Easy Data Visualization with Graph

Aja Hammerly

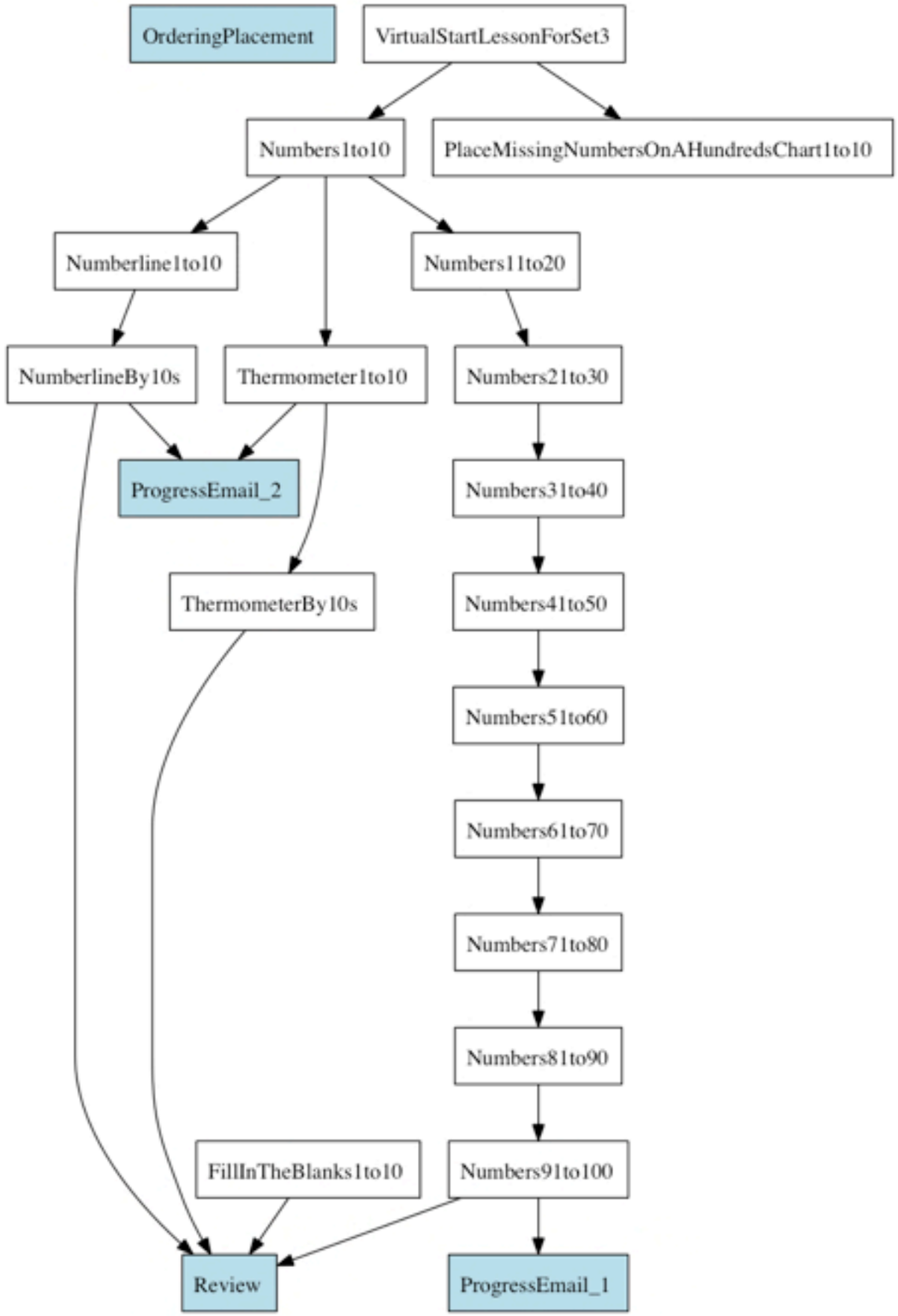
Background

- I work for DreamBox Learning
- We build adaptive educational software for children
- 500 lessons with manually specified dependencies
- Need to find patterns and bugs in that data

Which is easier to comprehend?

```
<topics>
<topic description="Ordering Numbers" id="9" name="OrderingNumbers" standard_id="2"/>
</topics>
<lessons>
<lesson curriculum_id="5" game_set="3" grade_id="13" id="19" layout_x="444" layout_y="117" name="Numbers11to20"
topic_id="9" type="NORMAL"/>
<lesson curriculum_id="5" game_set="3" grade_id="13" id="20" layout_x="444" layout_y="78" name="Numbers1to10"
topic_id="9" type="NORMAL"/>
<lesson curriculum_id="5" game_set="3" grade_id="13" id="21" layout_x="444" layout_y="156" name="Numbers21to30"
topic_id="9" type="NORMAL"/>
<lesson curriculum_id="5" game_set="3" grade_id="13" id="22" layout_x="444" layout_y="194" name="Numbers31to40"
topic_id="9" type="NORMAL"/>
<lesson curriculum_id="5" game_set="3" grade_id="13" id="23" layout_x="445" layout_y="236" name="Numbers41to50"
topic_id="9" type="NORMAL"/>
<lesson curriculum_id="5" game_set="3" grade_id="13" id="24" layout_x="445" layout_y="275" name="Numbers51to60"
topic_id="9" type="NORMAL"/>
<lesson curriculum_id="5" game_set="3" grade_id="13" id="25" layout_x="445" layout_y="314" name="Numbers61to70"
topic_id="9" type="NORMAL"/>
<lesson curriculum_id="5" game_set="3" grade_id="13" id="26" layout_x="445" layout_y="353" name="Numbers71to80"
topic_id="9" type="NORMAL"/>
<lesson curriculum_id="5" game_set="3" grade_id="13" id="27" layout_x="445" layout_y="392" name="Numbers81to90"
topic_id="9" type="NORMAL"/>
<lesson curriculum_id="5" game_set="3" grade_id="13" id="28" layout_x="445" layout_y="431" name="Numbers91to100"
topic_id="9" type="NORMAL"/>
<lesson curriculum_id="5" game_set="3" grade_id="13" id="38" layout_x="287" layout_y="117" name="Numberline1to10"
topic_id="9" type="NORMAL"/>
<lesson curriculum_id="5" game_set="3" grade_id="13" id="39" layout_x="287" layout_y="156" name="NumberlineBy10s"
topic_id="9" type="NORMAL"/>
<lesson curriculum_id="5" game_set="3" grade_id="13" id="50" layout_x="104" layout_y="116"
name="VerticalNumberline1to10" topic_id="9" type="NORMAL"/>
<lesson curriculum_id="5" game_set="3" grade_id="13" id="51" layout_x="105" layout_y="155"
name="VerticalNumberlineBy10s" topic_id="9"type="NORMAL"/>
<lesson curriculum_id="5" game_set="3" grade_id="13" id="197" layout_x="671" layout_y="76"
name="MissingNumbers1to10" topic_id="9" type="NORMAL"/>
</lessons>
<mappings>
<mapping assess_max="0" assess_min="0" id="48" lesson_id="19" mo_id="570" problem_type="" question_type=""
req_min="80"/>
<mapping assess_max="100" assess_min="0" id="49" lesson_id="19" mo_id="572" problem_type="Numbers"
question_type="Decade" req_min="0"/>
<mapping assess_max="100" assess_min="0" id="119" lesson_id="20" mo_id="570" problem_type="Numbers"
question_type="Decade" req_min="0"/>
<mapping assess_max="0" assess_min="0" id="960" lesson_id="20" mo_id="631" problem_type="" question_type=""
```

Or This?


OrderingPlacement
VirtualStartLessonForSet3
Numbers1to10
PlaceMissingNumbersOnAHundredsChart1to10
Numberline1to10
Numbers11to20
NumberlineBy10s
Thermometer1to10
Numbers21to30
ProgressEmail_2
Numbers31to40
ThermometerBy10s
Numbers41to50
Numbers51to60
Numbers61to70
Numbers71to80
Numbers81to90
FillInTheBlanks1to10
Numbers91to100
Review
ProgressEmail_1

Making pictures by hand is easy

But it doesn't scale

- Time consuming
- Underlying data changes frequently
- Different people want different views

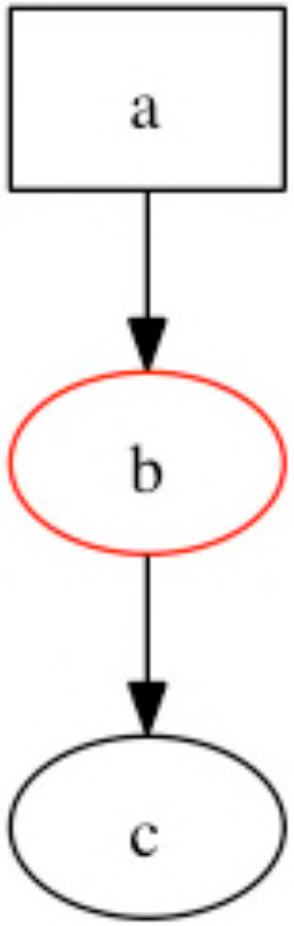
graphviz to the rescue

DOT

- Simple language to describe graphs
- Graphs are nodes and edges
- Can edit attributes such as color and shape

Example

```
digraph example {
  a -> b;
  b -> c;
  a[shape=box]
  b[color=red]
}
```



Viewing DOT files

- GraphViz
- Tulip

But let's use Ruby

```
sudo gem install graph
```

A simple graph

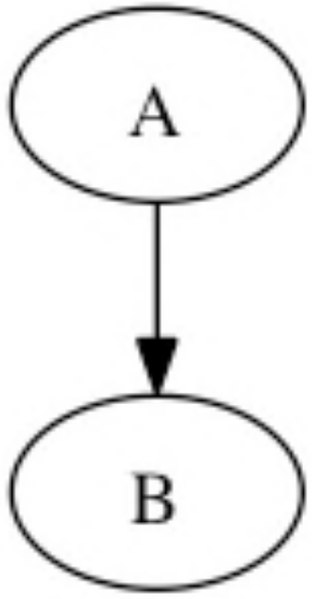
```
digraph do
  node("B")
end
```

Nodes with Labels

```
digraph do
  node("B").label "Hello"
end
```

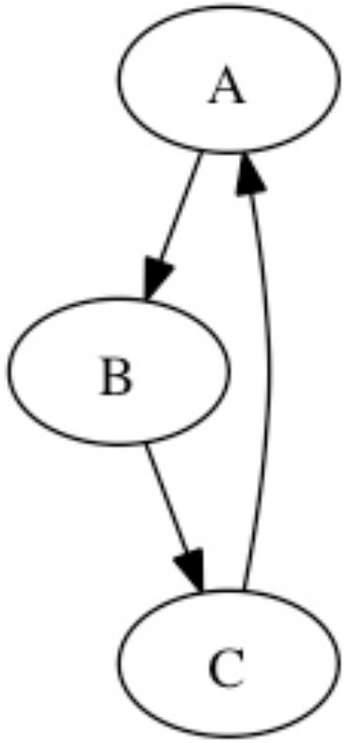
Adding Edges

```
digraph do
  edge "A", "B"
end
```



Saving

```
digraph do
  edge "A", "B"
  edge "B", "C"
  edge "C", "A"
  save "cycle"
end
```

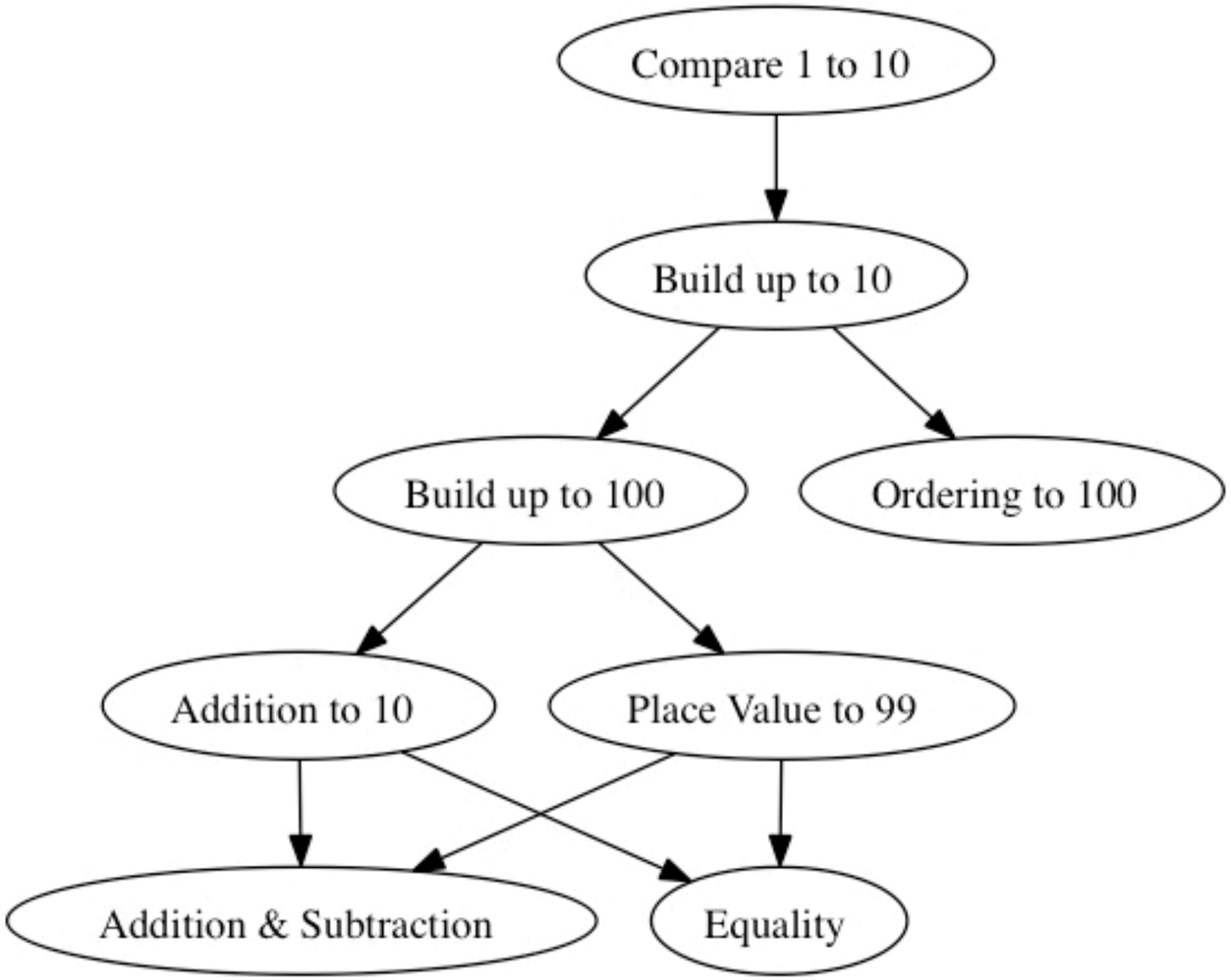


Exporting

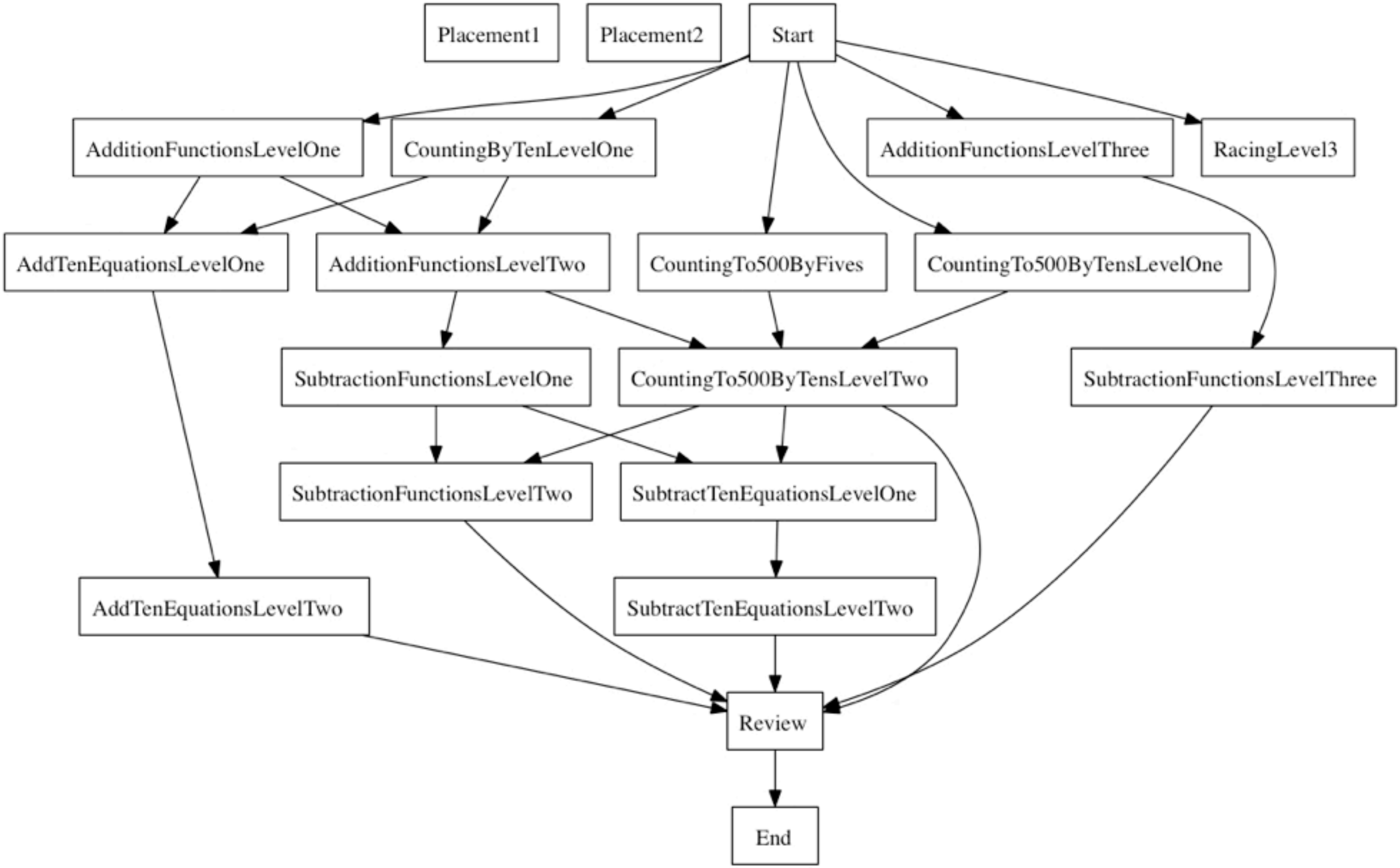
```
digraph do
  edge "a", "b"
  save "example", "png"
  save "example", "jpg"
end
```

Format list: http://www.graphviz.org/doc/info/output.html

Now you can build this


Compare 1 to 10
Build up to 10
Build up to 100
Ordering to 100
Addition to 10
Place Value to 99
Addition & Subtraction
Equality

Or This

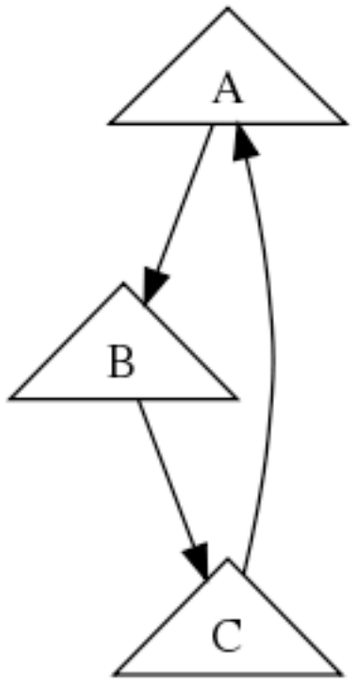

Placement1
Placement2
Start
AdditionFunctionsLevelOne
CountingByTenLevelOne
AdditionFunctionsLevelThree
RacingLevel3
AddTenEquationsLevelOne
AdditionFunctionsLevelTwo
CountingTo500ByFives
CountingTo500ByTensLevelOne
SubtractionFunctionsLevelOne
CountingTo500ByTensLevelTwo
SubtractionFunctionsLevelThree
SubtractionFunctionsLevelTwo
SubtractTenEquationsLevelOne
AddTenEquationsLevelTwo
SubtractTenEquationsLevelTwo
Review
End

But that's boring

Shapes

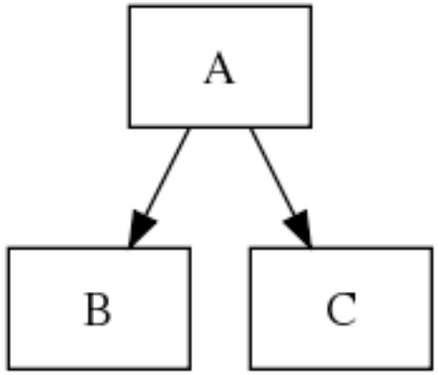
Shapes

```
digraph do
  node_attribs << triangle
  edge "A", "B"
  edge "B", "C"
  edge "C", "A"
end
```



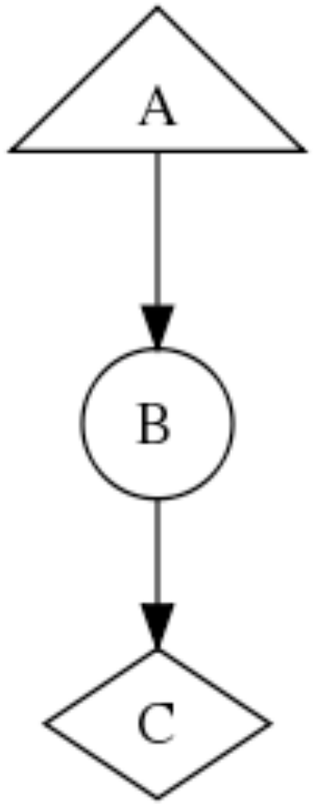
Boxes are Special

```
digraph do
  boxes
  edge "A", "B"
  edge "A", "C"
end
```



Many Shapes

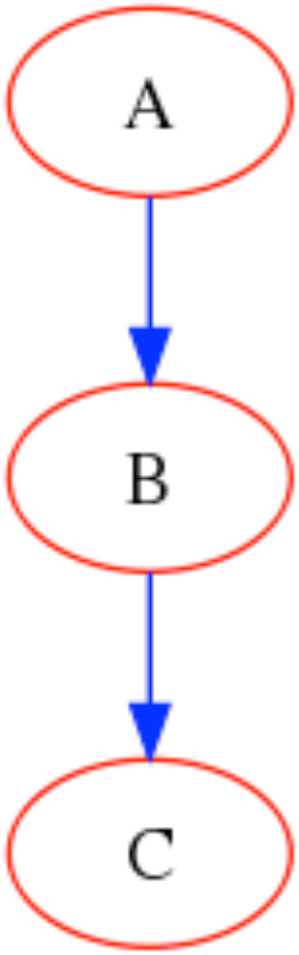
```
digraph do
  edge "A", "B", "C"
  triangle << node("A")
  circle   << node("B")
  diamond  << node("C")
end
```



Color

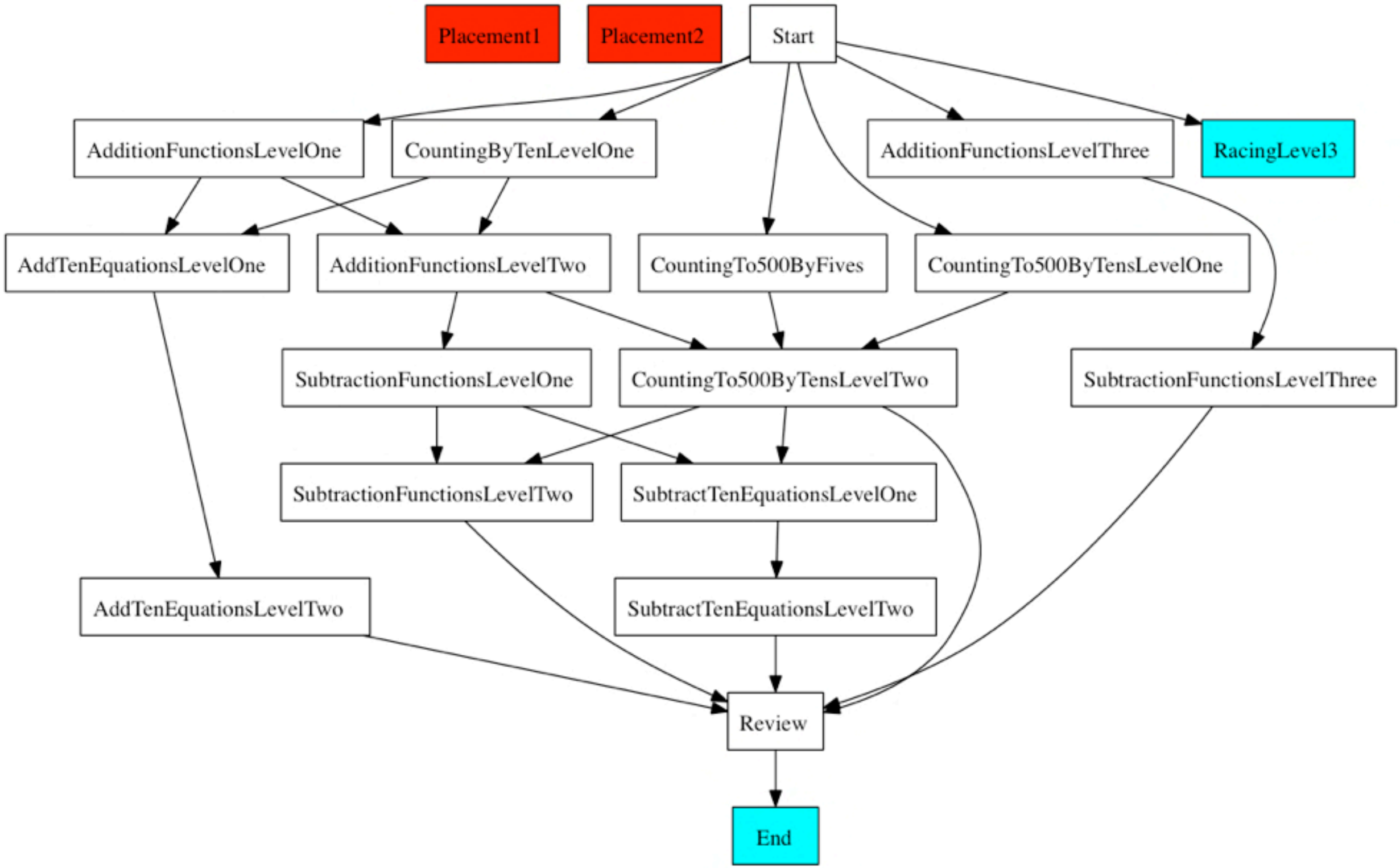
One Color for All

```
digraph do
  node_attribs << red
  edge_attribs << blue
  edge "A", "B", "C"
end
```



Many Colors

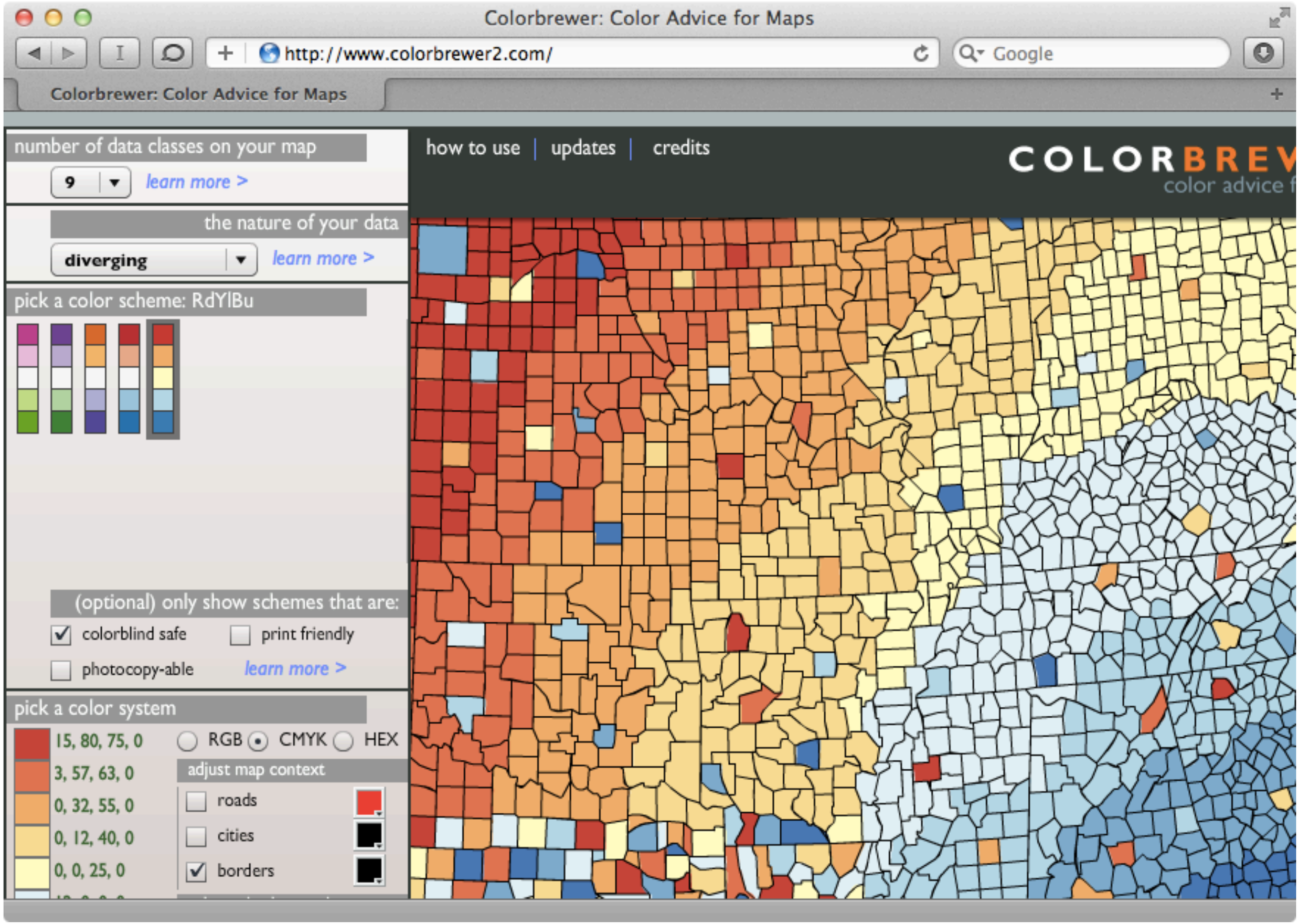
```
digraph do
  node_attribs << filled
  edge "G", "O", "R", "P"
  green  << node("G")
  orange << node("O")
  red    << node("R")
  purple << node("P")
end
```


Placement1
Placement2
Start
AdditionFunctionsLevelOne
CountingByTenLevelOne
AdditionFunctionsLevelThree
RacingLevel3
AddTenEquationsLevelOne
AdditionFunctionsLevelTwo
CountingTo500ByFives
CountingTo500ByTensLevelOne
SubtractionFunctionsLevelOne
CountingTo500ByTensLevelTwo
SubtractionFunctionsLevelThree
SubtractionFunctionsLevelTwo
SubtractTenEquationsLevelOne
AddTenEquationsLevelTwo
SubtractTenEquationsLevelTwo
Review
End

Help for the design impaired

Color Schemes

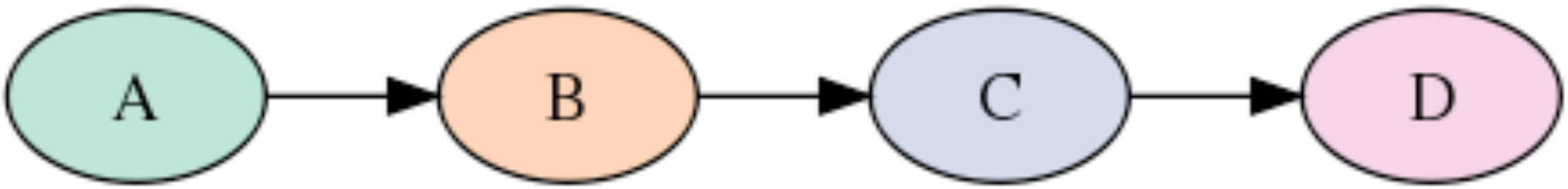
- Uses Brewer Color Schemes
  - http://www.graphviz.org/doc/info/colors.html
- Preview schemes here
  - http://colorbrewer2.com

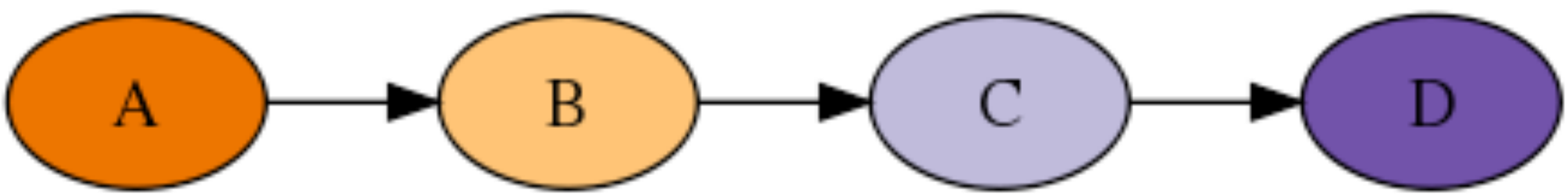

Colorbrewer: Color Advice for Maps
http://www.colorbrewer2.com/
Google
Colorbrewer: Color Advice for Maps
number of data classes on your map
9
learn more >
how to use | updates | credits
the nature of your data
diverging
learn more >
pick a color scheme: RdYlBu
(optional) only show schemes that are:
colorblind safe
print friendly
photocopy-able
learn more >
pick a color system
15, 80, 75, 0
3, 57, 63, 0
0, 32, 55, 0
0, 12, 40, 0
0, 0, 25, 0
RGB
CMYK
HEX
adjust map context
roads
cities
borders

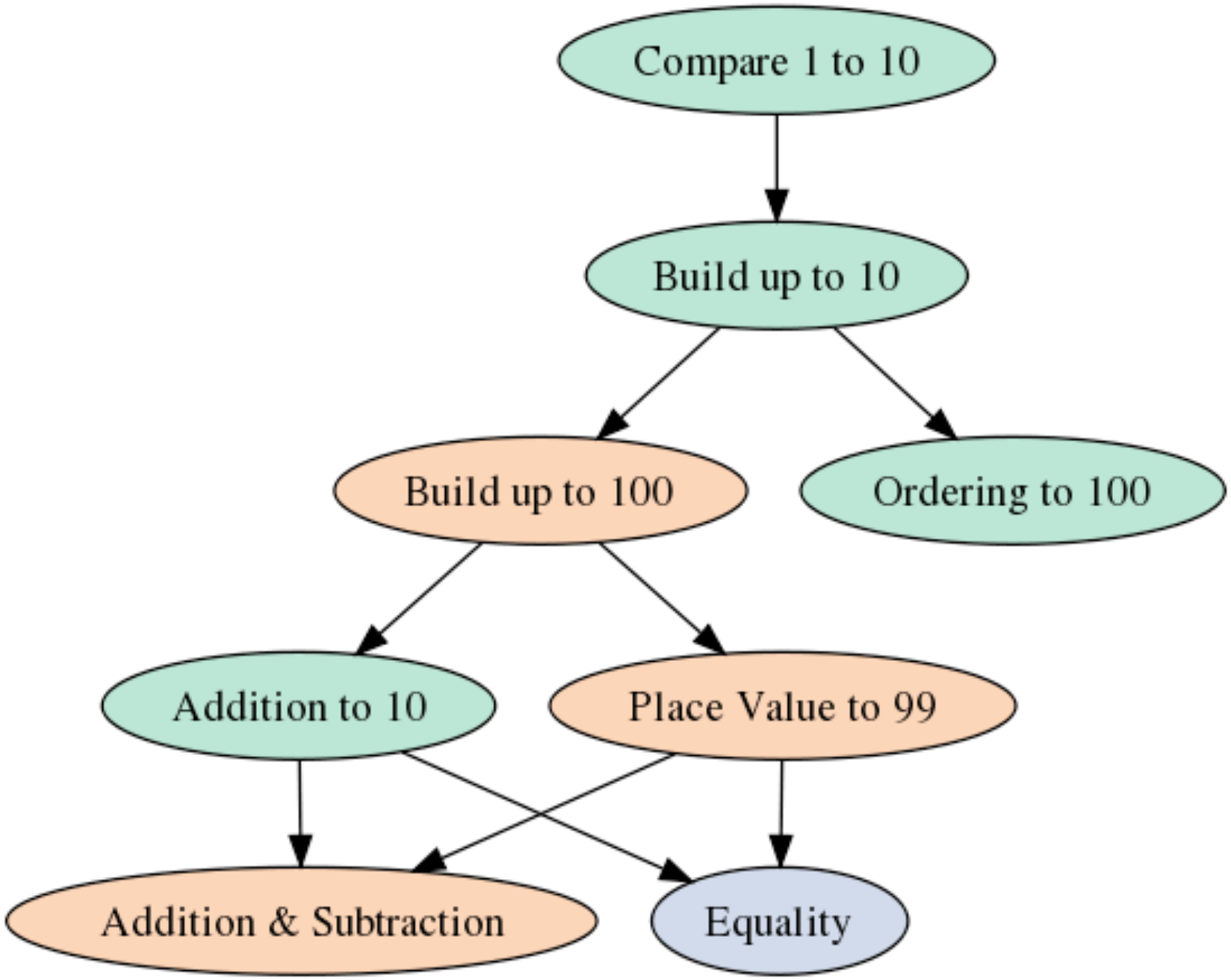
Color Scheme Example

```
digraph do
  node_attribs << filled
  colorscheme(:set1, 4)
  c1 << node("A")
  c2 << node("B")
  c3 << node("C")
  c4 << node("D")
  edge "A", "B", "C", "D"
end
```

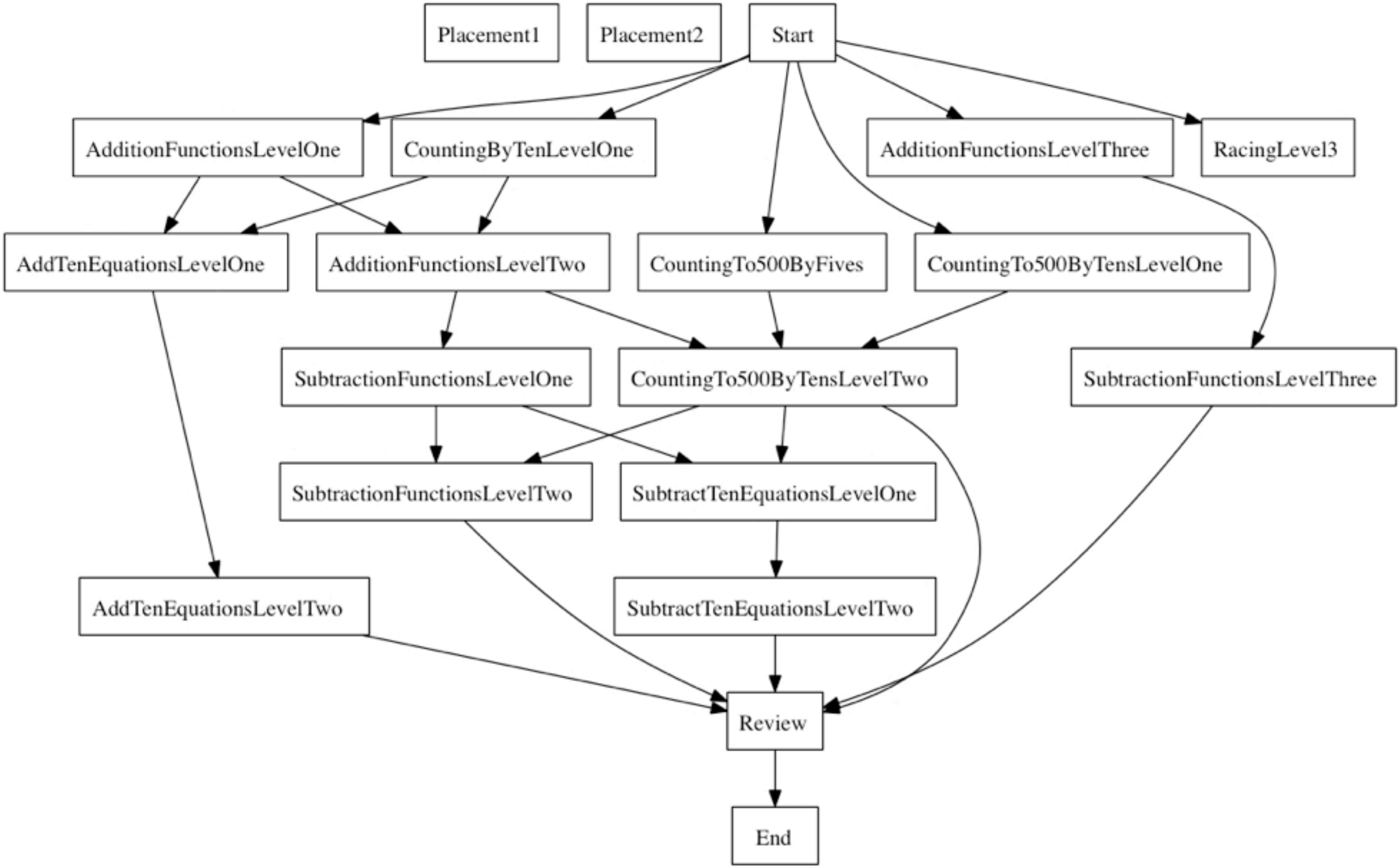

A
B
C
D


A
B
C
D

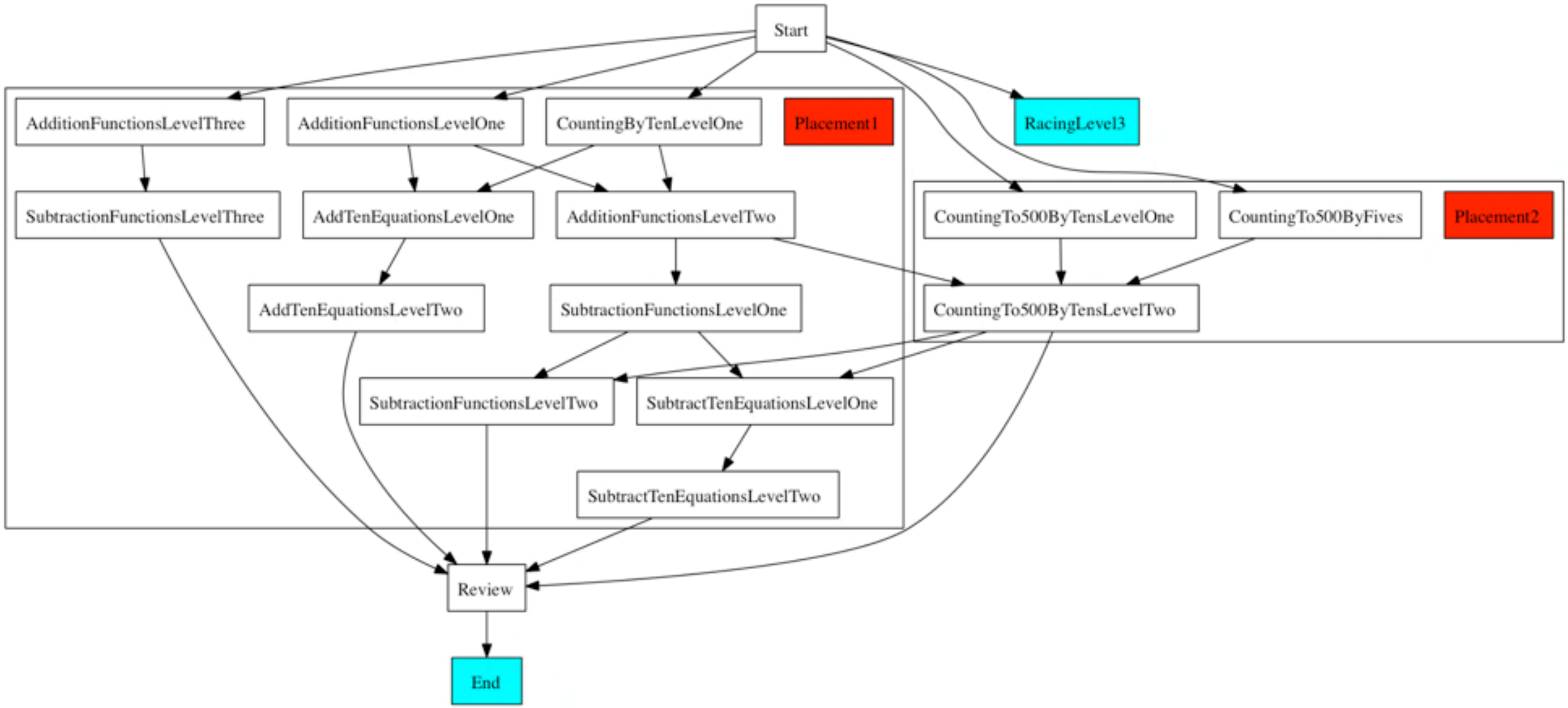

A
B
C
D


Compare 1 to 10
Build up to 10
Build up to 100
Ordering to 100
Addition to 10
Place Value to 99
Addition & Subtraction
Equality

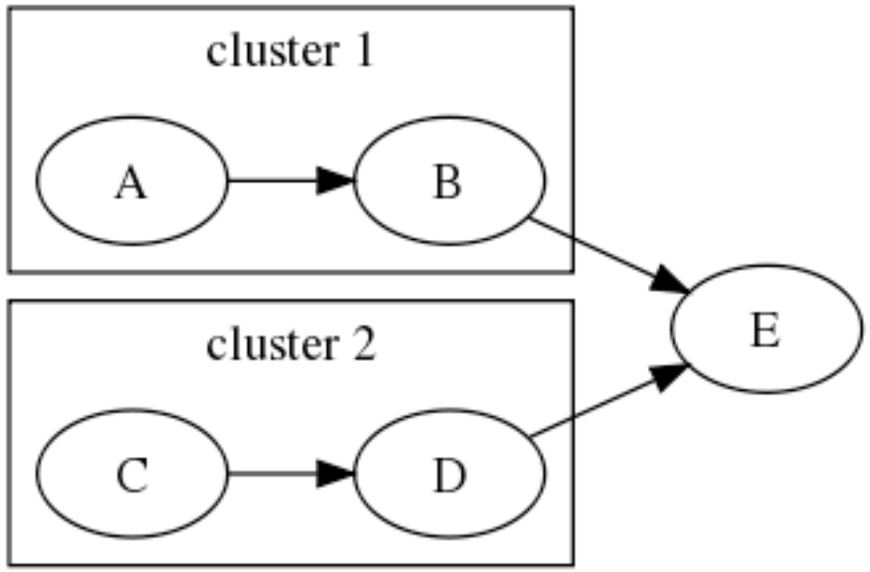
Clustering


Placement1
Placement2
Start
AdditionFunctionsLevelOne
CountingByTenLevelOne
AdditionFunctionsLevelThree
RacingLevel3
AddTenEquationsLevelOne
AdditionFunctionsLevelTwo
CountingTo500ByFives
CountingTo500ByTensLevelOne
SubtractionFunctionsLevelOne
CountingTo500ByTensLevelTwo
SubtractionFunctionsLevelThree
SubtractionFunctionsLevelTwo
SubtractTenEquationsLevelOne
AddTenEquationsLevelTwo
SubtractTenEquationsLevelTwo
Review
End

Into This


Start
AdditionFunctionsLevelThree
AdditionFunctionsLevelOne
CountingByTenLevelOne
Placement1
RacingLevel3
SubtractionFunctionsLevelThree
AddTenEquationsLevelOne
AdditionFunctionsLevelTwo
CountingTo500ByTensLevelOne
CountingTo500ByFives
Placement2
AddTenEquationsLevelTwo
SubtractionFunctionsLevelOne
CountingTo500ByTensLevelTwo
SubtractionFunctionsLevelTwo
SubtractTenEquationsLevelOne
SubtractTenEquationsLevelTwo
Review
End

```
digraph do
  cluster "1" do
    label "cluster 1"
    edge "A", "B"
  end
  cluster "2" do
    label "cluster 2"
    edge "C", "D"
  end
  edge "B", "E"
  edge "D", "E"
end
```



Building from data

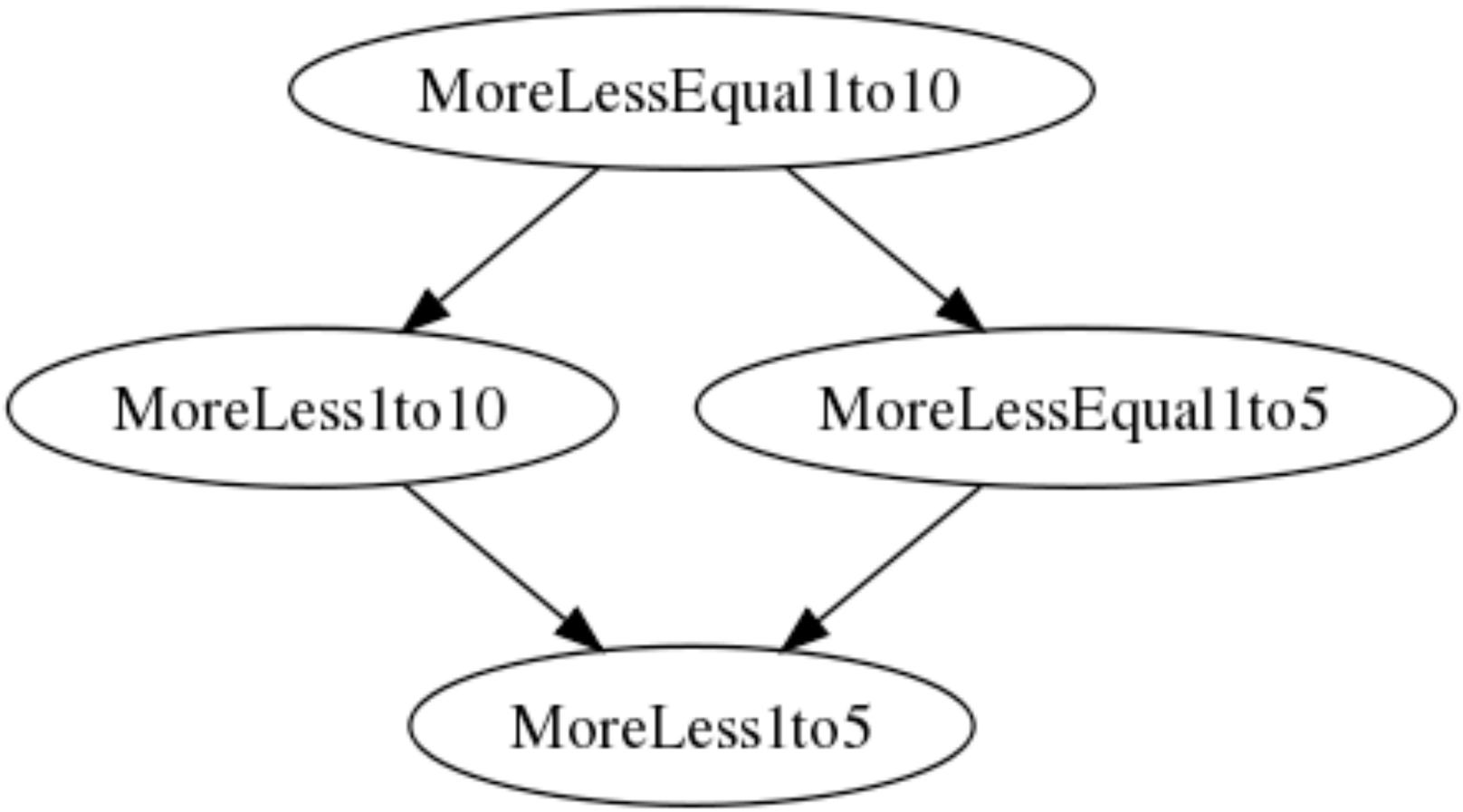
Data

```
<lessons>
  <lesson id="1" name="MoreLess1to5"/>
  <lesson id="2" name="MoreLess1to10"/>
  <lesson id="3" name="MoreLessEqual1to5"/>
  <lesson id="4" name="MoreLessEqual1to10"/>
  <sequence lesson_id="2" pre_req="1"/>
  <sequence lesson_id="3" pre_req="1"/>
  <sequence lesson_id="4" pre_req="2"/>
  <sequence lesson_id="4" pre_req="3"/>
</lessons>
```

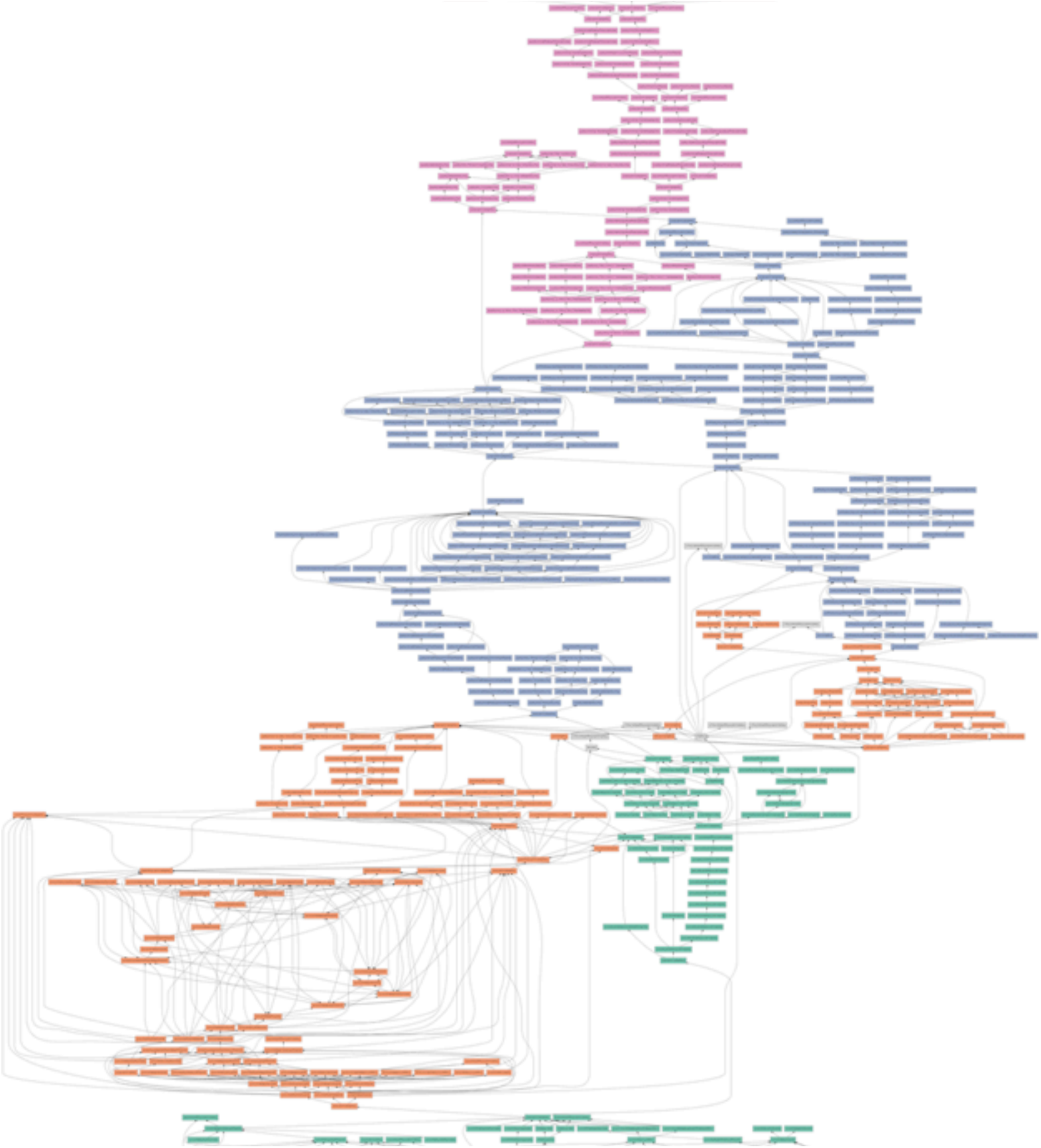
Extract Data

```
File.open("sample.xml") do |f|
  doc       = Nokogiri::XML(f)
  lessons   = doc.xpath("//lesson")
  sequences = doc.xpath("//sequence")

  draw_graph(lessons, sequences)
end
```

```
def draw_graph(lessons, sequences)
  digraph do
    lessons.each do |l|
      node(l["id"]).label l["name"]
    end

    sequences.each do |s|
      edge s["pre_req"],s["lesson_id"]
    end
  end
end
```


MoreLessEqual1to10
MoreLess1to10
MoreLessEqual1to5
MoreLess1to5

On a Larger Scale



Automated Updates

Hudson/Jenkins

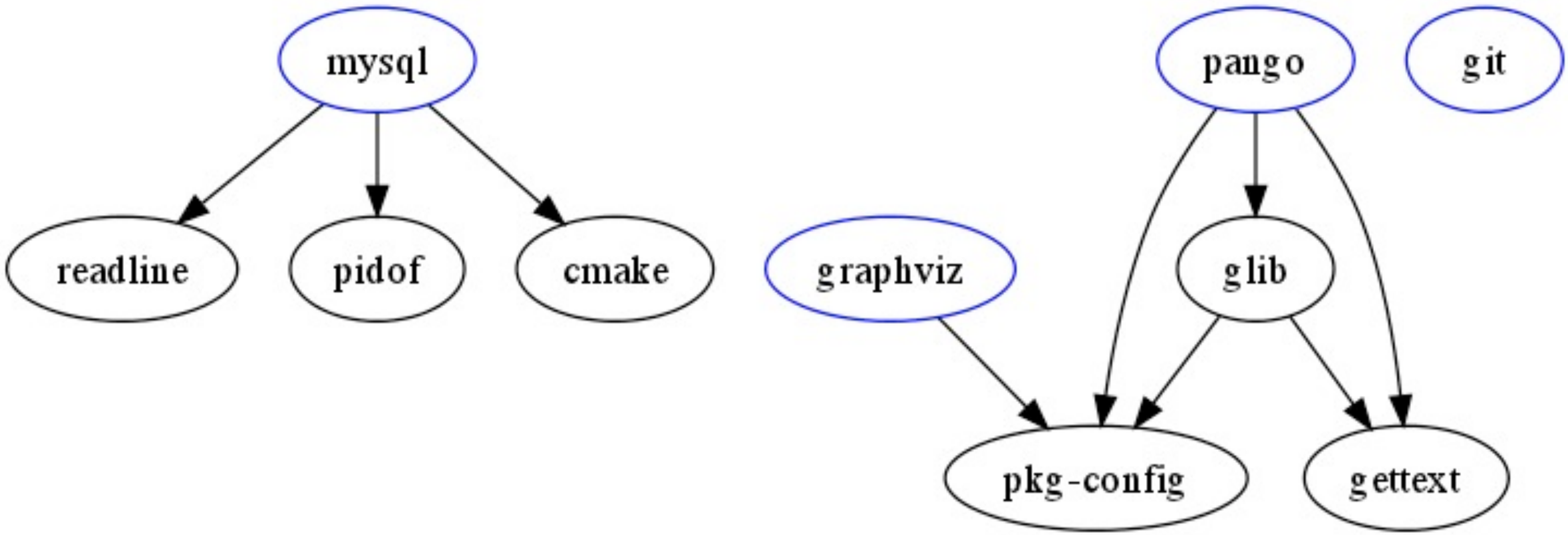
- When data changes:
  - Automatically regenerates the graphs
  - Copies graphs to the internal network
  - Sends mail

More Fun with Graph

Visualize Dependencies

Example

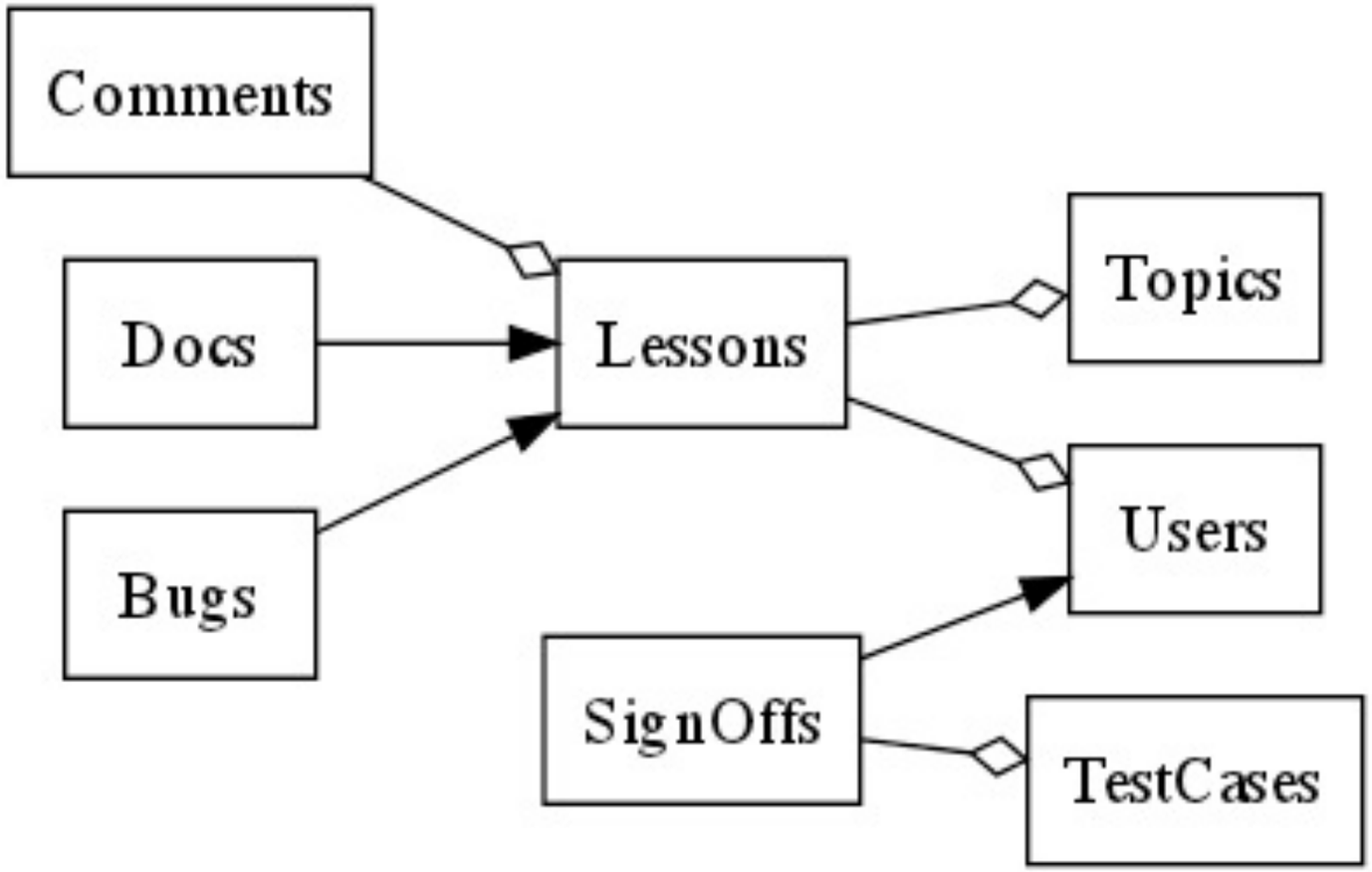
```
$ graph homebrew
```


mysql
readline
pidof
cmake
pango
git
graphviz
glib
pkg-config
gettext

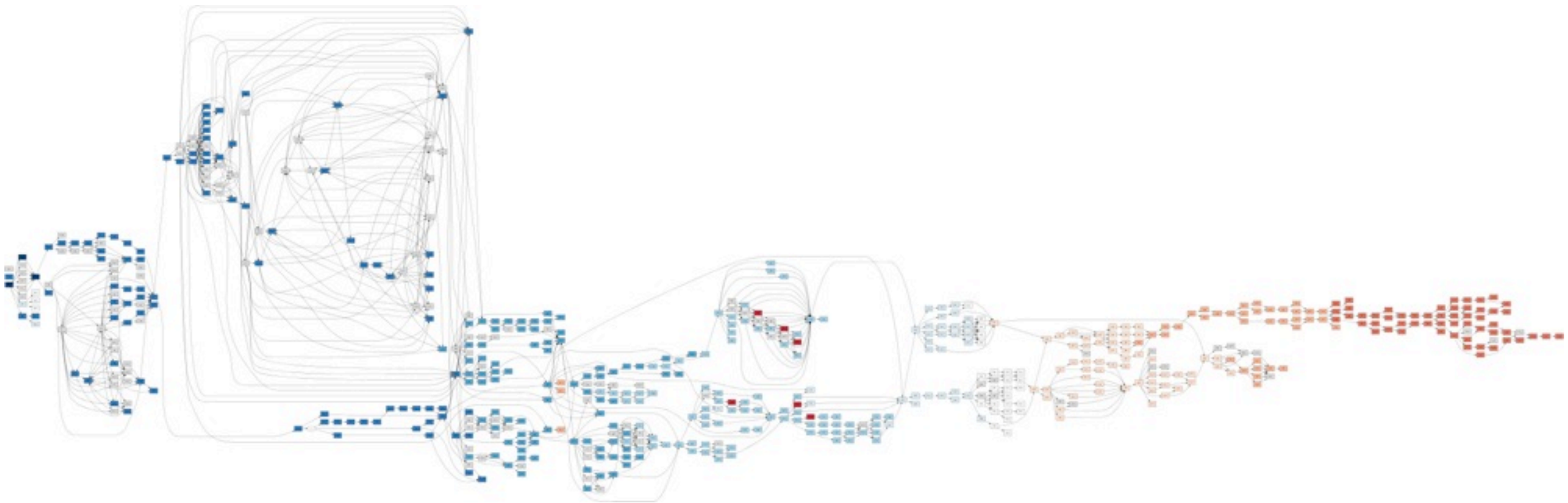
Analyzers

- RubyGems
- Homebrew
- FreeBSD Ports
- MacPorts

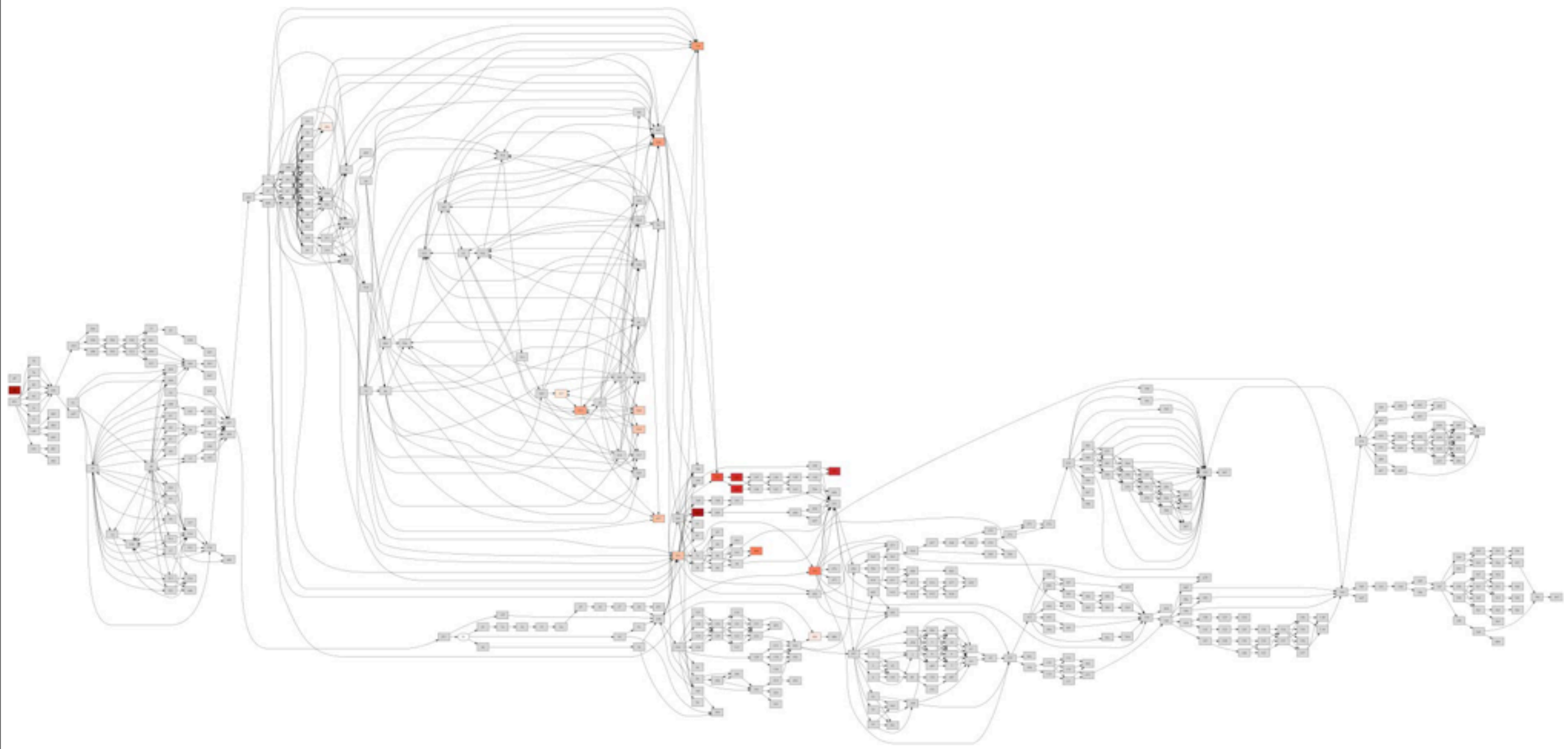
Dynamic Schema Diagrams


Comments
Docs
Bugs
Lessons
Topics
Users
SignOffs
TestCases

Illustrate History



Animation



Thank You

- Ryan Davis for graph
- Aaron Patterson for Nokogiri

Thank You