The Journal of Open Source Software

The Keçeci Layout: A Deterministic, Order-Preserving Visualization Algorithm for Structured Systems

Mehmet Keçeci [1]

1 Independent Researcher, Türkiye

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @openjournals

Submitted: 01 January 1970
Published: unpublished

Summary

Graph visualization is a cornerstone of network analysis, yet traditional algorithms often prioritize topological representation over the preservation of inherent node order. This can obscure sequential or procedural information critical in many scientific and structural analyses. This paper introduces the *Keçeci Layout*, a deterministic, order-preserving graph layout algorithm designed to arrange nodes in a structured zigzag pattern. This method provides a clear, predictable, and structurally informative visualization for systems where the sequence of nodes is meaningful. The layout is implemented in the open-source `kececilayout` Python package, which offers seamless interoperability with major graph analysis libraries, including NetworkX, igraph, rustworkx, Networkit, and Graphillion. `kececilayout` is open source, licensed under the MIT license, and the source code is available on GitHub at https://github.com/WhiteSymmetry/kececilayout. The version of the software described in this paper is archived on Zenodo (Keçeci, 2025g). We detail the algorithm's methodology, showcase its implementation, and discuss its applications as a cross-disciplinary framework for structural analysis. The deterministic nature of the layout ensures that any given graph will always be rendered identically, facilitating reproducible research and comparative analysis.

Statement of Need

The visualization of complex networks is fundamental to understanding their structure, function, and underlying patterns. Algorithms such as force-directed layouts (Fruchterman & Reingold, 1991) are highly effective at revealing topological features like clusters and central nodes. However, they achieve this by optimizing node positions to minimize edge crossings and regularize edge lengths, a process that inherently disregards any pre-existing order among the nodes.

To address this gap, the Keçeci Layout was developed as a structural approach for interdisciplinary scientific analysis (Keçeci, 2025m, 2025n, 2025o). It is a deterministic algorithm that explicitly preserves the order of nodes, arranging them in a predictable zigzag pattern (Keçeci, 2025d, 2025f). This approach moves beyond purely topological representations to provide a "structural thinking" framework, enabling clearer insight into ordered systems (Keçeci, 2025c). As described by Keçeci (2025l), this paper describes the core principles of the Keçeci Layout, details its implementation, and highlights its utility in cross-disciplinary contexts.

The Keçeci Layout Algorithm

The Keçeci Layout is fundamentally a sequential algorithm that places nodes one by one according to their given order. Its defining characteristic is the combination of linear progression along a primary axis and an alternating, expanding offset along a secondary axis. This generates the distinctive zigzag shape (Keçeci, 2025d). The algorithm is deterministic: for a given list of nodes and a fixed set of parameters, the resulting layout is always identical (Keçeci, 2025b).

Algorithmic Principles

The position of each node is determined by its index in the sorted node list. Let $N = (n_0, n_1, \dots, n_{k-1})$ be the ordered sequence of k nodes. For each node n_i at index i, its coordinates (x_i, y_i) are calculated based on four key parameters:

- **`primary_direction`**: Defines the main axis of progression. It can be vertical ('top-down', 'bottom-up') or horizontal ('left-to-right', 'right-to-left').
- **`primary_spacing`**: The constant distance separating consecutive nodes along the primary axis.
- **`secondary_spacing`**: The base unit of distance for the offset along the secondary axis.
- **`secondary_start`**: Defines the direction of the first offset on the secondary axis (e.g., 'right' or 'left' for a vertical primary axis).

The core logic for a 'top-down' primary direction is as follows:

1. **Primary Coordinate Calculation:** The primary coordinate (in this case, y) is determined by the node's index i.
$$y_i = -i \times \text{primary\_spacing}$$
2. **Secondary Coordinate Calculation:** The secondary coordinate (x) is calculated based on an alternating and growing offset. The magnitude of the offset for node n_i is proportional to $\lceil i/2 \rceil$, and its direction depends on whether i is odd or even.
$$\text{side} = \begin{cases} 1 & \text{if } i \text{ is odd} \\ -1 & \text{if } i \text{ is even} \end{cases}$$
$$x_i = \text{start\_direction} \times \lceil i/2 \rceil \times \text{side} \times \text{secondary\_spacing}$$
The node at index $i = 0$ is placed at the origin of the secondary axis ($x_0 = 0$).

This deterministic procedure ensures that nodes are arranged sequentially, making it easy to trace paths and understand flow while effectively utilizing two-dimensional space to avoid overlap. The graph-theoretic underpinnings of this structured approach facilitate cross-disciplinary inquiry by providing a common visual language (Keçeci, 2025a).

Implementation: The `kececilayout` Package

The Keçeci Layout algorithm is implemented and distributed as an open-source Python package named `kececilayout`. The package is designed for ease of use and seamless integration with the scientific Python ecosystem.

Availability and Installation

The package is available on both the Python Package Index (PyPI) and Anaconda, and its source code is hosted on GitHub. It can be installed using standard package managers:

Using pip:

```
pip install kececilayout
```

Using conda (from the 'bilgi' channel):

```
conda install -c bilgi kececilayout
```

Relevant resources, including the source code and data sets, are publicly available (Keçeci, 2025j, 2025g, 2025h, 2025i, 2025k).

Interoperability and Usage

A key design goal of the `kececilayout` package is to provide a unified interface for various graph libraries. The main function, `kececilayout.kececi_layout()`, automatically detects the input graph type and returns a position dictionary in the format expected by that library. This promotes a cross-disciplinary graphical framework by allowing researchers to use the same visualization logic regardless of their preferred analysis tool (Keçeci, 2025l).

Supported libraries include:

- **NetworkX**: The most popular graph analysis library in the Python data science community.
- **igraph**: A high-performance library widely used in academic research.
- **rustworkx**: A fast, thread-safe graph library written in Rust, often used in performance-critical applications.
- **Networkit**: A library focused on high-performance analysis of large-scale networks.
- **Graphillion**: A specialized library for very large sets of graphs.

The following example demonstrates how to apply the Keçeci Layout to a simple NetworkX path graph and visualize it with Matplotlib.

```python
import networkx as nx
import kececilayout as kl
import matplotlib.pyplot as plt

# 1. Generate a graph (e.g., a path graph with 25 nodes)
G = nx.path_graph(25)

# 2. Compute the node positions using Kececi Layout
# The node order is preserved (0, 1, 2, ...)
pos = kl.kececi_layout(G, primary_spacing=1.5, secondary_spacing=0.8)

# 3. Visualize the graph
plt.figure(figsize=(8, 10))
nx.draw(
    G,
    pos=pos,
    with_labels=True,
    node_color='skyblue',
    node_size=500,
    font_size=8,
    edge_color='gray'
)
plt.title("Kececi Layout Applied to a Path Graph (n=25)")
plt.axis('equal') # Ensure aspect ratio is not distorted
plt.show()
```

The Journal of Open Source Software

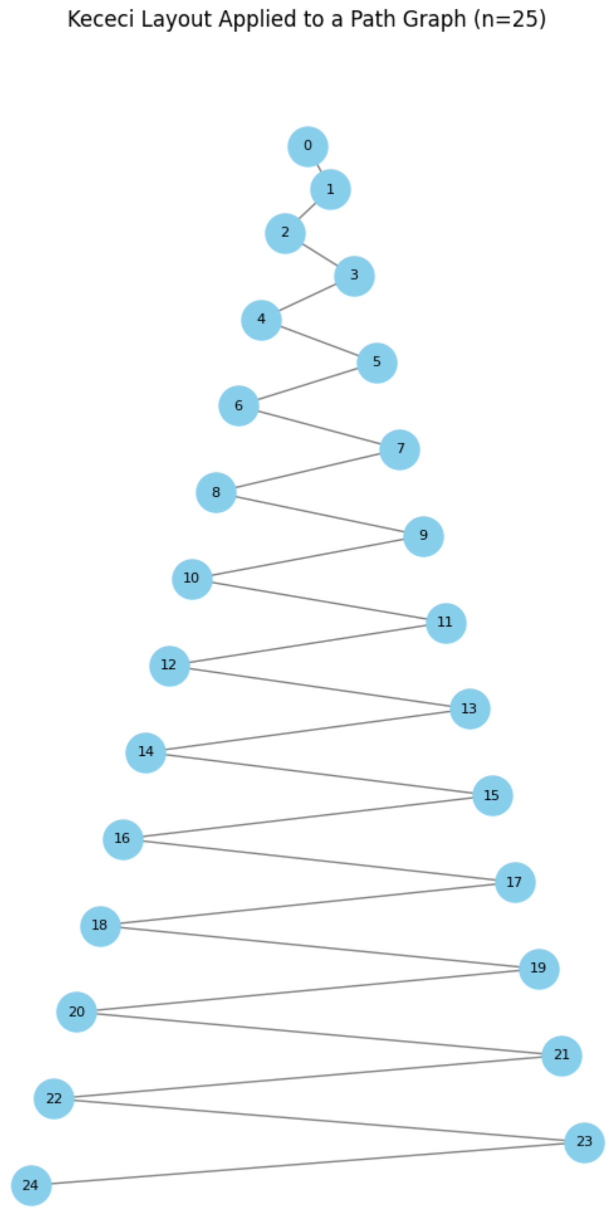

Figure 1: An example visualization of a path graph using the Keçeci Layout. The nodes are ordered sequentially from top to bottom, with their positions determined by the deterministic zigzag algorithm. This preserves the inherent one-dimensional structure of the path.

Applications and Use Cases

The primary strength of the Keçeci Layout is its ability to visualize systems "when nodes have an order" (Keçeci, 2025p). Its application is particularly relevant in fields where sequential data is modeled as a graph.

- **Workflow and Process Visualization:** Representing business processes, experimental workflows (Keçeci, 2025d, 2025b), or CI/CD pipelines where the sequence of steps is paramount. The layout clearly shows the progression from start to finish.

- **Narrative and Structural Analysis:** Analyzing the structure of stories, legal arguments, or scientific papers where nodes represent events, sections, or concepts in a specific order.
- **Time-Series and Event Logs:** Visualizing sequences of events from logs or time-series data, where the temporal order must be maintained.
- **Comparative Structural Analysis:** As a standardized graphical framework, it allows for the visual comparison of different ordered systems, fostering interdisciplinary insights (Keçeci, 2025m, 2025n).

By providing a stable and structured visual representation, the layout encourages a "structural thinking" approach, where the focus shifts from complex topological entanglements to the clear, sequential architecture of the system being studied (Keçeci, 2025c, 2025e).

Conclusion

The Keçeci Layout provides a much-needed alternative to traditional graph drawing algorithms for the visualization of ordered systems. Its deterministic, zigzag-based approach ensures that the inherent sequence of nodes is not only preserved but becomes the primary organizing principle of the visualization. This results in clear, predictable, and structurally informative diagrams that are easy to interpret.

The implementation of this algorithm in the user-friendly and interoperable `kececilayout` Python package makes it an accessible tool for a wide range of researchers and practitioners. By offering seamless support for dominant graph libraries, it serves as a robust, cross-disciplinary framework for structural analysis. Ultimately, the Keçeci Layout champions the idea that for many systems, visualization should go beyond topology to faithfully represent structure and order (Keçeci, 2025b).

Future Work

Future work will focus on extending the layout's principles to three dimensions and developing adaptive spacing strategies for graphs with highly variable node density or connectivity.

Fruchterman, T. M., & Reingold, E. M. (1991). Graph drawing by force-directed placement. *Software: Practice and Experience*, *21*(11), 1129–1164. https://doi.org/10.1002/spe.4380211102

Keçeci, M. (2025a). *A Graph-Theoretic Perspective on the Keçeci Layout: Structuring Cross-Disciplinary Inquiry*. Preprints.org. https://doi.org/10.20944/preprints202507.0589.v1

Keçeci, M. (2025b). *Beyond Topology: Deterministic and Order-Preserving Graph Visualization with the Keçeci Layout*. WorkflowHub. https://doi.org/10.48546/workflowhub.document.34.4

Keçeci, M. (2025c). *Beyond Traditional Diagrams: The Keçeci Layout for Structural Thinking*. Knowledge Commons. https://doi.org/10.17613/v4w94-ak572

Keçeci, M. (2025d). *Keçeci Deterministic Zigzag Layout*. WorkflowHub. https://doi.org/10.48546/workflowhub.document.31.1

Keçeci, M. (2025e). *Keçeci Layout*. Zenodo. https://doi.org/10.5281/zenodo.15314328

Keçeci, M. (2025f). *Keçeci Zigzag Layout Algorithm*. Authorea. https://doi.org/10.22541/au.175087581.16524538/v1

Keçeci, M. (2025g). *Kececilayout*. Zenodo. https://doi.org/10.5281/zenodo.15313946

Keçeci, M. (2025h). *kececilayout*. Python Package Index (PyPI). https://pypi.org/project/kececilayout/

Keçeci, M. (2025i). *kececilayout*. Anaconda Cloud. https://anaconda.org/bilgi/kececilayout

Keçeci, M. (2025j). *kececilayout [Data set]*. WorkflowHub. https://doi.org/10.48546/workflowhub.datafile.17.2

Keçeci, M. (2025k). *kececilayout: A deterministic, order-preserving, zigzag-shaped graph layout algorithm*. GitHub. https://github.com/WhiteSymmetry/kececilayout

Keçeci, M. (2025l). *The Keçeci Layout: A Cross-Disciplinary Graphical Framework for Structural Analysis of Ordered Systems*. Authorea. https://doi.org/10.22541/au.175156702.26421899/v1

Keçeci, M. (2025m). The Keçeci Layout: A Structural Approach for Interdisciplinary Scientific Analysis. *Zenodo*. https://doi.org/10.5281/zenodo.15792684

Keçeci, M. (2025n). The Keçeci Layout: A Structural Approach for Interdisciplinary Scientific Analysis. *Figshare*. https://doi.org/10.6084/m9.figshare.29468135

Keçeci, M. (2025o). *The Keçeci Layout: A Structural Approach for Interdisciplinary Scientific Analysis*. OSF. https://doi.org/10.17605/OSF.IO/9HTG3

Keçeci, M. (2025p). *When Nodes Have an Order: The Keçeci Layout for Structured System Visualization*. HAL open science. https://doi.org/10.13140/RG.2.2.19098.76484