ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

Chương 5

OWL và OWL 2

Nội dung

5.1. Cú pháp OWL
5.2. Ngữ nghĩa trực quan OWL
5.3. Các tầng OWL
5.4. OWL 2

Nội dung

5.1. Cú pháp OWL

5.2. Ngữ nghĩa trực quan OWL

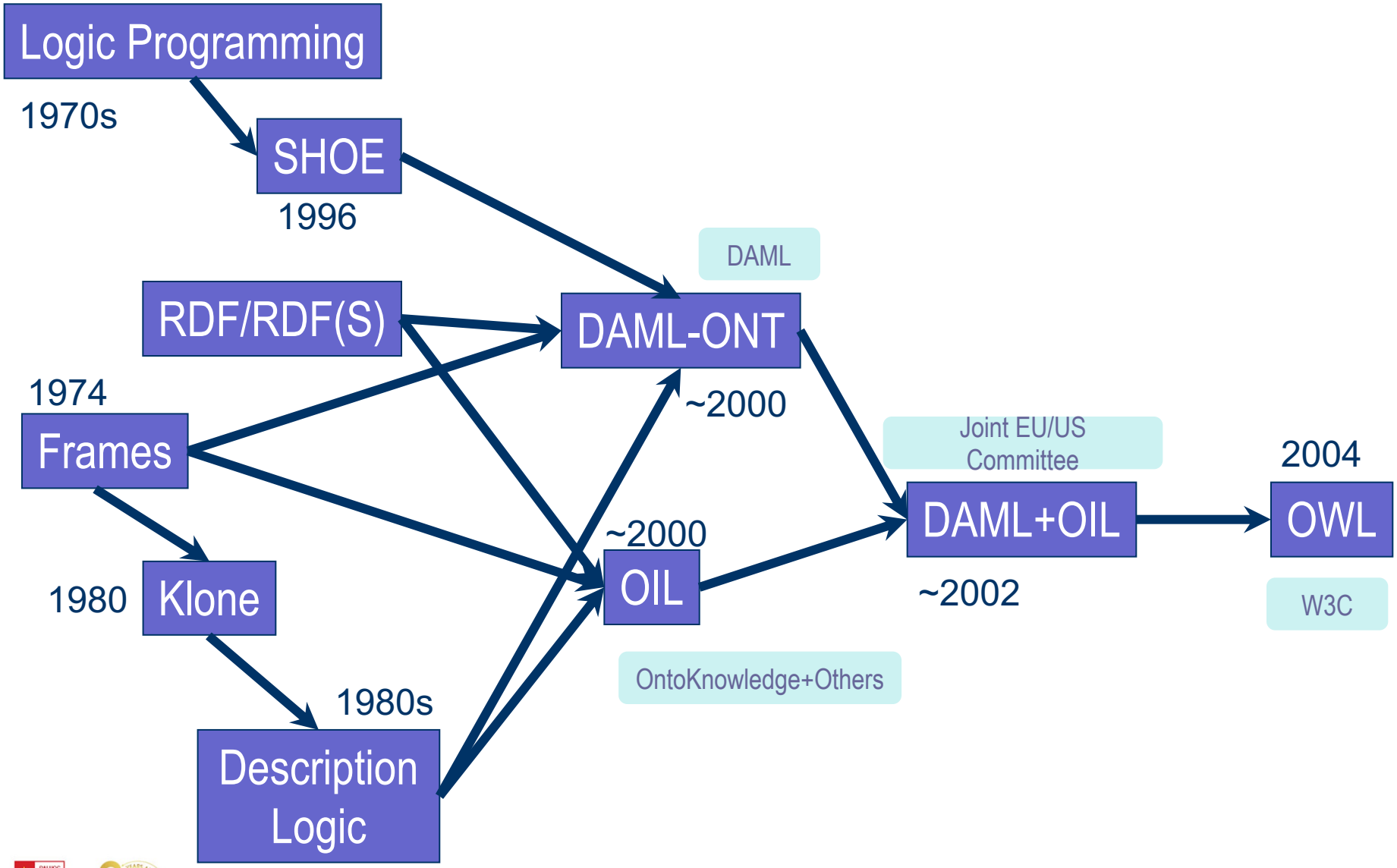
5.3. Các tầng OWL

5.4. OWL 2

Tiến trình phát triển của 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 kinh nghiệm 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.
- OWL sử dụng cú pháp RDF/XML và có ngữ nghĩa hình thức được xây dựng trên cơ sở lô-gic bậc nhất.
- Nhiều công cụ phần mềm chất lượng.

Sơ đồ tiến trình phát triển OWL



Một số điểm quan trọng đối với ngôn ngữ Ontology

- Ngôn ngữ Ontology được sử dụng để viết mô tả hình thức cho mô hình lĩnh vực. Vì vậy có một số điểm quan trọng như sau:
  - Khả năng diễn đạt đủ dùng
  - Ngữ nghĩa hình thức
  - Khả năng suy diễn hiệu quả
  - Cú pháp rõ ràng
  - Cách diễn đạt 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 là không khả quyết hoặc bán khả quyết và nếu khả quyết thì có độ phức tạp lũy thừa
- Chúng ta cần cân đối giữa:
  - Khả năng suy diễn hiệu quả
  - Khả năng mô tả nhiều ontologies và tri thức phức tạp

Một số hình thức suy diễn về tri thức

- Thành phần của lớp
  - Nếu x là một phần tử của lớp C và C là lớp con của lớp D, thì chúng ta có thể kết luận x là phần tử của D
- 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, nhưng A và B không giao nhau
  - => Dấu hiệu của lỗi trong Ontology hoặc dữ liệu
- Phân lớp
  - Dựa trên một số cặp thuộc tính-giá trị có thể xác định một đối tượng là phần tử của lớp A.

Một số ứng dụng của 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 Ontology và tri thức
  - Kiểm tra những mối quan hệ giữa các lớp
- Những khả năng suy diễn là cần thiết để
  - Thiết kế những ontologies lớn, với nhiều người cùng tham gia biên soạn
  - Tích hợp và chia sẻ các ontologies từ những nguồn khác nhau

Thực hiện suy diễn OWL

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

Kết hợp OWL với lược đồ RDF

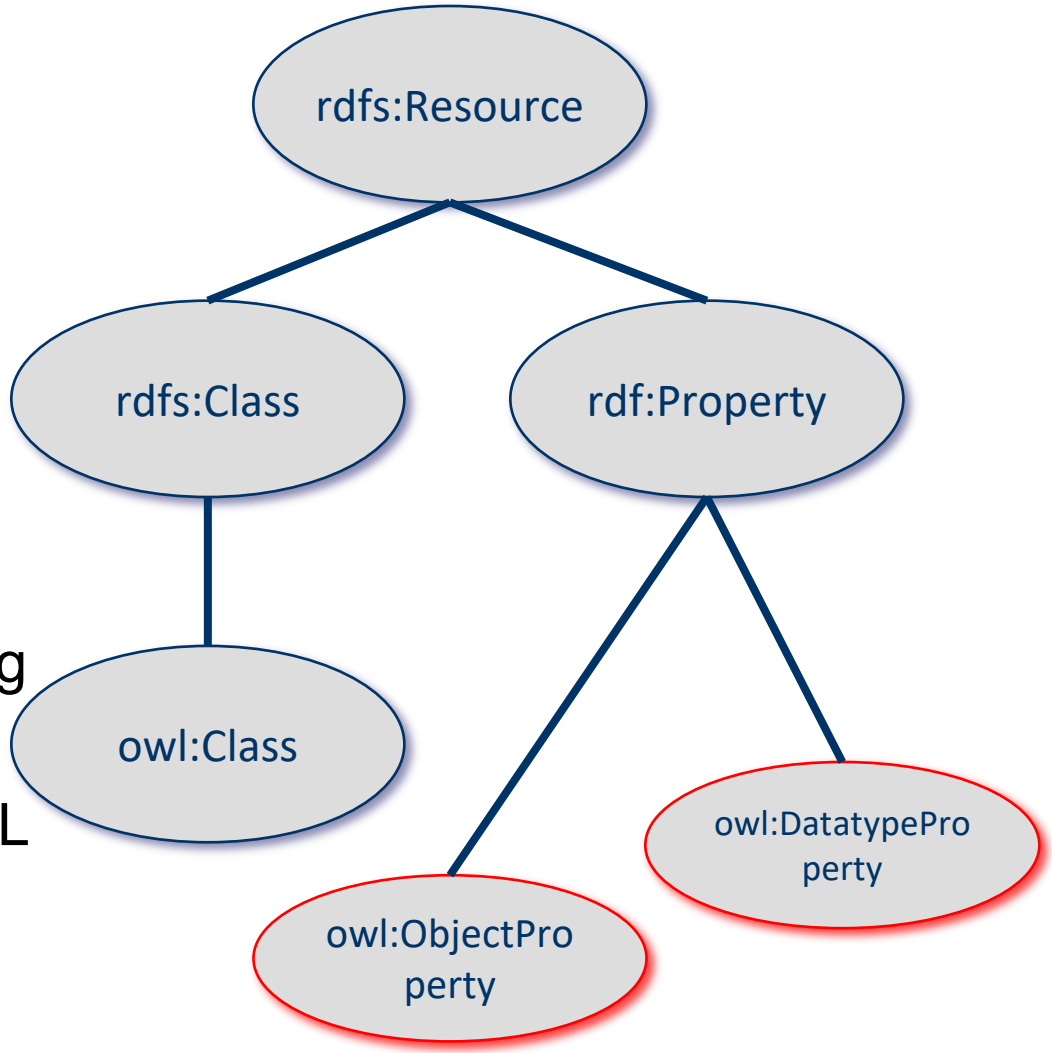
- Trong trường hợp lý tưởng, OWL cần mở rộng lược đồ RDF, nhất quán với kiến trúc phân tầng của Web ngữ nghĩa.
- Nhưng đơn giản mở rộng lược đồ RDF đi ngược với mục tiêu khả năng diễn đạt và suy diễn hiệu quả
  - Kết hợp lược đồ RDF với lô-gic dẫn đến các vấn đề tính toán không kiểm soát được.
- OWL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

Mô tơ suy diễn *đúng đắn* và *đầy đủ*

- Một mô-đun suy diễn *đúng đắn* chỉ đưa ra các kết luận có thể suy ra được từ đầu vào theo lô-gic - tất cả các kết luận của nó là đúng
  - Chúng ta thường yêu cầu các mô-đun suy diễn phải 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
  - Chúng ta không thể đảm bảo mô-đun suy diễn đầy đủ cho FOL đầy đủ và nhiều tập con của nó
  - Vì vậy mô-tơ suy diễn đầy đủ không được xây dựng cho OWL Full.

Tính tương thích của OWL với lược đồ RDF

- 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 là chi tiết hóa của cấu trúc tương ứng trong RDF(S)
- Các lớp và thuộc tính OWL có các ràng buộc bổ xung so với RDF(S)



Các cú pháp OWL

- OWL được phát triển trên cơ sở RDF
  - Có thể viết các ontologies OWL bằng các định dạng RDF
- Các định dạng khác cho OWL cũng đã được xây dựng
  - Các định dạng chuyên dụng có thể dễ đọc, dễ viết hơn
  - Thường được sử dụng trong các công cụ soạn thảo Ontology, giống như Protege

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

Khai báo không gian tên OWL

@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 một số khai báo không gian tên nhất định
- Quy chuẩn OWL của W3C có không gian tên là http://www.w3.org/2002/07/owl#

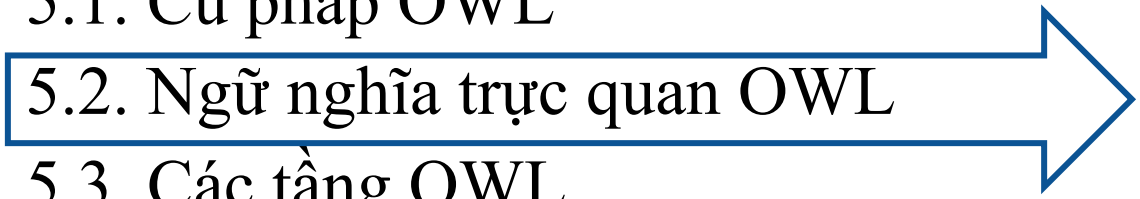
Nội dung

5.1. Cú pháp OWL

5.2. Ngữ nghĩa trực quan OWL

5.3. Các tầng OWL

5.4. OWL 2



owl:Ontology

<> a owl:Ontology;
rdfs:comment "Example OWL ontology";
owl:priorVersion <http://example.org/uni-ns-old>;
owl:imports <http://example.org/persons> ;
rdfs:label "University Ontology" .

- owl:imports, một thuộc tính bắc cầu, chỉ thị nhập toàn bộ tài liệu vào tài liệu đích giống như được khai báo trong tài liệu đích.
- owl:priorVersion chỉ tới một phiên bản sớm hơn của tài liệu này

Các lớp OWL

:AssociateProfessor a owl:Class ;
owl:disjointWith (:Professor :AssistantProfessor) .

- Các lớp được định nghĩa bằng owl:Class
  - owl:Class là lớp con của lớp rdfs:Class
- owl:Class không giao cắt với các kiểu dữ liệu (các hằng giá trị)
- Tính không giao cắt được định nghĩa bằng owl:disjointWith
  - Hai lớp không giao cắt không có phần tử chung

Một ví dụ khác

:Man rdfs:subClassOf foaf:Person .

:Woman rdfs:subClassOf foaf:Person .

:Man owl:disjointWith :Woman .

Câu hỏi:

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

Các lớp OWL

:Faculty a owl:Class;

owl:equivalentClass :AcademicStaffMember .

- owl:equivalentClass cho rằng hai lớp là tương đương
  - Mỗi lớp đều có các phần tử giống nhau
- owl:Thing là lớp khái quát nhất, chứa tất cả mọi thứ
  - Ví dụ, tất cả các lớp owl đều là rdfs:subClassOf owl:Thing
- owl:Nothing là lớp rỗng
  - Ví dụ, owl:Nothing là rdfs:subClassOf tất cả các lớp owl

Các thuộc tính OWL

- OWL có hai loại thuộc tính
- Thuộc tính đối tượng gắn các đối tượng với các đối tượng khác
  - owl:ObjectProperty, ví dụ, isTaughtBy, supervises
- Thuộc tính kiểu dữ liệu gắn các đối tượng với kiểu dữ liệu
  - owl:DatatypeProperty, ví dụ, SĐT, tên, tuổi, v.v…
- Được tách biệt để dễ dàng hơn cho việc xây dựng các mô-đun suy diễn đúng đắn và đầy đủ.

Các thuộc tính kiểu dữ liệu

- OWL sử dụng các kiểu dữ liệu lược đồ XML, khai thác kiến trúc phân tầng của Web ngữ nghĩa.

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

Các thuộc tính đối tượng OWL

Thường là các kiểu dữ liệu do người dùng tự định nghĩa.

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

Các thuộc tính nghịch đảo

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

Hoặc chỉ đơn giản là:

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

Một phần danh sách mệnh đề:

```
 owl:inverseOf rdfs:domain owl:ObjectProperty;
    rdfs:range owl:ObjectProperty;
    a owl:SymmetricProperty.
 {?P  owl:inverseOf ?Q. ?S ?P ?O} => {?O ?Q ?S}.
 {?P owl:inverseOf ?Q. ?P  rdfs:domain ?C} => {?Q rdfs:range ?C}.
 {?A owl:inverseOf ?C. ?B owl:inverseOf ?C} =>
          {?A rdfs:subPropertyOf ?B}.
```

Các thuộc tính tương đương

:lectures owl:equivalentProperty :teaches .

- Hai thuộc tính có cùng mở rộng
  - Tập tất cả các cặp subject-object mà nó gắn kết.
- Mệnh đề

  {?A rdfs:subPropertyOf ?B .
  ?B rdfs:subPropertyOf ?A.}
  <=> {?A owl:equivalentProperty ?B .}.

Giới hạn thuộc tính

- Khai báo rằng lớp C thỏa mãn các điều kiện nhất định
  - Tất cả các phần tử của C đáp ứng các điều kiện đó
- Điều này tương đương với: C là lớp con của một lớp C', trong đó C' tập hợp tất cả các đối tượng thỏa mãn các điều kiện đó (C' có thể là ẩn danh)
- Ví dụ:
  - Người có giới tính nữ và có ít nhất một con.
  - Những thứ có chính xác hai chân và hai tay

Các giới hạn thuộc tính (2)

- Phần tử owl:Restriction mô tả một lớp như thế.
- Phần tử có một thuộc tính owl:onProperty và một hoặc nhiều hơn các khai báo giới hạn (restriction declarations).
- VD giới hạn về số lượng:
  Một phụ huynh phải có tối thiểu một con

```
:Parent rdfs:subClassOf
      [a owl:Restriction;
        owl:onProperty :hasChild;
        owl:minCardinalityQ "1" ] .
```

Ví dụ: Giới hạn thuộc tính

- Câu sau mô tả phụ huynh là bất kỳ người nào có thể có tối thiểu một con (Parent is any Person who has at least one child):

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

Các giới hạn thuộc tính (3)

- Các kiểu giới hạn khác mô tả các ràng buộc về loại giá trị mà thuộc tính có thể nhận
  - **owl:allValuesFrom** xác định giá trị khái quát
  - **owl:hasValue** đặc tả một giá trị cụ thể
  - **owl:someValuesFrom** xác định sự tồn tại

owl:allValuesFrom

- Mô tả một lớp theo đó tất cả các giá trị của một thuộc tính thỏa mãn yêu cầu đặt ra
- Ví dụ: lớp (owl:Class) các môn toán được dạy bởi các giáo sư:

```
[a :mathCourse,
    [a owl:Restriction;
     owl:onProperty :isTaughtBy;
     owl:allValuesFrom :Professor] ].
```

Ví dụ: Offspring của người là người

:Person *a owl:Class,*

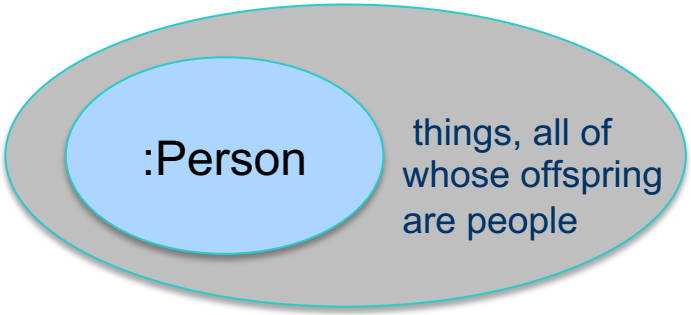
rdfs:subClassOf

[a owl:Restriction;

owl:onProperty bio:offspring;

owl:allValuesFrom :Person] .

Một lớp của tất cả các phần tử có offspring là người



Offspring của người là người

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

:john a :Person;
        bio:offspring :mary
```

Các hệ quả là gì

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

:bio:offspring rdfs:domain :animal;
                    rdfs:range :animal.
???
:alice a foaf:Person;
        bio:offspring :bob.
???
:carol a foaf:Person.
:don bio:offspring :carol.
???
```

"people give birth to people"

Các hệ quả (2)

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

bio:sprungFrom rdfs:domain :animal;
                    rdfs:range :animal;
                    owl:inverseOf bio:offspring.

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

"people are born of people"

owl:hasValue

- Mô tả một lớp với một giá trị cụ thể cho một thuộc tính
- Ví dụ, Môn toán được dạy bởi Giáo sư NVA

```
<!– Math courses taught by #949352 →
<owl:Class>
  <rdfs:subClassOf>rdf:resource="#mathCourse”/>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource= "#isTaughtBy"/>
      <owl:hasValue rdf:resource= "#949352"/>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

owl:hasValue

- Mô tả một lớp với một giá trị cụ thể cho một thuộc tính
- Ví dụ, Môn toán được dạy bởi Giáo sư NVA

```
# Math courses taught by NVA
[ rdfs:subclassOf :mathCourse,
  [ a owl:restriction;
      owl:onProperty :isTaughtBy;
      owl:hasValue :NVA] .
```

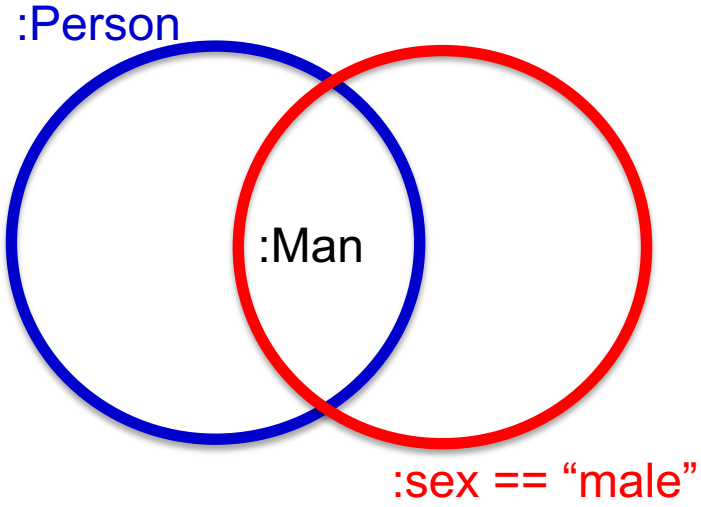
- Các câu hỏi:
  - Điều này có phải là tất cả các môn toán đều được dạy bởi NVA?
  - Điều này có phải là một số môn được dạy bởi NVA?

Ví dụ điển hình

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

Ví dụ: owl:hasValue

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



Có thể suy ra được gì

```
:ed a :Man
???
:frank a foaf:Person; :sex "male" .
???
:pat a foaf:Person; :sex "male"; sex "female" . ???
```

Lớp là tập hợp trong OWL

owl:someValuesFrom

- Mô tả lớp, yêu cầu nó phải có ít nhất một giá trị cho thuộc tính khớp với mô tả
- Ví dụ, cán bộ có dạy một môn trình độ đại học

```
[ a :academicStaffMember;
    a [owl:onProperty :teaches;
       owl:someValuesFrom :undergraduateCourse]]
```

Giới hạn cơ số

- Chúng ta có thể thiết lập giá trị tối thiểu và tối đa 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
- Có thể thiết lập giá trị cụ thể bằng cách sử dụng cùng một giá trị cho cực tiểu và cực đại
  - Khóa với đúng 7 sinh viên
- Để thuận tiện, OWL đưa ra owl:cardinality cho mô tả này.

Ví dụ: Giới hạn cơ số

Ví dụ, khóa được dạy bởi ít nhất hai người

[a owl:Restriction;

owl:onProperty :isTaughtBy;

owl:minCardinality “2”^^xsd;nonNegativeInteger] .

Điều này cho biết thông tin gì

:Parent owl:equivalentClass

[a owl:Restriction;

owl:onProperty :hasChild;

owl:minCardinality “1”^^xsd:integer] .

Câu hỏi:

- Parents có bắt buộc phải là human không?
- Children của Parents có bắt buộc phải là human không?

Định nghĩa Parent

- Parent là một lớp tương đương với lớp của các phần tử có ít một một con

  All(x):Parent(x) ⇔ Exists(y) hasChild(x, y)

Nếu hasChild được định nghĩa có miền là Person, thì Parents cũng là người.

Các tính chất đặc biệt

- owl:TransitiveProperty (thuộc tính bắc cầu)
  - Ví dụ, "có điểm tốt hơn", "là tổ tiên của"
- owl:SymmetricProperty (đối xứng)
  - Ví dụ, "có cùng điểm với", "là láng giềng của"
- owl:FunctionalProperty định nghĩa một thuộc tính có **tối đa một giá trị cho mỗi đối tượng**
  - Ví dụ, "tuổi", "chiều cao", "người hướng dẫn trực tiếp"
- owl:InverseFunctionalProperty xác định một thuộc tính mà **hai đối tượng không thể có cùng giá trị**
  - Ví dụ, "ssn", "số điện thoại di động"

Biểu thức Boolean

- Chúng ta có thể kết hợp các lớp bằng toán tử Boolean (hợp, giao, phần bù)
- Phủ định được mô tả bởi complementOf (phần bù), ví dụ, Khóa không được dạy bởi staffMembers

```
[ a :course,
    owl:Restriction;
       owl:onProperty :taughtBy;
       owl:allValuesFrom [a owl:Class;
                                owl:complementOf :staffMember]
 ] .
```

Biểu thức Boolean

- Lớp mới không phải là lớp con của hợp, nhưng bằng với hợp
  - Chúng ta phải khẳng định các lớp tương đương
- Ví dụ, người trong trường đại học là hợp của cán bộ và sinh viên

```
:peopleAtUni
    owl:equivalentClass
        owl:unionOf (:staffMember :student) .
```

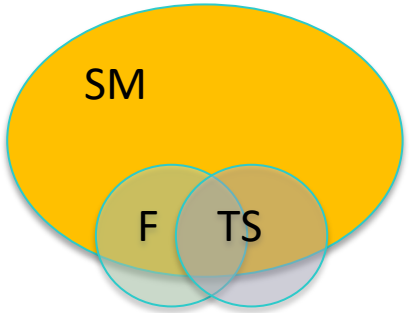
Biểu thức Boolean

Ví dụ, CS faculty là giao của faculty và những thứ thuộc về bộ phận CS Department

```
:facultyInCS owl:equivalentClass
  owl:intersectionOf
     (:faculty
       [ a owl:Restriction;
         owl:onProperty :belongsTo;
         owl:hasValue :CSDepartment ]
     ) .
```

Các toán tử Boolean lồng nhau

Ví dụ, ban quản lý là thành viên không phải là faculty hoặc đội ngũ kỹ thuật




```
:adminStaff owl:equivalentClass
  owl:intersectionOf
    (:staffMember
     [a owl:Class;
      owl:complementOf [a owl:Class;
                        owl:equivalentClass
                        owl:unionOf (:faculty :techSupportStaff)]])
```

Liệt kết với owl:oneOf

Ví dụ, một thứ có thể là Monday, Tuesday, …

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

Khai báo phần tử

Các phần tử của một lớp OWL được khai báo trong RDF như sau:

```
:john
    a :academicStaffMember;
    uni:age 39^^xsd:integer.
```

Giả thuyết không có tên duy nhất

- OWL không áp dụng giả thuyết tên duy nhất như trong các hệ CSDL
  - Hai phần tử có tên hoặc ID khác nhau không có nghĩa rằng đó là hai phần tử khác nhau.
- Giả sử chúng ta khẳng định mỗi khóa được dạy bởi tối đa một giảng viên, và một khóa được dạy bởi #949318 và được dạy bởi #949351
  - Bộ suy diễn OWL không thiết lập cờ lỗi
  - Thay vào đó nó suy diễn hai tài nguyên là tương đương

Các đối tượng duy nhất

Để đảm bảo rằng các đối tượng được nhận diện duy nhất, chúng ta phải đặt giả thuyết tường minh về tính không tương đương

:john owl:differentFrom :mary.

Các đối tượng duy nhất (2)

OWL cung cấp cách viết tắt để kiểm tra rằng các phần tử là khác nhau theo cặp

[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 lớn hơn 18
- Những kiểu dữ liệu như vậy không dùng được trong OWL
  - Tài liệu OWL liệt kê tất cả các kiểu từ lược đồ XML có thể được sử dụng
  - Danh sách này bao gồm những kiểu thông dụng như string, integer, Boolean, time, và date

Suy diễn tính duy nhất

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

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

Ví dụ: Ontology động & thực vật Châu Phi

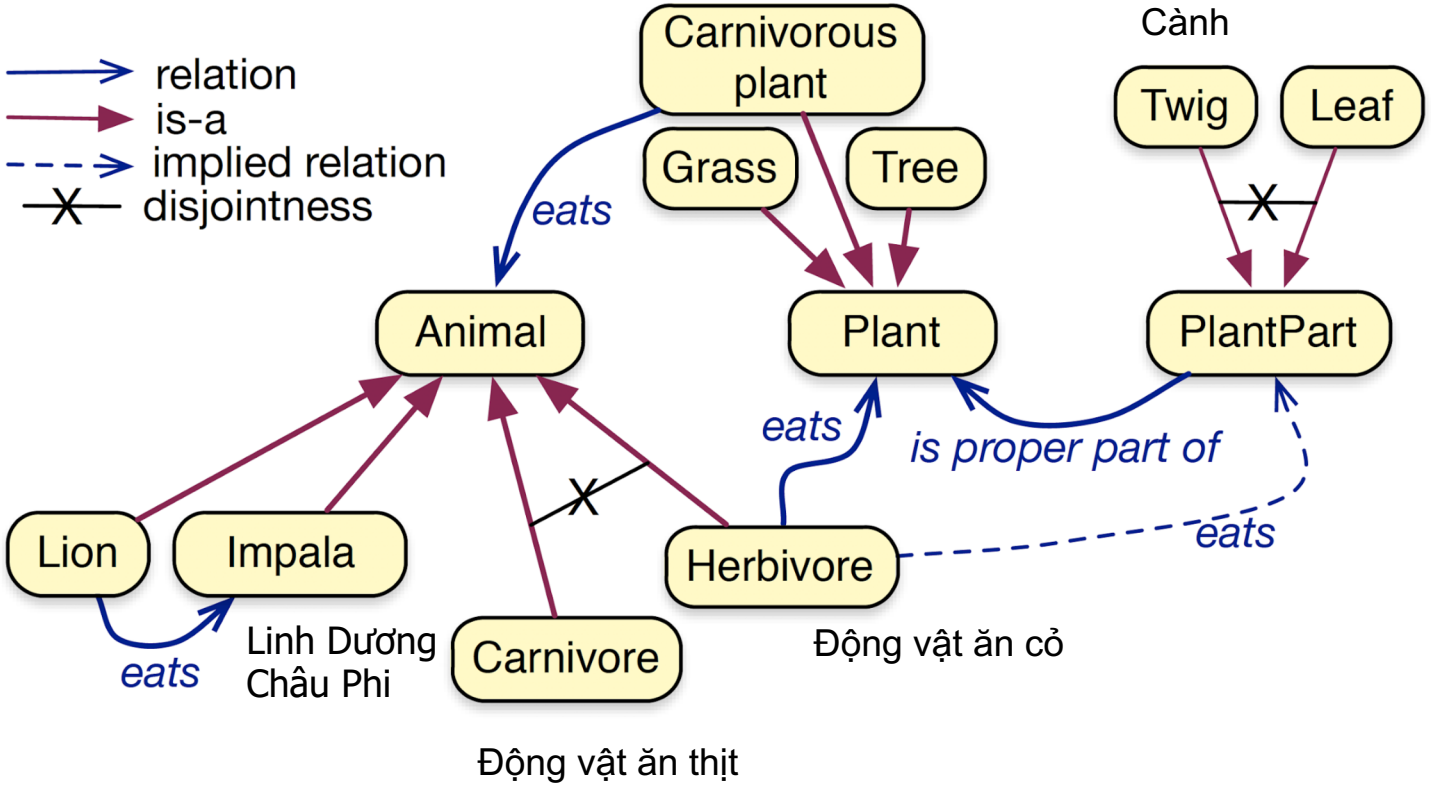
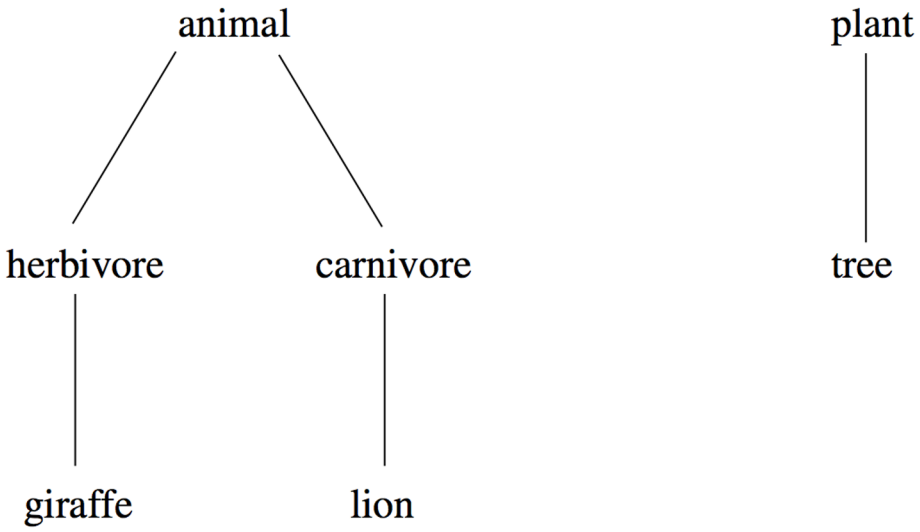
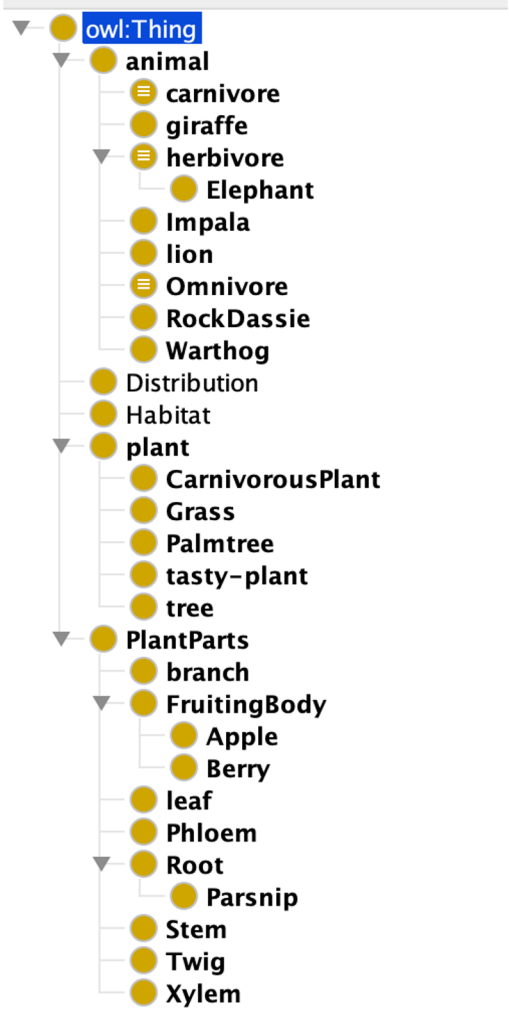


Figure 1 The African Wildlife Ontology at a glance. The main classes and relations of the African Wildlife ontology (v1) and an illustrative selection of its subclasses.

Được sử dụng trong phiên bản 2nd của The Semantic Web Primer

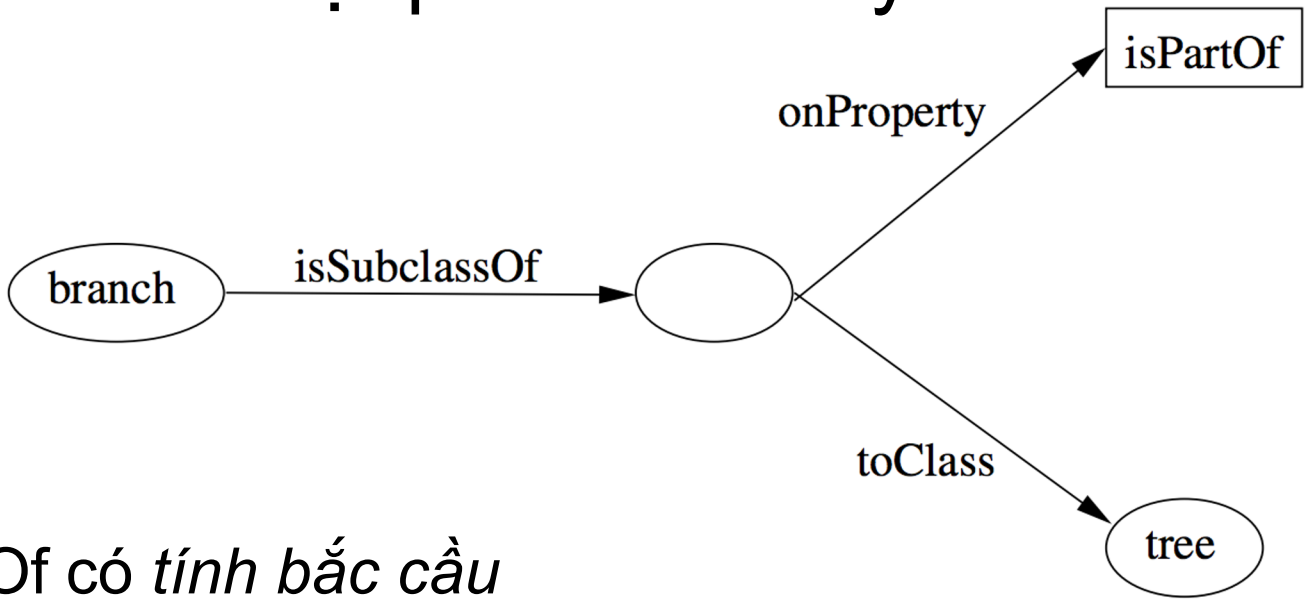
Ví dụ cuộc sống hoang dã Châu Phi: Các Lớp






```
:animal owl:disjointWith :plant .
:herbivore rsds:subClassOf :animal;
  owl:disjointWith :carnivore .
:giraffe rdfs:subClassOf :herbivore .
:carnivore rdfs:subClassOf :animal .
:lion rdfs:subClassOf :carnivore .
```

Các nhánh là một phần của cây



isPartOf có *tính bắc cầu*
:isPartOf a owl:TransitiveProperty .

chỉ có động vật ăn các thứ
:eats :domain :animal.

thuộc tính ngược của ăn là bị ăn
:eats owl:inverseOf :eatenBy.

Các nhánh (2)

```
# Thực vật và động vật không giao cắt
:plant owl:disjointWith :animal

# cây là thực vật
:tree rdfs:subClassOf :plant

# 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:someValuesFrom :Animal
            owl:onProperty :eats]
    ) .
```

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

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] .
```

TastyPlant

TastyPlant 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 .]
```

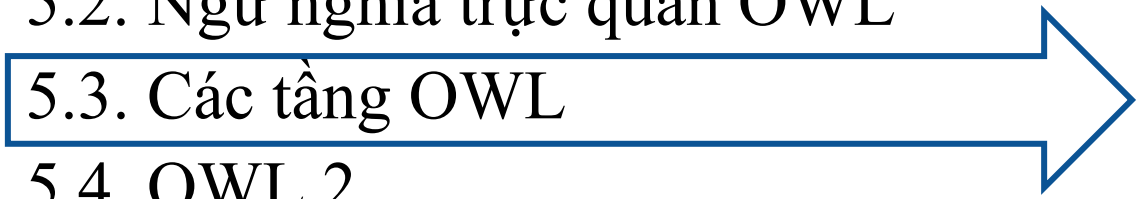
Nội dung

5.1. Cú pháp OWL

5.2. Ngữ nghĩa trực quan OWL

5.3. Các tầng OWL

5.4. OWL 2



Phân loại OWL

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

OWL Full

- Sử dụng tất cả các thành phần của OWL
- Cho phép kết hợp các thành phần này theo cách tùy ý với RDF và lược đồ RDF
- OWL Full hoàn toàn tương thích với RDF, cả cú pháp và ngữ nghĩa
- Khả năng diễn đạt của OWL Full là quá mạnh, tới mức phạm vi suy diễn của nó là không thể thực hiện

OWL DL

- OWL DL (Lô-gic mô tả - Description Logic) là một tập con của OWL Full giới hạn ứng dụng các thành phần từ OWL và RDF
  - Tương ứng với lô-gic mô tả
- OWL DL cho phép hỗ trợ suy diễn hiệu quả
- Nhưng chúng ta mất tính tương thích với RDF
  - Không phải tất cả các tài liệu RDF đều là 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 ngôn ngữ này:
  - Đơn giản đối với người dùng
  - Triển khai, cho những người làm 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

- Phân loại từ trong bộ từ vựng

Tài nguyên chỉ có thể là lớp, kiểu dữ liệu, hoặc thuộc tính của kiểu dữ liệu, thuộc tính đối tượng, một phần tử, một giá trị dữ liệu, hoặc một phần của bộ từ vựng định sẵn.

- Định kiểu tường minh

Phân loại tài nguyên phải được thực hiện tường minh (ví dụ, một lớp phải được khai báo nếu được sử dụng kết hợp với rdfs:subClassOf).

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

- Tách biệt thuộc tính
  - Tập mở rộng của các thuộc tính đối tượng và tập mở rộng của các thuộc tính dữ liệu không giao nhau
  - Vì vậy không thiết lập được các tính chất sau cho các thuộc tính kiểu 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 các thuộc tính bắc cầu
  - ví dụ, người với nhiều hơn 5 hậu duệ
- Các lớp ẩn danh bị giới hạn:
  - Các lớp ẩn danh chỉ được phép xuất hiện như:
    - Miền và khoảng của owl:equivalentClass hoặc owl:disjointWith
    - Khoảng (không cho phép miền) 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, chính xác) chỉ có thiết lập với giá trị 0 hoặc 1
- Không được thiết lập owl:equivalentClass giữa các lớp ẩn danh, chỉ được thiết lập giữa các lớp có định danh

Nội dung

5.1. Cú pháp OWL
5.2. Ngữ nghĩa trực quan OWL
5.3. Các tầng OWL
5.4. 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 chèn

- Khả năng chèn của OWL là rất giới hạn
  - Nó chỉ cho phép chèn cả một ontology, không phải một 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 sẽ 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 nào về sự hình thành đúng của 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 tên khác nhau là các phần tử khác nhau
- OWL sử dụng cách tiếp cận lô-gic, các phần tử có tên khác nhau chưa chắc đã khác nhau
  - Hợp lý 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 khái niệm khá thịnh hành 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) trên các thuộc tính khác.
- Tích hợp biểu diễn tri thức dựa trên luật và biểu biễn tri thức dựa trên lô-gic mô tả là một 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 tiễn, các mở rộng đang được thực hiện
  - Có gắng tiếp tục cung cấp các tính năng lô-gic, bao gồm cả các luật

ĐẠI HỌC BÁCH KHOA

25 YEARS ANNIVERSARY SOICT

VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

soict.hust.edu.vn/ fb.com/groups/soict