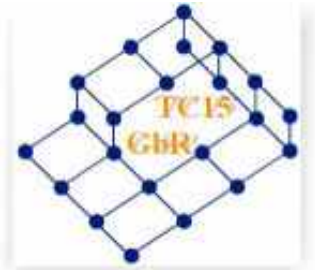
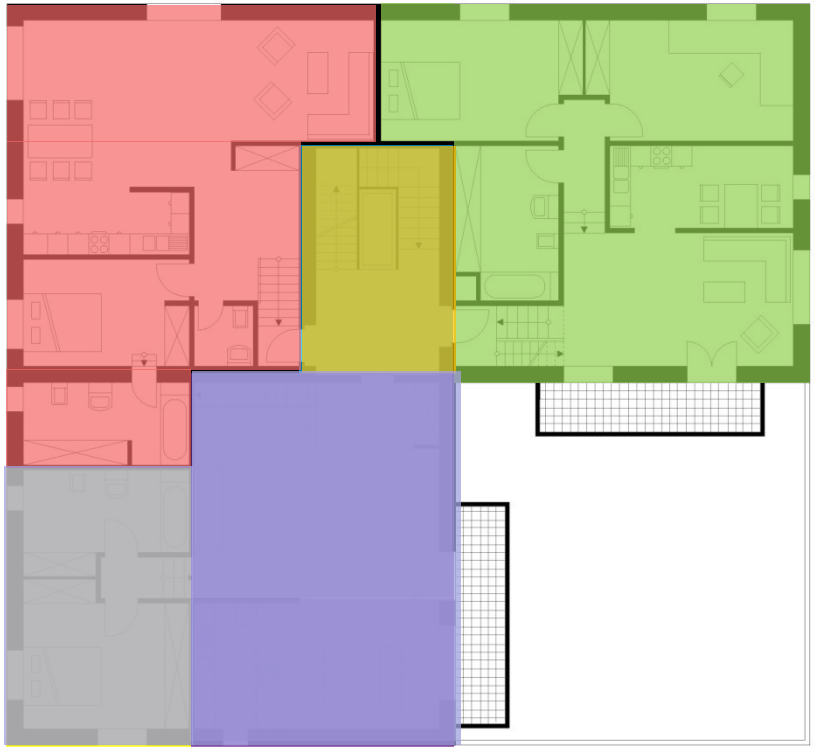
Indexing with Well-Founded Total Order for Faster Subgraph Isomorphism Detection

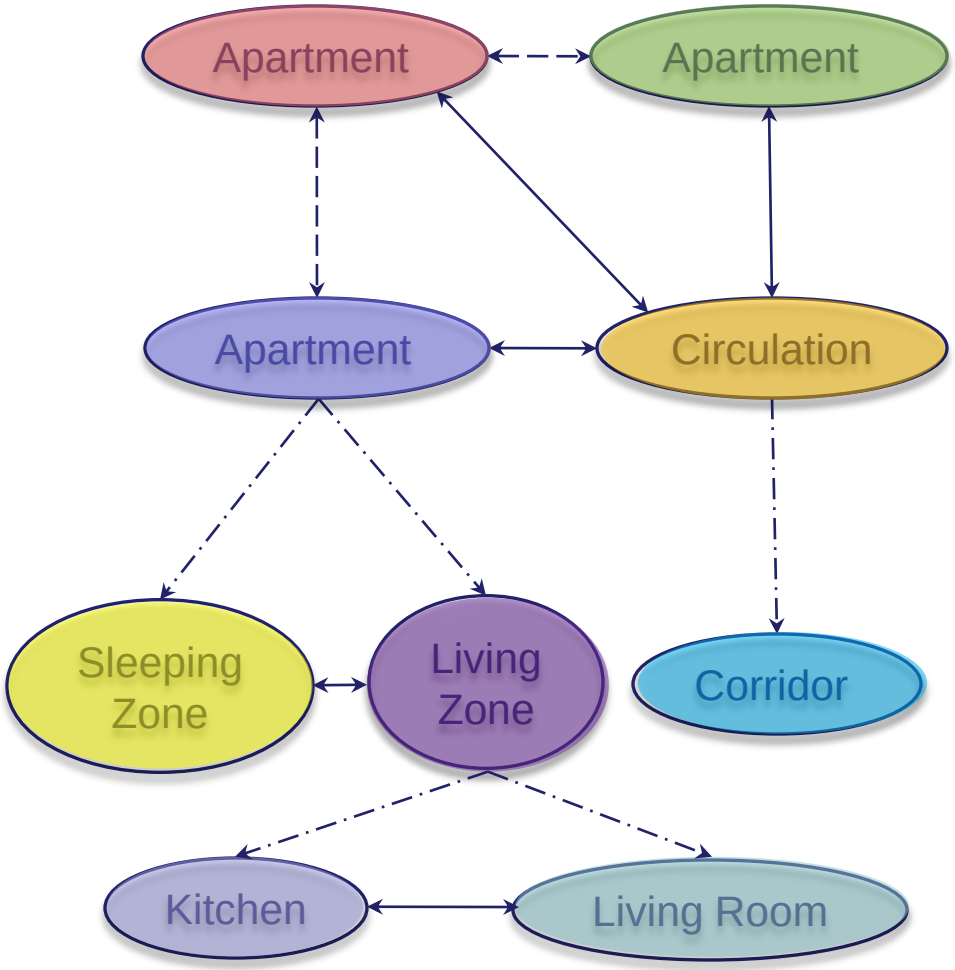


Markus Weber,
Marcus Eichenberger-Liwicki,
Andreas Dengel

http://www.dfki.de/km

Graph-Based Semantic Structure for Floor plans



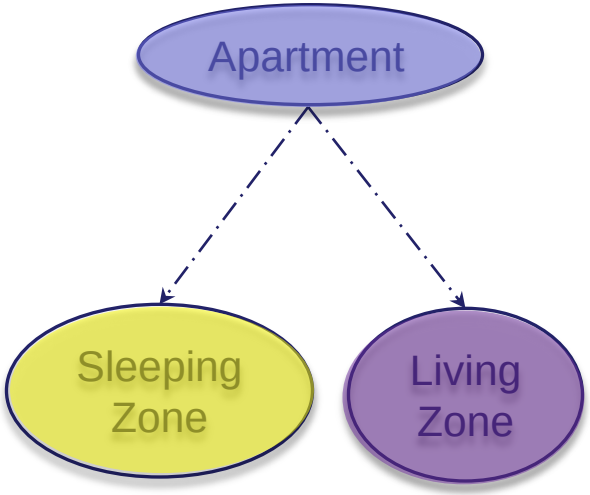


has part

direct connection

adjacent connection

The Intelligent Architect's Chest



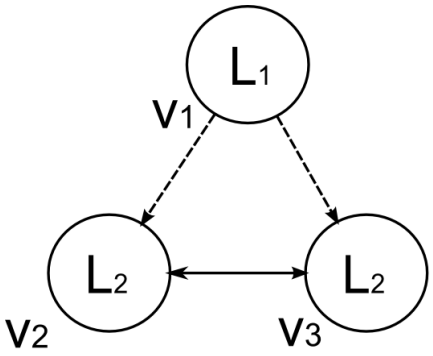
Basic Idea:

- Represent graph as adjacency matrix,
- compute all permutations of adjacency matrix,
- transform matrices into row-column vectors,
- arrange row-column vectors to a decision tree.

Reference:

Messmer, B. & Bunke, H., 1999**. A decision tree approach to graph and subgraph isomorphism detection**. *Pattern Recognition*, 32,1998

Building Index – Determine Permutations



	V1	V2	V3
V1	L1	1	1
V2	0	L2	2
V3	0	2	L2

A

	V1	V3	V2
V1	L1	1	1
V3	0	L2	2
V2	0	2	L2

B

	V2	V1	V3
V2	L2	0	2
V1	1	L1	1
V3	2	0	L2

C

	V2	V3	V1
V2	L2	2	0
V3	2	L2	0
V1	1	1	L1

D

	V3	V1	V2
V3	L2	0	2
V1	1	L1	1
V2	2	0	L2

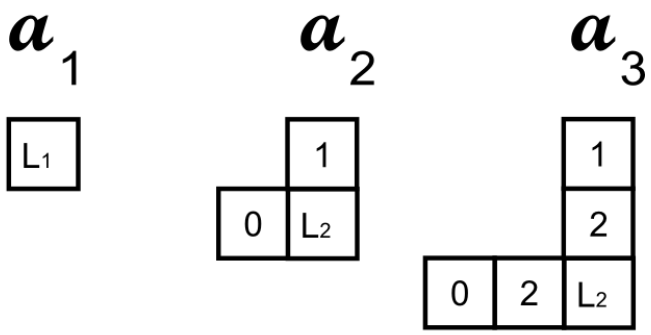
E

	V3	V2	V1
V3	L2	2	0
V2	2	L2	0
V1	1	1	L1

F

Building Index – Adjacency matrix to row-column vectors

	V1	V2	V3
V1	L1	1	1
V2	0	L2	2
V3	0	2	L2

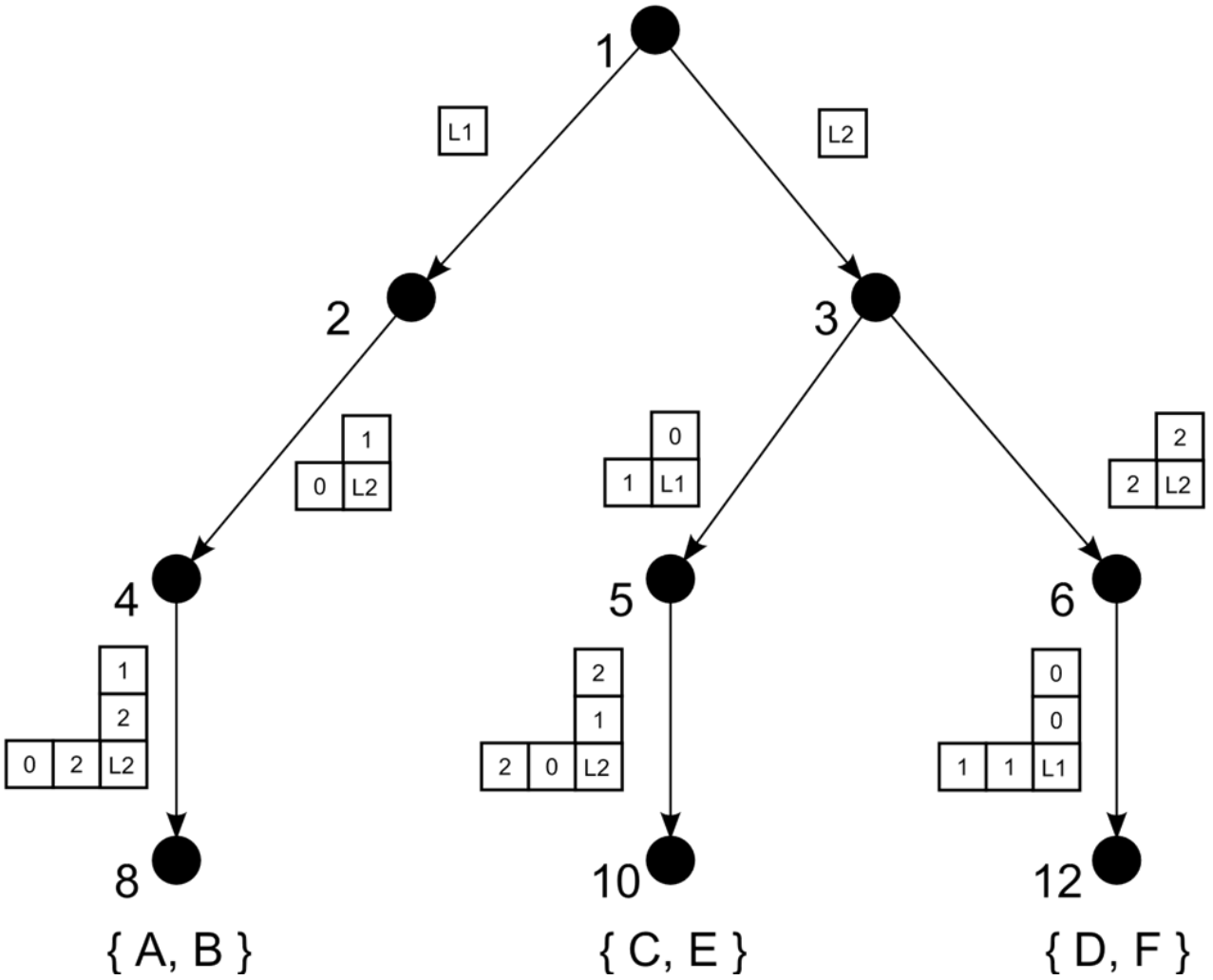


$$a_1 = (L_1)$$

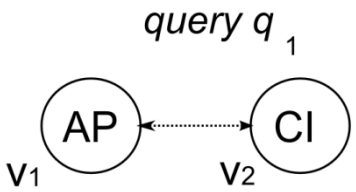
$$a_2 = (0, L_2, 1)$$

$$a_3 = (0, 2, L_2, 2, 1)$$

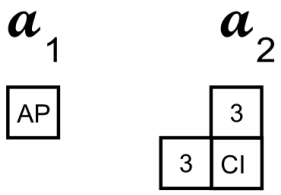
Building Index – Decision Tree



Subgraph Matching

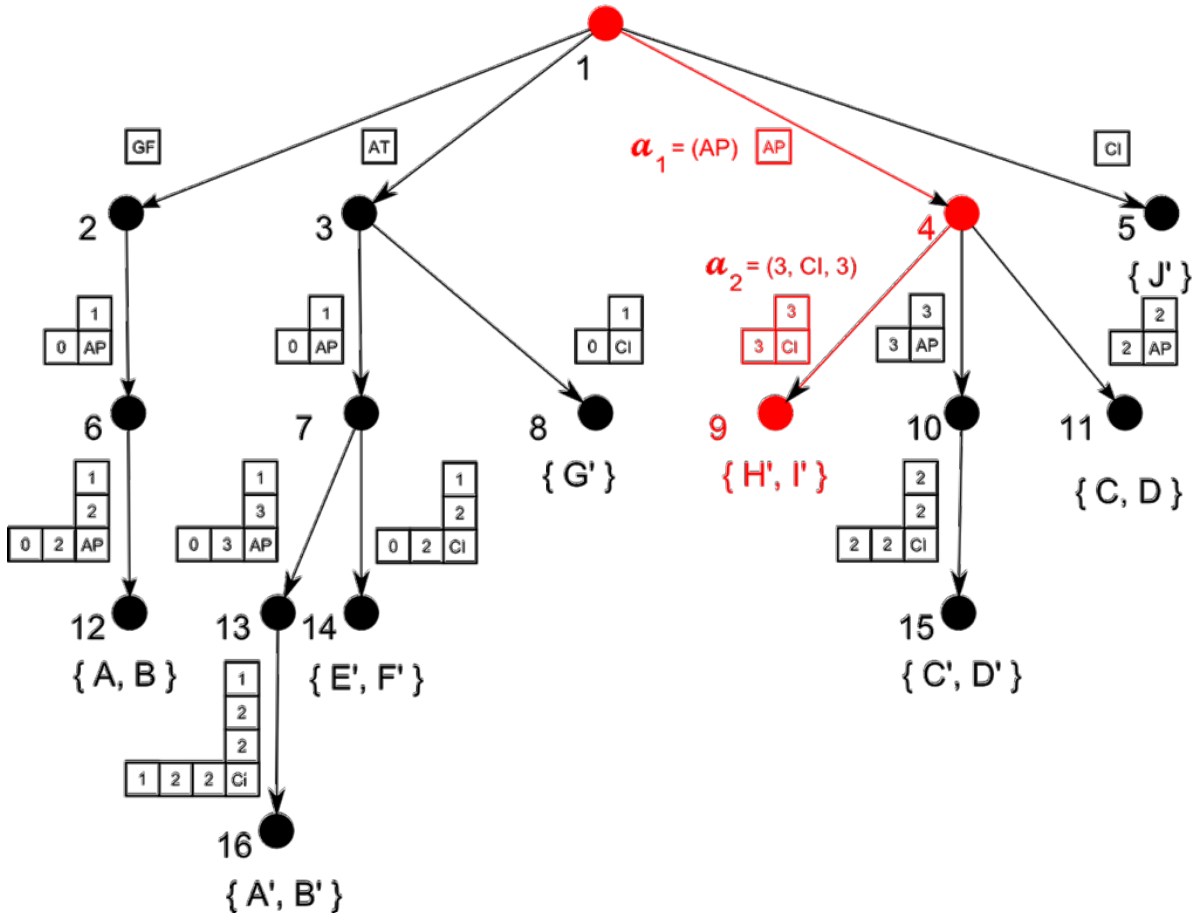


	V1	V2
V1	AP	3
V2	3	CI



a_1 = (AP)

a_2 = (3, CI, 3)



Permutations:

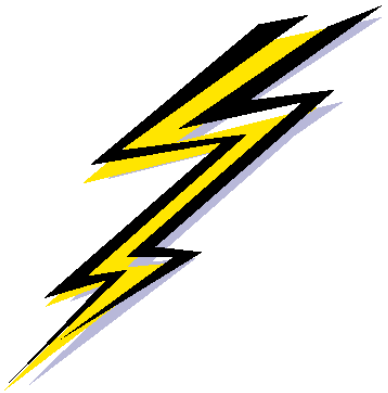
…

$18! = 6.4 \times 10^{15}$

$19! = 1.22 \times 10^{17}$

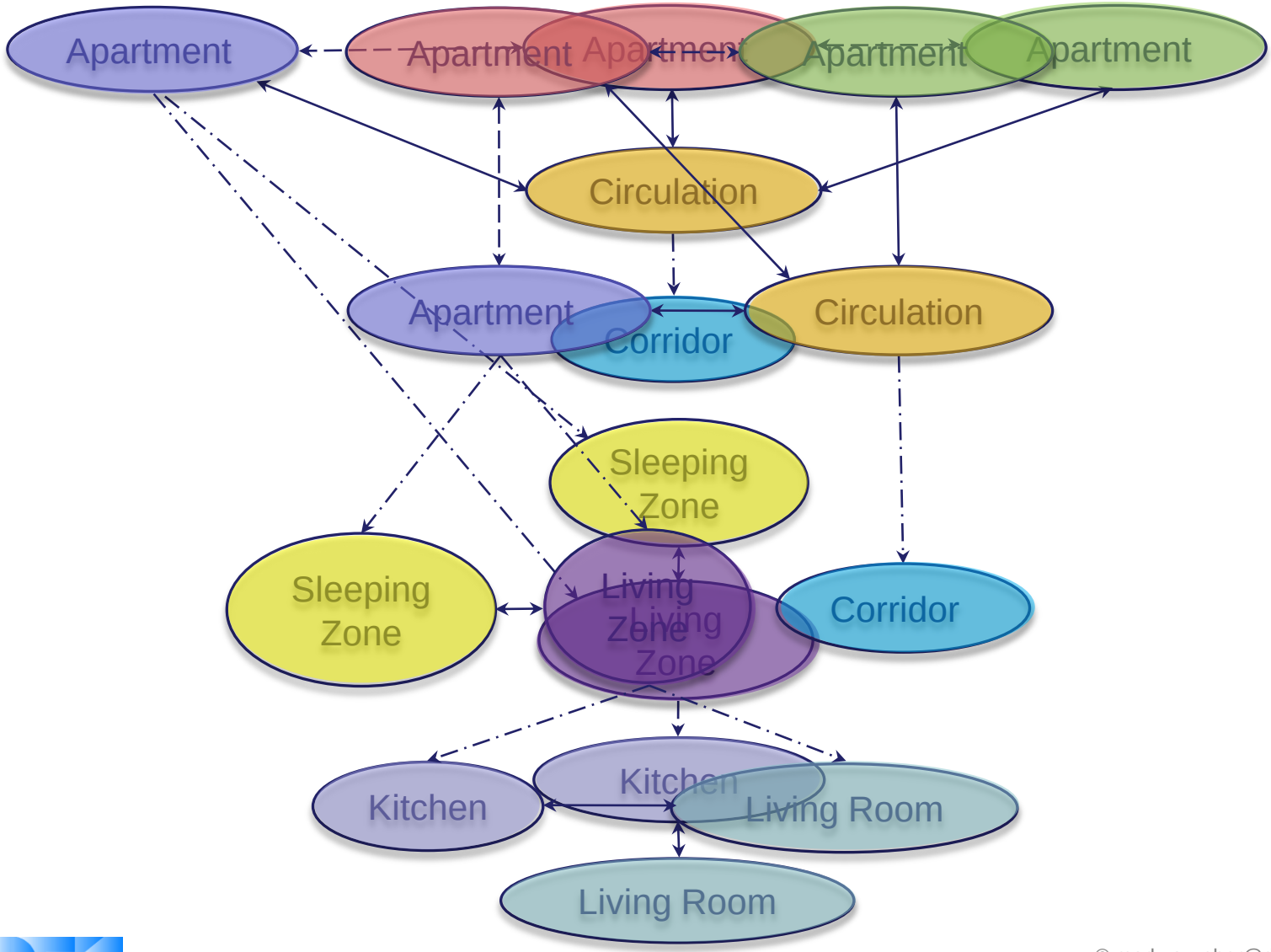
$20! = 2.43 \times 10^{18}$

….



Becomes intractable with graphs larger than 19 nodes!

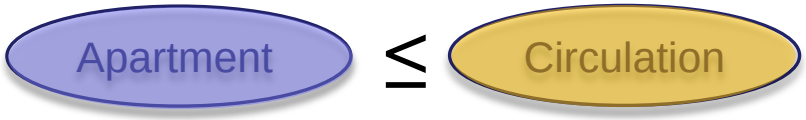
Basic Idea – Define an order for graph

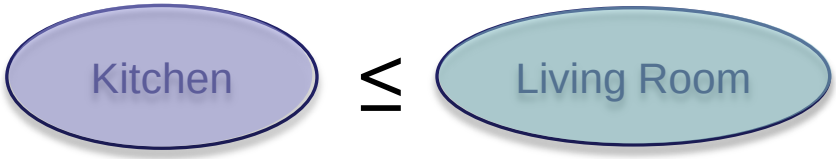


Definition – Total order

Definition 1 *A **total order** is a binary relation $\leq$ over a set P which is transitive, anti-symmetric, and total, thus for all a, b and c in P, it holds that:*

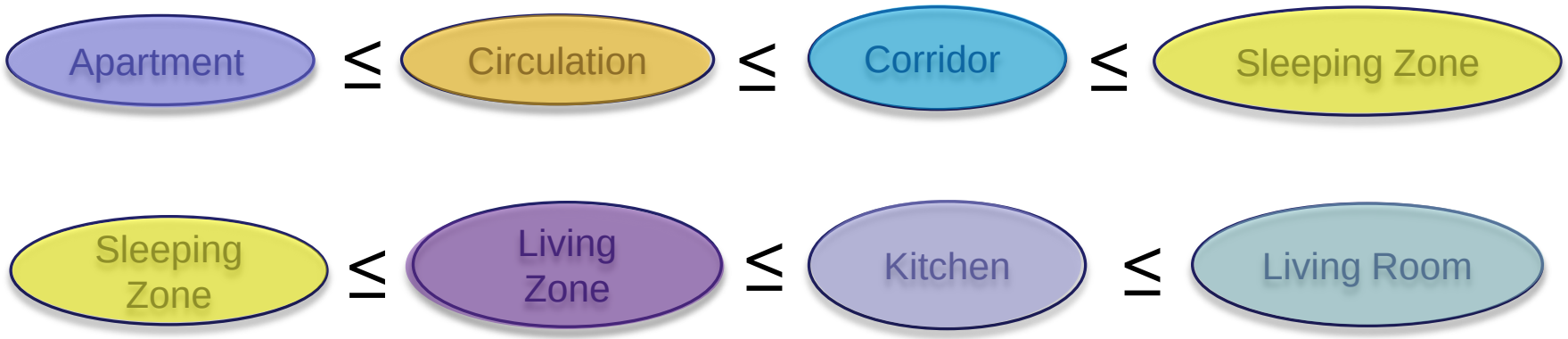
- *if $a \leq b$ and $b \leq a$ then $a = b$ (anti-symmetry);*
- *if $a \leq b$ and $b \leq c$ then $a \leq c$ (transitivity);*
- *$a \leq b$ or $b \leq a$ (totality).*





Definition – Well-founded order

Definition 2 *A partial or total order* $\leq$ *over a set* X *is* ***well-founded****,*
iff $(\forall\ Y \subseteq X\ :\ Y \neq \emptyset \rightarrow (\exists y \in Y\ :\ y$ *minimal in* Y *in respect of* $\leq))$.



Definition 3 *The* ***weight function*** σ *is defined as:* $\sigma : L_v \rightarrow \mathbb{N}$.

$$
\begin{aligned}
\sigma\,(\text{Apartment}) &= 1,\\
\sigma\,(\text{Circulation}) &= 2,\\
\sigma\,(\text{Corridor}) &= 3,\\
\sigma\,(\text{Sleeping Zone}) &= 4,\\
\sigma\,(\text{Living Zone}) &= 5,\\
\sigma\,(\text{Kitchen}) &= 6,\\
\sigma\,(\text{Living Room}) &= 7.
\end{aligned}
$$

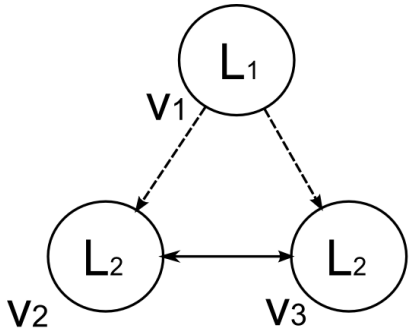
Definition 4 *A **labeled graph** consists of a 7-tuple,* $G = (V, E, L_v, L_e, \mu, \upsilon, \sigma)$, *where*

- V *is a set of vertices,*
- $E \subseteq V \times V$ *is a set of edges,*
- L_v *is a set of labels for the vertices,*
- L_e *is a set of labels for the edges,*
- $\mu : V \rightarrow L_v$ *is a function which assigns a label to the vertices,*
- $\upsilon : E \rightarrow L_e$ *is a function which assigns a label to the edges,*
- $\sigma : L_v \rightarrow \mathbb{N}$ *is a function which assigns a weight to the label of the vertices,*

and a binary relation $\leq$ *which defines a well-founded total order on the weights of the labels:*

$$\forall x, y \in L_v \ : \ \sigma(x) \leq \sigma(y) \ \vee \ \sigma(y) \leq \sigma(x)$$

Modification – Building Index



$$L_v = \{L_1, L_2\}$$
$$\sigma(L_1) = 1,$$
$$\sigma(L_2) = 2.$$

	V1	V2	V3
V1	L1	1	1
V2	0	L2	2
V3	0	2	L2

A

	V1	V3	V2
V1	L1	1	1
V3	0	L2	2
V2	0	2	L2

B

	V2	V1	V3
V2	L2	0	2
V1	1	L1	1
V3	2	0	L2

C

	V2	V3	V1
V2	L2	2	0
V3	2	L2	0
V1	1	1	L1

D

	V2	V3
V2	L2	2
V3	2	L2

C

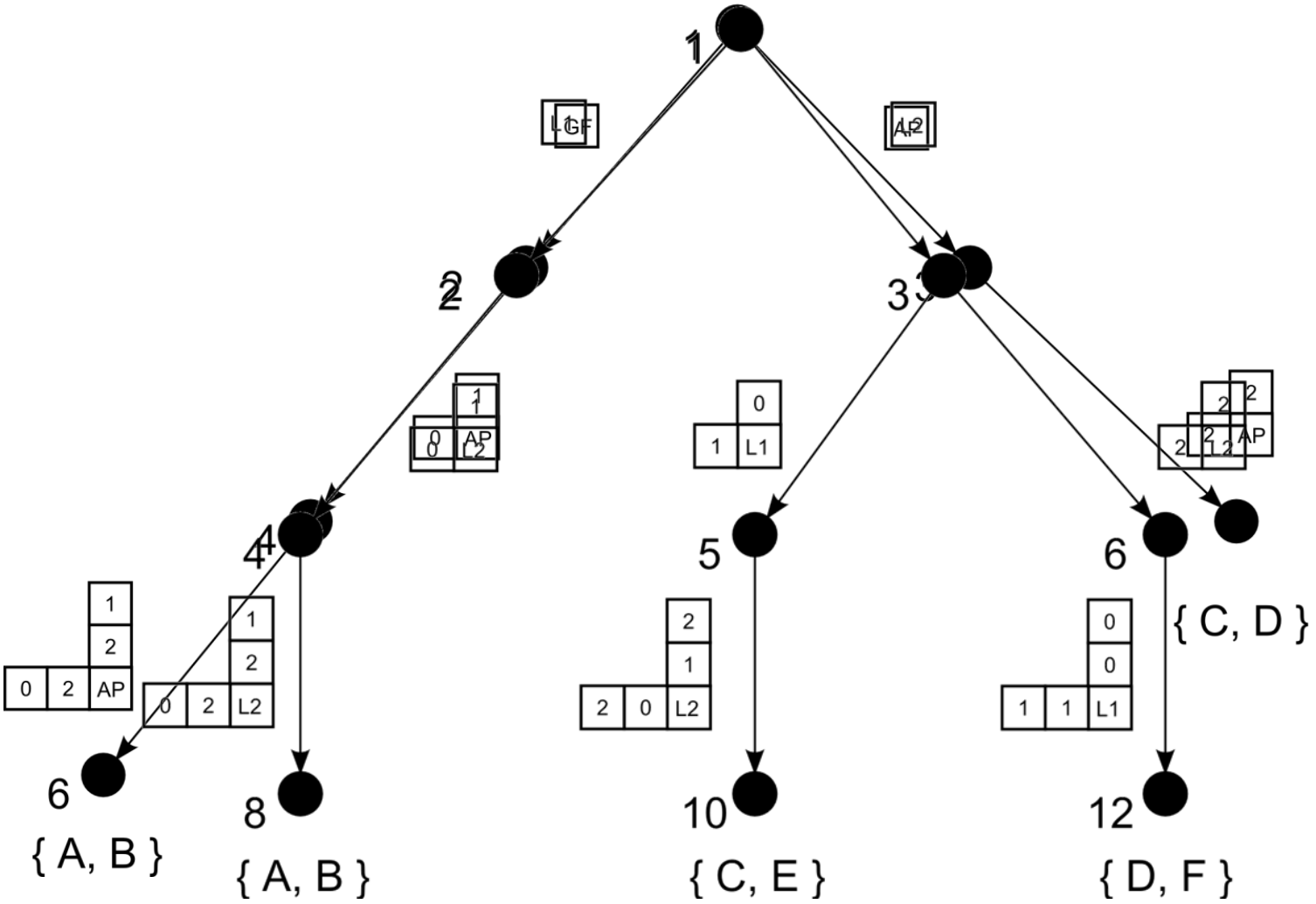
E

	V3	V2
V3	L2	2
V2	2	L2

D

F

Modification – Decision Tree



Algorithm 1 BUILD_INDEX($G = (V, E, L_v, L_e, \mu, \upsilon, \sigma)$, Tree)

Require: Unsorted set V of vertices, μ labeling function, σ weight function.

1: sort(V, L_v, μ, σ)

Ensure: Vertices V are sorted according to the defined order.

2: Let O be an empty list.
3: **for all** $l_i \in L_V$ **do**
4: Let interval $\{v_a, \ldots, v_b\}$ contain all v with $\mu(v) = l_i$
5: $O_i \leftarrow VARIATIONS(\{v_a, \ldots, v_b\})$
6: **end for**
7: Let $AG \leftarrow O_1 \times \ldots \times O_{|Lv|}$.
8: **for all** m_i in AG **do**
9: Add row column vector for sequence of m_i to Tree.
10: **end for**

Graph	Vertices #	Permutations (modified)	Permutations (original)	Same lables (max.)
1	17	3.26×10^{6}	3.55×10^{14}	5
2	21	3.59×10^{9}	5.10×10^{19}	8
3	17	1.08×10^{7}	3.55×10^{14}	5
4	20	2.50×10^{8}	2.43×10^{18}	6
5	24	1.64×10^{12}	6.20×10^{23}	10
6	17	1.63×10^{6}	3.55×10^{14}	3
7	21	2.04×10^{7}	5.10×10^{19}	3
8	30	1.39×10^{12}	2.65×10^{32}	5
9	22	8.01×10^{8}	1.12×10^{21}	6
⋮	⋮	⋮	⋮	⋮
100	23	1.00×10^{9}	2.58×10^{22}	6
∅	23.05	1.09×10^{13}	3.73×10^{31}	5.23

- Depending on the number of same labels, graphs up to 30 vertices can be handled,
- Decision Tree contains less nodes, thus less memory consumption,
- Complexity analysis, proof of completeness and pseudo code in publication.

$$\mathcal{O}(|V|!) \quad vs. \quad \mathcal{O}(((n_{max}+1)!)^{|L_v|})$$

References:

Weber, M. et al., 2010**. a.SCatch: Semantic Structure for Architectural Floor Plan Retrieval**.
Advances in Case-Based Reasoning, Proc. of ICCBR 2010.

Weber, M., Liwicki, M. & Dengel, A., 2010**. a.SCAtch - A Sketch-Based Retrieval for Architectural Floor Plans.**
In *12th International Conference on Frontiers of Handwriting Recognition.*

Ahmed S., Liwicki M., Weber M. & Dengel A., 2011. **Improved Automatic Analysis of Architectural Floor Plans**.
11th International Conference on Document Analysis and Recognition (ICDAR 2011) **SUBMITTED**

Future Work

- Work on document analysis system to automatically extract graph from floor plans,
- Test algorithm on real data,
- Open-Source C++ implementation of method in progress.

a.SCatch System

Do you have any questions?