deti universidade de aveiro
departamento de electrónica,
telecomunicações e informática

Semantic Web

Engenharia de Dados e Conhecimento

2019/2020

Expressão da Semântica

O Triplo

A Sua Formalização

Grafos

Triplos

- No exemplo concreto, utilizado antes, que descreve Lugares, não existe nada no formato da tabela que restrinja o seu uso a locais com música e/ou que servem comida e bebidas.
- Na verdade, é perfeitamente possível representar factos acerca de outras entidades quaisquer no modelo chave-valor associado a IDs.
- O modelo de dados constituído por 3 campos é conhecido por triplo:
  - (ID – Propriedade – Valor)
  - Este consiste no bloco elementar e fundamental para implementar representações semânticas.

Triplos

- O modelo triplo é formalmente definido por:
  - (Sujeito – Predicado – Objeto)
- E possui os seguintes significados:
  - Sujeito = Entidade (Identificada por um ID)
    - uma "coisa" para a qual temos uma classe conceptual
    - exs: pessoas, lugares, objetos concretos, assim como coisas menos concretas, como períodos de tempo e ideias

Triplos

- … significados:
  - Predicado = Propriedade
    - uma propriedade da entidade, à qual está anexada
    - exs: nome de uma pessoa, data de nascimento, símbolo de uma ação no mercado de transações, endereços de email, etc.
  - Objeto = Valor (Entidade / Valor Literal)
    - os objetos podem ser de duas classes:
      - entidades, que por sua vez podem ser sujeitos noutros triplos
      - ou valores literais, como texto ou números.

Grafos

- Utilizando os mesmos sujeitos e/ou os mesmos objetos, em diferentes triplos, é possível:
  - manter unidos múltiplos triplos, passando a ter uma teia de triplos;
  - representar as relações semânticas entre múltiplos dados.

- Esta teia de relações forma um **grafo** orientado.

Grafos

- Flexibilizando o esquema particular anterior para que o ID da 1ª coluna represente qualquer entidade, obtemos:
  - Locais e vizinhanças no mesmo modelo.

Subject	Predicate	Object
S1	Cuisine	"Deli"
S1	Price	"$"
S1	Name	"Deli Llama"
S1	Address	"Peachtree Rd"
S2	Cuisine	"Chinese"
S2	Price	"$$$"
S2	Specialty Cocktail	"Scorpion Bowl"
S2	DJ?	"No"
S2	Name	"Peking Inn"
S2	Address	"Lake St"
S3	Live Music?	"Yes"
S3	Music Genre	"Jazz"
S3	Name	"Thai Tanic"
S3	Address	"Branch Dr"
S4	Name	"North Beach"
S4	Contained-by	"San Francisco"
S5	Name	"SOMA"
S5	Contained-by	"San Francisco"
S6	Name	"Gourmet Ghetto"
S6	Contained-by	"Berkeley"

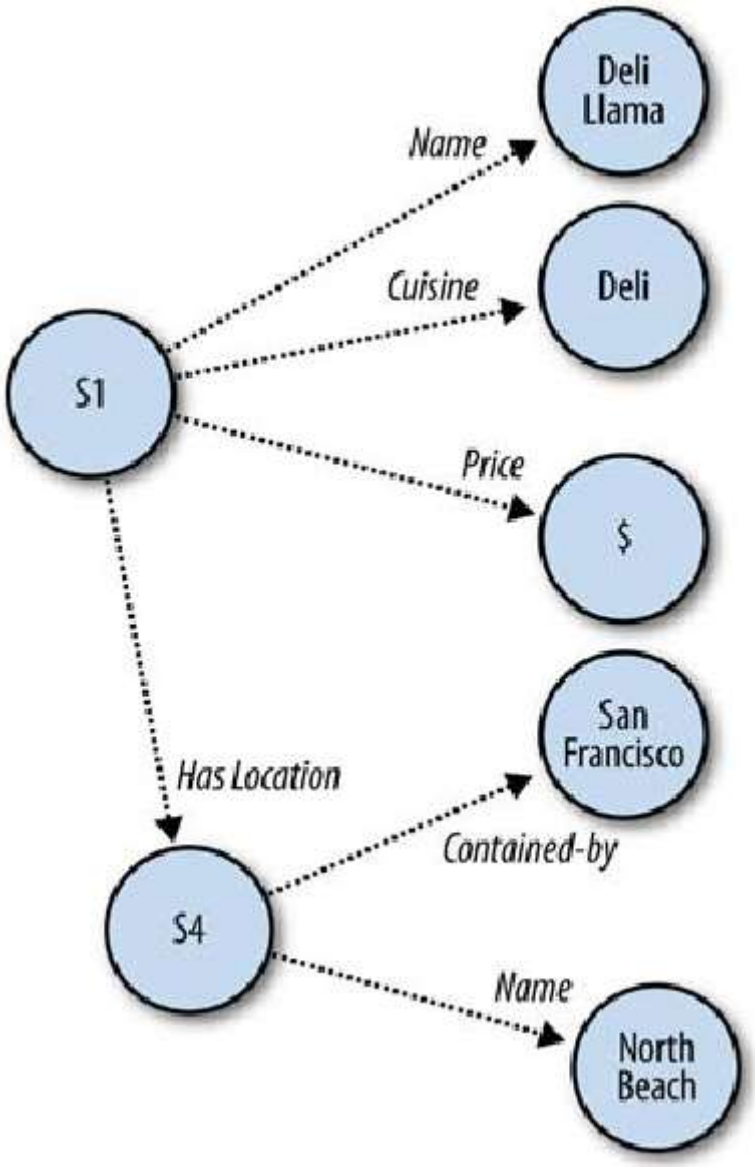
Grafos

- Na tabela anterior, as entidades não possuem ligação umas com as outras.
- Mas sabendo que é possível incluir sujeitos num triplo como objetos noutro triplo:
  - Podemos fazer:

Subject	Predicate	Object
S1	Has Location	S4
S2	Has Location	S6
S3	Has Location	S5

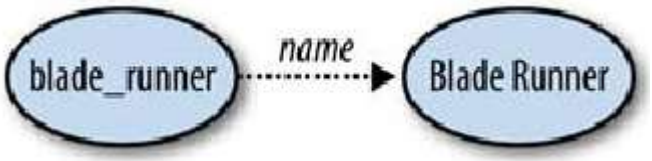
Grafos

- Diagrama representando um grafo com parte dos triplos das tabelas anteriores.



O Exemplo “Movies”

- Uso do modelo de triplos para representar os dados relativos a filmes.
- Exemplo de um triplo:
  - (blade_runner name “Blade Runner”)
    - blade_runner = sujeito, ID que identifica o filme
    - name = predicado, propriedade do filme
    - “Blade Runner” = objeto, valor tipo texto da propriedade
  - Descreve o título de um filme.



O Exemplo “Movies”

- De seguida, pretende-se declarar que o filme foi realizado por um determinado individuo.
  - Uma forma de o fazer:
    - (blade_runner directed_by “Ridley Scott”)
  - Esta forma tem 2 inconvenientes: o objeto, sendo um literal,
    - não pode ser sujeito noutros triplos o que não permite fazer declarações acerca do realizador
    - não identifica univocamente o realizador
  - O individuo “Ridley Scott” é uma pessoa e um realizador, entre mais coisas, o que o qualifica para ser uma entidade.

O Exemplo “Movies”

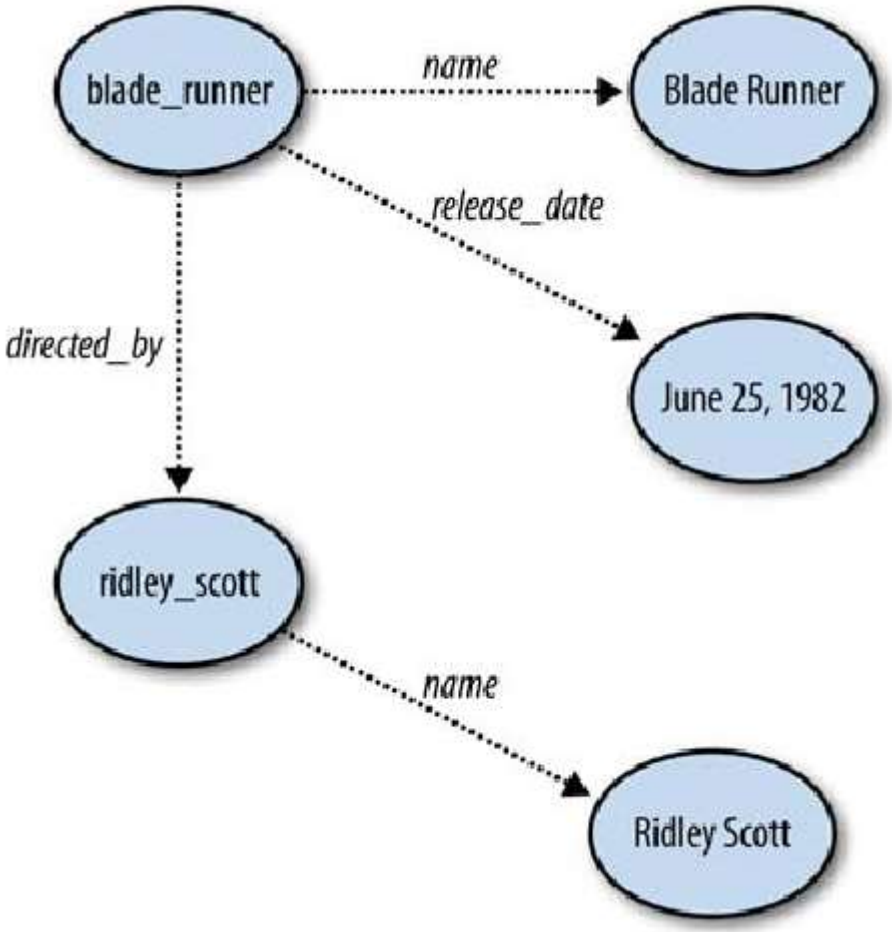
- Se for atribuído o id “ridley_scott” ao individuo, é então possível fazer as seguintes declarações:
  - (ridley_scott name “Ridley Scott”)
  - (blade_runner directed_by ridley_scott)

- É reutilizado o predicado <u>nome</u>, visto que este está a ser utilizado com o mesmo significado.

- Desta forma é implementado um triplo entre duas entidades em vez de uma entidade e um objeto.

O Exemplo “Movies”

- Se adicionarmos ainda o triplo seguinte
  - (blade_runner release_date "June 25, 1982")
  - Obtemos o grafo seguinte:



TripleStore
Uma Implementação

Filtragem

Triplestore

- Uma *Triplestore* serve precisamente para guardar, gerir e pesquisar triplos de dados.
- Existem presentemente várias implementações de t*riplestores* no mercado, contudo trabalhar com estas exige de imediato o domínio dos vários conceitos e padrões que compõem a web semântica.
- Presentemente, vamos recorrer a uma *triplestore* implementada em linguagem Python, que também ela possui vários toolkits para a programação da web semântica.
- Teremos assim a oportunidade de observar “por dentro” a sua construção.

Uma Implementação

- Esta implementação é baseada em dicionários e faz a indexação cruzada dos três termos do triplo, por forma a permitir o acesso direto aos triplos através de qualquer dos termos.
- Índices

```
class SimpleGraph:
    def __init__(self):
        self._spo = {}  # subject - predicate - object
        self._pos = {}  # predicate - object - subject
        self._osp = {}  # object - subject - predicate
```

  - Um índice vai consistir num dicionário de dicionários que contem *sets*. Ex:

```
self._spo = {subject:{predicate:set([object])}}
```

  - Um *set* é uma lista não ordenada com elementos únicos.

Uma Implementação

- O método add()
  - Adiciona todas as permutações dos termos do triplo a todos os índices.

```
def add(self, sub, pred, obj):
    self._addToIndex(self._spo, sub, pred, obj)
    ...
```

- O método _addToIndex()
  - Adiciona todos os termos do triplo ao índice, caso estes ainda não estejam presentes

```
def _addToIndex(self, index, a, b, c):
    if a not in index:
        index[a] = {b:set([c])}
    ...
```

Uma Implementação

- O método remove()
  - Remove o triplo de todos os índices

```
def remove(self, sub, pred, obj):
    triples = list(self.triples(sub, pred, obj))
    for (delSub, delPred, delObj) in triples:
        self._removeFromIndex(self._spo, delSub, delPred, delObj)
        ...
```

- O método _removeFromIndex()
  - Percorre o índice e limpa-o na remoção do triplo

```
def _removeFromIndex(self, index, a, b, c):
    try:
        bs = index[a]
        cset = bs[b]
        cset.remove(c)
        ...
```

Uma Implementação

- Filtragem de Triplos
  - Método triples() – filtra os triplos existentes, dado um triplo padrão
  - Os termos com valor “None”, assumem o valor “*” (qualquer)

```
def triples(self, sub, pred, obj):
    try:
        if sub != None:
            if pred != None:
                # sub pred obj
                if obj != None:
                    if obj in self._spo[sub][pred]:
                        yield (sub, pred, obj)
                # sub pred None
                else:
                    for retObj in self._spo[sub][pred]:
                        yield (sub, pred, retObj)
            ...
```

Uma Implementação

- O método load()
  - Carrega os índices com triplos lidos de um ficheiro CSV

```
def load(self, filename):
    f = open(filename, "r", newline="", encoding="utf-8")
    reader = csv.reader(f)
    for sub, pred, obj in reader:
        self.add(sub, pred, obj)
    f.close()
```

Uma Implementação

- O método save()
  - Guarda todos os triplos num ficheiro CSV

```
def save(self, filename):
    f = open(filename, "w", newline="", encoding="utf-8")
    writer = csv.writer(f)
    for sub, pred, obj in self.triples(None, None, None):
        writer.writerow([sub, pred, obj])
    f.close()
```

TripleStore
Uma Implementação

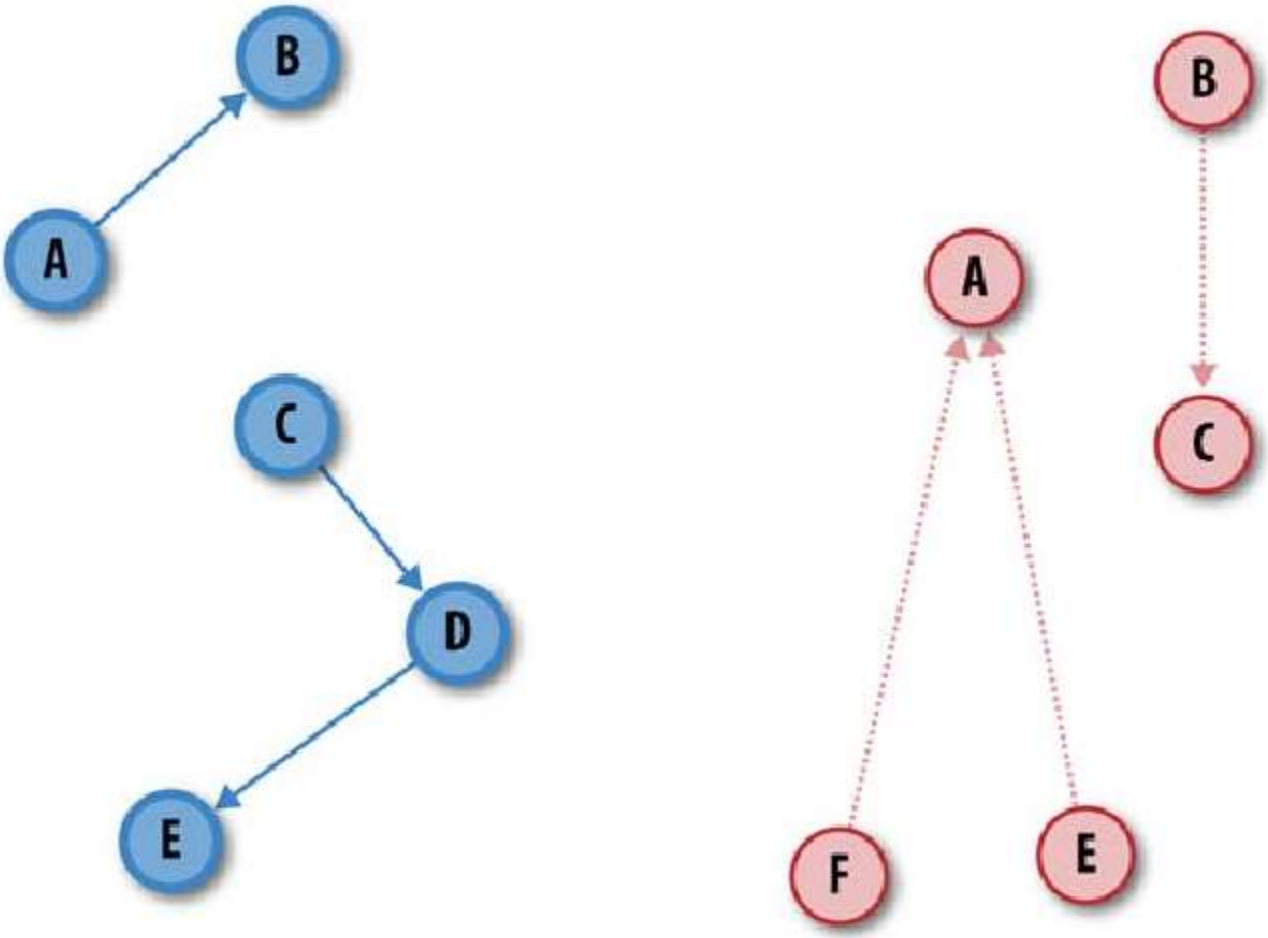
Fusão de Grafos

Fusão de Grafos

- A utilização de grafos na modelação de informação permite com facilidade a fusão entre múltiplos grafos diferentes.
  - Isto desde que estes possuam um sistema consistente de identificadores para os sujeitos e objetos.
  - Se um triplo se encontrar em mais que um grafo, os 2 triplos fundem-se de forma transparente, visto serem idênticos.
  - Isto é devido a que os nós e suas relações, nos grafos, são entidades de 1ª classe, isto é, são independentes e constituem “pedaços” de dados significativos.

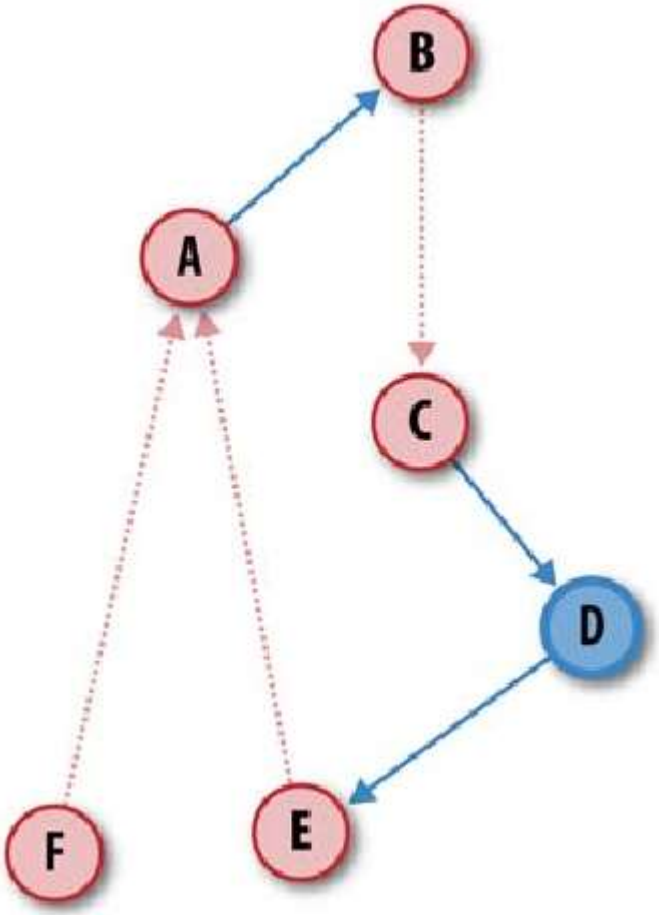
Fusão de Grafos

- Exemplo: grafos separados:



Fusão de Grafos

- Exemplo: fusão dos grafos



Fusão de Grafos

- Exemplo de uma função que recebe um grafo e faz a sua fusão com um grafo lido de um ficheiro:

```
def mergeFromFile(graph):
    tmp = SimpleGraph()
    tmp.load(input("Nome do ficheiro: "))
    for sub, pred, obj in tmp.triples(None, None, None):
        graph.add(sub, pred, obj)
```