知识图谱在数据结构课程教学中的应用

董敏 [1,2] 李彩红 [1] 董诚义 [1]
1. 西安思源学院 电子信息工程学院，中国·陕西 西安 710038
2. 西安思源学院 智能信息处理研究中心，中国·陕西 西安 710038

摘 要：随着人工智能技术的飞速发展，知识图谱作为新兴的研究领域，正逐步渗透到教育教学的各个层面。论文聚焦于数据结构课程知识图谱的构建，旨在通过深入分析知识图谱的基本概念、构建方法论及其在数据结构教学中的独特应用，提出一系列创新策略，并结合具体实践案例，展示这些策略在提高学生学习效率、深化概念理解方面的显著成效。论文的研究不仅为数据结构课程的教学创新提供了有力支持，也为其他学科领域的知识图谱构建与应用树立了典范。

关键词：人工智能；知识图谱；数据结构；教学创新

The Application of Knowledge Graph in the Teaching of Data Structures Course

Min Dong[1,2] Caihong Li[1] Chengyi Dong[1]
1. School of Electronic Information Engineering, Xi'an Siyuan University, Xi'an, Shaanxi, 710038, China
2. Intelligent Information Processing Research Center, Xi'an Siyuan University, Xi'an, Shaanxi, 710038, China

Abstract: With the rapid development of artificial intelligence technology, knowledge graph, as an emerging research field, is gradually penetrating into various levels of education and teaching. This paper focuses on the construction of a knowledge graph for data structure courses, aiming to propose a series of innovative strategies through in-depth analysis of the basic concepts, construction methodology, and unique applications of knowledge graphs in data structure education. Combined with specific practical cases, it demonstrates the significant effectiveness of these strategies in improving students' learning efficiency and deepening conceptual understanding. This study not only provides strong support for the teaching innovation of data structure courses, but also sets an example for the construction and application of knowledge graphs in other disciplinary fields.

Keywords: artificial intelligence; knowledge graph; data structure; teaching innovation

0 前言

数据结构是本科计算机学科中的一门核心课程，主要研究数据的逻辑结构和物理结构以及它们之间的相互关系，并通过对这种结构的定义、数据运算和算法的设计来分析和解决问题。课程目标为学生能清晰理解数据结构与算法基础，掌握线性、树形、图形结构及其运算，能运用这些知识解决复杂问题。同时，学生需具备批判性思维，能针对具体问题选择合适的数据结构和设计高效算法。然而，数据结构中的概念，如链表、树、图等都较为抽象，传统的教学方式往往难以有效整合并直观呈现复杂的知识体系，学生理解和掌握存在困难，很难形成系统性的认知结构。知识图谱作为一种新型的数据结构，以图的形式表示实体及其之间的关系，为数据结构课程的教学提供了新的思路和方法。

1 知识图谱概述

知识图谱（Knowledge Graph）是一种用于表示知识之间关系的图形化数据结构，通过将知识点和概念之间的关系以图的形式呈现出来，不仅能够表示单个实体的信息，还能揭示实体之间的复杂关系，从而实现更智能的知识检索和推理。知识图谱由实体（Entity）、属性（Attribute）和关系（Relation）构成，其中实体是知识图谱中的基本单位，可以是任何可以独立存在的事物，如人、地点、事件等；属性用于描述实体的特征；关系则用于表示实体之间的关联。知识图谱的构建通常包括数据获取、实体识别、关系抽取、知识融合和知识推理过程。知识图谱构建过程如图 1 所示。

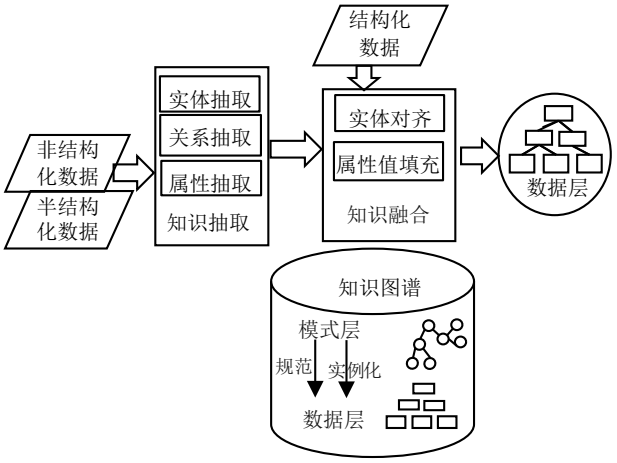

图 1 知识图谱构建过程

1.1 数据收集

数据采集是获取原始数据的过程。数据来源一般分为结构化数据、半结构化数据和非结构化数据。结构化数据一般具有较高的准确性和一致性如关系数据库、链接数据等。半结构化数据需要一定的处理才能提取出有用的信息如XML、JSON、百科类网站数据等。非结构化数据需要通过自然语言处理等技术进行解析和抽取如图片、音频、视频、文本等。需要对采集后的数据删除重复或错误的数据、将数据转换为统一的格式、处理不同数据源之间的数据冲突进行数据清晰和预处理过程，确保数据的一致性、准确性和完整性。

1.2 实体识别

实体识别（也称为命名实体识别，NER）是从数据中识别出独立的实体。是知识图谱构建过程中的一个重要步骤，旨在从文本中自动识别出具有特定意义的实体，如数据结构课程中的线性表、图、集合等。实体识别技术经历了从基于规则的方法到基于统计学习的方法，再到深度学习方法的发展过程。当前，深度学习方法（如循环神经网络RNN、卷积神经网络 CNN、Transformer 等）在实体识别任务中取得了显著成效。

1.3 关系抽取

关系抽取是提取实体之间的关系。从文本中抽取出两个或多个实体之间的语义关系，形成知识图谱中的边。关系抽取技术包括模式匹配、规则挖掘和深度学习方法等。深度学习方法通过自动提取文本特征并学习实体间的关系表示，实现了较高的抽取准确率和效率。关系抽取涉及多种关系类别，如“属于”“位于”“导致”等。

1.4 知识融合

知识融合是将不同来源的知识进行合并、整合和统一的过程，以消除实体指称项（实体的不同表示或称呼）与实体对象之间的歧义，构建出更加完整、准确和一致的知识体系，得到一系列基本的事实表达。知识融合技术涉及实体对齐、关系对齐、属性对齐等多个方面。实体对齐是确定不同数据源中相同实体的过程；关系对齐是确定不同数据源中相同关系的过程；属性对齐则是确定不同数据源中相同实体属性的过程。

1.5 知识推理

知识推理是利用已有的实体和关系信息，通过逻辑推理、归纳推理等方法推导出新的关系或属性，从而丰富和扩展知识图谱的过程。知识推理方法包括基于规则、基于路径、基于深度学习方法等。这些方法可以单独使用，也可以组合使用以提高推理的准确性和效率。

2 构建数据结构课程知识图谱

基于数据结构的核心概念和算法，构建一个完整的知识图谱。例如，可以将数组、链表、栈、队列、树、图等基本数据结构作为节点，而它们之间的关系则可以通过继承、组合、关联等方式表示。

2.1 构建数据结构课程知识核心节点

首先定义数据结构知识图谱的核心节点，即数据结构课程中实体。

①线性表：由一系列数据元素组成，数据元素之间是一对一的关系。线性表可以是顺序存储的（如数组），也可以是链式存储的（如链表）。

②数组（Array）：固定大小的连续内存空间，用于存储相同类型的数据。数组中的每个元素都可以通过索引快速访问。

③串：串是由两个或多个字符组成的有限序列，用于表示文本数据。

④栈（Stack）：后进先出（LIFO）的数据结构，可以基于数组或链表实现。

⑤队列（Queue）：先进先出（FIFO）的数据结构，同样可以基于数组或链表实现。

⑥广义表：是线性表的推广，可以包含零个或多个元素，且这些元素本身也可以是广义表。广义表是一种非线性的数据结构。

⑦图：图是由顶点（Vertex）和边（Edge）组成的复杂数据结构，用于表示对象之间的复杂关系，如网络分析、路径查找等。

⑧树（Tree）：是一种重要的非线性数据结构，由 n（n≥0）个有限节点组成一个具有层次关系的集合。每个节点最多有一个前趋（称为父节点），可以有 0 个或多个后继（称为子节点）。

⑨二叉树：是树的一种特殊形式，每个节点最多有两个子节点，通常被称为左子节点和右子节点。二叉树在数据结构和算法中有着广泛的应用，如搜索树、堆等。

⑩查找：查找是在数据结构中寻找满足某种条件的数据元素的过程。查找算法的性能直接影响到数据处理的效率。常见的查找算法包括顺序查找、二分查找、哈希查找等。

⑪ 排序：排序是将一组数据元素按照某种顺序重新排列的过程。排序算法的性能也是数据处理中的一个重要方面。常见的排序算法包括冒泡排序、选择排序、插入排序、快速排序、归并排序等。

2.2 定义核心节点间的关系与构建详细属性

2.2.1 定义关系

明确各核心节点之间的层级和依赖关系，旨在清晰表达数据结构核心节点的内在联系、转换方式以及它们在实际应用中的作用。例如，以线性表与其他核心节点为例，在数据结构课程中，线性表作为一种基础而重要的数据结构，与栈、队列、串、数组、广义表、图、树、二叉树、查找、排序等核心节点之间都存在着密切的关系。

①线性表与栈的关系：线性表是栈的基础，栈是一种

特殊的线性表，其操作被限制在表的一端（称为栈顶）进行。栈的“后进先出”（LIFO）特性正是基于线性表的序列性来实现的。线性表支持在任意位置进行插入和删除操作，而栈仅允许在栈顶进行这些操作。

②线性表与队列的关系：线性表是队列的基础，队列同样是一种特殊的线性表，但操作被限制在表的两端进行，一端用于插入（队尾），另一端用于删除（队头）。队列的“先进先出”（FIFO）特性也依赖于线性表的序列性。线性表允许在任意位置进行插入和删除，而队列的插入和删除分别被限制在表的两端。

③线性表与串的关系，串是线性表的特例，串是由零个或多个字符组成的有限序列，可以看作是线性表的一种特殊形式，其中数据元素为字符。串和线性表都支持遍历、查找、插入、删除等操作，但串的操作通常针对字符进行。

④线性表与数组的关系，数组是线性表的物理实现。在内存中，数组通过连续存储相同类型的数据元素来实现线性表，数组支持随机访问，即可以在常数时间内访问任意位置的元素。数组的大小在创建时通常就确定了，是静态的；而线性表的概念更加抽象，可以是静态的（如数组）也可以是动态的（如链表）。

⑤线性表与广义表的关系，广义表是线性表的推广，广义表是线性表的扩展，其元素不仅可以是单个数据元素，还可以是另一个广义表。这使得广义表能够表示更加复杂的数据结构。广义表支持嵌套，即一个广义表可以包含另一个广义表作为元素，这种嵌套关系在逻辑上与线性表的序列性相似，但更加复杂和灵活。

⑥线性表与图、树、二叉树的关系，线性表是这些复杂数据结构的基础，虽然图、树和二叉树在结构上比线性表更复杂，但它们都是由节点（或顶点）和边（或链接）组成的。在某种程度上，可以将这些复杂数据结构中的节点看作是线性表中的元素，而边则定义了这些元素之间的关系。图、树和二叉树的遍历过程（如深度优先搜索、广度优先搜索）可以看作是线性表遍历操作的扩展。此外，这些复杂数据结构上的许多操作（如查找、插入、删除）也可以借鉴线性表上的操作思想。

⑦线性表与查找、排序的关系，查找和排序操作适用于线性表。查找是数据结构中查找具有特定值的元素的过程，而排序是将一组数据按照某种顺序重新排列的过程。这两种操作都广泛应用于线性表上。查找和排序算法的效率与所使用的数据结构密切相关。例如，在有序数组中可以使用二分查找来提高查找效率；而在链表上进行排序时，可能需要采用归并排序等特定的算法来优化性能。

2.2.2 构建详细属性

为核心节点添加基本概念描述，包括其定义、特点、应用场景等。列出每个数据结构支持的主要操作（如插入、删除、查找等）以及相关的算法实现。

①线性表的属性：

元素类型：存储元素的类型。

长度：线性表中元素的个数。

存储结构：顺序存储（数组）或链式存储（链表）。

基本操作：插入、删除、查找、遍历等。

②栈的属性：

容量：栈能存储的最大元素数。

栈顶：栈中最后插入的元素的位置。

栈底：栈中最早插入的元素的位置。

基本操作：入栈（push）、出栈（pop）、查看栈顶元素（peek/top）、判断栈空（isEmpty）等。

③队列的属性：

容量：队列能存储的最大元素数。

队头：队列中最早插入的元素的位置。

队尾：队列中最后插入的元素的位置。

基本操作：入队（enqueue）、出队（dequeue）、查看队头元素（front）、判断队列空（isEmpty）等。

④串的属性：

长度：串中字符的个数。

字符集：串中字符所属的集合。

基本操作：连接（concat）、查找（find）、替换（replace）、子串（substring）、插入（insert）、删除（delete）等。

⑤数组的属性：

维度：数组的维数（一维、二维等）。

长度：数组在每一维的大小。

元素类型：数组中元素的类型。

索引：访问元素的位置标识。

⑥广义表的属性：

深度：广义表展开后所含括号的层数。

表头：非空广义表的第一个元素。

表尾：非空广义表中除第一个元素之外其余元素组成的广义表。

基本操作：创建、访问、修改、遍历等。

⑦图的属性：

顶点数：图中顶点的个数。

边数：图中边的个数（无向图）或弧的个数（有向图）。

连通性：图中顶点间的连接情况。

存储结构：邻接矩阵、邻接表、边表等。

基本操作：遍历（DFS/BFS）、查找路径、求最短路径等。

⑧树的属性：

根节点：树的唯一入口点。

叶子节点：没有子树的节点。

深度：树中节点的最大层次。

存储结构：通常使用指针或引用来表示节点间的关系。

⑨二叉树的属性：

左子树：节点的左孩子构成的二叉树。

右子树：节点的右孩子构成的二叉树。

特殊类型：满二叉树、完全二叉树、二叉搜索树等。

2.3 知识图谱可视化

对数据结构课程中线性表的操作和算法知识图谱可视化如图 2 所示。

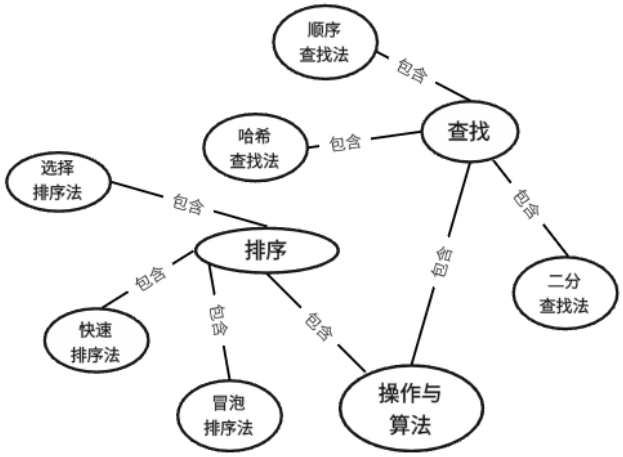

图 2 线性表的操作和算法知识图谱

3 结语

论文详细阐述了知识图谱的基本概念、构建方法及其在数据结构课程教学中的具体应用。通过知识图谱能够将复杂的数据结构知识点进行系统化、层次化的组织和展示，帮助学生更好地理解各知识点之间的内在联系和逻辑关系。同时，知识图谱还能够提供丰富的关联信息和可视化展示，激发学生的兴趣和积极性，从而提升教学效果。而知识图谱在数据结构课程教学中的应用仍面临一些挑战。例如，如何高效地构建和维护知识图谱、如何确保知识图谱的准确性和完整性、如何更好地将知识图谱与具体的教学任务相结合等，都需要在未来的研究和实践中不断探索和完善。未来，随着技术的不断进步和应用的不断深入，相信知识图谱将能够更好地服务于数据结构课程的教学工作，为培养具有创新精神和实践能力的计算机科学人才做出更大的贡献。

参考文献：

[1] 甘书灵,史东辉,王园园.基于知识图谱技术的计算机教育学情内涵探索[C]//全国高等学校计算机教育研究会、中国计算机学会、教育部高等学校计算机类专业教学指导委员会、全国高等学校计算机教育研究会,2023.

[2] 刘泓宇,王晓静.国内计算机辅助翻译知识图谱计量分析[J].信息与电脑(理论版),2024,36(3):1-4.

[3] Computers. Findings from Chung-Ang University in Computers Reported (Social Event Decomposition for Constructing Knowledge Graph)[J]. Computer Technology Journal,2020.

[4] Computers - Computer Security. New Findings from Illinois Institute of Technology in the Area of Computer Security Reported (Social Network De-anonymization and Privacy Inference With Knowledge Graph Model)[J]. Computer Weekly News,2019.

[5] Vincent Lully, Philippe Laublet, Milan Stankovic, et al. Exploring the synergy between knowledge graph and computer vision for personalisation systems[J]. Procedia Computer Science, 2018(137):175-186.

[6] 邓文锋.计算机基础课程的知识图谱教学分析[J].电子技术,2022,51(5):250-251.

[7] 郑理欣.面向计算机领域的多模态知识图谱构建方法研究[D].石家庄:河北科技大学,2022.

作者简介：董敏（1976-），女，中国陕西西安人，高级工程师，从事人工智能和软件工程研究。

基金项目：西安思源学院 2024 年校级基于知识图谱的一流课程建设项目“数据库原理及应用课程设计”。