# Visualisation of graphs

## Planarity with y-Files

Antonios Symvonis · Chrysanthi Raftopoulou

Fall semester 2020

# Introduction

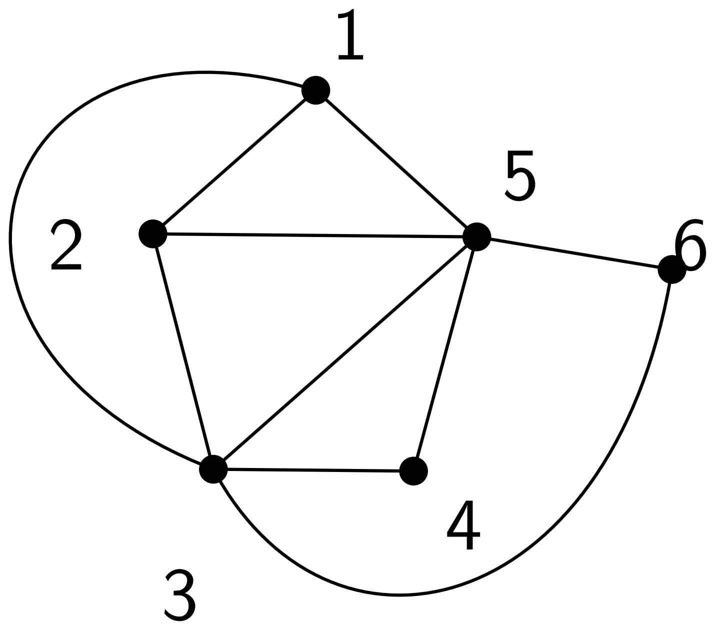
**Classes:**

- **Dart**
  - models an edge as part of a face
- **PlanarEmbedding**
  - models the planar embedding

# Introduction

**Classes:**

- **Dart**
  - models an edge as part of a face
- **PlanarEmbedding**
  - models the planar embedding

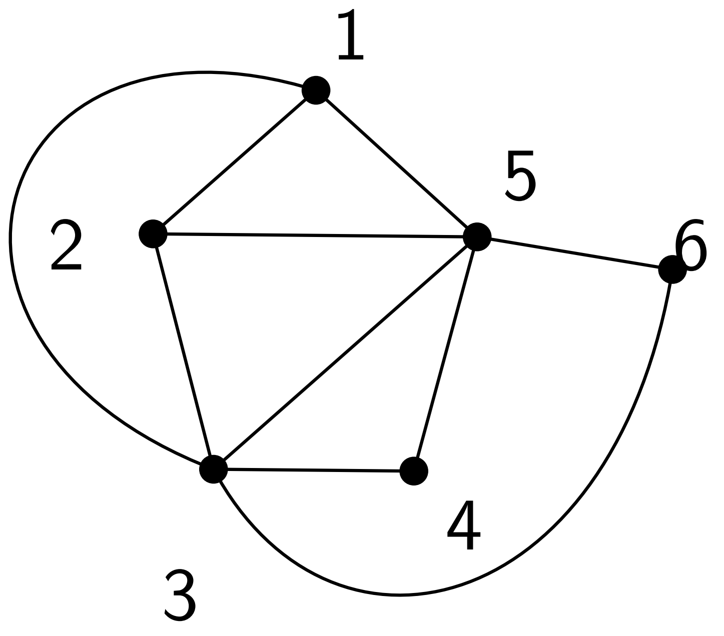


# Introduction

**Classes:**

- **Dart**
  - models an edge as part of a face
- **PlanarEmbedding**
  - models the planar embedding

- each edge is associated to two darts
- each face is a list of darts
  – *in clockwise order along the boundary of the face*
- the cyclic order of the darts around a vertex is part of the embedding

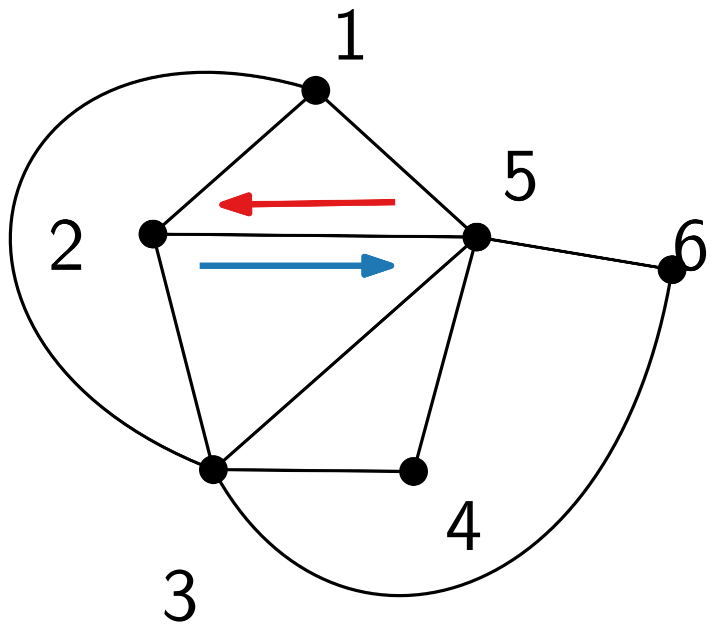


# Introduction

**Classes:**

- **Dart**
  - models an edge as part of a face
- **PlanarEmbedding**
  - models the planar embedding

- each edge is associated to two darts
- each face is a list of darts
  – *in clockwise order along the boundary of the face*
- the cyclic order of the darts around a vertex is part of the embedding

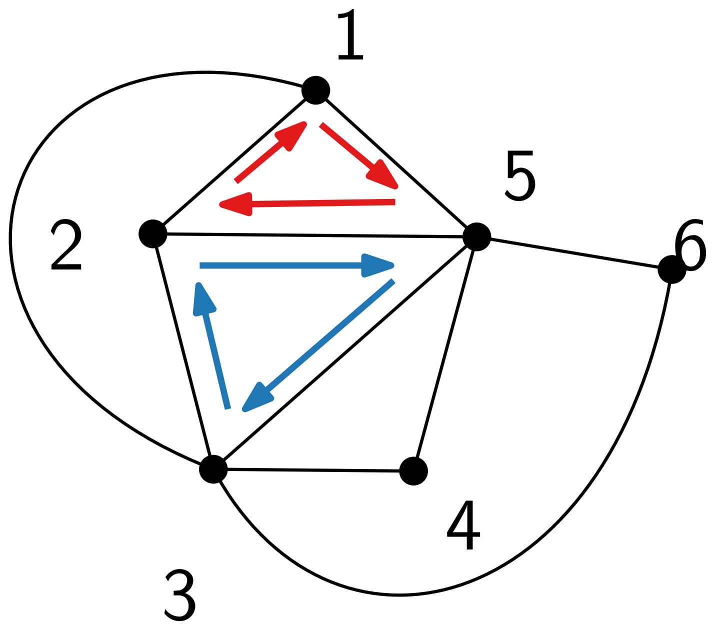


## Introduction

**Classes:**

- **Dart**
  - models an edge as part of a face
- **PlanarEmbedding**
  - models the planar embedding

- each edge is associated to two darts
- each face is a list of darts
  – *in clockwise order along the boundary of the face*
- the cyclic order of the darts around a vertex is part of the embedding

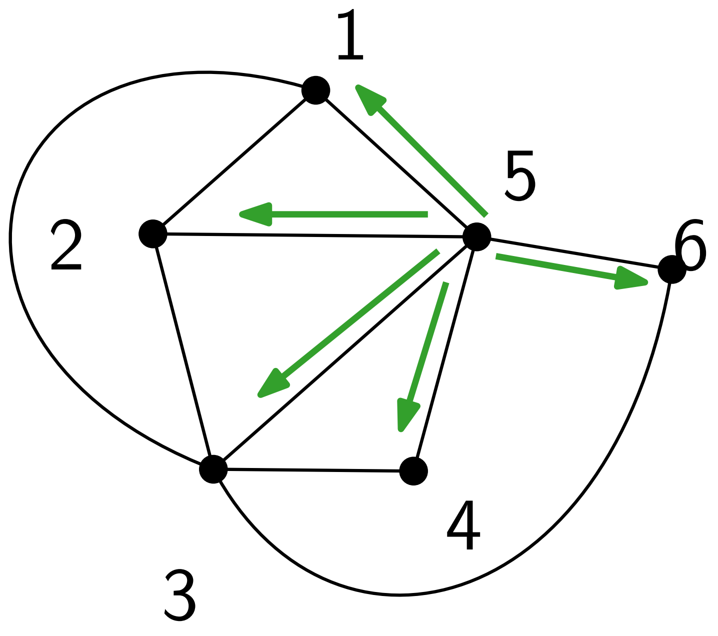


# Introduction

**Classes:**

- **Dart**
  - models an edge as part of a face
- **PlanarEmbedding**
  - models the planar embedding

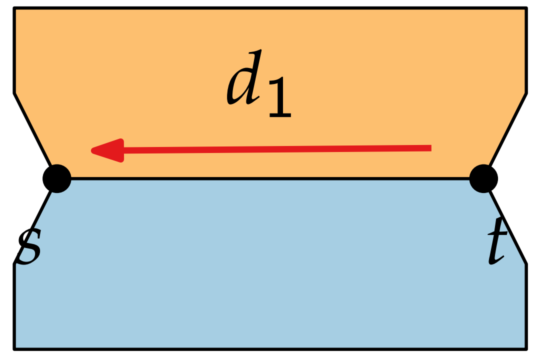
- each edge is associated to two darts
- each face is a list of darts
  – *in clockwise order along the boundary of the face*
- the cyclic order of the darts around a vertex is part of the embedding



# Dart

**Methods:**

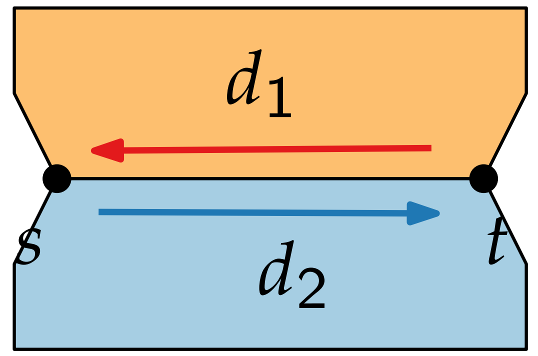
- Dart getOppositeDart( )
- Edge getAssociatedEdge( )
- boolean isReversed( )
- List <Dart> getFace( )



# Dart

**Methods:**

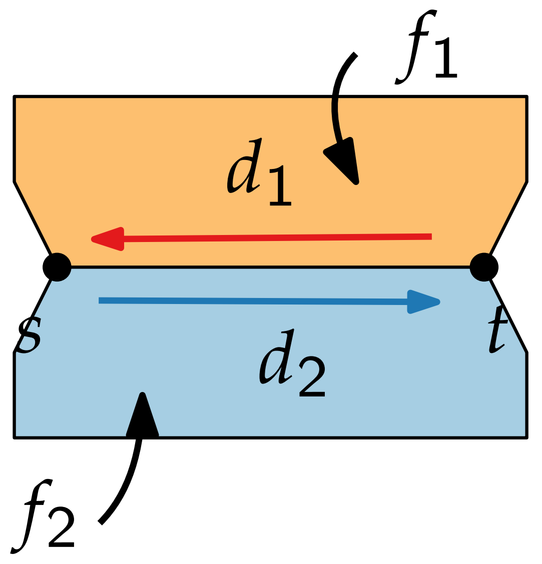
- Dart getOppositeDart( )
- Edge getAssociatedEdge( )
- boolean isReversed( )
- List <Dart> getFace( )



# Dart

**Methods:**

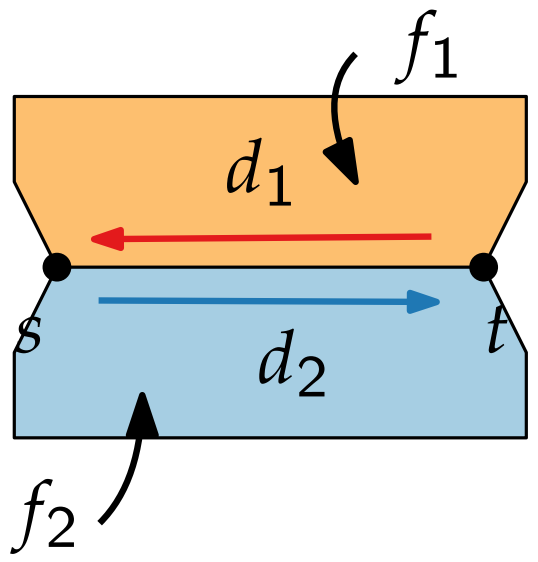
- Dart getOppositeDart( )
- Edge getAssociatedEdge( )
- boolean isReversed( )
- List <Dart> getFace( ) in clockwise-order



# Dart

**Methods:**

- Dart getOppositeDart( )
- Edge getAssociatedEdge( )
- boolean isReversed( )
- List <Dart> getFace( ) in clockwise-order



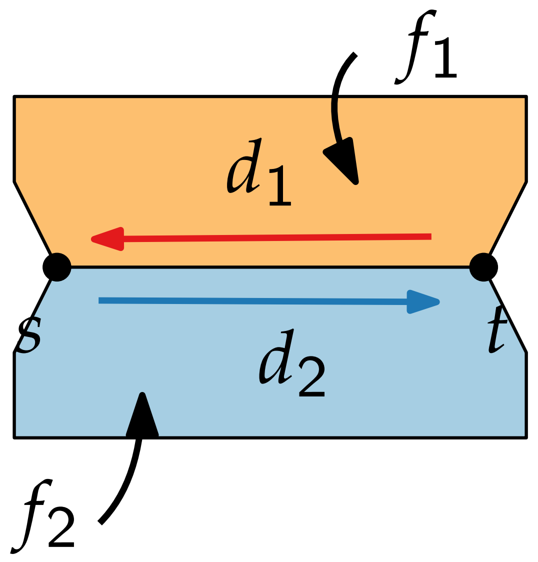
**Example:**

```
{
    Dart d1 = ...;
    List<Dart> f1 = d1.getFace( );
    Dart d2 = d1.getOppositeDart( );
    List<Dart> f2 = d2.getFace( );
}
```

## Dart

**Methods:**

- Dart getOppositeDart( )
- Edge getAssociatedEdge( )
- boolean isReversed( )
- List <Dart> getFace( ) in clockwise-order



**Example:**

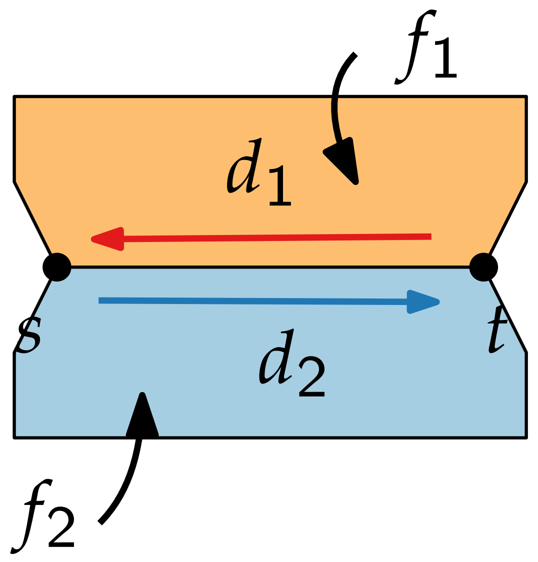
```
{
    Dart d1 = ...;
    List<Dart> f1 = d1.getFace( );
    Dart d2 = d1.getOppositeDart( );
    List<Dart> f2 = d2.getFace( );
}
```

Classes Dart and PlanarEmbedding are associated with instances of Graph (or GraphLayout), not IGraph!!

# Dart

**Methods:**

- Dart getOppositeDart( )
- Edge getAssociatedEdge( )
- boolean isReversed( )
- List <Dart> getFace( )



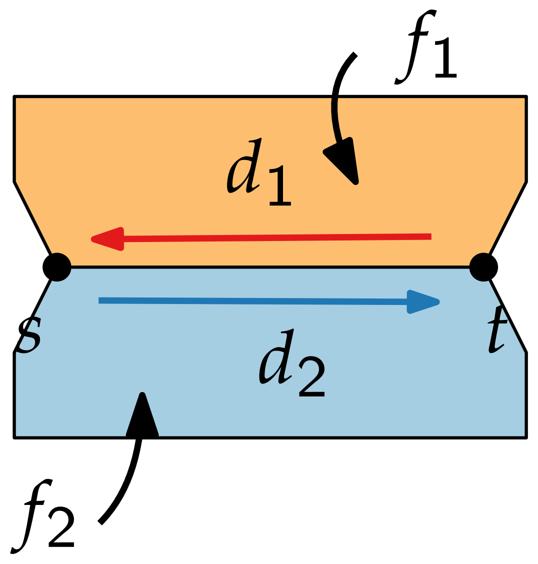
# Dart

**Methods:**

- Dart getOppositeDart( )
- Edge getAssociatedEdge( )
- boolean isReversed( )
- List <Dart> getFace( )

**Example:**

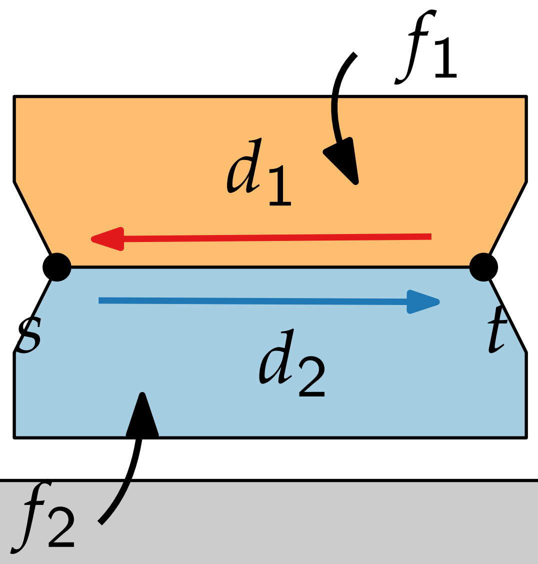
```
{
    Dart d1 = ...;
    Dart d2 = d1.getOppositeDart( );
    Edge e1 = d1.getAssociatedEdge( );
    Edge e2 = d2.getAssociatedEdge( );
     //e1==e2
}
```



# Dart

**Methods:**

- Dart getOppositeDart( )
- Edge getAssociatedEdge( )
- boolean isReversed( )
- List <Dart> getFace( )



**Example:**

```
{
    Dart d1 = ...;
    Dart d2 = d1.getOppositeDart( );
    Edge e1 = d1.getAssociatedEdge( );
    Edge e2 = d2.getAssociatedEdge( );
     //e1==e2
}
```

**Example:**

```
{
    Dart d1 = ...;
    Edge e1 = d1.getAssociatedEdge( );
    Node s1 = null;
    if (!d1.isReversed( ))
        s1 = e1.source( );
    else
        s1 = e1.target( );
}
```

# PlanarEmbedding

**Constructor:**

- PlanarEmbedding(Graph)

## PlanarEmbedding

**Constructor:**

- PlanarEmbedding(Graph)

**Methods:**

- List<List<Dart>> getFaces( )
- List<Dart>getOuterFace( )

] darts of faces in clockwise order

# PlanarEmbedding

**Constructor:**

- PlanarEmbedding(Graph)

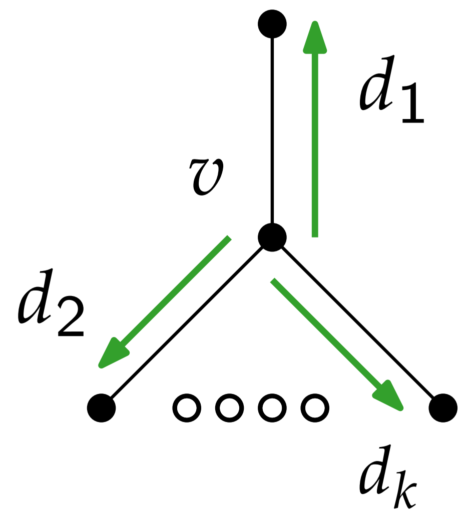
**Methods:**

- List<List<Dart>> getFaces( )
- List<Dart>getOuterFace( )

darts of faces in clockwise order

- Dart getCyclicNext(Dart)
- Dart getCyclicPrevious(Dart)
- List<Dart> getOutgoindDarts(Node)

darts around a vertex in counter-clockwise order



# PlanarEmbedding

**Constructor:**

- PlanarEmbedding(Graph)

**Methods:**

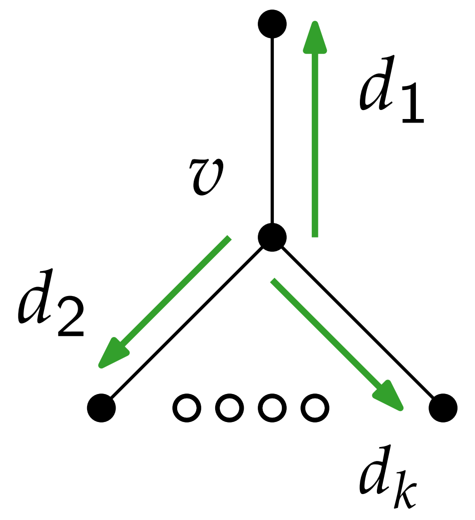
- List<List<Dart>> getFaces( )
- List<Dart>getOuterFace( )

darts of faces in clockwise order

- Dart getCyclicNext(Dart)
- Dart getCyclicPrevious(Dart)
- List<Dart> getOutgoindDarts(Node)

darts around a vertex in counter-clockwise order

- (static) boolean isPlanar(Graph)



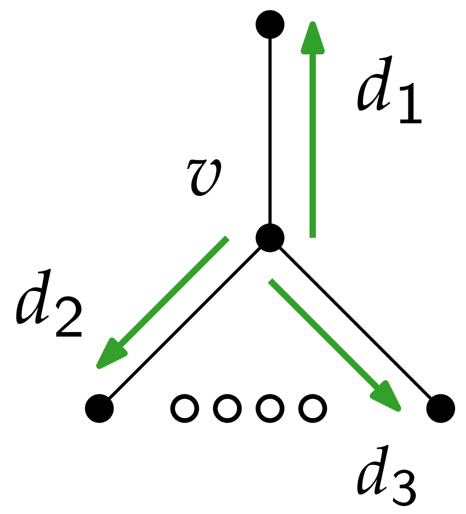
# PlanarEmbedding

**Constructor:**

- PlanarEmbedding(Graph)

**Methods:**

- List<List<Dart>> getFaces( )
- List<Dart>getOuterFace( )
- Dart getCyclicNext(Dart)
- Dart getCyclicPrevious(Dart)
- List<Dart> getOutgoindDarts(Node)
- (static) boolean isPlanar(Graph)



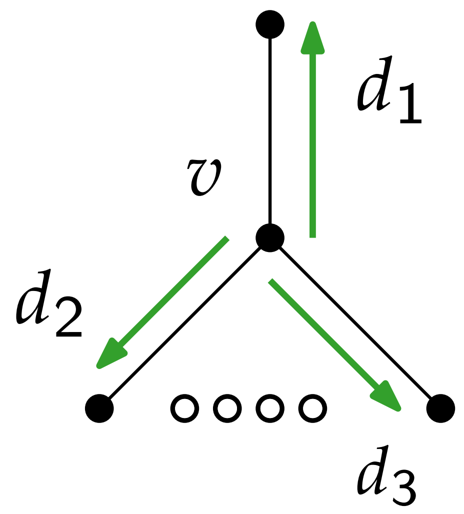
# PlanarEmbedding

**Constructor:**

- PlanarEmbedding(Graph)

**Methods:**

- List<List<Dart>> getFaces( )
- List<Dart>getOuterFace( )
- Dart getCyclicNext(Dart)
- Dart getCyclicPrevious(Dart)
- List<Dart> getOutgoindDarts(Node)
- (static) boolean isPlanar(Graph)



**Example:**

```
{
    Graph g = ...;
    PlanarEbedding emb =
        new PlanarEmbedding(g);
    Dart d1 = ...;
    Dart d = emb.getCyclicNext(d1);
    while (d != d1) {
        //do some process
        d = emb.getCyclicNext(d);
    }
}
```