Combinatorial Data Models for Topological Visualization

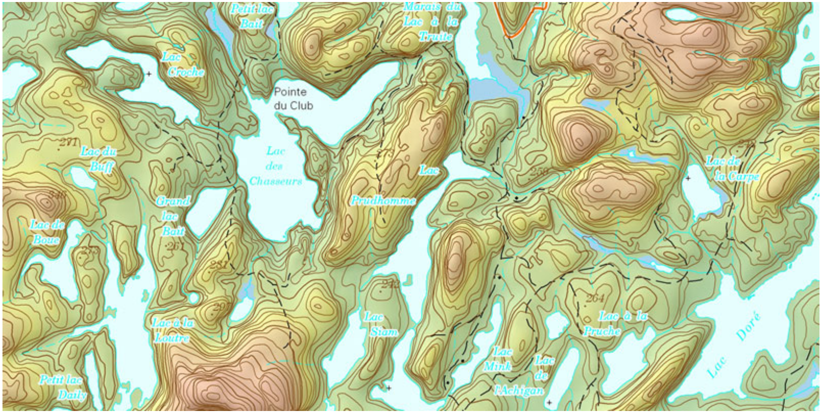
Rachel Kitchen

May 12, 2019

Table of contents

Topological Modeling

Most topological modeling ignores the interior of buildings.



http://www.macarte.ca/topographic.php

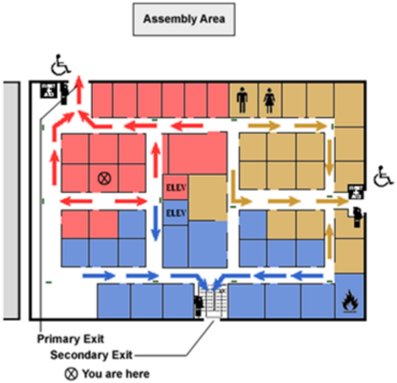
Data Model Description

This data model models the interior layout of structures.
Accomplished using the following combinatorial methods:

- Poincaré Duality Theorem
- Graph Theory
- Node Reference Structure

Problem Scenarios

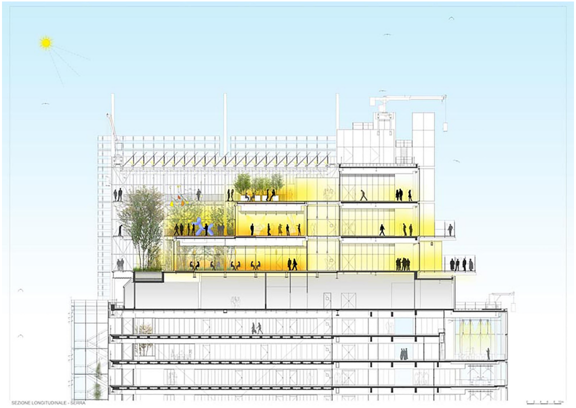
Emergency Response: Finding the shortest path to a room.



https://www.osha.gov/SLTC/emergencypreparedness/gettingstarted$_e$*vacuation.html*

Problem Scenarios

Company Building: Mapping the relationships between each room.



https://www.archdaily.com/630496/intesa-sanpaolo-office-building-renzo-piano/55529c20e58ece8a26000023-intesa-sanpaolo-office-building-renzo-piano-longitudinal-section-2

Poincaré Duality

Use Poincaré Duality Theorem to map 3D objects to simple graph elements. In a graph G=(V,E):

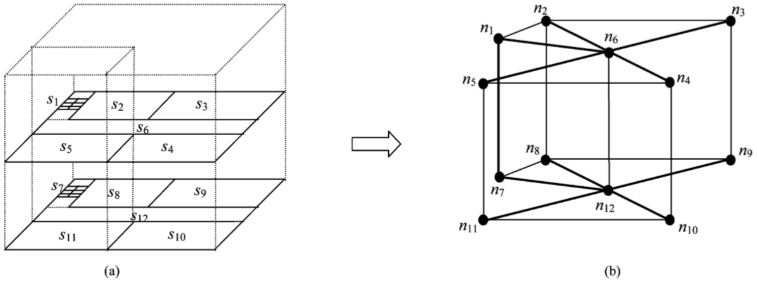
- $Rooms \longrightarrow vertices(V)$
- $Hallways/stairwells \longrightarrow edges(E)$

The theorem preserves the features of the original object, for example:

- Room capacity
- Room type (conference room, office, etc.)
- Company ownership
- and any more features which are important for the application.

Creating a Graph

The purpose of the combinatorial data model is to map a building structure as a simplified structure. An example is visualized below.



The node relation structure for a building. [1]

Creating a Graph

All vertices are horizontally and vertically related to other vertices. To represent this, the sets of related vertices are modeled by graphs:

- Horizontal: $G_h = G_h(R, C_h)$
- Vertical: $G_v = G_v(R, C_v)$

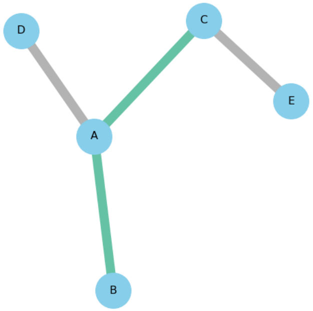
These relations apply to multiple levels: rooms, apartments, floors.

Rooms can be horizontally related to other rooms, and floors can be vertically related to other floors.

Node Reference Structure

The node structure represents the building as a network of nodes.

In the network below, imagine the vertices mapped to rooms in a house: A is the living room, D is the kitchen, B is the dining room, and C is a bedroom, where E is an attached bathroom.



https://python-graph-gallery.com/325-map-colour-to-the-edges-of-a-network/

Node Reference Structure

The relationships between nodes can be represented by their adjacency and connectivity relations.

- Adjacency: $A = (V, EA)$
- Connectivity: $C = (V, EC)$

These can be modeled in a single matrix by:

$$M = [m_{ij}]$$

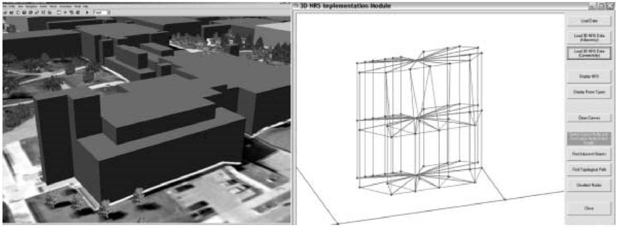
$$\begin{aligned} M = [m_{ij}] &= A + C \\ &= [a_{ij}] + [c_{ij}] \end{aligned}$$

where,

$$m_{ij} = \begin{cases} 1 \text{ if } n_i n_j \in EA \\ 2 \text{ if } n_i n_j \in (EA \cap EC) \\ 0 \text{ if } n_i n_j \notin EA \end{cases}$$

Data Model Visualization

The data model in [1] has been applied in a visualization program at Minnesota State University. A screenshot from the program is shown below:



Topological representation of Trenton East Hall at Minnesota State University at Mankato, MN [1]

The image on the right is the node structure associated with the building on the left.

Data Model Queries

The program could successfully:

- create a 3D model
- model the associated node structure
- process queries on room attributes
- process queries for adjacency relations

However, it did not implement an algorithm for finding the shortest path within a building.

Project Implementation

I created a program which finds the shortest path within a building model, since the research paper did not cover this step.

The program found at this github page uses the combinatorial methods from [1] to construct a building model, and finds the shortest path between two given vertices. The program:

- maps a given set of vertices
- draws the (given) connections between them
- calculates and draws the shortest path using Dijkstra's Algorithm

GitHub link: https://github.com/rakitch/Topological-Visualization-CDM

Parameters

The program takes in a text file as a parameter. In the text file, there will be:

- All vertices with name and coordinates (x, y, z)
- All connections between the given vertices

After passing the input file, the user will be prompted to enter the start vertex and the end vertex for the path to find.

Preconditions

The parameters must follow the given preconditions:

- The graph must be simple and connected.
- Vertex coordinate values can not exceed 10.
- Vertically connected vertices must be directly above/below each other.

These parameters are restrictive due to non-optimized memory handling, and would be addressed in future versions.

Algorithm

The program uses the following structures from [1] to construct the graph model:

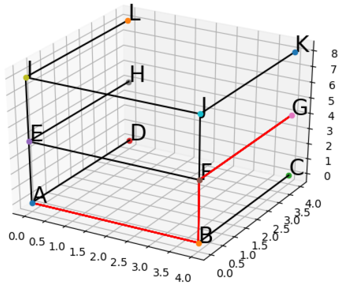
- Sets of horizontal and vertical relations
- Connectivity matrix

Using these structures, the program can run Dijkstra's algorithm to find the shortest path connecting two vertices.

Output

The outcome is visualized on a 3D graph which plots all vertices and all connections, with the calculated path highlighted in red.

An example is shown below:



Visualized shortest path from A to G.

Future Work

The aspects of the program which would be improved in future versions are listed below in order of importance.

1. Horizontal and vertical relations are calculated but not plotted. Future versions would include these relations as light gray lines.
2. The ceiling limit of 10 for coordinate values would be removed so vertices could have any value.
3. Inefficient use of memory should be improved and adaptable to wide ranges of values for the vertices.
4. Input format could be more user-friendly with a GUI that lets the user add vertices and connections easily.

Related Papers

Naumann, U., & Schenk, O. (2012). Combinatorial scientific computing. CRC Press.

Clauset, A., Moore, C., & Newman, M. (2008). Hierarchical structure and the prediction of missing links in networks. Nature Publishing Group, 453(7191), 98.

Gallier, J., & Quaintance, J. (2016). A Gentle Introduction to Homology, Cohomology, and Sheaf Cohomology. University of Pennsilvania Press.

Bibliography

Lee, J., & Kwan, M. P. (2005). A combinatorial data model for representing topological relations among 3D geographical features in microspatial environments. International Journal of Geographical Information Science, 19(10), 1039-1056.

Weisstein, Eric W. "Topology." From MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/Topology.html

Rowland, Todd. "Dual Vector Space." From MathWorld–A Wolfram Web Resource, created by Eric W. Weisstein. http://mathworld.wolfram.com/DualVectorSpace.html