2-uniform words: cycle graphs, and an algorithm to verify specific word-representations of graphs

Ameya Daigavane, Mrityunjay Singh, Benny K. George

Indian Institute of Technology, Guwahati

February 20, 2018

Introduction

What are word-representations of graphs?

Introduction

What are word-representations of graphs?

- A way to represent certain graphs $G = (V, E)$ by a word on the alphabet V.

Introduction

What are word-representations of graphs?

- A way to represent certain graphs $G = (V, E)$ by a word on the alphabet V.
- An alternation between the letters i and j in the word corresponds to an edge (i, j) in the graph.

Introduction

What are word-representations of graphs?

- A way to represent certain graphs $G = (V, E)$ by a word on the alphabet V.
- An alternation between the letters i and j in the word corresponds to an edge (i, j) in the graph.
- $G(w)$ is the graph generated by considering all possible alternations in w.

Introduction

What are word-representations of graphs?

- A way to represent certain graphs $G = (V, E)$ by a word on the alphabet V.
- An alternation between the letters i and j in the word corresponds to an edge (i, j) in the graph.
- $G(w)$ is the graph generated by considering all possible alternations in w.
- If $G(w) = G$ for some word w, then w is said to be a word-representation or word-representant of the graph G.

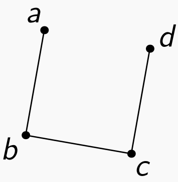


Figure: $G(w)$ for $w = bcabadc$.

Word-Representations

Does every graph G have a word-representation?

Word-Representations

Does every graph G have a word-representation? Unfortunately, no.

Word-Representations

Does every graph G have a word-representation? Unfortunately, no.

Theorem (Halldorsson, Kitaev, Pyatkin)

A graph is word-representable iff it admits a semi-transitive orientation.

- But, some very important and well-known classes of graphs - the circle graphs, transitively orientable graphs, and graphs of vertex degree at most 3 - are all word-representable.

Word-Representations

Does every graph G have a word-representation? Unfortunately, no.

Theorem (Halldorsson, Kitaev, Pyatkin)

A graph is word-representable iff it admits a semi-transitive orientation.

- But, some very important and well-known classes of graphs - the circle graphs, transitively orientable graphs, and graphs of vertex degree at most 3 - are all word-representable.

Word-Representations

Does every graph G have a word-representation? Unfortunately, no.

Theorem (Halldorsson, Kitaev, Pyatkin)

A graph is word-representable iff it admits a semi-transitive orientation.

- But, some very important and well-known classes of graphs - the circle graphs, transitively orientable graphs, and graphs of vertex degree at most 3 - are all word-representable.
- Also, asymptotically, a large number of graphs are word-representable!

Word-Representations

Does every graph G have a word-representation? Unfortunately, no.

Theorem (Halldorsson, Kitaev, Pyatkin)

A graph is word-representable iff it admits a semi-transitive orientation.

- But, some very important and well-known classes of graphs - the circle graphs, transitively orientable graphs, and graphs of vertex degree at most 3 - are all word-representable.
- Also, asymptotically, a large number of graphs are word-representable!

Theorem (Collins, Kitaev)

The number of n-vertex word-representable graphs is $2^{\frac{n^2}{3}+o(n^2)}$.

Word-Representations

Are word-representations unique?

- Nope.

Word-Representations

Are word-representations unique?

- Nope.
- Given a word w, the reversed word w^R also generates the same graph.
- In general, a graph can have many word-representations.

Word-Representations

Are word-representations unique?

- Nope.
- Given a word w, the reversed word w^R also generates the same graph.
- In general, a graph can have many word-representations.

Theorem (Kitaev et al.)

Every word-representable graph has a k-uniform word-representation.

- In a k-uniform word, every letter occurs exactly k times.

Word-Representations

Are word-representations unique?

- Nope.
- Given a word w, the reversed word w^R also generates the same graph.
- In general, a graph can have many word-representations.

Theorem (Kitaev et al.)

Every word-representable graph has a k-uniform word-representation.

- In a k-uniform word, every letter occurs exactly k times.
- The minimal such k is the graph's representation number.

Word-Representations

Are word-representations unique?

- Nope.
- Given a word w, the reversed word w^R also generates the same graph.
- In general, a graph can have many word-representations.

Theorem (Kitaev et al.)

Every word-representable graph has a k-uniform word-representation.

- In a k-uniform word, every letter occurs exactly k times.
- The minimal such k is the graph's representation number.
- We focus on only the 2-uniform word-representations, henceforth.

Word-Representations

Are word-representations unique?

- Nope.
- Given a word w, the reversed word w^R also generates the same graph.
- In general, a graph can have many word-representations.

Theorem (Kitaev et al.)

Every word-representable graph has a k-uniform word-representation.

- In a k-uniform word, every letter occurs exactly k times.
- The minimal such k is the graph's representation number.
- We focus on only the 2-uniform word-representations, henceforth.

Cycle Graphs

- Consider the cycle graph C_n labelled $1, 2 \ldots n$ in the clockwise direction, where $n > 3$.

Cycle Graphs

- Consider the cycle graph C_n labelled $1, 2 ... n$ in the clockwise direction, where $n > 3$.
- Note that vertices k and $k + 1$ are connected for each $k \in \{1, 2, ... n - 1\}$, and the edge from n to 1 completes the cycle.

Cycle Graphs

- Consider the cycle graph C_n labelled $1, 2 \ldots n$ in the clockwise direction, where $n > 3$.
- Note that vertices k and $k+1$ are connected for each $k \in \{1, 2, \ldots n-1\}$, and the edge from n to 1 completes the cycle.

The fundamental word

Define the word w_n, on the alphabet $\{1, 2 \ldots n\}$, by

$$w_n = 1n21324354\ldots(n-1)(n-2)n(n-1).$$

Cycle Graphs

- Consider the cycle graph C_n labelled $1, 2 \ldots n$ in the clockwise direction, where $n > 3$.
- Note that vertices k and $k+1$ are connected for each $k \in \{1, 2, \ldots n-1\}$, and the edge from n to 1 completes the cycle.

The fundamental word

Define the word w_n, on the alphabet $\{1, 2 \ldots n\}$, by

$$w_n = 1n21324354\ldots(n-1)(n-2)n(n-1).$$

It is easy to see that $G(w_n) = C_n$ for every $n > 3$. Our claim is that from w_n, we can 'obtain' every word-representation of C_n. We look at one last definition before going on to the proof.

Circle Representations

Observation

If w' is the word obtained by a cyclic shift or a reflection of a 2-uniform word w, then $G(w') = G(w)$.

Circle Representations

Observation

If w' is the word obtained by a cyclic shift or a reflection of a 2-uniform word w, then $G(w') = G(w)$.

Definition

- Represent a 2-uniform word w of length l on a circle, labelled by the letter occurring at each position from 1 to l, clockwise.
- We can imagine joining the two points where a specific letter repeats by a chord.
- Note that two letters alternate iff their corresponding chords intersect.

We call this the **circle representation** of w.

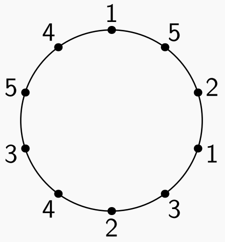


Figure: The circle representation of w_5, with chords not shown, and the start point as the topmost point. Each label represents the number at the position. We can start at any of the 10 positions, and read in either the clockwise or the anticlockwise direction, to get a total of $4 \times 5 = 20$ distinct words.

Main Theorem

Let w be any 2-uniform word that represents C_n labelled $1, 2, 3 ... n$ for $n > 3$, and let $w[i]$ be the letter at position i for all $1 \leq i \leq 2n$.

Main Theorem

Let w be any 2-uniform word that represents C_n labelled $1, 2, 3 \ldots n$ for $n > 3$, and let $w[i]$ be the letter at position i for all $1 \leq i \leq 2n$. Then, the **circle representation** of w satisfies the following property:

Main Theorem

Let w be any 2-uniform word that represents C_n labelled $1, 2, 3 \ldots n$ for $n > 3$, and let $w[i]$ be the letter at position i for all $1 \leq i \leq 2n$. Then, the **circle representation** of w satisfies the following property:

Lemma

For every $r \in \{1, 2, 3 \ldots n\}$, the two sets of positions,

$$U_r = \{i : (w[i] - r) > 1\} \text{ and } L_r = \{i : (r - w[i]) > 1\},$$

if both are non-empty, lie entirely in one of the two segments defined by the chord corresponding to r. If exactly one is non-empty, then that set lies entirely in one of two segments.

The Main Theorem - Visual

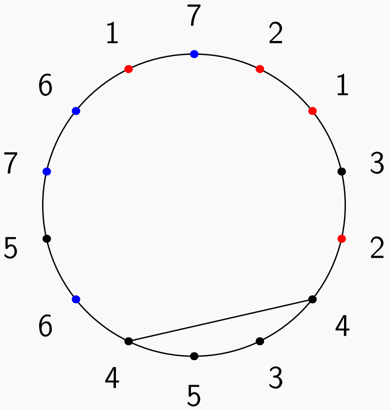
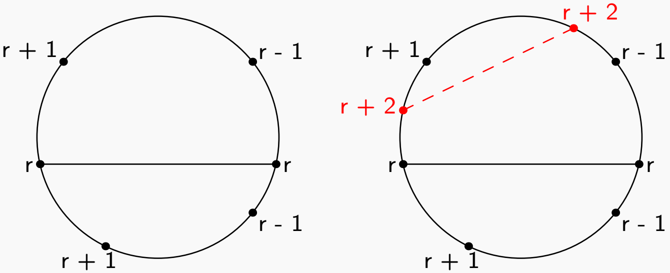
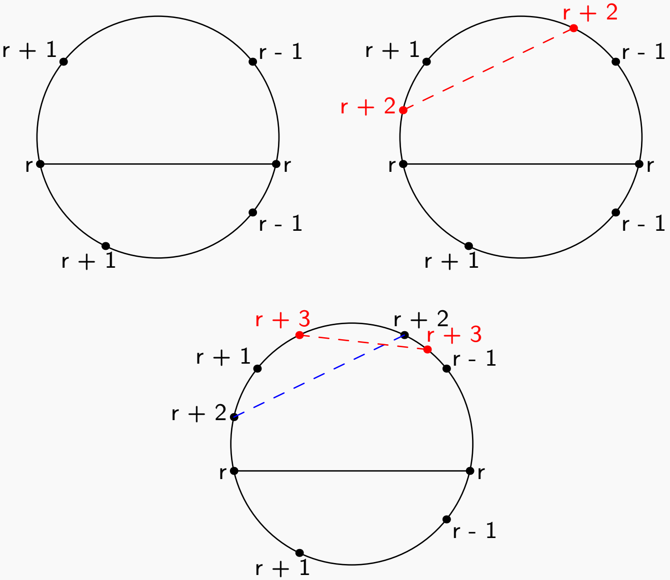


Figure: Circle representation of $w_7 = 17213243546576$, representing C_7. The chord corresponding to $r = 4$ has been drawn. The two sets of points, U_r and L_r have been coloured in blue and red, respectively. Black points belong to neither of the two sets.

The Proof Idea - Visual



The Proof Idea - Visual



The Graphcheck Algorithm - Preliminaries

Following on from the circle representation idea, we aim to obtain an algorithm that will help us check if a given word 2-uniform word w is a word-representation of a given graph G.

The Graphcheck Algorithm - Preliminaries

Following on from the circle representation idea, we aim to obtain an algorithm that will help us check if a given word 2-uniform word w is a word-representation of a given graph G.
For this, we utilize a data structure that handles **Dynamic Range Sum Queries** efficiently - the Fenwick Tree.

The Graphcheck Algorithm - Preliminaries

Following on from the circle representation idea, we aim to obtain an algorithm that will help us check if a given word 2-uniform word w is a word-representation of a given graph G.
For this, we utilize a data structure that handles **Dynamic Range Sum Queries** efficiently - the Fenwick Tree.

The Fenwick Tree (Tarjan et al.)

- Used to calculate prefix sums of an array - say, of length n.
- $O(n)$ additional space.
- $O(n \log n)$ time initialization for an arbitrary array. (Here, however, this will not be necessary.)
- $O(\log n)$ time for a point update.
- $O(\log n)$ for an arbitrary prefix sum.

The Graphcheck Algorithm

Algorithm Details

- **Input**: 2-uniform word w on V, and graph $G = (V, E)$.
- **Result**: Returns **true** if $G(w) = G$, and **false** otherwise.

Initialization

- Initialize FenwickTree with 0 in all positions with total length $w.length()$.
- Initialize array of positions $pos[]$ to (NULL, NULL) for all letters in w.
- edgecount $= 0$

```
for k = 0 to w.length() − 1 do
    if pos[w[k]].first = NULL then
        pos[w[k]].first = k
    else
        pos[w[k]].second = k

        i = pos[w[k]].first
        j = pos[w[k]].second

        // add the number of unmarked nodes in w[i...j]
        edgecount +=
         j − i − FenwickTree.rangesum(i + 1, j − 1) − 1

        // mark the positions i and j
        FenwickTree.update(i, 1)
        FenwickTree.update(j, 1)
    end
end
```

```
if edgecount ≠ |E| then
    return false
else
    for edge (u, v) in E do
        if u and v do not alternate then
            return false ;            // only a O(1) comparison
        end
    end
    return true
end
```

Algorithm 1: GraphCheck

References and Credits

Beamer Theme by Cédric Mauclair.

Sergey Kitaev and Vadim V. Lozin.
Words and Graphs.
Monographs in Theoretical Computer Science. An EATCS Series. Springer, 2015.

A. Collins, S. Kitaev, and V. Lozin.
New results on word-representable graphs.
ArXiv e-prints, July 2013.

Magnús M. Halldórsson, Sergey Kitaev, and Artem V. Pyatkin.
Semi-transitive orientations and word-representable graphs.
Discrete Applied Mathematics, 201:164–171, 2016.

Peter M. Fenwick.
A new data structure for cumulative frequency tables.
Softw., Pract. Exper., 24(3):327–336, 1994.

Ameya Daigavane.
Implementing the graphcheck algorithm for 2-uniform words, 2017.

Thank you!