

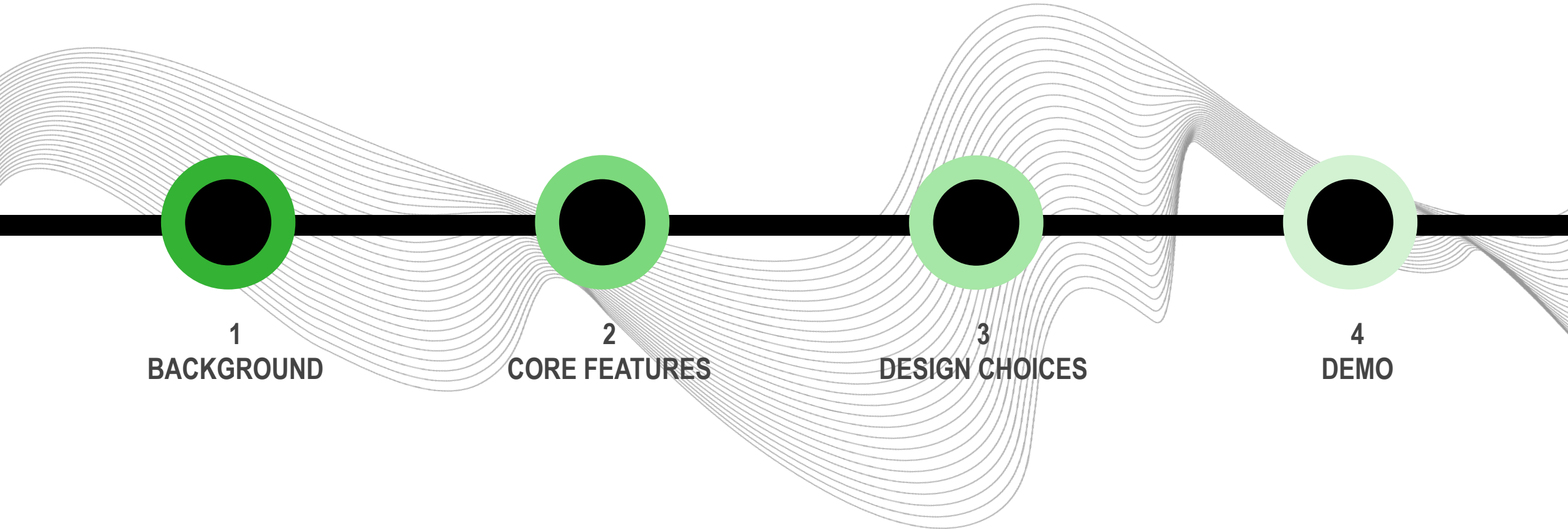
UNIVERSITY OF TWENTE.

GUI FOR PARITY GAME

APR 8, 2021

GROUP 11,
WEITING CAI S2067684
DI ZHUANG S2131080
RUILIN YANG S2099497

IN THIS PRESENTATION:



1

BACKGROUND

2

CORE FEATURES

3

DESIGN CHOICES

4

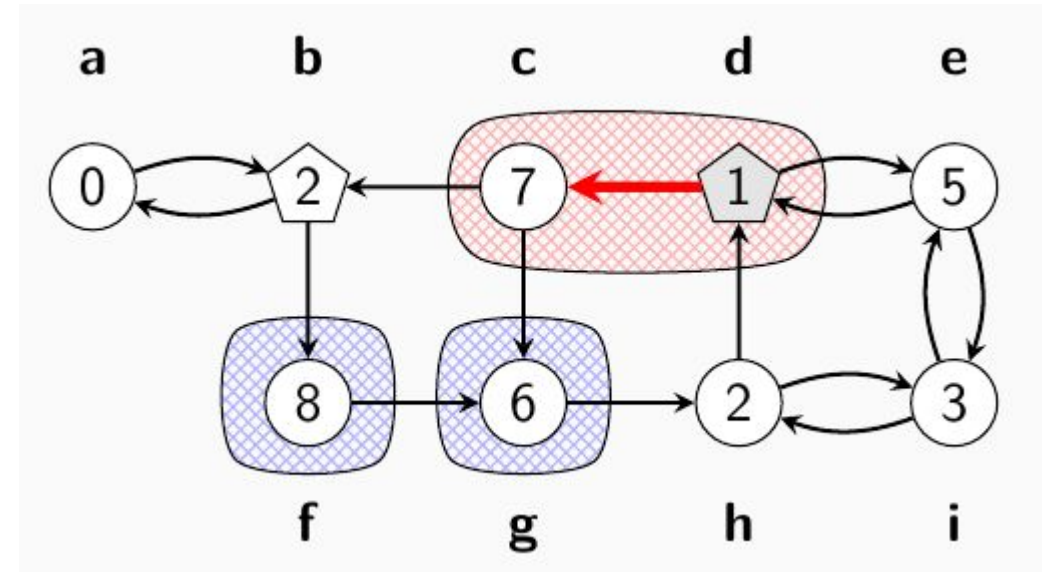
DEMO

The background features a series of thin, grey, wavy lines that flow across the slide. Overlaid on these are several green geometric shapes, including triangles and polygons, some of which are shaded in black to create a 3D effect. These shapes are arranged in a way that suggests a map of the Netherlands.

1 BACKGROUND

BACKGROUND - ABOUT THE PROJECT

- Parity game
 - Logic game, directed graph
 - Application in model checking
 - Winner of each vertex? Strategy?
 - Polynomial time?
- Algorithm research
 - Find game that defeats algorithm
 - Find algorithm that defeats game



BACKGROUND - WHY NEED A GUI? - 1

- Frequent edit of the graph
 - It's a pain to dive into .pg files :(
 - Laborious and error-prone



```
1 parity 10;
2 0 0 0 1,8;
3 1 1 0 7,6;
4 2 2 1 0,4;
5 3 4 1 9,5;
6 4 5 1 2;
7 5 6 0 5,2;
8 6 7 1 0,2;
9 7 9 1 6,3;
10 8 11 1 5;
11 9 12 1 6;
12
```

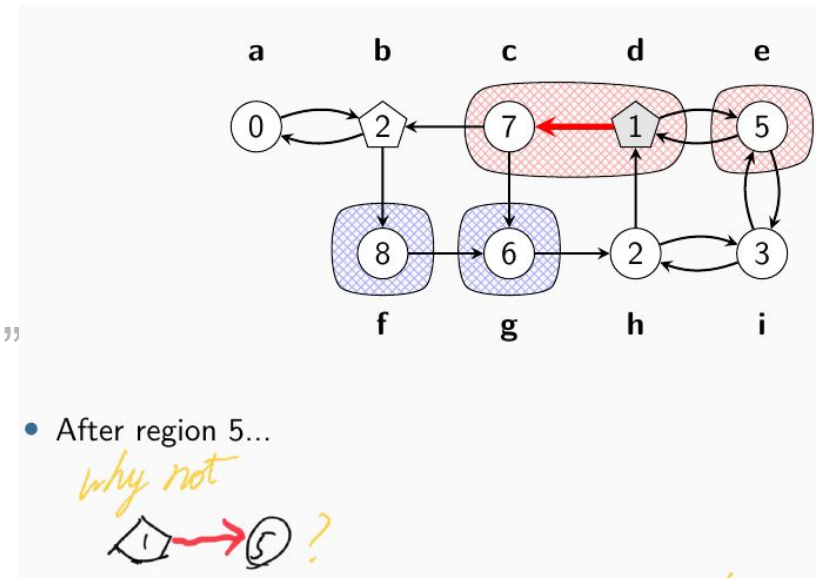

BACKGROUND - WHY NEED A GUI? - 2

- Frequent edit of the graph
 - It's a pain to dive into .pg files :(
 - Laborious and error-prone
- Better understand the algorithm
 - “A Picture is Worth a Thousand Words”

```
Variables
+  this = {DFI@1010}
  > f Z = {HashSet@1016} size = 1
  > f F = {HashMap@1011} size = 2
  > f S = {HashMap@1017} size = 1
  > f gameStatus = {GameStatus@1020} size = 5
    > {Integer@1420} 38 -> {HashMap@1421} size = 7
    > {Integer@1422} 39 -> {HashMap@1423} size = 7
    > {Integer@1424} 40 -> {HashMap@1425} size = 7
    > {Integer@1426} 41 -> {HashMap@1427} size = 7
    > {Integer@1428} 42 -> {HashMap@1429} size = 7
  > f steps = {ArrayList@1021} size = 1
    f solved = false
  > p pg = {Game@1012} "Parity game: \n[38, 39, 40, 41, 42]\n4"
  > vertices = {ArrayList@1013} size = 5
  01 somethingchanged = true
  > vertex = {Vertex@1403} "41"
```

BACKGROUND - WHY NEED A GUI? - 3

- Frequent edit of the graph
 - It's a pain to dive into .pg files :(
 - Laborious and error-prone
- Better understand the algorithm
 - "A Picture is Worth a Thousand Words"
- Secondary purposes
 - Teaching aid
 - Slides → "once and for all"
 - Extend to other graph algorithm research



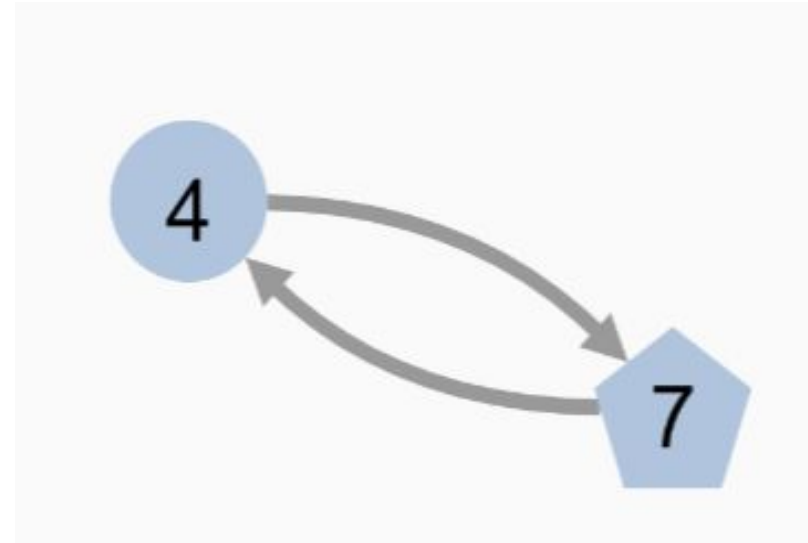
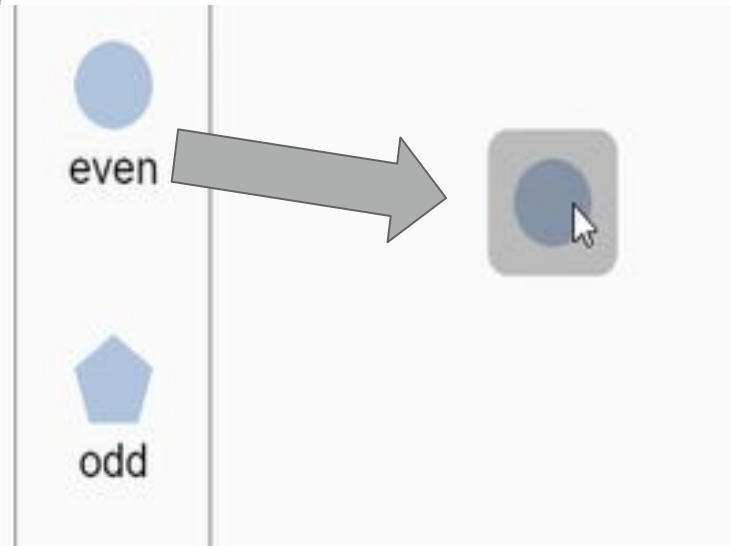


2 CORE FEATURES

(You will also see it in action in the demo later)

CORE FEATURES - ADD A GAME - 1

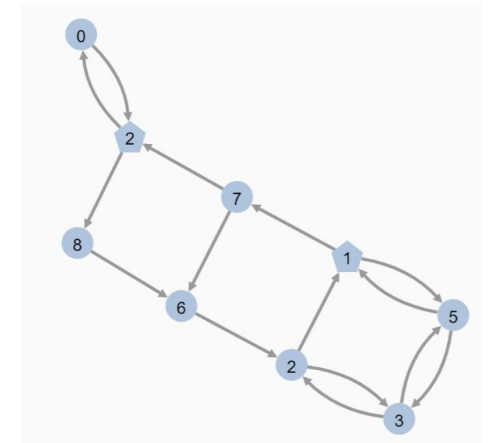
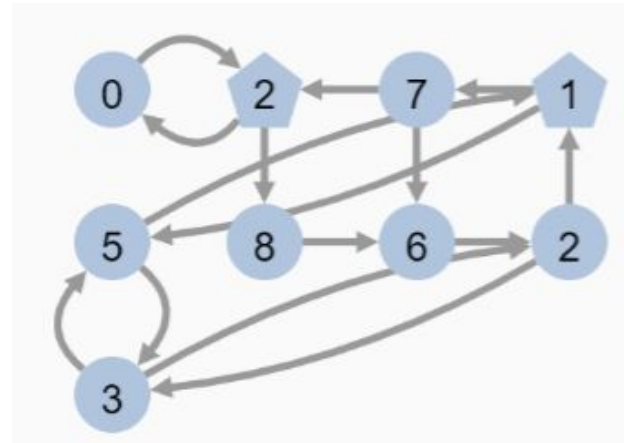
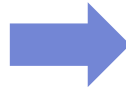
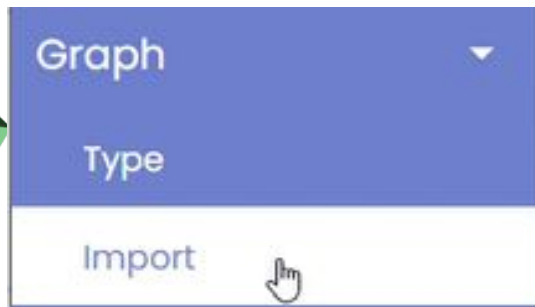
- By drag-and-drop



(You will also see it in action in the demo later)

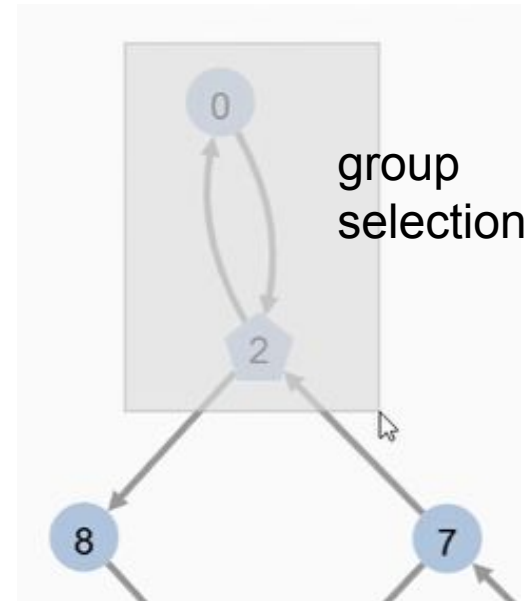
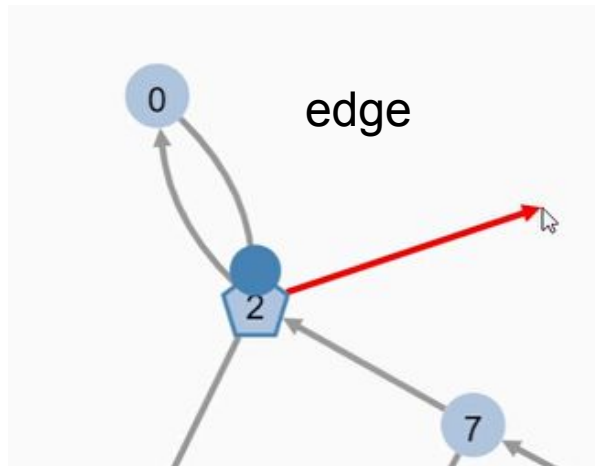
CORE FEATURES - ADD A GAME - 2

- By drag-and-drop
- By import from .pg file



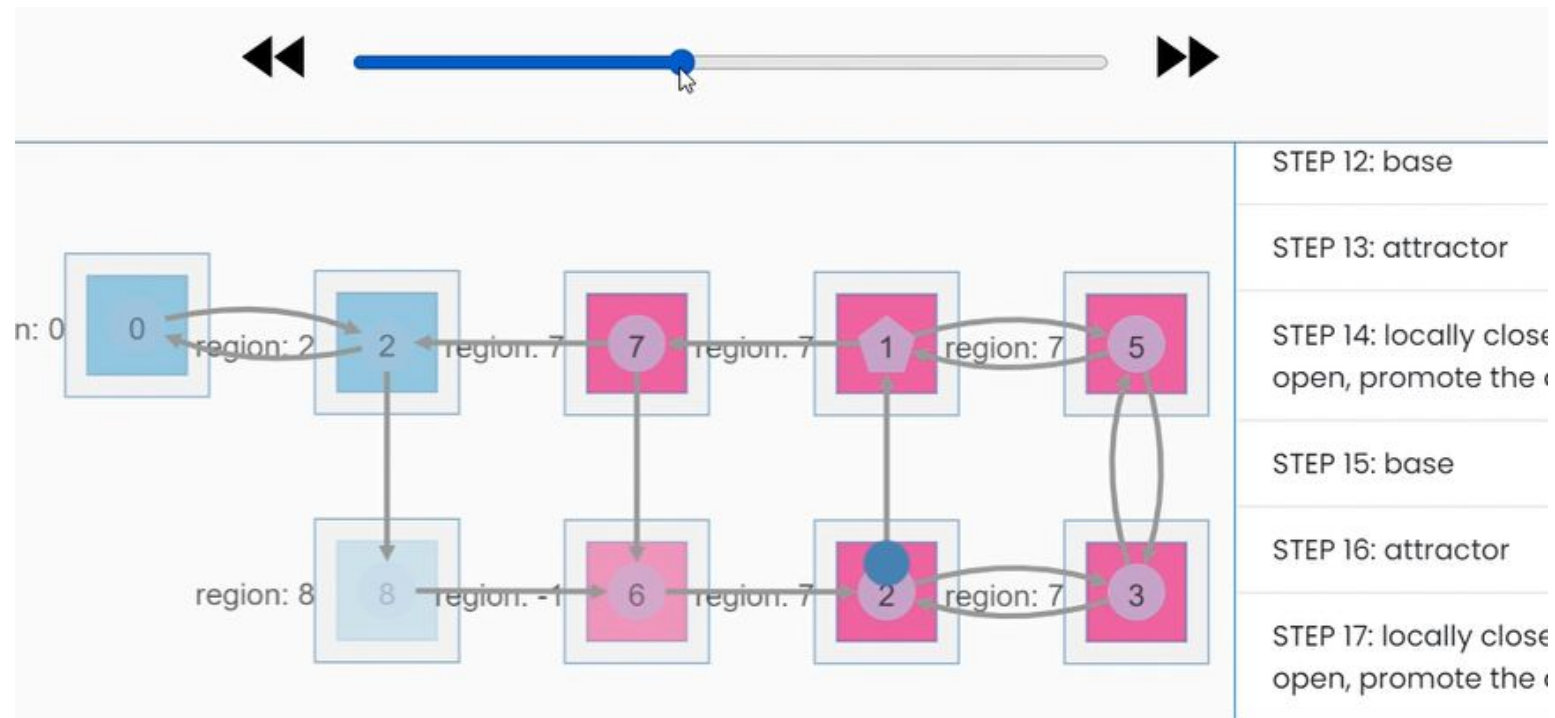
CORE FEATURES - CONFIGURE THE GAME

- Add/remove edge/node
- Change priority(single/group), change owner(single/group)
- Copy/paste



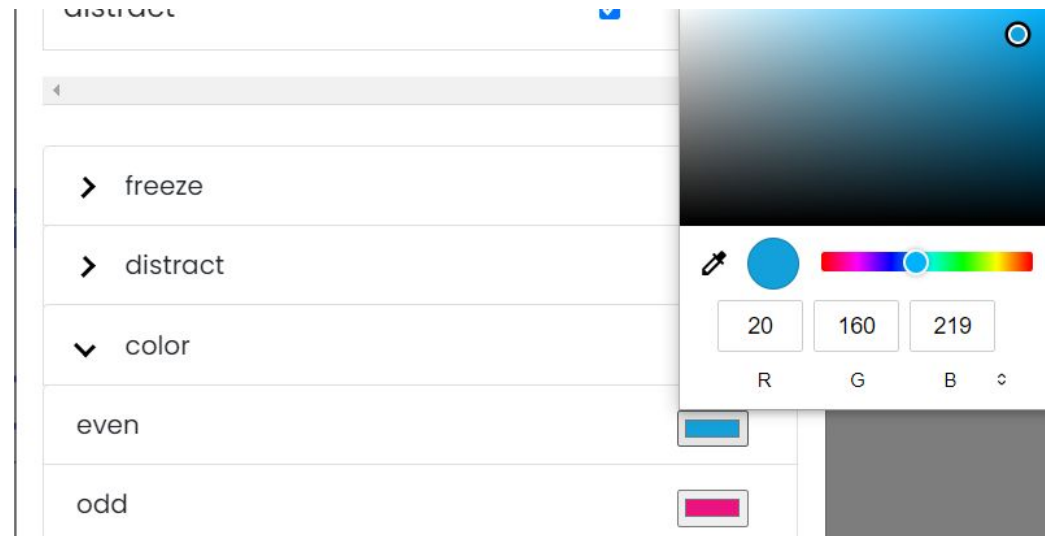
CORE FEATURES - VISUAL AID - 1

- Each step is reflected in the graph, can jump to any step



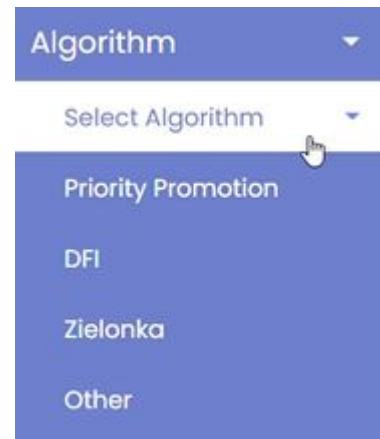
CORE FEATURES - VISUAL AID - 2

- Each step is reflected in the graph, can jump to any step
- Researcher can choose what label to see and how to see it (color/text)




CORE FEATURES - VISUAL AID - 3

- Each step is reflected in the graph, can jump to any step
- Researcher can choose what attribute to see and how to see it (color/text)
- Can add customized algorithms; 3 built-in algorithms to play with



CORE FEATURES - EXPORT

- You want to save interesting configuration of a game.
- After solving a game you can export its solution.



```
test010.pgsol
1 paritysol 10
2 0 0 8
3 1 1
4 2 1 4
5 3 1 9
6 4 1 2
7 5 0 5
8 6 1 2
9 7 1 6
10 8 0
11 9 1 6
```



```
test010.pg
1 parity 10;
2 0 0 0 1,8;
3 1 1 0 7,6;
4 2 2 1 0,4;
5 3 4 1 9,5;
6 4 5 1 2;
7 5 6 0 5,2;
8 6 7 1 0,2;
9 7 9 1 6,3;
10 8 11 1 5;
11 9 12 1 6;
```

CORE FEATURES - SUMMARY

- Add game
- Configure game
- See the steps & attributes
- Export

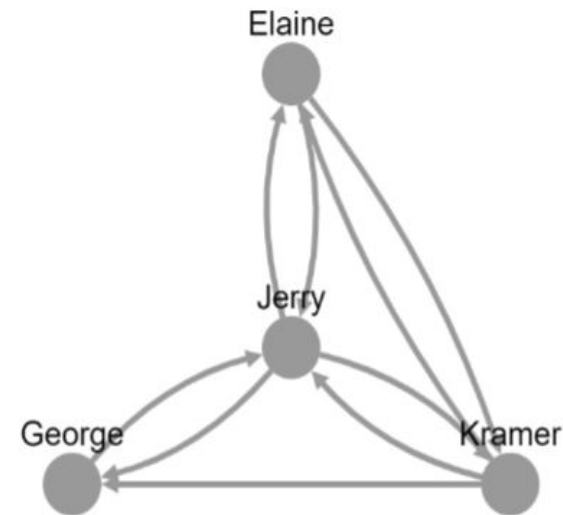
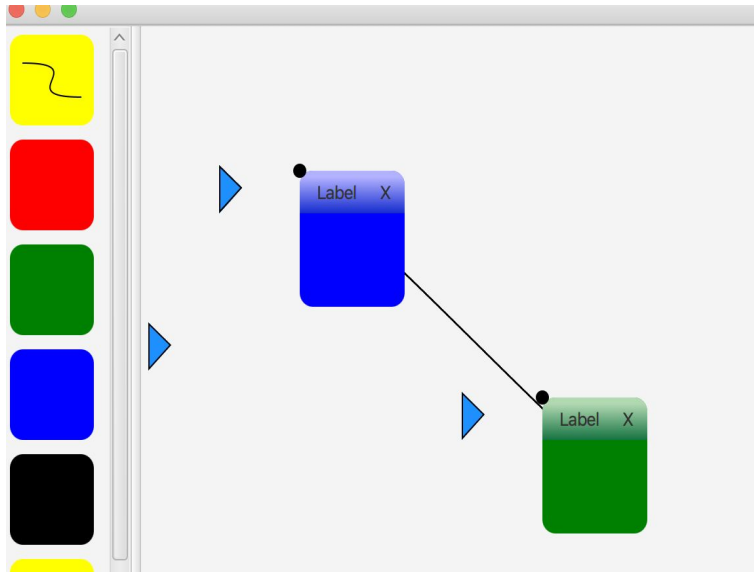




3 DESIGN CHOICES

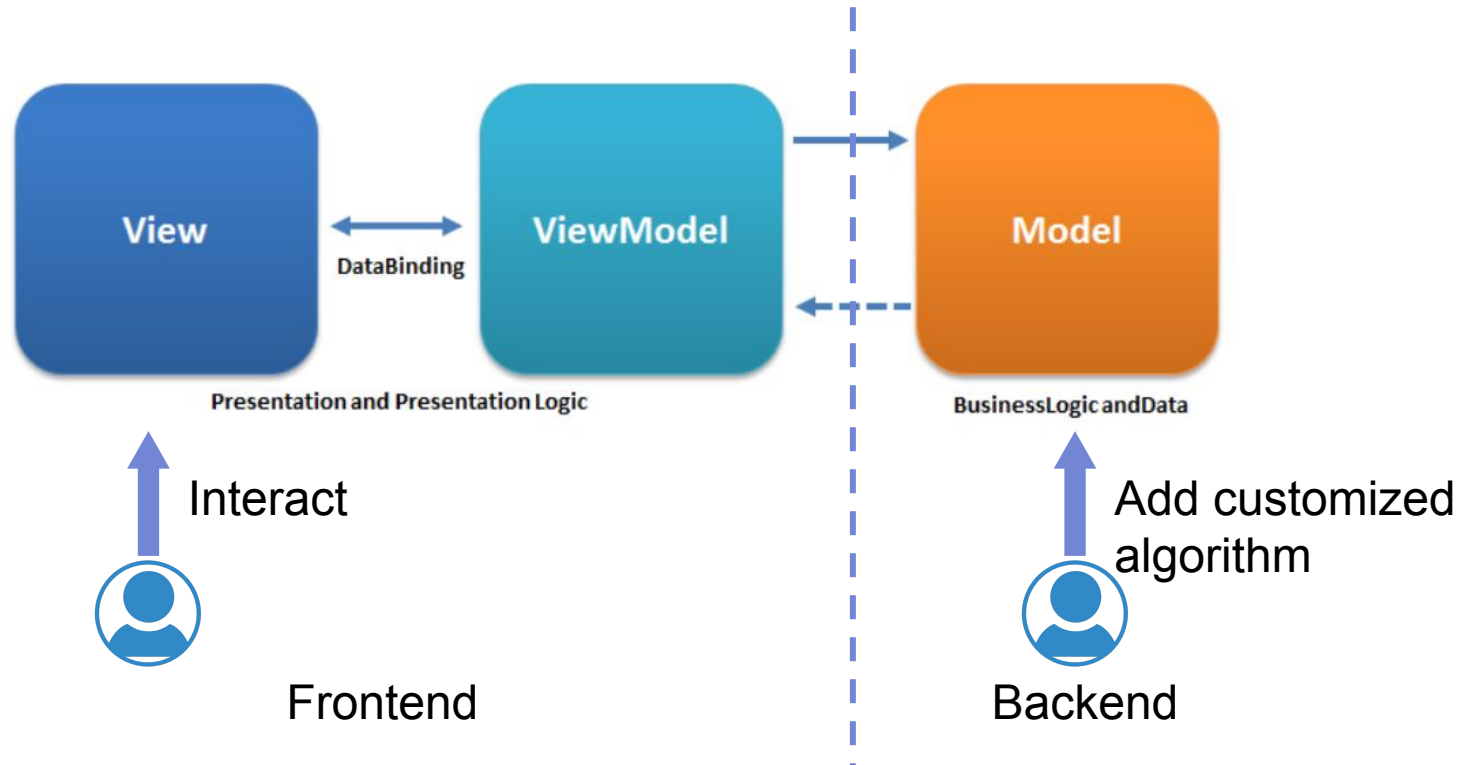
DESIGN CHOICES - TECHNOLOGY?

- JavaFX(desktop app) → cytoscape.js(webapp)



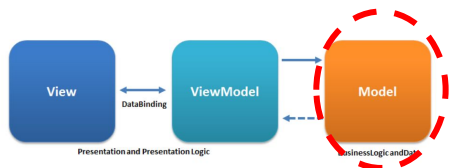
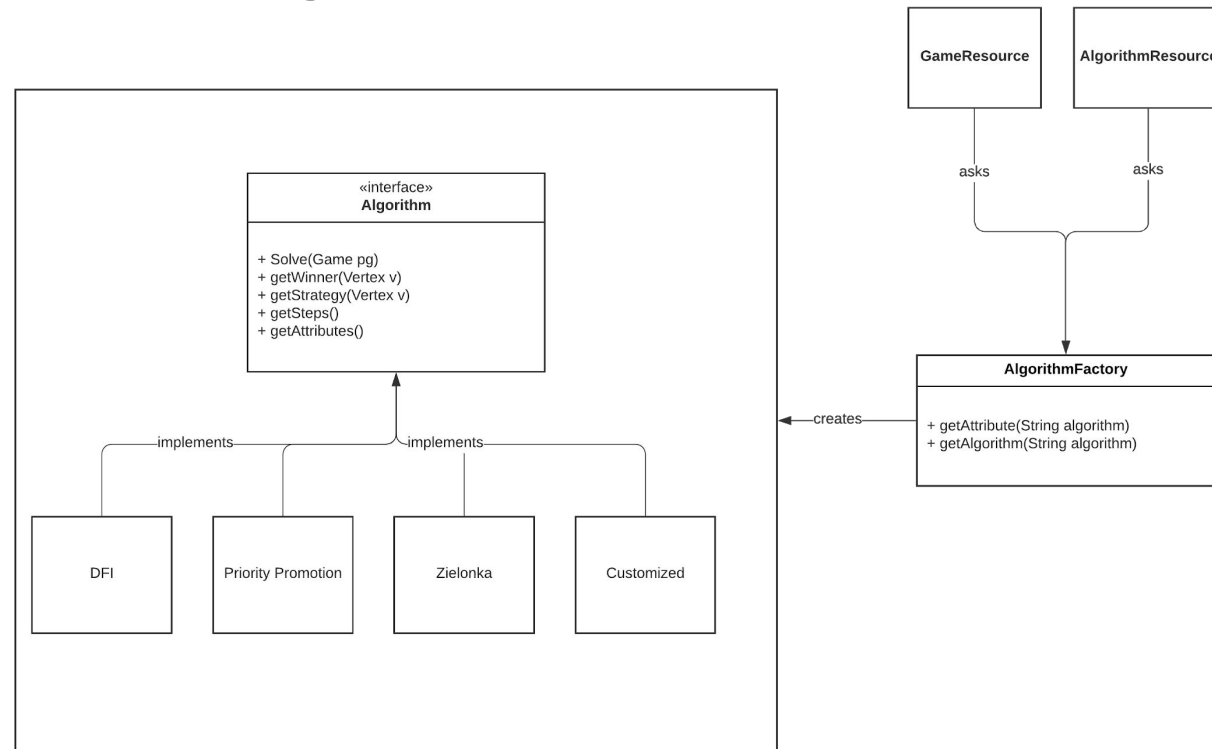
DESIGN CHOICES - ARCHITECTURE?

- JavaFX(desktop app) → cytoscape.js(webapp)



DESIGN CHOICES - FACTORY PATTERN

- Hide the creation logic from the user



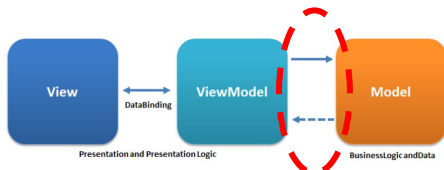
DESIGN CHOICES - PROTOCOL? - 1

- Attributes of a node differs per algorithm, the data returned to frontend need to be able to accommodate unseen attributes.

```
/**  
 * This class tend to save some typing of HashMap<Integer, HashMap<String, String>>.  
 * It's intended to be a generic template to describe the overall status / the update  
 * of the game.  
 */  
public class GameStatus extends HashMap<Integer, HashMap<String, String>>{
```

Node id
Can quickly access
the status of a node

Node status:
attributeName: attributeValue
A node can have any number of attributes
with any name.

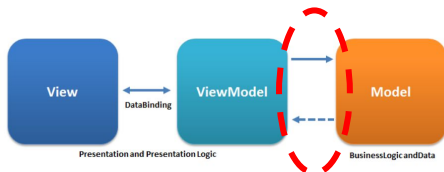


DESIGN CHOICES - PROTOCOL? - 2

- A closer look at the Attribute of each node



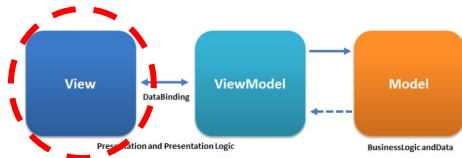
```
public class Attribute {  
    private String name;  
    // type can be either "color" or "text"  
    private AttributeType type;  
    // if type is "color",  
    // need to specify all possible values  
    // if type is "text" then it doesn't matter  
    private Collection<String> values;  
}
```



DESIGN CHOICES - FRONTEND DESIGN - 1

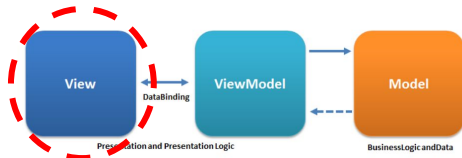
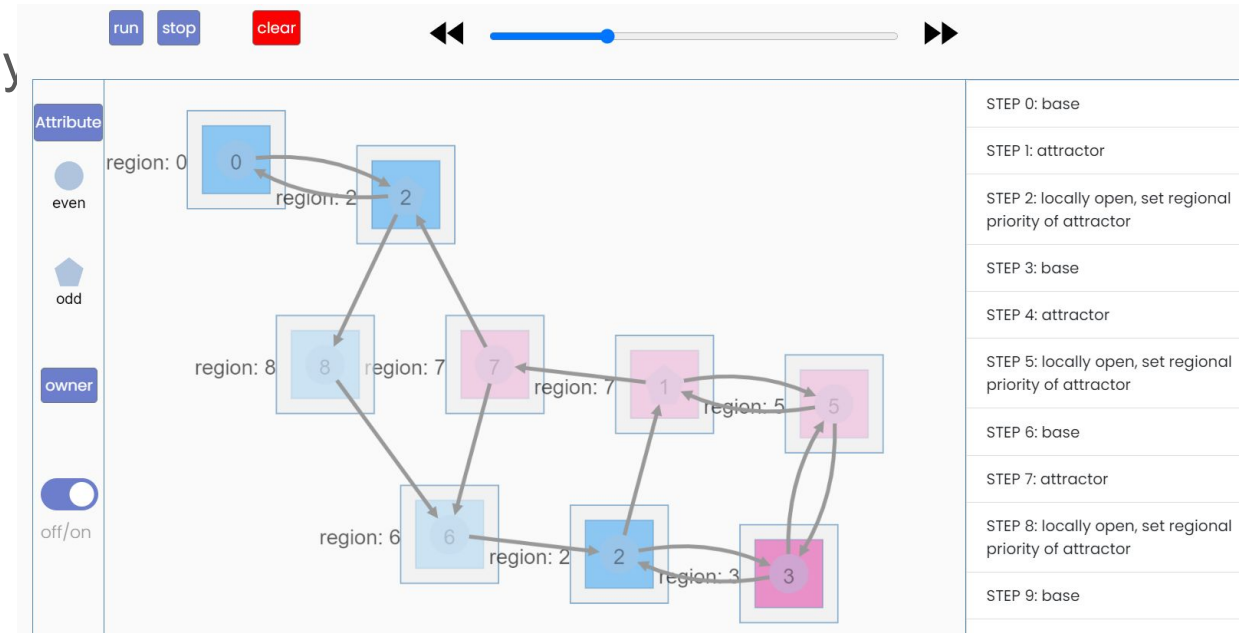
- Aesthetics and minimalist design

Collapsible sidebar:
Low-frequency features



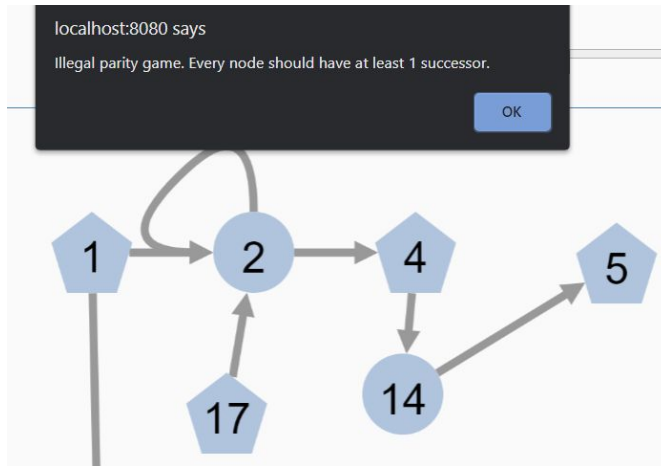
DESIGN CHOICES - FRONTEND DESIGN - 2

- Visibility of system status & user control and freedom
 - Every step is visible (by step message)
 - Can jump to whichever step
 - Can change attribute dynamically
 - High-frequency features at hand
 - run/stop/clear
 - Configure graph



DESIGN CHOICES - FRONTEND DESIGN - 3

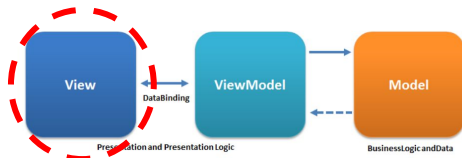
- Error prevention & recovery



If the game is incorrect, the user can't run the algorithm.


A screenshot of a "Select Attributes" dialog box. It features a search bar with the placeholder text "Search for attributes..". Below the search bar, there is a list of attributes: "color" and "region", each with an unchecked checkbox. At the bottom right, there are "Close" and "Apply" buttons.

When an algorithm is selected, nudge the user to configure attributes.



(Jakob Nielsen's 10 Usability Heuristics)

DESIGN CHOICES - SUMMARY

- 
- Technology?
 - Architecture?
 - How to accommodate customized algorithm?
 - Protocol between backend and frontend?
 - Frontend design?
- } ground
- backend
- frontend

4 DEMO

