Using Multi-Level Graphs for Timetable Information

Frank Schulz, Dorothea Wagner, and Christos Zaroliagis

Overview

1. Introduction
   - ▷ Timetable Information - A Shortest Path Problem
2. Multi-Level Graphs
   - ▷ Speed-Up Technique for Shortest Path Algorithms
3. Timetable Information Graphs
4. Experiments
5. Conclusion & Outlook

Overview

Application Scenario

Timetable Information System

- Large timetable
  (e.g., 500.000 departures, 7.000 stations)
- Central server
  (e.g., 100 on-line queries per second)

→ Fast Algorithm

Simple Queries

- Input: departure and arrival station, departure time
- Output: train connection with earliest arrival time

Shortest Path Problem

Graph model

- Solving a query $\Leftrightarrow$ finding a shortest path

Speed-up techniques needed

- Commercial products
  - ▷ Heuristics that don't guarantee optimality
- Scientific work
  - ▷ Geometric techniques
  - ▷ Hierarchical graph decomposition

Contribution

Multi-Level Graph Approach

- Hierarchical graph decomposition technique
- For general digraphs
- Idea
  - ▷ Preprocessing: Construct multiple levels of additional edges
  - ▷ On-Line Phase: Compute shortest paths in small subgraphs

Experimental Evaluation

- For the given application scenario
- With real data

Overview

1. Introduction √
2. Multi-Level Graphs
3. Timetable Information Graphs
4. Experiments
5. Conclusion & Outlook

Multi-Level Graphs

Given

- a weighted digraph $G = (V, E)$
- a sequence of subsets of V

$$V \supset S_1 \supset \ldots \supset S_l$$

Outline

- Construct l levels of additional edges $\rightarrow \mathcal{M}(G)$
- Component Tree
- Define subgraph of $\mathcal{M}(G)$ for a pair $s, t \in V$
- Use subgraph to compute s-t shortest path

Multi-Level Graphs

Given

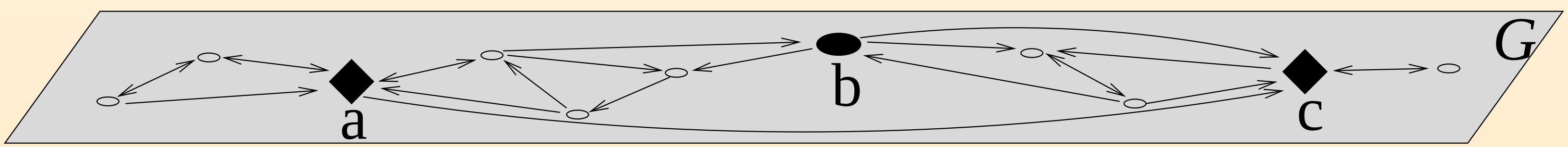
- a weighted digraph $G = (V, E)$
- a sequence of subsets of V

$$V \supset S_1 \supset \ldots \supset S_l$$

Outline

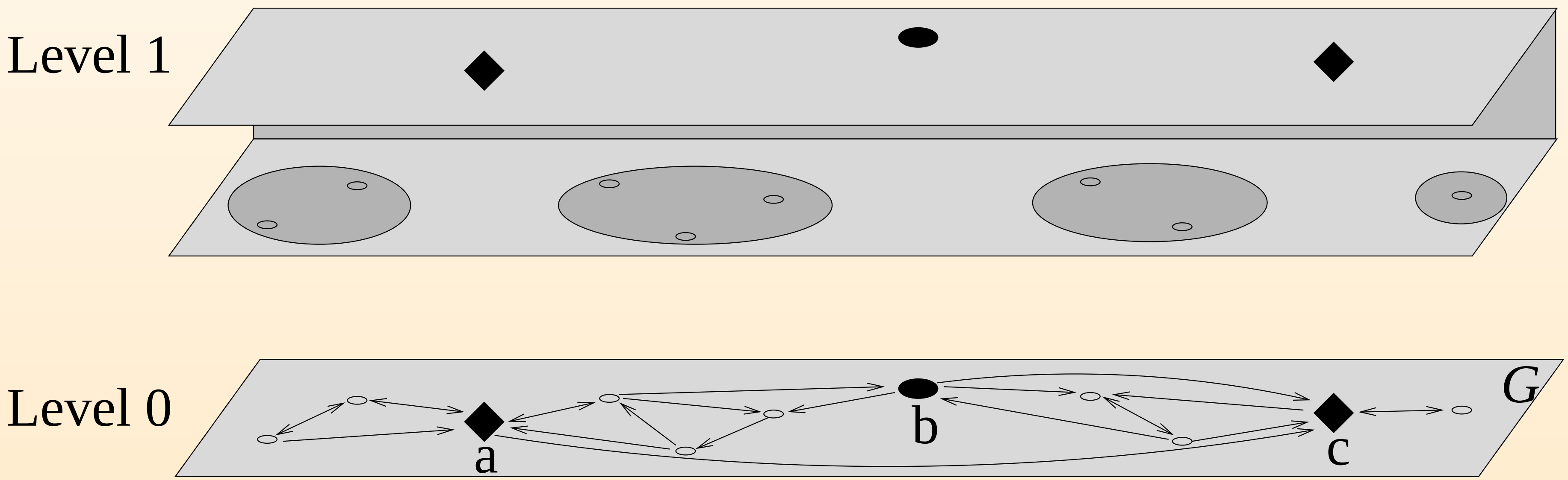
- Construct l levels of additional edges $\rightarrow \mathcal{M}(G)$
- Component Tree
- Define subgraph of $\mathcal{M}(G)$ for a pair $s, t \in V$
- Use subgraph to compute s-t shortest path

Level Construction



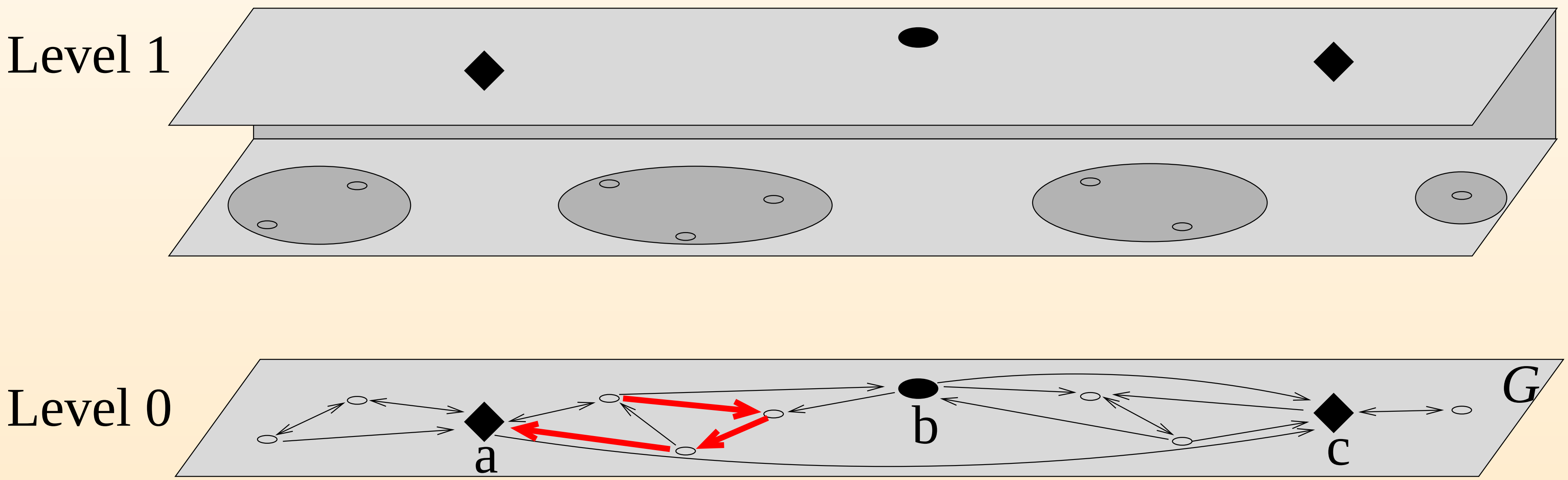
$$G = (V, E),\ S_1 = \{a, b, c\},\ S_2 = \{a, c\}$$

Level Construction



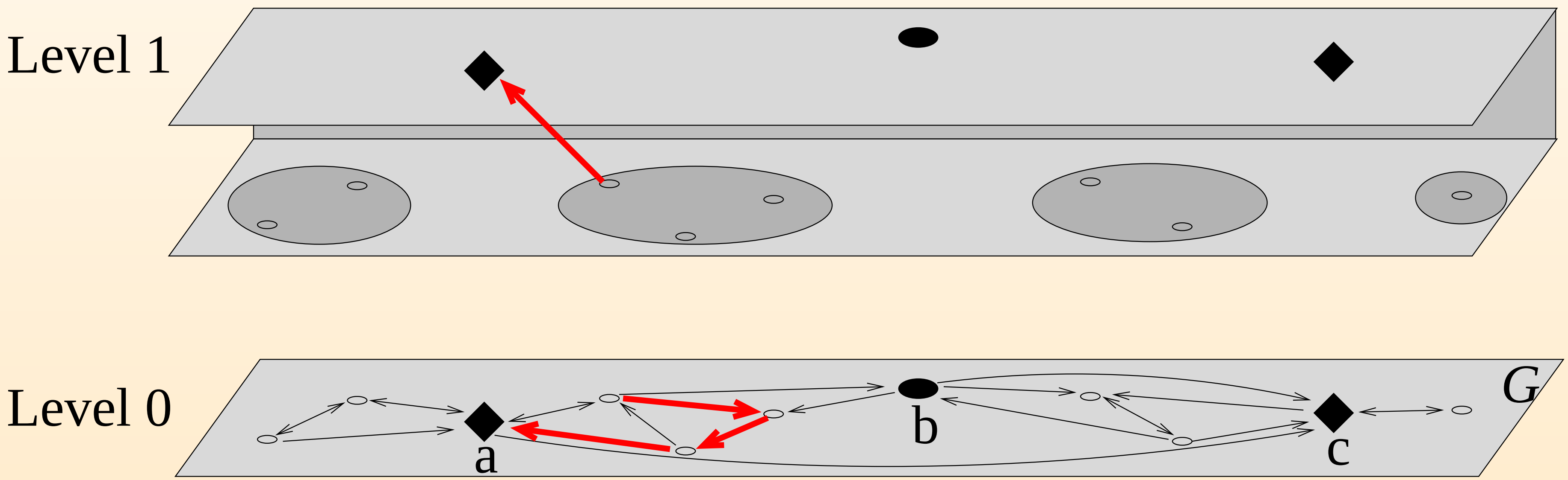
Connected Components in $G - S_1$

Level Construction



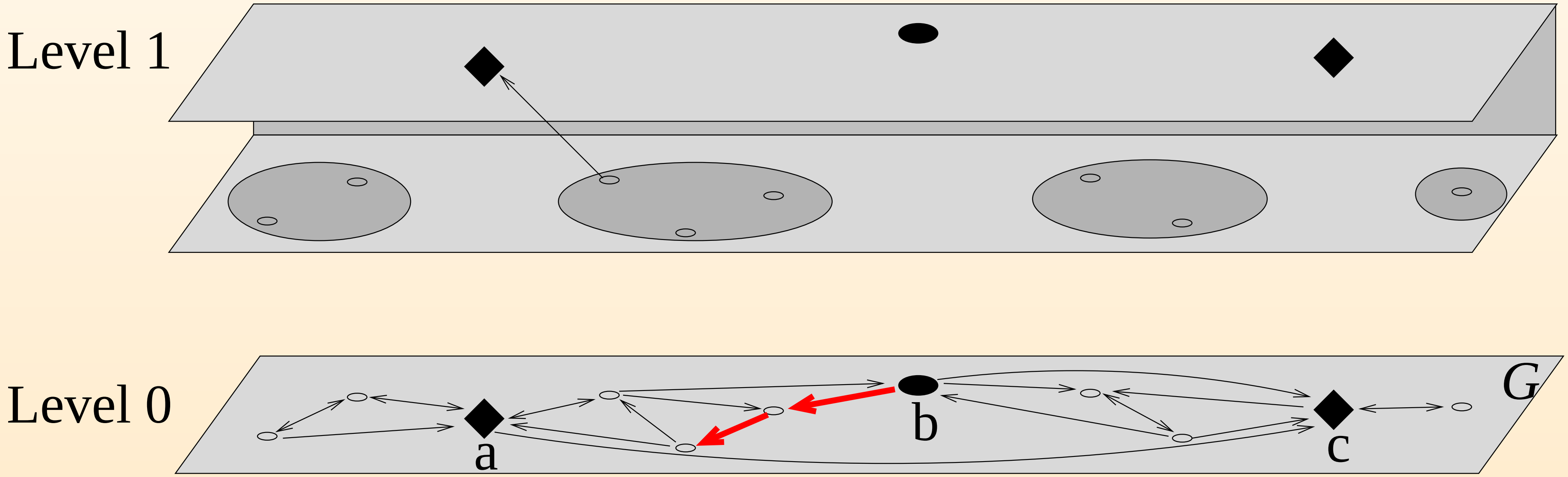
No internal vertex of path belongs to S_1

Level Construction



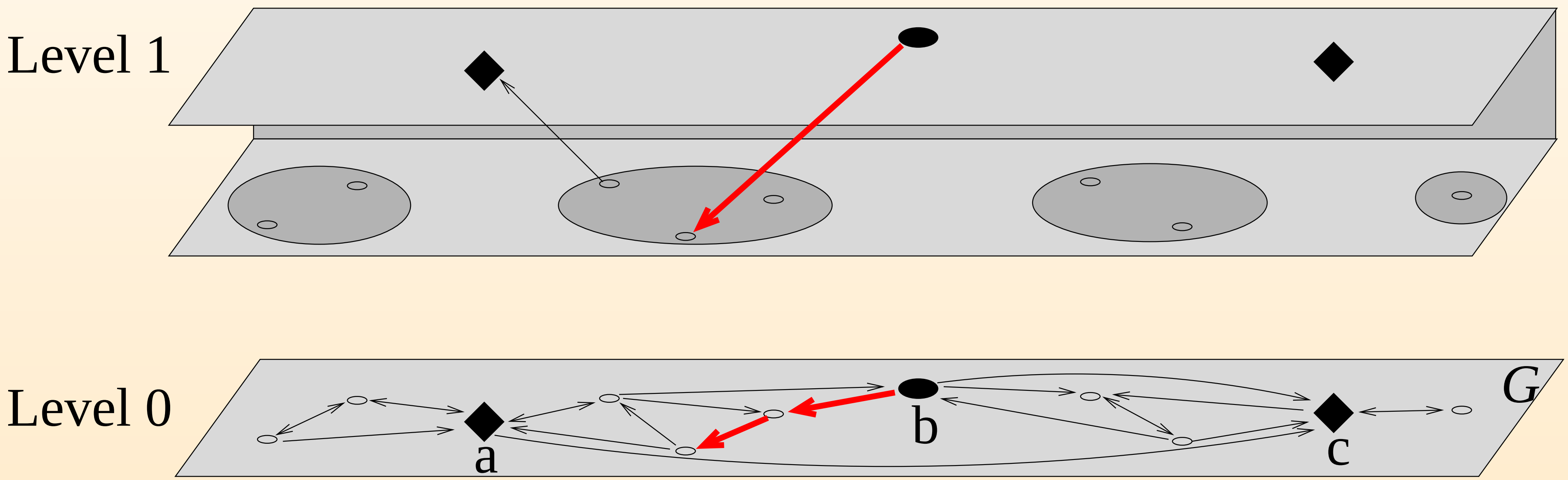
$\Rightarrow$ Edge in level 1

Level Construction



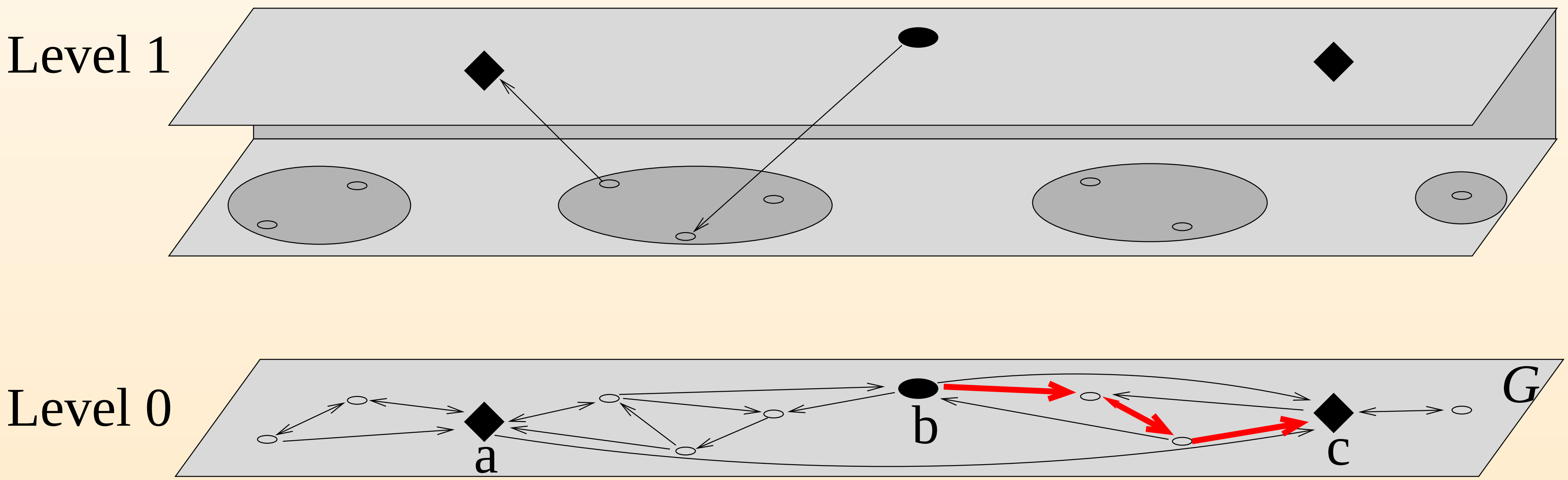
No internal vertex of path belongs to S_1

Level Construction



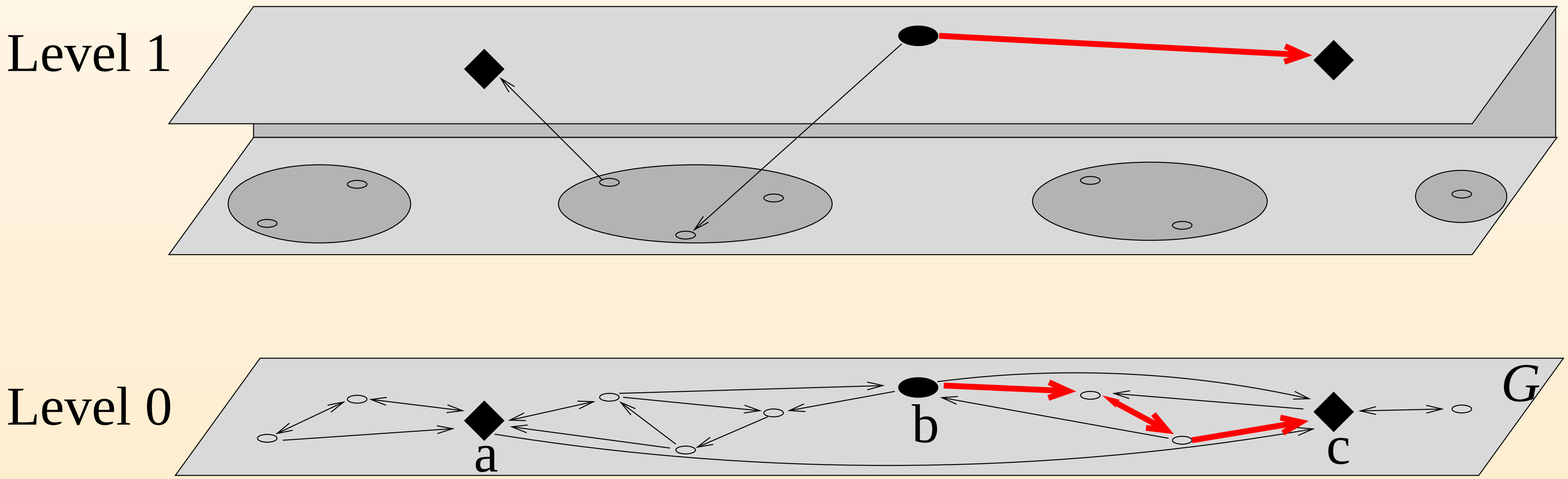
$\Rightarrow$ Edge in level 1

Level Construction



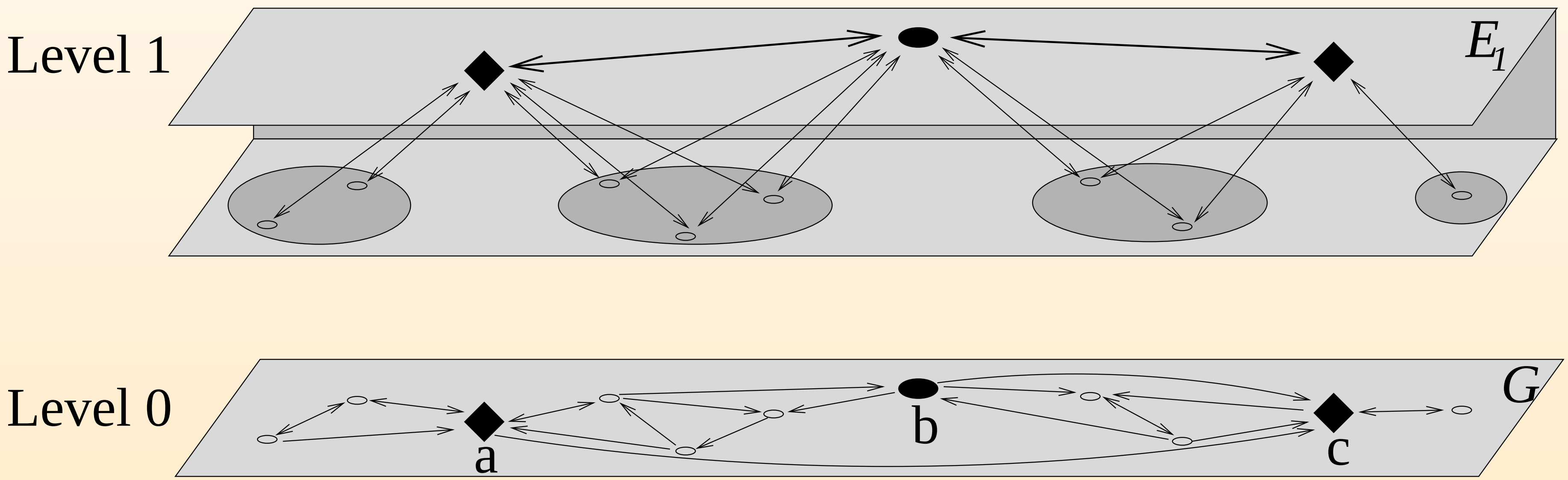
No internal vertex of path belongs to S_1

Level Construction



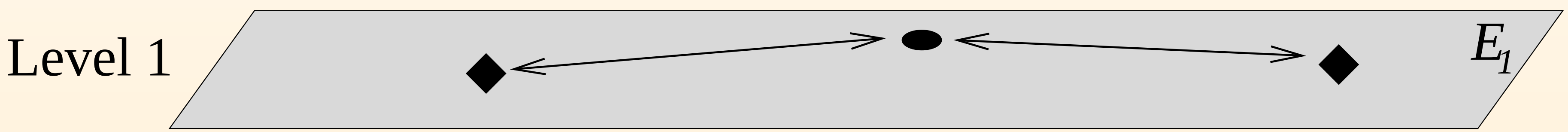
⇒ Edge in level 1

Level Construction



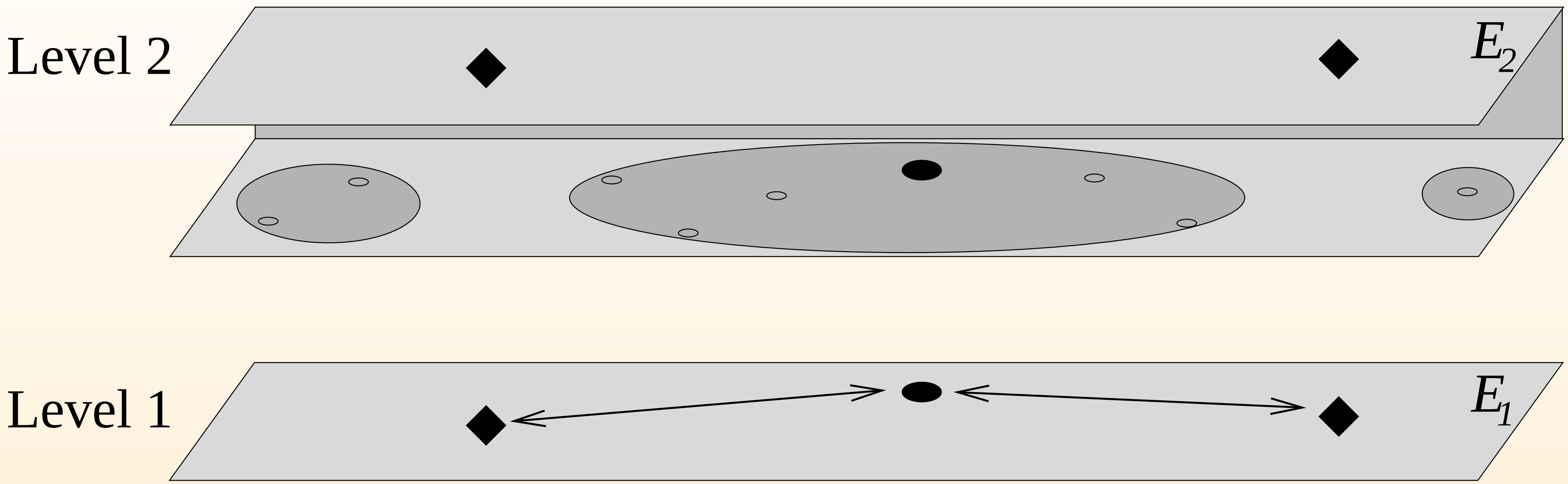
All edges in level 1

Level Construction



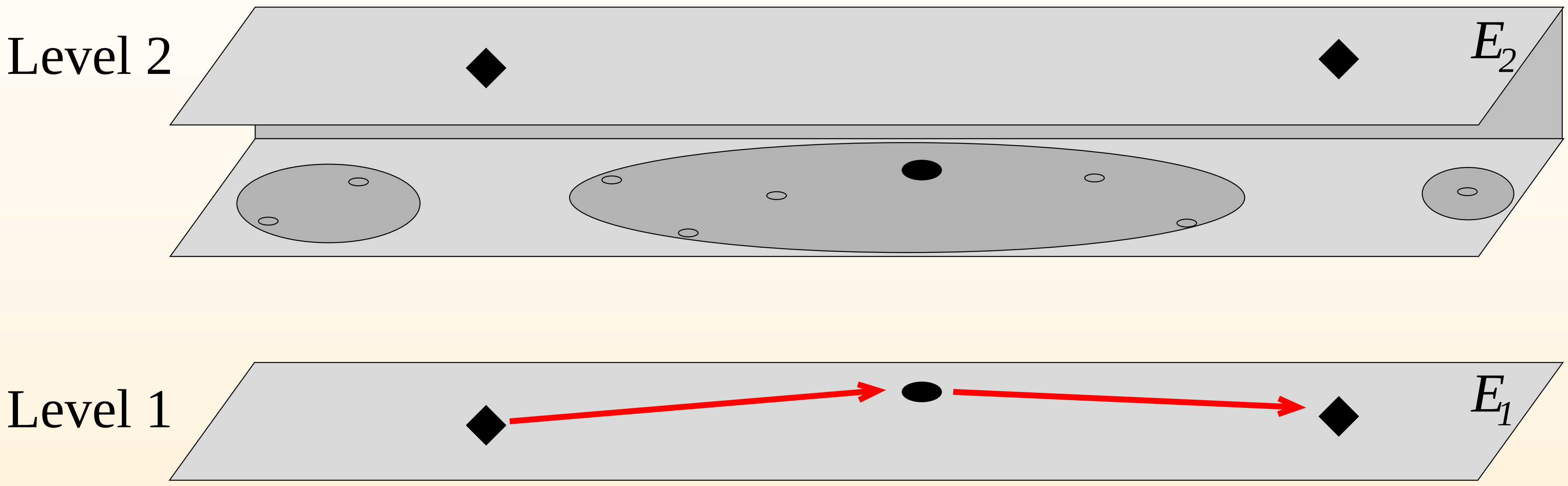
Consider now iteratively graph (S_1, E_1)

Level Construction



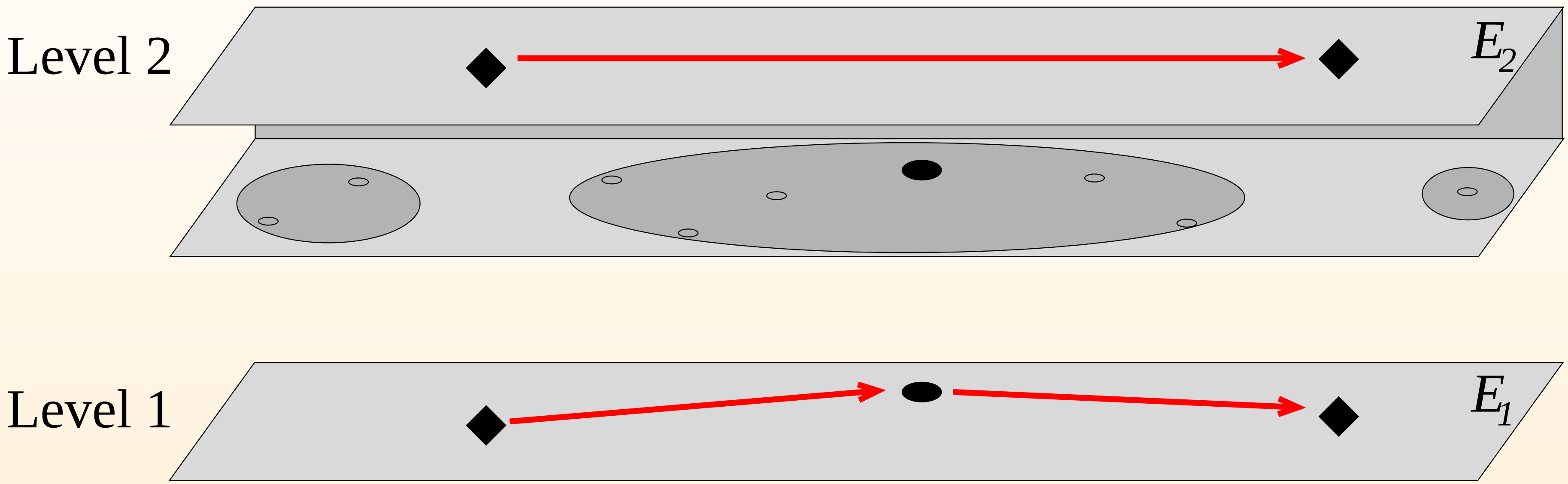
Connected components in $G - S_2$

Level Construction



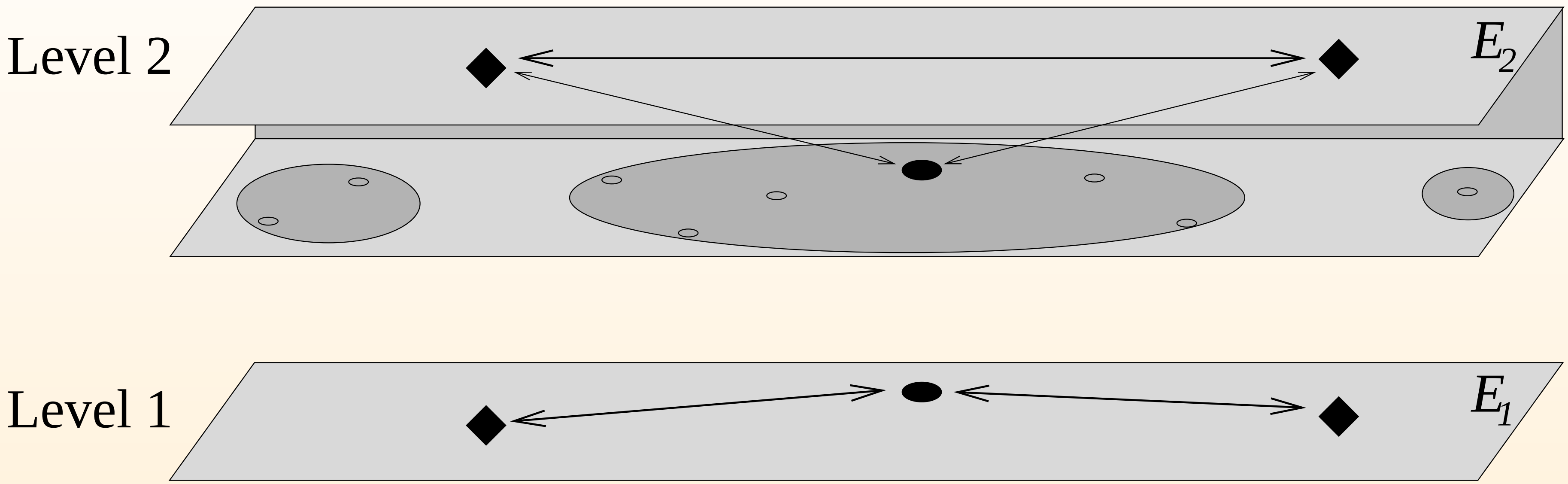
No internal vertex of path belongs to S_2

Level Construction



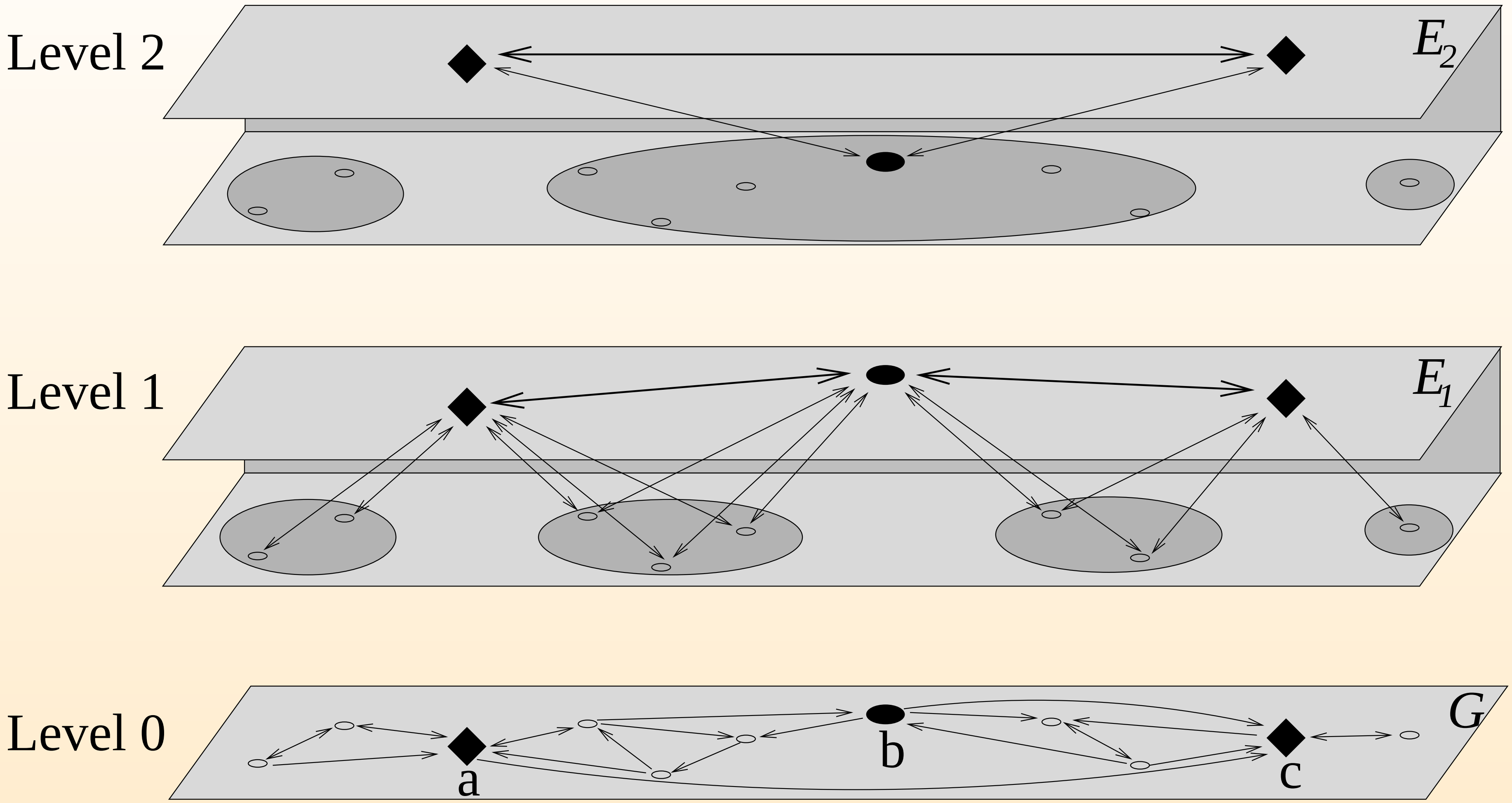
$\Rightarrow$ Edge in level 2

Level Construction



All edges in level 2

Level Construction



3-Level Graph $\mathcal{M}(G, S_1, S_2)$

Multi-Level Graphs

Given

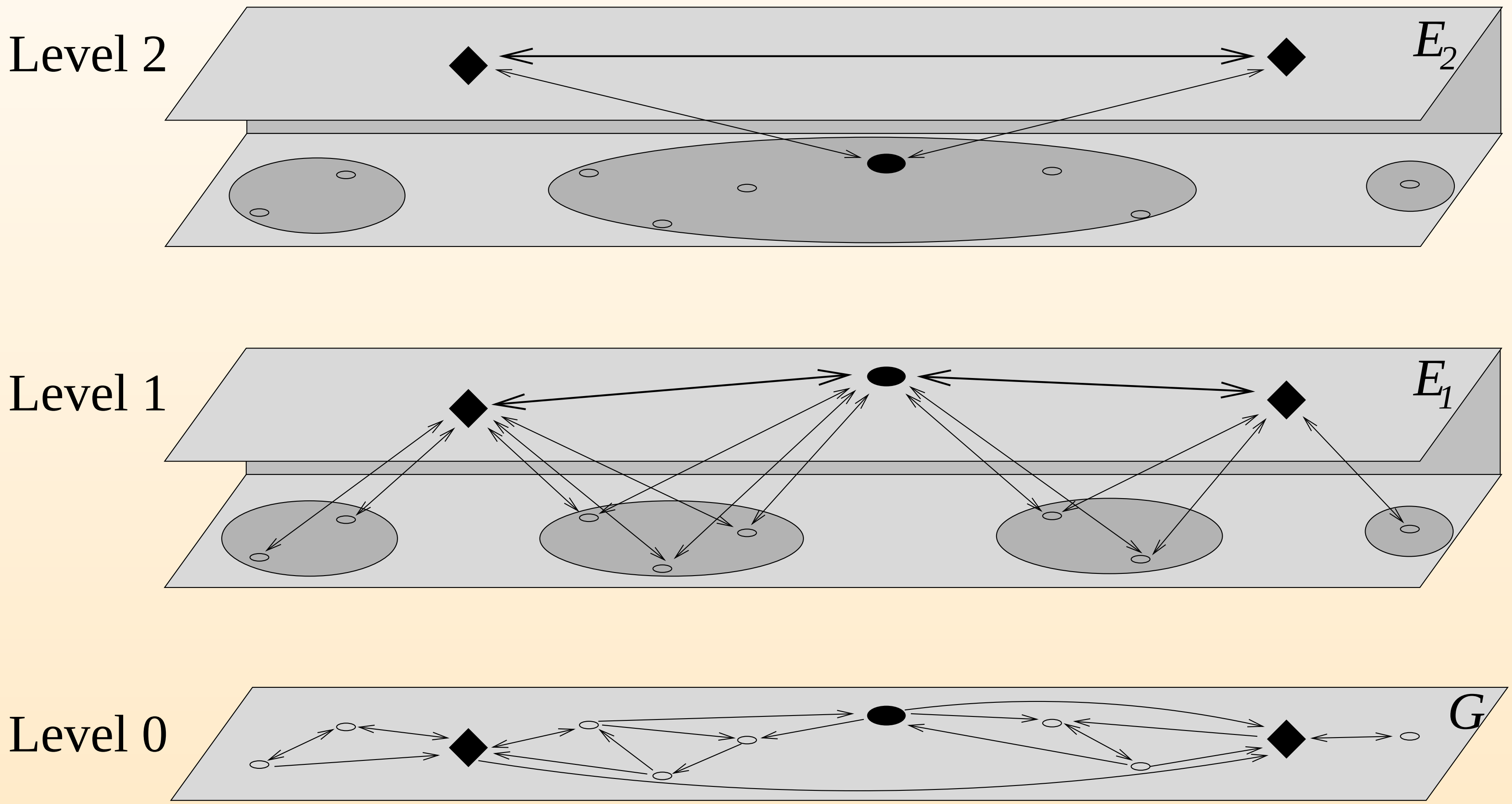
- a weighted digraph $G = (V, E)$
- a sequence of subsets of V

$$V \supset S_1 \supset \ldots \supset S_l$$

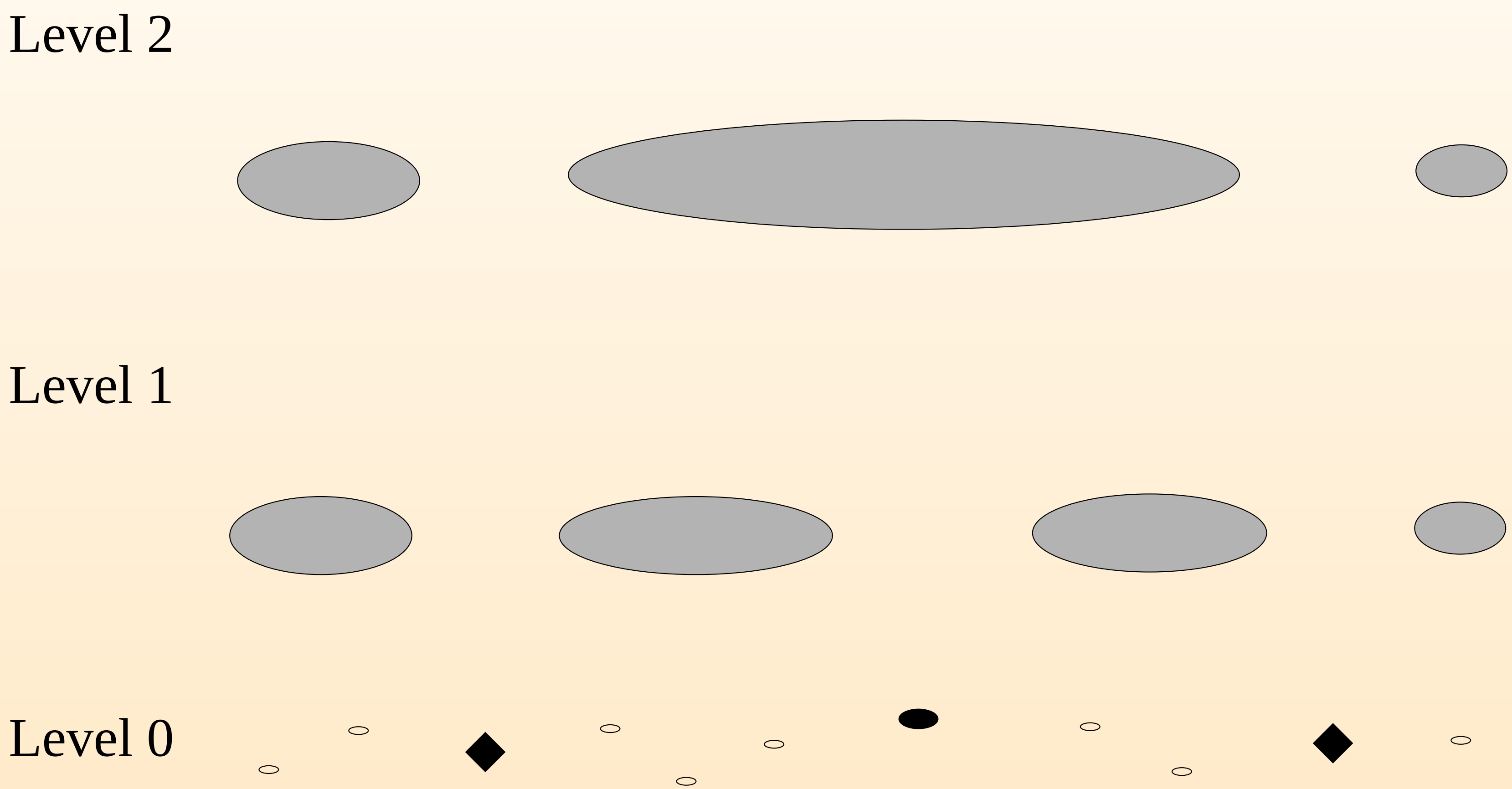
Outline

- Construct l levels of additional edges $\longrightarrow \mathcal{M}(G)$ $\surd$
- **Component Tree**
- Define subgraph of $\mathcal{M}(G)$ for a pair $s, t \in V$
- Use subgraph to compute s-t shortest path

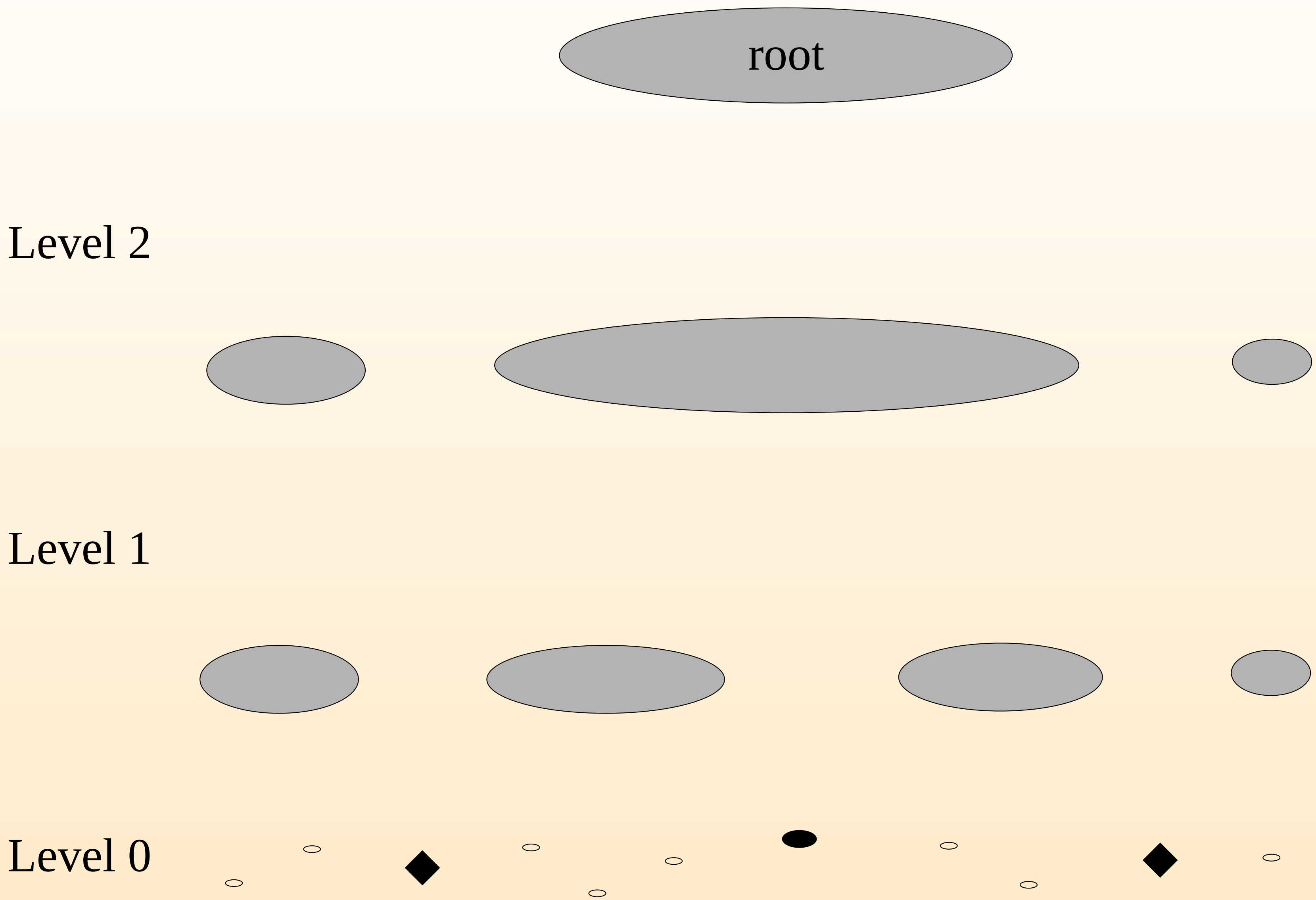
Component Tree



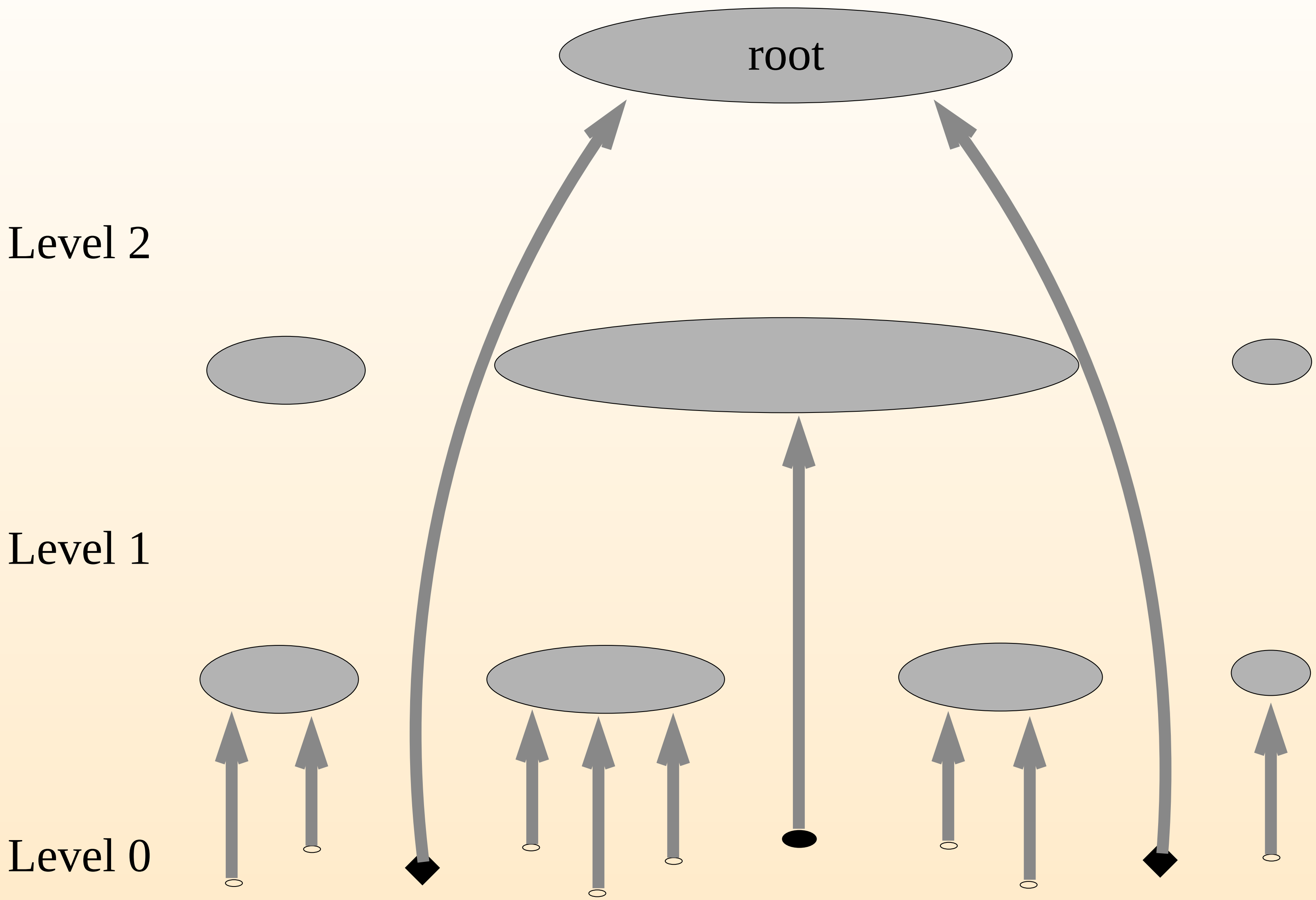
Component Tree



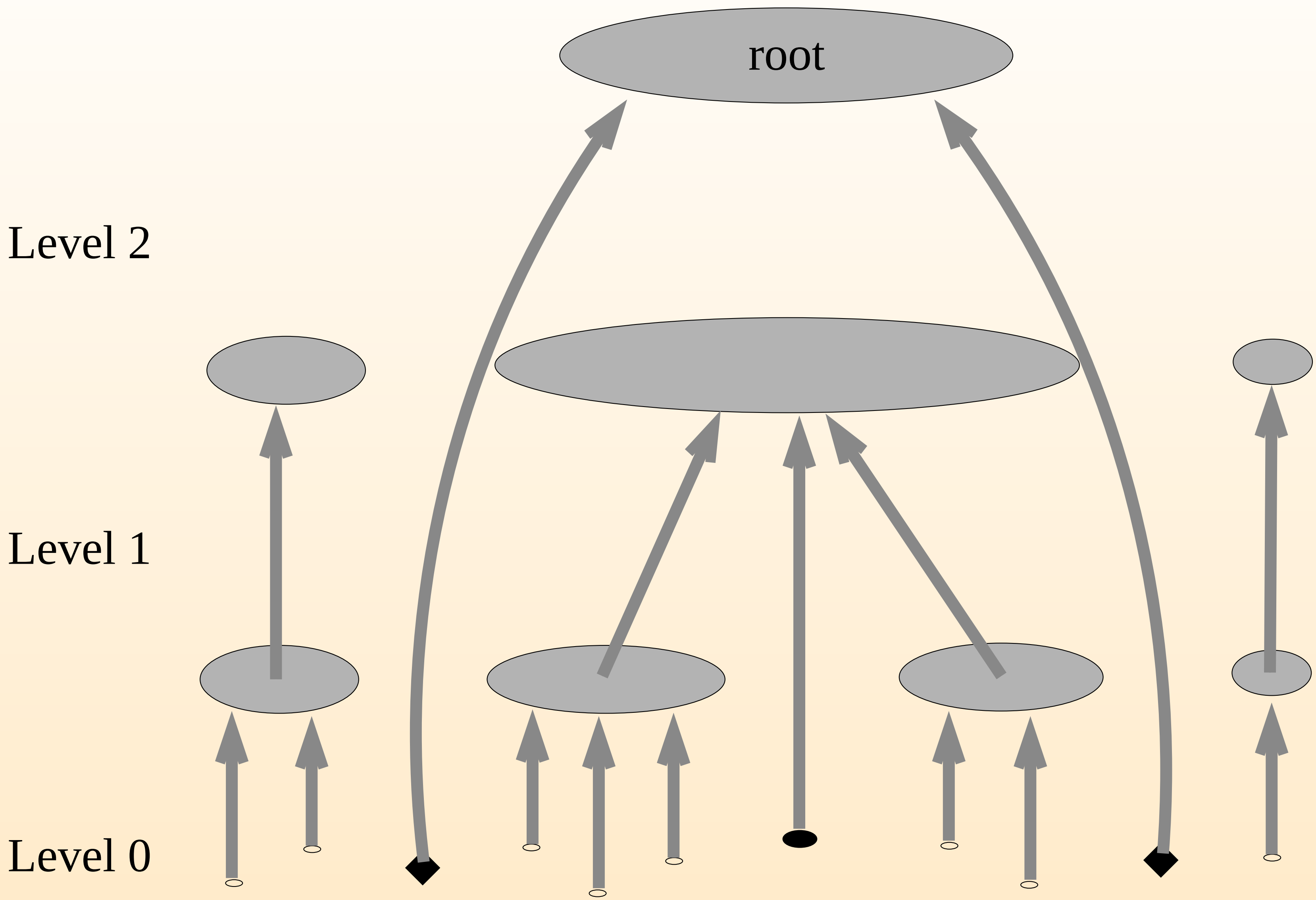
Component Tree



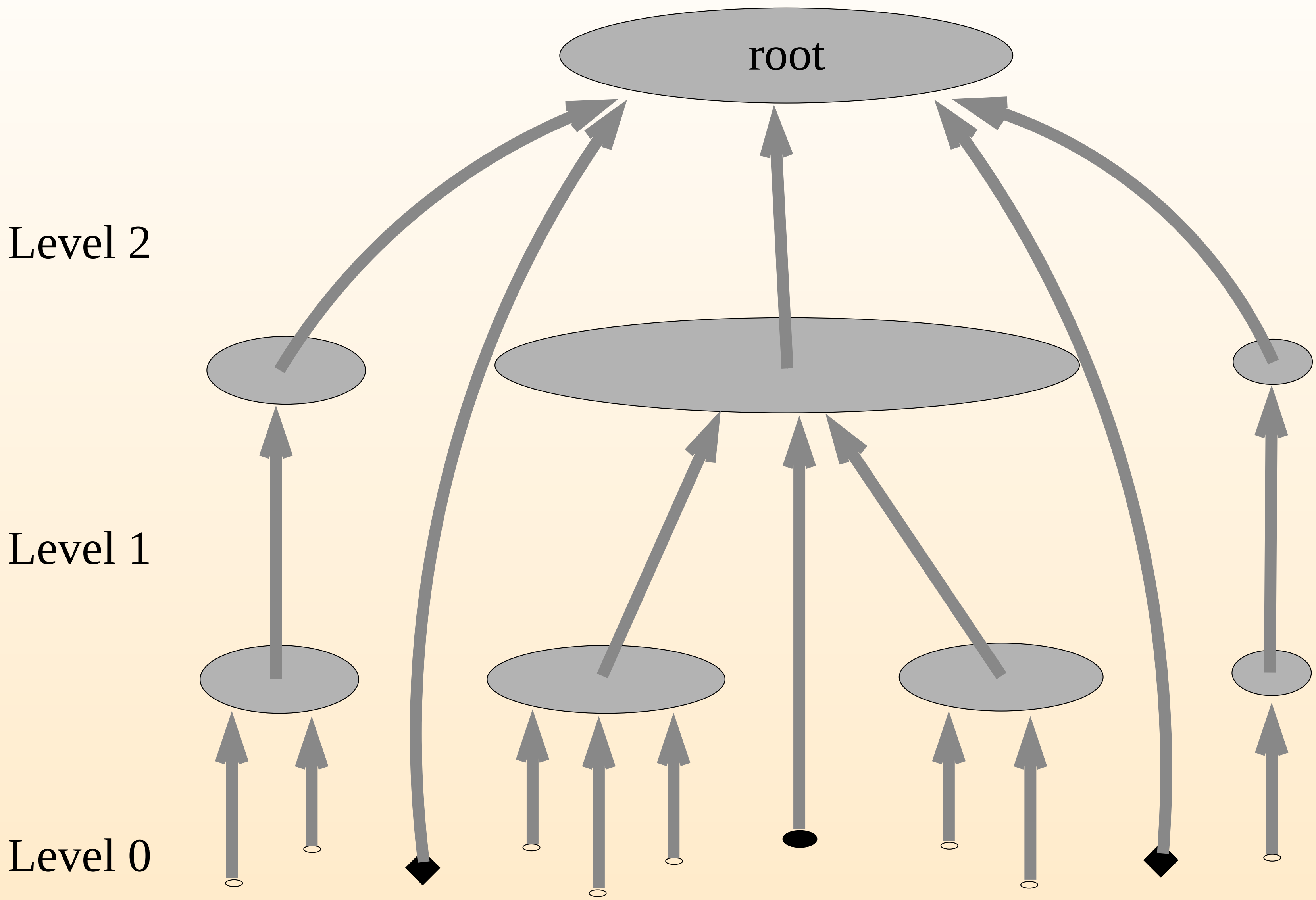
Component Tree



Component Tree



Component Tree



Multi-Level Graphs

Given

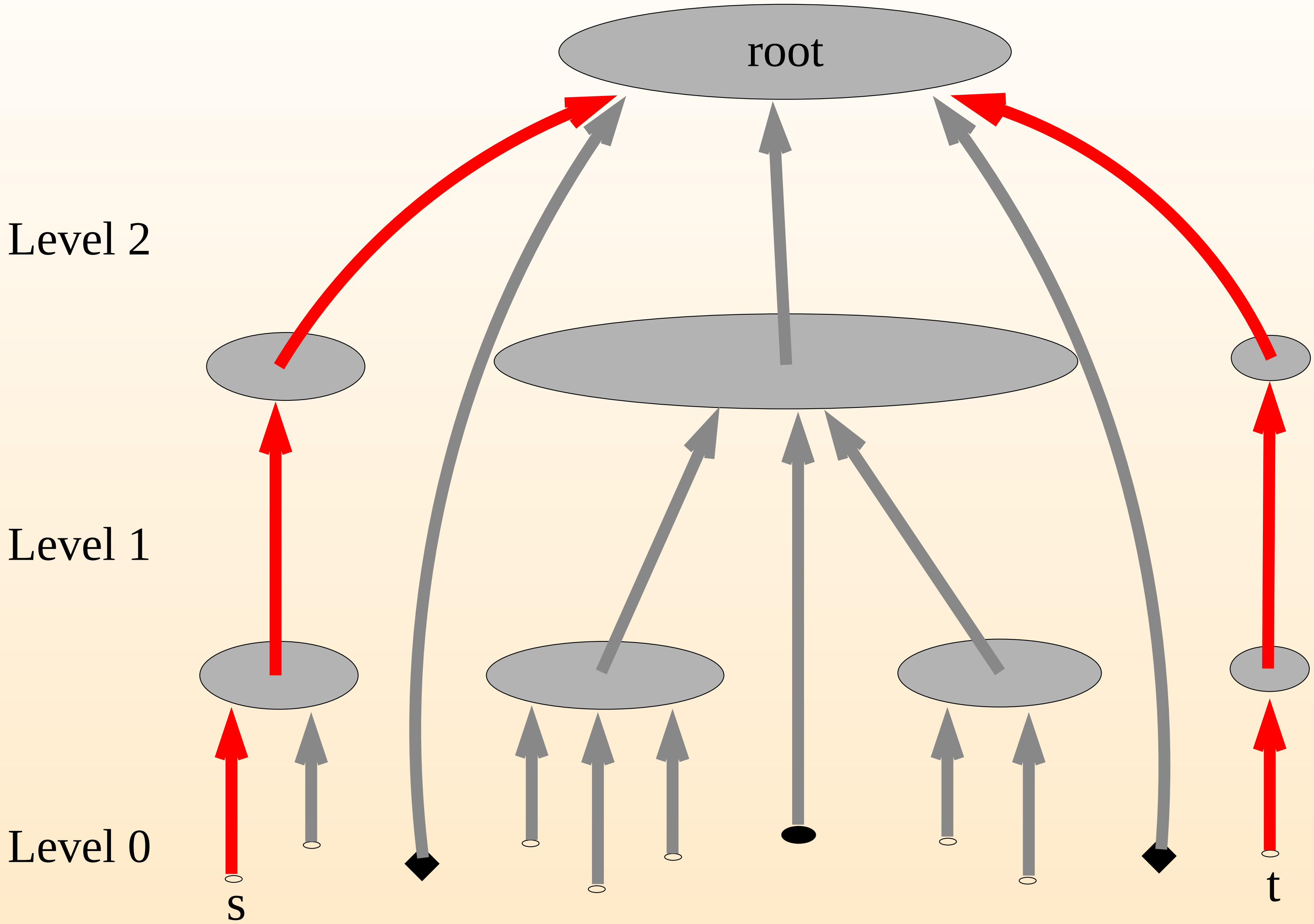
- a weighted digraph $G = (V, E)$
- a sequence of subsets of V

$$V \supset S_1 \supset \ldots \supset S_l$$

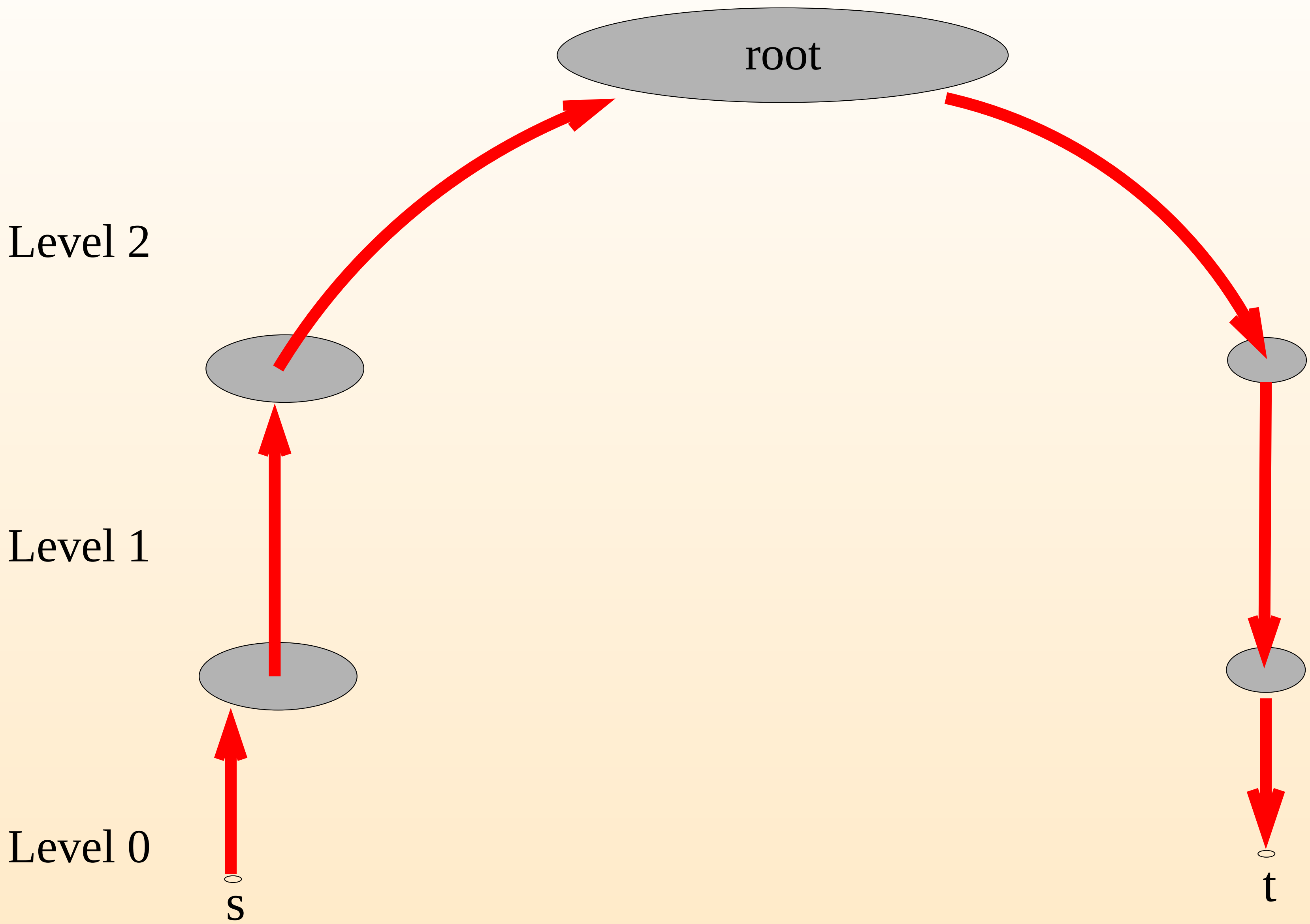
Outline

- Construct l levels of additional edges $\rightarrow \mathcal{M}(G)$ $\surd$
- Component Tree $\surd$
- Define subgraph of $\mathcal{M}(G)$ for a pair $s, t \in V$
- Use subgraph to compute s-t shortest path

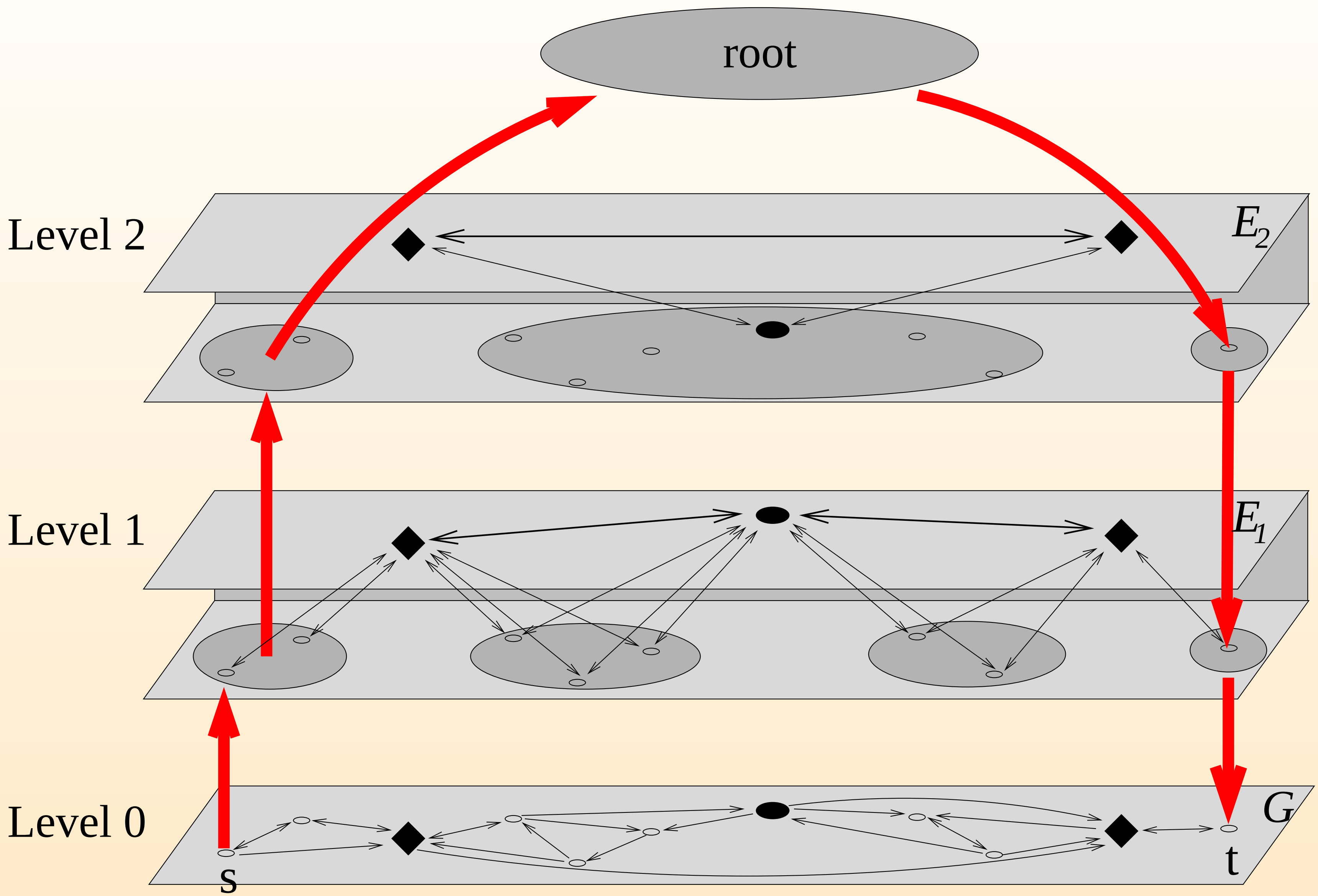
Define Subgraph



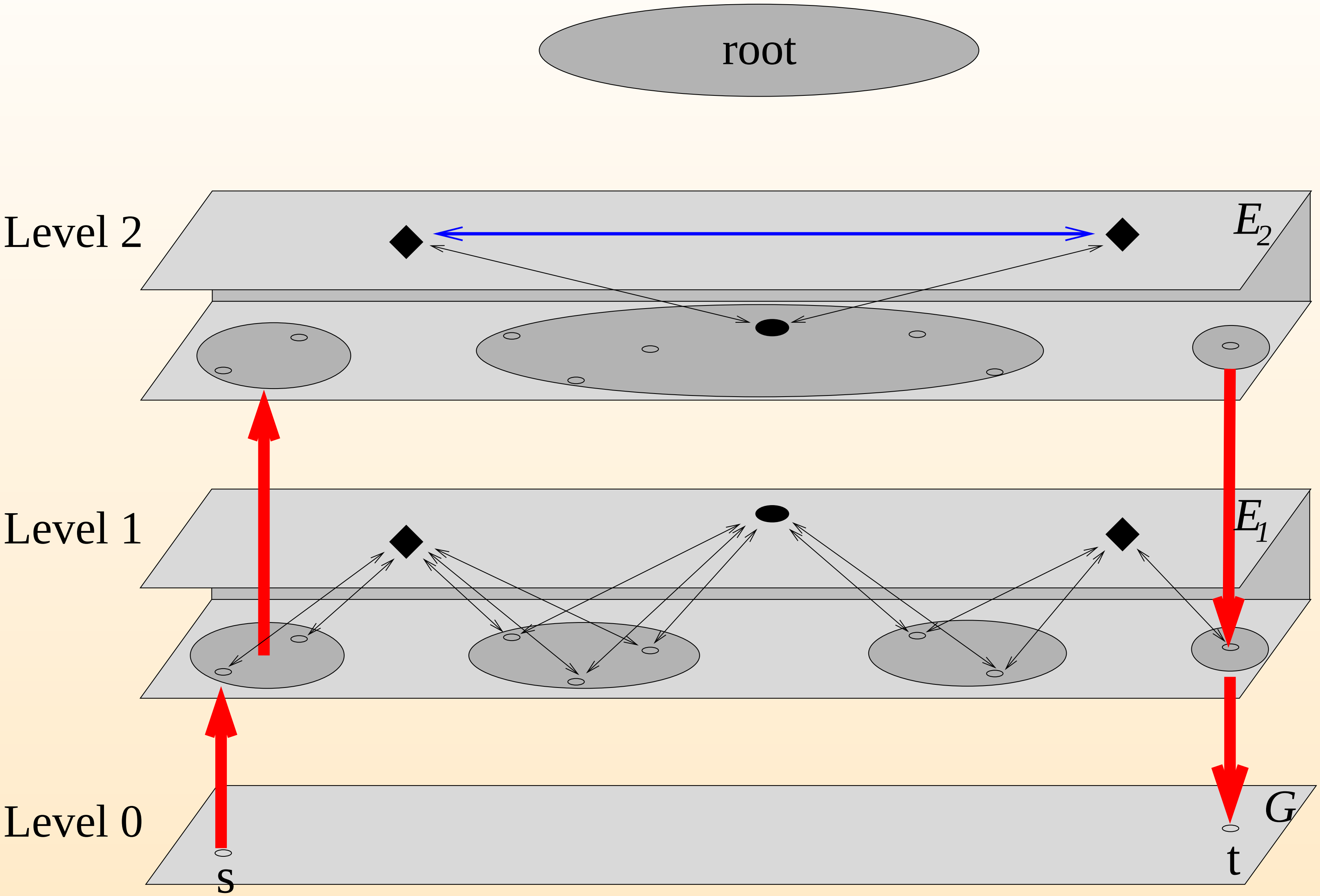
Define Subgraph



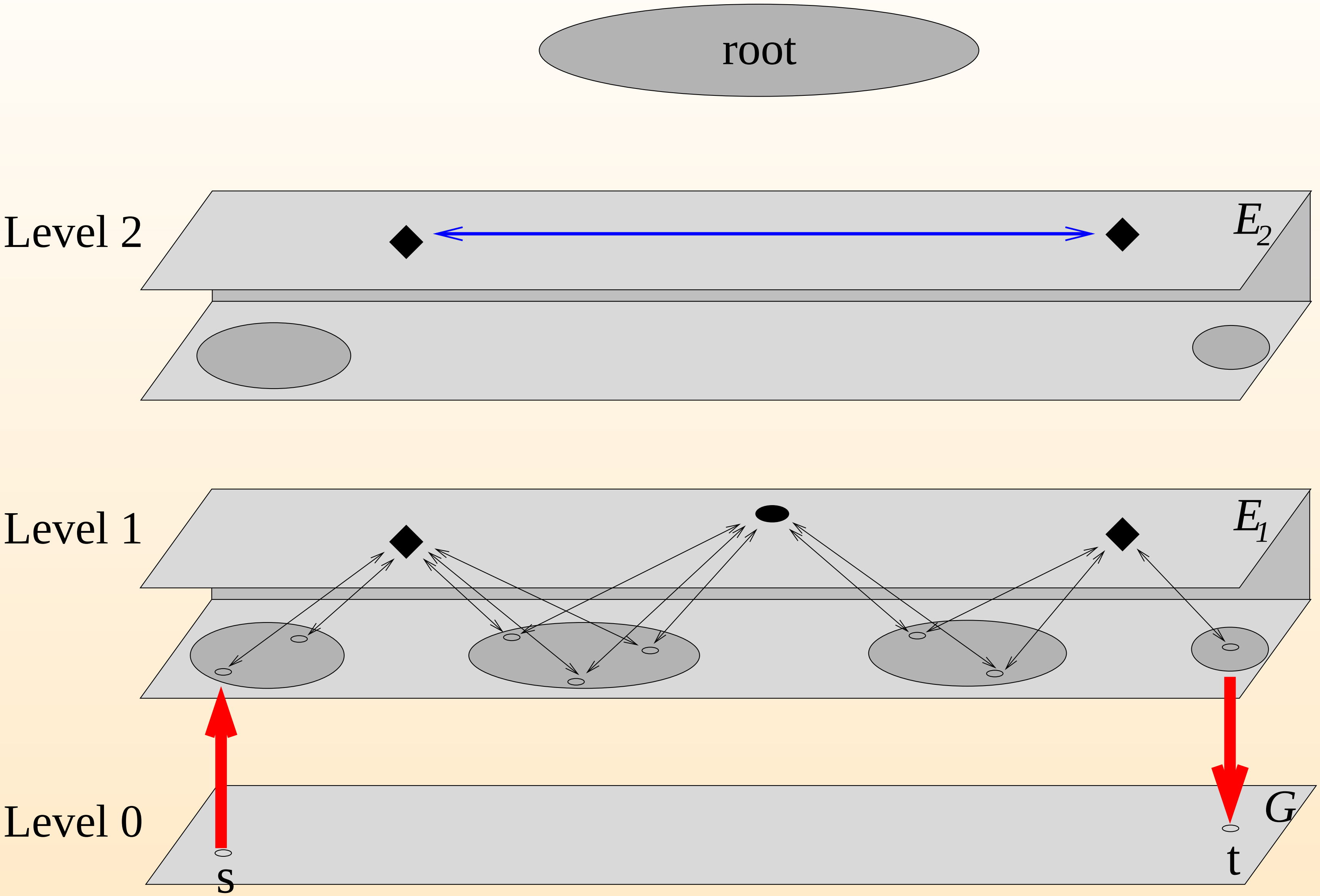
Define Subgraph



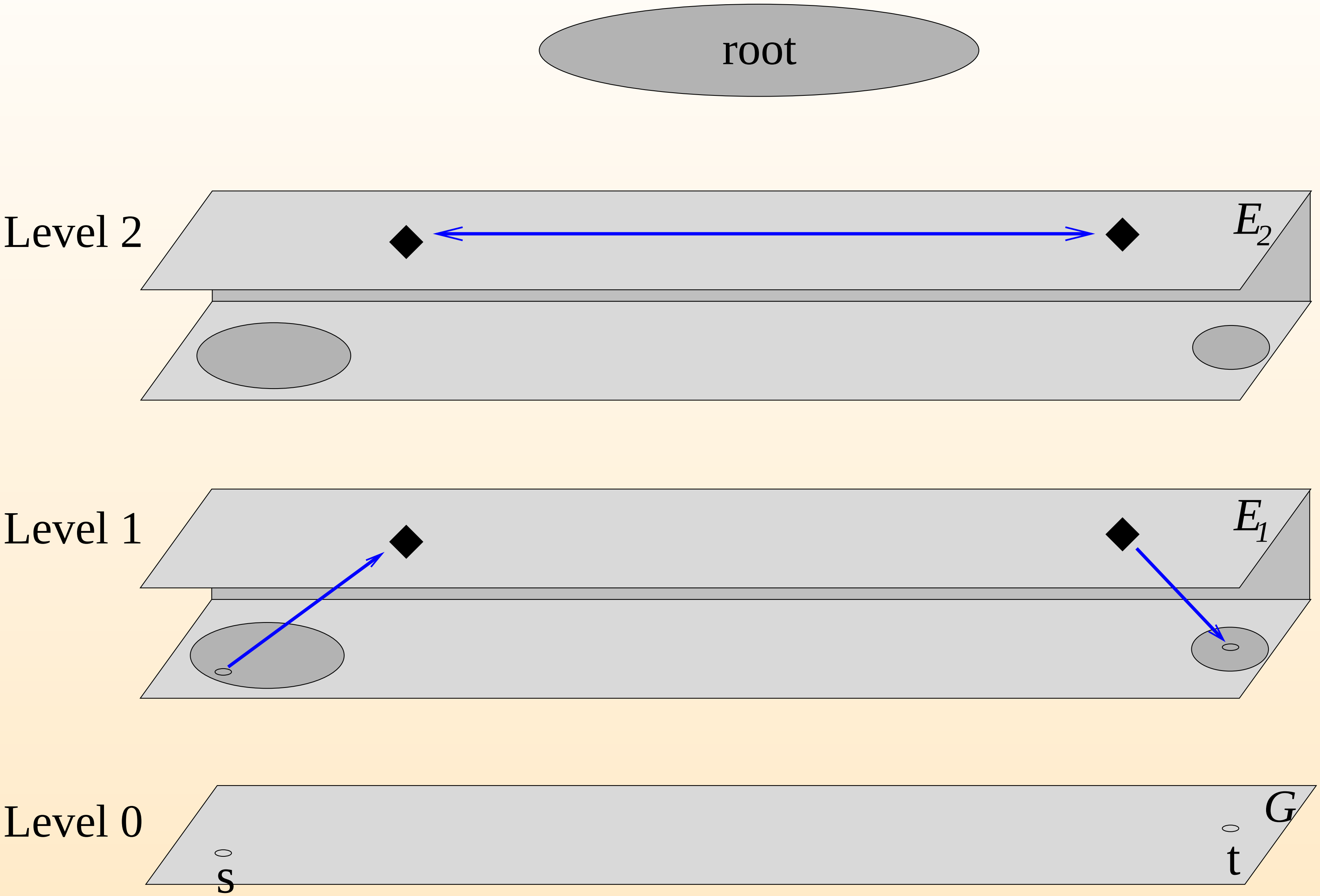
Define Subgraph



Define Subgraph



Define Subgraph



Multi-Level Graphs

Given

- a weighted digraph $G = (V, E)$
- a sequence of subsets of V

$$V \supset S_1 \supset \ldots \supset S_l$$

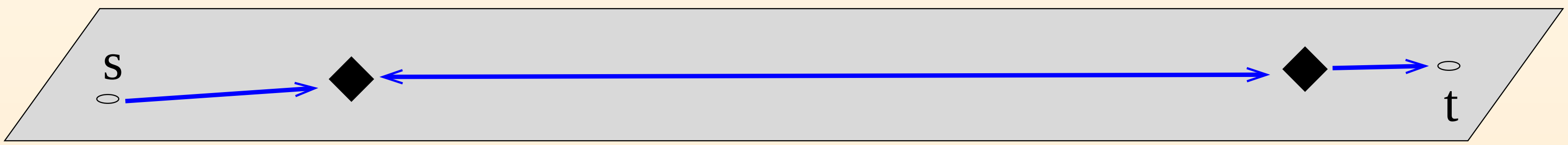
Outline

- Construct l levels of additional edges $\rightarrow \mathcal{M}(G)$ $\surd$
- Component Tree $\surd$
- Define subgraph of $\mathcal{M}(G)$ for a pair $s, t \in V$ $\surd$
- Use subgraph to compute s-t shortest path

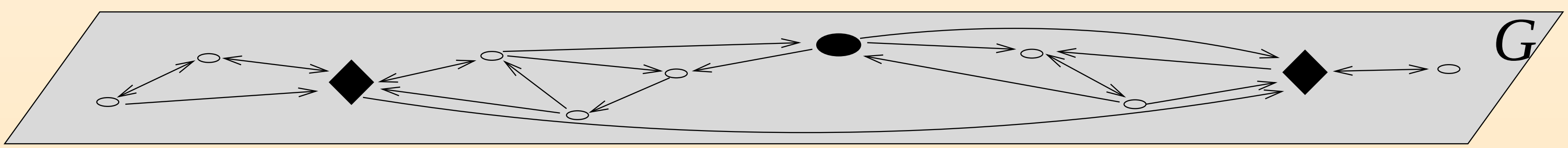
Lemma

The length of a shortest s-t path is the same in the s-t subgraph of $\mathcal{M}(G)$ and G.

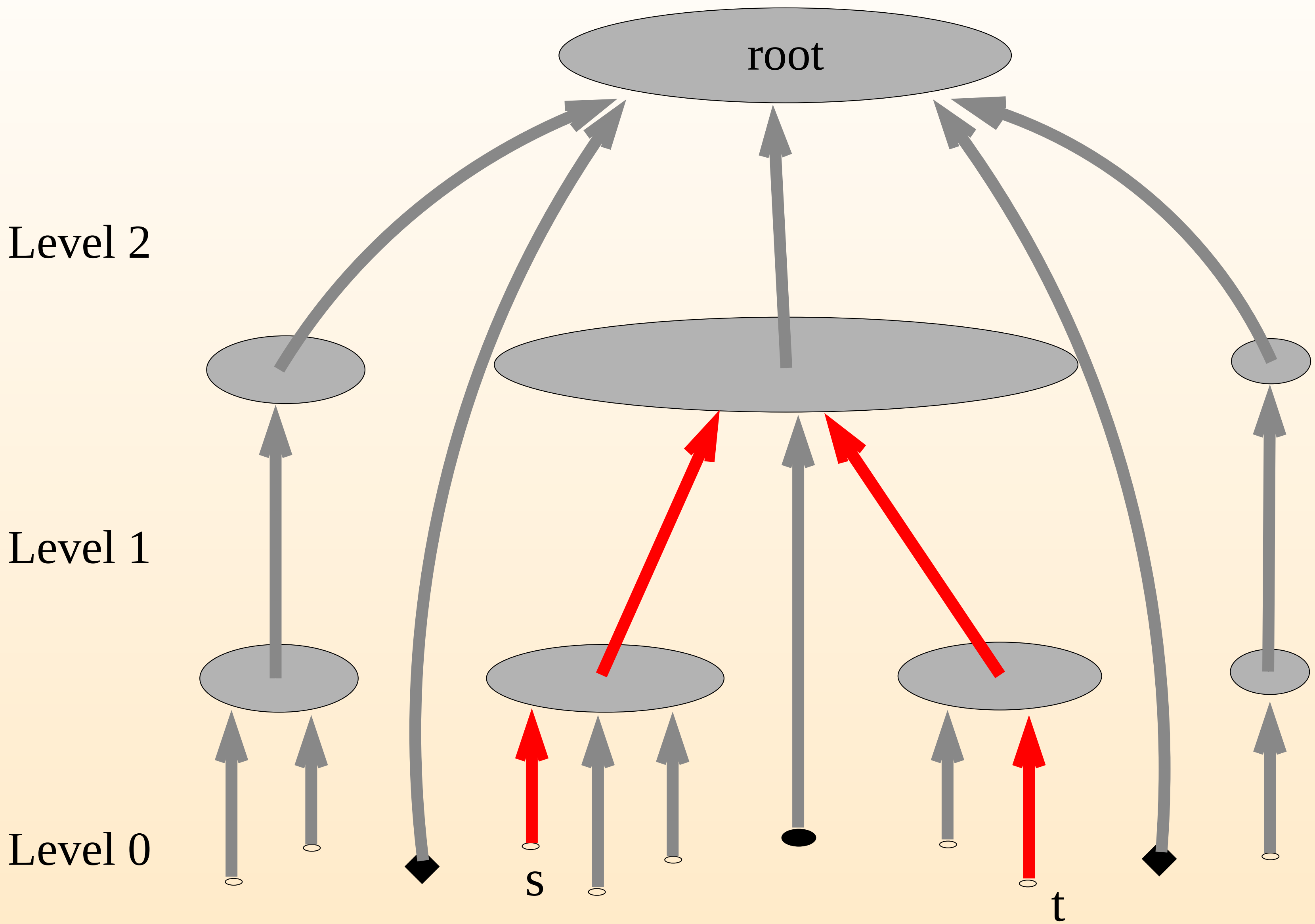
Subgraph of $\mathcal{M}(G)$:



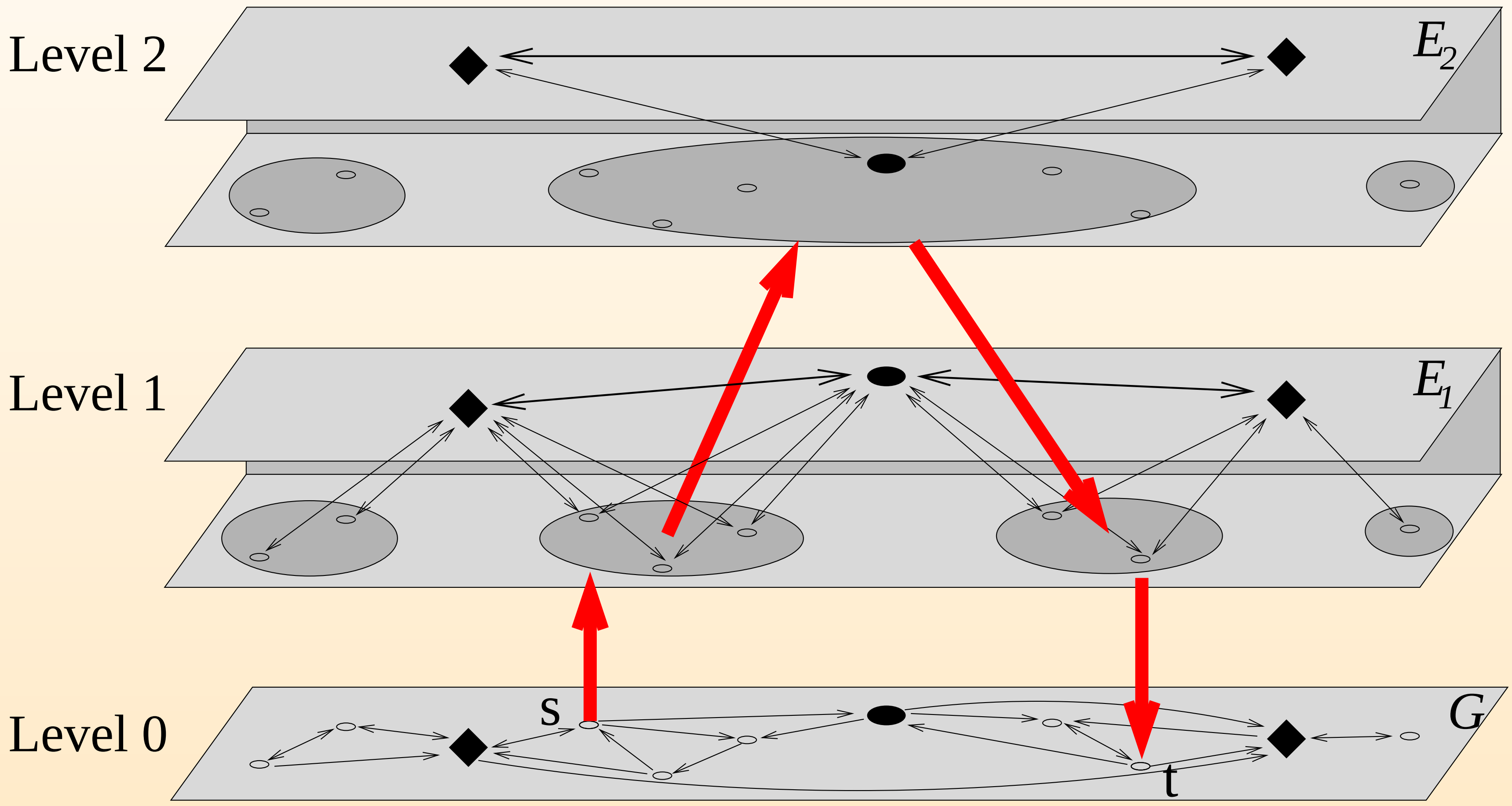
Original Graph G:



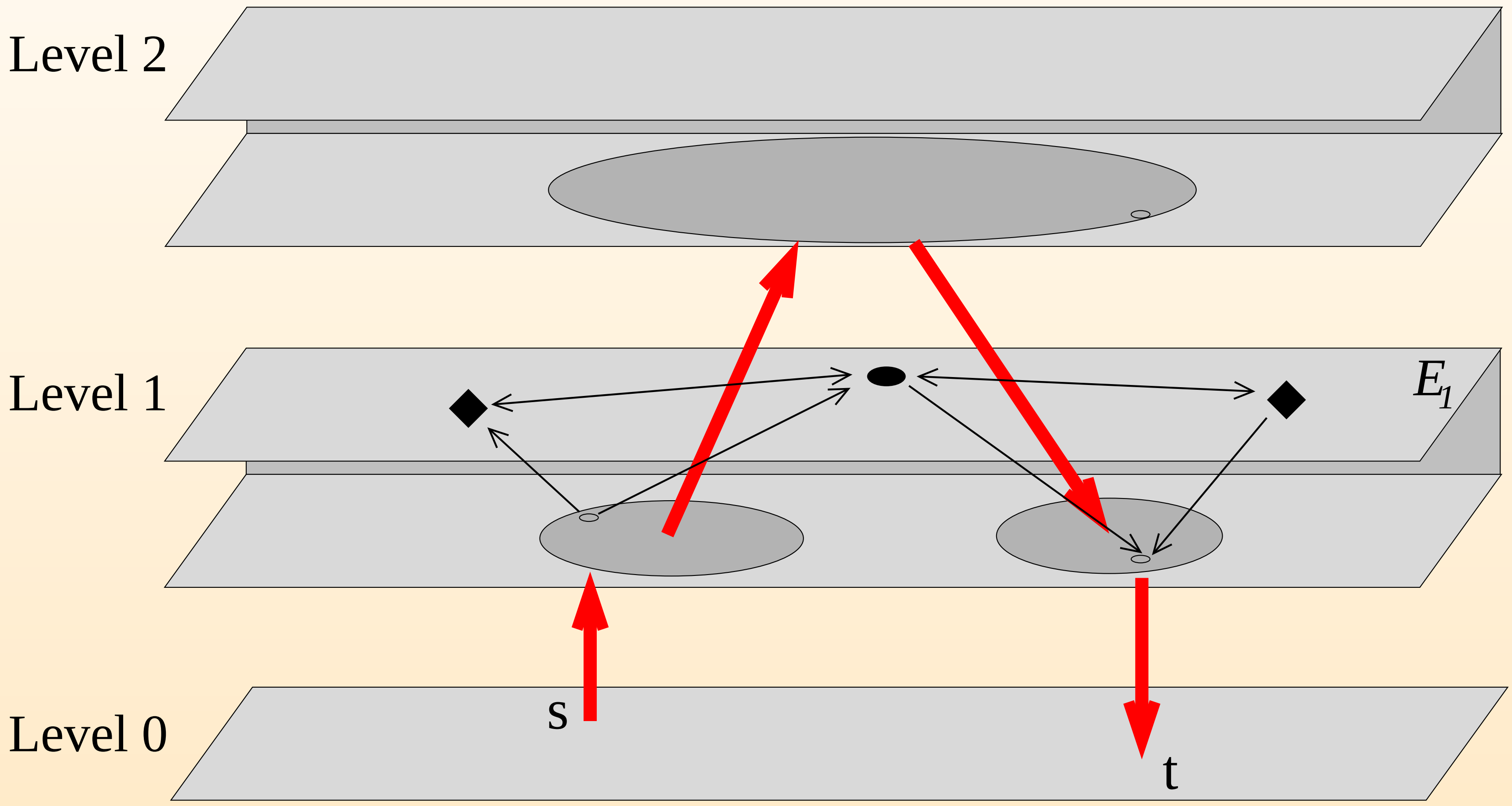
A Different Pair s, t



A Different Pair s, t



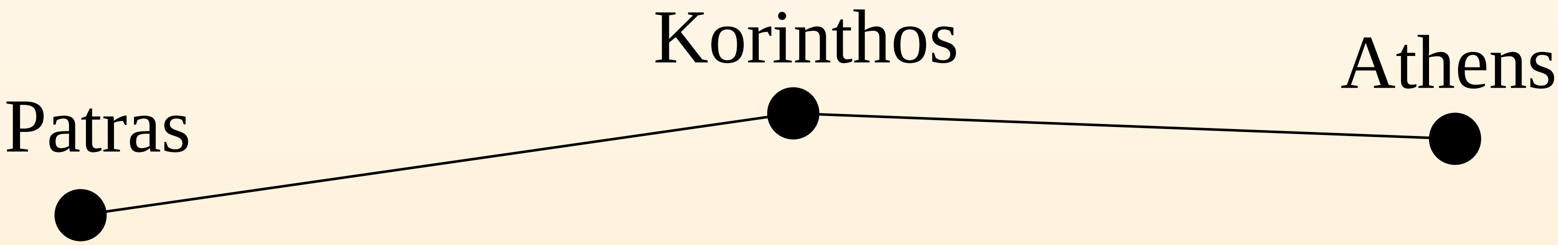
A Different Pair s, t



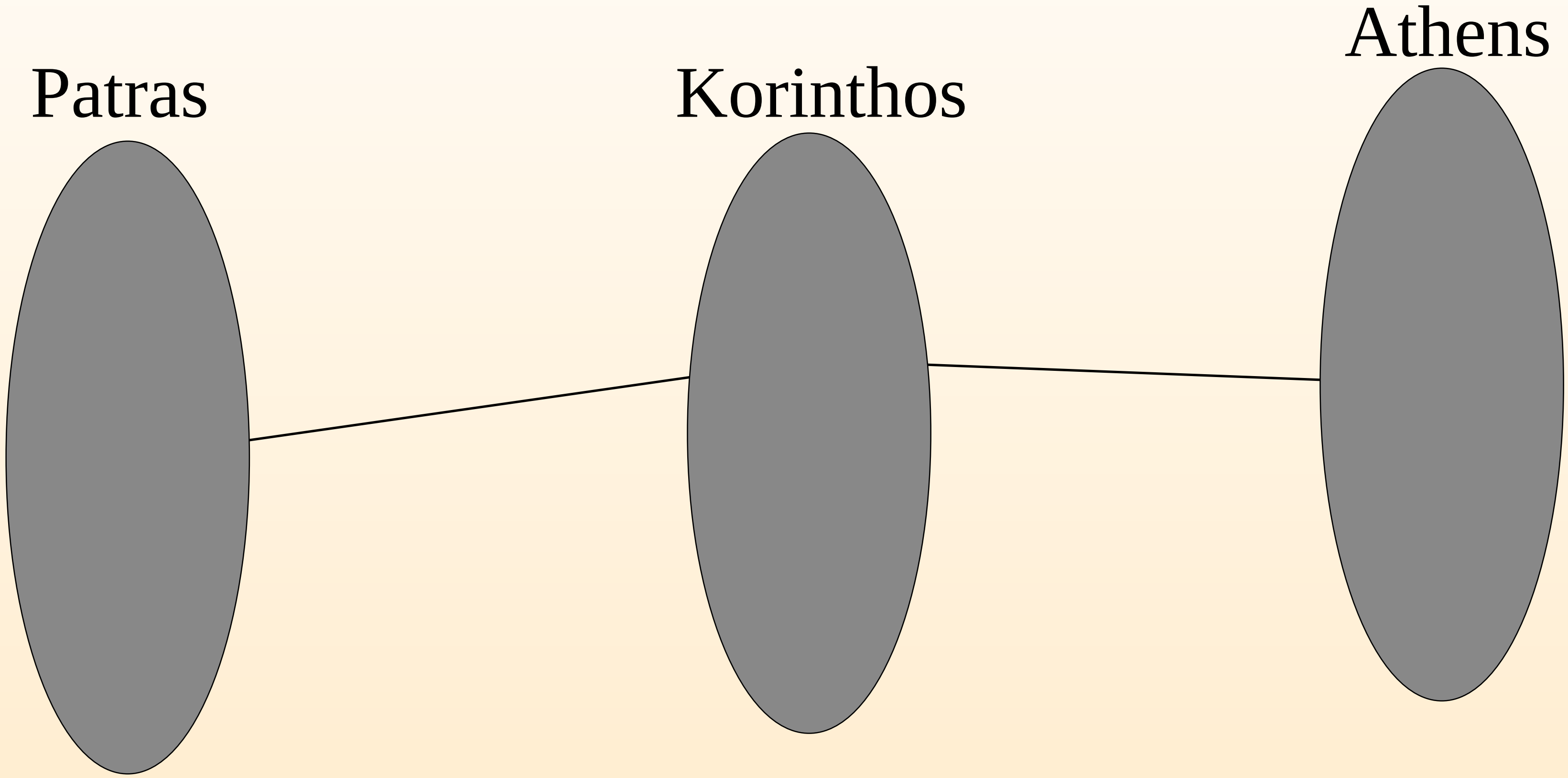
Overview

1. Introduction √
2. Multi-Level Graphs √
3. Timetable Information Graphs
4. Experiments
5. Conclusion & Outlook

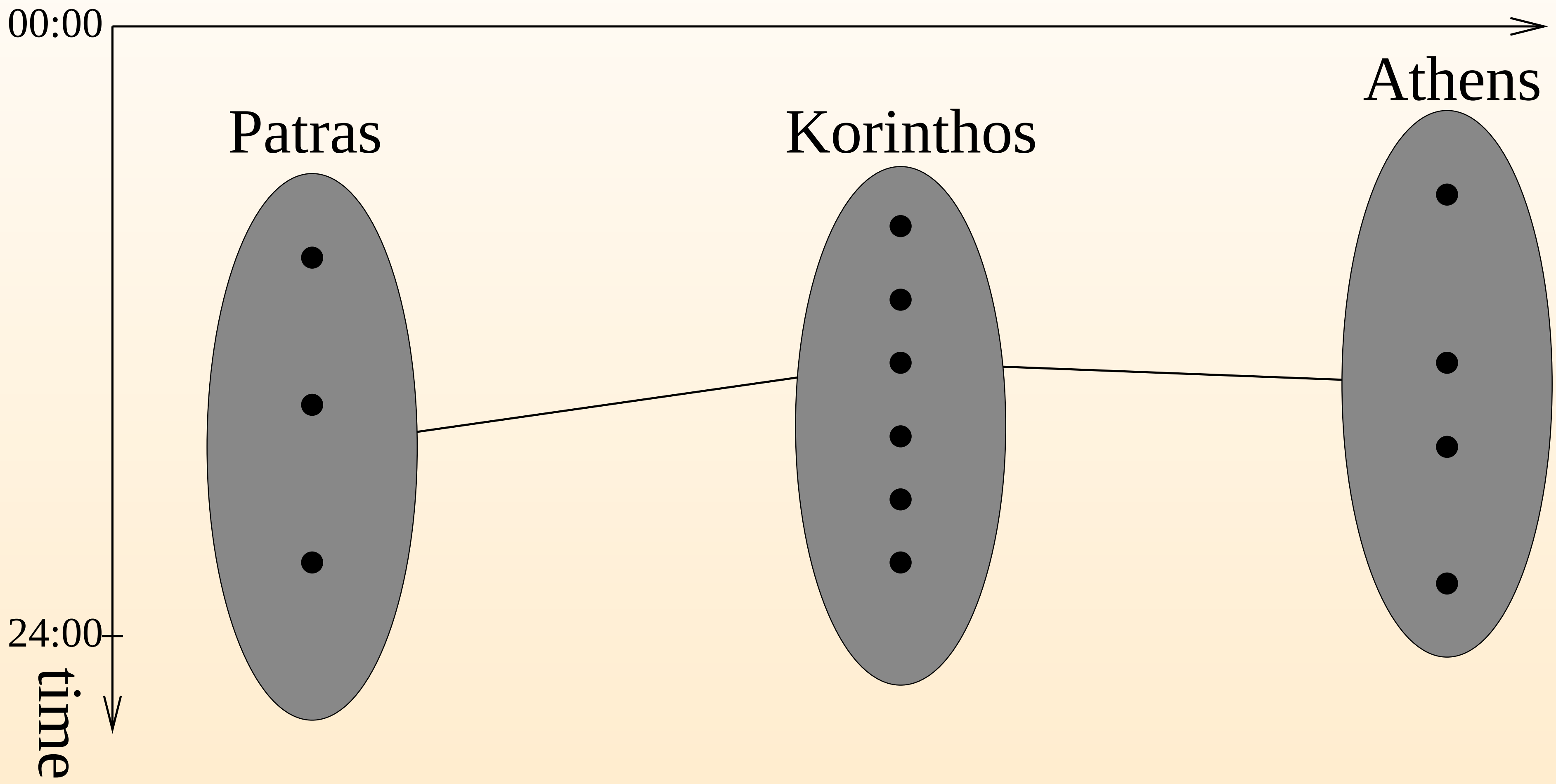
Station Graph



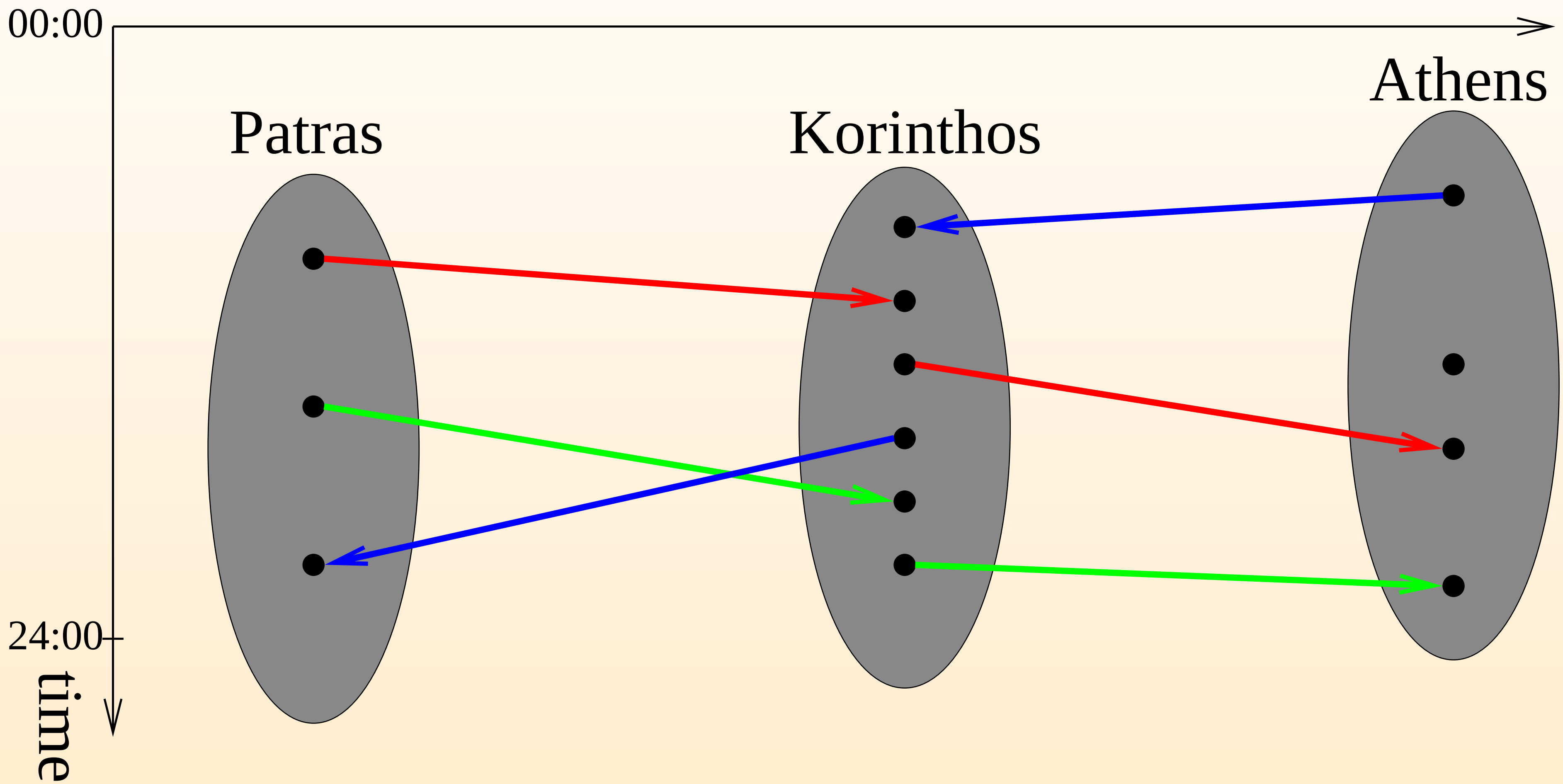
Train Graph



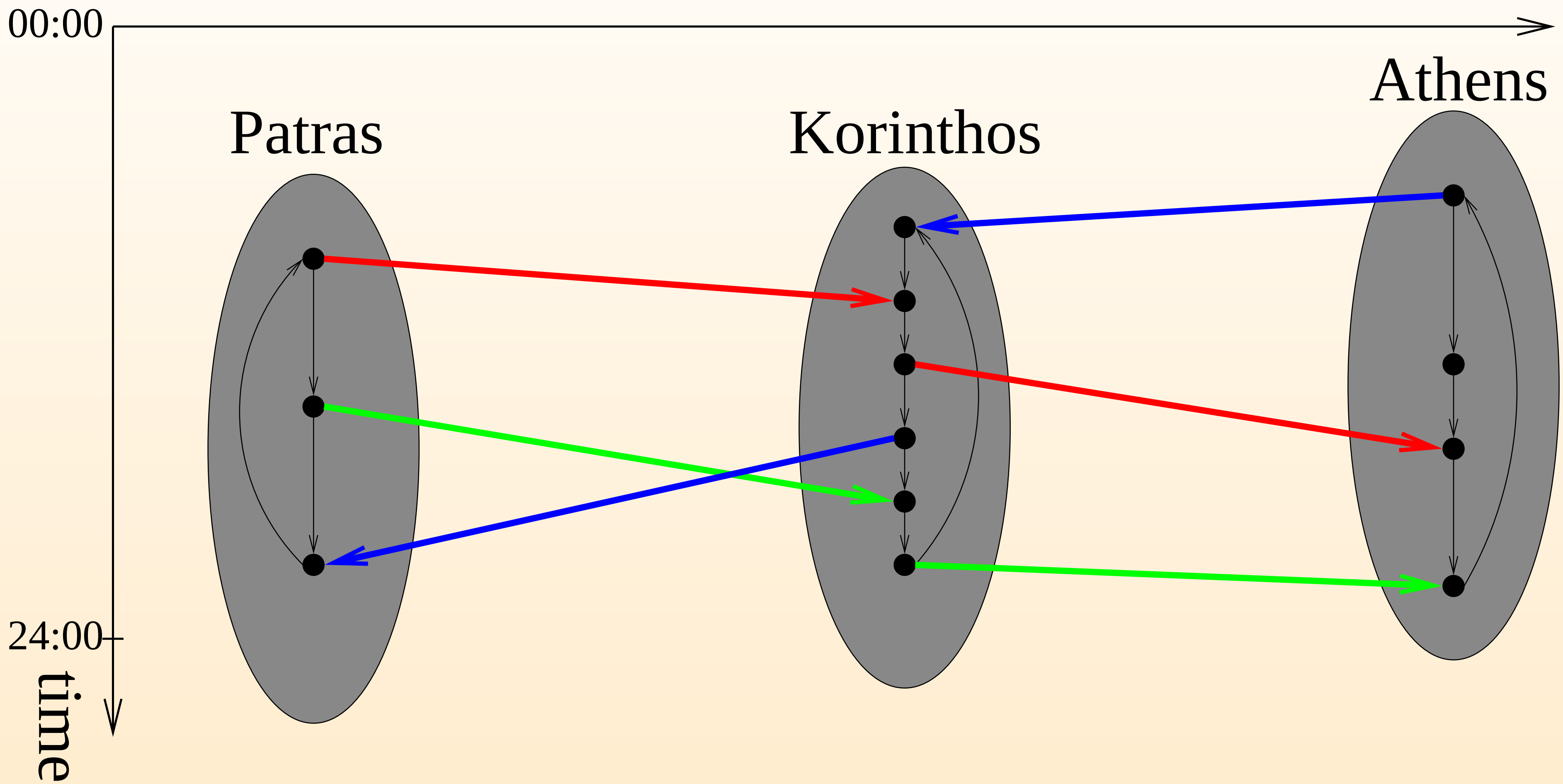
Train Graph



Train Graph

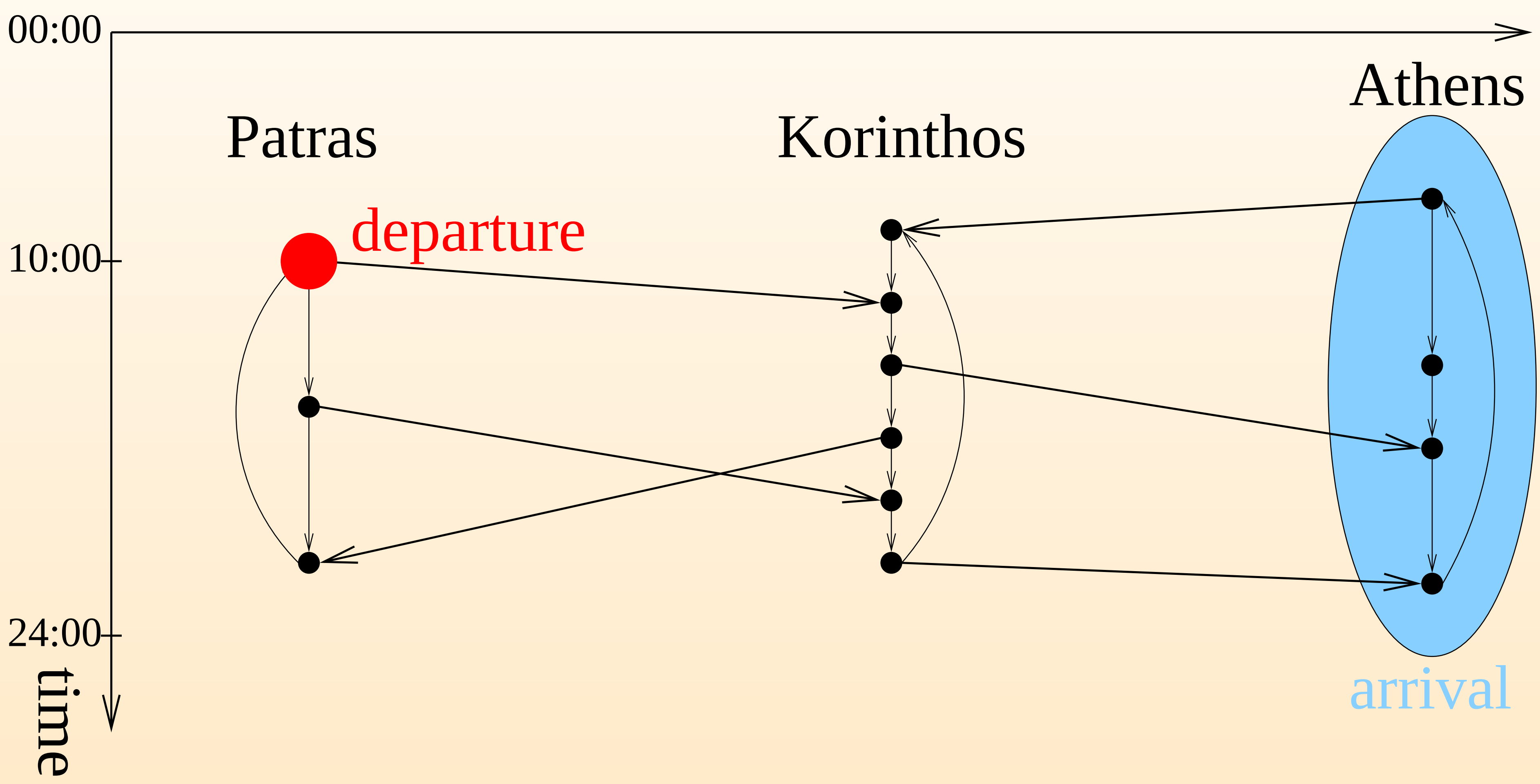


Train Graph



Query

Single-source **some-targets** shortest path problem



Multi-Level Train Graph

- Given l subsets of **stations** $\Sigma_1 \supset \ldots \supset \Sigma_l$
- Component tree in station graph
- Define subgraph for a pair of **stations**

Overview

1. Introduction √
2. Multi-Level Graphs √
3. Timetable Information Graphs √
4. Experiments
5. Conclusion & Outlook

Experiments

Given one instance G of a traingraph

- Timetable of German trains, winter 1996/97
- $\sim$ 500000 vertices, $\sim$ 7000 stations

Evaluate multi-level train graph $\mathcal{M}(G, \Sigma_1, \ldots, \Sigma_l)$

Investigate dependence on

- number of levels
- input sequence $\Sigma_1 \supset \ldots \supset \Sigma_l$

Sets of stations Σ_i

Three criteria to sort stations

A Importance regarding train changes

B Degree in station graph

C Random

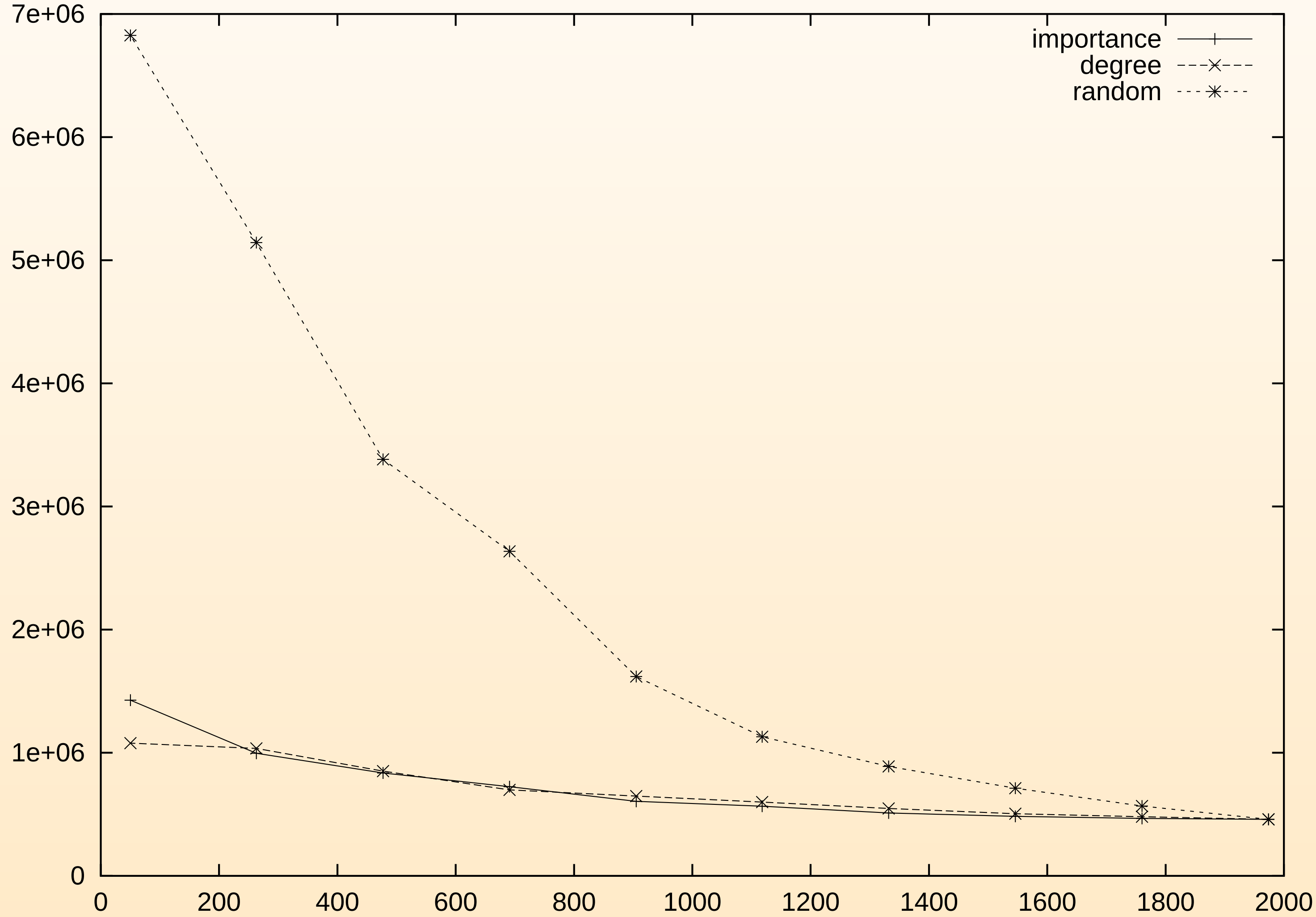
Consider the first n_j stations according to A, B, C

- $n_1 = 1974, \dots, n_{10} = 50$

$\rightsquigarrow$ Sets of stations Σ_j^A, Σ_j^B, Σ_j^C

2-Level Graphs $\mathcal{M}(G, \Sigma_j^X)$

Number of Additional Edges



Evaluation of $\mathcal{M}(G, \Sigma_1, \ldots, \Sigma_l)$

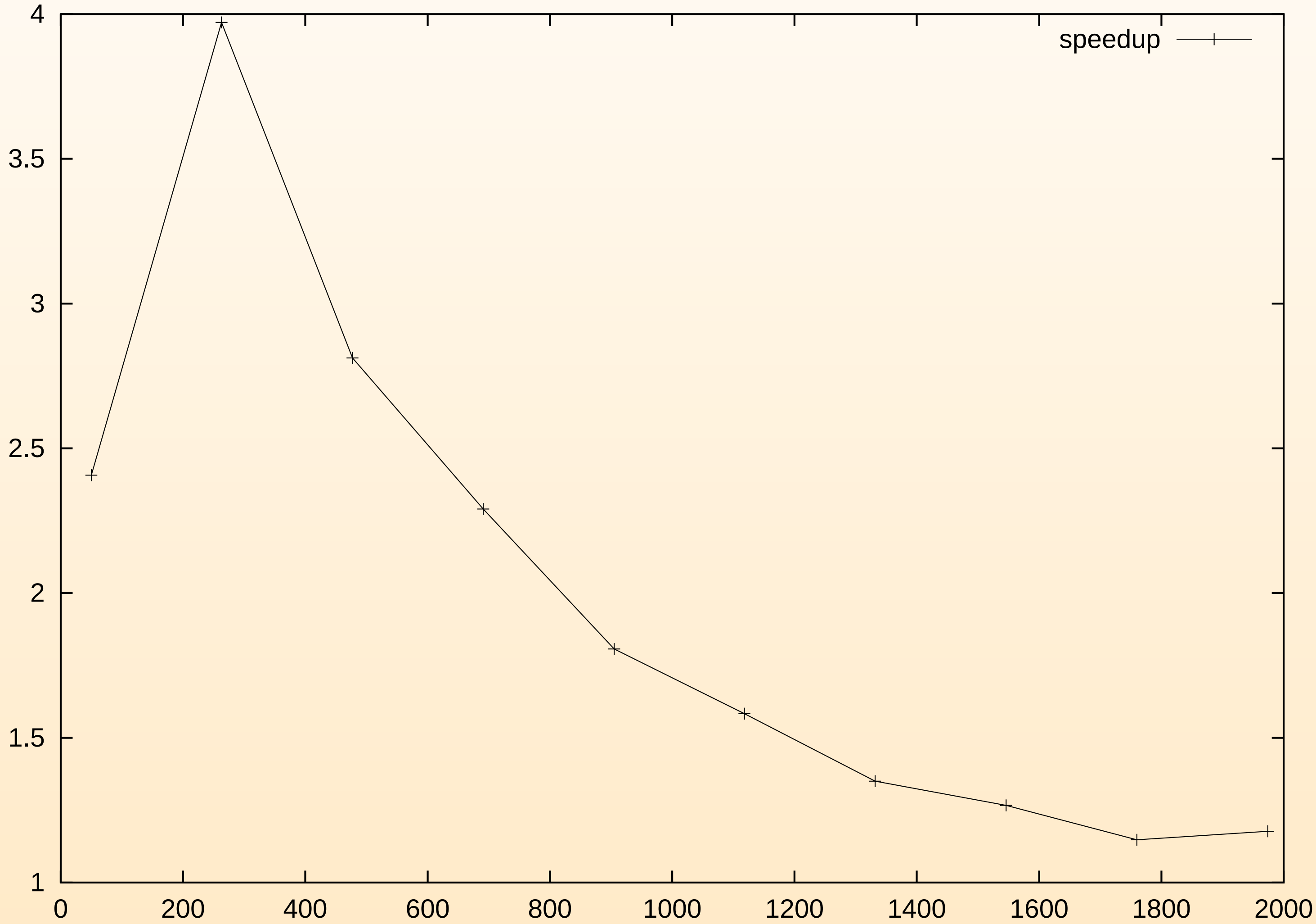
Consider only criterion A in the following

Average Speed-up over 100000 queries

- $r =$ Runtime for Dijkstra in G
- $q =$ Runtime for Dijkstra in subgraph of $\mathcal{M}$
- Speed-up $= r/q$

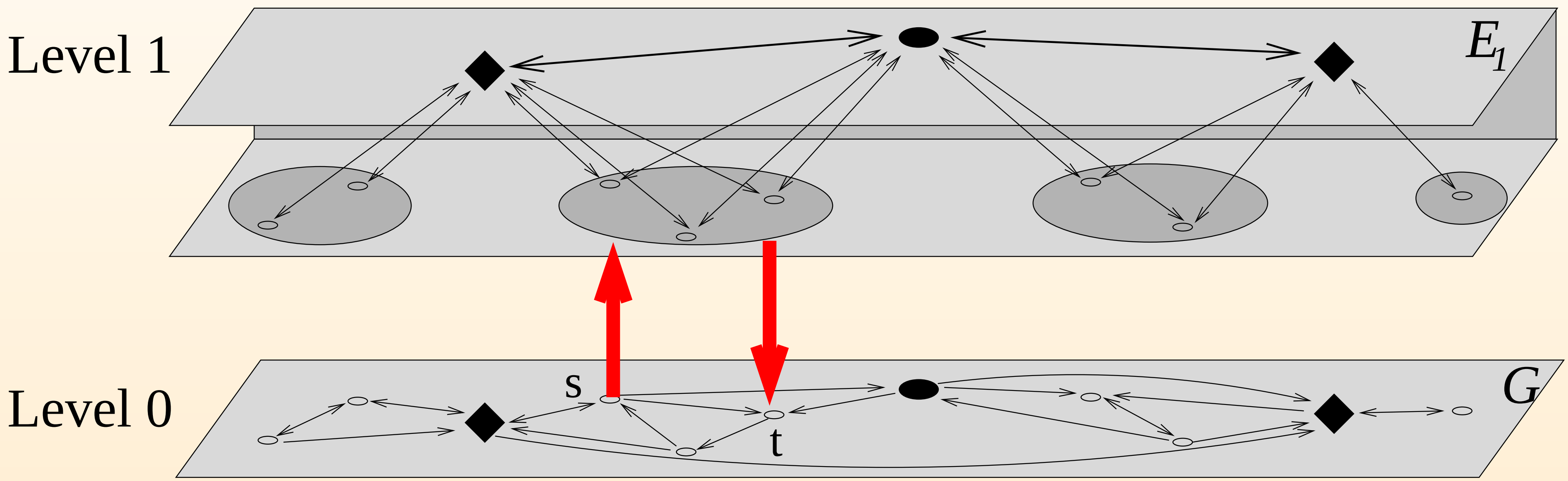
2-Level Graphs $\mathcal{M}(G, \Sigma_j^A)$

Average Speed-up



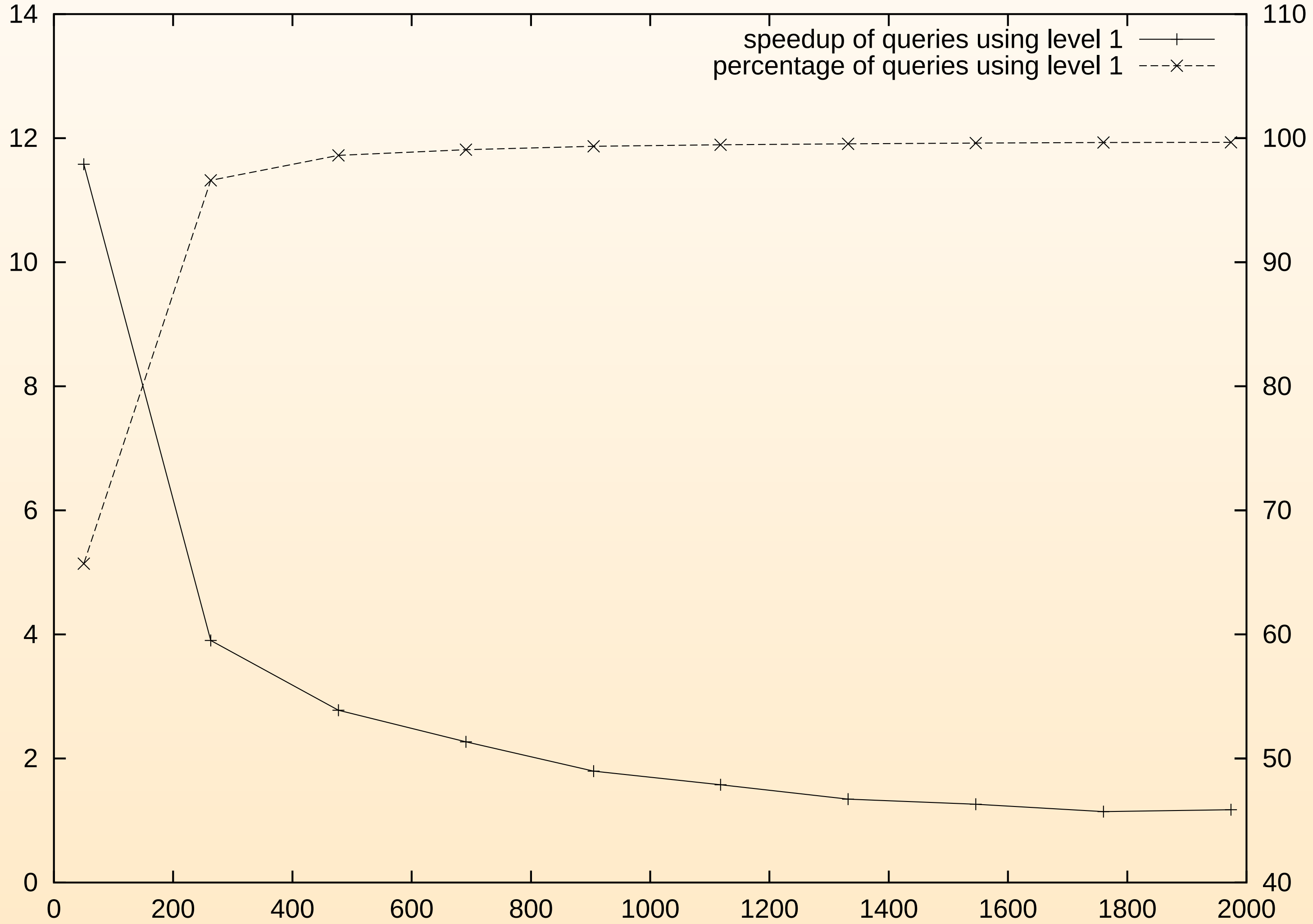
Problem with small Σ_j^A

Big components $\Rightarrow$ Level 1 rarely used



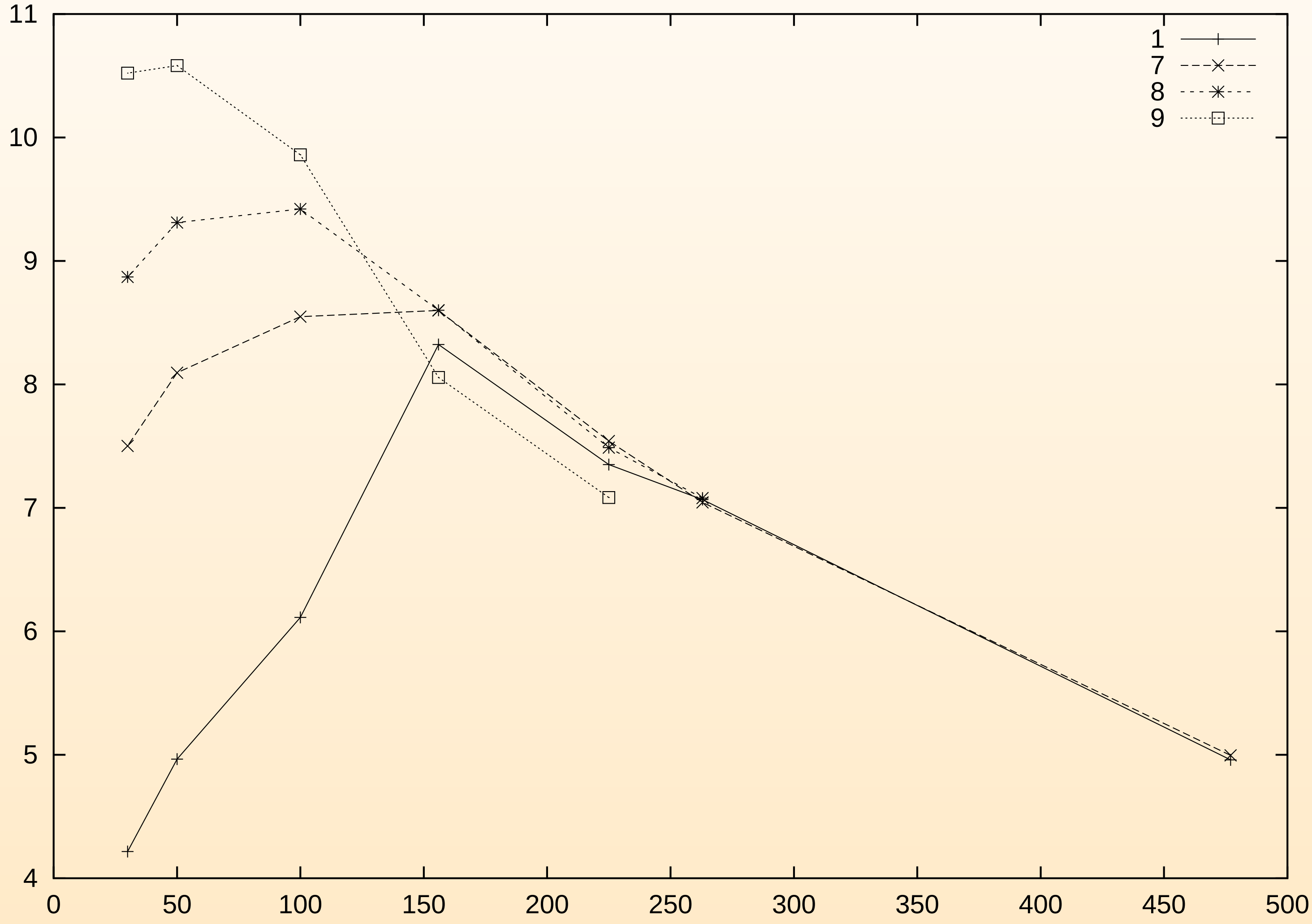
2-Level Graphs $\mathcal{M}(G, \Sigma_j^A)$

Queries Using Level 1



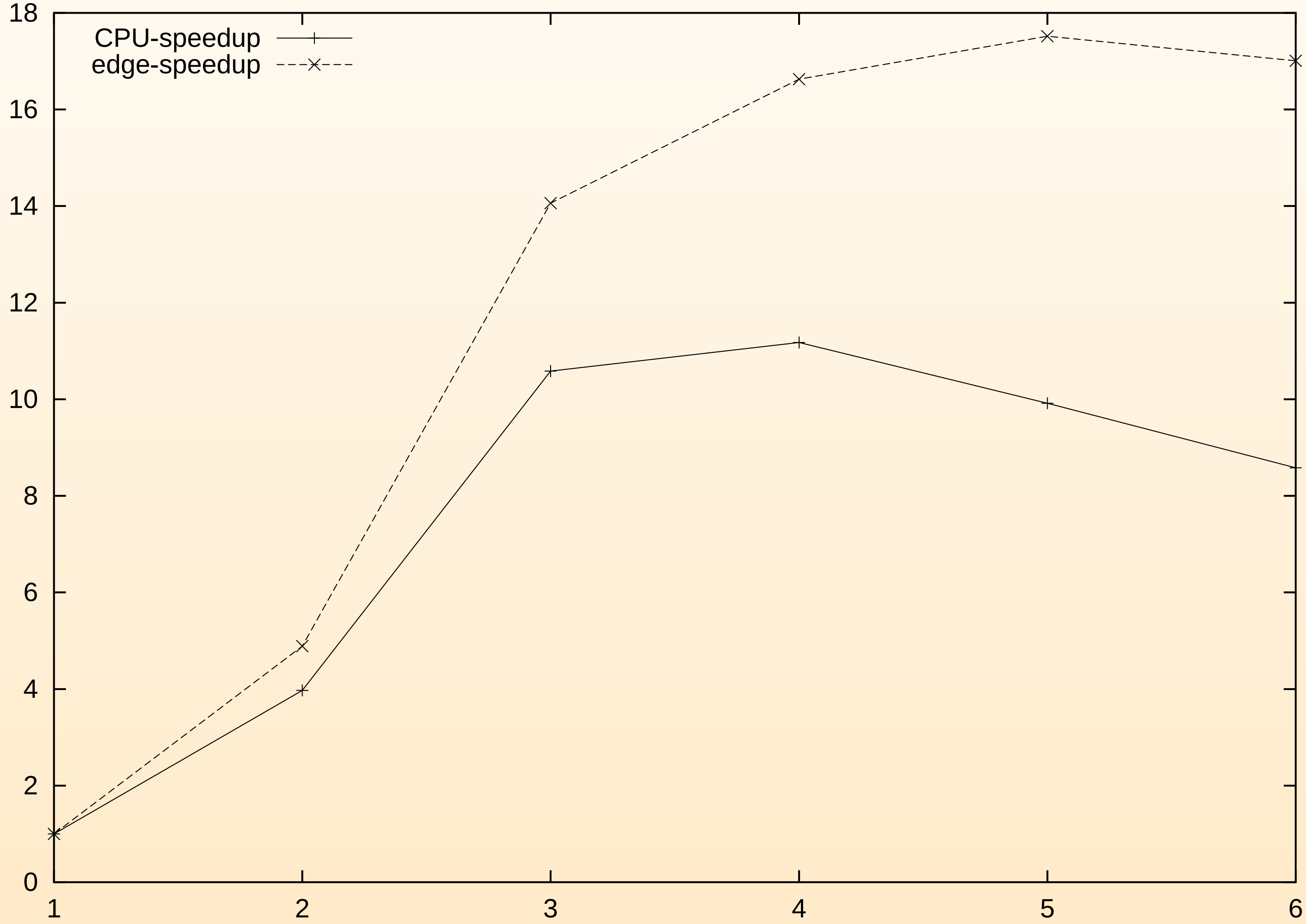
3-Level Graphs $\mathcal{M}(G, \Sigma_j^A, \Sigma_k^A)$

Average Speed-up



l-Level Graphs $\mathcal{M}(G, \Sigma_1, \ldots, \Sigma_{l-1})$

Best Average Speed-up (CPU-time, #Edges hit)



Overview

1. Introduction √
2. Multi-Level Graphs √
3. Timetable Information Graphs √
4. Experiments √
5. Conclusion & Outlook

Conclusion & Outlook

Experimental evaluation of multi-level graphs

- Input graph from timetable information
- Best number of levels: 4, 5
- Sizes of Σ_i are crucial

Outlook

- Other input graphs
- Relation $(\Sigma_1, \ldots, \Sigma_l) \leftrightarrow$ speed-up
  - ▷ Here: Criteria A, B, C; Sizes of Σ_i
- Theoretical analysis

Overview

1. Introduction $\surd$
2. Multi-Level Graphs $\surd$
3. Timetable Information Graphs $\surd$
4. Experiments $\surd$
5. Conclusion & Outlook $\surd$