The data arrangement problem on binary trees

Rostislav Staněk

Eranda Çela   Joachim Schauer

Dep. of Production and Logistics, University of Graz
Dep. of Optimization and Discrete Mathematics, Graz University of Technology
Dep. of Statistics and Operations Research, University of Graz

Short overview

- problem definition

Short overview

- problem definition
- solution algorithm

Short overview

- problem definition
- solution algorithm
- lower bound

Short overview

- problem definition
- solution algorithm
- lower bound:
  - problem transformation and approximation guarantee

Short overview

- problem definition
- solution algorithm
- lower bound:
  - problem transformation and approximation guarantee
- $\mathcal{NP}$-hardness for a special case

Problem definition

- Given
  - an undirected graph $G = \big(V(G), E(G)\big)$,

Problem definition

- Given
  - an undirected graph $G = \big(V(G), E(G)\big)$,
  - an undirected graph $T = \big(V(T), E(T)\big)$ with $|V(T)| \geq |V(G)|$ and

Problem definition

- Given
  - an undirected graph $G = \big(V(G), E(G)\big)$,
  - an undirected graph $T = \big(V(T), E(T)\big)$ with $|V(T)| \geq |V(G)|$ and
  - a subset $B \subseteq V(T)$ with $|B| \geq \big|V(G)\big|$,

Problem definition

- Given
  - an undirected graph $G = \big(V(G), E(G)\big)$,
  - an undirected graph $T = \big(V(T), E(T)\big)$ with $|V(T)| \geq |V(G)|$ and
  - a subset $B \subseteq V(T)$ with $|B| \geq \big|V(G)\big|$,

  the **generic graph embedding problem (GEP)** consists of finding an injective embedding of the vertices of G into the vertices in B such that some prespecified objective function is minimized.

Problem definition

- Given
  - an undirected graph $G = \big(V(G), E(G)\big)$,
  - an undirected graph $T = \big(V(T), E(T)\big)$ with $|V(T)| \geq |V(G)|$ and
  - a subset $B \subseteq V(T)$ with $|B| \geq \big|V(G)\big|$,

  the **generic graph embedding problem (GEP)** consists of finding an injective embedding of the vertices of G into the vertices in B such that some prespecified objective function is minimized.
- A commonly used objective function maps an embedding $\phi\colon V(G) \to B$ to

$$\sum_{(i,j)\in E(G)} d\big(\phi(i), \phi(j)\big),$$

Problem definition

- Given
  - an undirected graph $G = \big(V(G), E(G)\big)$,
  - an undirected graph $T = \big(V(T), E(T)\big)$ with $|V(T)| \geq |V(G)|$ and
  - a subset $B \subseteq V(T)$ with $|B| \geq \big|V(G)\big|$,

  the **generic graph embedding problem (GEP)** consists of finding an injective embedding of the vertices of G into the vertices in B such that some prespecified objective function is minimized.
- A commonly used objective function maps an embedding $\phi\colon V(G) \rightarrow B$ to

$$\sum_{(i,j)\in E(G)} d\big(\phi(i), \phi(j)\big), \tag{1}$$

  where $d(x, y)$ denotes the length of the shortest path between x and y in T.

Problem definition

- Different special cases of the GEP have been studied in the literature.

Problem definition

- Different special cases of the GEP have been studied in the literature:
  - The **linear arrangement problem (LAP)** is probably the most prominent one.

Problem definition

- Different special cases of the GEP have been studied in the literature:
  - The **linear arrangement problem (LAP)** is probably the most prominent one:
    - The problem is solvable in polynomial time for undirected trees [SHILOACH 1979[1], CHUNG 1984[2]].

[1] Y. Shiloach, A minimum linear arrangement algorithm for undirected trees, *SIAM Journal on Computing* **8 (1)**, 15–22, 1979.

[2] F. R. K. Chung, On optimal linear arrangements of trees, *Computers and Mathematics with Applications* **10 (1)**, 43–60, 1984.

Problem definition

- Different special cases of the GEP have been studied in the literature:
  - The **linear arrangement problem (LAP)** is probably the most prominent one:
    - The problem is solvable in polynomial time for undirected trees [SHILOACH 1979[1], CHUNG 1984[2]].
- In our case T is a d-regular tree and B is the set of its leaves.

[1]Y. Shiloach, A minimum linear arrangement algorithm for undirected trees, *SIAM Journal on Computing* **8 (1)**, 15–22, 1979.

[2]F. R. K. Chung, On optimal linear arrangements of trees, *Computers and Mathematics with Applications* **10 (1)**, 43–60, 1984.

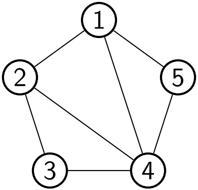
Problem definition

- Different special cases of the GEP have been studied in the literature:
  - The **linear arrangement problem (LAP)** is probably the most prominent one:
    - The problem is solvable in polynomial time for undirected trees [SHILOACH 1979[1], CHUNG 1984[2]].
- In our case T is a d-regular tree and B is the set of its leaves.
- We will call this problem **data arrangement problem on regular trees (DAPT)** and denote the objective value $OV(G, d, \phi)$.

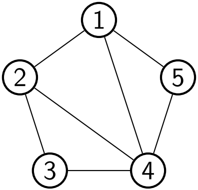
[1]Y. Shiloach, A minimum linear arrangement algorithm for undirected trees, *SIAM Journal on Computing* **8 (1)**, 15–22, 1979.

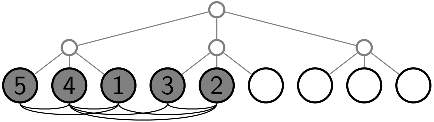
[2]F. R. K. Chung, On optimal linear arrangements of trees, *Computers and Mathematics with Applications* **10 (1)**, 43–60, 1984.

Problem definition



Problem definition





$$OV(G, 3, \phi) = 20$$

General properties and our special case

- DAPT is $\mathcal{NP}$-hard for every fixed $d \geq 2$ [LUCZAK, NOBLE 2002[3]].

[4]M.J. Luzcak and S.D. Noble, Optimal arrangement of data in a tree directory, *Discrete Applied Mathematics* **121 (1–3)**, 307–315, 2002.

General properties and our special case

- DAPT is $\mathcal{NP}$-hard for every fixed $d \geq 2$ [Luczak, Noble 2002[3]].
- Çela and Staněk introduce some heuristics for this problem [Çela, Staněk 2013[4]].

[4] M.J. Luzcak and S.D. Noble, Optimal arrangement of data in a tree directory, *Discrete Applied Mathematics* **121 (1–3)**, 307–315, 2002.

[5] E. Çela and R. Staněk, Heuristics for the data arrangement problem on regular trees, *Journal of Combinatorial Optimization* **30 (3)**, 768–802, 2015.

General properties and our special case

- DAPT is $\mathcal{NP}$-hard for every fixed $d \geq 2$ [Luczak, Noble 2002[3]].
- Çela and Staněk introduce some heuristics for this problem [Çela, Staněk 2013[4]].
- The question about the computational complexity of the DAPT in the case where the guest graph is a tree was open in [Luzcak, Noble 2002[4]].

[4] M.J. Luzcak and S.D. Noble, Optimal arrangement of data in a tree directory, *Discrete Applied Mathematics* **121 (1–3)**, 307–315, 2002.

[5] E. Çela and R. Staněk, Heuristics for the data arrangement problem on regular trees, *Journal of Combinatorial Optimization* **30 (3)**, 768–802, 2015.

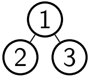
General properties and our special case

- DAPT is $\mathcal{NP}$-hard for every fixed $d \geq 2$ [Luczak, Noble 2002[3]].
- Çela and Staněk introduce some heuristics for this problem [Çela, Staněk 2013[4]].
- The question about the computational complexity of the DAPT in the case where the guest graph is a tree was open in [Luzcak, Noble 2002[4]].
- We deal with the special case where G and T are both binary regular trees.

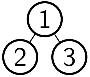
[4]M.J. Luzcak and S.D. Noble, Optimal arrangement of data in a tree directory, *Discrete Applied Mathematics* **121 (1–3)**, 307–315, 2002.

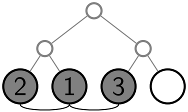
[5]E. Çela and R. Staněk, Heuristics for the data arrangement problem on regular trees, *Journal of Combinatorial Optimization* **30 (3)**, 768–802, 2015.

Solution algorithm



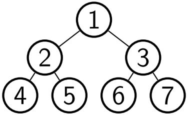
Solution algorithm



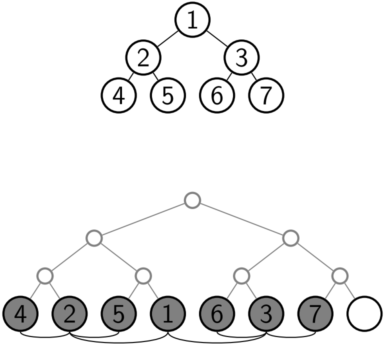


$$OV(G, 2, \phi^*) = 6$$

Solution algorithm

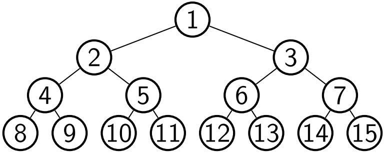


Solution algorithm

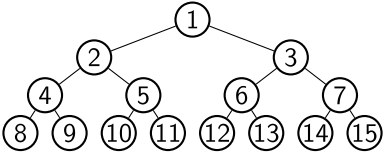


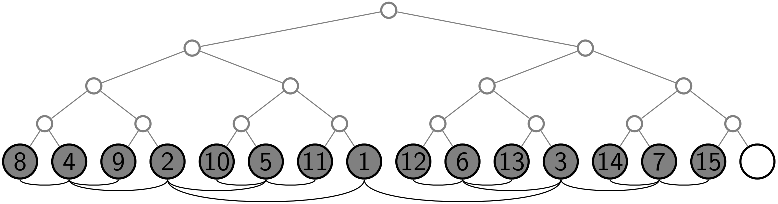
$$OV(G, 2, \phi^*) = 22$$

Solution algorithm



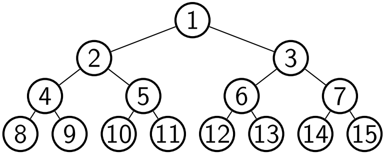
Solution algorithm

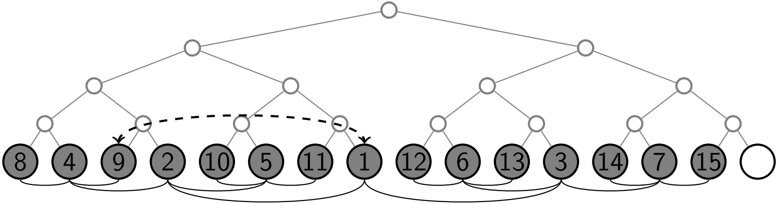




$$OV(G, 2, \phi^*) = 58$$

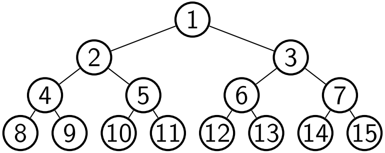
Solution algorithm

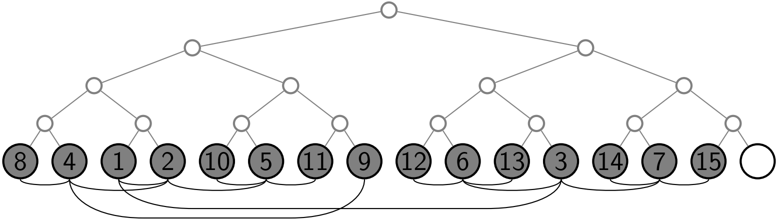




$$OV(G, 2, \phi^*) = 58$$

Solution algorithm





$$OV(G, 2, \phi^*) = 56$$

Solution algorithm

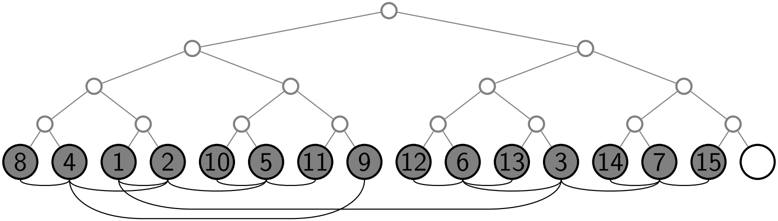
Theorem

Given the binary regular trees $G = (V, E)$ and T with heights h_G and $h = h_G + 1$, let G be the guest graph and T the host graph and let ϕ^* be the arrangement obtained from the described algorithm. Then

$$OV(G, 2, \phi^*) = \begin{cases} 0 & \text{for } h_G = 0 \\ \frac{29}{3} \cdot 2^{h_G} - 4h_G - 9 + \frac{1}{3}(-1)^{h_G} & \text{for } h_G \geq 1 \end{cases} \tag{2}$$

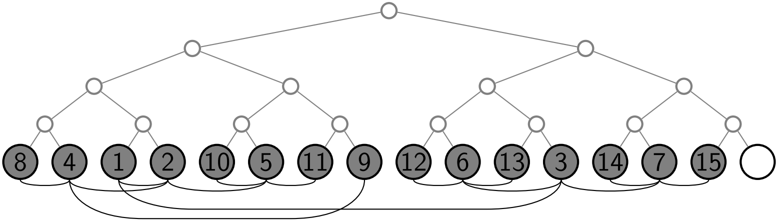
holds.

Lower bound – problem transformation



$$OV(G, 2, \phi^*) = 56$$

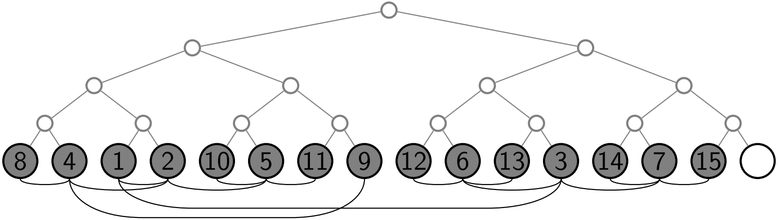
Lower bound – problem transformation



$$OV(G, 2, \phi^*) = 56$$

- $OV(G, 2, \phi) = 2(1 \cdot 4 + 3 \cdot 3 + 5 \cdot 2 + 5 \cdot 1) = 56$

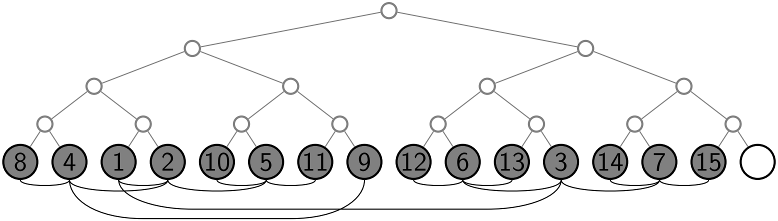
Lower bound – problem transformation



$$OV(G, 2, \phi^*) = 56$$

- $OV(G, 2, \phi) = 2(1 \cdot 4 + 3 \cdot 3 + 5 \cdot 2 + 5 \cdot 1) = 56$
- $OV(G, 2, \phi) = 2\big(a_h(\phi) \cdot h + a_{h-1}(\phi) \cdot (h-1) + \ldots + a_1(\phi) \cdot 1\big)$

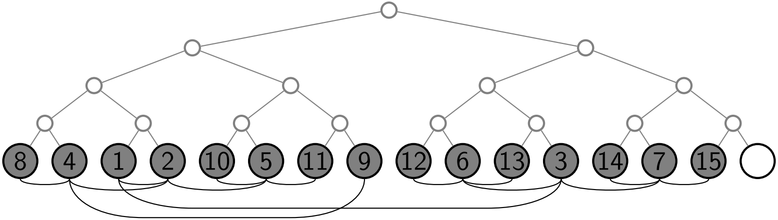
Lower bound – problem transformation



$$OV(G, 2, \phi^*) = 56$$

- $OV(G, 2, \phi) = 2\sum_{i=1}^{h} a_i(\phi) \cdot i$

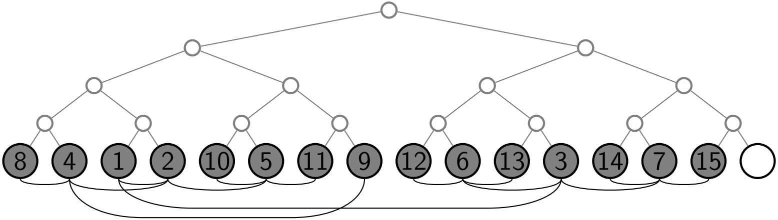
Lower bound – problem transformation



$$OV(G, 2, \phi^*) = 56$$

- $OV(G, 2, \phi) = 2 \sum_{i=1}^{h} a_i(\phi) \cdot i$
- $s_i(\phi) := \sum_{j=i}^{h} a_j(\phi)$ for all $1 \leq i \leq h$

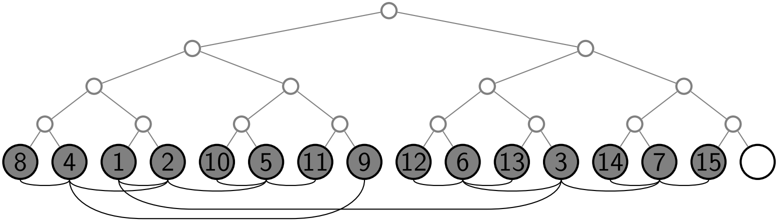
Lower bound – problem transformation



$$OV(G, 2, \phi^*) = 56$$

- $OV(G, 2, \phi) = 2 \sum_{i=1}^{h} a_i(\phi) \cdot i$
- $s_i(\phi) := \sum_{j=i}^{h} a_j(\phi)$ for all $1 \leq i \leq h$
- $a_i(\phi) = \begin{cases} s_i(\phi) - s_{i+1}(\phi) & \text{for } 1 \leq i \leq h-1 \\ s_i(\phi) & \text{for } i = h \end{cases}$

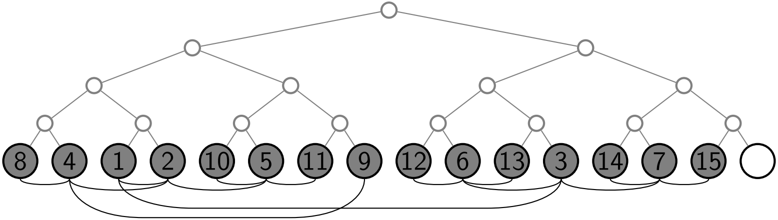
Lower bound – problem transformation



$$OV(G,2,\phi^*) = 56$$

- $OV(G,2,\phi) = 2\sum_{i=1}^{h} a_i(\phi)\cdot i$
- $s_i(\phi) := \sum_{j=i}^{h} a_j(\phi)$ for all $1 \le i \le h$
- $a_i(\phi) = \begin{cases} s_i(\phi) - s_{i+1}(\phi) & \text{for } 1 \le i \le h-1 \\ s_i(\phi) & \text{for } i = h \end{cases}$
- $OV(G,2,\phi) = 2\sum_{i=1}^{h} s_i(\phi)$

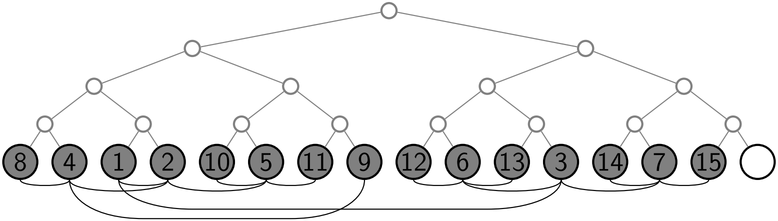
Lower bound – problem transformation



$$OV(G, 2, \phi^*) = 56$$

i	4	3	2	1
▶ a_i	1	3	5	5
s_i	1	4	9	14

Lower bound – problem transformation



$$OV(G, 2, \phi^*) = 56$$

i	4	3	2	1
a_i	1	3	5	5
s_i	1	4	9	14
- $OV(G, 2, \phi) = 2(1 + 4 + 9 + 14) = 56$

Lower bound – problem transformation

- Given
  - an undirected graph $G = (V, E)$,

Lower bound – problem transformation

- Given
  - an undirected graph $G = (V, E)$,
  - a constant $k \geq 2$,

Lower bound – problem transformation

- Given
  - an undirected graph $G = (V, E)$,
  - a constant $k \geq 2$,

  the ***k*-balanced partitioning problem (kBPP)** asks for a partition of the vertex set V into k non-empty vertex sets

Lower bound – problem transformation

- Given
  - an undirected graph $G = (V, E)$,
  - a constant $k \geq 2$,

the **k-balanced partitioning problem (kBPP)** asks for a partition of the vertex set V into k non-empty vertex sets

- $V_1 \neq \emptyset,\ V_2 \neq \emptyset,\ \ldots,\ V_k \neq \emptyset$, where
- $\cup_{i=1}^{k} V_k = V$, $V_i \cap V_j = \emptyset$ for every $i \neq j$ and
- $|V_i| \leq \left\lceil \frac{n}{k} \right\rceil$ for all $1 \leq i \leq k$,

Lower bound – problem transformation

- Given
  - an undirected graph $G = (V, E)$,
  - a constant $k \geq 2$,

the ***k*-balanced partitioning problem (kBPP)** asks for a partition of the vertex set V into k non-empty vertex sets

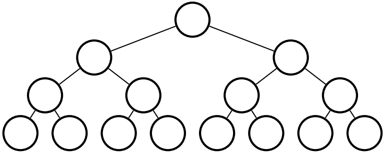
- $V_1 \neq \emptyset,\ V_2 \neq \emptyset,\ \ldots,\ V_k \neq \emptyset$, where
- $\cup_{i=1}^{k} V_k = V,\ V_i \cap V_j = \emptyset$ for every $i \neq j$ and
- $|V_i| \leq \left\lceil \frac{n}{k} \right\rceil$ for all $1 \leq i \leq k$,

such that the number of edges connecting these vertex sets

$$c(G, \mathscr{V}) := \left| \left\{ (u, v) \in E | u \in V_i,\ v \in V_j,\ i \neq j \right\} \right|, \tag{3}$$

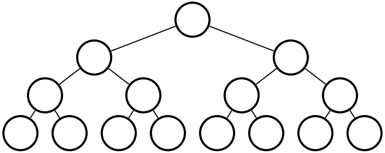
where $\mathscr{V} = \{V_i | 1 \leq i \leq k\}$, is minimized.

Lower bound – problem transformation



i	4	3	2	1
a_i	1	3	5	5
s_i	1	4	9	14

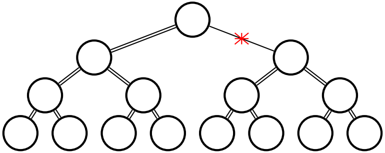
Lower bound – problem transformation



i	4	3	2	1
a_i	1	3	5	5
s_i	1	4	9	14

- It is obvious that $s_i \geq c(G, \mathscr{V})$, where $k = 2^{h-i+2}$ for all $2 \leq i \leq h$ and that $s_1 = \left|E(G)\right|$.

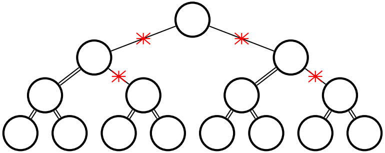
Lower bound – problem transformation



i	4	3	2	1
a_i	1	3	5	5
s_i	1	4	9	14

  , $k = 2^{3-4+2} = 2$
- It is obvious that $s_i \geq c(G, \mathscr{V})$, where $k = 2^{h-i+2}$ for all $2 \leq i \leq h$ and that $s_1 = \left|E(G)\right|$.

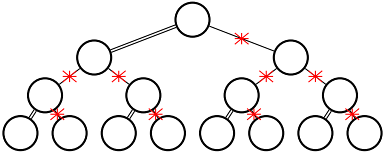
Lower bound – problem transformation



i	4	3	2	1
a_i	1	3	5	5
s_i	1	4	9	14

  , $k = 2^{3-3+2} = 4$
- It is obvious that $s_i \geq c(G, \mathscr{V})$, where $k = 2^{h-i+2}$ for all $2 \leq i \leq h$ and that $s_1 = \left|E(G)\right|$.

Lower bound – problem transformation



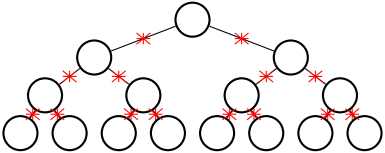
-

i	4	3	2	1
a_i	1	3	5	5
s_i	1	4	9	14

  , $k = 2^{3-2+2} = 8$

- It is obvious that $s_i \geq c(G, \mathscr{V})$, where $k = 2^{h-i+2}$ for all $2 \leq i \leq h$ and that $s_1 = \left|E(G)\right|$.

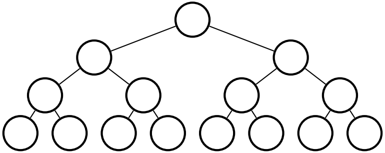
Lower bound – problem transformation



i	4	3	2	1
a_i	1	3	5	5
s_i	1	4	9	14

- It is obvious that $s_i \geq c(G, \mathscr{V})$, where $k = 2^{h-i+2}$ for all $2 \leq i \leq h$ and that $s_1 = \left|E(G)\right|$.

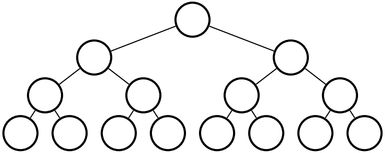
Lower bound – problem transformation



i	4	3	2	1
a_i	1	3	5	5
s_i	1	4	9	14

- It is obvious that $s_i \geq c(G, \mathscr{V})$, where $k = 2^{h-i+2}$ for all $2 \leq i \leq h$ and that $s_1 = \big|E(G)\big|$.
- All but one components have the size $\frac{|V|+1}{k}$.

Lower bound – problem transformation



i	4	3	2	1
a_i	1	3	5	5
s_i	1	4	9	14

- It is obvious that $s_i \geq c(G, \mathscr{V})$, where $k = 2^{h-i+2}$ for all $2 \leq i \leq h$ and that $s_1 = \left|E(G)\right|$.
- All but one components have the size $\frac{|V|+1}{k}$.
- One component has the size $\frac{|V|+1}{k} - 1$.

Lower bound – problem transformation

- kBPP is $\mathcal{NP}$-hard (we get the *minimum bisection problem* which is $\mathcal{NP}$-hard for $k = 2$ [Garey, Johnson 2002[5]]).

[6]M.R. Garey and D.S. Johnson, *Computers and intractability: A guide to the theory of NP-completeness.* Series of books in the mathematical sciences, 1979.

Lower bound – problem transformation

- kBPP is $\mathcal{NP}$-hard (we get the *minimum bisection problem* which is $\mathcal{NP}$-hard for $k = 2$ [Garey, Johnson 2002[5]]).
- Andreev and Räcke prove further complexity results for a generalization allowing near-balanced partitions [Andreev, Räcke 2006[6]].

[6]M.R. Garey and D.S. Johnson, *Computers and intractability: A guide to the theory of NP-completeness.* Series of books in the mathematical sciences, 1979.

[7]K. Andreev and H. Räcke, Balanced Graph Partitioning, *Theory of Computing Systems* **39 (6)**, 929–939, 2006.

Lower bound – problem transformation

- kBPP is $\mathcal{NP}$-hard (we get the *minimum bisection problem* which is $\mathcal{NP}$-hard for $k = 2$ [Garey, Johnson 2002[5]]).
- Andreev and Räcke prove further complexity results for a generalization allowing near-balanced partitions [Andreev, Räcke 2006[6]].
- Krauthgamer, Naor and Schwartz provide an approximation algorithm achieving an approximation of $O(\sqrt{\log n \log k})$ [Krauthgamer, Naor, Schwartz 2009[7]].

[6] M.R. Garey and D.S. Johnson, *Computers and intractability: A guide to the theory of NP-completeness.* Series of books in the mathematical sciences, 1979.

[7] K. Andreev and H. Räcke, Balanced Graph Partitioning, *Theory of Computing Systems* **39 (6)**, 929–939, 2006.

[8] R. Krauthgamer, J. Naor and R. Schwartz, Partitioning graphs into balanced components, *Proceeding SODA '09 Proceedings of the twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, 942–949, 2009.

Lower bound – problem transformation

- kBPP remains $\mathcal{APX}$-hard even if the graph is an unweighted tree with constant maximum degree [FELDMANN, FOSCHINI 2013[8]].

[8] A.E. Feldmann and L. Foschini, Balanced Partitions of Trees and Applications, *Algorithmica* 2013, published online.

Lower bound – problem transformation

- kBPP remains $\mathcal{APX}$-hard even if the graph is an unweighted tree with constant maximum degree [FELDMANN, FOSCHINI 2013[8]].

Theorem

Let $G = (V, E)$ be a binary regular tree of height $h \geq 1$ and let $k = 2^{k'}$, where $1 \leq k' \leq h$, and $\mathscr{V}^*$ an optimal k-balanced partition. Then

$$c(G, \mathscr{V}^*) = \left(3 \cdot 2^{h+1} - 2^{k'+1}\right)\left(\frac{1}{2^s - 1} - \frac{1}{(1 - 2^{-s})\, 2^{sl}}\right) + 3 \cdot 2^{h-sl+1} - 2, \tag{4}$$

where $s = h - k' + 2$ and $l = \left\lfloor \frac{h+1}{s} \right\rfloor$.

[8] A.E. Feldmann and L. Foschini, Balanced Partitions of Trees and Applications, *Algorithmica* 2013, published online.

Lower bound – problem transformation

i	4	3	2	1
a_i	1	3	5	5
s_i	1	4	9	14
$c(G, \mathscr{V}^*)$	1	4	9	14

Lower bound – problem transformation

i	4	3	2	1
a_i	1	3	5	5
s_i	1	4	9	14
$c(G, \mathscr{V}^*)$	1	4	9	14

  $\Rightarrow$ optimality in this case ✓
- $OV(G, 2, \phi^*) = 56$

Lower bound – problem transformation

i	4	3	2	1
a_i	1	3	5	5
s_i	1	4	9	14
$c(G, \mathscr{V}^*)$	1	4	9	14

  $\Rightarrow$ optimality in this case ✓
- $OV(G, 2, \phi^*) = 56$

i	5	4	3	2	1
a_i	1	3	6	10	10
s_i	1	4	10	20	30
$c(G, \mathscr{V}^*)$	1	4	10	20	30

Lower bound – problem transformation

i	4	3	2	1
a_i	1	3	5	5
s_i	1	4	9	14
$c(G, \mathscr{V}^*)$	1	4	9	14

  $\Rightarrow$ optimality in this case ✓
- $OV(G, 2, \phi^*) = 56$

i	5	4	3	2	1
a_i	1	3	6	10	10
s_i	1	4	10	20	30
$c(G, \mathscr{V}^*)$	1	4	10	20	30

  $\Rightarrow$ optimality in this case ✓
- $OV(G, 2, \phi^*) = 130$

Lower bound – problem transformation

i	6	5	4	3	2	1
a_i	1	3	6	12	19	21
s_i	1	4	10	22	41	62
$c(G, \mathscr{V}^*)$	1	4	10	21	41	62

Lower bound – problem transformation

i	6	5	4	3	2	1
a_i	1	3	6	12	19	21
s_i	1	4	10	22	41	62
$c(G, \mathscr{V}^*)$	1	4	10	21	41	62

  $\Rightarrow$ problem ✗
- $278 \leq OV(G, 2, \phi^*) \leq 280$

Lower bound – problem transformation

i	6	5	4	3	2	1
a_i	1	3	6	12	19	21
s_i	1	4	10	22	41	62
$c(G, \mathscr{V}^*)$	1	4	10	21	41	62

  $\Rightarrow$ problem ✗
- $278 \leq OV(G, 2, \phi^*) \leq 280$
- In fact, the lower bound is tight for all $\left\lfloor \frac{h}{2} \right\rfloor + 1 \leq i \leq h$ and for $i = 1$ and $i = 2$.

Lower bound – problem transformation

i	6	5	4	3	2	1
a_i	1	3	6	12	19	21
s_i	1	4	10	22	41	62
$c(G, \mathscr{V}^*)$	1	4	10	21	41	62

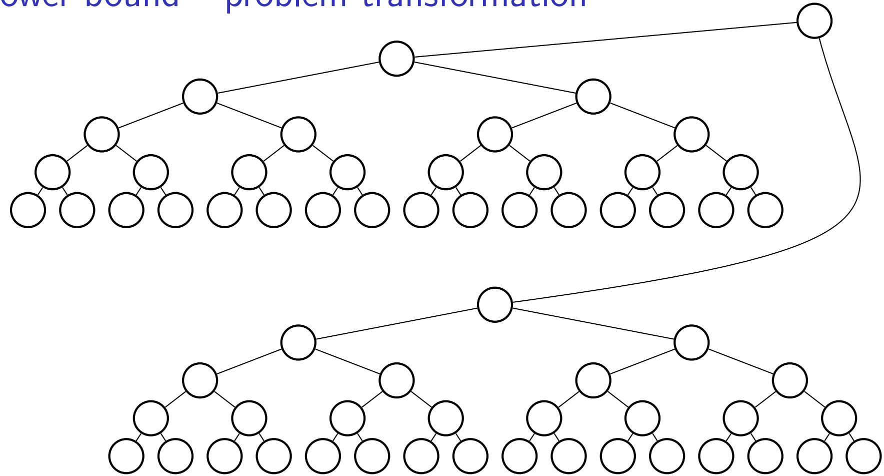
  $\Rightarrow$ problem ✗
- $278 \leq OV(G, 2, \phi^*) \leq 280$
- In fact, the lower bound is tight for all $\left\lfloor \frac{h}{2} \right\rfloor + 1 \leq i \leq h$ and for $i = 1$ and $i = 2$.
- A straightforward analysis yields an approximation ratio 2.

Lower bound – problem transformation

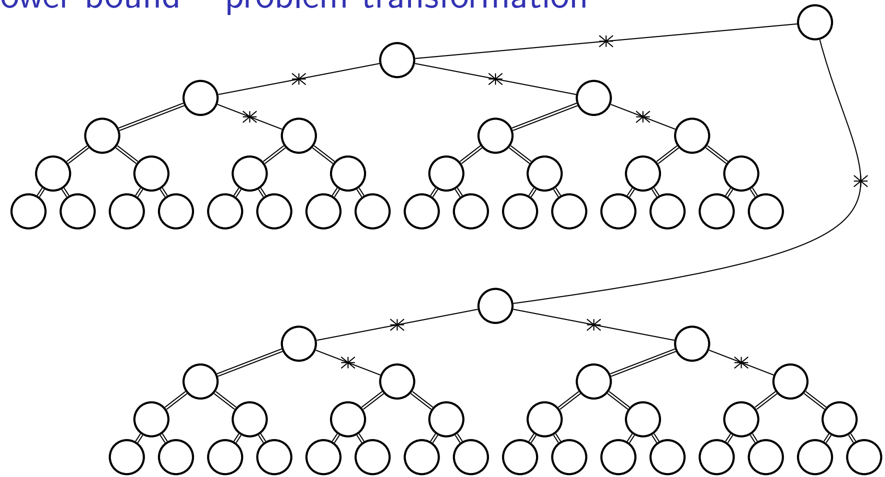
i	6	5	4	3	2	1
a_i	1	3	6	12	19	21
s_i	1	4	10	22	41	62
$c(G, \mathscr{V}^*)$	1	4	10	21	41	62

  $\Rightarrow$ problem ✗
- $278 \leq OV(G, 2, \phi^*) \leq 280$
- In fact, the lower bound is tight for all $\left\lfloor \frac{h}{2} \right\rfloor + 1 \leq i \leq h$ and for $i = 1$ and $i = 2$.
- A straightforward analysis yields an approximation ratio 2.
- A $\frac{203}{200}$-approximation ratio follows by a deeper analysis.

Lower bound – problem transformation

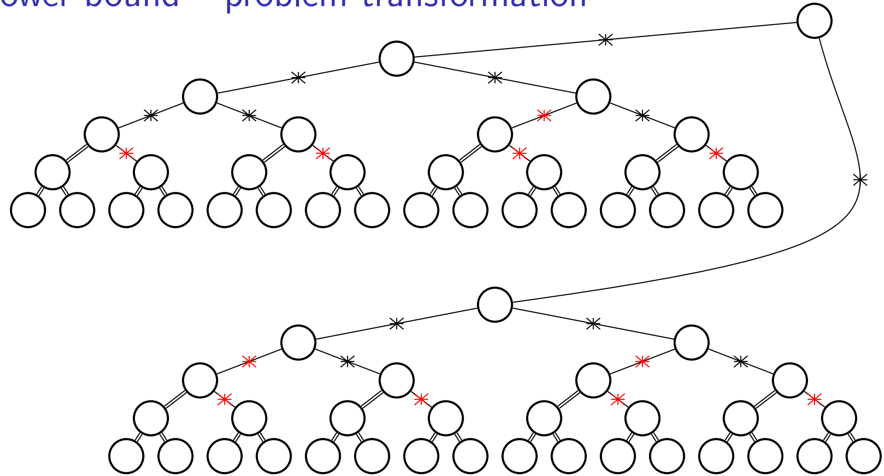


Lower bound – problem transformation



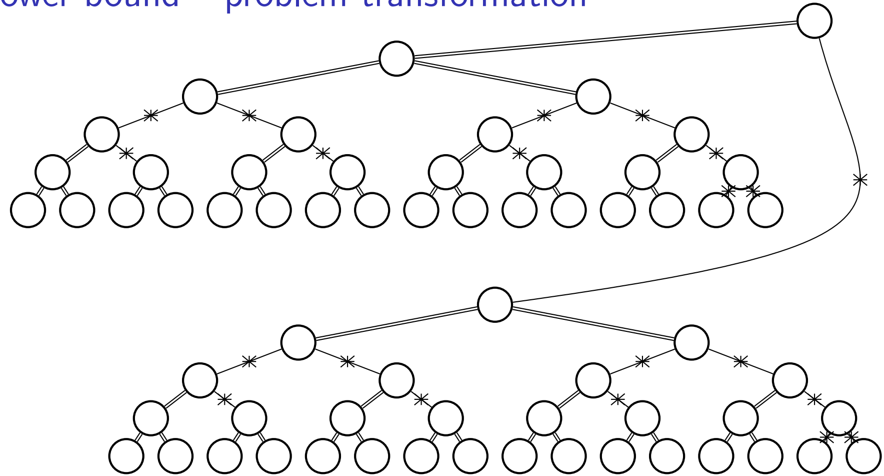
$$c(G, k, \mathscr{V}) = 10$$

Lower bound – problem transformation



$$c(G, k, \mathscr{V}) = 10 + 12 = 22$$

Lower bound – problem transformation



$$c(G, k, \mathscr{V}) = 21$$

Lower bound – problem transformation

- The presented algorithm does not yield an optimal solution for larger guest graphs (there exists a counterexample e.g. for $h_G = 6$ and $h = 7$).

Data arrangement problem

$\mathcal{NP}$-hardness result for trees
proof sketch

Open problem from [LUCZAK, NOBLE 2002]:

Computational complexity when Guest graph is a tree and host is a complete d-regular tree?

Data arrangement problem

$\mathcal{NP}$-hardness result for trees
proof sketch

Open problem from [LUCZAK, NOBLE 2002]:

Computational complexity when Guest graph is a tree and host is a complete *d*-regular tree?

Theorem

The DAPT with a host graph T being a complete d-regular tree is $\mathcal{NP}$-hard for every fixed $d \geq 2$ even if the guest graph G is a tree.

Data arrangement problem

$\mathcal{NP}$-hardness result for trees
proof sketch

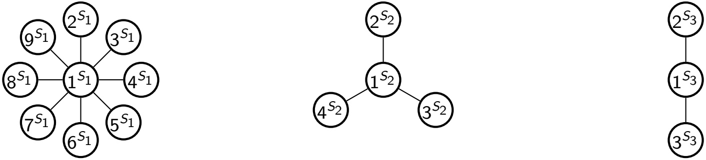
- The $\mathcal{NP}$-hardness is proven by a reduction from the strongly $\mathcal{NP}$-hard **numerical matching with target sums (NMTS)** problem:
  - Given three sets of positive integers
    - $X = \{x_1, x_2, \dots, x_n\}$, $Y = \{y_1, y_2, \dots, y_n\}$ and $Z = \{z_1, z_2, \dots, z_n\}$,

    such that
    - $\sum_{i=1}^{n} z_i = \sum_{i=1}^{n} x_i + \sum_{i=1}^{n} y_i$.
  - The NMTS asks whether there exist two permutations
    - $(j_1, j_2, \dots, j_n)$ and
    - $(k_1, k_2, \dots, k_n)$

    of the indices $\{1, 2, \dots, n\}$, such that
    - $z_i = x_{j_i} + y_{k_i}$ for all $i = 1, 2, \dots, n$.

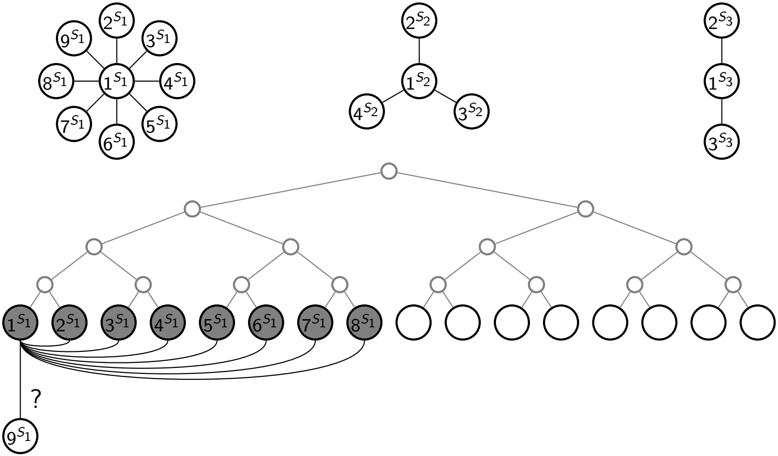
Data arrangement problem

$\mathcal{NP}$-hardness result for trees
proof sketch



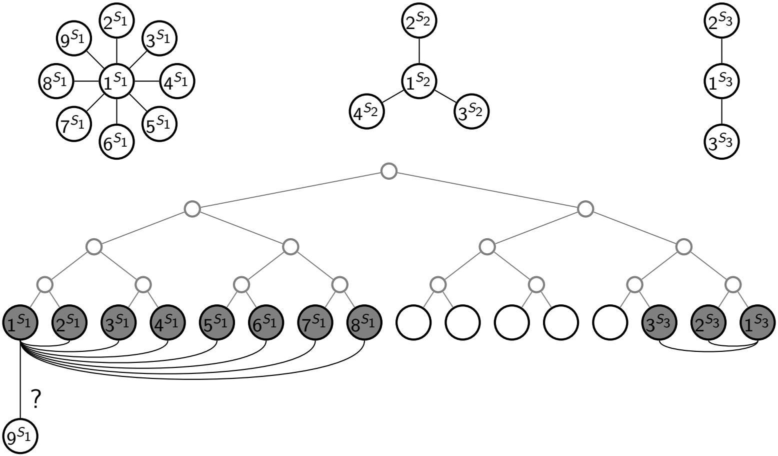
Data arrangement problem

$\mathcal{NP}$-hardness result for trees
proof sketch



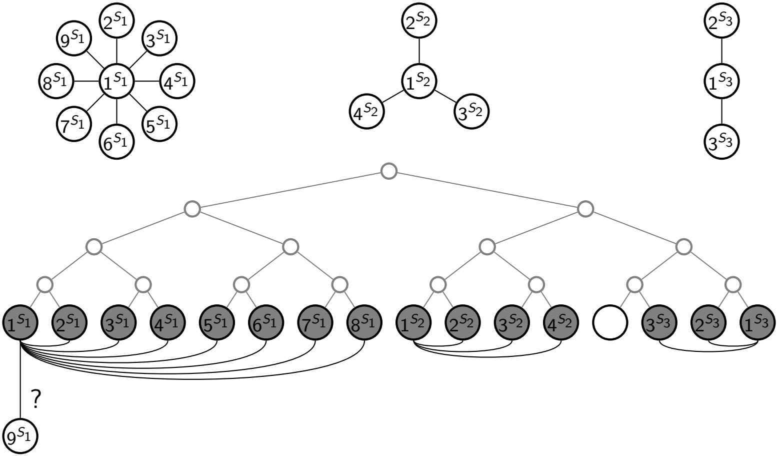
Data arrangement problem

$\mathcal{NP}$-hardness result for trees
proof sketch



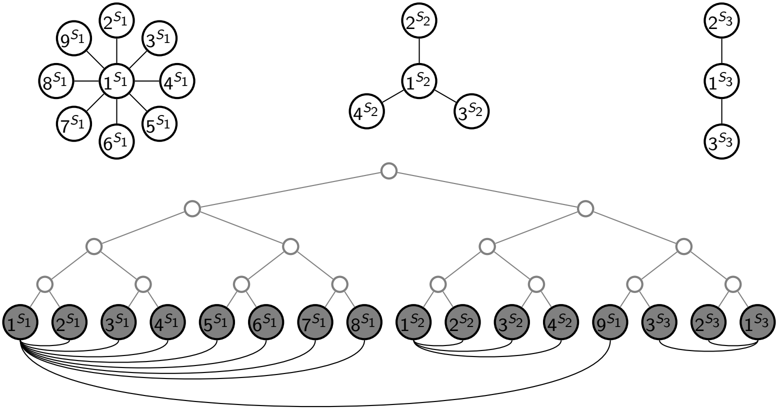
Data arrangement problem

$\mathcal{NP}$-hardness result for trees
proof sketch



Data arrangement problem

$\mathcal{NP}$-hardness result for trees
proof sketch



Recapitulation, future research and open questions

- We provide an approximation algorithm for one (very) special case of the DAPT.

Recapitulation, future research and open questions

- We provide an approximation algorithm for one (very) special case of the DAPT.
- We would like to get the optimal algorithm or prove that the problem is hard (which seems unlikely).

Recapitulation, future research and open questions

- We provide an approximation algorithm for one (very) special case of the DAPT.
- We would like to get the optimal algorithm or prove that the problem is hard (which seems unlikely).

Thank you for your attention!