

# PTAS For MIS On Planar Graphs

Sourabh Aggarwal

Department of Computer Science And Engineering  
IIT Palakkad

March 2019

## 1 Prerequisite

- Recall
- Outerplanar Graph
- Fixed Parameter Tractability

## 2 High Level Plan

## 3 Bakers Algorithm

- Theorem 1
- Main Algorithm
- Analysis of the given Algorithm
- Theorem 2
- Linear Time Algorithm for Outerplanar Graphs

## 1 Prerequisite

- Recall
- Outerplanar Graph
- Fixed Parameter Tractability

## 2 High Level Plan

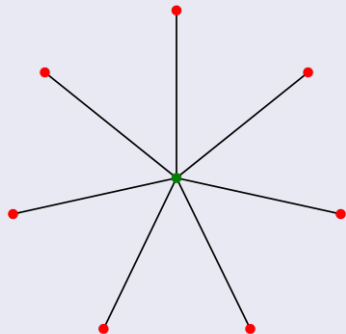
## 3 Bakers Algorithm

- Theorem 1
- Main Algorithm
- Analysis of the given Algorithm
- Theorem 2
- Linear Time Algorithm for Outerplanar Graphs

# Prerequisite: Recall

## Example

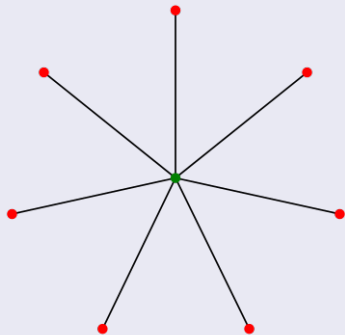
What is  $|MIS|$  for  $S_n$ ?



# Prerequisite: Recall

## MIS

What is  $|MIS|$  for  $S_n$ ? Ans.  $n - 1$



# Prerequisite: Recall

## Example

What is  $|MIS|$  for  $K_n$ ?

# Prerequisite: Recall

MIS

What is  $|MIS|$  for  $K_n$ ? Ans. 1

# Prerequisite: Recall

## Euler's formula

If a finite, connected, planar graph is drawn in the plane without any edge intersections, and  $v$  is the number of vertices,  $e$  is the number of edges and  $f$  is the number of faces (regions bounded by edges, including the outer, infinitely large region), then  $v - e + f = 2$



## 1 Prerequisite

- Recall
- **Outerplanar Graph**
- Fixed Parameter Tractability

## 2 High Level Plan

## 3 Bakers Algorithm

- Theorem 1
- Main Algorithm
- Analysis of the given Algorithm
- Theorem 2
- Linear Time Algorithm for Outerplanar Graphs

# Outerplanar Graphs

## Definition

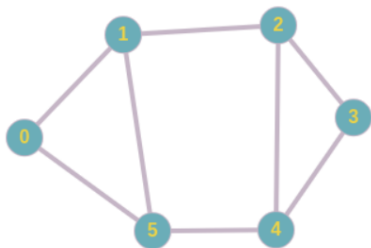
A graph is called outerplanar (1-outerplanar) if it has an embedding in the plane such that every vertex lies on the unbounded face.

## In Simple Terms

$\Leftrightarrow$  has a planar drawing for which all vertices belong to the outer face of the drawing.

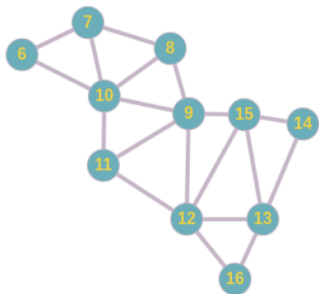
# Outerplanar Graphs

## Example



# Outerplanar Graphs

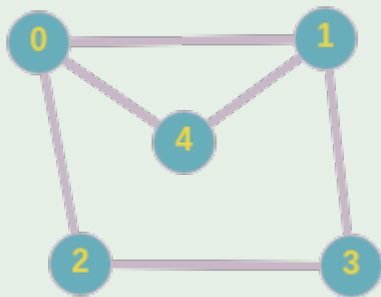
## Example



# Outerplanar Graphs

## Example

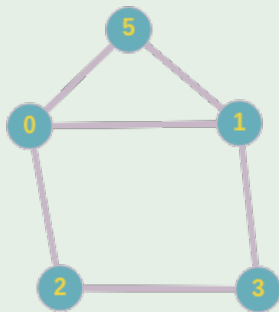
Is this outerplanar?



# Outerplanar Graphs

## Example

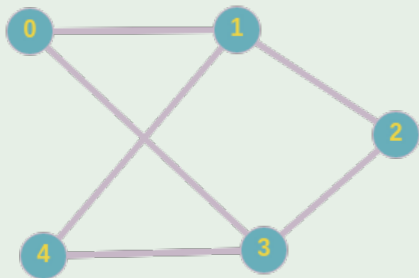
Yes, as it is isomorphic to the following graph.



# Outerplanar Graphs

## Example

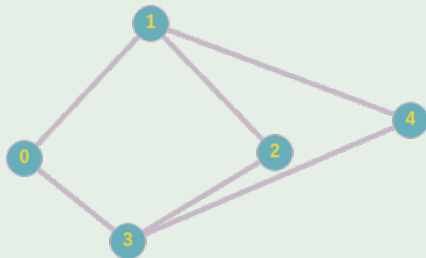
Is this outerplanar?



# Outerplanar Graphs

## Example

No, but it is planar.





## Definition

Given a planar embedding  $E$  of a planar graph  $G$ , we define  $k$ -level nodes according to the faces of the embedding. Every node on the exterior face is called an exterior or level 1 node. Exterior edges are defined analogously. Call a cycle of level  $i$  nodes a *level  $i$  face* if it is an interior face in the subgraph induced by the level  $i$  nodes. For each *level  $i$  face*  $f$ , let  $G_f$  be the subgraph induced by all nodes placed inside  $f$  in this embedding. Then the nodes on the exterior face of  $G_f$  are at level  $i + 1$ .

## Definition

Basically, in each step, starting at 1, remove all nodes on the exterior face. Repeat until there are no nodes left. The level of a node is the step in which it was removed. Using an appropriate data structure for the planar embedding, such as the one used by Lipton and Tarjan [Lipton and Tarjan, 1979], this can be done in linear time. A planar embedding is called  $k$ -outerplanar if there are no nodes of level  $> k$ , which means the algorithm terminates after at most  $k$  steps. A planar graph is called  $k$ -outerplanar if it has a  $k$ -outerplanar embedding.

# Example

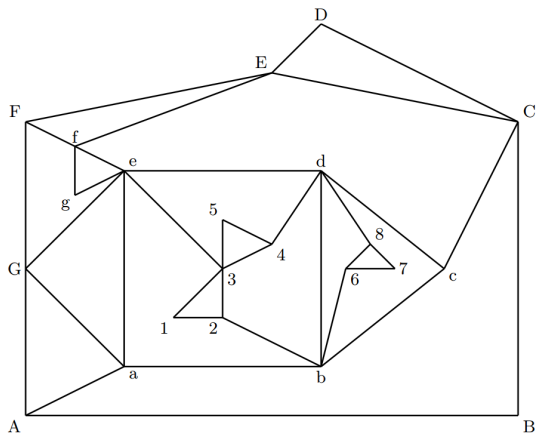


Figure 1. A planar embedding of a 3-outerplanar graph [Baker, 1994].

## 1 Prerequisite

- Recall
- Outerplanar Graph
- Fixed Parameter Tractability

## 2 High Level Plan

## 3 Bakers Algorithm

- Theorem 1
- Main Algorithm
- Analysis of the given Algorithm
- Theorem 2
- Linear Time Algorithm for Outerplanar Graphs

## Definition

A problem is considered fixed parameter tractable if an algorithm exists which solves the problem in running time  $f(k) * n^{O(1)}$  where  $k$  is an additional parameter depending upon input and  $n$  is the size of the input (in our case we will treat it as number of nodes) and  $f$  is an arbitrary function depending only on  $k$ .

# High Level Plan

We will use this new notion of  $k$ -outerplanarity to construct a polynomial time approximation scheme for maximum independent set on planar graphs, which can be easily adapted to other NP-complete problems. The goal is an algorithm that, for a freely chosen but fixed  $k$ , solves maximum independent set with an approximation ratio of  $k/(k+1)$  and a linear complexity with respect to the number of nodes  $n$ . Basically, we want to exactly solve maximum independent on disjunct subgraphs that are  $k$ -outerplanar and ignore some nodes in the graph to merge the subgraphs without violating the conditions of the maximum independent set.

**PS:** Approximation Ratio? For maximization problems it is atleast  $(1 - \epsilon)$  therefore fix  $k = (1/\epsilon) - 1$  (do math).

## 1 Prerequisite

- Recall
- Outerplanar Graph
- Fixed Parameter Tractability

## 2 High Level Plan

## 3 Bakers Algorithm

- Theorem 1
- Main Algorithm
- Analysis of the given Algorithm
- Theorem 2
- Linear Time Algorithm for Outerplanar Graphs

# Bakers Algorithm: Theorem 1

## Theorem (Baker, 1994)

*Let  $k$  be a positive integer. Given a  $k$ -outerplanar embedding of a  $k$ -outerplanar graph  $G$ , an optimal solution for maximum independent set can be obtained in time  $O(8^k n)$ , where  $n$  is the number of nodes.*

Its proof will be done later.



## 1 Prerequisite

- Recall
- Outerplanar Graph
- Fixed Parameter Tractability

## 2 High Level Plan

## 3 Bakers Algorithm

- Theorem 1
- **Main Algorithm**
- Analysis of the given Algorithm
- Theorem 2
- Linear Time Algorithm for Outerplanar Graphs

# Bakers Algorithm: Main Algorithm

- ① Generate a planar embedding of  $G$  using the linear time algorithm of Hopcroft and Tarjan [Hopcroft and Tarjan, 1974] and compute the level of every node.
- ② For each  $i$ ;  $0 \leq i \leq k$ , do the following:
  - ① Remove all nodes where the nodes level mod  $(k + 1)$  equals  $i$ , splitting the graph into components with an (already computed)  $k$ -outerplanar embedding.
  - ② Use Theorem 1 to exactly solve maximum independent set on all those components and take the union of the solutions. This is possible as the components are disjunct and no edges can exist between two components.
- ③ Take the best solution for all  $i$  as the solution for the entire graph.

## 1 Prerequisite

- Recall
- Outerplanar Graph
- Fixed Parameter Tractability

## 2 High Level Plan

## 3 Bakers Algorithm

- Theorem 1
- Main Algorithm
- Analysis of the given Algorithm
- Theorem 2
- Linear Time Algorithm for Outerplanar Graphs

# Bakers Algorithm: Analysis of the given Algorithm

To see that this gives a solution with approximation ratio  $k/(k+1)$ , consider the optimal solution  $S_{OPT}$ . For some  $i$  as defined above, at most  $1/(k+1)$  of the nodes in  $S_{OPT}$  are at a level congruent to  $i$

[As suppose this was not the case  $\rightarrow S_{OPT} > (k+1) * (1/(k+1) * S_{OPT})$  i.e.  $S_{OPT} > S_{OPT}$  which is absurd].

As the other components are solved exactly, the union of their solutions has at least the size of  $S_{OPT} - 1/(k+1) * S_{OPT}$  i.e.  $k/(k+1) * S_{OPT}$  giving our desired approximation factor.

[As  $Rem \leq S_{OPT}/(k+1) \rightarrow -Rem \geq -S_{OPT}/(k+1) \rightarrow S_{OPT} - Rem \geq k/(k+1) * S_{OPT}$ ]

This leads to the second theorem:

## 1 Prerequisite

- Recall
- Outerplanar Graph
- Fixed Parameter Tractability

## 2 High Level Plan

## 3 Bakers Algorithm

- Theorem 1
- Main Algorithm
- Analysis of the given Algorithm
- **Theorem 2**
- Linear Time Algorithm for Outerplanar Graphs

# Bakers Algorithm: Theorem 2

## Theorem (Baker, 1994)

*For fixed  $k$ , there is an  $O(8^k kn)$ -time algorithm for maximum independent set that achieves a solution of size at least  $k/k + 1$  optimal for general planar graphs.*

We have proved this [assuming theorem 1]. So, what remains is to prove theorem 1

## 1 Prerequisite

- Recall
- Outerplanar Graph
- Fixed Parameter Tractability

## 2 High Level Plan

## 3 Bakers Algorithm

- Theorem 1
- Main Algorithm
- Analysis of the given Algorithm
- Theorem 2
- Linear Time Algorithm for Outerplanar Graphs

# Bakers Algorithm: Linear Time Algorithm for Outerplanar Graphs

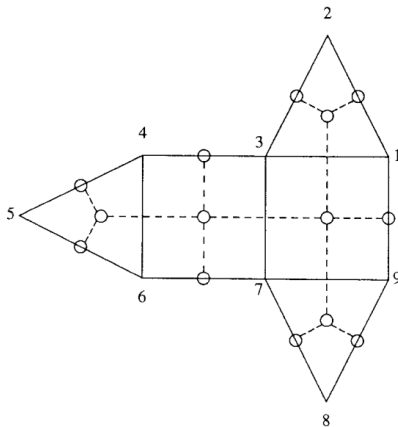
Given a connected outerplanar graph, do the following

- Replace each bridge  $(x, y)$  by two edges between  $x$  and  $y$ . This will allow us to treat a bridge as a face rather than as a special case. Call the resulting graph  $G$ .
- The maximum independent set for  $G$  will be computed by recursively processing an ordered, rooted tree  $\bar{G}$  that represents the structure of  $G$ . Each leaf of  $\bar{G}$  will represent an exterior edge of  $G$  and every other vertex will represent a face of  $G$  and will be called a *face vertex*.  $\bar{G}$  is constructed as follows:
  - 1 Construct a graph vertex for every interior face and every exterior edge.
  - 2 Draw edges between every edge vertex and the vertex of the face the edge is contained in, and between every two vertices of faces that share an edge.
  - 3 Need to handle cut points (presented later)



# Bakers Algorithm: Linear Time Algorithm for Outerplanar Graphs

FIG. 3. An outerplanar graph (with no cutpoints) and a tree (drawn with dashed lines).



# Bakers Algorithm: Linear Time Algorithm for Outerplanar Graphs

## Question

Why are we getting a Tree?

# Bakers Algorithm: Linear Time Algorithm for Outerplanar Graphs

## Question

Why are we getting a Tree? Will we always get a tree for a planar embedding?

# Bakers Algorithm: Linear Time Algorithm for Outerplanar Graphs

## Question

Why are we getting a Tree? Will we always get a tree for a planar embedding - Ans. No, consider planar embedding of  $K_4$ . So, this property is actually the result of outerplanarity condition.

# Bakers Algorithm: Linear Time Algorithm for Outerplanar Graphs

## Labelling our tree

- 1 The planar embedding induces a cyclic ordering on the edges of each vertex in the tree [for our discussion, we will follow counterclockwise orientation]. Choosing a face vertex  $v$  as the root and choosing which child of  $v$  is to be its leftmost child determine the parent and ordering of children for every other vertex of  $G$  [We will see this in diagram].
- 2 Label the vertices of  $\bar{G}$  recursively, as follows: Label each leaf of the tree with the oriented exterior edge it represents. Label each face vertex with the first and last nodes in its childrens labels.
- 3 If a face vertex is labeled  $(x, y)$ , the leaves of its subtree represent a directed walk of exterior edges in a counterclockwise direction from  $x$  to  $y$ . For the root,  $x = y$  and the directed walk covers all the exterior edges. For any other face vertex  $v$ ,  $x \neq y$ , and  $(x, y)$  is an interior edge shared by the face represented by  $v$  and the face represented by its parent in the tree.

# Bakers Algorithm: Linear Time Algorithm for Outerplanar Graphs

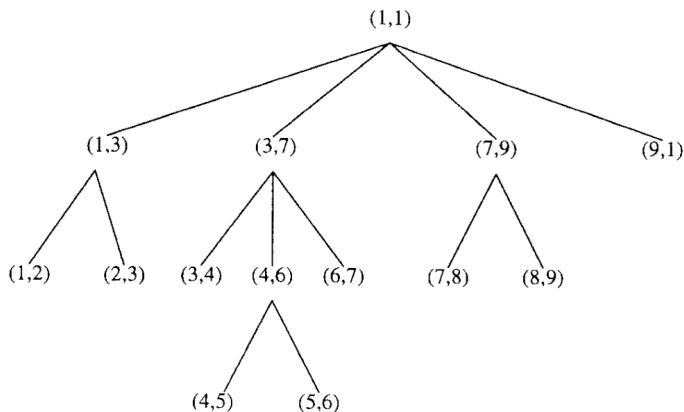


FIG. 4. A rooted, ordered, labeled tree  $\overline{G}$  for the graph  $G$  of Figure 3.

# Bakers Algorithm: Linear Time Algorithm for Outerplanar Graphs

## Handling Cutpoints

Cutpoints are nodes, whose removal disconnects the graph. As we removed bridges, a cutpoint is a node where at least two faces meet without sharing an edge, therefore disconnecting the tree for  $G$ . Draw an edge between two vertices of faces that both contain the cutpoint and are part of different components until  $G$  is connected.

# Bakers Algorithm: Linear Time Algorithm for Outerplanar Graphs

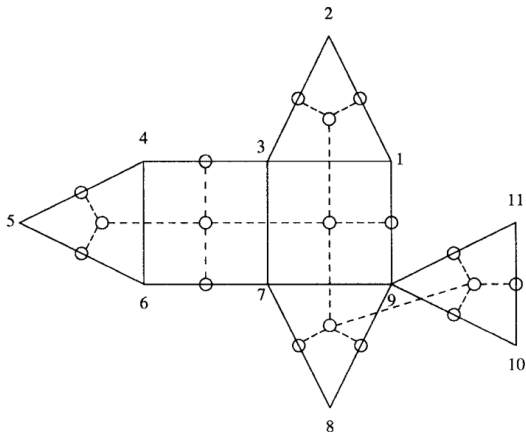


FIG. 5. An outerplanar graph with a cutpoint (node 9) and a corresponding tree (shown by dashed lines).



# Bakers Algorithm: Linear Time Algorithm for Outerplanar Graphs

## Labelling our tree

Labelling is done same as before. Note that now it is possible for a face vertex labelled  $(x, y)$  and  $x = y$  in which case, the label represents a cutpoint shared by two faces, as for the vertex labeled  $(9,9)$  in below figure.

# Bakers Algorithm: Linear Time Algorithm for Outerplanar Graphs

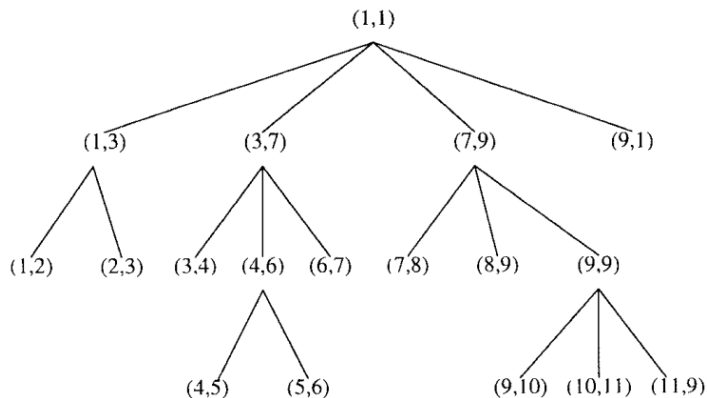


FIG. 6. A rooted ordered, labeled tree  $\overline{G}$  for the graph  $G$  of Figure 5.