

Stanford·CS520 | Knowledge Graphs (2021)

CS520(2021)·课程资料包 @ShowMeAI



视频

中英双语字幕



课件

一键打包下载



笔记

官方笔记翻译



代码

作业项目解析



视频·B站 [扫码或点击链接]

<https://www.bilibili.com/video/BV1hb4y1r7fE>



课件 & 代码·博客 [扫码或点击链接]

<http://blog.showmeai.tech/cs520>

斯坦福

实体关系

图谱应用

图谱构建

图谱 schema

实体

非结构化数据

知识图谱

知识推理

Awesome AI Courses Notes Cheatsheets 是 [ShowMeAI](#) 资料库的分支系列，覆盖最具知名度的 **TOP20+** 门 AI 课程，旨在为读者和学习者提供一整套高品质中文学习笔记和速查表。

点击课程名称，跳转至课程**资料包**页面，**一键下载**课程全部资料！

| 机器学习 | 深度学习 | 自然语言处理 | 计算机视觉 |
|--|-------------------|--------------------|-------------------|
| Stanford · CS229 | Stanford · CS230 | Stanford · CS224n | Stanford · CS231n |
| # Awesome AI Courses Notes Cheatsheets·持续更新中 | | | |
| 知识图谱 | 图机器学习 | 深度强化学习 | 自动驾驶 |
| Stanford · CS520 | Stanford · CS224W | UCBerkeley · CS285 | MIT · 6.S094 |



微信公众号

资料下载方式 2: 扫码点击**底部菜单栏**

称为 **AI 内容创作者?** 回复 [添砖加瓦]

What are some Knowledge
Graph Inference Algorithms?



Introduction

- Knowledge Graph Retrieval
 - Query languages
 - SPARQL and Cypher
- Knowledge Graph Inference
 - Drawing conclusions that are not explicit in the knowledge graph
 - E.g., shortest path, concluding new connections



Outline

- Knowledge Graph Inference
 - Graph-based inference algorithms
 - Ontology-based inference algorithms

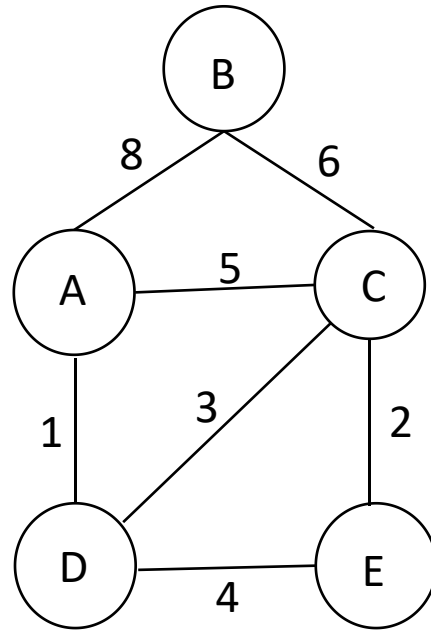


Graph-based Inference Algorithms

- Path finding
- Centrality Detection
- Community Detection

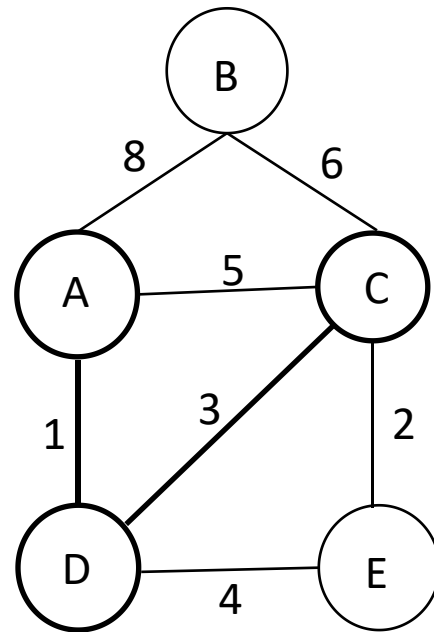


Path Finding



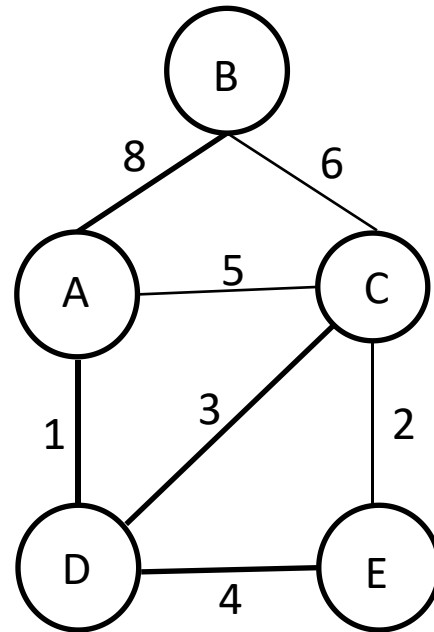
Path Finding

- Shortest path
 - Optimal path in traffic planning



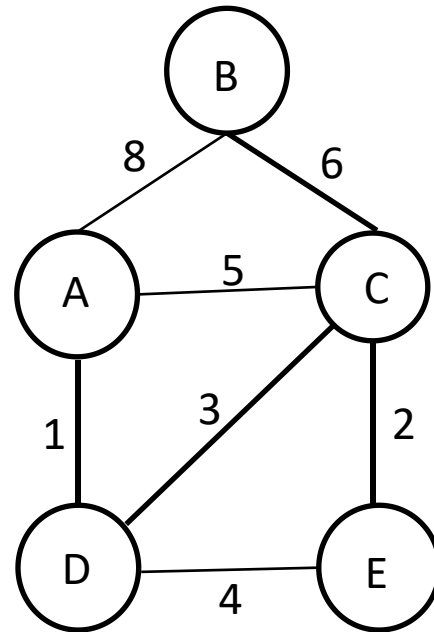
Path Finding

- Single Source Shortest path



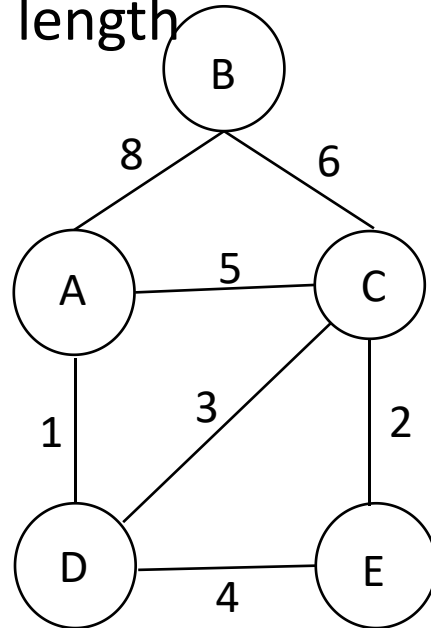
Path Finding

- Minimum Spanning Tree
 - Trip planning



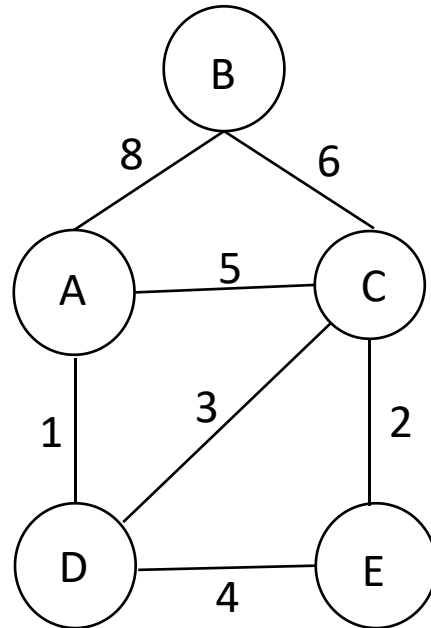
Path Finding

- A* Algorithm
 - Maintain a tree of paths from the start node
 - Extend them until termination criteria is met
 - Extend based on estimated length
 - $f(n) = g(n) + h(n)$
 - $f(n)$ = cost until now
 - $g(n)$ – current cost
 - $h(n)$ – estimated cost



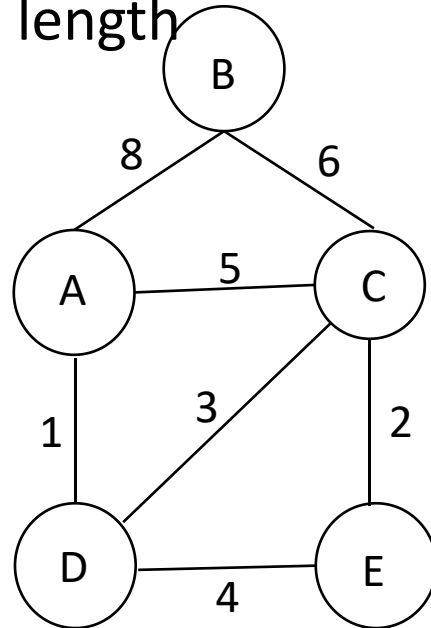
Path Finding

- A* Algorithm
 - Well-known path finding algorithm
 - Originally developed for AI planning



Path Finding

- A* Algorithm
 - Maintain a tree of paths from the start node
 - Extend them until termination criteria is met
 - Extend based on estimated length
 - $f(n) = g(n) + h(n)$
 - $f(n)$ = cost until now
 - $g(n)$ – current cost
 - $h(n)$ – estimated cost

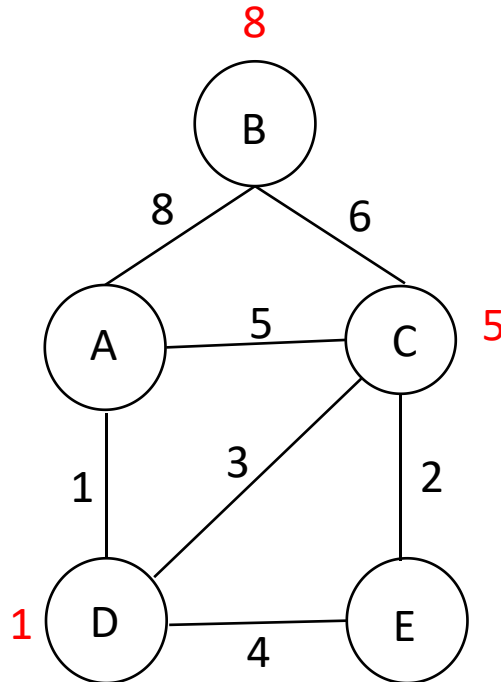


Admissible heuristic
- Never over estimate



Path Finding

- A* Algorithm
 - Shortest path from A to E
 - $h(n)=0$
 - Breadth first search
 - Dijkstra's algorithm

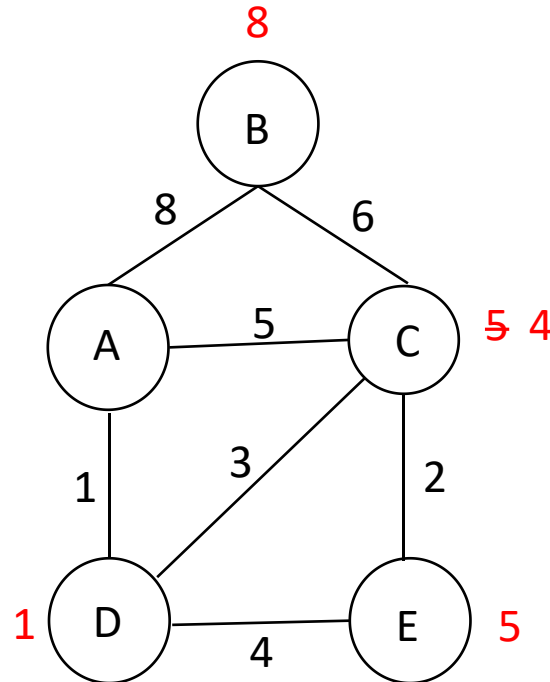


| | |
|------|---|
| A, B | 8 |
| A, C | 5 |
| A, D | 1 |



Path Finding

- A* Algorithm
 - Shortest path from A to E
 - $h(n)=0$
 - Breadth first search
 - Dijkstra's algorithm

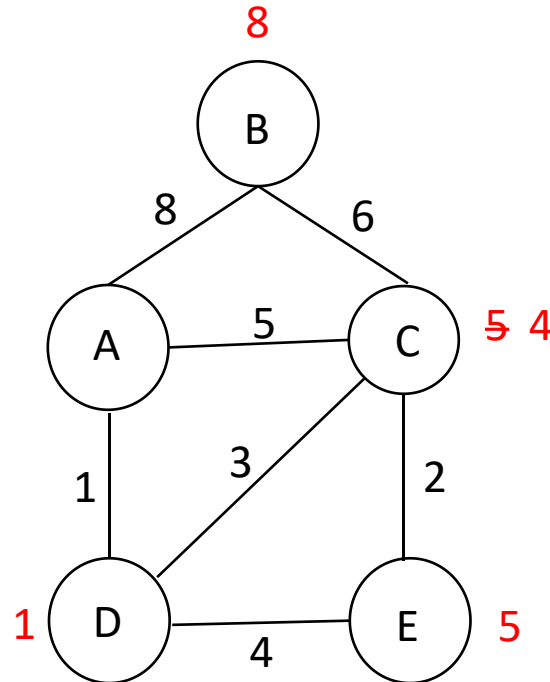


| | |
|---------|---|
| A, B | 8 |
| A, C | 5 |
| A, D | 1 |
| A, D, C | 4 |
| A, D, E | 5 |



Path Finding

- A* Algorithm
 - Shortest path from A to E
 - $h(n)=0$
 - Breadth first search
 - Dijkstra's algorithm



| | |
|------------|---|
| A, B | 8 |
| A, C | 5 |
| A, D | 1 |
| A, D, C | 4 |
| A, D, C, E | 6 |
| A, D, E | 5 |



Graph-based Inference Algorithms

- Path finding
- Centrality Detection
- Community Detection



Centrality Detection

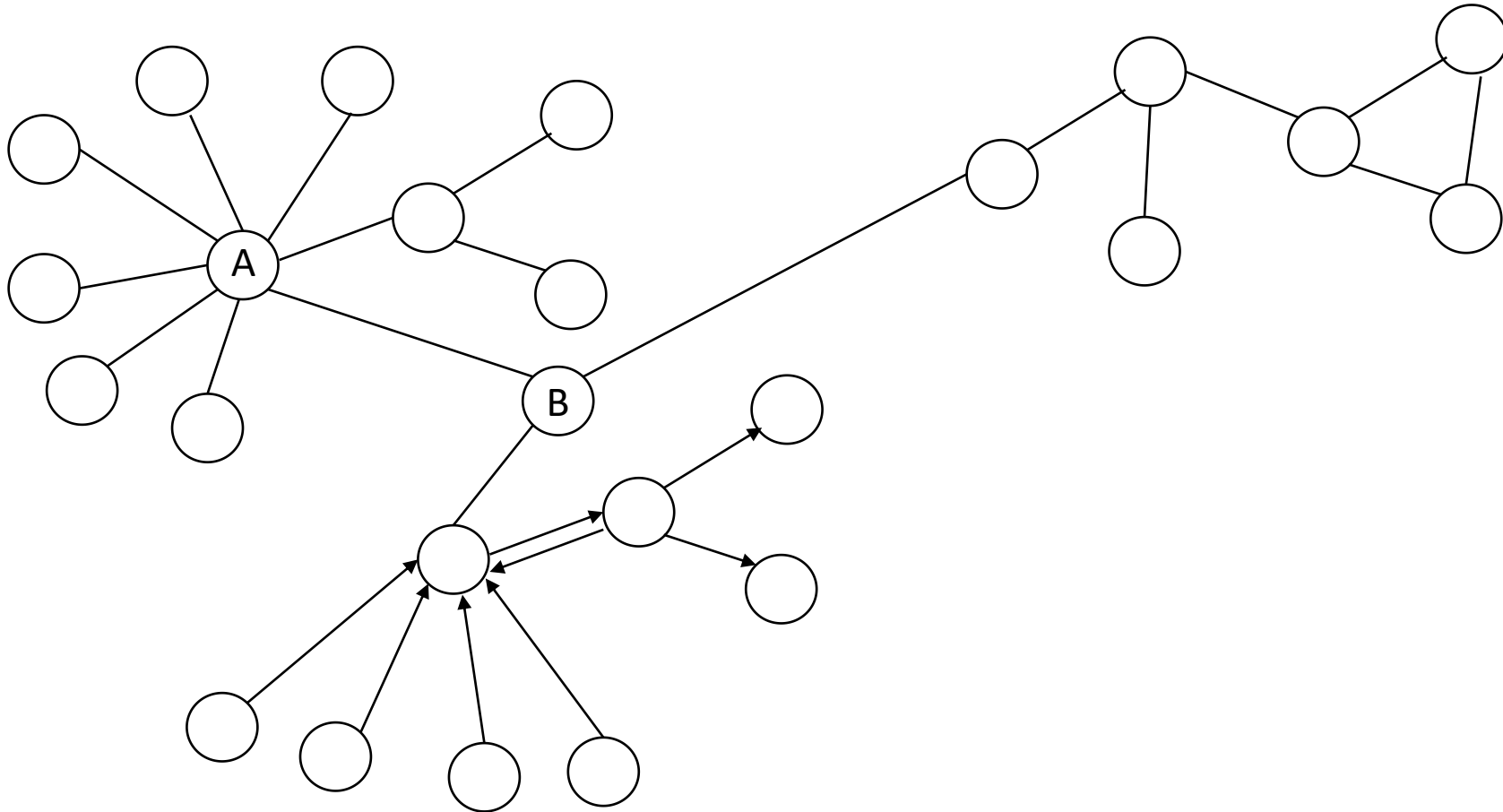
- Helps in understanding the importance of a node in a network
 - Most important nodes
 - Bridges in a network



Centrality Detection

- Helps in understanding the importance of a node in a network

Degree Centrality
A has the highest
degree

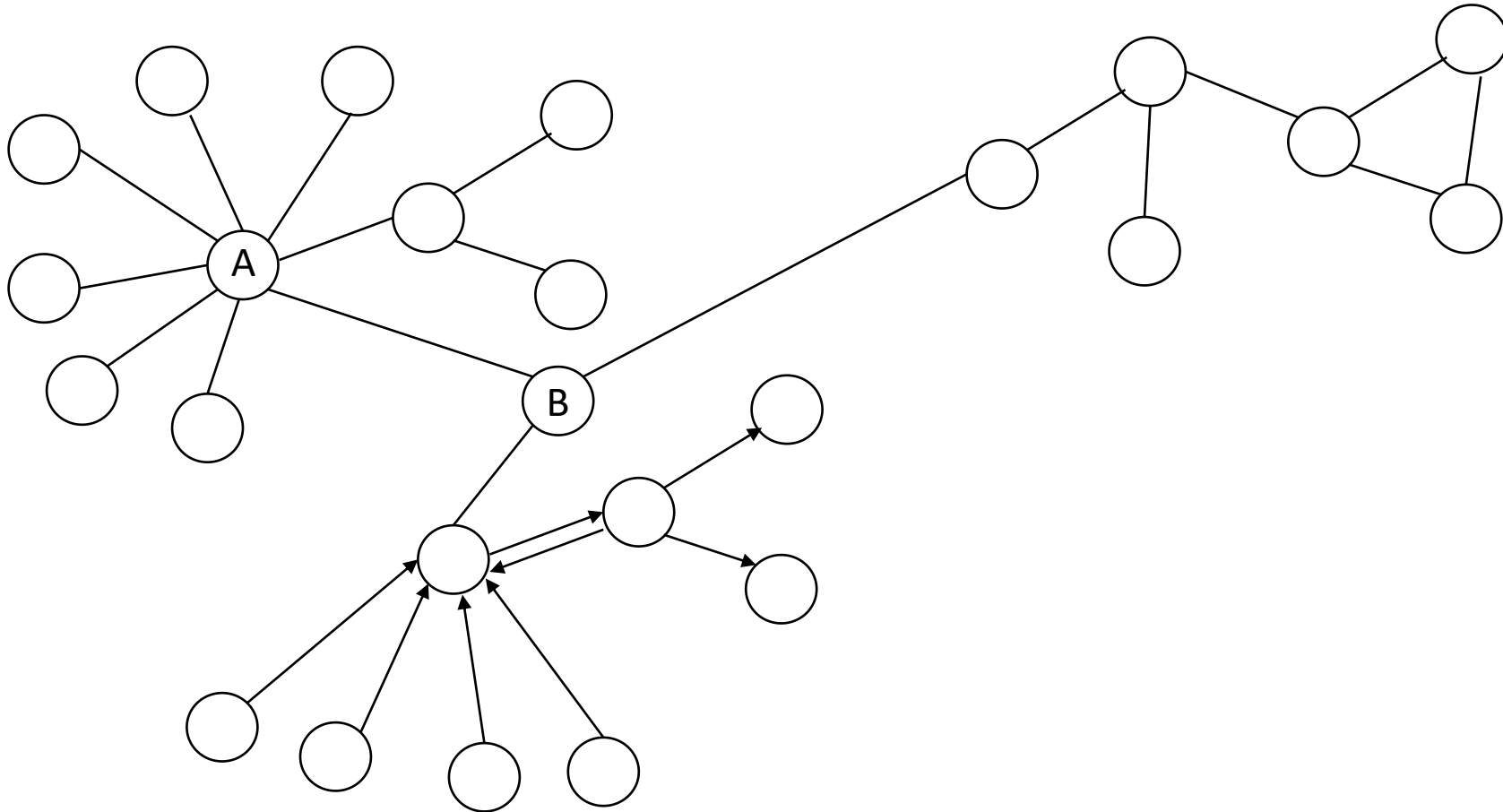


Centrality Detection

- Helps in understanding the importance of a node in a network

Degree Centrality
A has the highest
degree

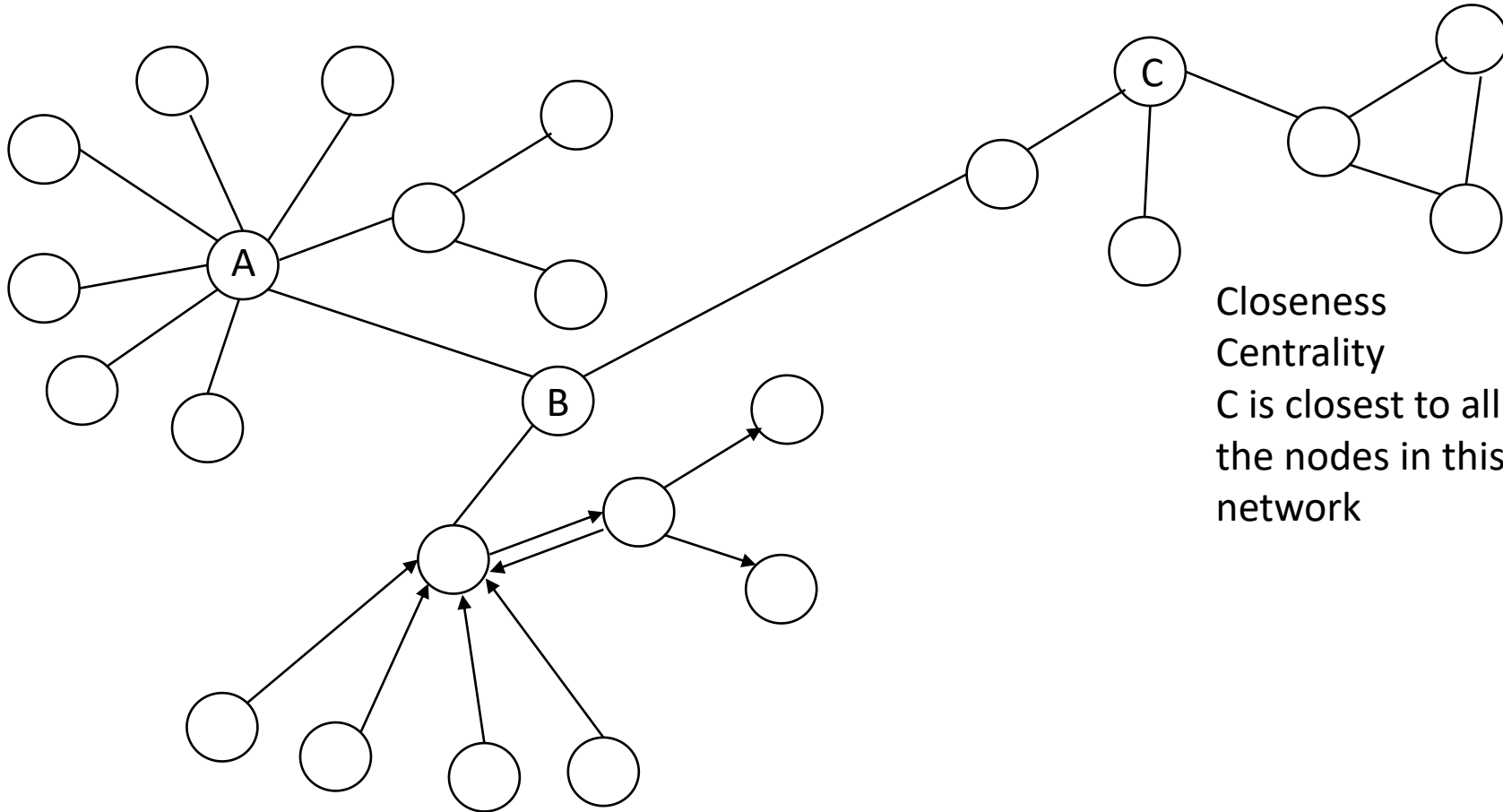
Betweenness
Centrality
B has the largest
number of
shortest paths
passing through it



Centrality Detection

- Helps in understanding the importance of a node in a network

Degree Centrality
A has the highest degree



Betweenness
Centrality
B has the largest
number of
shortest paths
passing through it

Closeness
Centrality
C is closest to all
the nodes in this
network

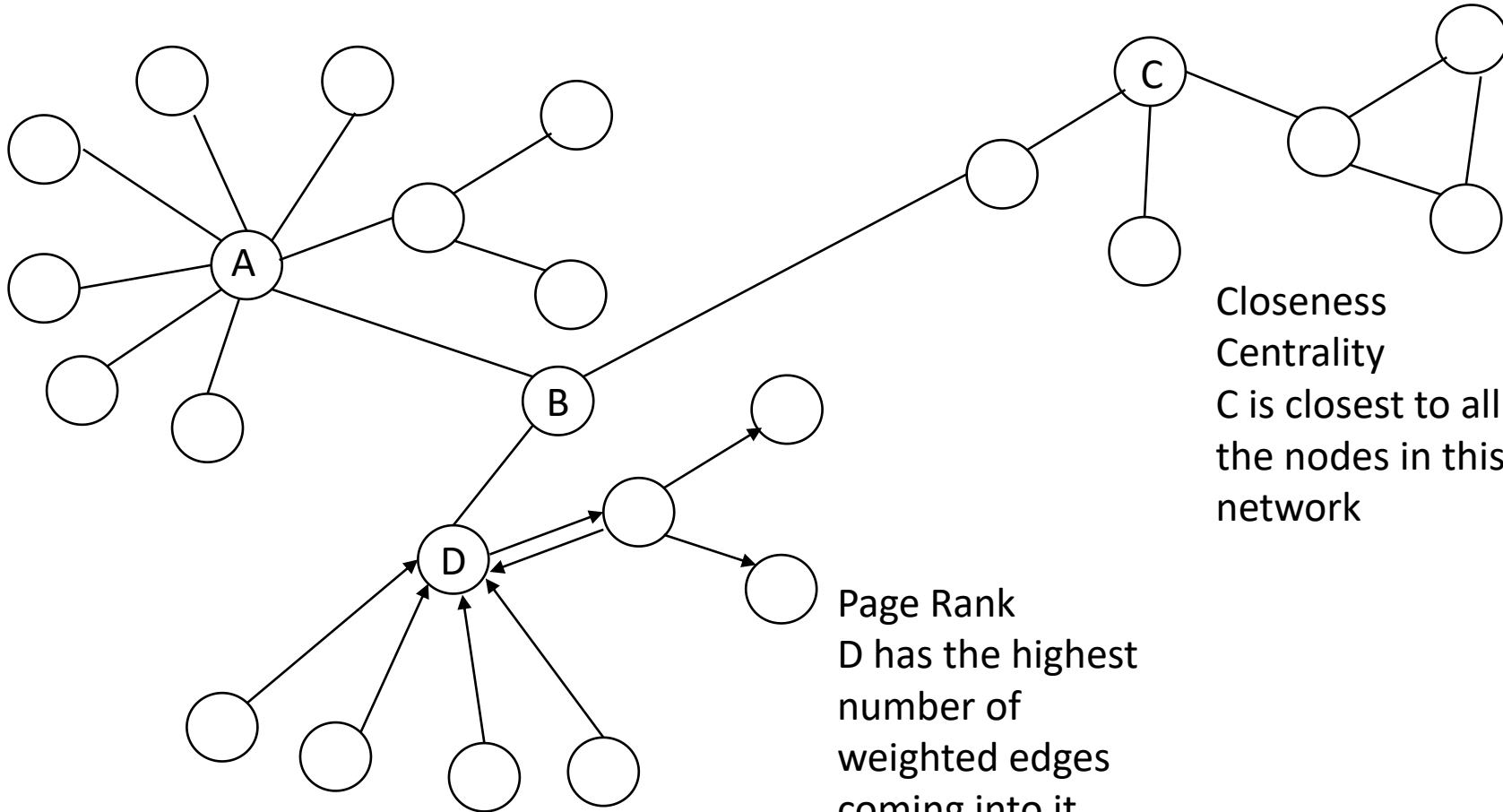


Centrality Detection

- Helps in understanding the importance of a node in a network

Degree Centrality
A has the highest degree

Betweenness Centrality
B has the largest number of shortest paths passing through it



Closeness Centrality
C is closest to all the nodes in this network

Page Rank
D has the highest number of weighted edges coming into it



Page Rank

- Measure the transitive influence of nodes
 - A node connected to a few influential nodes could be more important than a node connected to lots of unimportant nodes

$$PR(u) = (1 - d) + d * \left(\frac{PR(T1)}{C(T1)} + \dots + \frac{PR(Tn)}{C(Tn)} \right)$$

d – damping factor is usually set to 0.85

Set the values of PR for all nodes to same value, and iteratively improve it

Graph-based Inference Algorithms

- Path finding
- Centrality Detection
- Community Detection



Community Detection

- Identify nodes in the graph that are closely connected to each other
 - More relationships between nodes within the community
- Could be the first step in understanding a graph
 - More in-depth analysis of nodes within the community



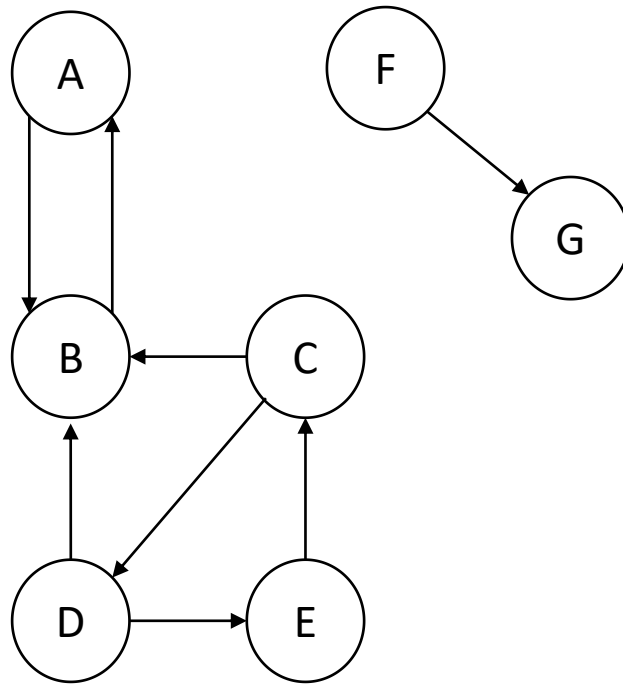
Community Detection

- Two families of algorithms
 - Standard graph algorithms
 - Bottom up algorithms



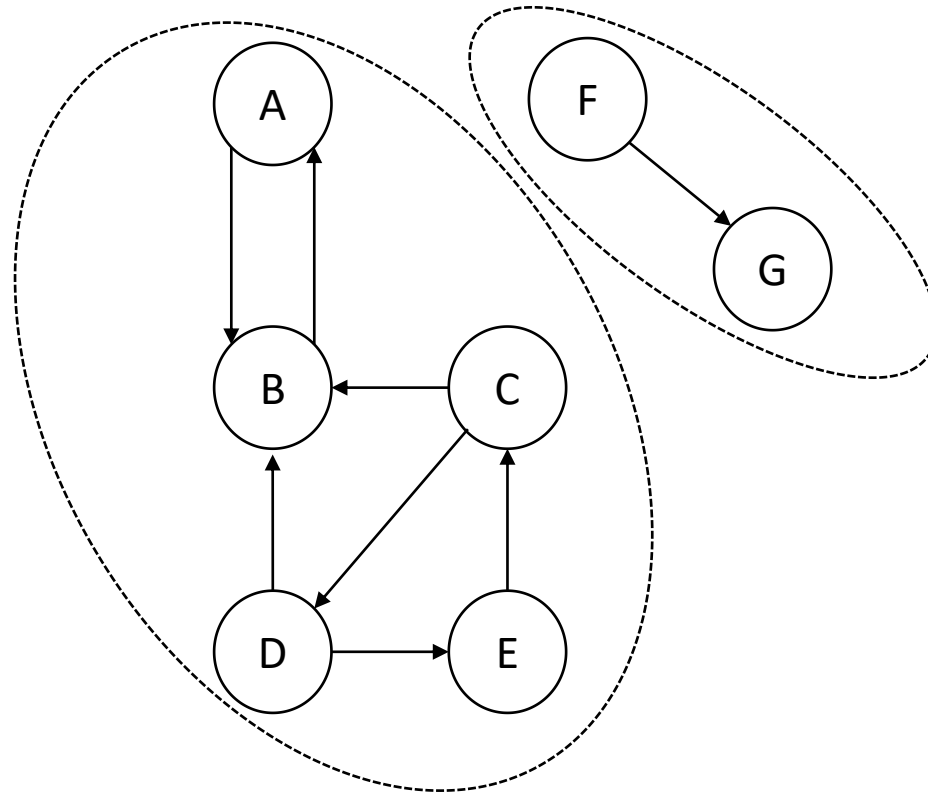
Community Detection Algorithms

- Standard Graph Algorithms



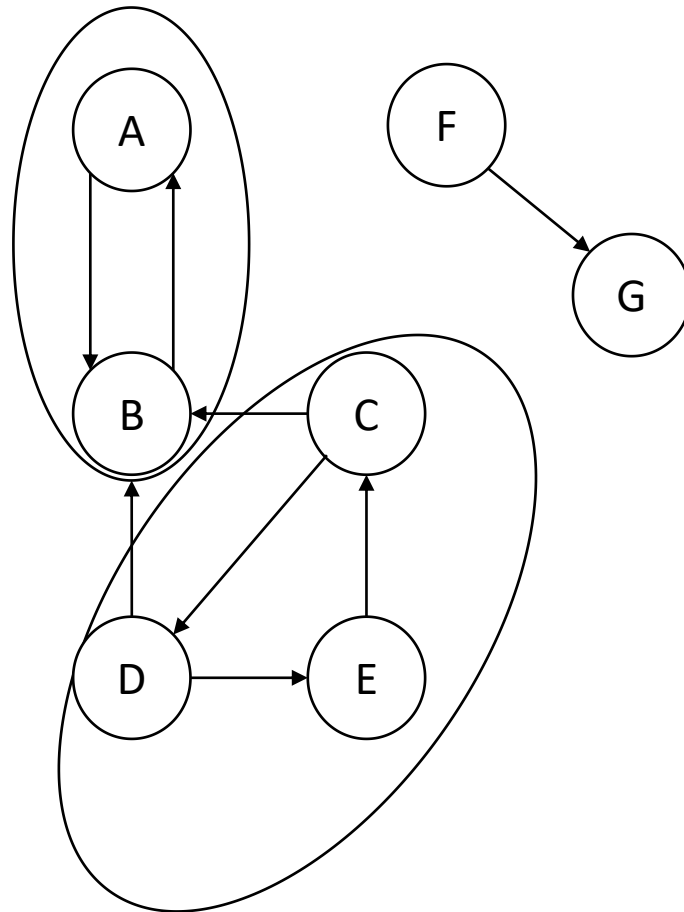
Community Detection Algorithms

- Standard Graph Algorithms – Connected components



Community Detection Algorithms

- Standard Graph Algorithms – Strongly connected components



Community Detection Algorithms

- Bottom up algorithms
 - Label propagation
 - Unfolding



Community Detection Algorithms

- Label Propagation
 - Assign each node to be in a different community
 - Examine all nodes in a fixed order
 - Update the community of a node that is shared by most of its neighbors
 - Break Ties in a random order
 - Terminate when each node is in a community shared by most of its neighbors



Community Detection Algorithms

- Unfolding (Also known as Louvain)
 - Phase I
 - Assign each node into a separate community
 - Examine each node and its neighbors to test if there will be an overall gain in modularity by placing it in the same community as a neighbor
 - Modularity calculated by a formula
 - Phase II
 - Create a new graph in which each node represents a community from Phase I
 - If there are edges between nodes in a community, represent it as a self-loop
 - Repeat Phase I on the Phase II graph



Outline

- Knowledge Graph Inference
 - Graph-based inference algorithms
 - **Ontology-based inference algorithms**



Ontology-based Inference

- Key differentiator between a general graph and a knowledge graph
 - Associates classes with nodes
 - Defines semantic properties of relationships
 - Two major categories of inference
 - Class-based Inference or Taxonomic Reasoning
 - Rule-based inference



Taxonomic Reasoning

- Applicable when it is useful to organize knowledge into classes
 - Classes are nothing but unary relations or types
 - Membership, specialization, disjointness, value restriction
- Both Property graph and RDF data models support classes
 - In property graphs model, node type is the same as a class
 - RDF has an RDF schema layer
 - More advanced systems use a full-fledged ontology language OWL

Our discussion here will be independent of either of the two models



Example of classes

male(art)
male(bob)
male(cal)
male(cam)

female(bea)
female(coe)
female(cory)

class(male)
instance_of(art,male)
instance_of(bob,male)
instance_of(cal,male)
instance_of(cam,male)

class(female)
instance_of(bea,female)
instance_of(coe,female)
instance_of(cory,female)



Class specialization

- We can organize classes into a hierarchy
 - e. g., male and female are subclasses of person
 - subclass-of(female, person)
 - subclass-of(male, person)
 - subclass relationship is transitive
 - subclass_of(A,C) :- subclass_of(A,B) & subclass_of(B,C)
 - subclass and instance-of relationships are related
 - instance_of(I,B) :- subclass_of(A,B) & instance_of(I,A)



Disjoint classes

- Classes can be declared to be disjoint
 - e.g, `disjoint(male,female)`
 - i.e., they do not have any instances in common
- illegal :- `disjoint(A,B) & instance_of(I,A) & instance_of(I,B)`

or

illegal(“Disjoint classes cannot have an instance in common”) :-
`disjoint(A,B) & instance_of(I,A) & instance_of(I,B)`



Class Definition

- Necessary properties of a class
 - Will have instance-of in the body of the rule
has_hair_color(X,brown) :-
 instance_of(X,brown_haired_person)
- Sufficient properties of a class
 - Will have instance-of in the head of the rule
instance_of(X,brown_haired_person) :-
 instance_of(X,person) &
 has_hair_color(X,brown)



Value Restriction

- We can restrict the arguments of a relation to be instances of a specific class
 - domain is the restriction on the first argument
illegal :- domain(parent, person) &
parent(X, Y) &
~instance_of(X, person)
 - range is the restriction on the second argument
illegal :- range(parent, person) &
parent(X, Y) &
~instance_of(Y, person)



Cardinality and Number Restrictions

- We can further restrict the values of relations by specifying cardinality and number restrictions
 - A cardinality restriction limits the number of values of a relation
illegal :- instance_of(X,person) & ~countofall(P,parent(P,X),2)
 - A numeric range restriction limits the minimum and maximum value
illegal :- instance_of(X,person) & age(X,Y) & min(0,Y,Y)
illegal :- instance_of(X,person) & age(X,Y)& min(125,Y,100)



Inheritance

- The relation values are said to inherit to the instances of a class
 - If art is an instance of the class brown-haired-person, we can conclude that art has brown hair



Taxonomic Inference

- Given two classes A and B, whether A is a subclass of B?
- Given a class A and an instance I, whether I is an instance of A?
- Given a ground relation atom determine whether it is true or false?
- Given a relation atom, determine values which values make it true?



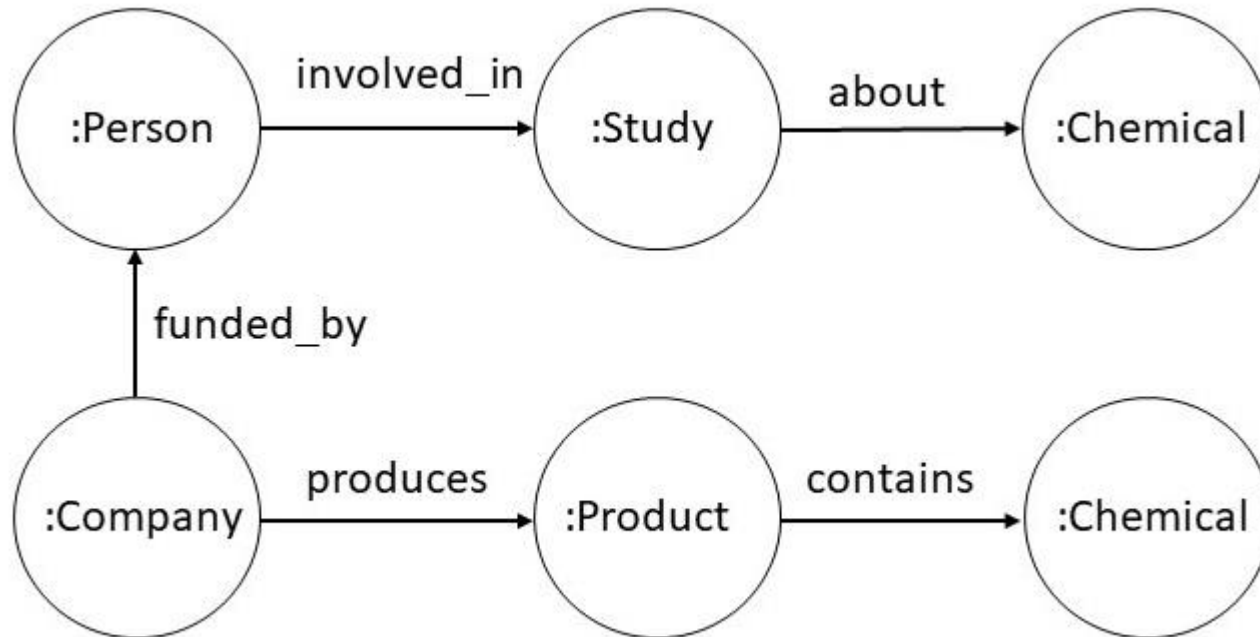
Rule-based Inference

- Boundary between taxonomic inference and rule-based inference is not sharp
 - It is generally a matter of the implementation approach
 - Taxonomic inferences can be usually implemented using rules



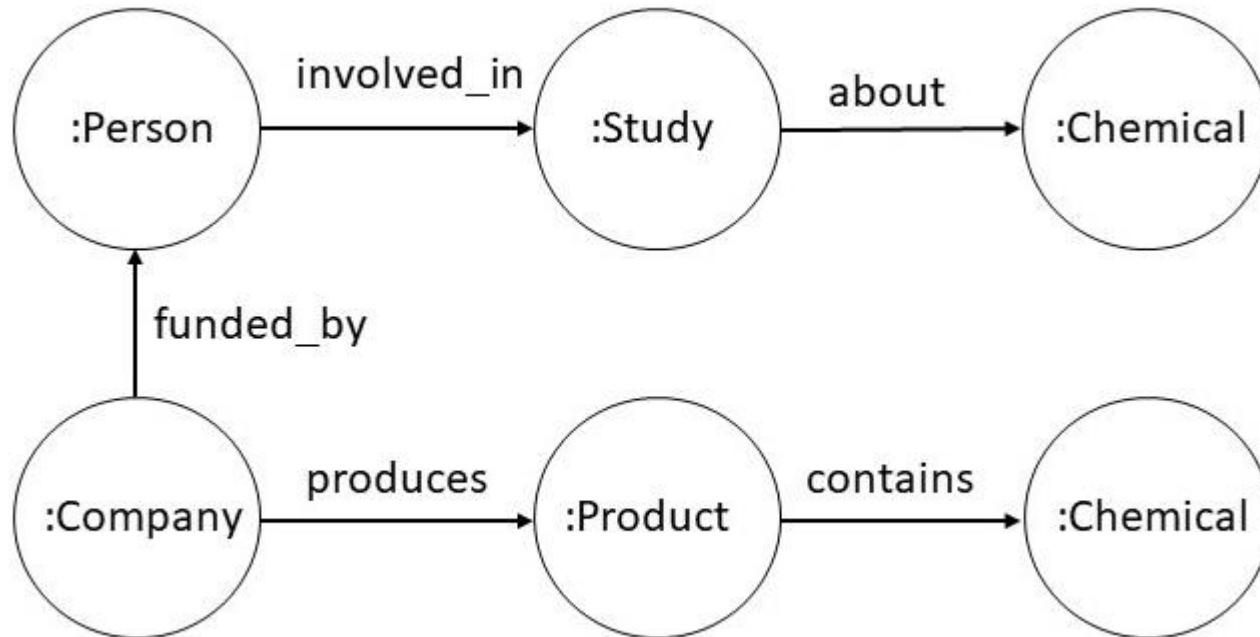
Rule-based inference

- Example Scenario



Rule-based inference

- Example Scenario



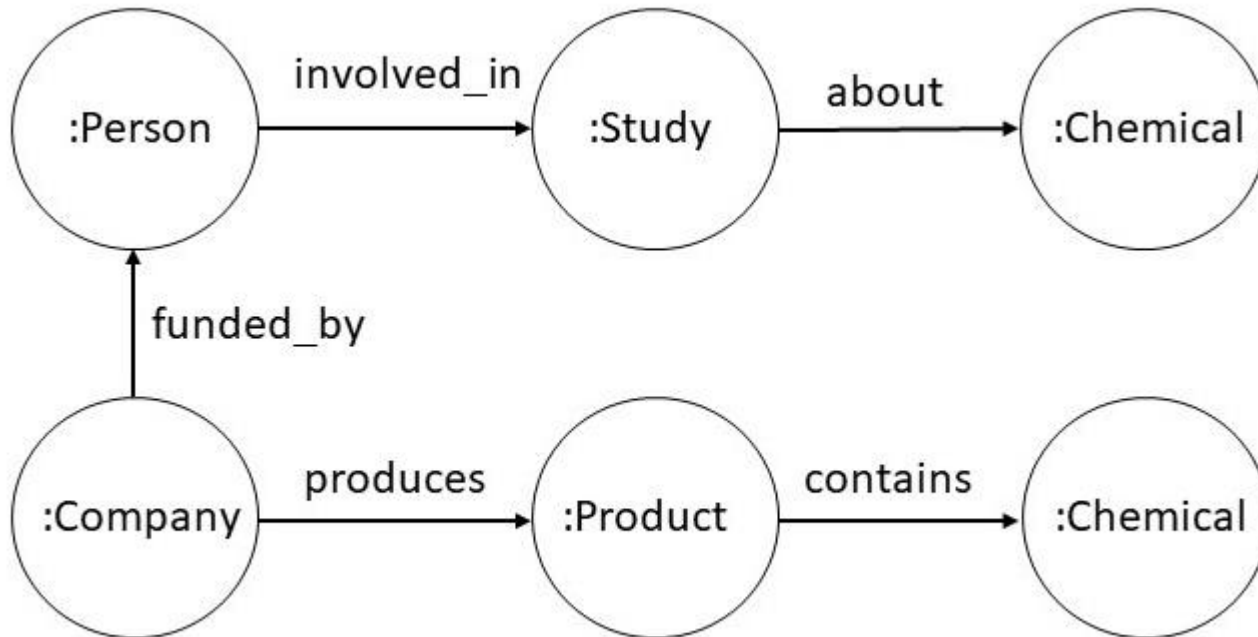
$\text{has_interest}(X,Z) \text{ :- produces}(X,Y) \ \& \ \text{contains}(Y,Z)$

$\text{coi}(X,Y,Z) \text{ :-}$
 $\text{involved_in}(X,Y) \ \& \ \text{about}(Y,P) \ \&$
 $\text{funded_by}(X,Z) \ \& \ \text{has_interest}(Z,P)$



Rule-based inference

- Example Scenario



$\text{has_interest}(X,Z) \text{ :- produces}(X,Y) \ \& \ \text{contains}(Y,Z)$

$\text{coi}(X,Y,Z) \text{ :-}$
 $\text{involved_in}(X,Y) \ \& \ \text{about}(Y,P) \ \&$
 $\text{funded_by}(X,Z) \ \& \ \text{has_interest}(Z,P)$

$\exists c \text{ conflict_of}(c,X) \ \& \ \text{conflict_reason}(c,Y) \ \&$
 $\text{conflict_with}(c,Z) \text{ :-}$
 $\text{involved_in}(X,Y) \ \& \ \text{about}(Y,P) \ \&$
 $\text{funded_by}(X,Z) \ \& \ \text{has_interest}(Z,P)$



Rule-based Inference

- Approaches
 - Bottom up strategy (also known as Chase)
 - Apply the rules against the data, and add new facts
 - Ensure termination
 - Process queries as usual
 - Top-down strategy
 - Start from the query, and apply rules as needed
 - Requires tighter integration between the rule engine and query evaluation
 - Requires lot less space

Many efficient and scalable rule engines available today



Summary

- Graph inference algorithms fall into two broad categories
 - Traditional Graph Algorithms
 - Path finding, centrality, community detection
 - Ontology-based algorithms
 - Taxonomic reasoning, Rule-based reasoning



Stanford·CS520 | Knowledge Graphs (2021)

CS520(2021)·课程资料包 @ShowMeAI



视频

中英双语字幕



课件

一键打包下载



笔记

官方笔记翻译



代码

作业项目解析



视频·B站 [扫码或点击链接]

<https://www.bilibili.com/video/BV1hb4y1r7fE>



课件 & 代码·博客 [扫码或点击链接]

<http://blog.showmeai.tech/cs520>

斯坦福

实体关系

图谱应用

图谱构建

图谱 schema

实体

非结构化数据

知识图谱

知识推理

Awesome AI Courses Notes Cheatsheets 是 [ShowMeAI](#) 资料库的分支系列，覆盖最具知名度的 **TOP20+** 门 AI 课程，旨在为读者和学习者提供一整套高品质中文学习笔记和速查表。

点击课程名称，跳转至课程**资料包**页面，**一键下载**课程全部资料！

| 机器学习 | 深度学习 | 自然语言处理 | 计算机视觉 |
|--|-------------------|--------------------|-------------------|
| Stanford · CS229 | Stanford · CS230 | Stanford · CS224n | Stanford · CS231n |
| # Awesome AI Courses Notes Cheatsheets·持续更新中 | | | |
| 知识图谱 | 图机器学习 | 深度强化学习 | 自动驾驶 |
| Stanford · CS520 | Stanford · CS224W | UCBerkeley · CS285 | MIT · 6.S094 |



微信公众号

资料下载方式 2: 扫码点击**底部菜单栏**

称为 **AI 内容创作者**? 回复 [添砖加瓦]