasChild;
     owl:minCardinality "1"^^xsd:integer]].
```

Giới hạn thuộc tính(2)

- owl:Restriction được sử dụng để mô tả lớp với nhiều ràng buộc cụ thể.
- Mỗi phần tử thỏa mãn 1 hoặc nhiều giới hạn trên 1 thuộc tính được xác định bởi owl:onProperty

Ví dụ 5.3. Giới hạn thuộc tính

Lớp phụ huynh bao gồm những người có tối thiểu 1 con:

```
:Parent owl:equivalentClass
 [ owl:intersectionOf (:Person
    [a owl:Restriction;
    owl:onProperty
    :hasChild;
    owl:minCardinality "1"]) ] .
```

Ràng buộc đối với giá trị

Có thể ràng buộc kiểu hoặc giá trị cụ thể

- owl:allValuesFrom Tất cả giá trị từ … (lớp)
- owl:hasValue Có giá trị bằng … (giá trị cụ thể)
- owl:someValuesFrom Có giá trị từ … (lớp)

owl:allValuesFrom

- Được sử dụng để mô tả lớp của các đối tượng mà tất cả giá trị của 1 thuộc tính của nó đều thuộc 1 lớp nào đó.
- Ví dụ giá trị của thuộc tính ăn của các phần tử thuộc lớp người ăn chay đều thuộc lớp các món ăn có nguồn gốc thực vật:

```
:Vegetarian owl:subClassOf
    [a owl:Restriction;
     owl:onProperty :eats;
     owl:allValuesFrom :VegetarianFood].
```

Ví dụ 5.4. owl:allValuesFrom

Người sinh ra người

```
:Person a owl:Class;
   rdfs:subClassOf
      [ a owl:Restriction;
        owl:onProperty bio:offspring;
        owl:allValuesFrom :Person
       ] .

:john a :Person; bio:offspring :mary.
    => :mary a :Person.   ???
```

Các hệ quả

"người sinh ra người"

```
:Person a owl:Class; rdfs:subClassOf
    [a owl:Restriction;
     owl:onProperty bio:offspring;
     owl:allValuesFrom :Person] .
:bio:offspring rdfs:domain :Human;
                    rdfs:range :Human.
```

-> ???

```
:carol a foaf:Person.
:don bio:offspring :carol.
```

-> :don a :Person. ???

Các hệ quả(2)

"người được sinh bởi người"

```
:Person rdfs:subClassOf
    [a owl:Restriction;
     owl:onProperty bio:sprungFrom;
     owl:allValuesFrom :Person] .

bio:sprungFrom rdfs:domain :Animal;
                        rdfs:range :Animal;
                        owl:inverseOf bio:offspring.
:carol a :Person.
:don bio:offspring :carol.
-> :don a :Person. ???
```

owl:hasValue

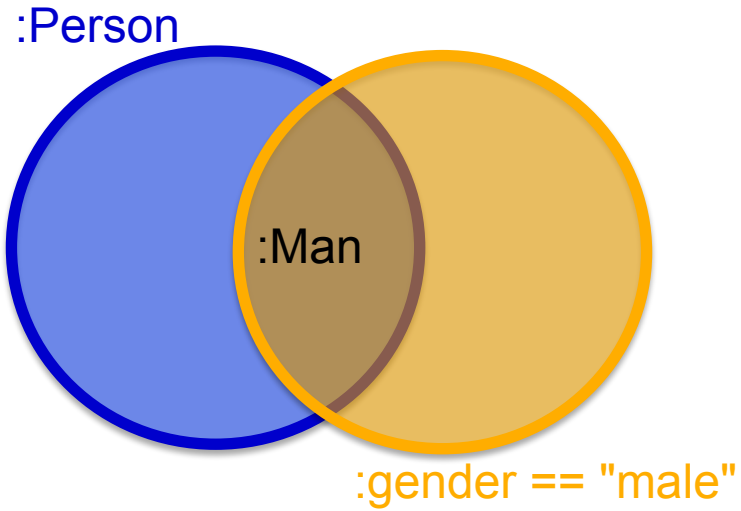
- Được sử dụng để mô tả 1 lớp bao gồm các phần tử với 1 giá trị cụ thể của 1 thuộc tính
- Ví dụ, Môn toán được dạy bởi Giáo sư NVA

```
:NVAMathCourse rdfs:subClassOf
    [owl:intersectionOf(:mathCourse
        [ a owl:restriction;
            owl:onProperty :isTaughtBy;
            owl:hasValue :NVA]) ] .
```

- Các câu hỏi:
  - Tất cả các lớp toán đều được dạy bởi NVA?
  - Một số lớp được dạy bởi NVA?

Ví dụ 5.5. owl:hasValue

```
:Man owl:equivalentClass
  [owl:intersectionOf
   (:Person,
     [a owl:Restriction;
       owl:onProperty :gender;
       owl:hasValue "male"] )].
```



Có thể suy ra được gì

Lớp trong OWL giống như tập hợp

:ed a :Man .
-> :ed :gender "male". ???
:frank a :Person; :gender "male" .
-> :frank a :Man. ???
:pat a :Person; :gender "male"; gender "female" .
-> :pat a :Man. ???

owl:someValuesFrom

- Mô tả lớp bao gồm các phần tử có ít nhất 1 giá trị của 1 thuộc tính thuộc 1 lớp nào đó.
- Ví dụ, Giảng viên có dạy ít nhất 1 môn trong chương trình đại học

[a owl:Restriction;
owl:onProperty :teaches;
owl:someValuesFrom :UndergraduateCourse].

Giới hạn cơ số

- Giá trị tối thiểu và tối đa có thể được mô tả bằng owl:minCardinality & owl:maxCardinality
  - Các khóa với ít hơn 10 sinh viên;
  - Các khóa với số lượng sinh viên trong khoảng 10 và 100;
  - Các khóa với nhiều hơn 100 sinh viên.
- Giá trị cụ thể được mô tả bằng owl:cardinality
  - Khóa với đúng 7 sinh viên;
  - Cũng có thể được biểu diễn dựa trên giá trị cực tiểu và giá trị cực đại.

Ví dụ 5.6. Giới hạn cơ số

- Khóa được dạy bởi ít nhất hai người

[a owl:Restriction;
  owl:onProperty :isTaughtBy;
  owl:minCardinality "2"^^xsd:nonNegativeInteger] .

- Phụ huynh có ít nhất 1 con:

:Parent owl:equivalentClass
  [a owl:Restriction;
    owl:onProperty :hasChild;
    owl:minCardinality "1"^^xsd:integer] .

Giả thuyết tên duy nhất

- OWL không áp dụng giả thuyết tên duy nhất như trong các hệ quản trị CSDL
  - Các phần tử có ID khác nhau vẫn có thể là các phần tử tương đương.
- Giả sử chúng ta khẳng định mỗi khóa được dạy bởi tối đa 1 giảng viên, nhưng nếu có 1 khóa được dạy bởi x và y.
  - Bộ suy diễn OWL sẽ không thiết lập cờ lỗi
  - Thay vào đó nó suy diễn x tương đương với y.

Các đối tượng khác nhau

- Có thể mô tả các đối tượng khác nhau với owl:differentFrom
  - Ví dụ: :john owl:differentFrom :mary .
- OWL cung cấp cách viết tắt để mô tả 1 tập phần tử không trùng lặp:
  - Ví dụ:
  - [a owl:allDifferent;
    owl:distinctMembers (:alice :bob :carol :don)].

Các kiểu dữ liệu trong OWL

- Lược đồ XML cung cấp một cơ chế để người dùng tự tạo các kiểu dữ liệu
  - Ví dụ, kiểu dữ liệu AdultAge bao gồm các số nguyên >= 18
- Những kiểu dữ liệu suy diễn như vậy không dùng được trong OWL
  - Tài liệu suy diễn OWL liệt kê tất cả các kiểu từ lược đồ XML có thể được sử dụng
  - Danh sách này bao gồm những kiểu được sử dụng thường xuyên như string, integer, boolean, time, và date

Suy diễn tính tách biệt

Một ontology có thể cung cấp nhiều cách để suy ra các phần tử là khác nhau từ những gì đã biết về chúng, ví dụ:

- Thuộc các tập không giao nhau (như :Man, :Woman)
  - :pat1 a :man. :pat2 a :woman. :man owl:disjointWith :woman.
- Có các thuộc tính hàm nghịch với các giá trị khác nhau
  - :ssn a owl:inverseFunctionalProperty .
  - :pat1 :ssn "249148660". :pat2 :ssn "482962271".
- Có các giá trị khác nhau từ một thuộc tính hàm
  - :ssn a owl:FunctionalProperty .
  - :pat1 :ssn "249148660". :pat2 :ssn "482962271" .
- Được kết nối bằng mối quan hệ không phản xạ
  - :pat1 :hasChild :pat2. :hasChild a owl:IrreflexiveProperty .

Nội dung

5.1. Cú pháp OWL

5.2. Ngữ nghĩa trực quan OWL

5.3. Ví dụ tổng hợp: Thiên nhiên Châu Phi

5.4. Các tầng OWL

5.5. OWL 2

Ontology Thiên nhiên Châu Phi

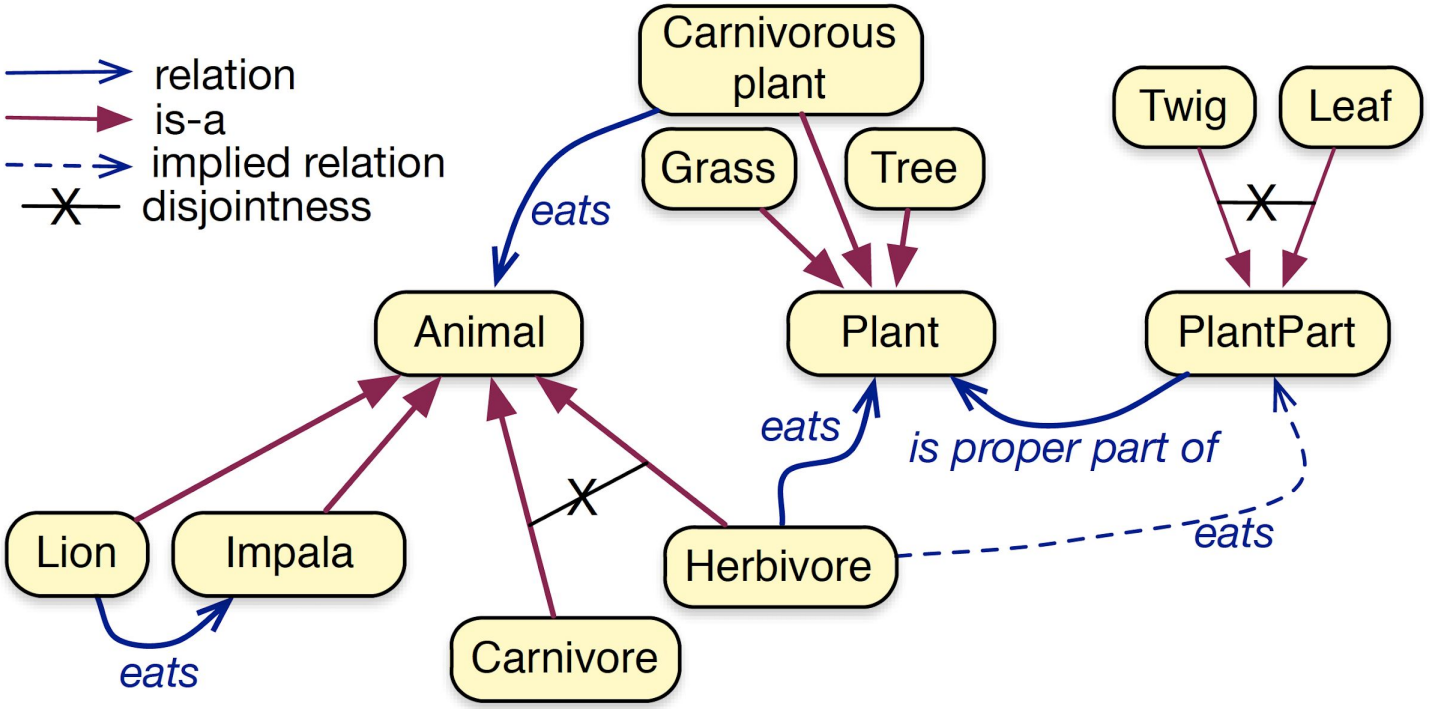


Figure 1 The African Wildlife Ontology at a glance. The main classes and relations of the African Wildlife ontology (v1) and an illustrative selection of its subclasses.

[Tái bản lần 2 của The Semantic Web Primer]

Các lớp cơ bản

:Plant owl:disjointWith :Animal

:Tree rdfs:subClassOf :Plant

:Animal owl:disjointWith :Plant .

:Herbivore rsds:subClassOf :Animal;

owl:disjointWith :Carnivore .

:Giraffe rdfs:subClassOf :Herbivore .

:Carnivore rdfs:subClassOf :Animal .

:Lion rdfs:subClassOf :Carnivore .

Các thuộc tính

:isPartOf a owl:TransitiveProperty .

:eats :domain :Animal.

động vật có thuộc tính ăn

Carnivorous plant ??

:eats owl:inverseOf :eatenBy.

thuộc tính ngược / nghịch đảo của ăn là bị ăn

Các lớp suy diễn

```
# cành là bộ phận của cây
:Branch rdfs:subClassOf
  [a owl:Restriction;
    owl:allValuesFrom :Tree
    owl:onProperty :isPartOf]

# lá là bộ phận của cành
:Leaf rdfs:subClassOf
  [a owl:Restriction;
    owl:allValuesFrom :Branch
    owl:onProperty :isPartOf]
```

Vẽ các đồ thị RDF

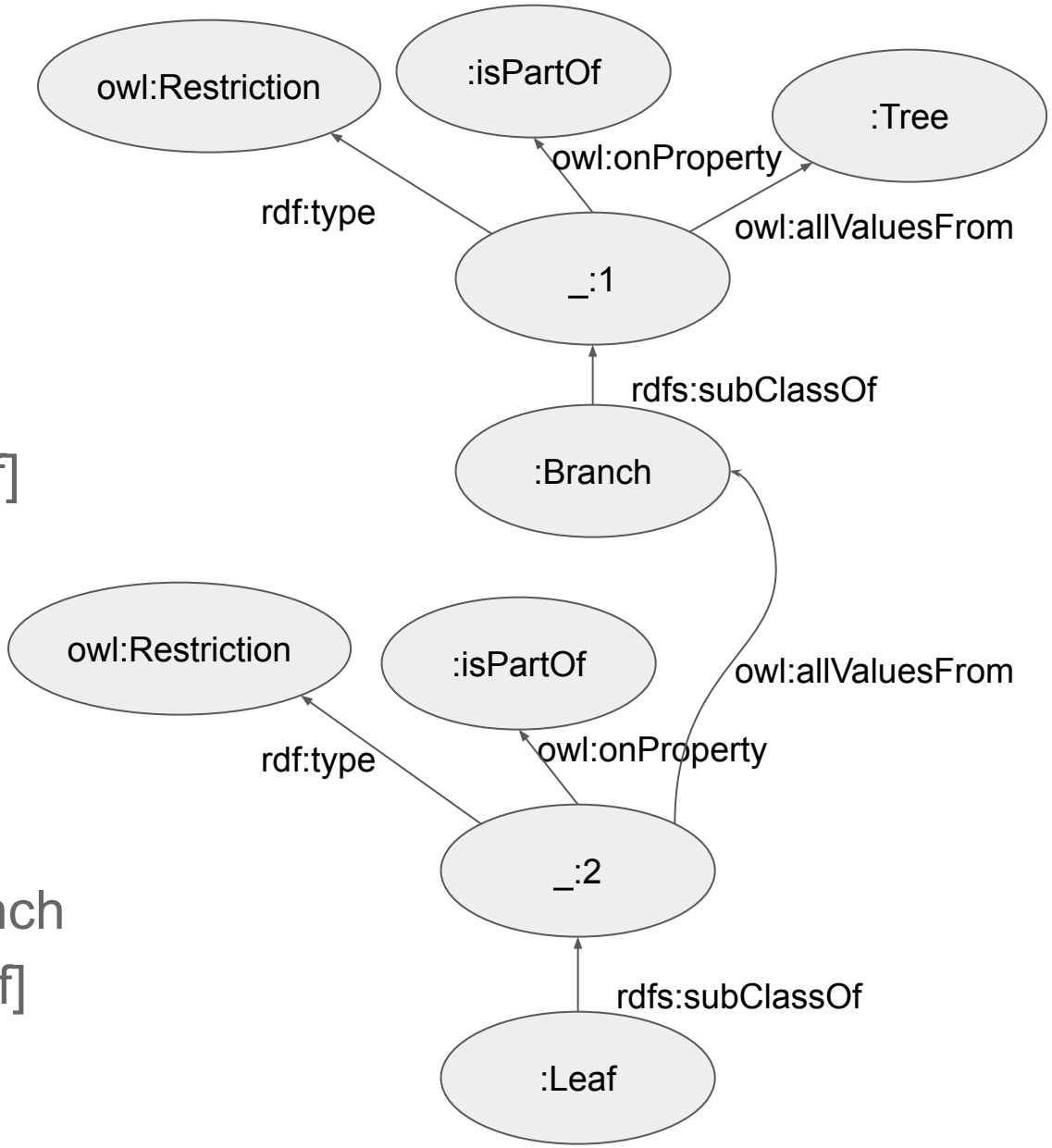
Các nhánh

 # cành là bộ phận của cây

```
:Branch rdfs:subClassOf
  [a owl:Restriction;
    owl:allValuesFrom :Tree
    owl:onProperty :isPartOf]
```

 # lá là bộ phận của cành

```
:Leaf rdfs:subClassOf
  [a owl:Restriction;
    owl:allValuesFrom :Branch
    owl:onProperty :isPartOf]
```



Động vật ăn thịt

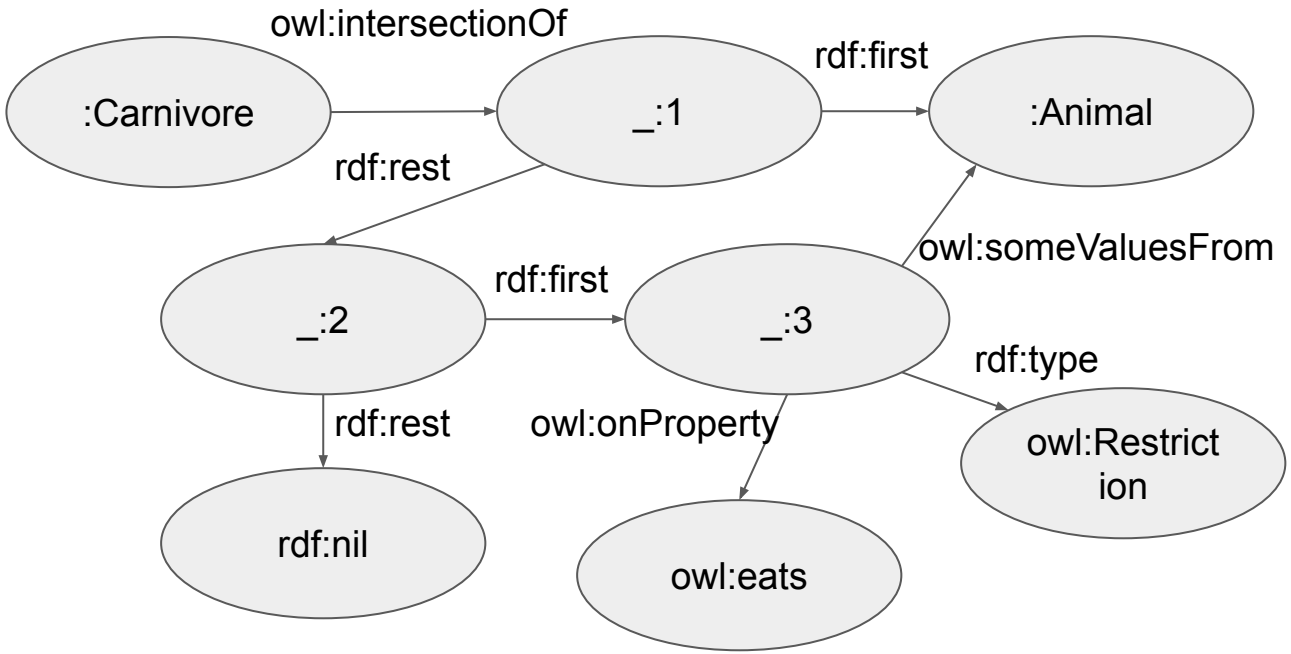
```
# Động vật ăn thịt là những động vật ăn động vật khác
:Carnivore owl:intersectionOf
    (:Animal,
        [a owl:Restriction;
            owl:onProperty :eats;
            owl:someValuesFrom :Animal]
    ) .
```

Động vật ăn thịt có ăn cây cỏ không?
Chúng ta định nghĩa động vật ăn cỏ bằng cách nào?

Động vật ăn thịt(2)

```
:Carnivore owl:intersectionOf
    (:Animal
        [a owl:Restriction;
            owl:onProperty :eats;
            owl:someValuesFrom :Animal]
    ) .
```

Vẽ đồ thị rdf



Động vật ăn cỏ

```
:Herbivore a owl:Class;
  rdfs:comment "Herbivores
      are exactly those
      animals that eat only
      plants or parts of
      plants" .
```

```
:Herbivore owl:equivalentClass
  [a owl:Class;
   owl:intersectionOf
    (:Animal
     [a owl:Restriction;
       owl:onProperty :eats;
       owl:allValuesFrom
         [a owl:Class;
          owl:equivalentClass
           [ owl:unionOf
            (:Plant
             [a owl:Restriction;
              owl:onProperty :isPartOf;
              owl:allValuesFrom :Plant])]]])].
```

Hươu cao cổ

 # Hươu cao cổ là động vật ăn cỏ, và chỉ ăn lá

```
Giraffe rdfs:subClassOf
    :Herbivore,
    [owl:Restriction
        owl:onProperty :eats;
        owl:allValuesFrom :Leaf] .
```

Sư tử

```
# Sư tử là động vật chỉ ăn các loài ăn cỏ
:Lion rdfs:subClassOf
    :Carnivore,
    [a Restriction
       owl:onProperty :eats;
       owl:allValuesFrom :Herbivore] .
```

Cây ngon

```
# Cây ngon bị ăn bởi cả loài ăn thịt và loài ăn cỏ
:TastyPlant
    rdfs:subClassOf :Plant,
        [a Restriction
            owl:onProperty :eatenBy;
            owl:someValuesFrom :Herbivore],
        [a Restriction
            owl:onProperty :eatenBy;
            owl:someValuesFrom :Carnivore ].
```

Bộ phận của cây

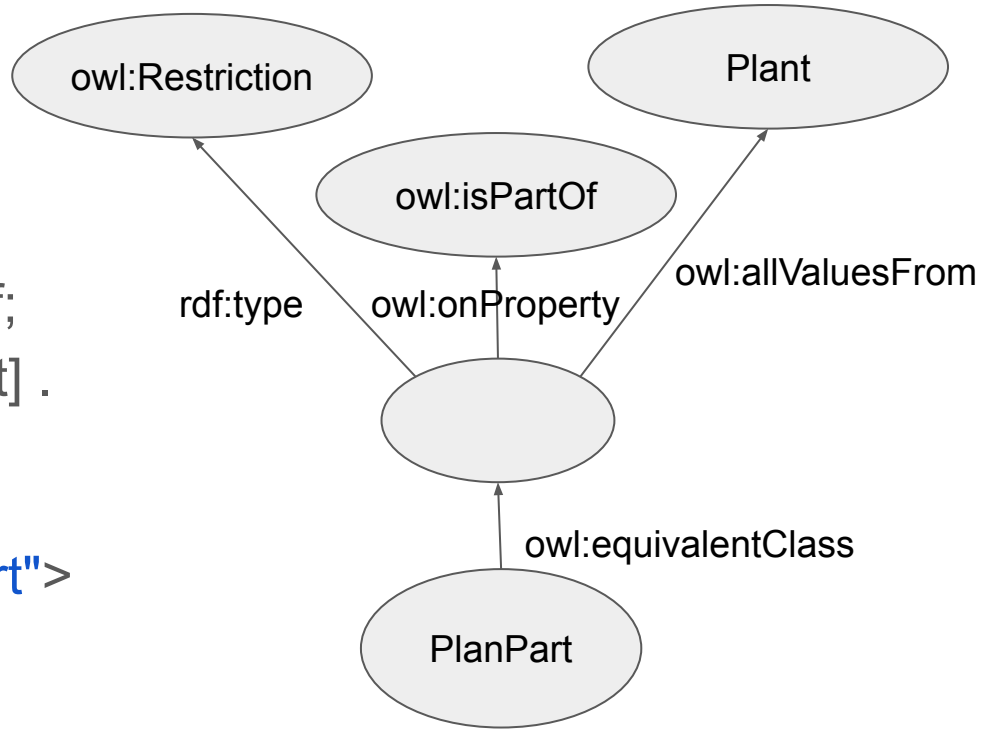
Định nghĩa lớp PlantPart?

- Cú pháp Turtle
- Cú pháp RDF/XML
- Đồ thị RDF

Bộ phận của cây(2)

Định nghĩa lớp PlantPart

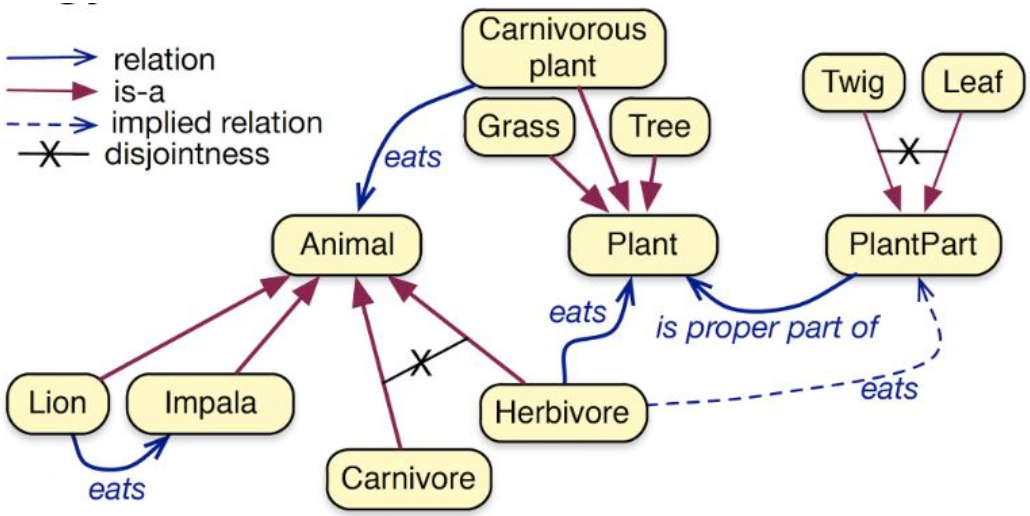
```
:PlantPart owl:equivalentClass
      [a owl:Restriction;
            owl:onProperty :isPartOf;
            owl:allValuesFrom :Plant] .
```




```
<rdf:Description rdf:ID="PlantPart">
    <owl:equivalentClass>
     <owl:Restriction>
       <owl:allValuesFrom rdf:ID="Plant"/>
       <owl:onProperty rdf:ID="isPartOf"/>
      </owl:Restriction>
    </owl:equivalentClass>
</rdf:Description>
```

Thử nghiệm

Biểu diễn khái niệm và suy diễn



- Miền của eats là hợp của Animal và CarniPlant
- Khoảng của eats là hợp của Animal và Plant
- CarniPlant là giao của Plant và lớp của các thứ có giá trị thuộc tính eats thuộc lớp Animal.
- CarniVore là giao của Animal và lớp của các thứ có tất cả giá trị thuộc tính eats thuộc lớp Animal.
- Herbivore Là giao của Animal và lơp của các thứ có tất cả giá trị thuộc tính eats thuộc lớp Plant.

a) Biểu diễn các khái niệm sử dụng cú pháp Turtle

b) Thử nghiệm suy diễn với bộ từ vựng thu được từ mục a.

Thử nghiệm suy diễn

Tổng hợp tài liệu Turtle

Định nghĩa thuộc tính :eats

Định nghĩa các lớp :Herbivore và :Carnivore

… Định nghĩa các từ liên quan khác

:impala :eats :TastyPlant .

:pig :eats [unionOf (:Plant :PlantPart)].

:Simba :eats :pig, :impala.

=> Suy ra được ??

Thử với mô tơ suy diễn, ví dụ trong Apache Jena??

Thử nghiệm suy diễn: Từ vựng$_{(2)}$

```
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd: <http://www.w3.org/2001/XLMSchema#> .
@prefix : <http://semweb.edu.vn/family#> .

:Parent owl:equivalentClass [
  a owl:Restriction;
  owl:onProperty :hasChild;
  owl:minCardinality 1
].
:Female rdfs:subClassOf [
  a owl:Restriction;
  owl:onProperty :gender;
  owl:hasValue "female"^^xsd:string
].
:Father owl:equivalentClass [
  owl:intersectionOf (:Parent :Male)
].
```

```
[ a owl:Restriction;
  owl:onProperty :gender;
  owl:hasValue "male"^^xsd:string
] rdfs:subClassOf :Male.
```

Thử nghiệm suy diễn: Dữ kiện(3)

@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd: <http://www.w3.org/2001/XLMSchema#> .
@prefix : <http://semweb.edu.vn/family#> .

:john :gender "male"^^xsd:string.
:mary :gender "female"^^xsd:string.
:john :hasChild :mary.

a) Có thể suy ra những gì dựa trên ngữ nghĩa RDFS và OWL?
   Tìm 3 câu và giải thích cơ sở của suy diễn?
b) Thực hiện suy diễn với OWL Reasoner của Apache Jena.

./fuseki-server --config fuseki-config.ttl

Apache Jena Fuseki

https://jena.apache.org/documentation/fuseki2/fuseki-configuration.html

https://github.com/apache/jena/tree/main/jena-fuseki2/examples

Nội dung

5.1. Cú pháp OWL

5.2. Ngữ nghĩa trực quan OWL

5.3. Ví dụ tổng hợp: Thiên nhiên Châu Phi

5.4. Các tầng OWL

5.5. OWL 2

Các phiên bản OWL

- W3C chia OWL thành ba tầng ngôn ngữ/phiên bản:
  - OWL Lite
  - OWL DL
  - OWL Full
- Mỗi phiên bản hướng tới đáp ứng các nhu cầu sử dụng khác nhau

OWL Full

- Bao gồm tất cả các thành phần của OWL
- Cho phép kết hợp các thành phần này theo cách tùy ý với RDF và RDFS
- OWL Full hoàn toàn tương thích với RDF, cả cú pháp và ngữ nghĩa
- Khả năng diễn đạt của OWL Full rất mạnh, không thể thực hiện suy diễn hiệu quả

OWL DL

- OWL DL (Lô-gic mô tả - Description Logic) là 1 tập con của OWL Full
  - Cùng tập từ khóa như OWL Full, tuy nhiên ràng buộc chặt hơn về ý nghĩa
  - Tương đương với lô-gic mô tả
- Có thể thực hiện suy diễn hiệu quả với OWL DL
- Không hoàn toàn tương thích với RDF
  - Tài liệu RDF có thể không phải tài liệu OWL DL hợp lệ
  - Tất cả các tài liệu OWL DL đều là tài liệu RDF hợp lệ.

OWL Lite

- Tập con của OWL DL, với nhiều giới hạn hơn nữa
  - Ví dụ, OWL Lite loại bỏ các lớp liệt kê, các câu về tính chất không giao nhau, và cơ số tùy ý
- Ưu điểm của phiên bản này:
  - Đơn giản đối với người dùng
  - và người phát triển công cụ
- Nhược điểm: Khả năng diễn đạt hạn chế.

Các tập tính năng OWL

- Các phiên bản OWL khác nhau có những tập tính năng khác nhau
- Trong OWL Full, tất cả các cấu trúc ngôn ngữ có thể được sử dụng theo bất kỳ tổ hợp nào và kết quả thu được vẫn là tài liệu RDF hợp lệ
- OWL DL loại bỏ hoặc hạn chế một số tính năng để đảm bảo khả năng thực hiện suy diễn đầy đủ hoặc đơn giản hóa việc xây dựng các mô tơ suy diễn.

Các ràng buộc tính năng trong OWL DL

- Khai báo tường minh lớp và thuộc tính
  - Các lớp và thuộc tính trong mệnh đề phải được khai báo tường minh là owl:Class
  - Ví dụ, một lớp phải được khai báo là owl:Class nếu được sử dụng với rdfs:subClassOf.
- Phân biệt các định kiểu theo cặp:
  - Khác với OWL Full, OWL DL không cho phép sử dụng lớp như bản thực hoặc thuộc tính, và cũng không cho phép sử dụng thuộc tính như lớp hoặc bản thực.

[https://www.w3.org/TR/owl-ref/]

Các ràng buộc tính năng trong OWL $DL_{(2)}$

- Tách biệt thuộc tính
  - Tập mở rộng của thuộc tính đối tượng và tập mở rộng của thuộc tính dữ liệu không giao nhau
  - Vì vậy không thiết lập được các tính chất sau cho các thuộc tính dữ liệu
    - owl:InverseOf
    - owl:FunctionalProperty
    - owl:InverseFunctionalProperty
    - owl:SymmetricProperty

Các ràng buộc tính năng trong OWL $DL_{(3)}$

- Giới hạn cơ số không có tính chất bắc cầu
  - Không thiết lập được giới hạn cơ số cho thuộc tính bắc cầu
  - ví dụ, người với nhiều hơn 5 hậu duệ
- Các lớp ẩn danh (được biểu diễn bằng nút rỗng) bị giới hạn:
  - Chỉ được phép xuất hiện như:
    - Chủ thể và và giá trị của owl:equivalentClass hoặc owl:disjointWith
    - Giá trị (không cho phép chủ thể) của rdfs:subClassOf

Các giới hạn tính năng trong OWL Lite

Các giới hạn của OWL DL và hơn nữa:

- Không được sử dụng owl:oneOf, owl:disjointWith, owl:unionOf, owl:complementOf, owl:hasValue
- Các mô tả cơ số: tối thiểu, tối đa, bằng giá trị cụ thể, giá trị số bị giới hạn chỉ có thể là 0 hoặc 1
- Không được thiết lập owl:equivalentClass giữa các lớp ẩn danh, chỉ được thiết lập giữa các lớp có định danh
- Hạn chế chỉ sử dụng owl:intersectionOf với các lớp đã định danh và các giới hạn.

Các tập từ khóa OWL

- OWL Lite:
  - equivalentClass, equivalentProperty, sameAs, differentFrom, AllDifferent,
  - inverseOf, TransitiveProperty, SymmetricProperty, FunctionalProperty, InverseFunctionalProperty,
  - allValuesFrom, someValuesFrom,
  - minCardinality, maxCardinality, cardinality, - *0 hoặc 1*
  - intersectionOf - *Lớp đã định danh hoặc Restriction*
- OWL DL, OWL Full:
  - oneOf, hasValue, disjointWith,
  - unionOf, complementOf, intersectionOf - *tổ hợp tùy ý*
  - minCardinality, maxCardinality, cardinality - *giá trị tùy ý*

[https://www.w3.org/TR/2004/REC-owl-features-20040210/]

Nội dung

5.1. Cú pháp OWL

5.2. Ngữ nghĩa trực quan OWL

5.3. Ví dụ tổng hợp: Thiên nhiên Châu Phi

5.4. Các tầng OWL

5.5. OWL 2

Các mở rộng của OWL

- Các mô-đun và imports
- Mặc định
- Giả thuyết thế giới đóng
- Giả thuyết tên duy nhất
- Tiến trình đính kèm
- Luật chuỗi thuộc tính

Các mô-đun và các thành phần cấu trúc

- Khả năng chèn của OWL rất giới hạn
  - Nó chỉ cho phép chèn toàn bộ ontology, không phải 1 phần của nó
- Các mô-đun trong các ngôn ngữ lập trình dựa trên ẩn dữ liệu: Chức năng trạng thái, ẩn các chi tiết triển khai

Câu hỏi mở - Làm sao để thiết lập cơ chế mô-đun thích hợp cho các ngôn ngữ ontology Web?

Các mặc định

- Nhiều hệ thống biểu diễn tri thức ứng dụng cho phép kế thừa các giá trị bị ghi đè bởi các lớp chi tiết hơn trong cây
  - Coi các giá trị kế thừa là mặc định
- Chưa đi đến nhận định cuối cùng về hành vi không đơn điệu của các giá trị mặc định.

Giả thuyết thế giới đóng

- OWL hiện đang sử dụng giả thuyết thế giới mở:
  - Một câu không thể được cho là đúng dựa trên cơ sở không thể chứng minh nó
  - Trên một phần tri thức khổng lồ của WWW thì giả thuyết này là đúng đắn
- Giả thuyết thế giới đóng: Một câu là đúng nếu không thể phủ nhận nó
  - Gắn chặt với khái niệm mặc định, dẫn đến hành vi không đơn điệu

Giả thuyết tên duy nhất

- Các ứng dụng CSDL bình thường cho rằng các phần tử với định danh khác nhau là các phần tử khác nhau
- OWL sử dụng cách tiếp cận lô-gic, các phần tử có định danh khác nhau chưa chắc đã khác nhau
  - Hợp lý hơn trong không gian WWW
- Trong một số trường hợp có thể chỉ ra phần ontology mà giả thuyết đúng hoặc không đúng

Tiến trình đính kèm

- Một kỹ thuật tiêu biểu trong biểu diễn tri thức là thiết lập ý nghĩa của từ bằng cách gắn một đoạn mã có thể thực thi được để tính toán ý nghĩa của từ
  - Không thông qua định nghĩa tường minh của ngôn ngữ
- Tuy được sử dụng rộng rãi, những khái niệm này không tương thích tốt trong hệ thống với ngữ nghĩa hình thức, và chưa được bao gồm trong OWL

Các luật đối với chuỗi thuộc tính

- OWL không cho phép kết hợp các thuộc tính làm cơ sở quyết định.
- Trong nhiều ứng dụng đó là một thao tác hữu ích.
- Một người có thể muốn định nghĩa thuộc tính như các luật tổng quát (Luật Horn hoặc luật khác) dựa trên các thuộc tính khác.
- Tích hợp biểu diễn tri thức dựa trên luật và biểu diễn tri thức dựa trên lô-gic mô tả là 1 lĩnh vực nghiên cứu.

OWL 2 thêm vào

- Cơ số có điều kiện
  - Một bàn tay có năm ngón, một ngón cái và bốn ngón khác
- Hỗ trợ mạnh hơn kiểu dữ liệu/miền
- Các đặc điểm thuộc tính bổ xung
  - Ví dụ, tính phản xạ.
- Chuỗi vai trò
  - Ví dụ, hasParent.hasSibling.hasChild
- Một mô hình tốt hơn để cắt nhánh trong DL
  - Cho phép một từ vừa là khái niệm vừa là phần tử
- Khả năng ghi chú mạnh hơn

Tổng kết

- OWL là quy chuẩn ontology cho Web
- OWL được xây dựng trên cơ sở RDF và RDFS
  - Cú pháp RDF dựa trên XML được sử dụng
  - Các phần tử được xác định bởi các mô tả RDF
  - Hầu hết các thành phần mô hình hóa RDFS được sử dụng
- Ngữ nghĩa hình thức và hỗ trợ suy diễn được cung cấp thông qua các ánh xạ của OWL lên lô-gic
  - Lô-gic vị từ và lô-gic mô tả được sử dụng cho mục đích này
- Trong khi OWL là đủ khả năng diễn tả để ứng dụng trong thực tiến, các mở rộng đang được thực hiện
  - Có gắng tiếp tục cung cấp các tính năng lô-gic, bao gồm cả các luật