# Application of Welsh-Powell Algorithm for Instrument Frequency Separation in Audio Mixing

Ryuza Nadif Aldebaran - 13525026
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail: ryu.aldebaran@gmail.com , 13525026@std.stei.itb.ac.id

***Abstract*—In audio mixing, auditory masking occurs when multiple instruments compete for the same frequency band, traditionally requiring a tedious trial-and-error process of equalization to achieve clarity. This paper proposes a systematic, mathematical framework to separate frequency-clashing elements by modeling instrument separation as a graph coloring problem. By representing individual audio tracks as vertices and their overlapping characteristic frequency peaks (within a critical band of one-third of an octave) as edges, the Welsh-Powell algorithm is applied to organize the tracks. Tested on a 14-piece musical ensemble, the algorithm successfully partitioned the instruments into seven frequency-compatible groups. This deterministic approach establishes an upper bound on the chromatic number of the arrangement, allowing mixing engineers to group non-conflicting instruments together before the detailed balancing process begins, thereby reducing reactive mixing conflicts and optimizing overall clarity.**

***Keywords—Audio Mixing; Auditory Masking; Graph Theory; Vertex Coloring; Welsh-Powell Algorithm; Frequency Separation; Equalization.***

## I. INTRODUCTION

In modern audio production, mixing is the crucial step where individual audio tracks—ranging from raw instrument takes to synthesized loops—are processed and blended together to create a final, unified stereo file [1]. The aim of mixing is to produce a sonically pleasing blend of the musical performance, emotions, and ideas of the track [2].

There are many different styles and approaches to mixing, which leads many authors to refer to audio mixing as an artform. However, there are generally agreed-upon qualities of what constitutes a good mix. Owsinski [3] defines six elements of a mix:

1. Balance: Establishing the appropriate relative loudness for each instrument.
2. Frequency Range: Ensuring the entire audio spectrum is adequately utilized.
3. Panorama: Positioning sounds across the left-to-right stereo image to give each element its own location.
4. Dimension: Creating a sense of acoustic space and three-dimensional depth by having ambiance.
5. Dynamics: Managing the overall volume envelope of individual tracks over time to maintain energy.
6. Interest: Introducing creative variations to capture and hold the listener's attention throughout the song.

A great mix must start with the first element, balance [3]. Balance is greatly affected by the arrangement of the instruments, therefore each instrument part must fit well with one another. When two instruments that have the same frequency band play at the same volume at the same time, the result is a muddy clash, known as auditory masking.

The mixing engineer has a few ways to resolve this clash, which are (1) re-recording the track with a different arrangement, (2) muting the offending instruments so that they never play simultaneously,

(3) lowering the volume of the offending instruments, (4) tailoring the EQ so that the offending instrument takes up a different frequency space, and (5) panning the offending instrument to a different location.

While these solutions are effective, implementing them typically relies on an iterative process of trial-and-error. Adjusting the equalizer or panning position of one instrument often creates a new conflict with another, leading to a tedious pattern of reactive mixing choices.

To eliminate this guesswork, this paper proposes a systematic, mathematical framework for instrument separation using graph theory. By modelling individual audio tracks as vertices and their overlapping frequency profiles as conflicting edges, mixing becomes a graph coloring problem. Utilizing the Welsh-Powell algorithm, instruments are grouped so that equalizing one does not disturb another within its group. Ultimately, the number of treatment groups approximates the minimum number of treatment groups required to separate clashing instruments, improving balance in the final mix.

## II. THEORY BASIS

### *A. Graph*

A graph is defined as a discrete structure consisting of vertices and edges that connect a pair of vertices. Formally, a graph $G = (V, E)$ has a non-empty set of vertices $V$ and a set of edges $E$. Based on the existence of loops and parallel edges, there are two types of graphs, simple and unsimple graphs.

Simple graphs contains neither loops nor parallel edges. In other words, connections in a simple graph only happen directly between two different vertices, with no duplicate edges.

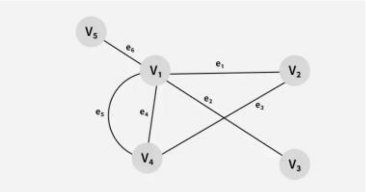


Fig. 1. A simple graph [4].

Unsimple graphs are graphs that contain loops or parallel edges. Graphs that contain loops are referred to as pseudographs, and graphs that do not contain loops but contain parallel edges are referred to as multigraphs.

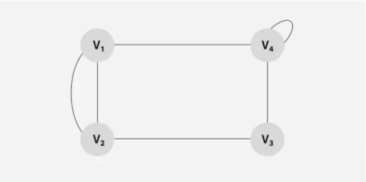


Fig. 2 A pseudograph [4].

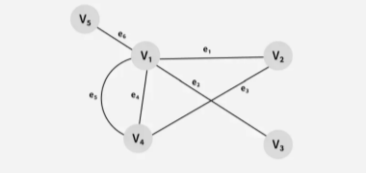


Fig. 3 A multigraph [4].

Based on the orientation of the edges, graphs are classified into two categories, directed graphs and undirected graphs.

Undirected graphs are graphs whose edges have no direction. In an undirected graph, the edge connecting two vertices *u* and *v* may be written as either *e = (u, v)* or *e = (v, u).*

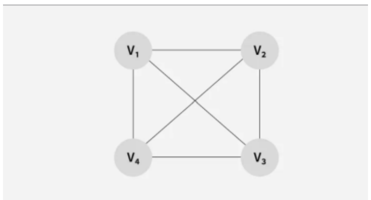


Fig. 4. An undirected graph [4].

Directed graphs are graphs where each edge has a direction. In a directed graph, an edge *e = (u, v)* is an edge that starts at vertex *u* and end at vertex *v*. Therefore, *e = (u, v)* and *e = (v, u)* are considered distinct. Directed graphs that contain parallel edges are referred to as directed multigraphs.

### *B. Graph Terminology*

The following terms are often used in the discussion of graphs:

1. Adjacency

   Two vertices *u* and *v* are adjacent if and only if *u* and *v* are endpoints of an edge *e*.

2. Incidency

   An edge *e* is incident to the vertices *u* and *v* if and only if *u* and *v* are adjacent.

3. Degree

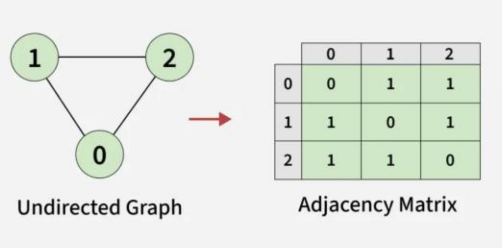
   The degree of a vertex *v* (denoted as *deg(v)*) represents the number of edges incident with *v*.

### *C. Graph Representation*

There are various ways to present graphs, notably:

1. Adjacency Matrix

   An adjacency matrix is a 2D square matrix of size *N* x *N*, where *N* is the number of vertices. The rows and columns are labeled by the graph vertices. If there is an edge between vertex *i* and vertex *j*, the matrix element $M_{ij}$ is set to 1 (or the number of parallel edges for a multigraph). If there is no edge between *i* and *j*, $M_{ij}$ is set to 0.

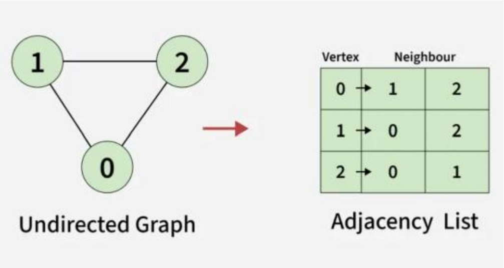


|   | 0 | 1 | 2 |
|---|---|---|---|
| 0 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 2 | 1 | 1 | 0 |

Fig. 5 Adjacency matrix of an undirected graph [5].

2. Adjacency List

   Unlike a matrix, an adjacency list maps each vertex to a corresponding array. This array dynamically stores the specific endpoints of every edge incident to that vertex, making it a highly memory-efficient representation for sparse graphs [5].



| Vertex | Neighbour |   |
|---|---|---|
| 0 → | 1 | 2 |
| 1 → | 0 | 2 |
| 2 → | 0 | 1 |

Fig. 6 Adjacency list of an undirected graph [5]

### *D. Graph Coloring*

Graph coloring involves systematically assigning distinct labels (or "colors") to the structural components of a graph based on specific constraints. Vertex coloring assigns a color to each vertex such that no two adjacent vertices share the same color. Conversely, edge coloring assigns colors to edges so that no two incident edges share a color. In this paper, the focus is restricted exclusively to vertex coloring. The minimum number of colors needed to validly color a graph $G$ is called its chromatic number, denoted by $\chi(G)$.

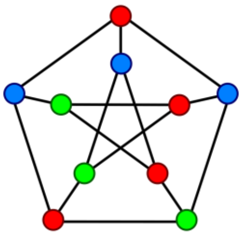


Fig. 7 Vertex coloring for a graph [6].

There are many algorithms to find a valid vertex coloring for an arbitrary graph. However, finding its chromatic number cannot be done in polynomial time.

Some of the algorithms for vertex coloring include:

1. Brute Force / Backtracking

   Systematically checking every possible combination of colors to guarantee finding the chromatic number $\chi(G)$. Because vertex coloring is NP-hard, a brute-force approach slows down exponentially as the graphs grow, rendering this algorithm impractical.

2. Naive / Greedy Algorithm

   Assigning the first available color from a list to each vertex as they appear in the dataset. Because of the arbitrary order, the algorithm might assign the first available colors to vertices with very low degrees simply because they happen to appear early in the dataset. This will cause later high degree vertices which have a massive number of conflicting edges to be processed late in the sequence. Consequently, the algorithm is often driven into using a much larger number of colors than necessary.

3. Welsh-Powell Algorithm

   The Welsh-Powell algorithm is a heuristic based on the greedy algorithm designed to find a tighter upper bound for the chromatic number in polynomial time. Unlike the naive sequential approach, Welsh-Powell eliminates arbitrary ordering by establishing a priority based on vertex degrees.

   The formal procedural steps of the Welsh-Powell algorithm are defined as follows:

   1. Compute the degree of each vertex in the given graph.
   2. Sort all vertices into a sorted list $L$ in descending order according to their degrees. If multiple vertices have identical degrees, their relative order is determined by their original indexing in the dataset.
   3. Select the first color $C_1$ and assign it to the first vertex in the sorted list (which is a vertex with the current maximum degree).
   4. Traverse down the sorted list $L$ sequentially. For each vertex encountered, assign $C_1$ if and only if the vertex is not adjacent to any other vertex that has already been assigned $C_1$.
   5. Remove all colored vertices from the list L. Increment the color identifier to the next available color $C_{k+1}$, return to the top of the remaining uncolored list, and repeat steps 4 and 5. This loop terminates when all vertices in the graph have been assigned a color.

   Using this algorithm, the number of colors used ($p$) will be:

   $$p \leq \Delta(G) + 1,$$

   where $\Delta(G)$ is the maximum degree in the given graph.

### E. Auditory Masking

Auditory masking is a phenomenon in the human hearing system that occurs when a strong audio signal makes weaker audio signals less perceptible. When the presence of one sound source reduces the clarity and audibility of another sound source, the phenomenon is referred to as partial masking. In audio mixing, multiple sound sources often partially mask one another, reducing the clarity and distinguishability of individual instruments [7].

The masking effect depends on the characteristics of the sound. A strong signal within the same frequency band (usually thought of a third of an octave) can mask weaker signals occupying the same frequency interval [8]. Consequently, instruments whose dominant frequency peaks overlap are more likely to interfere with one another and become difficult to distinguish in a mix.

Equalization is commonly used to reduce auditory masking by altering the frequency characteristics of instruments, thereby reducing frequency masking between competing sound sources [7]. Therefore, identifying overlapping frequency regions is an important step in achieving effective instrument separation during the mixing process.

## III. METHODOLOGY

### A. Problem Scenario

Suppose we were to mix a recording of the award-winning group *Earth, Wind, and Fire*. The group's ensemble consists of 14 instruments, with their frequency peaks, drawn from [3], listed in the following table.

TABLE I. INSTRUMENT FREQUENCY DATA

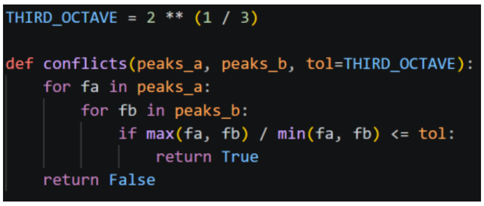
| *No.* | *Instrument* | *Characteristic Peaks (Hz)* |
|---|---|---|
| 1. | Bass Guitar | 65, 700, 2500 |
| 2. | Kick drum | 90, 400, 4000 |
| 3. | Snare | 180, 900, 5000, 10000 |
| 4. | Toms | 370, 6000 |
| 5. | Floor Tom | 80, 5000 |

| *No.* | *Instrument* | *Characteristic Peaks (Hz)* |
|---|---|---|
| 6. | Hi-hat and cymbals | 200, 9000 |
| 7. | Electric Guitar | 370, 2000 |
| 8. | Acoustic Guitar | 80, 240, 3500 |
| 9. | Organ | 80, 240, 3500 |
| 10. | Piano | 80, 2500, 4000 |
| 11. | Horns | 120, 5000 |
| 12. | Voice | 120, 240, 5000, 5500, 12500 |
| 13. | Strings | 240, 8500 |
| 14. | Conga | 200, 5000 |

An instrument is defined as clashing with another instrument if any of their characteristic peaks fall within one-third of an octave with each other (sharing a critical band). Mathematically instruments I and j are considered clashing if and only if:

$$\exists\ f_1 \in F_i,\ f_2 \in F_j \text{ such that}$$

$$\max(f_1, f_2) / \min(f_1, f_2) \leq 2\text{\^{}}(1/3) \approx 1.26$$



```
THIRD_OCTAVE = 2 ** (1 / 3)

def conflicts(peaks_a, peaks_b, tol=THIRD_OCTAVE):
    for fa in peaks_a:
        for fb in peaks_b:
            if max(fa, fb) / min(fa, fb) <= tol:
                return True
    return False
```

Fig. 8 Python Algorithm to check instrument conflicts

If two instruments peak within the same critical band, adjusting EQ of an instrument around that peak unavoidably affects the balance with the other. Therefore, these instruments cannot belong to the same group. Conversely, instruments whose peaks never collide may share a group, since adjusting the EQ of the group at any peak only affects one instrument. The aim of the experiment is to group these instruments such that they never clash with other instruments in the same group.

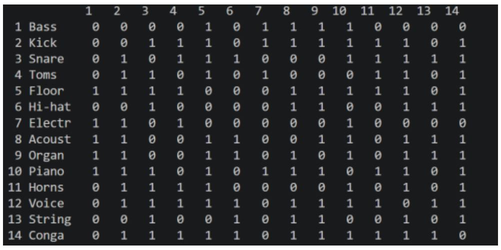
### B. *Graph Theory Model*

The band setup is translated into a graph where each instrument corresponds to a vertex, and a frequency clash between two instruments corresponds to an edge that is incident to both instrument's vertices. Assigning groupings to the band is now a vertex coloring problem, where a valid coloring means no instruments within a group have clashing frequency peaks.

```
def build_graph(instruments):
    graph = defaultdict(set)
    names = list(instruments)
    for i, a in enumerate(names):
        for b in names[i + 1:]:
            if conflicts(instruments[a], instruments[b]):
                graph[a].add(b)
                graph[b].add(a)

    for name in names:
        _ = graph[name]
    return graph
```

Fig. 9 Graph building algorithm in Python.

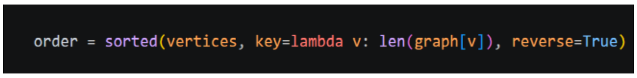


```
           1   2   3   4   5   6   7   8   9  10  11  12  13  14
 1 Bass    0   0   0   0   1   0   1   1   1   1   0   0   0   0
 2 Kick    0   0   1   1   1   0   1   1   1   1   1   1   0   1
 3 Snare   0   1   0   1   1   1   0   0   0   1   1   1   1   1
 4 Toms    0   1   1   0   1   0   1   0   0   0   1   1   0   1
 5 Floor   1   1   1   1   0   0   0   1   1   1   1   1   0   1
 6 Hi-hat  0   0   1   0   0   0   0   1   1   0   0   1   1   1
 7 Electr  1   1   0   1   0   0   0   0   0   1   0   0   0   0
 8 Acoust  1   1   0   0   1   1   0   0   1   1   0   1   1   1
 9 Organ   1   1   0   0   1   1   0   1   0   1   0   1   1   1
10 Piano   1   1   1   0   1   0   1   1   1   0   1   1   0   1
11 Horns   0   1   1   1   1   0   0   0   0   1   0   1   0   1
12 Voice   0   1   1   1   1   1   0   1   1   1   1   0   1   1
13 String  0   0   1   0   0   1   0   1   1   0   0   1   0   1
14 Conga   0   1   1   1   1   1   0   1   1   1   1   1   1   0
```

Fig. 10 Adjacency matrix of frequency clash.

### C. *Application of Welsh-Powell Algorithm*

The Welsh-Powell algorithm can be utilized to find a valid coloring (grouping). The algorithm starts by finding the degree of each vertex and sorting them in descending order.



```
order = sorted(vertices, key=lambda v: len(graph[v]), reverse=True)
```

Fig 11. Python program to sort degrees in descending order.

TABLE II. VERTICES SORTED IN DESCENDING ORDER

| *No.* | *Vertex* | *Degree* |
|---|---|---|
| 1. | Voice | 11 |
| 2. | Conga | 11 |
| 3. | Kick drum | 10 |
| 4. | Floor tom | 10 |
| 5. | Piano | 10 |
| 6. | Snare | 9 |
| 7. | Acoustic Guitar | 9 |
| 8. | Organ | 9 |
| 9. | Toms | 7 |
| 10. | Horns | 7 |
| 11. | Hi-hat and cymbals | 6 |
| 12. | Strings | 6 |
| 13. | Bass guitar | 5 |
| 14. | Electric Guitar | 4 |

After a sorted list is made, the algorithm begins its first iteration, assigning the first vertex in the list with the first color. Then, it moves to the next vertex in the list and checks the colors of its neighbors.

If any of its neighbors are assigned the first color, the algorithm checks the next color until it finds a valid color to assign the vertex. If no such color is found, the vertex is assigned a new color, and that color is appended to the color list. The process is done for every vertex in the graph.

```
color = {}
for step, v in enumerate(order, 1):
    colored_neighbors = {u: color[u] for u in graph[v] if u in color}
    used = set(colored_neighbors.values())

    c = 1
    while c in used:
        c += 1
    color[v] = c
```

Fig. 12 Welsh-Powell iteration algorithm in Python.

Table 3 shows the assignment process of each vertex graph.

TABLE III. VERTEX COLORING STEPS

| ***No.*** | ***Vertex*** | ***Degree*** | ***Blocked Colors*** | ***Assigned Color*** |
|---|---|---|---|---|
| 1. | Voice | 11 | - | 1 |
| 2. | Conga | 11 | [1] | 2 |
| 3. | Kick drum | 10 | [1, 2] | 3 |
| 4. | Floor tom | 10 | [1, 2, 3] | 4 |
| 5. | Piano | 10 | [1, 2, 3, 4] | 5 |
| 6. | Snare | 9 | [1, 2, 3, 4, 5] | 6 |
| 7. | Acoustic Guitar | 9 | [1, 2, 3, 4, 5] | 6 |
| 8. | Organ | 9 | [1, 2, 3, 4, 5, 6] | 7 |
| 9. | Toms | 7 | [1, 2, 3, 4, 6] | 5 |
| 10. | Horns | 7 | [1, 2, 3, 4, 5, 6] | 7 |
| 11. | Hi-hat and cymbals | 6 | [1, 2, 6, 7] | 3 |
| 12. | Strings | 6 | [1, 2, 3, 6, 7] | 4 |
| 13. | Bass guitar | 5 | [4, 5, 6, 7] | 1 |
| 14. | Electric Guitar | 4 | [1, 3, 5] | 2 |

## IV. RESULTS

The 14-instrument band configuration can be represented with a graph with 14 vertices and 57 edges, or a density of 62.6%. This suggests that the arrangement is heavily crowded, and most instruments are clashing with each other. Voice and conga conflict with the most instruments (degree of 11), meanwhile electric guitar conflict with the least instruments (degree of 4). This finding is unsurprising since Voice has frequency peaks that covers most of the audio spectrum.

The algorithm assigned colors 1-5 to vertices 1-5 in the sorted list, since they are tightly interconnected to each other. The first color being reused happened on the 7th vertex, Acoustic Guitar, with it reusing color 6. The final result yields 7 different colors (groupings), shown in the following table.

TABLE IV. FINAL INSTRUMENT GROUPINGS

| ***Group*** | ***Instruments*** |
|---|---|
| 1 | Voice, Bass guitar |
| 2 | Conga, Electric guitar |
| 3 | Kick drum, Hi-hat and cymbals |
| 4 | Floor tom, Strings |
| 5 | Piano, Toms |
| 6 | Snare, Acoustic guitar |
| 7 | Organ, Horns |

The Welsh-Powell Algorithm has shown that 7 groups are enough to separate all 14 instruments so that they won't fight for the same critical band. For example, the closest frequency peaks between the bass guitar (2500 Hz) and the vocals (5000 Hz) are a full octave apart — far outside the one-third-octave critical band — so adjusting either instrument's EQ never disturbs the other within their shared group. However, it is important to note that 7 may not be the minimum amount needed (the chromatic number $\chi(G)$).

## V. CONCLUSION

This paper set out to address a persistent difficulty in audio mixing, where many instruments compete for the same regions of the frequency spectrum, and balancing them becomes an iterative cycle in which correcting one instrument's tone often disturbs another. To approach this systematically rather than by trial and error, the problem of organizing instruments by their spectral content was reformulated as a graph-coloring problem.

Each instrument in a fourteen-piece ensemble (modelled after *Earth, Wind and Fire*) was abstracted as a vertex, a conflict between two instruments whose characteristic frequency peaks fall within one-third of an octave was abstracted as an edge, and a frequency-compatible group of instruments was abstracted as a color.

Applying Welsh-Powell partitioned the ensemble into seven frequency-compatible groups, a 50% reduction compared to treating every instrument independently. The degree analysis further identified where contention concentrates. Voice and conga, each conflicting with eleven of the other thirteen instruments, emerged as the arrangement's bottlenecks, while the electric guitar, with only four graph neighbors, proved the most flexible. In this way the chromatic number functions as a property of the arrangement itself: it measures how many distinct frequency zones the music genuinely requires, independent of the engineer's preferences.

The practical value of this result is modest but real. While the engineer must apply an adjustment to separate the instruments, whether by tuning levels, panning, or dynamics, all of which remain matters of engineering judgment, this paper provides a systematic process that replaces one portion of the trial-and-error cycle with an algorithm that can be computed before detailed balancing begins.

There are several limitations in these findings that should be kept in mind. The model represents each instrument only by its characteristic frequency peaks rather than its full spectrum, and it treats all peaks as equal regardless of amplitude, even though a louder instrument masks more than a quieter one at the same frequency. The one-third-octave conflict criterion, while grounded in the standard notion of a critical band, is a fixed approximation of a quantity that in reality varies across the frequency range. A valid coloring, moreover, represents one compatible grouping rather than a uniquely correct one.

Finally, because Welsh-Powell is a greedy algorithm, the seven groups it produced constitute an upper bound on the true chromatic number. While seven groups are proved to be sufficient, the algorithm does not prove that fewer are impossible.

These limitations point naturally toward future work. The simple conflict graph could be extended to a weighted graph in which edges carry a weight derived from amplitude or from the degree of frequency overlap, allowing the model to prioritize a severe collision from a minor one. The fixed one-third-octave threshold could be replaced with a frequency-dependent critical bandwidth to be more accurate with the nonlinear way humans perceive audio. The Welsh-Powell result could also be compared against other coloring strategies, such as DSATUR or an exact solver, to determine how close the heuristic comes to the true minimum.

In summary, modeling frequency separation as a graph-coloring problem and solving it with the Welsh-Powell algorithm offers a systematic, deterministic way to quantify the frequency clashing within a musical arrangement and to identify where that clashing is most severe. While the judgment of the mixing engineer remains essential, demonstrating that the tools of discrete mathematics can formalize and quantify a tedious process that is otherwise guided largely by intuition.

## VI. APPENDIX

The implementation of Welsh-Powell algorithm can be accessed through the following GitHub link:

https://github.com/ryualdebaran/welsh-powell-frequency-separation

## REFERENCES

[1] R. Izhaki, *Mixing Audio: Concepts, Practices, and Tools*, 4th ed. New York, NY, USA: Focal Press, 2023.

[2] S. Stickland and R. Athauda, "A new audio mixing paradigm: evaluation from professional practitioners' perspectives," *Journal of Music, Technology & Education*, vol. 14, no. 2-3, pp. 163–183, Dec. 2022.

[3] B. Owsinski, *The Mixing Engineer's Handbook*, 2nd ed. Boston, MA, USA: Thomson Course Technology, 2006.

[4] "Types of Graphs with Examples," GeeksforGeeks. [Online]. Available: https://www.geeksforgeeks.org/types-of-graphs-with-examples/ [Accessed: June 18, 2026].

[5] "Representation of Graph," GeeksforGeeks. [Online]. Available: https://www.geeksforgeeks.org/graph-and-its-representations/ [Accessed: June 18, 2026].

[6] Wikimedia Commons, "File:Petersen graph 3-coloring.svg." [Online]. Available: https://commons.wikimedia.org/wiki/File:Petersen_graph_3-coloring.svg [Accessed: June 18, 2026].

[7] D. Ronan, Z. Ma, P. McNamara, H. Gunes, and J. D. Reiss, "Automatic Minimisation of Masking in Multitrack Audio using Subgroups," *arXiv preprint*, arXiv:1803.09960, pp. 1–11, 2018.

[8] S. Mallat, *A Wavelet Tour of Signal Processing: The Sparse Way*, 3rd ed. Burlington, MA, USA: Academic Press, 2008.

## PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Jakarta, 19 Juni 2026

Ryuza Nadif Aldebaran
13525026