Automated mapping from IFC data model to relational database model

Hongling Guo[1*], Ying Zhou[1], Xiaotian Ye[1], Zhubang Luo[1], Fan Xue[2]

[1] Department of Construction Management, Tsinghua University, Beijing 100084, China;
[2] Department of Real Estate and Construction, The University of Hong Kong, Hong Kong SAR, China

This is the post-print version of the paper:

Guo, H., Zhou, Y., Ye, X., Luo, Z., & Xue, F. (2020). Automated mapping from IFC data model to relational database model (IFC 数据模型至关系型数据库模型的自动映射). *Journal of Tsinghua University (Science and Technology)*, in press. Doi: 10.16511/j.cnki.qhdxxb.2020.22.030

Abstract

BIM (Building Information Modelling) plays an important role in the construction industry, while IFC (Industry Foundation Classes), as an international data exchange model, provides a useful support for the research and application of BIM, particularly BIM-related redevelopment. However, the limitations of IFC make it difficult to use IFC as the main data format in redevelopment. Therefore, an automatic mapping method from IFC data model to relational database model is proposed in this research. According to the structure of IFC data model, first, mapping rules for Entity, Type, Function, Rule, Property Set, and Quantity Set are defined respectively to realize data analysis, data mapping, and information retrieval. Then two experiments are conducted to test the validity of the proposed method, indicating that the relational database is superior to IFC files in terms of information integrity and the efficiency of data management. Moreover, two cases are used to show the potential of the mapped relational database. The results show that the mapping method not only improves the integrity and efficiency of BIM data management, but also promotes the use of intelligent construction as well as the development of informatization of the construction industry.

Keywords

BIM; IFC; relational database; automatic mapping

IFC 数据模型至关系型数据库模型的自动映射 [1]

郭红领 [1]，周颖 [1]，叶啸天 [1]，罗柱邦 [1]，薛帆 [2]

（1 清华大学 建设管理系，北京 100084；2 香港大学 房地产与建设系，香港）

摘要：建筑信息模型（Building Information Modelling, BIM）为建筑业信息化提供重要支撑，而 IFC（Industry Foundation Classes）作为国际通用的数据模型，则为 BIM 的研究与应用提供有效支持。然而，IFC 的局限性使其不宜作为信息系统二次开发时数据存储的主要形式。为此，本研究提出一种自动映射方法，实现 IFC 数据模型到关系型数据库模型的自动映射。首先，根据 IFC 数据模型结构，依次定义实体、类型、函数、规则、属性集和数量集的映射规则，实现数据解析、数据映射存取和信息检索处理；然后，结合工程数据，证明映射得到的关系型数据库在信息传递完整性和数据管理效率方面均优于直接操作 IFC 文件；此外，通过两个案例探索映射的关系型数据库的专业应用前景。结果表明，该映射方法不仅能提高 BIM 数据管理的完整性和效率，还能促进智能建造的拓展应用和建筑业信息化的发展。

关键词：BIM；IFC；关系型数据库；自动映射

中图分类号：TU71

Introduction

信息化是建筑业发展战略的重要组成部分，而建筑信息模型（Building Information Modelling, BIM）是建筑业信息化的基础，其核心是工程各种信息的集成、共享与交互[1]。建筑模型信息的传递主要涉及三个维度，一是时间维度，即在建筑工程全生命周期（如规划、设计、施工、运维和拆除）内传递；二是专业维度，即在建筑工程涉及的多个设计专业（如建筑、结构、给排水、暖通等）间传递，避免或减少设计冲突[2]；三是参与方维度，即在建筑工程利益相关方（如建设方、施工方、设计方等）间传递，避免信息孤岛和信息断层[3]。而 BIM 在建筑工程不同阶段、不同设计专业、不同参与方的应用，需要不同 BIM 应用软件（如 Revit，Tekla Structures，Rhinoceros 等）和数据类型的支持，即需要标准数据转换实现工程各阶段、各专业和各参与方之间的有效信息传递。

为此，buildingSMART 提出 IFC（Industry Foundation Classes）[4]作为 BIM 信息交换的标准[3]。IFC 标准采用面向对象的 EXPRESS 语言描述建筑产品数据，通过 IFC 文件进行存储和管理。尽管 IFC 标准为 BIM 综合应用及相关软件二次开发提供一定支持，但 IFC 文件自身存在局限性，例如不支持并行操作、缺乏安全性、且常常包含过多冗余信息，故不能成为信息系统数据存储的主要形式[1][5]。相比之下，数据库是长期存储在计算机内、有组织的、可共享的数据集合。因此，将 IFC 数据模型转换为数据库模型，不仅有利于 BIM 数据的共享和并行操作，还能有效提高数据的安全性、可靠性和持久性[7]。

目前将 IFC 数据模型映射为数据库模型的相关研究，可分为映射为关系型数据库和非关系型数据库两类。关系型数据库是依据关系模型创建的数据库，其中关系模型指二维表格模型，因而关系型数据库是由二维表及其之间的联系组成的数据组织[8]。非关系型数据库是指非关系型的，分

[1] **基金项目：**国家自然科学基金资助项目(51578318、51208282)；清华大学-广联达建筑信息模型联合研究中心资助项目；清华大学自主科研计划资助项目（2019Z02HKU）。

作者简介：郭红领（1978—），男，副教授。

通信作者：郭红领，副教授，E-mail: hlguo@tsinghua.edu.cn。

布式的，且一般不保证遵守 ACID（原子性（Atomicity）、一致性（Consistency）、隔离性（Isolation）和持久性（Durability））原则的数据存储系统。相比之下，关系型数据库成本低、有利于持久存储，而非关系型数据库支持多种存储格式、查询效率高、可扩展性强。对于关系型数据库，李小林等[9]针对 STEP 标准利用数据字典实现不同数据类型从 EXPRESS 数据向关系型数据库的转换，局限是需要手动创建数据表；张洋[7]利用关系型数据库存储 IFC 数据，实现 IFC 模型结构化数据的读取、保存、提取、集成和扩展，但侧重于建筑设计阶段，未对后期施工应用作进一步研究分析；成于思等[1]在考虑继承关系和聚类关系的基础上定义了从 IFC 数据模型向关系型数据模型的转换规则，但未考虑属性约束，因此难以转换全部信息； Solihina 等[10]根据星型模型将 IFC 数据转化为 BIMRL 模式，从而实现 BIM 数据的标准化 SQL 查询，可根据构件属性和空间位置检索，但仅提供只读访问，且与空间相关的检索效率较低。而对于非关系型数据库，Faraj 等[11]提出了基于 Web 的 IFC 共享项目环境 ŽWISPER，利用 STEP 工具 ST-Developer 实现 EXPRESS 的解析并映射到 ObjectStore，但未系统比较其数据管理效率和对工程项目产生的经济价值；Tanyer 等[12]将 IFC 数据转存入面向对象数据库，开发新的 4D（Four-Dimension）规划工具，但转存后的 IFC 文件大小是原来的 2-5 倍，降低系统性能；Lee 等[13]基于对象-关系数据库开发了对象-关系 IFC 服务器，通过简化继承结构和聚合概念的映射过程来提高查询性能，但未考虑数据库的其他操作性能；张迪等[14]用对象关系型数据库 Oracle 管理 IFC 数据，存储建筑数据模型，但仅定义四类数据类型的映射规则，亦未进一步分析其数据管理效率。

此外，一些研究专门对映射后的关系型数据库和面向对象数据库的文件管理效率进行比较分析。例如，黄忠东等[15]比较基于关系型数据库和面向对象数据库实现标准数据存储界面的优缺点，虽然面向对象数据库的模式转换更直接，但是考虑到关系型数据库相关技术更成熟，拥有标准的数据模式和接口，且应用更广泛，故关系型数据库在现阶段更适用。Jeong 等[16]比较使用关系型数据库和对象-关系型数据库查询含有 9115 个对象的 IFC 文件的效率，发现除查询简单对象外，对象-关系型数据库的查询效率更高。陆宁等[17]对两种具有代表性的数据库——面向对象数据库 Versant Object Database 8 和关系型数据库 SQL Server 2005 在插入、查询、更新和删除数据方面的效率进行比较，发现对于超过数万实体构成的 IFC 数据，面向对象数据库的管理效率相对更高。Lee 等[13]通过案例测试发现对象-关系数据库的查询性能优于关系型数据库，且数据规模越大，差别越大，原因在于 IFC 采用面向对象的 EXPRESS 语言，因此面向对象数据库能够更直接、自然地表示 EXPRESS 的语义，对大量 IFC 数据的处理效率也相对更高。然而，EXPRESS 和面向对象数据库的数据类型并不一一对应，且不同面向对象数据库的模式差异较大，可移植性较低[15]。另外，由于关系型数据库拥有以下优点，即丰富的计算机编程语言接口，成熟的商业软件和配套工具软件，从促进建筑信息有效传递的角度出发，选择将 IFC 数据模型映射为关系型数据库模型更有利于实现 BIM 数据的传递和共享。

综上所述，虽然现有研究对于 IFC 数据到关系型数据库的自动映射已经取得一定进展，但仍存在以下局限，即未考虑完整约束导致信息丢失，未进一步分析其映射效率和数据管理效率，且应用主要集中于建筑设计阶段，较少考虑施工阶段。因此，仍需要一种通用的转换方法，不仅实现各种 BIM 软件导出的 IFC 数据模型到关系型数据库模型的自动映射，还在关系型数据库中尽量保留原始的构件信息，以便在此基础上针对后期施工应用进行二次开发。为此，本研究基于面向对象的 Java 语言提出一种 IFC 数据模型至关系型数据库模型的自动映射方法，进一步比较该方法和直接操作 IFC 文件在信息传递完整性和数据管理效率方面的差异，并通过不同的施工应用案例，验证所提出方法的专业应用前景。

1 自动映射方法构建

The proposed automatic mapping

1.1 IFC 模型数据结构概述

IFC 标准作为国际通用的 BIM 数据标准，采用面向对象的 EXPRESS 语言描述建筑产品数据。这意味着既可以描述真实的物理对象，如门、窗等，也可以表示抽象的概念，如组织过程[7]。IFC 数据模型由 4 个层次构成，从低到高依次划分为资源层、核心层、共享层和领域层，各层之间遵循“重力原则”，即每个层次只能引用同层次或低层次的信息。

一个完整的 IFC 模型由实体（Entity）、类型（Type）、函数（Function）、规则（Rule）、属性集（Property Set）和数量集（Quantity Set）构成。实体是有属性与约束定义的信息集，是 IFC 数据模型的核心。类型定义作为 IFC 模型的主要组成部分，包括三种数据类型：定义类型（Defined Type）、枚举类型（Enumeration Type）和选择类型（Select Type）。函数和规则用于计算或者约束实体的属性，亦可用于验证模型的正确性。属性集则是属性的集合，用于描述事物和概念，可以被不同的对象引用。数量集是定量信息的集合，也可以被不同的对象引用。表 1 展示 IFC4x1 中模型元素在各功能层的数量分布。

表 1 IFC 模型元素在各功能层中的分布

	类型	实体	函数	规则	属性集	数量集	合计
领域层	102	198	0	0	471	70	841
共享层	40	100	0	0	118	17	275
核心层	35	125	5	1	31	6	203
资源层	223	378	47	1	17	0	666
合计	400	801	52	2	637	93	1985

注：表中数据针对 IFC4x1 统计

1.2 映射规则构建

为实现 IFC 数据在关系型数据库中的持久存储，本研究将构建从 IFC 到 Java 再到 MySQL 数据库的映射规则。一个 IFC 模型由大量实体对象组成，实体是 IFC 信息交换的载体，通过实体、类型、函数、规则、属性集值和数量集值的引用进行定义，下面对此分别建立相应的映射规则。

1）实体映射规则

对于实体映射规则，区别于 EXPRESS 语言支持多重继承，IFC 的实体是单继承，这与 Java 面向对象继承的概念一致，因此 IFC 的实体在映射过程中依次对应 Java 类和关系型数据库的表。

2）类型映射规则

如表 2 所示，IFC 的定义类型转换为 Java 中的基本数据类型或者 String 类型，在 MySQL 数据库中则依次存储为对应的基本类型。IFC 的枚举类型映射为 Java 中的枚举类型，由于枚举类型的取值均为字符串，因此对应 MySQL 数据库用 varchar 类型存储。对于选择类型，把 IFC 的选择类型对应的类的值存储到 Java 中的 String 类型，在 MySQL 数据库中则存储为 varchar 类型。比如，IfcActorSelect 是选择类型，可供选择的类型有 IfcPerson、IfcPersonAndOrganization 和 IfcOrganization 三种，解析过程中根据 IFC 文件的内容首先映射到 Java 对应的这三种类型

（IfcPersonAndOrganization、IfcOrganization 和 IfcPerson）中，各类型中的值用 String 类型存储，而在 MySQL 数据库中用 varchar 类型存储。

3）函数和规则的映射规则

IFC 的函数对应 Java 类中的函数名，但这些函数在 MySQL 数据库中不做存储。而规则对应于 Java 类中的逻辑过程，也不在 MySQL 数据库中存储。

4）属性集和数量集的映射规则

IFC 的属性集和数量集，虽然在不同层次的不同模块中有不同的字段，但都是映射为 Java 类中基本数据类型或者 String 类型，因此在 MySQL 数据库中存储为基本类型中的 varchar 类型。

表 2 IFC 的定义类型映射规则

IFC	Java		MySQL
定义类型	基本数据类型	String 类型	基本类型
INTEGER	int	-	int
DOUBLE	double	-	double
BINARY	byte	-	binary
BOOLEAN	boolean	-	int
STRING	-	String	varchar
LOGICAL	-	String	varchar
LIST	list	-	varchar
SET	set	-	varchar
REAL	double	-	double

1.3 IFC 数据解析和映射

完成 IFC 数据模型到 MySQL 数据模型映射规则的构建后，需要对 IFC 数据进行解析。本研究在 jdk1.8 开发环境下，采用 java 语言，使用 hibernata 开源库，采用 buildingSmart 提供的开源解析引擎（IFCjar 包为 com.apstex.ifc4x2toolbox.jar），通过设计与 IFC 标准对应的 Java 类以及类的属性对 IFC 数据进行深度解析，部分属性采用一对一映射，部分属性则采用一对多映射 ，实现 IFC 共享层实体类（IFC4x1 共 48 种）的自动解析与映射。以某个实际工程 IFC 模型中的 IfcBeam 实体为例，如表 3 所示，该实体包含 GlobalId、OwnerHistory、Representation 等多种属性。其中，梁的总编号属性属于一对一映射，其属性名为 GlobalId，数据类型为 IfcGloballyUniqueId，对应的 Java 类自定义属性名为 GlobalId，数据类型为 String；而梁的表示属性属于一对多映射，其属性名为 Representation，数据类型为 IfcProductRepresentation，对应的 Java 类的自定义属性名有多个，包括体积属性名 Volume，数据类型为 double，长度属性名 Length，数据类型为 double，横截面尺寸属性名 SectionB 和 SectionH，数据类型均为 double。

表 3 IfcBeam 实体在 IFC 标准，Java 类和 MySQL 中包含属性的对比

IFC		Java		MySQL	
IfcBeam实体		Java类		IfcBeam表	
属性	数据类型	自定义属性	数据类型	字段	数据类型
GlobalId	IfcGloballyUniqueId	GlobalId	String	GlobalId	varchar
OwnerHistory	IfcOwnerHistory	ownerHistory	String	OwnerHistory	varchar
…	…	…	…	…	…
Representation	IfcProductRepresentation	Volume	double	Volume	double
		Length	double	Length	double
		SectionB	double	SectionB	double
		SectionH	double	SectionH	double

...	...	...	...	...	...

1.4 IFC 数据自动存取

经过上述数据解析与映射，IFC 数据将暂存在 Java 对象中，可通过开源的对象关系映像框架 Hibernate 将 Java 对象数据进一步映射到 MySQL 数据库中，进而实现 IFC 数据在关系型数据库中的持久存储。仍以表 3 中的 IfcBeam 实体为例，根据映射规则，IfcBeam 实体映射为 MySQL 数据库中的 IfcBeam 表；Java 类中自定义属性 GlobalId 在 MySQL 数据库 IfcBeam 表中对应的字段名为 GlobalId，数据类型为 varchar；Java 类中自定义属性 Volume 在 MySQL 数据库 IfcBeam 表中对应的字段名为 Volume，数据类型为 double。需要说明的是，本研究采用的 Hibernate 是一个全自动的 Object Relational Mapping（ORM）框架，能够对连接 Java 和数据库的 Java DataBase Connectivity（JDBC）进行封装，实现自动创建数据库表并生成 SQL 操作语句。

1.5 信息自动检索与处理

对于映射得到的 MySQL 关系型数据库，还应实现数据库的高效访问，即信息自动检索与处理。实际操作中通常需要返回大量数据并面向互联网传输，因此本研究在进行数据库输出时把相关属性封装在一个类中，序列化为 JSON 数据，实现同时返回多行数据，也便于后期进行网络数据交换。其操作过程如下：首先，客户端向服务端请求数据查询接口，同时传递所查询构件的特征信息（如关键字段 GlobalId）；然后，服务端根据接受到的特征信息（如关键字段 GlobalId）检索 MySQL 数据库，将查询到的构件数据转换为 JSON 数据并返回给客户端；最后，客户端通过解析返回的 JSON 数据，获得构件的尺寸、位置、体积、方向等信息。

2 自动映射方法性能分析

Performance analysis

为验证上述自动映射方法的有效性，下文通过一个实际工程项目的 BIM 数据，对比直接操作 IFC 数据和映射得到的关系型数据库在信息传递完整性和数据管理效率方面的差异，选择上述测试指标的原因是信息传递完整性保证该方法的可用性，数据管理效率则体现其经济效益。

2.1 信息传递完整性对比

BIM 实现有效信息传递的前提是保证信息传递的完整性。当前 IFC 数据是以 IFC 文件的形式存储，通过兼容 IFC 的建模软件（如 Revit）将建筑模型导出为 IFC 文件，但此形式不便于直接和关系型数据库比较。考虑到 Revit 可以通过“导出-ODBC 数据库-机器数据源-新建-用户数据源（系统数据源）-SQL Server Native Client-自定义名称和服务器-身份验证-选择数据库-选择账号-登录服务器-Revit 导出数据”实现关系型数据库中 IfcBeam 表的自动导入，为更直观地比较，本研究对比 Revit 自动导出的 IfcBeam 表和利用本研究所提出方法映射得到的关系型数据库中 IfcBeam 表在包含属性上的差别，进而分析得到二者在信息传递完整性方面的差异，结果如表 4 所示。由此可见，映射得到的关系型数据库中 IfcBeam 表所包含的信息更完整，新增 Coordinates、Directions 等十多种属性。根据下文提出的应用前景，这些新增的属性可用于智慧工地，支持自动化吊装等，比如可以根据构件的基本信息如 Span、SectionB、SectionH、Weight 和 Coordinates 属性信息确定吊车选型，利用 IsExternal 和 ContourVertex 属性信息筛选吊臂长度。

表 4 两种方式得到的 IfcBeam 表包含属性信息的对比

属性	描述	Revit直接导出的IfcBeam表中包含与否	映射为关系型数据库后得到的IfcBeam表中包含与否
GlobalId	编号	是	是
Name	名称	是	是
ShearLength	剪切长度	是	是
TypeId	类型ID	是	是
Volume	体积	是	是
Length	长度	是	是
Elevation	标高	是	是
ReferenceElevation	参照标高	是	是
StructuralMaterial	结构材质	是	是
TopElevation	顶高程	是	是
BottomElevation	底高程	是	是
Purpose	结构用途	是	是
PhaseOfCreation	创建阶段	是	是
PhaseOfDismantling	拆除阶段	是	是
EstimatedRebar Volume	估计的钢筋体积	是	是
DesignOptions	设计选项	是	是
Annotation	注释	是	是
Sign	标记	是	是
IfcGUID	GUID	是	是
Coordinates	坐标	否	是
Directions	方向	否	是
IsExternal	是否外部	否	是
LoadBearing	是否承重	否	是
Slope	斜率	否	是
Span	跨度	否	是
SectionB	截面长度	否	是
SectionH	截面高度	否	是
Weight	重量	否	是
ContourVertex	轮廓顶点	否	是
YAxisOffsetValue	Y轴偏移值	否	是
YAxisPairPositive	Y轴对正	否	是
ZAxisOffsetValue	Z轴偏移值	否	是
ZAxisPairPositive	Z轴对正	否	是
YZAxisPairPositive	YZ轴对正	否	是

2.2 数据管理效率

为验证所提出方法的映射效率满足实时性要求，本研究分别统计实现 1 万、5 万、10 万、15 万和 20 万行 IFC 数据自动映射需要的时间，且为保证实验数据的可靠性，每个统计结果是 10 次测试结果的平均值，如表 5 所示。实验环境为搭载 Windows 10 操作系统的个人电脑，配备 Intel Core i7-6700K @ 4.00GHz 四核处理器和 32GB 内存。对于 10 万行及其以下数量级的 IFC 数据，其自动映射时间控制在 2 分钟以内，可满足工程需求。此外，数据库管理主要涉及数据的增删查改。工程实践中通常依靠应用软件对 IFC 实体对象进行增加和删除，而对 IFC 文件的数据操作主要是数据

查询和修改。为比较上述两个方法在数据管理效率上的差异，本研究还进一步对比直接操作 IFC 文件和操作关系型数据库对不同数量级 IFC 数据进行修改和查询的用时情况。

二者的数据查询效率对比结果如表 6 所示。对于同一构件类型的数据（即同一张表上的数据），发现虽然查询关系型数据库的首次用时高于直接查询 IFC 文件，但是查询关系型数据库的二次用时明显低于直接查询 IFC 文件。原因在于关系型数据库首次查询后将同一构件类型的数据缓存于内存中，从而降低其二次查询此类数据的耗时，且数据量越大，关系型数据库二次耗时的降低效果越显著。而直接查询 IFC 文件则与此相反，每次都需要重新检索 IFC 文件，因此其首次用时和二次用时相差不大。实际工程应用中通常需要多次查询同一构件类型数据，因此使用关系型数据库的查询效率高于直接查询 IFC 文件。

二者的数据修改效率对比结果如表 7 所示。修改关系型数据库的效率明显高于直接修改 IFC 文件，且随着构件数量从 1 万行增加到 20 万行，查询效率从 186 倍提升到 429 倍。这是由于 IFC 对象的属性修改后，需要更改整个文件的内容使修改生效，该过程相当于重写一次文件，所花费的时间长于修改关系型数据库，且随着数据量的增加，修改 IFC 文件所需的时间也加倍。

表 5 IFC 文件自动映射时间对比

构件数据量（万行）	映射时间（s）
1	6
5	44
10	117
15	163
20	475

表 6 IFC 文件和关系型数据库的查询效率对比

构件数据量（万行）	直接查询IFC文件时间（ms）		查询关系型数据库时间（ms）	
	首次用时	二次用时	首次用时	二次用时
1	54	61	607	25
5	137	141	656	41
10	147	158	658	52
15	246	204	661	54
20	351	319	677	60

表 7 IFC 文件和关系型数据库的修改效率对比

构件数据量（万行）	直接修改IFC文件时间（ms）	修改关系型数据库时间（ms）	前者与后者的效率比
1	19345	104	1：186
5	44893	173	1：259
10	60563	232	1：261
15	74702	369	1：202
20	174563	406	1：429

3 应用前景分析

Application scenarios

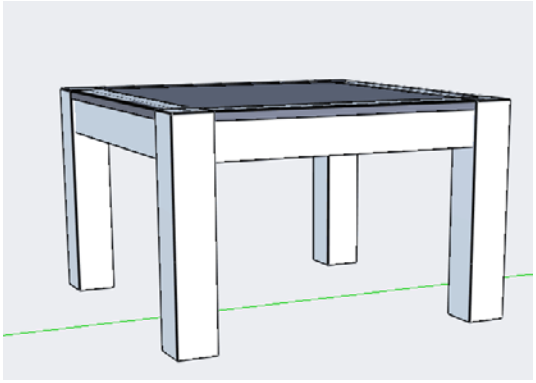
本研究提出的映射方法，不仅实现 IFC 对象从 IFC 数据模型到关系型数据库模型的自动映射，还在一定程度上保证了信息传递的完整性，因此有潜力支持相关专业的二次开发。下面通过两个案例展示映射得到的关系型数据库的专业应用前景。

3.1 自动生成构件级进度计划

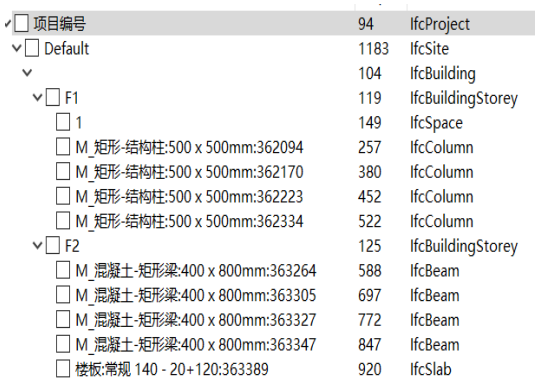
进度控制是工程项目管理三大控制之一，传统进度计划编制的基础是工作分解结构[18]。由于 IFC 数据模型具有面向对象的特征，本方法映射得到的关系型数据库是构件级的（见图 1）。在此基础上，自动生成进度计划需要进一步确定各构件的工作时间并增加工序约束。以某装配式框架结构项目为例，数据库中包含项目相关构件的基本信息，其吊装时间根据平均作业工效自动计算得到。而系统确定构件吊装顺序的原则是：（1）根据构件的标高自动确定其所在楼层，按所在楼层从低到高的顺序排列，（2）对于在同一实际楼层内的构件，初始默认吊装顺序为柱－梁－板－楼梯－阳台，从而自动生成构件级进度计划，如图 2 所示。

id	Area	Breadth	Category	GlobalId	Height	IsExternal	Length	LoadBearing	Name
1	214519.9999999978	0	基本墙:内部 - 砌块墙 100:88944	0Pu3RA_UPF_fHflos0hxZU	1730.0000	1	3429124	1	基本墙:内部 - 砌块墙 100:399031
2	3519000.000000012	0	基本墙:常规 - 300mm:88945	3DE07a59DCL8g3twyke70G	11730	1	3429124	1	基本墙:常规 - 300mm:470779
3	3609000.000000012	0	基本墙:常规 - 300mm:88945	0Pu3RA_UPF_fHflos0hxSp	12030	1	3429124	1	基本墙:常规 - 300mm:399706
4	555000.0000000019	0	基本墙:弹涂陶粒砖墙300:88953	3DE07a59DCL8g3twykeO43	1850	1	3429124	1	基本墙:弹涂陶粒砖墙300:450536
5	510000.00000000774	0	基本墙:常规 - 300mm:88945	3DE07a59DCL8g3twyke646	1700.0000	1	3429124	1	基本墙:常规 - 300mm:475117
6	510000.00000000175	0	基本墙:常规 - 300mm:88945	3DE07a59DCL8g3twykeRxg	1700	1	3429124	1	基本墙:常规 - 300mm:450561
7	4857599.999999939	0	基本墙:砖墙240mm:88951	0Pu3RA_UPF_fHflos0hxZm	20240	1	3429124	1	基本墙:砖墙240mm:399001
8	4319999.999999946	0	基本墙:砖墙240mm:88951	3DE07a59DCL8g3twyke4St	18000	1	3429124	1	基本墙:砖墙240mm:465372
9	509999.9999999987	0	基本墙:常规 - 300mm:88945	0Pu3RA_UPF_fHflos0hx97	1699.9999	1	3429124	1	基本墙:常规 - 300mm:400430
10	509999.9999999987	0	基本墙:常规 - 300mm:88945	0Pu3RA_UPF_fHflos0h_j4	1699.9999	1	3429124	1	基本墙:常规 - 300mm:410925
11	519000	0	基本墙:常规 - 300mm:88945	3DE07a59DCL8g3twyke6YO	1730	1	3429124	1	基本墙:常规 - 300mm:472691
12	210799.9999999966	0	基本墙:内部 - 砌块墙 100:88944	3DE07a59DCL8g3twykeRFJ	1700	1	3429124	1	基本墙:内部 - 砌块墙 100:453944
13	210799.99999999785	0	基本墙:内部 - 砌块墙 100:88944	3DE07a59DCL8g3twyke785	1700.0000	1	3429124	1	基本墙:内部 - 砌块墙 100:470254
14	555000.0000000019	0	基本墙:弹涂陶粒砖墙300:88953	0Pu3RA_UPF_fHflos0hupF	1850	1	3429124	1	基本墙:弹涂陶粒砖墙300:402086
15	3609000.000000012	0	基本墙:常规 - 300mm:88945	3DE07a59DCL8g3twyke7Xi	12030	1	3429124	1	基本墙:常规 - 300mm:468615
16	4857599.999999939	0	基本墙:砖墙240mm:88951	0Pu3RA_UPF_fHflos0hu4e	20240	1	3429124	1	基本墙:砖墙240mm:405313
17	509999.99999999825	0	基本墙:常规 - 300mm:88945	0Pu3RA_UPF_fHflos0hvnb	1700	1	3429124	1	基本墙:常规 - 300mm:406028
18	425000.0000000093	0	基本墙:面砖陶粒砖墙250:88187	3DE07a59DCL8g3twyke7GB	1700.0000	1	3429124	1	基本墙:面砖陶粒砖墙250:469728
19	438749.9999999912	0	基本墙:面砖陶粒砖墙250:88187	3DE07a59DCL8g3twyke6P6	1755	1	3429124	1	基本墙:面砖陶粒砖墙250:473261
20	210800.0000000139	0	基本墙:内部 - 砌块墙 100:88944	3DE07a59DCL8g3twyke77u	1700.0000	1	3429124	1	基本墙:内部 - 砌块墙 100:470803
21	425000.0000000043	0	基本墙:面砖陶粒砖墙250:88187	0Pu3RA_UPF_fHflos0hv6k	1699.9999	1	3429124	1	基本墙:面砖陶粒砖墙250:409543
22	3519000.000000012	0	基本墙:常规 - 300mm:88945	3DE07a59DCL8g3twyke0Ik	11730	1	3429124	1	基本墙:常规 - 300mm:481861
23	424999.999999989	0	基本墙:面砖陶粒砖墙250:88187	3DE07a59DCL8g3twyke4Jh	1699.9999	1	3429124	1	基本墙:面砖陶粒砖墙250:465408
24	424999.999999989	0	基本墙:面砖陶粒砖墙250:88187	3DE07a59DCL8g3twyke5JQ	1699.9999	1	3429124	1	基本墙:面砖陶粒砖墙250:461361
25	564000	0	基本墙:弹涂陶粒砖墙300:88953	0Pu3RA_UPF_fHflos0hvgS	1880	1	3429124	1	基本墙:弹涂陶粒砖墙300:406773
26	509999.9999999987	0	基本墙:常规 - 300mm:88945	0Pu3RA_UPF_fHflos0hvdW	1699.9999	1	3429124	1	基本墙:常规 - 300mm:407433
27	438750.0000000236	0	基本墙:面砖陶粒砖墙250:88187	3DE07a59DCL8g3twyke6sm	1755.0000	1	3429124	1	基本墙:面砖陶粒砖墙250:471899
28	438749.99999999825	0	基本墙:面砖陶粒砖墙250:88187	0Pu3RA_UPF_fHflos0hxYW	1755	1	3429124	1	基本墙:面砖陶粒砖墙250:399049
29	425000.0000000093	0	基本墙:面砖陶粒砖墙250:88187	3DE07a59DCL8g3twyke5Ru	1700.0000	1	3429124	1	基本墙:面砖陶粒砖墙250:460819

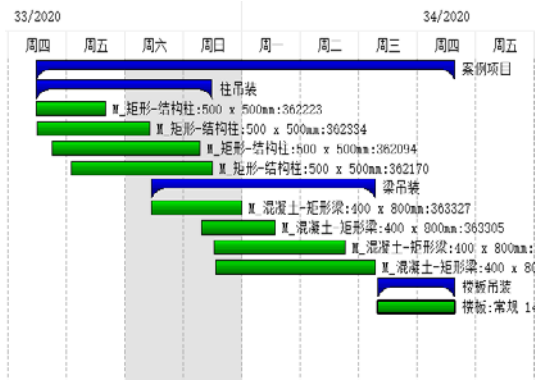
图 1 映射得到的关系型数据库 IfcWall 表（部分）



（a）结构模型



（b）构件明细



（c）进度计划甘特图

图 2 某装配式框架结构一层进度计划（部分）

3.2 自动吊装对象智能识别

起重机是施工现场使用最广泛的设备之一[19]，其作业自动化对于实现智能建造有重要意义，而吊装对象的自动识别是实现起重机自动化操作的前提。利用图像设备等采集构件信息是目前识别现场对象的常用手段，然而能够采集到的信息还非常有限，主要是构件的空间位置和颜色纹理信息。以移动式起重机的自动选型和定位为例，潘在怡[20]归纳出如下表 8 所示的三个约束条件和对应的信息需求。但表中大部分信息无法直接通过传感器采集，需要从解析的 BIM 模型中获取或进行转换计算，比如构件质量可基于解析得到的材质信息和体积信息确定。为此，可以结合图像设备采集的构件特征信息和吊装构件基础数据库确定吊装对象，从而实现吊装构件的自动识别（见表 9），从数据库中获得该构件的其他关键信息（如构件编码、重量、重心位置、安装位置等），并在数据库中更新吊装构件的真实原始坐标，为下一步吊装路径自动规划做准备。

表 8 移动式起重机自动选型和定位约束计算所需的信息

约束条件	所需信息	信息来源	需要BIM模型的数据
环境约束	建筑首层外轮廓坐标	解析的BIM模型需计算转换	首层墙的坐标、尺寸
	主干道、临时设施的轮廓坐标	解析的BIM模型直接获取	-
操作约束	吊装构件的起始位置	传感器获取	-
	吊装构件的目标位置	解析的BIM模型需计算转换	构件的相对位置、尺寸、方向、楼层层高等
安全约束	吊桩构件的质量	解析的BIM模型需计算转换	构件材质、体积
	吊装时建筑的高度、外部轮廓、吊臂位置	解析的BIM模型需计算转换	时间参数、楼板、墙、柱的坐标、位置

表 9 吊装构件自动识别（检索与更新）

Extracted features	Centroid coordinate: (0.466, -0.261, 2.157) Sizes (length*width*height): 0.82 * 0.44 * 0.26 Colors (B*G*R): (135, 124, 121)
Query	SELECT GlobalID, Weight, Target_Location FROM t_slab table WHERE Length between 0.77 and 0.87 and Width between 0.39 and 0.49 and Height between 0.21 and 0.31 and B between 115 and 155 and G between 104 and 144 and R between 101 and 141
Result 1	GlobalID: 0yrQFJS7b9auGYqnNNxxN3; Weight: 0.192; Target_Location:(5.0, 2.0, 6.0)
Update	Update t_slab SET Original_Location='(0.466, -0.261, 2.157)' WHERE GlobalID=0yrQFJS7b9auGYqnNNxxN3
Result 2	Rows matched: 1; Rows changed: 1

4 结论

Conclusion

本研究提出一种实现 IFC 数据模型到关系型数据库模型的自动映射方法，针对 IFC 模型元素建立从 IFC 到 Java 再到 MySQL 数据库的映射规则，实现建筑模型 IFC 数据到关系型数据库的自动映射，并结合实际工程的 BIM 数据从数据完整性和数据管理效率两个方面对映射方法的成效进行了分析。在此基础上，进一步结合案例分析该映射方法得到的关系型数据库在施工阶段的专业应用前景。结果表明，本方法不仅能够提高 BIM 数据管理的完整性和效率，还可促进智能建造的拓展应用和建筑业的信息化发展。具体如下：（1）利用本方法将 IFC 数据模型转换为关系型数据库模型后，信息更完整；（2）所提出方法的映射效率满足工程实时性要求，且关系型数据库的修改效率和综合查询效率都明显高于直接操作 IFC 文件的效率，且数据量越大，效率提升越显著；（3）在映射得到的关系型数据库基础上进行二次开发，有潜力支持智能建造的拓展应用（如进度计划自动生成、吊装构件自动识别等）。

此外，本研究还存在一定的局限性。特别是，在方法性能分析方面，仅考虑 Revit 导出的 IFC 文件及数据库表，没有考虑其他 BIM 软件导出的相关文件。在未来研究中，将对此进行更为广泛的分析。

Acknowledgements

本论文是在 2018 年第四届全国 BIM 学术会议上发表的《IFC 数据到关系型数据库的自动映射方法研究》[6]的扩展。

参考文献

References

[1] 成于思, 李启明, 成虎. IFC 数据在关系数据库上的实现研究与应用 [J]. 计算机应用与软件, 2014, 31(11): 34-44.
CHENG Y S, LI Q M, CHENG H. On implementing IFC data in rational database and its application [J]. Computer Applications and Software, 2014, 31(11): 34-44. (in Chinese)

[2] 赖华辉, 邓雪原, 刘西拉. 基于 IFC 标准的 BIM 数据共享与交换 [J]. 土木工程学报, 2018, 51(4): 121-128.
LAI H H, DENG X Y, LIU X L. IFC-based BIM data sharing and exchange [J]. China Civil Engineering Journal, 2018, 51(4): 121-128. (in Chinese)

[3] 何关培. 实现 BIM 价值的三大支柱-IFC/IDM/IFD [J]. 土木建筑工程信息技术, 2011, 03(01): 108-116.
HE G P. Three pillars to realize the value of BIM-IFC / IDM / IFD [J]. Journal of Information Technology in Civil Engineering and Architecture, 2011, 03(01): 108-116. (in Chinese)

[4] BuildingSMART International. Industry Foundation Classes (IFC) - An Introduction. [S/OL]. [2020-06-30]. https://technical.buildingsmart.org/standards/ifc/.

[5] SUN J, LIU Y S, GAO G, et al. IFCCompressor: A content-based compression algorithm for optimizing Industry Foundation Classes files [J]. Automation in Construction, 2015, 50: 1-15.

[6] 周颖, 郭红领, 罗柱邦. IFC 数据到关系型数据库的自动映射方法研究[C]// 第四届全国 BIM 学术会议论文集. 合肥, 2018, 311-317.
ZHOU Y, GUO H L, LUO Z B. Research on automatic mapping method from IFC data to relational database[C]// Proceedings of the 4th National BIM Academic Conference. Hefei, 2018, 311-317. (in Chinese)

[7] 张洋. 基于 BIM 的建筑工程信息集成与管理研究 [D]. 北京: 清华大学, 2009.
ZHANG Y. Research on BIM-based building information integration and management [D]. Beijing: Tsinghua University, 2009. (in Chinese)

[8] 张巨俭. 数据库基础案例教程与实验指导 [M]. 北京: 机械工业出版社, 2011.
ZHANG J J. Basic database case tutorial and experimental guidance [M]. Beijing: China Machinery Industry Press, 2011. (in Chinese)

[9] 李小林, 易红, 吴锡英. EXPRESS 数据模式在关系数据库上的实现 [J]. 计算机集成制造系统, 1999, (1): 64-68.
LI X L, YI H, WU X Y. Implementation of EXPRESS data mode on relational database [J]. Computer Integrated Manufacturing System,1999, (1): 64-68. (in Chinese)

[10] SOLIHIN W, EASTMAN C, LEE Y C, et al. A simplified relational database schema for transformation of BIM data into a query-efficient and spatially enabled database [J]. Automation in Construction, 2017, 84: 367-383.

[11] FARAJ I, ALSHAWI M, AOUAD G, et al. An industry foundation classes web-based collaborative construction computer environment: WISPER [J]. Automation in Construction, 2000, 10(1): 79-99.

[12] TANYER A M, AOUAD G. Moving beyond the fourth dimension with an IFC-based single project database [J]. Automation in Construction, 2005, 14(1):15-32.

[13] LEE G, JEONG J, WON J, et al. Query performance of the IFC model server using an object-relational database approach and a traditional relational database approach [J]. Journal of Computing in Civil Engineering, 2014, 28(2): 210-222.

[14] 张迪, 刘华, 李航. 基于对象关系型数据库的 IFC 存储模型 [J]. 城市建筑, 2018, No.271(02):40-42.
ZHANG D, LIU H, LI H. On database IFC storage model based on the object-relational type [J]. Urban and Architecture, 2018, No.271(02):40-42.

[15] 黄忠东, 杨小虎, 董金祥. SDAI 实现中若干关键技术的研究 [J]. 计算机辅助设计与图形学学报, 2001, 13(11): 977-982.
HUANG Z D, YANG X H, DONG J X. Research on key techniques in SDAI implementation [J]. Journal of Computer-Aided Design & Computer Graphics, 2001, 13 (11): 977-982. (in Chinese)

[16] JEONG J, LEE G, KANG H. Preliminary performance evaluation of an ORDB-based IFC server and an RDB-based IFC server by using the BUCKY benchmark method[C]// Proceedings of CIB World Congress 2010, Salford, UK, 2010, 192-200.

[17] 陆宁, 马智亮. 利用面向对象数据库与关系数据库管理 IFC 数据的比较[J]. 清华大学学报(自然科学版), 2012, 52(6):836-842.
LU N, MA Z L. Comparison of managing IFC data using object-oriented and relational databases [J]. Journal of Tsinghua University (Science and Technology), 2012, 52 (6): 836-842. (in Chinese)

[18] 丁士昭. 工程项目管理 [M]. 第二版. 北京: 中国建筑工业出版社, 2014.
DING S Z. Engineering project management [M]. 2nd ed. Beijing: China Construction Industry Press, 2014. (in Chinese)

[19] KANG S, MIRANDA E. Planning and visualization for automated robotic crane erection processes in construction [J].

Automation in Construction, 2006, 15(4): 398-414.
[20] 潘在怡. 虚拟施工场景下移动式起重机自动选型与定位方法研究 [D]. 北京: 清华大学, 2018.
PAN Z Y. Method of automated selection and location of mobile cranes in virtual construction[D]. Beijing: Tsinghua University, 2018. (in Chinese)