Graph Coloring RET 2018

Physics

Presented by

Becky Humbarger, Riley H.S.

- rhumbarger@sbcsc.k12.in.us
- Riley High School
- 1902 S. Fellows St.
- South Bend, IN 46613
- 574-393-5100
- 574-876-4328 cell

Erica Price
Trinity School at Greenlawn

eprice@trinityschools.org
Home: 574-855-3867
Cell: 574-607-0108

School: 574-287-5590
107 S. Greenlawn Ave
South Bend, IN 46617

Courses and Impact

Becky Humbarger, Riley H.S.

- Physics and ICP (Integrated chemistry and physics)
- plans to implement coding in a Physics motion unit.
- plans to teach about algorithms with ICP classes.

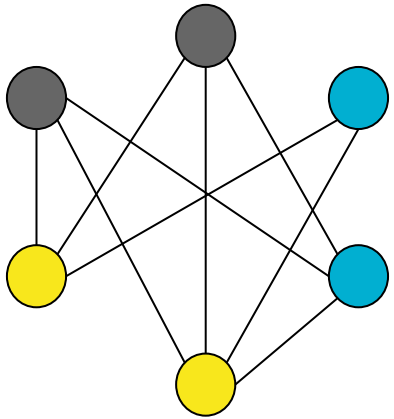
Erica Price teaches:

- Chemistry, Physics, Computer Programming, Algebra, and 7/8 Science
- Main module: 8th grade Science - learning/doing Graph Coloring and python coding

Graph Coloring

Vertex Coloring of Graphs

Used for such problems as:



Scheduling

Pattern matching

Fault diagnosis

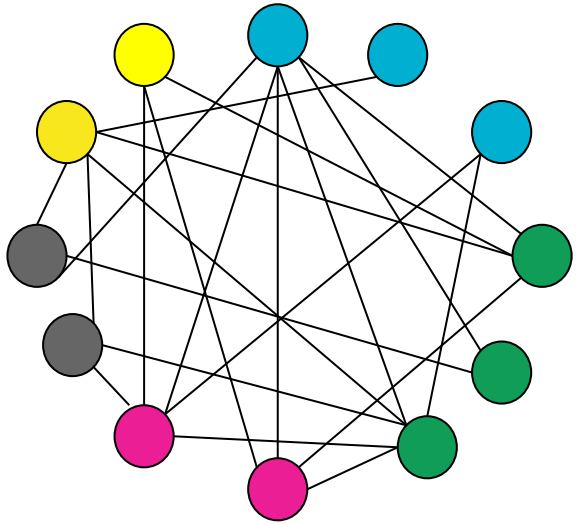
Resource Allocation

“Conflicting” Nodes cannot be the same color

NP-Hard Problem

As number of nodes increases, time to solve problem increases exponentially.

NP-hard:
Non-deterministic
polynomial-time hard



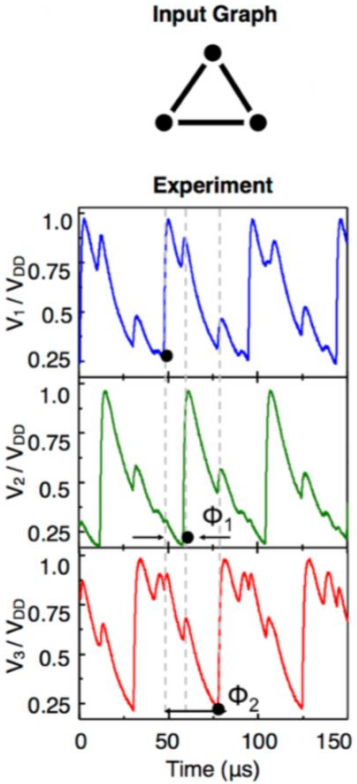
Natural Processes

- Von Neumann architecture has limits
- Explore solving the problem in massively parallel manner
- Natural processes provide many options
  - Life science
  - Physics
  - Earth science

Coupled Oscillatory Networks

Coupled Oscillatory networks

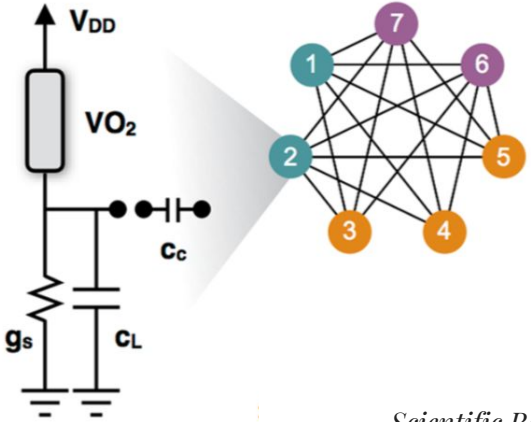
- Phase and frequency dynamics of circuit can be exploited to encode system states
- Computational kernels may mimic the spiking networks of the human brain
- Have been used successfully in associative computing with significant improvements in energy-efficiency and performance



Scientific Reports **volume 7**, Article number: 911 (2017)

Vanadium Dioxide Oscillators

- coupled relaxation oscillators in series circuit
- Vanadium dioxide metal-insulator-transition devices
- Phase dynamics of oscillators in circuit translates to graph coloring



Scientific Reports **volume 7**, Article number: 911 (2017)

Brelaz Heuristic for Graph Coloring

Traditional methods for solving problem

Decisions are made in the process of graph coloring. Those choices influence the outcome and the processing time and energy.

Heuristic: Not guaranteed to be the “best solution” but often “good-enough”

Brelaz heuristic (aka DSATUR) is a list of “rules” for those choices.

Brelaz (DSATUR)

Which node to color first?

Node with the most edges (called maximum degree)

Which node to color next?

Node that has the highest saturation number (connected to the most differently colored nodes)

Which color?

Start at the beginning of the list, choosing the first appropriate color, adding to list if needed.

Summer Results

Benchmark Testing

DIMACS (Center for Discrete Mathematics and Theoretical Computer Science) presents challenges for computer science.

In 1992-1993, they ran a challenge for Combinatorial Optimization, aka Graph Coloring.

Benchmark Testing

DIMACS set up sets of data on which to test different algorithms for graph coloring. Algorithms can then be compared by Chromatic Number (how many colors are required for the graph), energy used, time required.

Benchmark Testing

We used 12 of these DIMACS data sets to test our method, and compared these to results obtained by Parihar, et. al. using Mathematica to generate their results using a built-in function called VertexColoring (which is said to use the Brelaz (DSATUR) heuristic).

Results

Comparison of coloring performed on 2nd DIMACS Challenge graphs

Graph	Nodes	Known Chromatic Number	Minimum number of colors found		
			Brelaz, Parihar et al, using Mathematica	Coupled Oscillator Circuit, Parihar, et al	Brelaz, RET 2018, using Python
huck	74	11	11	12	11
david	87	11	11	13	11
queen5_5	25	5	7	6*	5
queen6_6	36	7	10	12	9
queen 7_7	49	7	12	12	11
queen8_8	64	9	15	14*	12
DSJC125.1	125	--	8	9	6
DSJC125.5	125	--	24	34	22

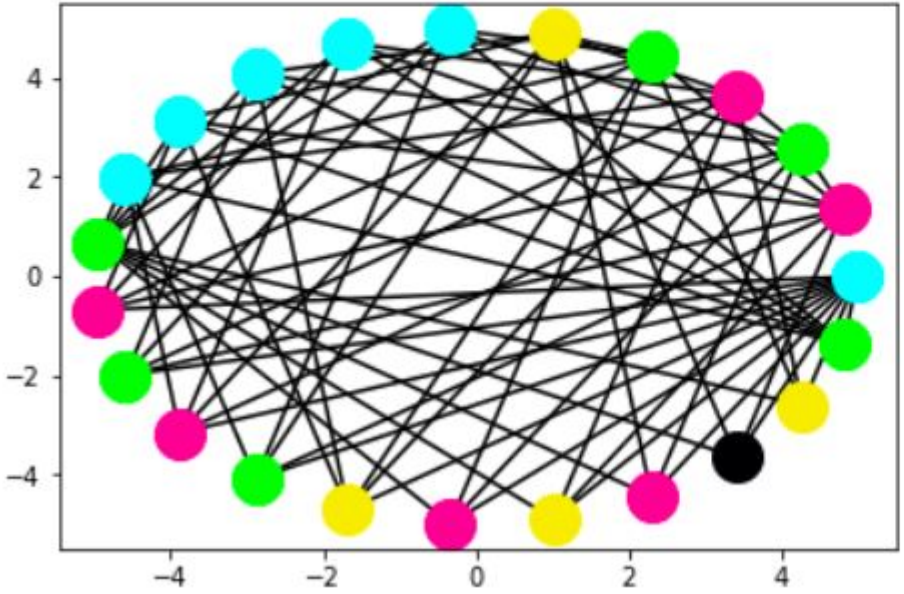
* These are the only two instances when the Coupled Oscillator Circuit beat Brelaz in the paper.

Results, continued

Graph	Nodes	Known Chromatic Number	Minimum number of colors found		
			Brelaz, Parihar et al, using Mathematica	Coupled Oscillator Circuit, Parihar, et al	Brelaz, RET 2018, using Python
myciel3	11	4	4	4	4
myciel4	20	5	5	5	5
myciel5	47	6	6	6	6
myciel6	95	7	7	8	7
tom	107	--	--	--	9

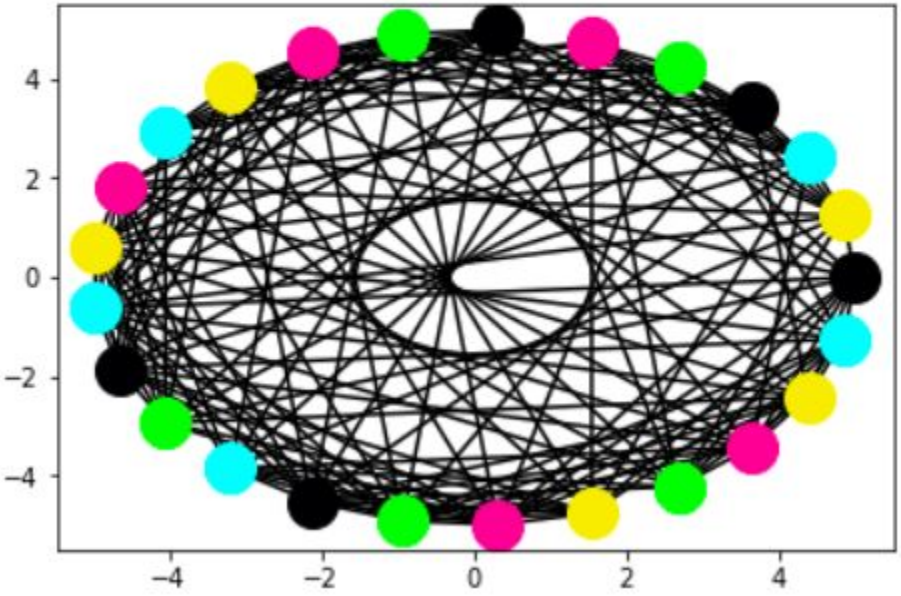
Results

```
<pyclustering.utils.graph.graph object at 0x1168cd198>
/Users/rhumbarger/pyclustering/pyclustering/RET2018/graphs/myciel4.grpr
Chromatic Number:  5
```



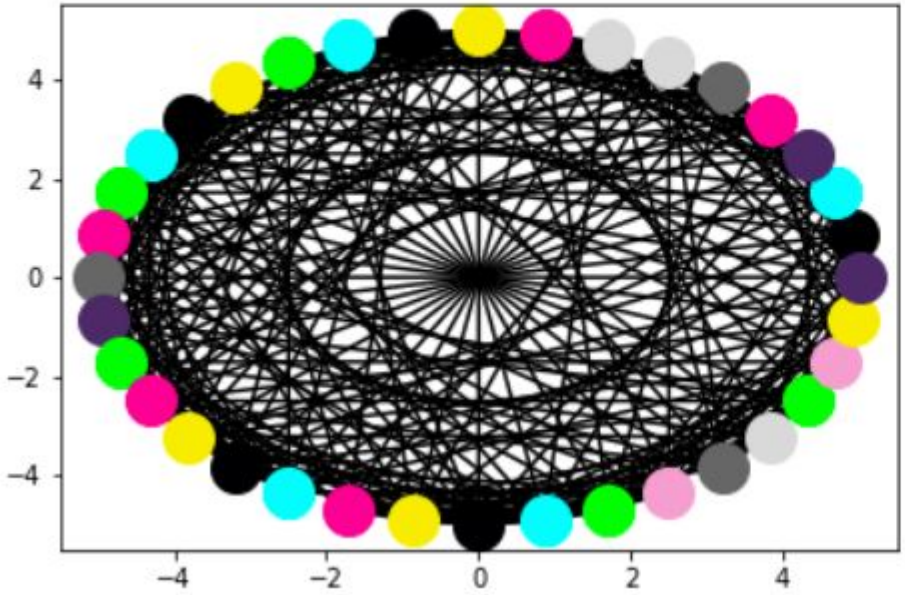
Results

```
<pyclustering.utils.graph.graph object at 0x1153a8518>
/Users/rhumbarger/pyclustering/pyclustering/RET2018/graphs/queen5_5.grpr
Chromatic Number:  5
```



Results

```
<pyclustering.utils.graph.graph object at 0x114ed7f28>
/Users/rhumbarger/pyclustering/pyclustering/RET2018/graphs/queen6_6.grpr
Chromatic Number:  9
```

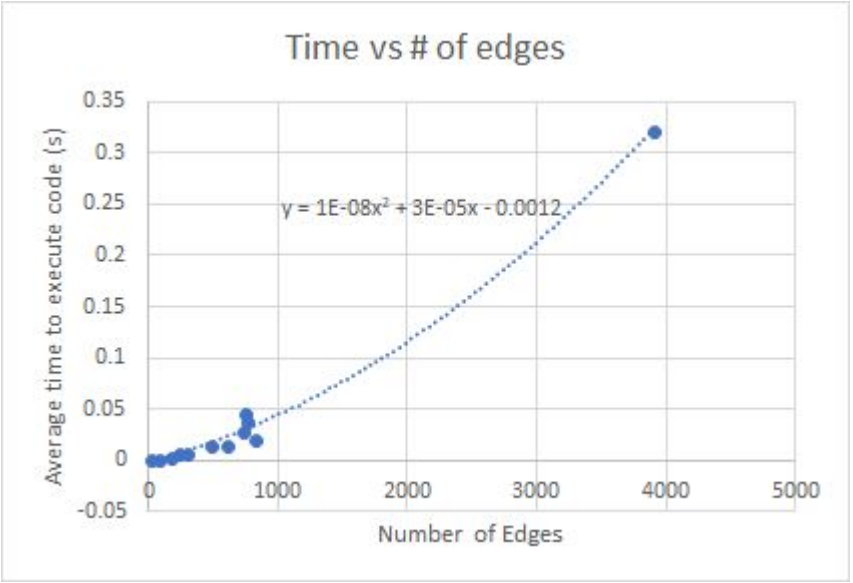


Results - Time Data

The code run using Jupyter Notebooks on a Surface Pro (Intel Core i5 2.6GHz with 8 GB RAM) took on average

37 μs per edge

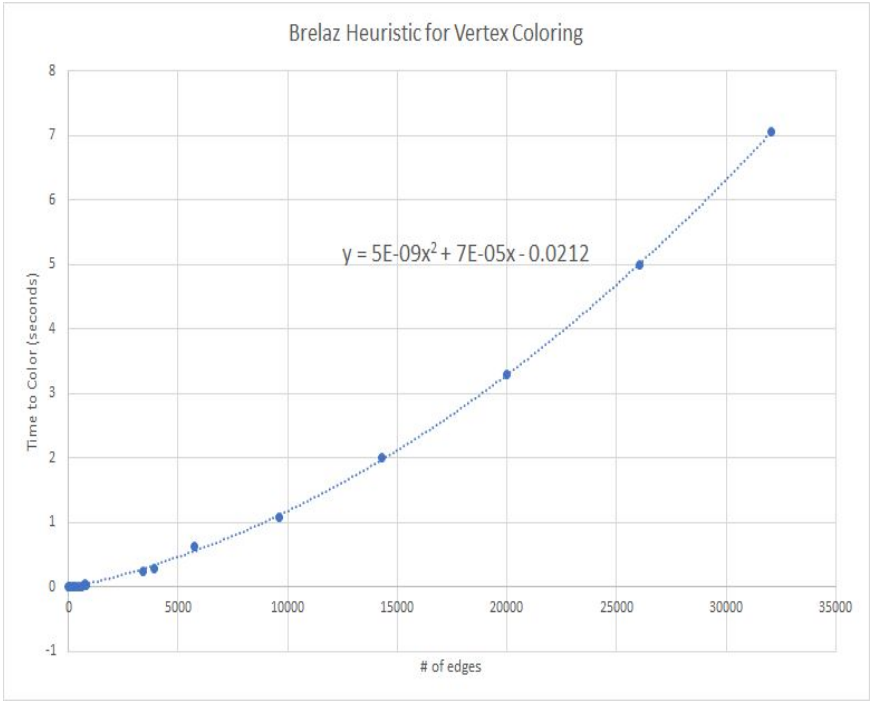
Time to color each graph averaged over 10 times →



Results - Time Data for high # of edges

	Fake Graphs						
Colors	18	22	26	30	36	42	43
Nodes	130	170	220	270	315	360	400
Edges	3382	5723	9597	14300	20010	26046	32070

“Fake” graphs were made using random selection of edges.



Results - Time Data

time.perf_counter() is the clock with the best precision. It is called before and after the process function:

```
graph = read_graph(filename);
dsatur_instance = dsatur(graph.data);
a=time.perf_counter()
dsatur_instance.process();
b=time.perf_counter()
time_elapsed=b-a;
coloring = dsatur_instance.get_colors();
#print("Number colors: ", max(coloring));
# draw_graph(graph, coloring);
```

Example data:

Name	DIMACS.HUCK	DIMACS.MYCIEL3	DIMACS.MYCIEL4
trial1_time	0.011714443	0.000354926	0.001374016
trial2_time	0.01705268	0.000436861	0.002215644
trial3_time	0.01214564	0.00032623	0.001300011
Average	0.014429249	0.000351792	0.001902743
Colors	11	4	5
Nodes	74	11	23
Edges	602	20	71

Time Data - Future Work

The code can be run on a different computer - possibly through Google Colaboratory using the “cloud”.

The code can be parallelized (best candidate is the get_neighbors() function) - check to see if time is improved for the longest-to-color graph (7 seconds).

Moving beyond “toy” problems

The graphs provided for the DIMACS challenge have some meanings:

huck and david refer to Mark Twain/Charles Dickens characters and the queen graphs have to do with chess.

But, we would really like to use this code to solve a real world problem:

Tom Falcone needs to assign 172 students to 100 classes using 29 teachers in 7 periods!

Current status of Tom's scheduling problem

With the current selection of section assignments, the DSATUR algorithm gives 9 colors. The population of each color is: [15, 15, 15, 14, 11, 12, 8, 10, 7]. This means there would be 14 classes First Period ... 6 classes Ninth Period.

Future work:

Figuring out a good way to deal with the classes that have multiple sections... Explore Simulated Annealing to address this issue.

Taking this to the Classroom

Programming, Solving Problems and Physics

Rebecca

- ErgoBot
  - Use programming to solve motion problems
- Algorithms
  - What is an algorithm?
  - Write and test algorithms

Erica

- Graph coloring
  - Do by hand
  - Decide on algorithm
- Python coding
  - Show them code
  - Teach how to modify

Solving Problems

Erica - Class: 8th grade Science

Objectives:

Explore the world, uncovering puzzles to solve.

Develop systematic methods to approach problems.

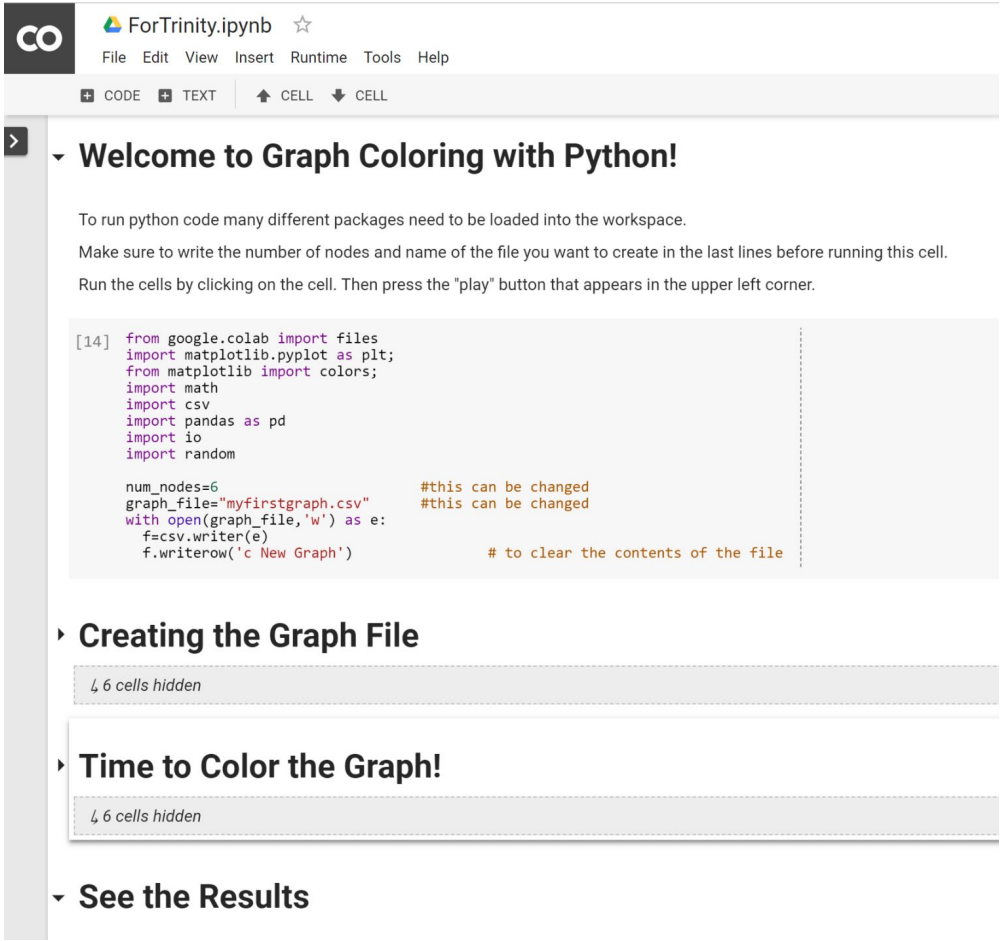
Graph Coloring - A very visual problem to solve.

Python Programming

Erica - Class: 8th grade Science

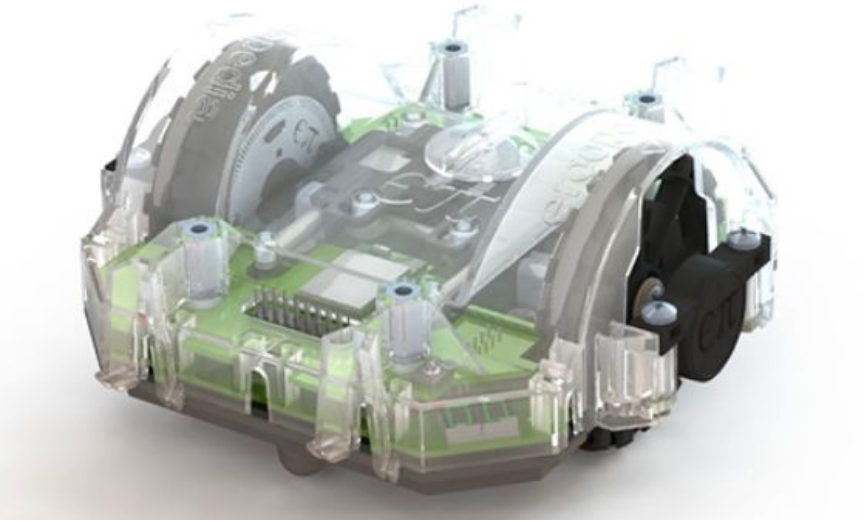
Google Colaboratory

Students access python notebooks (made by me) with text, executable code, and exercises to modify the code and therefore deepen their understanding.



ErgoBot

- Programmable rolling bot that gathers motion data



ErgoBot

- Module Unit starts with Newton's Second Law. Students will gather data, graph data, manipulate values of F, m and a to see the results, and learn to do the calculations themselves
- Students will use vectors to navigate the ErgoBot
- Students will program the ErgoBot using variables and nested loops to navigate a maze

Algorithms

- I will present slides explaining that algorithms take place between computer input and output
- I will challenge students to write algorithms describing several simple tasks that we can test in class, how to draw a cat, how to multiply two numbers
- Students will test their algorithms by having others students follow their algorithms to generate an answer
- Ultimately, students will generate an algorithm to determine the number of protons, neutrons, and electrons in an atom given the atomic number and mass number

Purchases

Becky Humbarger

2 ErgoBots with tripods and tracks

- measures and graphs motion data
- programmable
- lets you manipulate variables

Erica Price

Surface Pro and accessories

- streamlined computer for classroom use
- fast computer for executing code
- digital feedback for student's work