

# KNOWLEDGE GRAPHS

## Lecture 1: Introduction

Markus Krötzsch

Knowledge-Based Systems

TU Dresden, 15th Oct 2024

More recent versions of this slide deck might be available.  
For the most current version of this course, see  
[https://iccl.inf.tu-dresden.de/web/Knowledge\\_Graphs/en](https://iccl.inf.tu-dresden.de/web/Knowledge_Graphs/en)

# Introduction and Organisation

# Course Tutors



Markus Krötzsch  
Lectures



Maximilian Marx  
Exercises

# Organisation

- **Lectures**

Time and date yet to be determined (watch the web page)

Videos are available online:

<https://youtube.com/channel/UCCvDWNsR8YlQrB1tSj9Xqsw>

(see playlist “Knowledge Graphs”)

- **Exercise Sessions**

Tuesday, DS 5 (14:50–16:20), BAR 0218, **might also still change**

- **Web Page**

[https://iccl.inf.tu-dresden.de/web/Knowledge\\_Graphs\\_\(WS2024/25\)](https://iccl.inf.tu-dresden.de/web/Knowledge_Graphs_(WS2024/25))

- **Lecture Notes**

Slides of current and past lectures will be online.

- **Modules:** INF-B-510, INF-B-520, INF-BAS2, INF-VERT2, INF-BAS6, INF-VERT6, INF-E-3, INF-PM-FOR, MCL-KR, MCL-TC SL, CMS-COR-KM

# How exercises work

Exercise classes are **interactive** events:

- Tasks are published online
- Students solve the tasks before the date of the class
- In the meeting, students ask questions and present their solutions
- The tutor answers questions and helps with problems

# How exercises work

Exercise classes are **interactive** events:

- Tasks are published online
- Students solve the tasks before the date of the class
- In the meeting, students ask questions and present their solutions
- The tutor answers questions and helps with problems

**Corollary:** You must do your homework to benefit from the classes.

# How the examination works

The details of the examination depend on your module:

- Most modules: oral examination, sometimes in combination with other subjects
- For students of Computational Modelling and Simulation in module CMS-LM-COR:
  - written (60min) if more than 10 examinees
  - oral (20min) otherwise

All examinations are “closed book” (no auxiliary materials allowed)

# How to be prepared for examinations

Here are some hints that can help you to succeed in the examination:

- Follow the course closely during the semester
- Do all your homework
- When watching videos or attending classes: take hand-written notes
- Be prepared to reproduce most of the material that is on the slides when asked
- Be prepared to solve tasks that are like those in the exercises
- Expect that you will have to write answers on paper (also during oral exams)



# How to be prepared for examinations

Here are some hints that can help you to succeed in the examination:

- Follow the course closely during the semester
- Do all your homework
- When watching videos or attending classes: take hand-written notes
- Be prepared to reproduce most of the material that is on the slides when asked
- Be prepared to solve tasks that are like those in the exercises
- Expect that you will have to write answers on paper (also during oral exams)

**Main conclusion:** You need to be able to produce knowledge and solutions *actively* rather than merely understanding them *passively*.

# Goals and Prerequisites

## Goals

- Introduce basic notions of **graph-based knowledge representation(s)**
- Study important **graph data management approaches** (RDF, Property Graph) and **query languages**
- Learn about relevant **methods, tools, and datasets**
- Discuss aspects of **modelling and quality assurance**
- Get to know some **methods for analysing networks and graphs**

## (Non-)Prerequisites

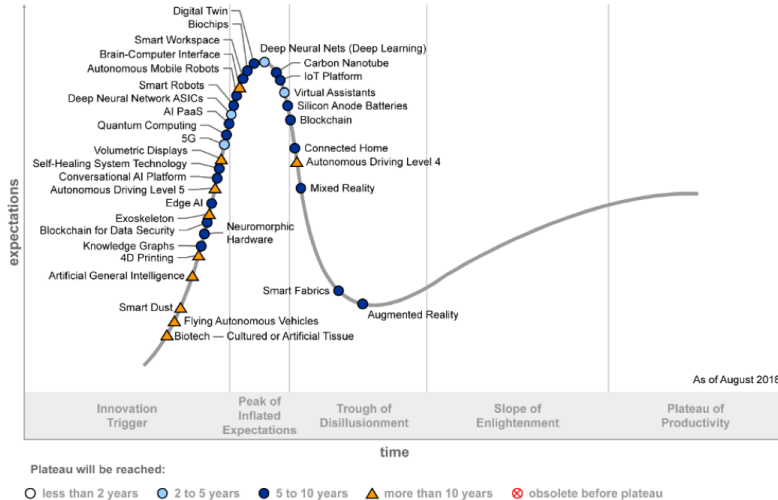
- No particular prior courses needed
- Basic programming skills are assumed; practical experience beyond basic courses will be helpful
- Interesting optional synergies: databases, machine learning, social networks, graph theory

# Course Outline

- **Resource Description Framework (RDF)**  
Underlying graph model; URIs; syntax
- **SPARQL**  
Query features; syntax and semantics; expressive power and complexity
- **Property graph**  
Underlying graph model; syntax and semantics of Cypher
- **Wikidata**  
Data model; applications; aspects of modelling; query answering
- **Ontologies and rules**  
Datalog; negation; existential rules; ontological models; OWL
- **RDF constraint languages**  
SHACL & ShEX; syntax and semantics; complexity and implementation
- **Network analysis**  
Centrality measures, PageRank, community detection

# Motivation

# The Hype



(c) 2018 Gartner, Inc. All rights reserved.

# Knowledge Graphs Everywhere



EDITOR: EU

CENTRAL EUROPE MIDDLE EAST SCANDINAVIA AFRICA UK ITALY SPAIN MORE

MUST READ: Australian encryption-busting Bill would create backdoors Cisco

## Knowledge graphs beyond the hype: Getting knowledge in and out of graphs and databases

What exactly are knowledge graphs, and what's with all the hype about them? Learning to tell apart hype from reality, defining different types of graphs, and picking the right tools and database for you want to be like the Airbnbs, Amazons, Googles, and LinkedIns of the world.

TechRepublic

BIG DATA

## Amazon Neptune is here: 6 ways customers use the AWS graph database

Customers including Samsung, Intuit, and Pearson previewed the database, building new graph applications and testing production workloads.

By Alison DeNisco Rayome | May 31, 2018, 7:44 AM PST

TC

- Startups
- Apps
- Gadgets
- Events
- Videos
- 
- Crunchbase
- More

acquire Lattice Data over the weekend. The startup was working to transform the way businesses deal with paragraphs of text and other information that lives outside neatly structured databases. These engineers are uniquely prepared to assist Apple with building a next-generation internal knowledge graph to power Siri and its next generation of intelligent products and services.

Broadly speaking, the Lattice Data deal was an acquihire. Apple paid roughly \$10 million for each of Lattice's 20 engineers. This is generally considered to be fair market value. Google paid

Forbes

Billionaires Innovation Leadership Money Consumer Industry

## Is The Enterprise Knowledge Graph Finally Going To Make All Data Usable?



All company logos subject to copyrights. All rights reserved.

# Knowledge Graphs in the Age of Large Language Models?

# Knowledge Graphs in the Age of Large Language Models?

Let's ask ChatGPT (version "mini4o")!

Where was Markus Krötzsch born?



# Knowledge Graphs in the Age of Large Language Models?

Let's ask ChatGPT (version "mini4o")!

Where was Markus Krötzsch born?



Markus Krötzsch was born in Germany. If you need more specific information about his background or contributions, feel free to ask!

# Knowledge Graphs in the Age of Large Language Models?

Let's ask ChatGPT (version "mini4o")!

Where was Markus Krötzsch born?



Markus Krötzsch was born in Germany. If you need more specific information about his background or contributions, feel free to ask!

Can you also tell me the city that Markus Krötzsch has been born in?

# Knowledge Graphs in the Age of Large Language Models?

Let's ask ChatGPT (version "mini4o")!

Where was Markus Krötzsch born?



Markus Krötzsch was born in Germany. If you need more specific information about his background or contributions, feel free to ask!

Can you also tell me the city that Markus Krötzsch has been born in?



Markus Krötzsch was born in Leipzig, Germany. If you have any more questions about him or his work, let me know!

# Knowledge Graphs in the Age of Large Language Models?

Let's ask ChatGPT (version "mini4o")!

Where was Markus Krötzsch born?



Markus Krötzsch was born in Germany. If you need more specific information about his background or contributions, feel free to ask!

Can you also tell me the city that Markus Krötzsch has been born in?



Markus Krötzsch was born in Leipzig, Germany. If you have any more questions about him or his work, let me know!

Can you give me a source for this fact?

# Knowledge Graphs in the Age of Large Language Models?

Let's ask ChatGPT (version "mini4o")!

Where was Markus Krötzsch born?



Markus Krötzsch was born in Germany. If you need more specific information about his background or contributions, feel free to ask!

Can you also tell me the city that Markus Krötzsch has been born in?



Markus Krötzsch was born in Leipzig, Germany. If you have any more questions about him or his work, let me know!


Can you give me a source for this fact?



I don't have direct access to external sources to provide citations, but you can find information about Markus Krötzsch and his background on academic websites, his personal or professional pages, or in interviews. Searching for him on platforms like Google Scholar or his university's website may yield the details you need. If you want, I can help guide you on how to

# Knowledge Graphs in the Age of Large Language Models?

Ok, maybe let's ask ChatGPT proper (version "4o") ...



Where was Markus Krötzsch born?

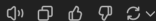
# Knowledge Graphs in the Age of Large Language Models?

Ok, maybe let's ask ChatGPT proper (version "4o") ...

Where was Markus Krötzsch born?



Markus Krötzsch was born in **Dresden**, Germany. He is a German computer scientist known for his contributions to knowledge representation, semantic technologies, and his work on the development of Wikidata.



# Knowledge Graphs in the Age of Large Language Models?

Ok, maybe let's ask ChatGPT proper (version "4o") ...

Where was Markus Krötzsch born?



Markus Krötzsch was born in **Dresden**, Germany. He is a German computer scientist known for his contributions to knowledge representation, semantic technologies, and his work on the development of Wikidata.



Can you give me a source for this fact?



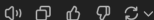
# Knowledge Graphs in the Age of Large Language Models?

Ok, maybe let's ask ChatGPT proper (version "4o") ...

Where was Markus Krötzsch born?



Markus Krötzsch was born in **Dresden**, Germany. He is a German computer scientist known for his contributions to knowledge representation, semantic technologies, and his work on the development of Wikidata.



Can you give me a source for this fact?



Searched 6 sites ▾

Markus Krötzsch was born in Dresden, Germany. He is a professor at the Technical University of Dresden, where he leads the chair for Knowledge-Based Systems. His research has significantly impacted fields like semantic technologies and ontology languages. Krötzsch also played a key role in the development of Wikidata and the Semantic MediaWiki COMPLIC CENTER

NEWS



# Knowledge Graphs in the Age of Large Language Models?

Ok, let's check Google.

The image shows a Google search interface with a dark theme. The search bar contains the text "Where was Markus Kröttsch born?". Below the search bar, there are navigation tabs for "All", "Images", "News", "Videos", "Web", "Books", "Finance", and "Tools". The "All" tab is selected. The search results are displayed in a list format. The first result is from Wikipedia, showing a profile card for Markus Kröttsch with a photo and a brief biography. The second result is from TU Dresden, showing a news article titled "Markus Kröttsch became the new Professor for ..." dated 1 Aug 2016. The third result is from Wikidata, showing a profile card for Markus Kröttsch with a brief biography and birth date.

Google

Where was Markus Kröttsch born?

All Images News Videos Web Books Finance Tools

Wikipedia  
[https://de.wikipedia.org/wiki/Markus\\_Kröttsch](https://de.wikipedia.org/wiki/Markus_Kröttsch) · Translate this page

**Markus Kröttsch**

Markus Kröttsch (\* 6. Mai 1981 in **Rodewisch**) ist ein deutscher Informatiker. Markus Kröttsch 2014. Kröttsch studierte nach dem Abitur in Auerbach im ...  
Missing: **born?** | Show results with: **born?**

TU Dresden  
<https://tu-dresden.de/informatik/die-fakultaet/news>

**Markus Kröttsch became the new Professor for " ...**

1 Aug 2016 — **He was born in Vogtland**, had studied Computer Science in Dresden and did his PhD at the Institute of Technology in Karlsruhe (KIT) about ...

Wikidata ID  
[https://www.wikidata.org/wiki/Markus\\_Kröttsch](https://www.wikidata.org/wiki/Markus_Kröttsch)

**Markus Kröttsch**

Markus Kröttsch. German computer scientist. Markus Kroetzsch. In more languages ... date of birth. **6 May 1981**. 1 reference. reference URL · <http://www.dfg.de> ...

# The Knowledge Graph at Google

More popular entities get richer information displays, fed directly from Google's KG:

The screenshot shows a Google search interface with the query "Where was Olaf Scholz born?". The search results include a knowledge panel for "Osnabrück, Germany". The panel provides biographical information about Olaf Scholz, a Wikipedia link, and a list of related locations. On the right, there is a detailed information box for Osnabrück, including its population, area, elevation, district, and postal codes.

Google

Where was Olaf Scholz born?

All Images Maps News Videos Web Books More Tools

Olaf Scholz / Place of birth

## Osnabrück, Germany

Scholz was born on 14 June 1958, in Osnabrück, Lower Saxony, and grew up in Hamburg's Rahlstedt district. His parents worked in the textile industry.

Wikipedia  
[https://en.wikipedia.org/wiki/Olaf\\_Scholz](https://en.wikipedia.org/wiki/Olaf_Scholz)

**Olaf Scholz - Wikipedia**

People also search for

- Oldenburg
- Osnabrück
- Münster
- Bielefeld
- Bremen
- Hanover
- Ibbenbüchel

[Feedback](#)

### Osnabrück

City in Germany

Osnabrück is a city in northwest Germany. The Town Hall is where the 1648 Peace of Westphalia was negotiated, bringing the 30 Years' War to an end. It sits on the market square, along with gabled houses and St. Marien, a 13th-century Gothic church. The Felix Nussbaum House shows a large collection of works by the local surrealist painter. To the south, the grounds of Osnabrück Castle are a venue for summer concerts. — Google

**Population:** 164,748 (2019) Eurostat

**Area:** 119.8 km<sup>2</sup>

**Elevation:** 64 m

**District:** Urban district

**Postal codes:** 49074–49090

[More about Osnabrück](#)

[Feedback](#)

# What is a Knowledge Graph?

# What is a Knowledge Graph?

The original “Knowledge Graph” (Google, 2012):

The image shows a screenshot of the Google Inside Search Knowledge Graph interface. At the top, the "Google Inside Search" logo is displayed. Below it is a navigation bar with links for Home, How Search Works, Tips & Tricks, Features, Search Stories, Playground, Blog, and Help. The main content area features a dark background with a network of interconnected nodes and lines, representing the Knowledge Graph. A large blue circular node in the center contains a portrait of Leonardo da Vinci. Other nodes include a red maple leaf, a green globe, and various smaller images. A blue button with a white right-pointing arrow is positioned to the left of the text "The Knowledge Graph". Below this text, it says "Learn more about one of the key breakthroughs behind the future of search." To the right, a grey button with a white right-pointing arrow is positioned to the left of the text "See it in action". Below this text, it says "Discover answers to questions you never thought to ask, and explore collections and lists." On the right side of the interface, there is a search results panel for "Leonardo da Vinci". It includes a list of three images: "Ginevra dei Benci 1478", "The Virgin &... 1508", and "Adoration of the M... 1481". Below the images, there is a "Feedback" link. The main text for "Leonardo da Vinci" reads: "Leonardo di ser Piero da Vinci was an Italian Renaissance polymath, painter, sculptor, architect, musician, scientist, mathematician, engineer, inventor, anatomist, geologist, cartographer, botanist, and writer. [View source](#)". Below this, it lists: "Born: April 15, 1452, Anchiano", "Died: May 2, 1519, Clus Lunz", "Buried: Chiusa d'Arbola", "Parents: Caterina da Vinci, Piero da Vinci", and "Structures: Volpiano Band Da Vinci Project".

(c) Google. All rights reserved.

# Many knowledge graphs, many technologies

There are a number of widely used publicly available knowledge graphs:



... and a variety of technologies for working with them:



# So what is a Knowledge Graph?

A first attempt at a definition:

**A Knowledge Graph is a knowledge base that is a graph.**

# So what is a Knowledge Graph?

A first attempt at a definition:

**A Knowledge Graph is a knowledge base that is a graph.**

**So what is a knowledge base?**



# So what is a Knowledge Graph?

A first attempt at a definition:

**A Knowledge Graph is a knowledge base that is a graph.**

## So what is a knowledge base?

- “A knowledge base is a technology used to store complex **structured and unstructured information** used by a computer system. [...] [It] represents **facts about the world**” – Wikipedia (26 Oct 2020, id 983269427)
- “A **collection of knowledge** expressed using some **formal knowledge representation language**.” – Free Online Dictionary of Computing, 15 Oct 2018
- 1. a **store of information or data** that is available to draw on.
  2. the underlying **set of facts, assumptions, and rules** which a computer system has available to solve a problem.– Lexico (Oxford University Press/Dictionary.com), 26 Oct 2020

# So what is a Knowledge Graph?

A first attempt at a definition:

**A Knowledge Graph is a knowledge base that is a graph.**

**So what is a graph?**

# So what is a Knowledge Graph?

A first attempt at a definition:

**A Knowledge Graph is a knowledge base that is a graph.**

## So what is a graph?

- “a **collection of points and lines** connecting some (possibly empty) subset of them” – Wolfram MathWorld, 26 Oct 2020
- “a **collection of vertices and edges** that join pairs of vertices” – Merriam-Webster, 26 Oct 2020
- “a structure amounting to a **set of objects** in which some pairs of the objects are **in some sense ‘related’**.” – Wikipedia (26 Oct 2020, id 984093316)

(we’ll have more to say about mathematical graphs later)

# So what is a Knowledge Graph?

A first attempt at a definition:

**A Knowledge Graph is a knowledge base that is a graph.**

**In summary:**

# So what is a Knowledge Graph?

A first attempt at a definition:

**A Knowledge Graph is a knowledge base that is a graph.**

**In summary:**

- a collection of facts, rules, or other forms of knowledge
- that express some kind of relationships or connections

↪ a paradigm rather than a specific class of things

# What is special about Knowledge Graphs?

A second attempt at a definition:

## **A Knowledge Graph is a data set that is:**

- **structured** (in the form of a specific data structure)
- **normalised** (consisting of small units, such as vertices and edges)
- **connected** (defined by the – possibly distant – connections between objects)

Moreover, knowledge graphs are typically:

- **explicit** (created purposefully with an intended meaning)
- **declarative** (meaningful in itself, independent of a particular implementation or algorithm)
- **annotated** (enriched with contextual information to record additional details and meta-data)
- **non-hierarchical** (more than just a tree-structure)
- **large** (millions rather than hundreds of elements)

# (Counter-)Examples

**Typical knowledge graphs:**

# (Counter-)Examples

## Typical knowledge graphs:

- Wikidata, Yago, Freebase, DBpedia (though hardly annotated)
- OpenStreetMap
- Google Knowledge Graph, Microsoft Bing Satori (presumably; we can't really know)



# (Counter-)Examples

## Typical knowledge graphs:

- Wikidata, Yago, Freebase, DBpedia (though hardly annotated)
- OpenStreetMap
- Google Knowledge Graph, Microsoft Bing Satori (presumably; we can't really know)

## Debatable cases:

# (Counter-)Examples

## Typical knowledge graphs:

- Wikidata, Yago, Freebase, DBpedia (though hardly annotated)
- OpenStreetMap
- Google Knowledge Graph, Microsoft Bing Satori (presumably; we can't really know)

## Debatable cases:

- Facebook's social graph: structured, normalised, connected, but not explicit (emerging from user interactions, without intended meaning beyond local relations)

# (Counter-)Examples

## Typical knowledge graphs:

- Wikidata, Yago, Freebase, DBpedia (though hardly annotated)
- OpenStreetMap
- Google Knowledge Graph, Microsoft Bing Satori (presumably; we can't really know)

## Debatable cases:

- Facebook's social graph: structured, normalised, connected, but not explicit (emerging from user interactions, without intended meaning beyond local relations)
- WordNet: structured dictionary and thesaurus, but with important unstructured content (descriptions); explicit, declarative model

# (Counter-)Examples

## Typical knowledge graphs:

- Wikidata, Yago, Freebase, DBpedia (though hardly annotated)
- OpenStreetMap
- Google Knowledge Graph, Microsoft Bing Satori (presumably; we can't really know)

## Debatable cases:

- Facebook's social graph: structured, normalised, connected, but not explicit (emerging from user interactions, without intended meaning beyond local relations)
- WordNet: structured dictionary and thesaurus, but with important unstructured content (descriptions); explicit, declarative model
- Global data from [schema.org](http://schema.org): maybe not very connected

# (Counter-)Examples

## Typical knowledge graphs:

- Wikidata, Yago, Freebase, DBpedia (though hardly annotated)
- OpenStreetMap
- Google Knowledge Graph, Microsoft Bing Satori (presumably; we can't really know)

## Debatable cases:

- Facebook's social graph: structured, normalised, connected, but not explicit (emerging from user interactions, without intended meaning beyond local relations)
- WordNet: structured dictionary and thesaurus, but with important unstructured content (descriptions); explicit, declarative model
- Global data from [schema.org](http://schema.org): maybe not very connected
- Document stores (Lucene, MongoDB, etc.): structured, but not normalised; connections sub-ordinary

# (Counter-)Examples

## Typical knowledge graphs:

- Wikidata, Yago, Freebase, DBpedia (though hardly annotated)
- OpenStreetMap
- Google Knowledge Graph, Microsoft Bing Satori (presumably; we can't really know)

## Debatable cases:

- Facebook's social graph: structured, normalised, connected, but not explicit (emerging from user interactions, without intended meaning beyond local relations)
- WordNet: structured dictionary and thesaurus, but with important unstructured content (descriptions); explicit, declarative model
- Global data from [schema.org](http://schema.org): maybe not very connected
- Document stores (Lucene, MongoDB, etc.): structured, but not normalised; connections sub-ordinary

## Primarily not knowledge graphs:

# (Counter-)Examples

## Typical knowledge graphs:

- Wikidata, Yago, Freebase, DBpedia (though hardly annotated)
- OpenStreetMap
- Google Knowledge Graph, Microsoft Bing Satori (presumably; we can't really know)

## Debatable cases:

- Facebook's social graph: structured, normalised, connected, but not explicit (emerging from user interactions, without intended meaning beyond local relations)
- WordNet: structured dictionary and thesaurus, but with important unstructured content (descriptions); explicit, declarative model
- Global data from [schema.org](http://schema.org): maybe not very connected
- Document stores (Lucene, MongoDB, etc.): structured, but not normalised; connections sub-ordinary

## Primarily not knowledge graphs:

- Wikipedia: mostly unstructured text; not normalised; connections (links) important but sub-ordinary (similar: [The Web](#))

# (Counter-)Examples

## Typical knowledge graphs:

- Wikidata, Yago, Freebase, DBpedia (though hardly annotated)
- OpenStreetMap
- Google Knowledge Graph, Microsoft Bing Satori (presumably; we can't really know)

## Debatable cases:

- Facebook's social graph: structured, normalised, connected, but not explicit (emerging from user interactions, without intended meaning beyond local relations)
- WordNet: structured dictionary and thesaurus, but with important unstructured content (descriptions); explicit, declarative model
- Global data from [schema.org](http://schema.org): maybe not very connected
- Document stores (Lucene, MongoDB, etc.): structured, but not normalised; connections sub-ordinary

## Primarily not knowledge graphs:

- Wikipedia: mostly unstructured text; not normalised; connections (links) important but sub-ordinary (similar: The Web)
- Relational database of some company: structured and possibly normalised, but no focus on connections (traditional RDBMS support connectivity queries only poorly)



# Graphs in Computer Science and Mathematics

# What is a graph?

**Definition 1.1:** A **simple undirected graph**  $G$  consists of a set  $V$  of **vertices** and a set  $E$  of **edges**, where each edge is a set of two vertices. Two vertices  $v_1, v_2 \in V$  are **adjacent** (in  $G$ ) if there is an edge  $\{v_1, v_2\} \in E$ .

Vertices are sometimes also called **nodes**; undirected edges are sometimes also called **arcs**.

Unless otherwise noted, we assume all graphs to be finite.

# What is a graph?

**Definition 1.1:** A **simple undirected graph**  $G$  consists of a set  $V$  of **vertices** and a set  $E$  of **edges**, where each edge is a set of two vertices. Two vertices  $v_1, v_2 \in V$  are **adjacent** (in  $G$ ) if there is an edge  $\{v_1, v_2\} \in E$ .

Vertices are sometimes also called **nodes**; undirected edges are sometimes also called **arcs**.

Unless otherwise noted, we assume all graphs to be finite.

Discrete mathematics considers a variety of other kinds of “graphs”:

- Directed or undirected
- Simple graph or multi-graph
- Possibly labelled edges or vertices
- Possibly with self-loops
- Possibly with higher arity edges (hypergraphs)

# Directed and other graphs

**Definition 1.2:** A simple directed graph (a.k.a. simple digraph)  $G$  consists of a set  $V$  of vertices and a set  $E \subseteq V \times V$  of (directed) edges from a source vertex to a target vertex.

Other terms are similar to undirected graphs; directed edges are also known as **arrows** and are often denoted as such, e.g.,  $v_1 \xrightarrow{e_1} v_2$ .

# Directed and other graphs

**Definition 1.2:** A **simple directed graph** (a.k.a. **simple digraph**)  $G$  consists of a set  $V$  of vertices and a set  $E \subseteq V \times V$  of (**directed**) **edges** from a **source vertex** to a **target vertex**.

Other terms are similar to undirected graphs; directed edges are also known as **arrows** and are often denoted as such, e.g.,  $v_1 \xrightarrow{e_1} v_2$ .

**Definition 1.3:** The following generalisations apply to directed and to undirected graphs.

- A **graph with self-loops** is a graph extended with the option of having edges that relate a vertex to itself.
- A **multi-graph** is a graph that may have multiple edges with the same vertices (in the same direction).
- An **edge-labelled graph** is a graph that has an additional labelling function  $\lambda : E \rightarrow L$  that maps each edge in  $E$  to an element from a set of labels  $L$  (similarly for vertex-labelled graphs).

## Other basic notions

**Definition 1.4:** An edge are said to be **incidental** to the vertices it connects. The **degree** of a vertex is the number of edges that are incidental to it. In a digraph, the **in-degree** of a vertex is the number of edges pointing towards it; analogously for **out-degree**.

## Other basic notions

**Definition 1.4:** An edge are said to be **incidental** to the vertices it connects. The **degree** of a vertex is the number of edges that are incidental to it. In a digraph, the **in-degree** of a vertex is the number of edges pointing towards it; analogously for **out-degree**.

**Definition 1.5:** A **directed path** in a digraph is a sequence of consecutive edges  $v_0 \xrightarrow{e_1} v_1 \xrightarrow{e_2} \dots \xrightarrow{e_n} v_n$ . An **undirected path** is a sequence of edges that may point either way (or that are simply undirected).

A **simple path** (directed or undirected) is a path without repeated vertices other than possibly the first and last node.

## Other basic notions

**Definition 1.4:** An edge are said to be **incidental** to the vertices it connects. The **degree** of a vertex is the number of edges that are incidental to it. In a digraph, the **in-degree** of a vertex is the number of edges pointing towards it; analogously for **out-degree**.

**Definition 1.5:** A **directed path** in a digraph is a sequence of consecutive edges  $v_0 \xrightarrow{e_1} v_1 \xrightarrow{e_2} \dots \xrightarrow{e_n} v_n$ . An **undirected path** is a sequence of edges that may point either way (or that are simply undirected).

A **simple path** (directed or undirected) is a path without repeated vertices other than possibly the first and last node.

**Definition 1.6:** Two vertices are **connected** if there is an undirected path from one to the other. A graph is connected if any pair of two distinct vertices is connected. A digraph is **strongly connected** if there is a directed path from any vertex to any other vertex (hence: one directed path in either direction).



# Representing graphs (1)

There are several obvious ways of representing graphs in computer science.

**Definition 1.7:** The **adjacency matrix** of a graph  $G = \langle V, E \rangle$  is the boolean  $|V| \times |V|$  matrix that contains, at any coordinate  $\langle v_1, v_2 \rangle$ , the value **1** if there is an edge connecting  $v_1$  and  $v_2$ .

## Notes:

- Adjacency matrices for undirected graphs are symmetric.
- Loops (if allowed) show up as **1** in the diagonal.
- The matrix could be adapted to multi-graphs by storing the numbers of edges.
- The matrix could be adapted to labelled simple graphs by storing the labels.

## Representing graphs (2)

There are several obvious ways of representing graphs in computer science.

**Definition 1.8:** The **adjacency list** of a graph  $G = \langle V, E \rangle$  is the list of all of its edges.

### Notes:

- We can write edges as pairs (order is irrelevant for undirected graphs).
- Loops (if allowed) show up as edges with repeated vertices.
- The list could be adapted to multi-graphs by adding the number of edges to each line, or by allowing repeated lines.
- The list could be adapted to labelled graphs by adding labels to each line (for multi-graph: repeat lines rather than also storing number).
- The list does not encode  $V$ : vertices without edges are missing (might be listed separately if relevant to application)

## Representing graphs (3)

There are also other options.

**Example 1.9:** We could also encode the adjacency matrix by giving, for each row, a list of all vertices whose column is set to **1**. This is equivalent to ordering edges by first vertex and combining them into a single line.

# Which graph representation to pick?

Each representation has its pros and cons:

- **Matrix:**
  - + space efficient for dense graphs (1 bit per edge);
  - + can be processed with matrix operations (highly parallel);
  - space inefficient for sparse graphs;
  - not natural for labelled multi-graphs
- **List:**
  - + space efficient for sparse graphs;
  - + easy to use for labelled multi-graphs;
  - harder to process (esp. if edge order can be random);
  - not space efficient for dense graphs

**Note:** Knowledge graphs are typically sparse and labelled, but parallel processing still makes matrices attractive in some applications.

# Summary

The course will be 50% lectures and 50% tutorial classes

Active participation is important to succeed

Knowledge Graphs are a data management concept of practical importance

Mathematics studies various types of graphs, which can be represented by several common data structures

## **What's next?**

- Encoding graphs in technical systems: the Resource Description Format RDF
- Formats for exchanging RDF
- Modelling knowledge in RDF graphs

Where was Olaf Scholz born?



Olaf Scholz was born in Hamburg, Germany.



(ChatGPT mini4o, 15 Oct 2024)