Communications in Combinatorics and Optimization
Vol. xx, No. x (xxxx), pp. 1-10
https://doi.org/10.22049/cco.2024.29114.1874

CCO
Commun. Comb. Optim.

Research Article

Algorithm for describing the Terwilliger and quantum adjacency algebras of a distance-regular graph

Abdillah Ahmad[1,*], John Vincent Morales[2], Pritta Etriana Putri[3]

[1] Master Program in Mathematics, Faculty of Mathematics and Natural Sciences, Institut Teknologi Bandung, Bandung, Indonesia
20123015@mahasiswa.itb.ac.id

[2] Department of Mathematics and Statistics, De La Salle University, Manila, Philippines
john.vincent.morales@dlsu.edu.ph

[3] Combinatorial Mathematics Research Group, Faculty of Mathematics and Natural Sciences, Institut Teknologi Bandung, Bandung, Indonesia
etrianaputri@itb.ac.id

Received: 30 October 2023; Accepted: 13 December 2024
Published Online: 26 December 2024

Abstract: In this paper we consider an algorithm for determining a basis for the Terwilliger and quantum adjacency algebras of a distance-regular graph. For the Terwilliger algebra, we consider the generating set. For the quantum adjacency algebra, we consider the generating set consisting of the raising, flat, and lowering matrices. We give optimization method by using generating matrices with a block-matrix structure so that the number of matrix multiplications required is reduced.

Keywords: distance-regular graphs, Terwilliger algebra, subconstituent algebra, quantum decomposition, algorithm optimization.

AMS Subject classification: 05C85, 68R10

1. Introduction

In [10], Terwilliger defined the subconstituent algebra, also known as the Terwilliger algebra, of a commutative association scheme. It is generated by the Bose-Mesner algebra and the dual Bose-Mesner algebra of the association scheme.
In terms of graphs, the original definition of Terwilliger algebra only covers distance-regular graphs, which are equivalent to P-polynomial association schemes [2]. The

* *Corresponding Author*

class of distance-regular graphs includes strongly regular graphs and therefore Moore graphs. Strongly regular graphs, in turn, are related to combinatorial designs [5].
In [11], the definition of Terwilliger algebra was extended to cover all finite, undirected, connected, simple graphs. The generalized Terwilliger algebra has been used to analyze combinatorial properties of graphs [3, 4, 8] and properties of their automorphism groups [6, 9, 12].
The Terwilliger algebra of a graph has a natural subalgebra called the quantum adjacency algebra, introduced in [7], generated by the quantum decomposition of the adjacency matrix (viz. the raising, flat, and lowering matrices). In some graphs, the quantum adjacency algebra is equal to the Terwilliger algebra. Such equality would make the Terwilliger algebra 3-generated (and even 2-generated for bipartite graphs). In [11], some conditions were proved to be equivalent to the equality of the Terwilliger and quantum adjacency algebras. Those conditions involve isomorphism classes of modules over the two algebras. In the same paper, the equality has been established for Hamming graphs and the inequality has been established for bipartite dual polar graphs.
The aim of this paper is to give an algorithmic approach to the problem of determining the equality of the Terwilliger and quantum adjacency algebras for distance-regular graphs. The naive algorithm (which always terminates) is first considered, then some optimizations are used to reduce computation time. Such an optimized algorithm would aid the computational study of distance-regular graphs through the use of the two algebras.

2. Basic concepts

Let $G = (X, E)$ be a finite, undirected, connected, simple graph with order n, diameter D, and i-th distance matrix A_i for $i = 0, 1, \ldots, D$. Fix any $x \in X$. For each i $(0 \leq i \leq D)$, define the i^{th} *dual idempotent* $E_i^*(x)$ (henceforth written simply E_i^*, as x is fixed) of G to be the diagonal matrix in $Mat_X(\mathbb{C})$ where

$$(E_i^*)_{yy} = (A_i)_{xy} = \begin{cases} 1 & \text{if } d(x,y) = i, \\ 0 & \text{if } d(x,y) \neq i. \end{cases}$$

From the definition, the following properties can be derived:

(i) $\overline{E_i^*}^t = E_i^*$ $(0 \leq i \leq D)$;

(ii) $E_i^* E_j^* = \delta_{ij} E_i^*$ $(0 \leq i, j \leq D)$ where $\delta_{ij} = \begin{cases} 1, & \text{if } i = j \\ 0, & \text{if } i \neq j \end{cases}$;

(iii) $E_0^* + E_1^* + \cdots + E_D^* = I$;

It can be concluded that $E_0^*, E_1^*, \ldots, E_D^*$ form a basis for a subalgebra $M^*(x) = M^*$ of $Mat_X(\mathbb{C})$, called the *dual Bose-Mesner algebra* of G with respect to x.

The *Terwilliger algebra* $T(x) = T$ of G with respect to x is the subalgebra of $Mat_X(\mathbb{C})$ generated by the adjacency matrix $A = A_1$ and the dual Bose-Mesner algebra M^* of G.

The graph G is said to be *distance-regular* whenever for every $i, j \in \{0, 1, \ldots, D\}$, the matrix $A_i A_j$ is a linear combination of $A_0, A_1, \ldots, A_D$, that is,

$$A_i A_j = \sum_{h=0}^{D} p_{ij}^h A_h.$$

Combinatorially, the above equation means that for every $y, z \in X$, if $d(y, z) = h$, then there are exactly p_{ij}^h vertices $w \in X$ such that $d(y, w) = i$ and $d(w, z) = j$.

If G is distance-regular, then $\{A_0, A_1, \ldots, A_D\}$ is linearly independent and its span is closed under multiplication, so they form a basis for a subalgebra M of the algebra $\mathrm{Mat}_X(\mathbb{C})$ consisting of complex matrices with coordinates indexed by X. Further, we call M the *Bose-Mesner algebra* of G, which is none other than the adjacency algebra of the graph. The Terwilliger algebra T is then also generated by the algebras M and M^*.

Next, define the raising, flat, and lowering matrices of a (not necessarily distance-regular) graph G with respect to x (respectively denoted $R(x) = R, F(x) = F, L(x) = L$) as follows:

$$R = \sum_{i=0}^{D} E_{i+1}^* A E_i^*, \quad F = \sum_{i=0}^{D} E_i^* A E_i^*, \quad L = \sum_{i=0}^{D} E_{i-1}^* A E_i^*.$$

Note that the matrices R, F, L have real entries. Furthermore, F is symmetric (in fact, by reordering the elements of X we can make F a block-diagonal matrix with each block symmetric), $R = L^t$, and $A = R + F + L$.

The *quantum adjacency algebra* $Q(x) = Q$ of graph G is defined as the algebra generated by the raising, flat, and lowering matrices R, F, L.

3. Naive algorithm for determining a basis for T and Q

Let $\mathcal{A}$ be a finite-dimensional algebra. Consider the following algorithm:

In the above algorithm, after every iteration of the while loop, S' will increase in cardinality but remain linearly independent, until it cannot increase any more because the product of every two elements of S' is already a linear combination of the elements of S'. When the algorithm terminates, $B = S'$ will span $\mathcal{A}$ as every element of $\mathcal{A}$ can be written as a polynomial in terms of the elements of B, and can therefore be written as the sum of linear combinations of the elements of B.

The above algorithm is very general (can be applied to any finite-dimensional algebra, including T and Q of a non-distance-regular graph), but very unoptimized. It multiplies all pairs of elements of S' in each iteration of the while loop, including the

Algorithm 1 Naive general algorithm for determining a basis for a finite-dimensional algebra.

Require: $r \in \mathbb{N}$; $S = \{a_1, a_2, \ldots, a_r\} \subseteq \mathcal{A}$ linearly independent and generates $\mathcal{A}$ as an algebra
Ensure: B is a basis of $\mathcal{A}$ (linearly independent and spans $\mathcal{A}$)
1: $S', S'' \leftarrow S$
2: **if** $1_{\mathcal{A}} \notin \text{span}(S')$ **then**
3: $\quad S' \leftarrow S' \cup \{1_{\mathcal{A}}\}$
4: **end if**
5: **while** $\text{span}(S') \neq \mathcal{A}$ **do**
6: $\quad r', r'' \leftarrow |S'|$
7: $\quad$ **for all** $i, j \in [1, r']_{\mathbb{Z}}$ **do**
8: $\quad\quad$ **if** $a_i a_j \notin \text{span}(S'')$ **then**
9: $\quad\quad\quad r'' \leftarrow r'' + 1$
10: $\quad\quad\quad a_{r''} \leftarrow a_i a_j$
11: $\quad\quad\quad S'' \leftarrow S'' \cup \{a_{r''}\}$
12: $\quad\quad$ **end if**
13: $\quad$ **end for**
14: $\quad S' \leftarrow S''$
15: **end while**
16: $B \leftarrow S'$

pairs whose products are already calculated in the previous iteration. To avoid that, we can skip multiplications that are already performed.

To algorithmically implement the determination of $\text{span}(S') \neq \mathcal{A}$, we can use a variable d (standing for "done") to end the while loop if the product of every two elements of S' is a linear combination of the elements of S'.

Now, how do we implement the determination of $1_{\mathcal{A}} \notin \text{span}(S')$ and $a_i a_j \notin \text{span}(S'')$? We do not yet have a basis for $\mathcal{A}$, so we cannot use that. Let us use the fact that T and Q are subalgebras of $\text{Mat}_X(\mathbb{C})$, for which we have the standard basis of cardinality $|X|^2$. Furthermore, we can define an inner product on $\text{Mat}_X(\mathbb{C})$ with $\langle B_1, B_2 \rangle = \tau(B_1 \circ \overline{B_2})$ for every $B_1, B_2 \in \mathcal{A}$, where τ denotes the entry sum of a matrix and $\circ$ denotes the Hadamard product of matrices.

Then, if we require that S be an orthonormal set, we can determine whether an element B of $\mathcal{A}$ is in the span of S' or S'' by projecting it onto the span. If the projection is equal to B, then B is in the span; otherwise it is outside the span. Therefore, B is in the span of V if and only if $B - \text{proj}_V(B) = 0$.

If $\mathcal{A} = Q$, we can use $S = \{R/\|R\|, F/\|F\|, L/\|L\|, I/\sqrt{n}\}$ if the graph G is not bipartite or $S = \{R/\|R\|, L/\|L\|, I/\sqrt{n}\}$ if the graph G is bipartite (as $F = 0$ if and only if G is bipartite). If $\mathcal{A} = T$ and the graph G is distance-regular, let $H = \{E_i^* A_h E_j^* \mid h, i, j \in [0, D]_{\mathbb{Z}}\}$ and $S = \{B/\|B\| \mid B \in H \setminus \{0\}\}$. In both cases, S is orthonormal and generates $\mathcal{A}$, and the identity matrix $I = 1_{\mathcal{A}}$ is always in the span of S.

After the aforementioned changes, the algorithm becomes as follows:

The time complexity of the above algorithm can be calculated as follows. The relevant lines are 7 (matrix multiplication), 9 (inner product), and 12 (norm calculation).

- Line 7 is executed $\dim(\mathcal{A})^2$ times, each with (asymptotic) complexity $O(n^{2.3728596})$ ([1]). If we use the more practical Strassen's algorithm, each

Algorithm 2 The improvement of Algorithm 1.

Require: $\mathcal{A} \in \{T, Q\}$; $S = \begin{cases} \{R/\|R\|, F/\|F\|, L/\|L\|, I/\sqrt{n}\} & \text{if } \mathcal{A} = Q \text{ and } G \text{ is not bipartite} \\ \{R/\|R\|, L/\|L\|, I/\sqrt{n}\} & \text{if } \mathcal{A} = Q \text{ and } G \text{ is bipartite} \\ \{B/\|B\| \mid B \in H \setminus \{0\}\} & \text{if } \mathcal{A} = T \end{cases}$

Ensure: B is a basis of $\mathcal{A}$

1: $S', S'' \leftarrow S$
2: $r'_{\text{prev}} \leftarrow 0$
3: $d \leftarrow \mathit{false}$
4: **while** $d = \mathit{false}$ **do**
5: $\quad r', r'' \leftarrow |S'|$
6: $\quad$ **for all** $(i, j) \in ([1, r']_{\mathbb{Z}})^2 \setminus ([1, r'_{\text{prev}}]_{\mathbb{Z}})^2$ **do**
7: $\quad\quad a \leftarrow a_i a_j$
8: $\quad\quad$ **for all** $k \in [1, r'']_{\mathbb{Z}}$ **do**
9: $\quad\quad\quad a \leftarrow a - \langle a, a_k \rangle a_k$
10: $\quad\quad$ **end for**
11: $\quad\quad$ **if** $a \neq 0$ **then** ▷ $a \notin \operatorname{span}(S'')$
12: $\quad\quad\quad r'' \leftarrow r'' + 1$
13: $\quad\quad\quad a_{r''} \leftarrow a/\|a\|$
14: $\quad\quad\quad S'' \leftarrow S'' \cup \{a_{r''}\}$
15: $\quad\quad$ **end if**
16: $\quad$ **end for**
17: $\quad$ **if** $r'' = r'$ **then**
18: $\quad\quad d \leftarrow \mathit{true}$
19: $\quad$ **end if**
20: $\quad r'_{\text{prev}} \leftarrow r'$
21: $\quad S' \leftarrow S''$
22: **end while**
23: $B \leftarrow S'$

matrix multiplication has complexity $O(n^{\log_2 7}) \approx O(n^{2.807})$.

- For every $(i, j) \in [1, \dim(\mathcal{A})]_{\mathbb{Z}}$, line 9 is executed r'' times; the value of r'' may change with (i, j) and is bounded above by $\dim(\mathcal{A})$. The number of executions of line 9 is thus bounded above by $\dim(\mathcal{A})^3$. Each execution of line 9 has complexity $O(n^2)$.
- Line 12 is performed $\dim(\mathcal{A}) - r$ times, each with complexity $O(n^2)$. It is therefore overshadowed by line 7.

From lines 7 and 9, we calculate an upper bound for the time complexity of the algorithm, if $\dim(\mathcal{A})$ is known:

$$k_1 \dim(\mathcal{A})^2 n^{2.3728596} + k_2 \dim(\mathcal{A})^3 n^2$$

As $\dim(\mathcal{A})$ is bounded above by n^2, we obtain the time complexity figure of $O(n^c)$ where

$$2.3728596 \leq c \leq 8,$$

or, with Strassen's algorithm,

$$2.807 \leq c \leq 8.$$

The complexity approaches the lower bound when $\dim(\mathcal{A}) << n$, and it approaches the upper bound when $\log_n \dim(\mathcal{A}) \approx 2$.

4. Block structure optimization for the algorithm for T

Let $B \in T$. Consider that if we sort the vertex set X of a distance-regular graph G by ascending distance from the fixed point x (by applying a permutation, say $\sigma : X \to X$), then B will be mapped to $[\sigma]B[\sigma]$, which is equivalent to B.
From now on, let us assume without loss of generality that the vertices of G are already sorted in ascending distance from x. Let k_i be the number of vertices of G with distance i from x. The dual idempotents will have the form

$$E_i^* = \operatorname{diag}(\delta_{0i}, \underbrace{\delta_{1i}, \ldots, \delta_{1i}}_{k_1}, \underbrace{\delta_{2i}, \ldots, \delta_{2i}}_{k_2}, \ldots, \underbrace{\delta_{Di}, \ldots, \delta_{Di}}_{k_D}).$$

Consider an arbitrary $B \in T$ with

$$B = \left[\begin{array}{c|c|c|c} B_{00} & B_{01} & \ldots & B_{0D} \\ \hline B_{10} & B_{11} & \ldots & B_{1D} \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline B_{D0} & B_{D1} & \ldots & B_{DD} \end{array}\right]$$

where B_{ij} is a block of size $k_i \times k_j$ for every $i, j \in [0, D]_{\mathbb{Z}}$. We call B_{ij} the (i, j)**-block** of B. For every $i, j \in [0, D]_{\mathbb{Z}}$ we have

$$E_i^* B E_j^* = \left[\begin{array}{c|c|c|c|c} 0 & \ldots & 0 & \ldots & 0 \\ \hline \vdots & \ddots & \vdots & \ddots & \vdots \\ \hline 0 & \ldots & B_{ij} & \ldots & 0 \\ \hline \vdots & \ddots & \vdots & \ddots & \vdots \\ \hline 0 & \ldots & 0 & \ldots & 0 \end{array}\right]$$

Define (i, j)-*monoblocks* to be matrices of the form $E_i^* B E_j^*$ where $B \in T$.
Let $B \in T$ be a (i, j)-monoblock and $B' \in T$ be a (j', k)-monoblock. We say that B *is compatible with* B' if $j = j'$. The compatibility relation is not symmetric; a $(1, 2)$-monoblock is compatible with a $(2, 3)$-monoblock, but not vice versa.
Now let $B, B' \in T$ be monoblocks. Because the dual idempotents are orthogonal ($E_i^* E_j^* = 0$ if $i \neq j$), the matrix BB' can be nonzero only if B is compatible with B'.
We consider Algorithm 2 where $\mathcal{A} = T$ (using $S = \{h/\|h\| \mid h \in H\}$). By definition, every element of $S = \{B/\|B\| \mid B \in H \setminus \{0\}\}$ is a monoblock, and likewise, the product of any number of elements of S is also a monoblock. Therefore, throughout the algorithm, all elements of the set S' are always monoblocks.
Let S'_{ij} be the set of all elements of S' with nonzero (i, j)-block. (The sets S'_{ij} change as S' changes.) Then $\{S'_{ij} \mid i, j \in [0, D]_{\mathbb{Z}}\}$ partitions S'. We obtain the following:

Proposition 1. *If $B, B' \in S'$ such that $BB' \neq 0$, then $B \in S'_{ij}, B' \in S'_{jk}$ for some $i, j, k \in [0, D]_{\mathbb{Z}}$.*

Algorithm 3 The optimized algorithm for T.

Require: $S = \{B/\|B\| \mid B \in H \setminus \{0\}\}$, where $H = \{E_i^* A_h E_j^* \mid h, i, j \in [0, D]_{\mathbb{Z}}\}$
Ensure: B is a basis of T

$S', S'' \leftarrow S$
$r'_{\text{prev}} \leftarrow 0$
$d \leftarrow$ *false*
while $d =$ *false* **do**
  $r', r'' \leftarrow |S'|$
  for all $i, j, k \in [0, D]_{\mathbb{Z}}$ **do**
    for all $(a', a'') \in S'_{ij} \times S'_{jk}$ such that the multiplication $a'a''$ has not been calculated **do**
      $a \leftarrow a'a''$
      for all $\tilde{a} \in S'_{ik}$ **do**
        $a \leftarrow a - \langle a, \tilde{a} \rangle \tilde{a}$
      end for
      if $a \neq 0$ **then**
        $r'' \leftarrow r'' + 1$
        $a_{r''} \leftarrow a/\|a\|$
        $S'' \leftarrow S'' \cup \{a_{r''}\}$
      end if
    end for
  end for
  if $r'' = r'$ **then**
    $d \leftarrow$ *true*
  end if
  $r'_{\text{prev}} \leftarrow r'$
  $S' \leftarrow S''$
end while
$B \leftarrow S'$

We calculate the time complexity of the above algorithm as follows. Let $C = (c_{ij})$ be such that $c_{ij} = |S'_{ij}|$. The optimization reduces the number of matrix multiplications to $\tau(C^2) = \text{tr}(C^2 J) \leq \text{tr}(C)\text{tr}(CJ) = \text{tr}(C)r' \leq \dim(\mathcal{A})^2 \leq n^4$. The number of inner product calculations ($\langle a, \tilde{a} \rangle$) is reduced to $(D+1)\text{tr}(CJ) = (D+1)r' \leq (D+1)\dim(\mathcal{A}) \leq n^3$. We obtain the time complexity figure of $O(n^c)$ where

$$2.3728596 \leq c \leq 6.3728596. \tag{4.1}$$

With Strassen's algorithm, we obtain

$$2.807 \leq c \leq 6.807.$$

5. Optimizations for the algorithm for Q

As discussed before, for distance-regular graphs, every element of the set $\{B/\|B\| \mid B \in H \setminus \{0\}\}$ only has one nonzero block, which enabled us to optimize the algorithm

for $\mathcal{A} = T$. On the other hand, for a general graph, R, L can be written as the sum of D monoblocks, I can be written as the sum of $D+1$ monoblocks, and F can be written as the sum of $\leq D+1$ monoblocks.

Therefore, we can optimize Algorithm 2 for $\mathcal{A} = Q$ by reducing the number of operations in the calculation of matrix product by only multiplying compatible monoblocks in the decomposition of each matrix. We can also reduce the number of operations in the calculation of inner product by only multiplying monoblocks with the same position ((i,j)-monoblock with (i,j)-monoblock). This optimization is not restricted to distance-regular graphs.

More specifically, consider that every element of $S \in \{\{R/\|R\|, F/\|F\|, L/\|L\|, I/\sqrt{n}\}, \{R/\|R\|, L/\|L\|, I/\sqrt{n}\}\}$ can be written as the sum of $\leq D+1$ monoblocks with different horizontal and vertical positions (for instance, $I/\sqrt{n}$ can be written as the sum of the matrices $E_0^*/\sqrt{n}, E_1^*/\sqrt{n}, \ldots, E_D^*/\sqrt{n}$, which are respectively a $(0,0)$-monoblock, $(1,1)$-monoblock, ..., and (D,D)-monoblock). From there we can inductively obtain that the product of finitely many elements of S (not necessarily distinct) can also be written as the sum of $\leq D+1$ monoblocks with different horizontal and vertical positions.

Therefore, the number of operations in a matrix multiplication in the algorithm (where $\mathcal{A} = Q$) is bounded above by $(D+1)\max_i k_i^{2.3728596}$, whereas the number of operations in an inner product calculation in the algorithm is bounded above by $(D+1)\max_i k_i^2$. Generally, they are bounded above by $n^{2.3728596}$ and n^2, and therefore the optimization reduces the coefficient of the time complexity.

6. Example and results of algorithm implementation

We consider as an example the working of Algorithm 3 for the Terwilliger algebra of the Petersen graph. The following chart shows the 14 elements of S, divided by monoblock positions, where ${}_i[h]_j := \frac{E_i^* A_h E_j^*}{\|E_i^* A_h E_j^*\|}$.

$$\left[\begin{array}{c|c|c} {}_0[0]_0 & {}_0[1]_1 & {}_0[2]_2 \\ \hline {}_1[1]_0 & {}_1[0]_1, {}_1[2]_1 & {}_1[1]_2, {}_1[2]_2 \\ \hline {}_2[2]_0 & {}_2[1]_1, {}_2[2]_1 & {}_2[0]_2, {}_2[1]_2, {}_2[2]_2 \end{array}\right].$$

The algorithm runs without finding a new element of S'' until it reaches $i = k = 2$, $j = 1$, $a = {}_2[1]_1$, and $a' = {}_1[1]_2$. At that point, we obtain a new element $M/\|M\|$ of S'', where $M = M_2 - \langle M_2, {}_2[2]_2\rangle {}_2[2]_2$, $M_2 = M_1 - \langle M_1, {}_2[1]_2\rangle {}_2[1]_2 = M_1 - 0 \cdot {}_2[1]_2 = M_1$, and $M_1 = aa' - \langle aa', {}_2[0]_2\rangle {}_2[0]_2$. The rest of the algorithm runs without finding any other element of S''. Therefore, we conclude that the dimension of the Terwilliger algebra is 15, with basis $B = S \cup \{M/\|M\|\}$.

Below is a table detailing the computational times, in seconds, of applying Algorithm 2 to Q (with and without optimization) and applying Algorithms 2 and 3 to T, in the context of the odd graphs O_3 (Petersen graph), O_4, and O_5. The exponents c and coefficients k of the time complexities kn^c for each algorithm were obtained by exponential regression.

Algorithm	O_3	O_4	O_5	c	$k \cdot 10^6$
Algorithm 2 for Q	0.044	5.111	763.625	3.853	6.026
Optimized Algorithm 2 for Q	0.013	0.968	184.858	3.775	1.894
Algorithm 2 for T	0.043	5.045	759.505	3.860	5.798
Algorithm 3 for T	0.004	0.140	16.845	3.295	1.671
Order of graph	10	35	126		

Table 1. Computational times of applying the algorithms to odd graphs.

We observe that the optimization for Algorithm 2 for Q reduces the time complexity exponent slightly and reduces the time complexity coefficient significantly, whereas Algorithm 3 for T reduces both the exponent and coefficient significantly with respect to Algorithm 2.

7. Conclusion and open problems

We have given an optimized algorithm for determining T in distance-regular graphs and optimizations for determining Q in general graphs. We also calculated the algorithmic time complexity both analytically and experimentally for odd graphs.
There are still some problems that remain:

- Can we obtain a tighter upper bound for (4.1)?
- What conditions of distance-regular graphs enable tighter upper bounds for (4.1)?
- Is there a more efficient algorithm for determining the basis of the Terwilliger and quantum adjacency algebras of a distance-regular graph?
- What optimizations can we do for determining the basis of the Terwilliger and quantum adjacency algebras of a non-distance-regular graph?

Acknowledgements: This research was funded by Penelitian, Pengabdian kepada Masyarakat dan Inovasi Kelompok Keilmuan (PPMI-KK) 2024 Grant No. FMIPA.PPMI-KK-PN-36-2024.

Conflict of Interest: The authors declare that they have no conflict of interest.

Data Availability: Data sharing is not applicable to this article as no datasets were generated or analyzed during the current study.

References

[1] J. Alman and V.V. Williams, *A refined laser method and faster matrix multiplication*, Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms

(SODA) (D. Marx, ed.), SIAM, 2024, pp. 522 – 539.
https://doi.org/10.1137/1.9781611976465.32.

[2] E. Bannai and T. Ito, *Algebraic Combinatorics I: Association Schemes*, Benjamin/Cummings, 1984.

[3] B. Fernández and Š. Miklavič, *On the terwilliger algebra of distance-biregular graphs*, Linear Algebra Appl. **597** (2020), 18–32.
https://doi.org/10.1016/j.laa.2020.03.016.

[4] ———, *On bipartite graphs with exactly one irreducible T-module with endpoint 1, which is thin*, European J. Combin. **97** (2021), 103387.
https://doi.org/10.1016/j.ejc.2021.103387.

[5] J.M. Goethals and J.J. Seidel, *Strongly regular graphs derived from combinatorial designs*, Canad. J. Math. **22** (1970), no. 3, 597–614.
https://doi.org/10.4153/CJM-1970-067-9.

[6] A. Hanaki and M. Yoshikawa, *Terwilliger algebras and some related algebras defined by finite connected simple graphs*, Discrete Math. **346** (2023), no. 9, 113509.
https://doi.org/10.1016/j.disc.2023.113509.

[7] A. Hora and N. Obata, *Quantum Probability and Spectral Analysis of Graphs*, Springer, Berlin, 2007.

[8] S.D. Li, Y.Z. Fan, T. Ito, M. Karimi, and J. Xu, *The isomorphism problem of trees from the viewpoint of terwilliger algebras*, J. Combin. Theory Ser. A **177** (2021), 105328.
https://doi.org/10.1016/j.jcta.2020.105328.

[9] Y. Tan, Y. Zhang, T. Xia, and X. Liang, *Terwilliger algebras of a fan graph*, Journal of Hefei University of Technology Natural Science Edition **46** (2023), no. 3, 419–425.

[10] P. Terwilliger, *The subconstituent algebra of an association scheme,(part I)*, J. Algebraic Combin. **1** (1992), no. 4, 363–388.
https://doi.org/10.1023/A:1022494701663.

[11] P. Terwilliger and A. Žitnik, *The quantum adjacency algebra and subconstituent algebra of a graph*, J. Combin. Theory Ser. A **166** (2019), 297–314.
https://doi.org/10.1016/j.jcta.2019.02.022.

[12] J. Xu, T. Ito, and S.D. Li, *Irreducible representations of the Terwilliger algebra of a tree*, Graphs Combin. **37** (2021), no. 5, 1749–1773.
https://doi.org/10.1007/s00373-021-02384-9.